Title: Testing the OLPC School Server

Author: Benjamin Tran

Date: 12/15/2010

Abstract: The One Laptop Per Child (OLPC) project oversees the development, construction, and deployment of an affordable educational laptop (XO) for use in the developing countries. OLPC has been deployed in many countries and as of August 2010, there are over 1.85 million XO laptops in the field. The low-cost XOs help to revolutionize the way we educate the world's children. In the school, each child's XO laptop connects to a central server that provides various networking, content, and communication services. The server also acts as the Internet gateway if an Internet connection is available. The hardware capabilities of the server depend on factors such as cost of hardware and availability of electricity. A more powerful machine means faster access to data and the capability to handle more users simultaneously, but this usually implies higher power consumption, a luxury that is unaffordable in many parts of the world. The software that runs on the server in the classroom, called the School Server (XS), is based on the Fedora distribution of the Linux operating system and provides networking infrastructure, services, as well as education and discovery tools to the XO laptops. Among the stack of customized software included in the XS, one of the most noticeable collaborations tools included is Moodle. Moodle is a free web-based course management system. With the customized software running on limited hardware resources, how many users can reasonably connect to the server and use Moodle? What hardware combination would be ideal for use with the XS given the estimated total number of students in a school? The purpose of this project is to come up with a method to determine the approximate number of users a potential server loaded with the OLPC School Server is capable of supporting. The approach taken will be to perform functional and load testing against different hardware systems loaded with the School Server software. The project makes use of industry-standard technologies and tools such as Selenium, Apache JMeter, and nmon for Linux. Selenium ensures Moodle functions according to specifications while JMeter simulates various load against the server to test its capability. Nmon for Linux monitors server-side resources and presents graphs of CPU usage, free memory, and disk I/O activities among many others. Test results from six different machines are compared. The outcome of this project is a formal process to test and measure the capability of a potential server that is being considered for deployment.

Keywords: OLPC, One Laptop Per Child, XS School Server, XO Laptop, Moodle, Selenium, Apache JMeter, Software Testing, Functional Testing, Performance Testing, Load Testing, Stress Testing

This work is the result of taking course CSC895 Applied Research Project (Computer Science Department, San Francisco State University), with the guidance and assistance from the following advisors:

- Professor Sameer Verma, Department of Information Systems, College of Business, SFSU
- Professor Dragutin Petkovic, Department of Computer Science, SFSU
- Professor Barry Levine, Department of Computer Science, SFSU

Table of Contents

List of Figures

1. Introduction

The One Laptop Per Child (OLPC) project [1] oversees the development, construction and deployment of an affordable educational laptop (XO) for use in the developing countries. Production of the XO started in late 2007 and many countries expressed interest in it. OLPC has been deployed in many countries such as Afghanistan, Argentina, Brazil, Colombia, El Salvador, Haiti, India, Mexico, Mongolia, Nicaragua, Nigeria, Peru, Uruguay, etc. As of August 10, there are over 1.85 million XOs in the field [2].

The low-cost XO laptop helps to revolutionize the way we educate the world's children. OLPC's mission is “to create educational opportunities for the world's poorest children by providing each children with a rugged, low-cost, low-power connected laptop with content and software designed for collaborative, joyful, self-empowered learning [1].” In order to achieve the key goals of cost reduction, lower power consumption, and a smaller footprint, there are many hardware and software limitations to the XO laptop compared to a standard laptop one can easily find in the market today.

In the school, each child's XO laptop connects to a central server that provides various networking, content, and communication services. It also acts as an Internet gateway if an Internet connection is available. The hardware capabilities of the server depend on factors such as cost of hardware and availability of electricity. A more powerful machine means faster access to data and the capability to handle more users simultaneously, but this usually

implies higher power consumption, a luxury that is unaffordable in many parts of the world.

1.1 Problem Statement

The software that runs on the server in the classroom, called the School Server (XS), is based on the Fedora distribution of the Linux operating system and provides networking infrastructure, services, as well as education and discovery tools to the XO laptops [3]. One of the most noticeable collaboration tools included with the School Server is Moodle. Moodle is a free web-based course management system (CMS) that educators use to create effective online learning sites [4]. Teachers can create online dynamic courses in Moodle. Then all the students can log on to access the resources and activities offered in a course. Being a very popular open-source software, Moodle has been deployed in over 200 countries thus far [5].

The School Server comes with a custom Moodle installation. That includes a simpler user interface, decreased number of modules and plug-ins, and preconfigured settings. This customized version of Moodle is tightly integrated with the Browse activity in the XO as well as other features of the XS. A brief description of the XS software stack is provided later on in this paper.

With the various factors limiting hardware resources combined with customizations of the software in the XS, it would benefit the OLPC project if there exists a process to test and measure the capabilities of a potential server that is being considered for deployment. Given a server, how many users can

reasonably connect to it and use Moodle simultaneously? On the other hand, what hardware combination would be ideal for use with the XS given the estimated total number of students in a school? Looking at this problem in a top down approach, it leads to more questions. What method should be used to solve this problem? What tools should be used to implement this method? What hardware systems are reasonable to use for experiments? What are common use cases of Moodle? This project will be the first of its kind since there currently does not exist any openly shared, structured process to solve this problem in the OLPC community.

1.2 Methods and Approach

The purpose of this project is to come up with a method to determine the approximate number of users a potential server loaded with the OLPC School Server is capable of supporting. The approach taken will be to perform functional and load testing against different hardware systems loaded with the School Server software. Functional testing ensures that the software works the way it was intended to. Effective test cases need to be written to cover major and common use cases of the software system. Load testing involves putting variable demands onto a system and measuring its response time. This ensures the system's hardware is capable of supporting an expected number of simultaneous users at peak times. Similarly, exemplary test cases need to be written for this type of testing. The test approach needs to be defined. The review and selection of software testing tools is also of great importance. Additionally, a comprehensive but yet easy-to-use system monitoring tool must

be chosen to monitor the server's resources, such as CPU and memory usage. Taking all of the required tasks into consideration, the following design objectives were finalized (with the advice of the candidate's committee members):

1. To design and write effective test cases to ensure the program's major functionality is within the scope of specifications.
2. To design and write exemplary test cases that can effectively resemble the activities and network traffic that occur when many users simultaneously access the system.
3. To select a quick and efficient approach to carry out these test cases.
4. To choose an effective functional testing framework that allows for easy creation of tests since a web application's user interface is expected to change frequently.
5. To choose an easy-to-use but yet powerful load testing tool that can simulate a heavy load with unique, dynamic data.
6. To select one or more comprehensive system monitoring tool that can graphically show the consumption of a system's resources.
7. To pick out and obtain several computer systems that can represent a broad spectrum of potential servers. The XS software will be loaded on to these systems and testing will be performed against them.
8. To write a thorough test plan containing information such as the testing environment, features to be tested, features not to be tested, test approach, hardware/software setup, etc.

All of the above design objectives that served as requirements for the project were successfully satisfied.

1.3 Solution Overview

Environmental and monetary factors place great limitations on hardware resources for the OLPC School Server. The effects of these limitations are determined by the maximum number of children who will likely send requests to the server over a given period of time. To find out what hardware combination would fit a school with a given total number of students, a test plan was written for performing functional and load tests against a set of representative systems on which the XS software was installed and configured. After the system was confirmed to be functioning according to specifications via automated tests, variable load was placed on the server while its resources were being monitored. The load consisted of requests to perform common user activities such as logging in to Moodle, accessing a course, viewing and replying to threads in forums, as well as taking a quiz at the same time. During the testing, other software packaged in the XS such as the Apache web server and PostgreSQL (Postgres) database also utilized the available hardware resources. In this way, by comparing and analyzing the resource consumptions of a set of representative systems, the hardware requirements can be estimated for a given school size.

1.4 Structure of Report

The rest of this project report is structured as follows. Chapter 2 provides background information about the development and deployments of the OLPC School Server, Moodle, as well as the purpose of software testing, specifically the goals of functional testing and load testing. Chapter 3 contains a short survey of previous work done by the Moodle and OLPC communities relevant to testing. In Chapter 4, some details of the OLPC XS software stack, customizations of these software, and a user guide for installing and configuring the XS are provided to help the reader become a little more familiar with the School Server. Chapter 5 covers some background information on the industry-standard technologies and tools used in this project. Chapter 6 describes the test plan design process, providing details about the significance of feedback from committee members, testing methodology and approach used, as well as the overall setup of the test environment. Implementation details, such as the decisions about the technologies and methods used for testing the OLPC School Server, descriptions of the various test cases and hardware are presented in Chapter 7. Chapter 8 describes the actual testing performed and includes the experimental results. Finally, in Chapter 9, the conclusion of this project and future work are described.

2. Background Information

This chapter gives an overview of the open-source community, development, and deployments of both the OLPC project and the Moodle project in Sections 2.1 and 2.2, respectively. Section 2.3 then goes on to talk briefly about the purpose of software testing and how that is important to software development. This background information will help the reader better understand the motive behind this project.

2.1 OLPC School Server

The School Server software [3] complements the OLPC XO laptop by providing additional infrastructure to extend the laptop's capabilities. The product provides connectivity, shared resources and a range of services. Core features include networking infrastructure, Internet gateway with HTTP proxy and content filtering, support for various wireless network setups, presence and collaboration service, backup and restore service, installation and upgrade services for the XO, and a tailored version of Moodle. More details about Moodle will be provided in later sections. The significance of the XS is growing since the functionality being offered is maturing at a fast pace.

2.1.1 Community and Development

Although being a key element for successful deployments, development of the School Server had been lagging behind the XO laptop until Martin Langhoff, one of the lead developers of Moodle, came in as the School Server Architect in March 2008. The challenge for development has been to balance

server functionality with administrative tools for the XOs and the ease-of-use to manage the server [7]. It became even more difficult when the OLPC project announced in January 2009 to slash its workforce by 50 percent, bringing the total number of people in the project down to 32, and restructuring its operations.

OLPC volunteers and those interested in the development of the XS can communicate with each other via the official mailing list and on an IRC channel [8]. Documentation is shared on wiki pages on the OLPC website. Source code and defect tracking is being maintained with Trac, which is an open-source product that provides an interface to Subversion. Subversion is an open-source version control system developed as a project of the Apache Software Foundation.

2.1.2 Deployments

Mass production of the XO laptops started in November 2007 and it has been deployed in a large number of countries around the world since then [2]. Pilot projects, consisting of a few dozen to a few hundred laptops, usually take place before a country's government decides to order the laptops by the thousands. A Deployment Guide [10] for large-scale deployment reflecting lessons learned from previous pilots and deployments is available on the OLPC website. Also, in 2007 and 2008, a “Give 1 Get 1” program took place; with a donation of $399 to OLPC, donors in the United States and Canada received an XO laptop and another one was sent to a child in a developing country on their behalf.

2.2 Moodle

Moodle, short for Modular Object-Oriented Dynamic Learning Environment, is a free, open-source course management system released to the public in late 2001 [4]. Its focus is on giving educators the best tools to manage and promote learning. Moodle is flexible and scalable; it can be used in a small classroom or in a school with hundreds of thousands of students. Allowing for the creation of dynamic web sites, Moodle makes it possible to fully conduct a course online. Or, it can be used as a course supplement where teachers and students come together to create a rich collaborative community for learning. Resources such as text pages, web pages, and file attachments can be posted to a course in Moodle. Activities such as forums, databases, and quizzes can be created to further enhance learning and discussion.

2.2.1 Community and Development

Just like any open-source project, Moodle is developed by volunteers that are distributed among different geographic regions. It is provided freely under the GNU Public License (version 2 or any later version). Concurrent Versions System (CVS) is the version control system being used to aid the programmers in the development of the source code. At the time of this writing, there are 264 Moodle developers with write access to the code repository [6]. The project uses Atlassian's JIRA product for recording and managing bugs, improvements, and feature requests. More information can be found in Moodle's Developer [6] documentation on its website.

Moodle's community of users contribute to the project in many different ways, such as adding to the documentation on the project web site, discuss about various topics in the forums, adding translations, etc. Moodle conferences are occasionally held at different locations throughout the world. Also, there exists a worldwide group of authorized service companies that are committed to supporting the project financially. They provide a variety of optional commercial services, too.

2.2.2 Deployments

Moodle is used in a variety of organizations. It can be found in schools, government offices, the healthcare industry, military departments, airlines, and oil companies, to name a few. Registration of the product is voluntary but there are an estimated 50,000 registered sites spanning over 211 countries that are using Moodle [5]. Moodle.org itself is the top site with close to one million registered users.

2.2.3 Moodle's Role in the OLPC School Server

A simple Moodle installation was first included in the 0.5 release of the School Server software around November 15, 2008 [42]. Moodle is the selected learning management system for the XS and serves as the configuration frontend. The Moodle installation on the School Server is completely automated and all settings are preset [43]. The first person to ever connect to Moodle will be granted admin access while all the irrelevant, complex settings are hidden. All the users go through Moodle to access their enrolled courses.

2.3 Software Testing

The main purpose of software testing is to ensure that the software system performs according to specifications [10]. The effects of software defects can range from a slight misbehavior to a mission-critical system failing and leading to the loss of human lives. For example, a recent glitch in a major anti-virus software caused trouble in some hospitals' emergencies rooms [11]. The hospitals had to turn away non-trauma patients and canceled some elective surgeries before the error was fixed. The importance of software testing definitely should not be overlooked.

Software testing can be categorized in many ways. The categories include, but are not limited to, black box testing, white box testing, unit testing, integration testing, functional testing, system testing, end-to-end testing, regression testing, acceptance testing, load testing, stress testing, performance testing, usability testing, recovery testing, and security testing [10]. Identifying defects in each of these categories requires a different type of skill set, different techniques, and different types of test cases.

2.3.1 Functional Testing

Functional testing focuses on the functional requirements of a system. It falls within the scope of black box testing where knowledge of how the system works internally is not required. In layman's terms, functional testing makes sure the software does what it is supposed to do and without any side effects. For example, clicking on the 'Save' button for the document in a word processor

should save it to disk but not do anything else such as saving any other document that is opened within the same application.

Functional testing can be performed manually or automated. Manual testing requires a human tester to use the features of the system from an end user's standpoint and ensure correct behaviors. Automated testing is the technique of using software to test software. A test program is written to imitate a user using the system and verify the outcome based on preset conditions. There exist many tools and frameworks to aid in test automation.

2.3.2 Performance/Load/Stress Testing

Many people use the terms performance testing, load testing, and stress testing interchangeably but they have quite different meanings [12]. Unlike functional testing, performance testing does not aim to uncover defects in the system. Instead, it is performed with the hopes of eliminating bottlenecks and establishing a baseline for future regression testing. It is of crucial importance to have a set of expectations for the testing so that the results can be meaningful. For example, in the context of this project, it is necessary to know the expected number of students that will be using the School Server's services at the same time and how long the acceptable response time will be. When the number of students reaches a certain level that causes the response time to be no longer acceptable, then bottlenecks can be looked for at different levels (application, web server, database, operating system, network) and then performance tuning can be performed.

Due to the many layers of complexity involved in a modern day web application, performance tuning has to be done with care. Measurements should be collected after every modification of a variable. Changing multiple variables for a single test session can lead to complications and unexpected results. The complete process should be to run the tests, measure the performance, tune the system and repeat the cycle until the system under test performs at an acceptable level. This process is also known as configuration testing.

The definition of load testing is to execute the largest tasks the system under test can handle and understand the behavior of the system under an expected load [12]. Load testing is usually performed when the maximum number of supported concurrent users for a system is known in advance. Thus, the system is put under such a load to ensure it can still function properly.

The goal of stress testing is to make sure a system fails and then recovers in a graceful manner [12]. This is done by either using up all the resources of the system or taking resources away from the system. For example, stress testing a web application could involve taking the database offline or running unrelated processes that consumes most, if not all, of the resources (CPU, memory, disk, etc.) on the web server.

3. Previous Work

This chapter gives an overview of the relevant work that exists for Moodle and the OLPC XS. This will provide the reader with information about the work that has already been done and further understand the intent of this project. Section 3.1 covers Moodle and Section 3.2 describes the work for the XS.

3.1 Work done on Moodle

Moodle is a mature product with a large user base and active forum discussions. Moodle 2.0, which will be the biggest release ever for the product, is tentatively scheduled to be released in November 2010 [13]. There exists a good number of formal literature about Moodle but most, if not all, of them deal with the administration and usage of the product. Since most institutions go through a phase of testing on their own setup prior to deploying the product, researching in this area should provide valuable and relevant information. However, the amount of information that can be found is limited to those that are publicly shared by these institutions.

3.1.1 Functional Testing

The formal Moodle Testing Group [14] was established in September 2006 and it consists of a small group of volunteers that are responsible for manually verifying bugs that were fixed by the developers. There exists a Quality Assurance page [15] under Moodle Docs but there is a note that says this article is a work in progress. It contains only very basic information of what

quality assurance (QA) is and its objectives. In addition, there is a Testing and QA forum [16] for discussions about recently implemented code, general testing methodologies, context problems, and new features.

Threads can be found in the Moodle forums [16] with discussions about automated testing for the product. There are a variety of threads started by different end users each representing their own institution where Moodle was deployed. The majority of these threads contain questions rather than solutions. Users mentioned tools such as Selenium IDE/RC, TestNG, Borland SilkTest, QA Wizard Pro by Seapine Software, and HP QuickTest Professional [17,18,19,20]. Tests are not shared though and discussions on the topic of functional testing is scattered unfortunately.

In 2008, Moodle was also one of the ten participating organizations in the first Google Highly Open Participation Contest, which aimed to introduce pre-university students to the many contributions that make open-source software development possible. Moodle's grand prize winner submitted some Selenium scripts among other contributions [21].

3.1.2 Performance/Load Testing

There does not seem to exist any official process, tool, or code to carry out performance/load testing for Moodle. There are mentions of various tools [22] in the Moodle forums and some scattered scripts [23] attached to some postings. The Moodle community went as far as setting up an official location in CVS where users can contribute their JMeter test scripts [24]. However, only one small file exists in the repository to date.

There are various docs related to Moodle's performance on the project website [25]. On the 'Development: Performance and scalability' page [26], there are tips on how to improve the performance of the code during development and in production. Developers can turn on display of performance information via a configuration file, use tools such as JMeter to subject the new code to load, make use of a generator for creating random courses and users, as well as using an existing script to parse Postgres logs to output the top 10 slowest queries. On the 'User site capacities' page [27], which has content that needs to be reviewed, there is information on determining the capacity of a Moodle installation. If the maximum number of users that are able to browse the Moodle site and the maximum number of concurrent database users are known, then the general rule of thumb for a single server is: the approximate maximum concurrent users supported = amount of RAM (in GB) * 50 and the approximate maximum number of browsing users is 5 times the approximate maximum concurrent users supported [27].

Finally, last but not least, there is a PHP script called “the Performance perspectives” [23] circulating among the Moodle community that can estimate the performance of a Moodle installation and give the approximate number of maximum concurrent users supported. This is probably the most significant work in this area to date. The script was introduced in the forums in October 2006 and the thread is still alive after three-and-a-half years.

3.1.2.1 iLearn at San Francisco State University

After investigating some potential alternatives for a learning management system (LMS) in Spring 2004, San Francisco State University (SFSU) installed Moodle for alpha testing and beta testing. Moodle was adopted in 2005 with a large-scale deployment test and became the sole LMS in the campus after June 2007. By Fall 2007, there were over 1,000 instructors and 23,500 students using the customized Moodle product called iLearn [28].

The deployment of Moodle at SFSU involved the planning of system architecture, a small development team, a version control system, a defect tracking system, definition of a build process, quality assurance, and tools. Tools used for testing included Selenium IDE/RC, TestNG/JUnit, and JMeter. A set of the JMeter scripts and documentation that was used for performance and load testing was contributed to this project by Dr. Wen Hao Chuang, who is a full-time staff at Academic Technology at SFSU and also one of the Moodle core developers with write access to the Moodle CVS tree [29].

3.1.2.2 Moodle 1.7 Quiz Handles 300 Simultaneous Users on One Server

In April 2007, Sapporo Gakuin University in Hokkaido, Japan gave an English Placement Test to its entire incoming first year students. The quiz consisted of 50 questions and was administered in 5 sessions with up to 300 students per session. This was an excellent case study of a Moodle installation and its performance under load. Detailed information of the entire hardware and software setup along with monitored results of the server were posted to

the Moodle forum [30]. The results suggested that the system used could potentially handle over 1,000 simultaneous users with audio files.

3.2 Work done on OLPC XS

In contrast to Moodle, the OLPC XS is still fairly new, with version 0.6 being its latest release in October 2009. Thus, there exists very little, if any, previous work in the testing area. On the OLPC wiki site, there is a page with instructions for validating the proper operation of the XS School Server software [31]. The list includes manual checks to test for internet connection, basic mesh connectivity, domain name system, IP routing, large file transfer, web caching, laptop registration, and collaboration services. Thus, this project can be viewed as a starting point for what will become the previous work of future relevant projects.

4. System Setup

This chapter gives a technical overview of the OLPC School Server to help the reader in getting familiar with the server setup. Section 4.1 briefly goes over the set of programs bundled in the School Server. Some of the customizations that were made to these software are described in Section 4.2. Finally, Section 4.3 provides instructions on setting up and configuring the XS for this project.

4.1 OLPC XS Software Stack

A software stack is a group of programs that work together in a system to produce a result. The OLPC XS [3] is based on the Fedora distribution of the Linux operating system. On top of that, a few major programs work together to provide a range of services to the school. That includes the Apache HTTP server, PostgreSQL database management system (DBMS), and the PHP: Hypertext Preprocessor engine. The OLPC School Server is not based on the LAMP (Linux, Apache, MySQL, Perl/PHP/Python) stack because the PostgreSQL DBMS handles recovery better than MySQL. This is important because OLPC deployments are usually in areas that lack or have an unstable supply of electricity. The XS provides collaboration and presence services via ejabberd, which is a Jabber/XMPP instant messaging server licensed under the GNU Public License v2. Activities such as Chat, Write, and Memorize use XMPP to collaborate [41]. As mentioned before, the XS is also packaged with the Moodle CMS. The presence service in Moodle uses ejabberd to control and show only fellow members of courses a certain user belongs to.

4.2 Customizations of the XS

Not surprisingly, the OLPC School Server software [3] has its own customizations. There are many reasons for this. The software needs to be customized to meet the needs of schools where it will be deployed. In addition, the XS was designed to run on low-end generic servers with limited hardware resources. That also requires customizations of the software for reasons such as to run less default processes or to run smoothly on a processor of limited speed. Starting at the bottom of the software stack such as the operating system kernel, every software in the package was configured, tweaked, and modified. A PHP accelerator was added to speed up the performance of scripts written in PHP. Apache was configured to point to Moodle by default. There does not exist a lot of information on the customizations of the XS and further discussion in this topic is out of the scope of this project.

4.3 Setup Guide

This section provides detailed instructions on setting up and configuring the School Server to prepare for test execution. Normally, the first XO laptop that successfully logs in to Moodle will be granted the “course creator” role with administration options and additional user accounts can be created afterward. Since test execution will be on a non-XO computer, it is necessary to set up the network connection and then gain access to Moodle.

4.3.1 Downloading and Installing

A disc image (ISO file) of the OLPC School Server [3] can be downloaded from the Internet. An ISO file contains an uncompressed collection of various files. It can be burned to an optical disc or opened using archival applications. The following are the steps for downloading and installing the School Server.

Open any web browser such as Google Chrome and type in this URL (http://xs-dev.laptop.org/xs). The browser loads a web page similar to what is shown below in Figure 4.a. Right-click on the file named OLPC_XS_LATEST.iso and save it to the hard disk. This is the disc image file for the latest version of the School Server.

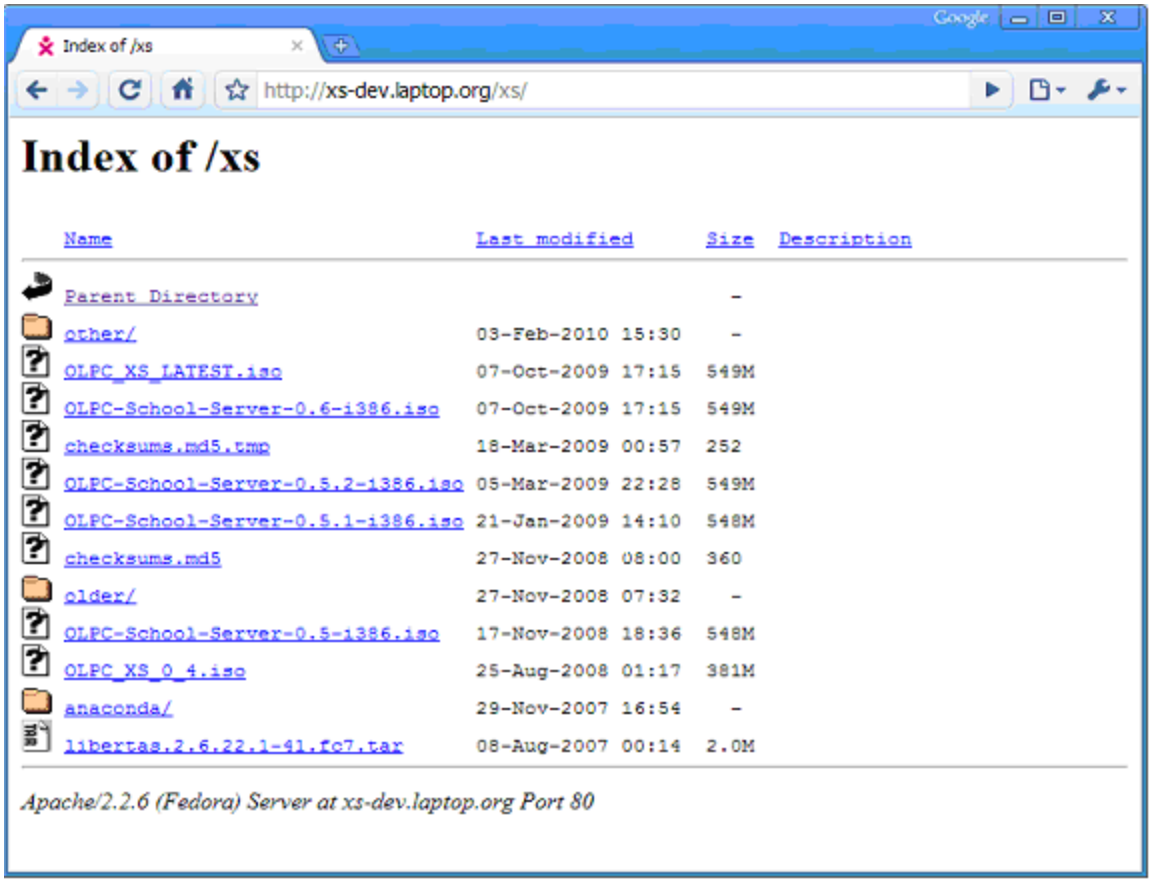


Figure 4.a Web page to download the OLPC School Server

The machine under test on which the server will be installed needs to boot up with the downloaded ISO file. Burn the ISO file to a CD/DVD or copy it to a USB flash drive. Then load the CD/DVD or plug the USB flash drive into the machine under test.

Figure 4.b Installation Menu for OLPC School Server

Figure 4.b shows what the screen looks like after the system boots up with the disc image. Proceed with the installation just like with any other software. Going forward, the machine under test will be simply referred to as the School Server.

4.3.2 Configurations

The School Server [3] needs to be configured after it is installed. This section provides information and instructions on the configuration of the School Server.

4.3.2.1 Setting the Server Domain Name

A domain name is a text name that identifies a web site. It corresponds to the numeric IP address of the computer hosting the web site. The domain name for the School Server needs to be set after the server is installed. Note that the domain name does not need to be real (discoverable from the root Domain Name System servers on the Internet) unless the presence service is allowed to be accessed from outside the school. There is a script called *domain_config* that can be used to change the domain name and it is located in the */etc/sysconfig/olpc*-scripts/ folder [32]. The script takes the new domain name as its argument on the command-line. After running the script to set the domain name, reboot the machine so the change can take effect. Finally, confirm the change as shown in Figure 4.c on the next page.

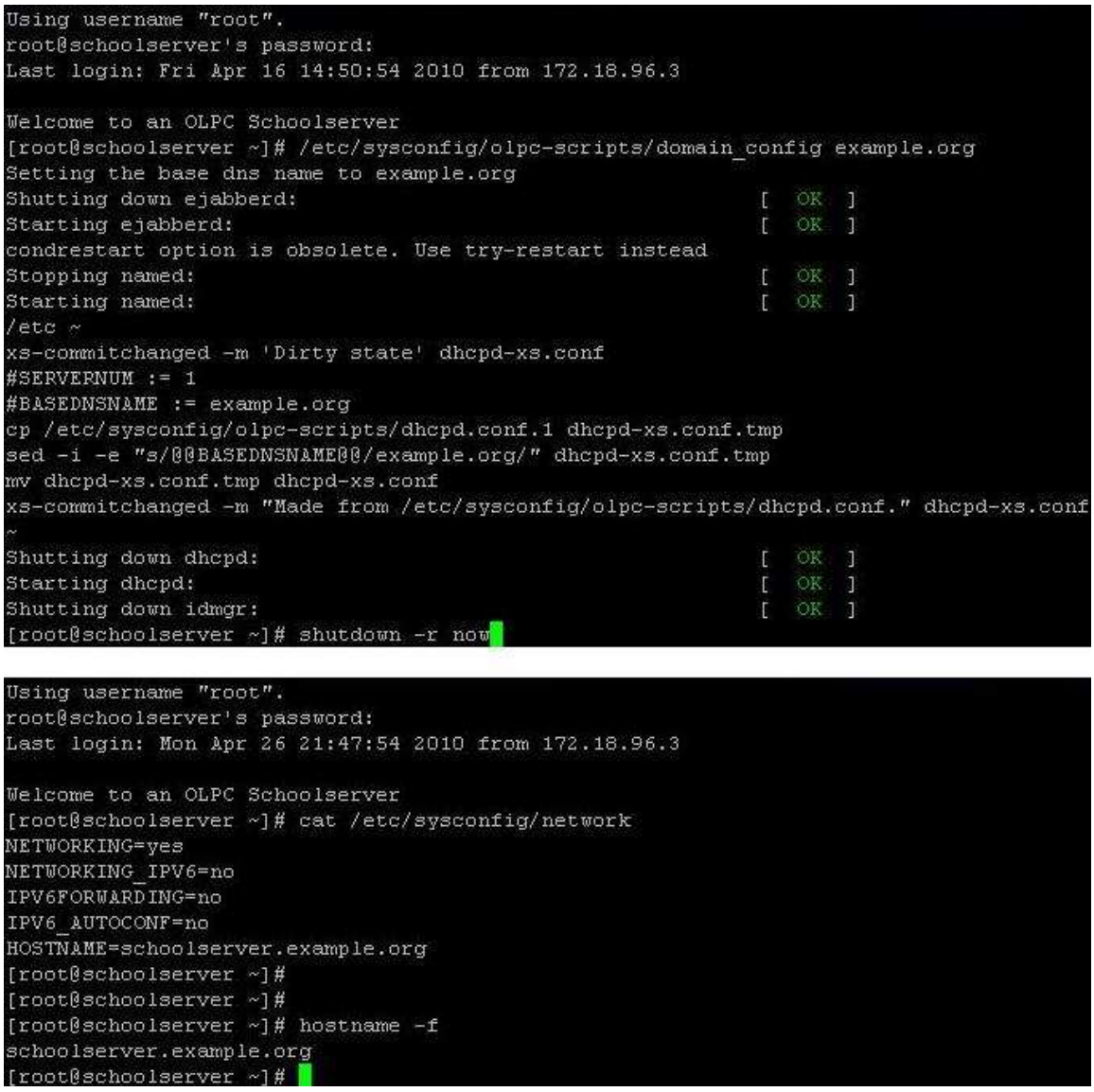


Figure 4.c Running Script to Set Server Domain Name

4.3.2.2 Setting up the Network Connection

The network needs to be set up so the machine running the tests can access the School Server. Among other network services, the School Server provides DHCP (Dynamic Host Configuration Protocol) service to assign network addresses to devices connected to the network. Connect the wired Ethernet network interface from the School Server to a network switch, which is a

computer networking device used to connect network segments together. Also, connect the machine that will run the tests to the same network switch. Figure 4.d shows what the network setup should be like [32]. The network setup is complete if the machine that will be executing tests can access or *ping* the School Server. Open any web browser and try to access this URL (http://schoolserver/), or execute the *ping* command against the School Server from a console. Ping is a tool for testing if a particular machine on the network is reachable. It sends packets to the destination computer and waits for a reply, at the same time measuring the round-trip time for the packets.

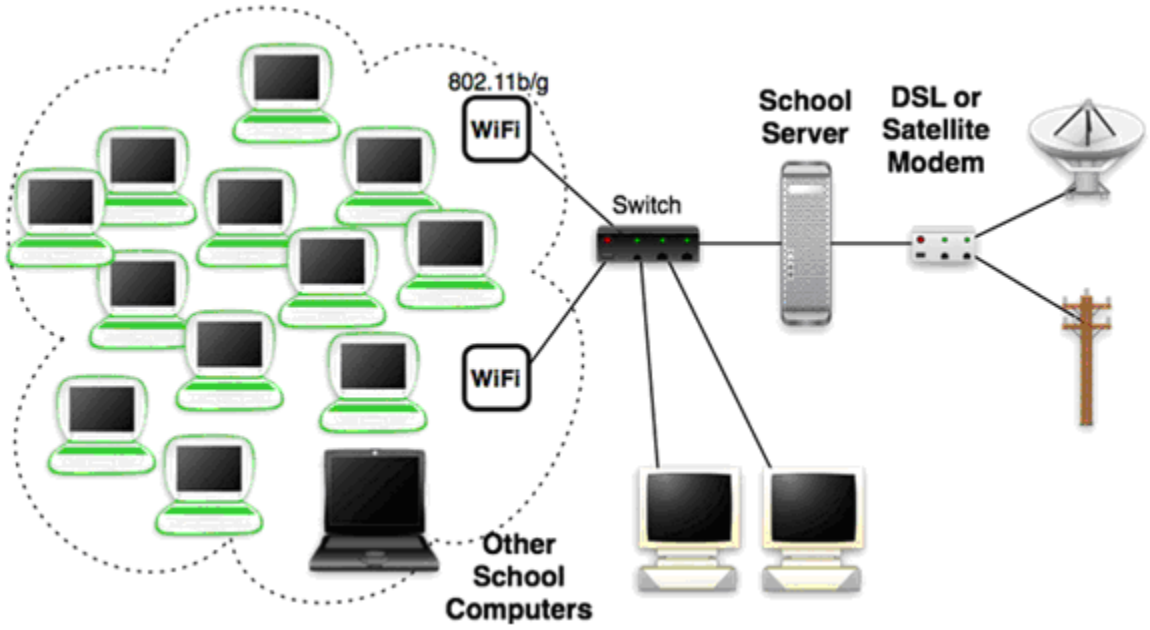


Figure 4.d Network Setup

Note that the default server setup is to have the first wired Ethernet network interface connect to the Internet, if that is available to the school. A second Ethernet network interface is configured to provide an internal local area network (LAN) within the school. If there is only one wired Ethernet network

interface, the *xs-swapnics* script can be used to change the aforementioned configuration. Alternatively, a single crossover cable can be used to connect the two computers together and eliminate the switch altogether. This latter solution eliminates a latency factor (the time it takes for data to go through the switch) and removes a potential fault point (the switch itself).

4.3.2.3 Enabling SSH

Secure Shell (SSH) is a network protocol that uses a secure channel to exchange data between two networked devices. It would be convenient to access the School Server via SSH from any computer on the network. Root access via SSH is disabled by default so it is necessary to change that setting. Please note that enabling root SSH is not recommended. It is enabled here only for testing purposes.

Edit the *sshd_config* file located in the */etc/ssh/* directory. Set the value of *PermitLoginRoot* to *Yes* as shown in Figure 4.e on the next page. Also, comment out the second "PasswordAuthentication no" line in the file. Finally, reload SSH via the following command and it should be possible to connect to the School Server from any computer on the network: */etc/init.d/sshd reload*.

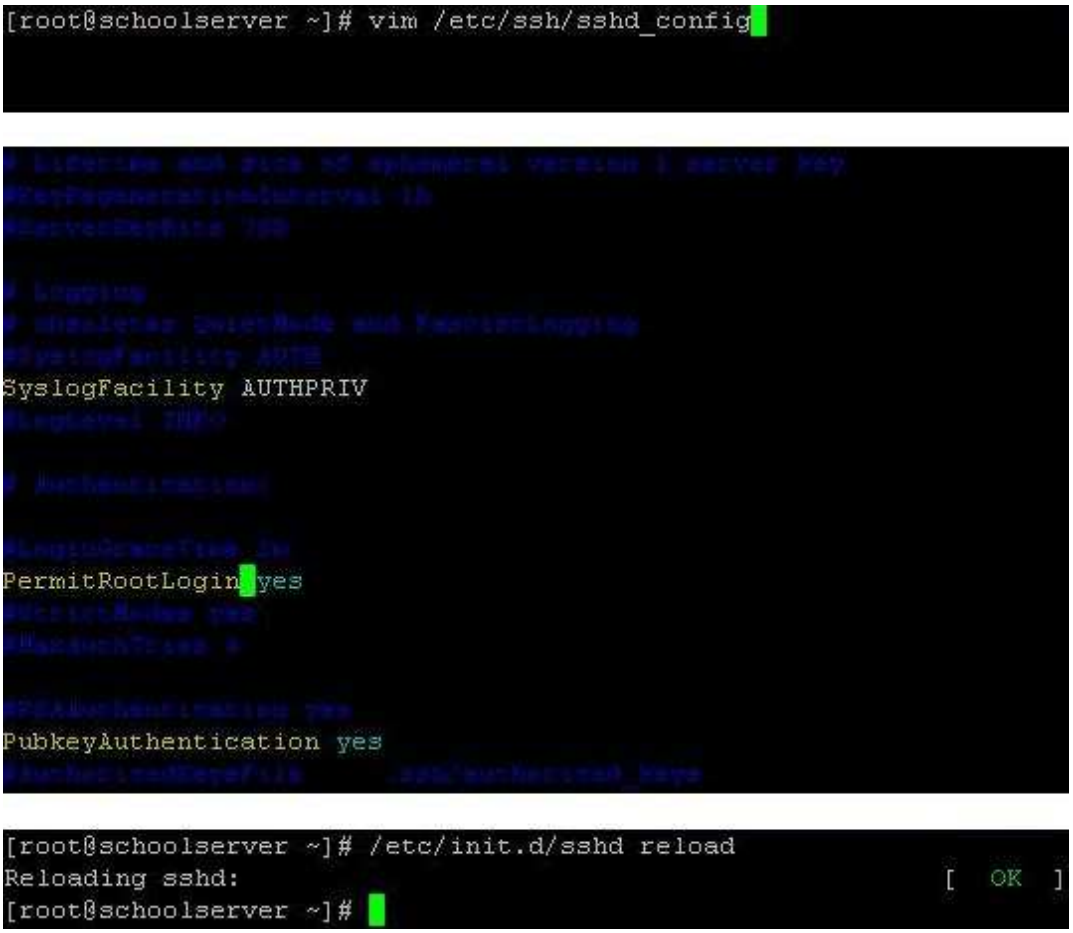


Figure 4.e Enabling SSH

4.3.2.4 Enabling the Moodle Admin Account

As previously mentioned, the first XO to successfully log in to Moodle is granted the "course creator" role with certain administration options. Since that case does not apply in this project, it is necessary to use the special admin account to gain access to Moodle. By default, this admin account is disabled [33]. It can be enabled by running a PHP script. See Figure 4.f for the command. The password for the account is stored in a file at */etc/moodle/adminpw* and this value is different for each installation of the School Server.

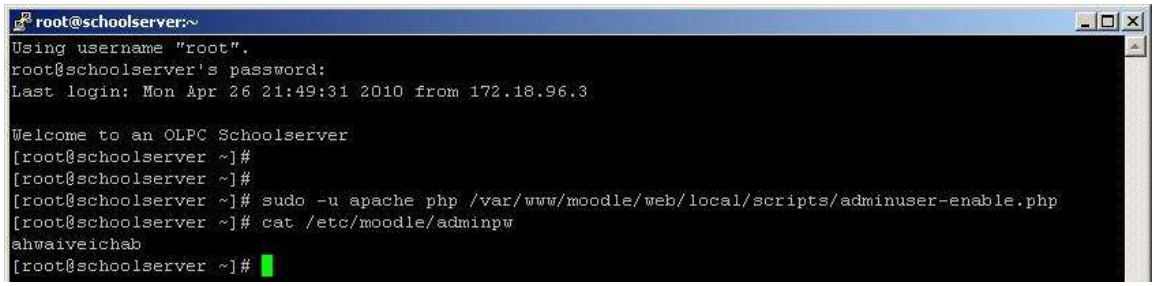


Figure 4.f Enabling Admin Account and Obtaining Password
sudo -u apache php /var/www/moodle/web/local/scripts/adminuser-enable.php

After enabling the admin account and obtaining the password, open any web browser and try to access the Moodle homepage (http://schoolserver/). It should be similar to what is shown in Figure 4.g.

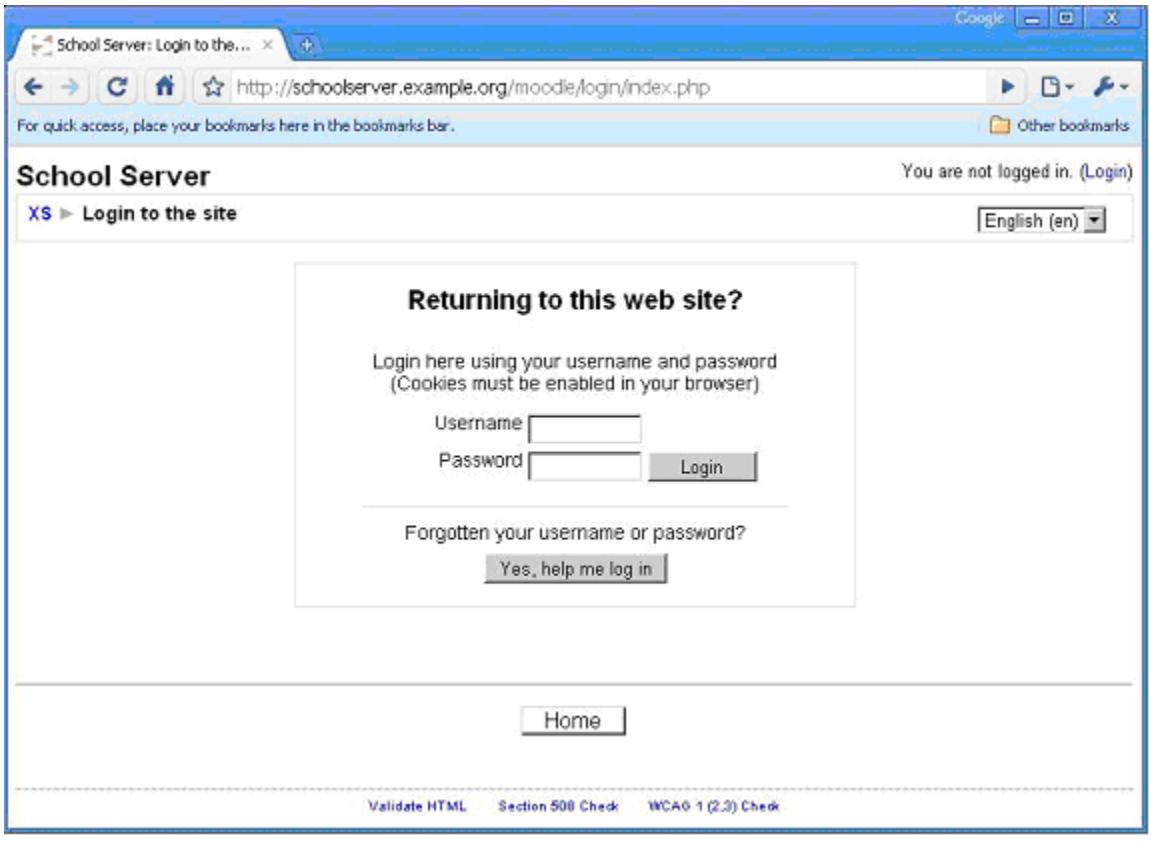


Figure 4.g Moodle Homepage

4.3.2.5 Changing Moodle's confirm_sesskey( ) Function

For the purpose of this project, it is necessary to modify Moodle's *confirm_sesskey( )* function to always return true [34]. The function is located in the /var/www/moodle/web/lib/moodlelib.php file. If this change is not made, an error message will be encountered when automated scripts try to submit information to Moodle but fail to include a valid session key. The error message is similar to “Incorrect sesskey submitted, form not accepted!”

5. Background Implementation Technologies

This chapter provides some information on the technologies and tools used in the project so the reader can better understand the processes and techniques implemented hereafter. Section 5.1 gives a description of Selenium, which is the testing framework used in the project. Likewise, Section 5.2 provides information about the performance testing tool called Apache JMeter. Finally, Section 5.3 describes the system monitoring tools used to monitor the School Server's resources during testing.

5.1 Selenium

Selenium is an open-source portable software testing framework for web applications [35]. In other words, it is a suite of tools that can be used to automate web application testing across many platforms and browsers. There are three variants of Selenium: Selenium IDE, Selenium Core, and Selenium Remote Control. They can be used independently or together to create a powerful suite of automated tests.

5.1.1 Selenium IDE

Selenium IDE is a complete integrated development environment (IDE) that is implemented as an extension of the Firefox web browser. The user interface is shown in Figure 5.a on the next page. Formerly known as Selenium Recorder, it allows the user to record, edit, playback, and debug Selenium tests. This is the easiest way to use Selenium. The user does not need to know any programming language and will still be able to easily record a test case.

The recorded tests can be exported to a variety of programming languages such as Java, C#, Perl, PHP, Python, and Ruby. The main drawback of Selenium IDE is that it is only available in the Firefox browser. However, the exported test scripts can be executed on other operating systems and browsers if Selenium Remote Control (RC) is used.

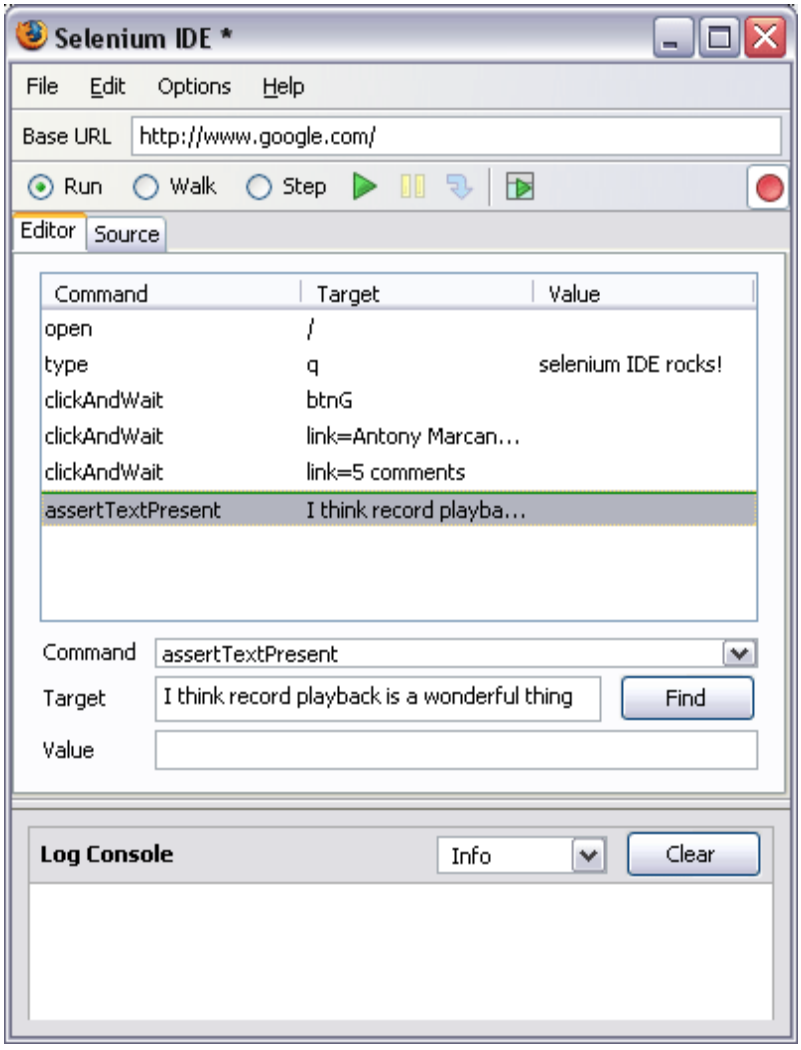


Figure 5.a Selenium IDE User Interface

5.1.2 Selenium Remote Control

Selenium RC is much more powerful than Selenium IDE. Selenium RC allows the user to write automated tests in any programming language against any HTTP website using any mainstream JavaScript-enabled browser. There are two parts to Selenium RC. The first part is the Java-based Selenium server that launches and kills the browser. The second part is the client that sends test commands to the server. Figure 5.b shows a simplified architectural representation. Selenium comes with client drivers for a variety of programming languages such as Java, Dot Net, Perl, PHP, Python, and Ruby. Tests can be written in any of these languages and then executed against the Selenium server. This includes the tests that are exported from Selenium IDE.

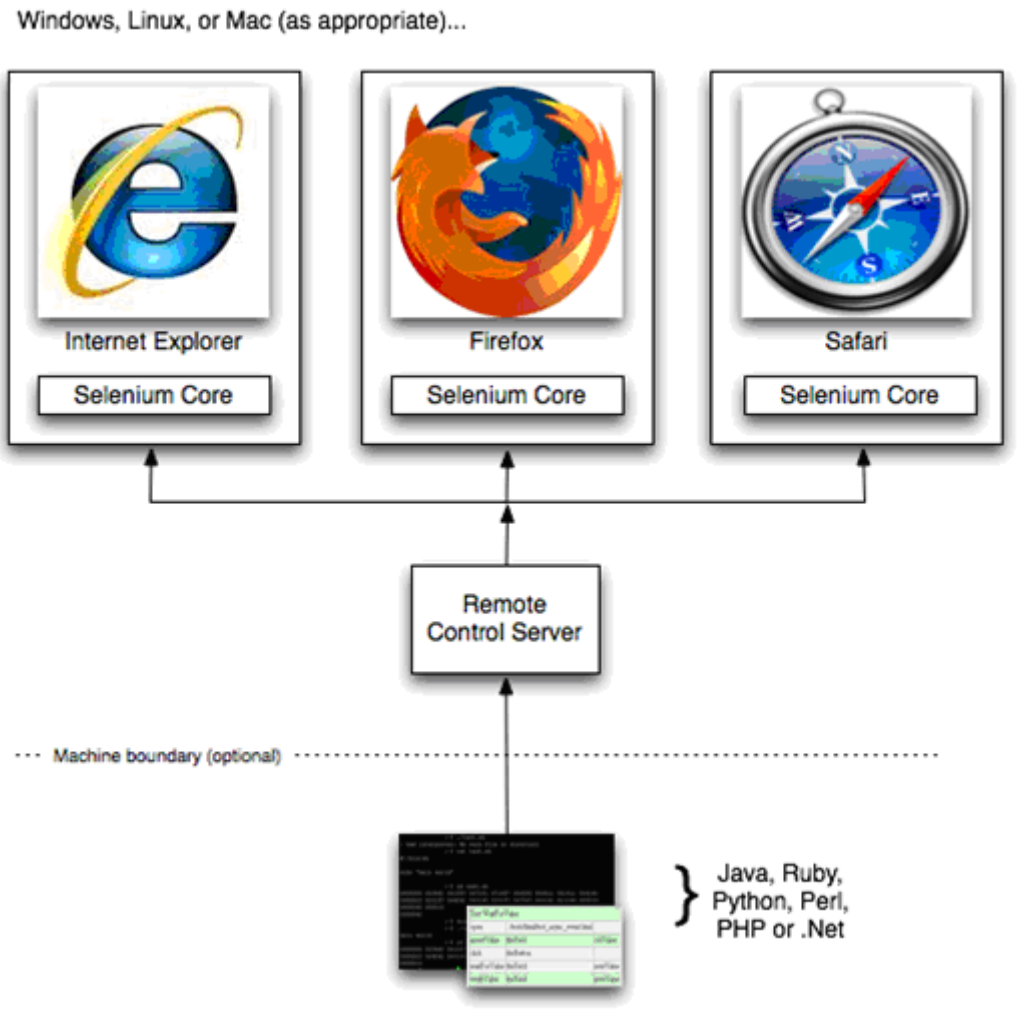


Figure 5.b Selenium RC Architectural Representation

5.1.3 Advantages of Selenium

Although there are many tools in the market to help with automating functional tests, Selenium has its advantages. First, Selenium is open-source and free to use. Literally anyone can use Selenium IDE to record a test case and export it to different programming languages just by clicking the mouse alone. The Selenium RC server is simply a Java *.jar* file that does not require any installation. It can be started on any platform that is Java-enabled, as shown in

Figure 5.c. Selenium comes with client drivers for many popular programming languages, which makes it very easy to get started with writing tests and integrate with any existing tools or frameworks. Being able to run one test script against many browsers is a major advantage, although that does not apply to the case in this project since the XO laptop only has one browser built in.

Figure 5.c Starting Selenium RC Server

5.2 Apache JMeter

Apache JMeter is an open-source software product that is designed to load test web applications and measure the performance [36]. It is Java-based and can be extended through a provided application programming interface (API). JMeter can be used to simulate high traffic against a web application or server and provides graphical tools for analyzing overall server performance.

5.2.1 HTTP Proxy Server

JMeter provides a component called the HTTP Proxy Server that can be used to record all the HTTP requests made while the user is browsing a web application. It can be set to filter out certain requests such as those for images

and CSS files that are not relevant to the test case. Similar to Selenium IDE, the HTTP Proxy Server makes it very easy for the user to start creating a test script from scratch.

5.2.2 Advantages of JMeter

JMeter is a highly popular load testing tool used in the testing field. It was built specifically for the purpose of running load and performance test against web sites. Like the Selenium RC server, the software is Java-based and can run on any platform that is Java-enabled. Unlike Selenium though, JMeter does not control any browser but instead, it emulates the requests and responses that would travel between the browser and the destination server.

Commercial tools such as LoadRunner, Silk Performer, and Rational Performance Tester carry a high price tag. Aside from JMeter, there exists other open-source tools such as WebLoad, OpenSTA, and the Grinder. Comparison factors include accuracy, cost, and features. Being a product that is free to use, having a simple user interface, and the small amount of time needed to start creating tests from scratch are major advantages of JMeter. Figure 5.d shows what JMeter looks like when it is initially started. The drawback is the lack of support for real time monitoring of server side metrics such as CPU and memory usage. This leads us to the next section, which covers some useful system monitoring tools that can be used to monitor the server under test.

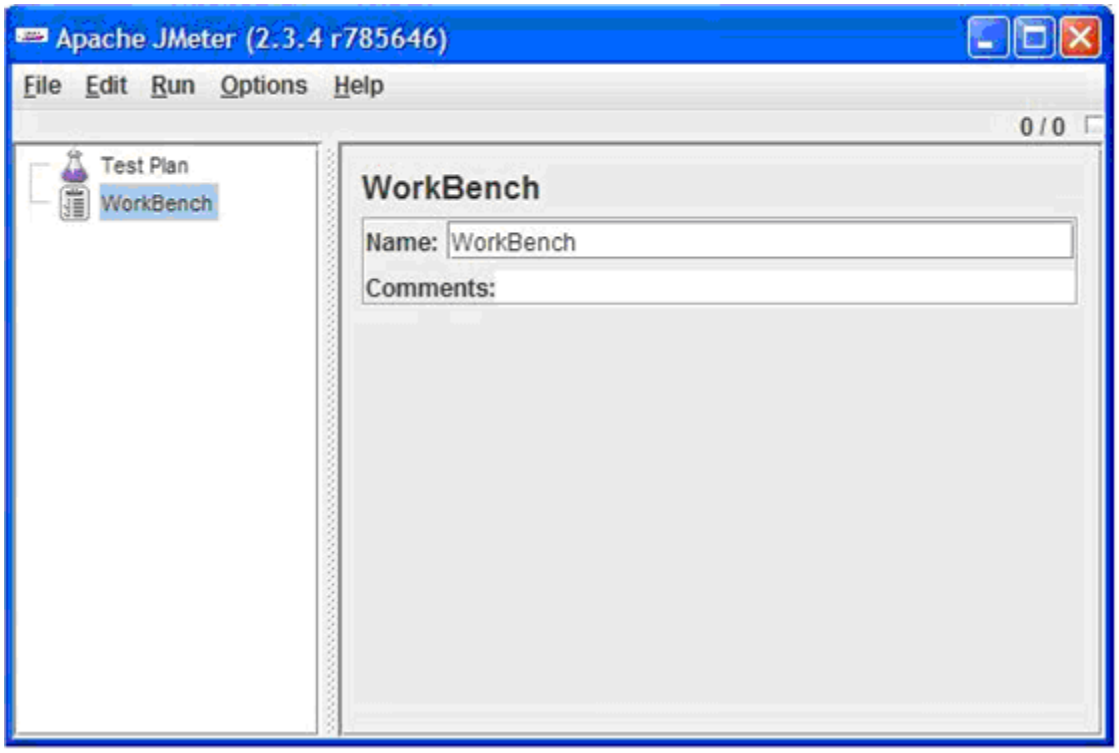


Figure 5.d Apache JMeter User Interface

5.3 System Monitoring Tools

Most performance testing tools support monitoring of client side metrics such as the response time and hit rate. However, support for real time monitoring of server side metrics such as CPU and memory usage are lacking in most open-source tools. Thus, it is necessary to utilize some popular built-in Linux commands and other monitoring tools on the server side to get this metric during the time the testing occurs.

5.3.1 Built-In Linux Commands

Most Linux distributions come with a lot of built-in monitoring tools and commands [37]. These tools can be used to provide metrics, find information about system activities, or find the possible causes of a performance problem.

The following are a few of the Linux system monitoring tools that are useful for this project:

- **top**: The top command shows a list of the actual processes that are running on the server. It displays the tasks that are CPU-intensive and updates the list every five seconds.
- **vmstat**: This command shows information about system activity, hardware information, and software information such as processes, memory, paging, and CPU activity.
- **iostat**: The iostat command shows statistics about the CPU and input/output statistics for devices, partitions and network file systems.

5.3.2 nmon for Linux

Nmon for Linux [38] is an open-source monitoring tool that consists of just one single executable file. It makes it easy to monitor the performance of computers and networks. No installation is necessary. The user just have to obtain the executable file fit for the underlying operating system and platform. Nmon provides information on the system's CPU, memory, network, disks, file systems, top processes, and more. Data can be displayed on the screen or output to a comma-separated (CSV) file for analysis. A separate tool called the nmon Analyser can be used to load the CSV file and automatically create dozens of graphs from the output data.

6. Test Design

This chapter gives details about the test design process used in developing the tests. In section 6.1, the significance of feedback from the committee members of this project is explained. Section 6.2 explains the test design process, which involves writing a test plan and its various sections.

6.1 Feedback from Committee Members

Committee members were involved throughout the test design process and contributed valuable feedback to the project. Professor Petkovic provided comments on both the organizational structure of the test plan and this paper. With his deep knowledge of the OLPC project, Professor Verma provided great guidance from beginning to end. All the important decisions during the test design process and implementation phase were made based on committee members' advice. The timely feedback assisted us in designing and carrying out the tests.

6.2 Test Plan

Prior to creating and executing tests, a test plan should be written, reviewed and approved. A test plan is a detailed document describing the objectives, scope, strategy, approach, focus, resources and schedule of a testing effort [10]. Formally, it serves as a contract between the quality assurance team and the rest of the project stakeholders such as management and development. The descriptions of the various parts that make up a test plan are as follows.

6.2.1 Introduction

The first part of a test plan should be a brief introduction of the software or system that will be tested, why there is a need for the testing to take place, the objectives, as well as the scope of the testing effort. The introduction should provide enough background information for someone who does not have any knowledge of the software or system under test to understand the rest of the test plan. It can also include links to other documents so the reader can go on to learn more about the software or system to be tested.

6.2.2 Testing Environment

This section of the test plan should list and describe all the hardware and software needed to perform the testing. It should act as a high level checkpoint for gathering all the necessary equipment and tools to prepare for test execution. Physical characteristics, communications, levels of security, and software testing tools should all be included when and if applicable. The hardware and software requirements for testing activities, including the versions of the products, should be specified. If applicable, the security and asset protection requirements for testing activities should be identified. The software tools, techniques, and methodologies to be used in the testing efforts should also be identified and have their purposes described. Any other documentation required to support the testing activities should be listed.

6.2.3 Features to be Tested

This section identifies all the features of the software or system to be tested. The design specifications of each of these feature should be identified and described here so the person executing the test plan knows what to expect.

6.2.4 Features Not to be Tested

It is equally important to explicitly specify the features that will not be tested in the software or system under test. If possible, specify the reason(s) to justify why a particular feature will not be tested. It could be due to the schedule, available resources, or the feature being out of the scope of the testing effort. It is extremely important to communicate precisely since the other project stakeholders might have expectations that are never explicitly mentioned or stated.

6.2.5 Test Approach

This section is the core of the test plan. It specifies the approach to be used for testing, whether it will be manual or automated, for example. A test strategy is devised here that focuses on the following points:

- What kind of testing will be done? What will be focused on and what will only be tested on the surface?
- Can the software or system be categorized by component or by level of testing?
- Can product requirements be mapped to testing areas?

- From whose viewpoint will testing be executed from? (naïve user, expert user, internal user, etc.)

With the test approach and strategy defined, there should be a list of detailed steps for setting up hardware and software to prepare for test execution. There should be a list of test cases that will be covered during testing. Each test case can have its own ID number for easy identification. Also, each test case can include a name, objective or summary, steps to be taken, expected results or what needs to be verified. Oftentimes, this list of test cases can reside in a separate document and linked to from the test plan.

6.2.6 Test Deliverables

This section identifies the deliverables from the test process and this includes release notes, defect logs, metric reports, and test summary reports. In other words, this section states how information is to be communicated between the quality assurance team and the other project stakeholders before, during, and after the testing effort.

6.2.7 Schedule and Staffing

A general timeline of the testing cycle should be provided. For example, after a new version of the software or system is available for testing, specify how long it will take to perform certain testing tasks. A list of the people involved in the project should be listed in case the reader of this document needs to contact someone with a question. This list should include people such

as the product manager, technical lead, frontend developers, backend developers, user interface designer, and testers.

6.2.8 Risks, Contingencies, and Dependencies

This section lists anything that can put the successful outcome of this test plan at risk, such as the release candidate not being available for testing, the test environment is incomplete, or there is a need to gain access to a certain resource. This list serves to help everyone involved to have a better understanding of what could become obstacles and hinder the progress of the testing effort.

6.2.9 Reference Materials

Finally, all the documents and sources referenced in the test plan should be identified and listed here. Relevant documents could be product requirements document (PRD), design documentations, and user interface mock-ups.

This chapter explained the test design process, describing the various sections of a test plan. Feedback from the committee members is another factor that greatly contributes to test design and should not be neglected. After feedback was received, the test plan was written, reviewed, revised, and approved. That is described in the next chapter.

7. Test Plan for the OLPC School Server

This chapter provides information about the actual test plan for this project, which can be found in Appendix A of this paper. The test plan follows the format introduced in the previous chapter. Section 7.1 describes the selected testing approach, various decisions made and the reasoning behind those decisions. In Section 7.2, all the test cases are explained in detail, including their purposes and importance to the testing effort.

7.1 Test Approach Decisions

After reviewing the objectives and scope of this project, the committee members and we agreed on having functional and performance/load tests designed, written, and executed. All the testing will be automated so the test plan can be quickly and easily implemented across different hardware systems. Test automation saves time by not having the need for a tester to manually perform each and every test case for the different machines under test. The testing can also be easily extended to more machines with the School Server loaded should that be necessary in the future.

7.1.1 Decisions regarding Functional Testing

Based on the information in Chapter 5, Selenium was chosen for automating the functional tests in this project. Selenium IDE allows for quick creation of tests by recording the user's actions in the Firefox browser. Then the tests can be exported to one of the supported major programming language and executed with Selenium RC on any operating system and browser

combination. In the future, another tester can easily load the saved test cases in Selenium IDE, export it to a language of his or her choice, and expand the tests from there.

Chapter 4 provided instructions on enabling the special admin account in Moodle. With access to that account, a teacher account and a student account can be easily created for testing purposes. The credentials of these accounts need to be entered into the Selenium test cases prior to their execution.

7.1.2 Decisions regarding Performance/Load Testing

JMeter was selected for automating the load and performance testing in this project due to its popularity and large user base in the testing community. Similar to Selenium, JMeter offers a HTTP Proxy Recorder component to allow for quick creation of tests by recording all the requests made in a browser. Unwanted requests can be easily filtered or deleted. Shared settings such as server and port address can be extracted to a central setup location in the test script. The tool allows for testing with dynamic data; a different record from a comma-separated file can be used for each thread in a test. For example, it is possible to simulate fifty different users logging in to a system instead of reusing the same account over and over again. Some systems do not allow multiple logins from the same account, too. Also, the built-in Linux commands and the nmon tool should be sufficient for monitoring resource usage on the School Server during test execution.

The load on a server primarily depends on the number of concurrent users and not on the total number of user accounts in the system nor the

number of users logged in at a given point in time. There are different definitions of concurrent users on different systems though [39]. Generally, concurrent users are those that are causing the server to actively do something for them at the same time, such as processing a page, querying the database, or transferring a file. In other words, many similar computations are being performed simultaneously and possibly interacting with each other on the server. Many users trying to post to a forum at the same time or many users trying to watch a certain video at the same time are considered concurrent users in this case. Therefore, test cases must be designed with this fact in mind.

It will take a very long time if the tester needs to manually create hundreds or thousands of accounts in the system for testing. Although Moodle provides the option to perform a bulk upload of user accounts, there is still a need to easily create a CSV file with many records of account information. Therefore, a Perl script was written to perform this task. It generates a CSV file for upload to Moodle to create many user accounts at once along with various settings such as enrolling each user into an existing course. The script also generates a CSV file that can be set to be used in a JMeter test so each thread can use a record in the file, thus simulating different users logging in at the same time. The script and its usage can be found in Appendix E.

In addition to the number of concurrent users, there are other factors that can have a significant impact on the test results. First, a course representative of the actual courses used in the field is needed. In the

countries where OLPC is deployed, the course materials are different from what a typical course here in the United States would contain. Each user loads the course page in Moodle and the amount of content on the course page contributes to the overall response time. For the project, a sample course was created, which is shown below in Figure 7.a. Due to the lack of data from the field, we could only estimate what should be in the course. Also, the ramp-up time should not be overlooked. It is highly unlikely that all students try to connect to the School Server at the exact same time after they were told to do so. The amount of time to spread out the connection requests to the XS can greatly affect the overall response time. Based on a video of an OLPC deployment, we estimated the average ramp-up time to be 60 seconds [40]. That was the value used for the experiments in this project. In addition, to make the load more realistic, a Gaussian Random Timer was added to each JMeter test case to add a small random delay between each request. This is equivalent to a Normal distribution of the load.

A ramp-up time of 60 seconds can be viewed as aggressive. It really depends on the number of students in the classroom. There are lots of variations, leading to a wide range of possible values. We do not have good estimates and cannot observe the actual ramp-up time in a classroom in a remote part of Peru, for example. The effects to system performance could be significant due to the load being spread out over a longer period of time. An experiment was performed to confirm this and the results can be found in

Appendix I. In the remainder of this paper, we will use 60 seconds as the average ramp-up time.

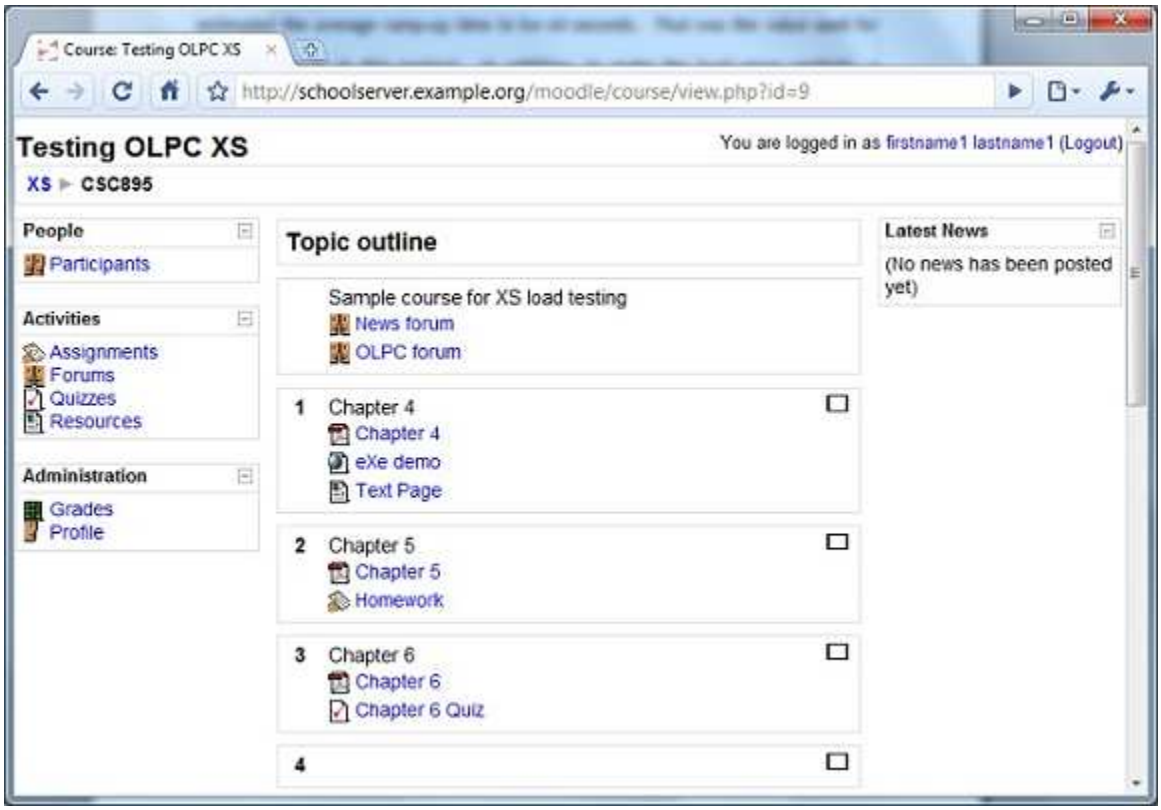


Figure 7.a Sample Course in a Typical OLPC deployment

It would be too time-consuming to set up the test environment over and over again for each execution of the JMeter test on each of the machines. User accounts have to be uploaded. The sample course has to be restored to Moodle. Various IDs in Moodle need to be known and configured into JMeter. These IDs are different for each installation. The forum should be restored to its original state for each test run. The aforementioned list is just some of the steps needed to set up the test environment. A couple of Selenium scripts were written for the purpose of setting up and cleaning up Moodle for test execution. These scripts can be found in Appendix F.

7.2 Test Case Details

This section provides details of all the test cases in the test plan, including why they were included and their significance in the testing process.

7.2.1 Descriptions of Functional Test Cases

The following table provides details about the functional test cases. The test cases are prefixed XSF, which stands for School Server Functional test.

Test Case ID	Description	Purpose/Significance
XSF001	The Moodle homepage for logging in is up and available.	This is the first and most important test case. It makes sure the system is available to users.
XSF002	A user cannot log in with incorrect credentials.	This test case makes sure only authorized users can access the system.
XSF003	The teacher is able to log in to the system.	It is important that the teacher can log in. The following 10 test cases depend on this one.
XSF004	The teacher has the ability and can successfully add a new course.	The teacher cannot teach if a course cannot be created.
XSF005	The teacher can adjust the settings of an existing course.	This is necessary for the teacher to organize the course and promote learning.
XSF006	The teacher can add a new topic to the News forum.	This is necessary for the teacher to organize the course and promote learning.
XSF007	The teacher can delete an existing topic from the News forum.	This is necessary for the teacher to organize the course and promote learning.
XSF008	The teacher is able to add a new event.	This is necessary for the teacher to organize the course and promote learning.

Test Case ID	Description	Purpose/Significance
XSF009	The teacher is able to delete an existing event.	This is necessary for the teacher to organize the course and promote learning.
XSF010	The teacher has the ability and can successfully add a resource (such as a label, text page, or web page) to the weekly outline.	This is necessary for the teacher to organize the course and promote learning.
XSF011	The teacher has the ability and can successfully add an activity (such as a forum, quiz, or glossary) to the weekly outline.	This is necessary for the teacher to organize the course and promote learning.
XSF012	The student is able to log in to the system.	It is important that the student can log in. The following 9 test cases depend on this one.
XSF013	The student is able to enroll in an existing course created by a teacher.	The student cannot access a course and learn if he/she is not able to enroll in it.
XSF014	The student can access the Participants page of an enrolled course.	The student should be able to see who else is participating in the same course.
XSF015	The student can access the News forum of an enrolled course.	The student needs to access what is provided by the teacher.
XSF016	The student is able to access a resource added by the teacher.	The student needs to access what is provided by the teacher.
XSF017	The student is able to access a forum created by the teacher and add a topic.	The student needs to be able to discuss a topic with others.
XSF018	The student can access the Grades page of an enrolled course.	The student needs to know what his/her grades are in the course.
XSF019	The student can access the Profile page of an enrolled course.	The student needs to be able to access his/her own profile and edit it for others to see.
XSF020	The student can modify his/her Profile page such as adding a	The student needs to be able to access his/her own profile and

Test Case ID	Description	Purpose/Significance
	description.	edit it for others to see.

Table 1. Functional Test Cases

7.2.2 Descriptions of Performance/Load Test Cases

The following table provides details about the load and performance test cases. The test cases are prefixed XSP, which stands for School Server Performance test.

Test Case ID	Description	Purpose/Significance
XSP001	Student logs in to the School Server. Since the XO automatically sends its credentials to the XS, the loading of the Moodle homepage is eliminated to better represent this scenario.	This is the most common action performed by users of the system. When a class starts, almost all of the students will try to connect to the system at the same time.
XSP002	Student accesses an enrolled course.	This is another common action that will be performed concurrently. All the students will try to access the course after logging in to the system.
XSP003	Student accesses a text page resource in the enrolled course. A page of text equivalent to 2MB in file size is used.	This test case simply checks the performance of the server when many students access a resource such as a text page stored in the database.
XSP004	Student accesses a forum in the enrolled course and replies to an existing thread.	This test case is more complex compared to the previous one. It involves a write to the database after the user clicks the button to post the reply.
XSP005	Student uploads a homework	This test case represents a

Test Case ID	Description	Purpose/Significance
	assignment to the enrolled course. The uploaded file is a simple picture that was taken with the XO's built-in webcam and has a size of 55KB.	common scenario where all the students submit their homework assignment when that is due.
XSP006	Student takes a quiz in the enrolled course. The quiz consists of 9 multiple choice questions based on lectures in the sample course.	The Quiz module is known to stretch database performance. This test case simulates many users taking a quiz at the same time.

Table 2. Performance/Load Test Cases

This chapter provided details about the decisions made when the test plan was written, including decisions on test approach, what testing tools to use, and what test cases to include. Once the test plan was written, the test plan implementation phase took place. The following chapter describes the test executions and results.

8. Test Plan Implementation and Results

This chapter provides information about the implementation phase of the test plan. This phase involves running each of the test cases described in the test plan and logging the results. Executing the functional test cases ensures that the system is performing as intended. If the system is functioning correctly, then the performance/load test cases can be executed to give a measure of the load the machine under test can handle. Therefore, the quality of the test cases described in the previous chapter is very important.

8.1 Test Execution

Testing was done on six different machines, ranging from a XO laptop to a Dell Xeon workstation. The XO laptop used for testing was the same as the ones used by students in the field. The FitPC and FitPC2 machines are compact, fanless, and power-efficient. They are potential candidates for an OLPC deployment. The OLPCorps SolidLogic machine is the actual hardware used in the field and thus must be included. A regular P4 desktop computer and a powerful Dell Xeon workstation were included in the project to see how their performance compare to the other machines. Specifications of all test machines can be found in Appendix H.

After the test environment was set up according to the test plan, a Selenium RC server was started on the machine executing the tests. Then the Selenium tests were executed against the School Server. After running the functional tests, JMeter was started on the machine executing the tests. The server monitoring tool, nmon, was started on the School Server so it could

monitor system resources during execution of the JMeter tests. The server was also restarted between each test run to avoid any data from being cached in memory.

For each JMeter test, we used 25 threads as an initial starting point. Since it would take a considerable amount of time to find the point where errors start occurring if we increased the number of threads by increments of 25, we took a guess at where this breaking point would be. In the case of the 'Log In' test (test case ID XSP001) being executed on the Dell Xeon workstation, the guess was 1000 threads. The corresponding error rate was 12.5% and so we started decreasing the number of threads by 25 until we found the breaking point to be at 850 threads. From there, we executed the test to get the average response time for four more data points (threads = 825, 800, 775, 750) prior to the breaking point. The same approach was used for all test executions.

8.2 Test Results

This section presents detailed test results and analysis for the OLPCorps SolidLogic machine since it is the actual hardware used in the field. Test results for the other machines can be found in Appendix J. The Selenium tests for checking functionality were executed on each machine. All the tests passed. The results from running the JMeter tests required more attention and analysis.

8.2.1 JMeter Test Results for OLPCorps SolidLogic

The following pages show the results from running the JMeter tests against the OLPCorps SolidLogic machine. Figures 13 – 18 show graphs in the following order: number of threads/users versus total average response time, CPU utilization, free memory, and disk data rate. The graphs of the server resources correspond to the test run where the number of threads caused the error rate to become non-zero.

Figure 8.a shows the JMeter test results for the test case where a user logs in to Moodle. The username and password were sent in one request to the School Server and upon successful verification, the user sees a landing page in Moodle. As can be seen in the graph, the average response time increased proportionally with the total number of users. 225 users were able to successfully log in with an average response time of about 65 seconds. Errors started occurring when the number of users was equal to 250. The average response time stayed around the same level but approximately 7% of users failed to log in. The error rate doubled to 15% with 275 users. Figure 8.b shows the graphs from nmon's monitoring of server resources. The server's CPU was at full load during the test execution. Free memory decreased from 700MB to 500MB. The disk writes were probably Moodle recording the log in time-stamp of each user.

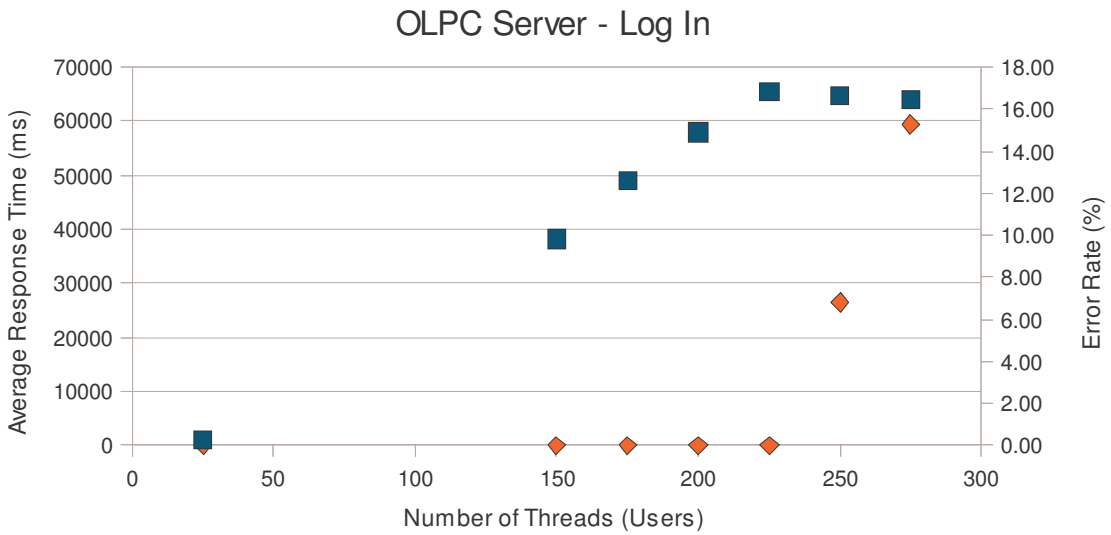


Figure 8.a JMeter Test Result - OLPCorps SolidLogic – Log In

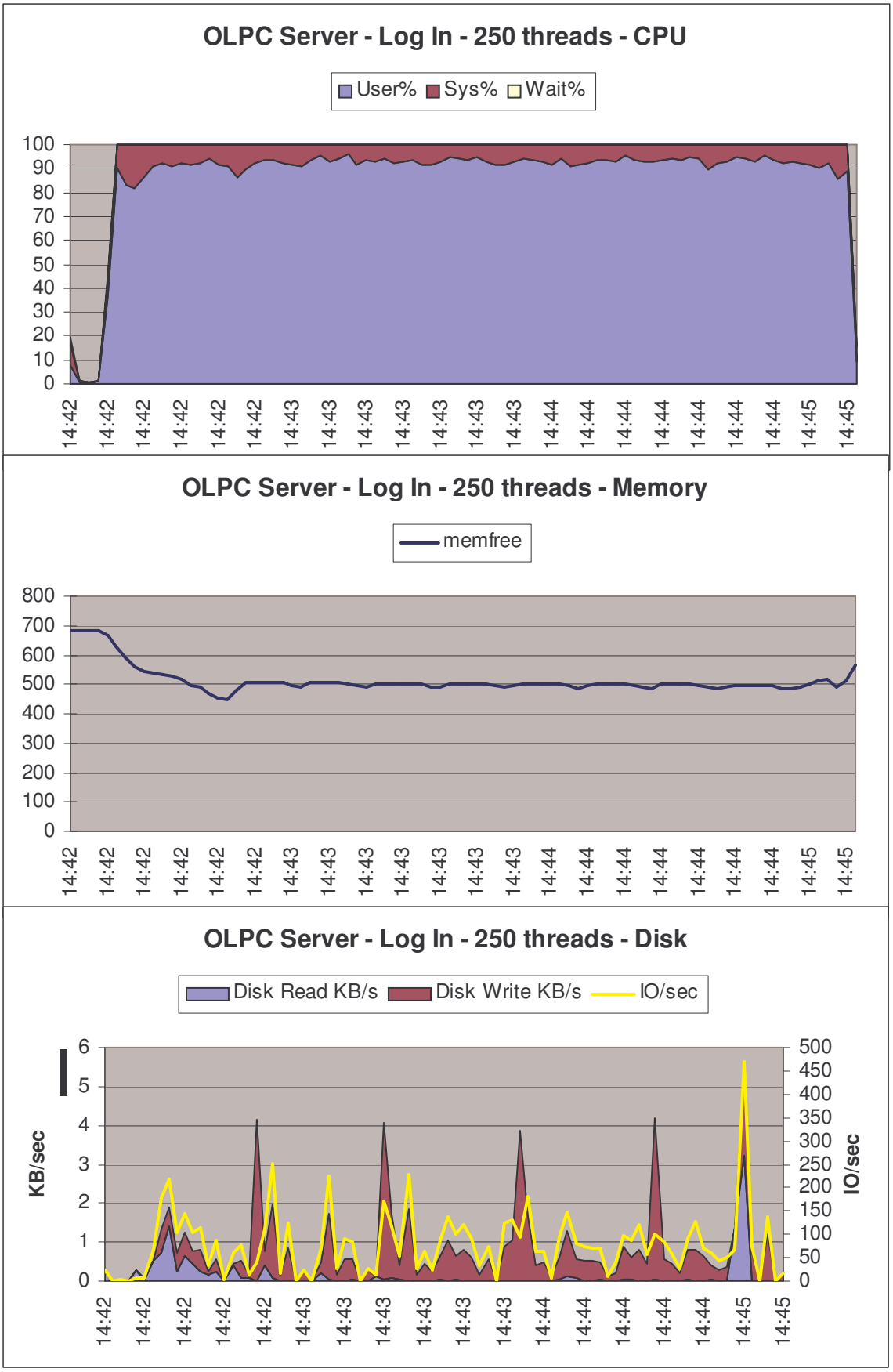


Figure 8.b Nmon Graphs - OLPCorps SolidLogic – Log In

Figure 8.c shows the JMeter test results for the next test case. After the user reaches the landing page, the user needs to click on an enrolled course to access the contents. This test case included an additional request and response compared to the previous test case. As we would expect, errors started occurring earlier this time. About 3% of the users either failed to log in or access the course when the total number of users was 175. Again, after this point, the total average response time stayed around the same value, which was around 100 seconds in this case, but the error rate increased at a fast rate as the number of users increased. Again, Figure 8.d indicates the server's CPU was at full load during test execution. Free memory decreased from 700MB to less than 500MB due to the increased requests and responses. Disk data rate also increased when compared to the previous test case.

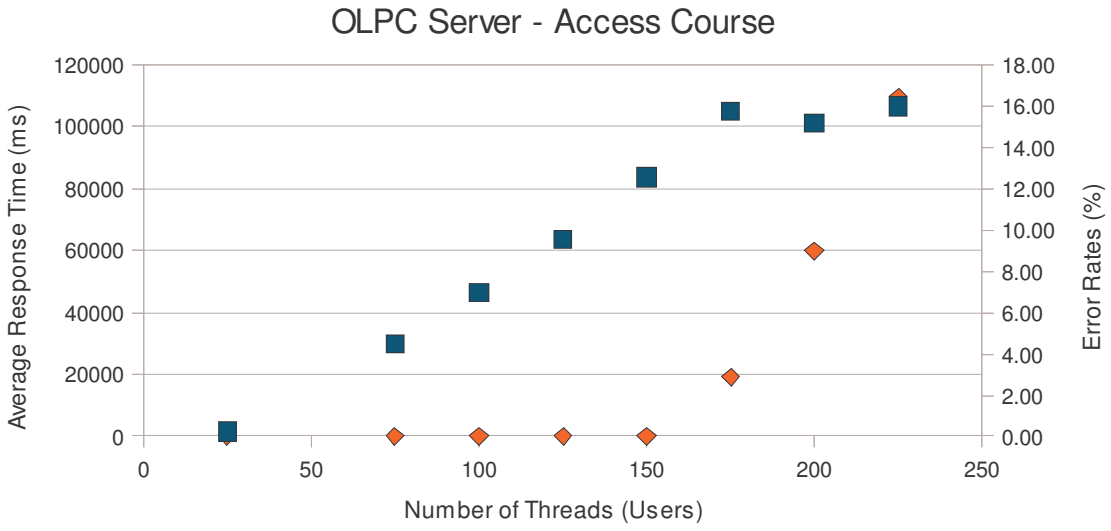


Figure 8.c JMeter Test Result - OLPCorps SolidLogic – Access Course

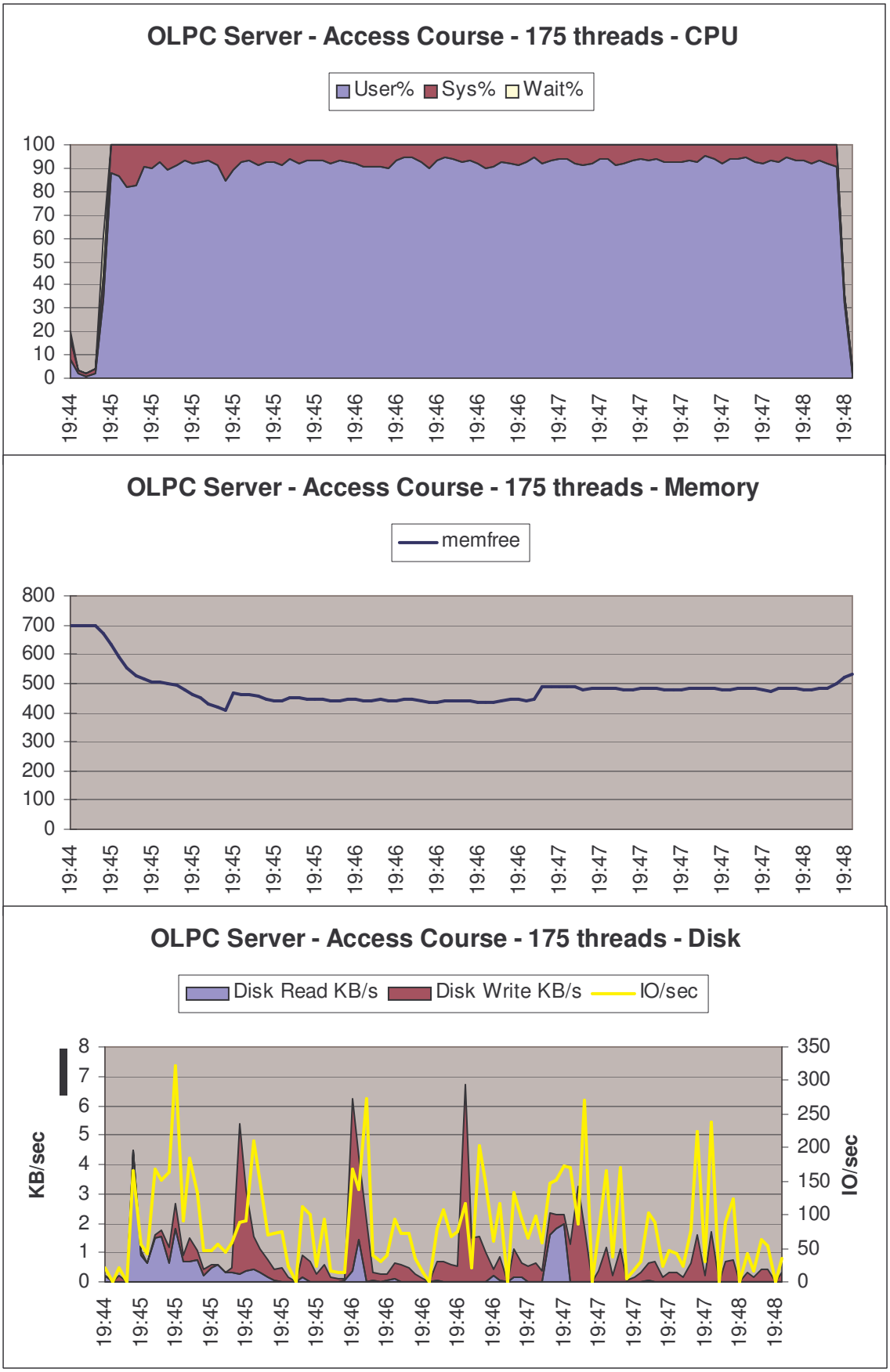


Figure 8.d Nmon Graphs - OLPCorps SolidLogic – Access Course

Figure 8.e shows the JMeter test results for accessing a resource within the course. This resource was a text page containing about 2MB of data. Similarly, errors started occurring when the total number of users was 175. The total average response time was 2 to 3 times higher than the previous test case due to the time needed to transfer the data to the users. As can be seen in Figure 8.f, free memory dramatically decreased from 700MB to close to 0 at 4 points during the 10 minute test. It looks like the server was busy for the first two minutes and then started the disk read/write when memory was at its lowest point. This can be seen a few more times in the graph too. Disk I/O happened when memory was near its lowest.

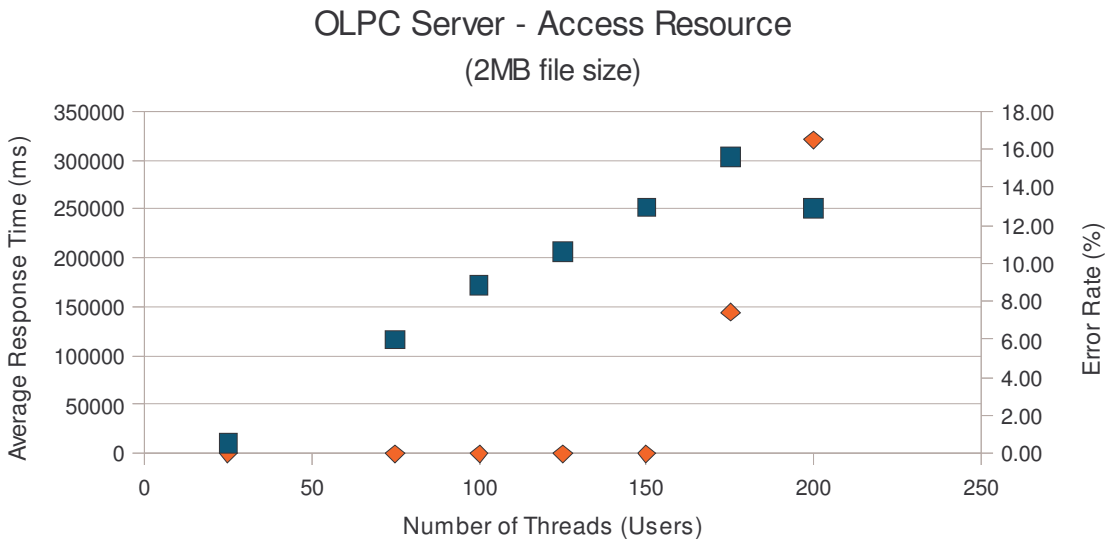


Figure 8.e JMeter Test Result - OLPCorps SolidLogic – Access Resource

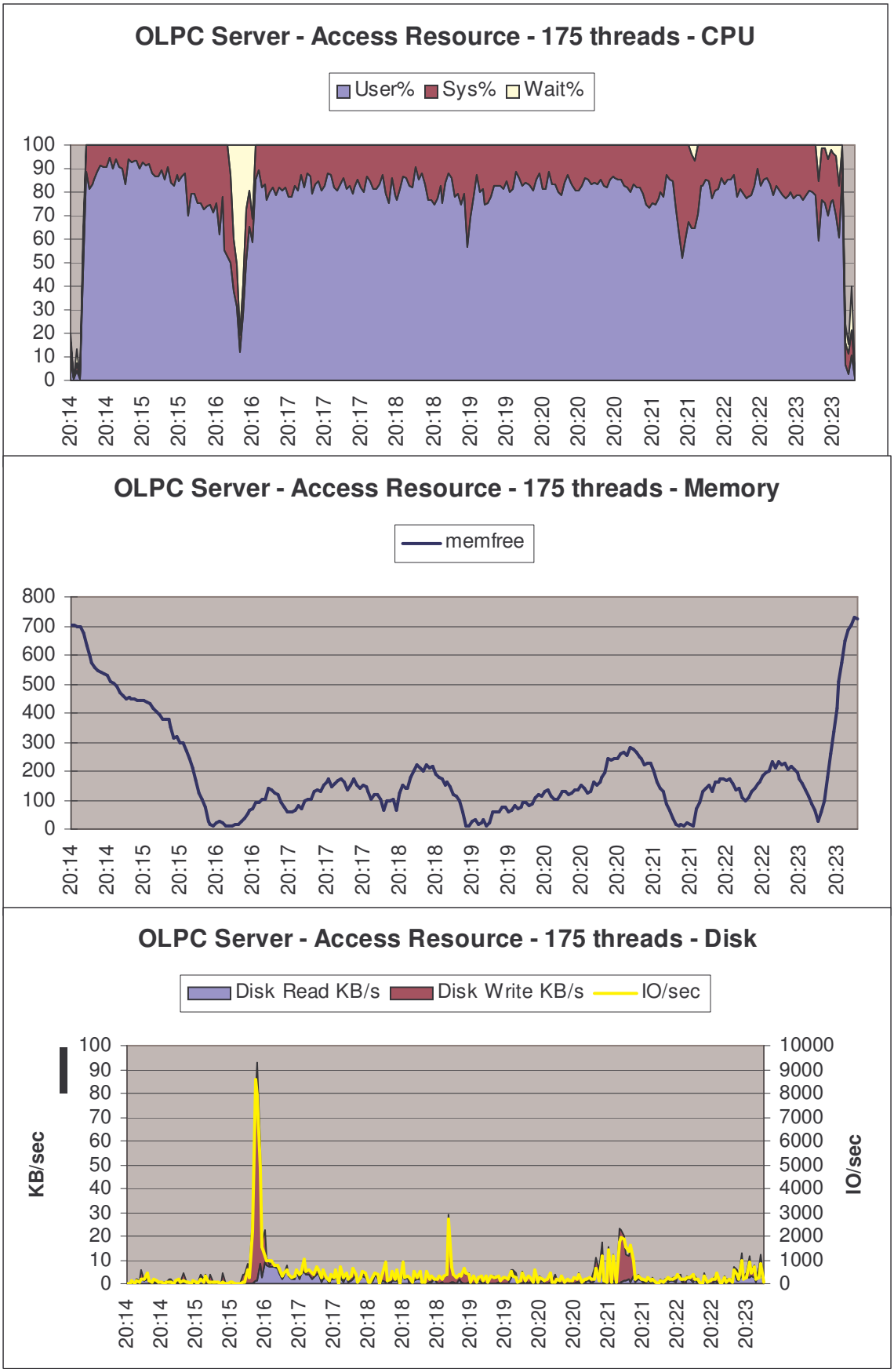


Figure 8.f Nmon Graphs - OLPCorps SolidLogic - Access Resource

Figure 8.g shows the JMeter test results for posting to a thread in the class forum. Each user logged in, access the course, loaded an existing thread in the forum, and responded to it by posting a randomly generated string. The total average response time was a little higher than the previous test case but errors started happening at 175 users, too. Adding 25 more users caused the error rate to more than double. This is similar to what we've seen in the previous cases. Figure 8.h confirms that there were a lot of disk activities while the CPU was fully utilized and free memory fluctuated between 400MB and 500MB.

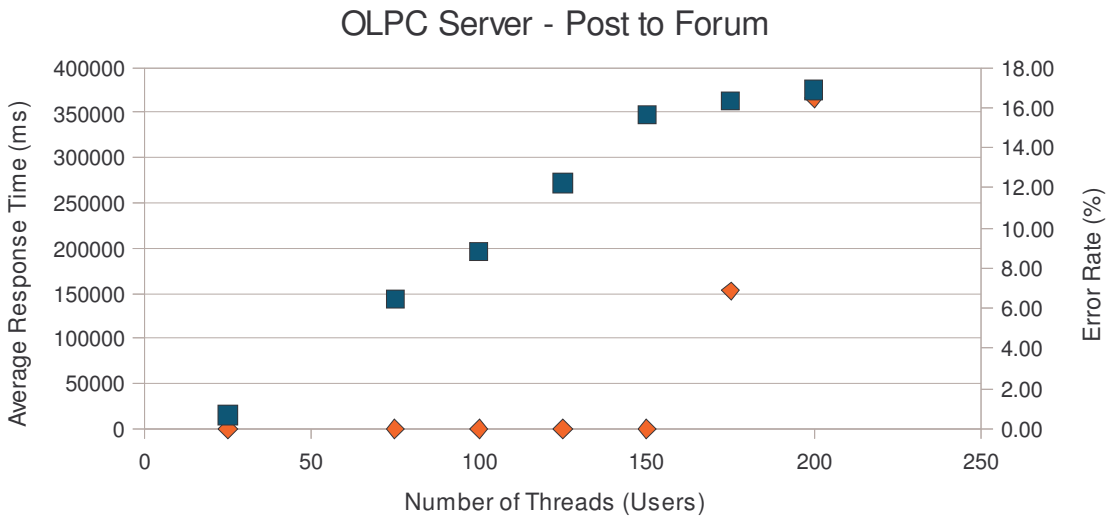


Figure 8.g JMeter Test Result - OLPCorps SolidLogic – Post to Forum

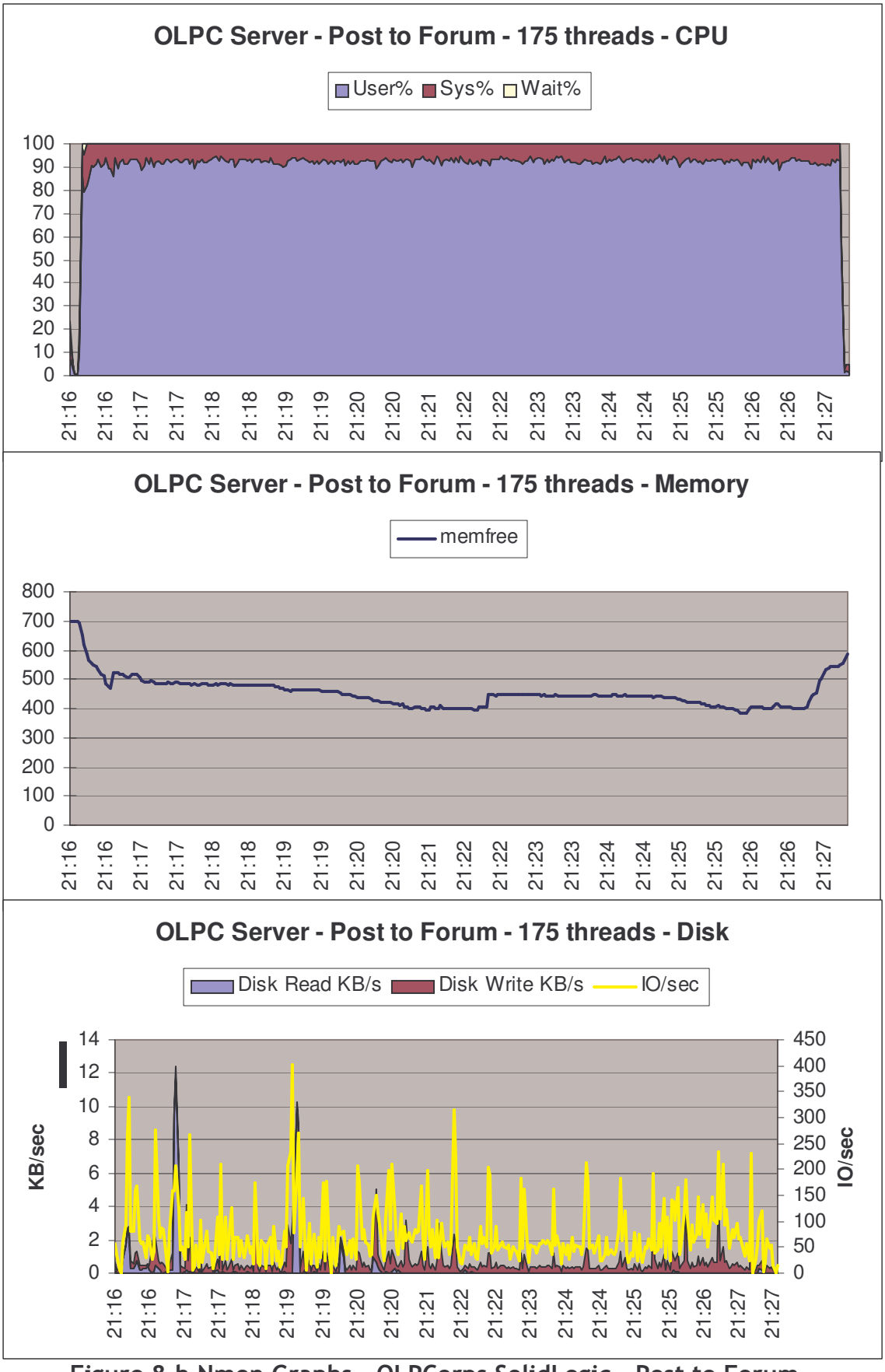


Figure 8.h Nmon Graphs - OLPCorps SolidLogic – Post to Forum

Figure 8.i shows the JMeter test results for uploading a homework assignment. The file uploaded was a small JPEG file taken with the built-in webcam in the XO. The file size was around 50KB. The total average response time was much lower compared to the test case for posting to the forum. Other than that, everything else was similar. Errors started appearing when the number of users was 175. Adding 25 more users resulted in a much higher error rate. The nmon graphs are shown in Figure 8.j. CPU was at full speed. Free memory stayed constant between 400MB to 500MB. There were plenty of disk reads and writes.

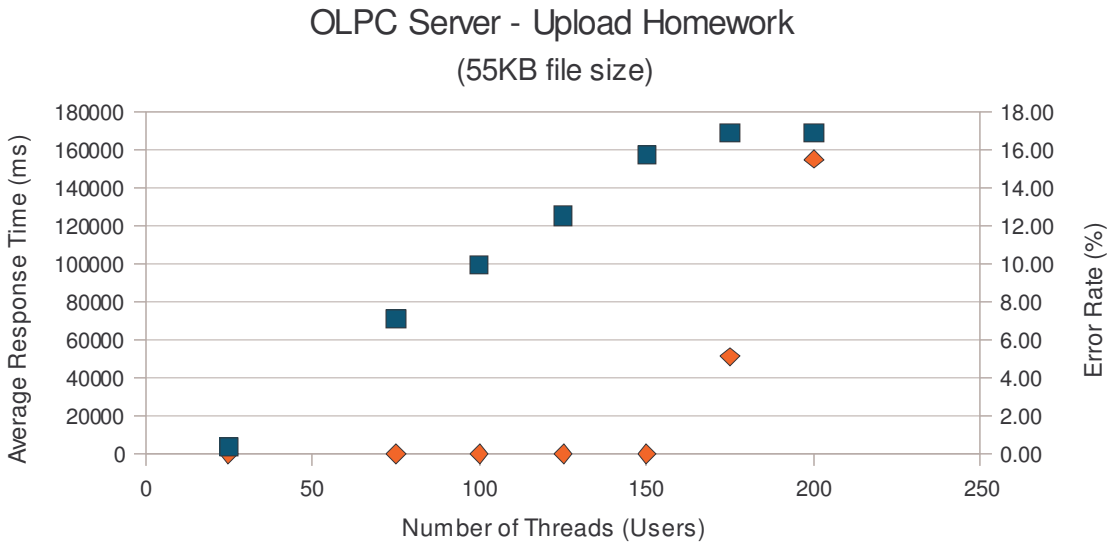


Figure 8.i JMeter Test Result - OLPCorps SolidLogic – Upload Homework

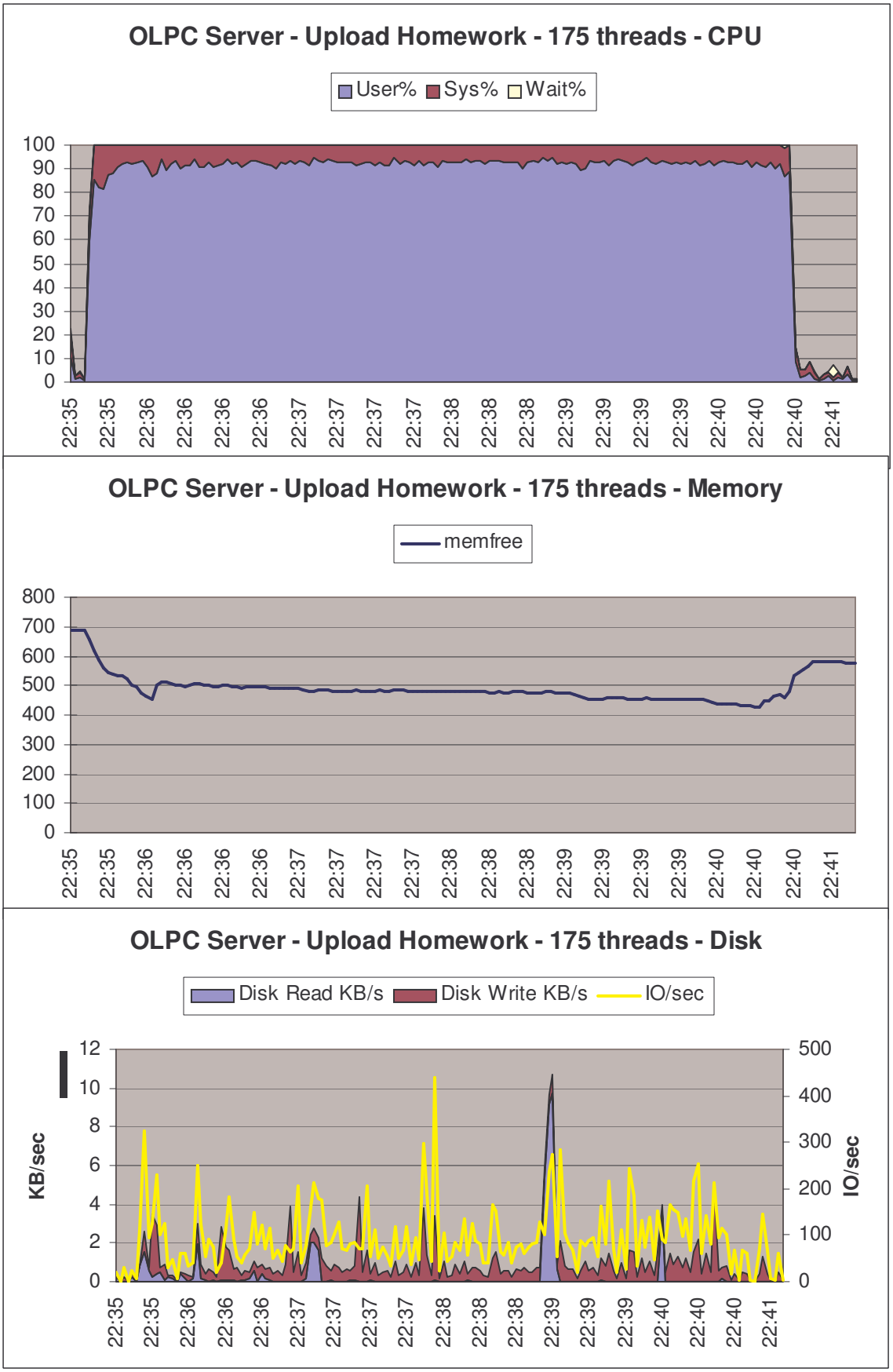


Figure 8.j Nmon Graphs - OLPCorps SolidLogic – Upload Homework

Figure 8.k shows the JMeter test results for users taking a quiz. The quiz module is known to be slow and stretches database performance [25]. Not surprisingly, at the point when errors started occurring, the total average response time was over 460 seconds. The graphs in Figure 8.l show a lot of disk reads/writes while the CPU was 100% busy and free memory ranged from 400MB to 500MB.

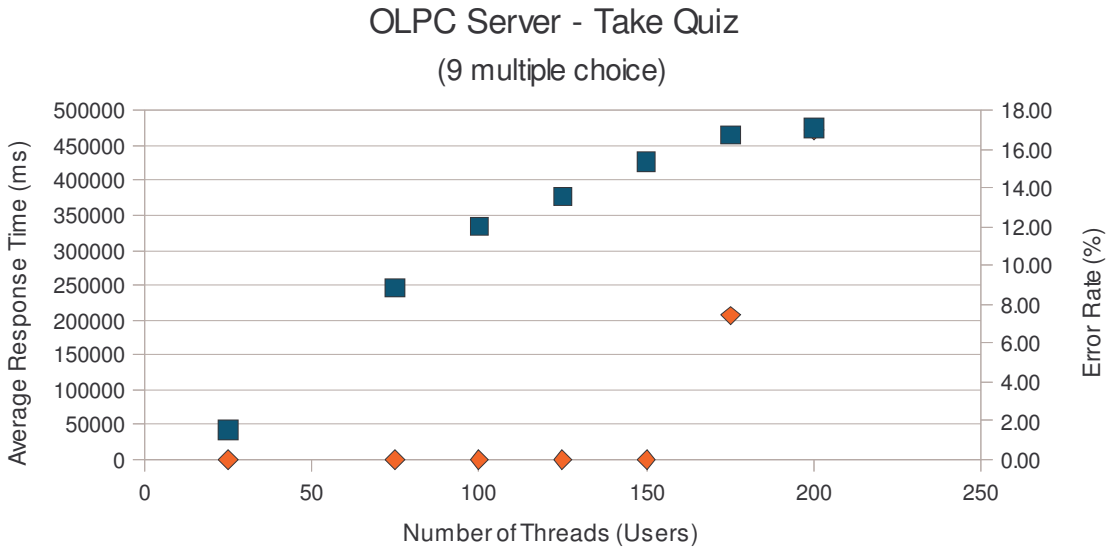


Figure 8.k JMeter Test Result - OLPCorps SolidLogic – Take Quiz

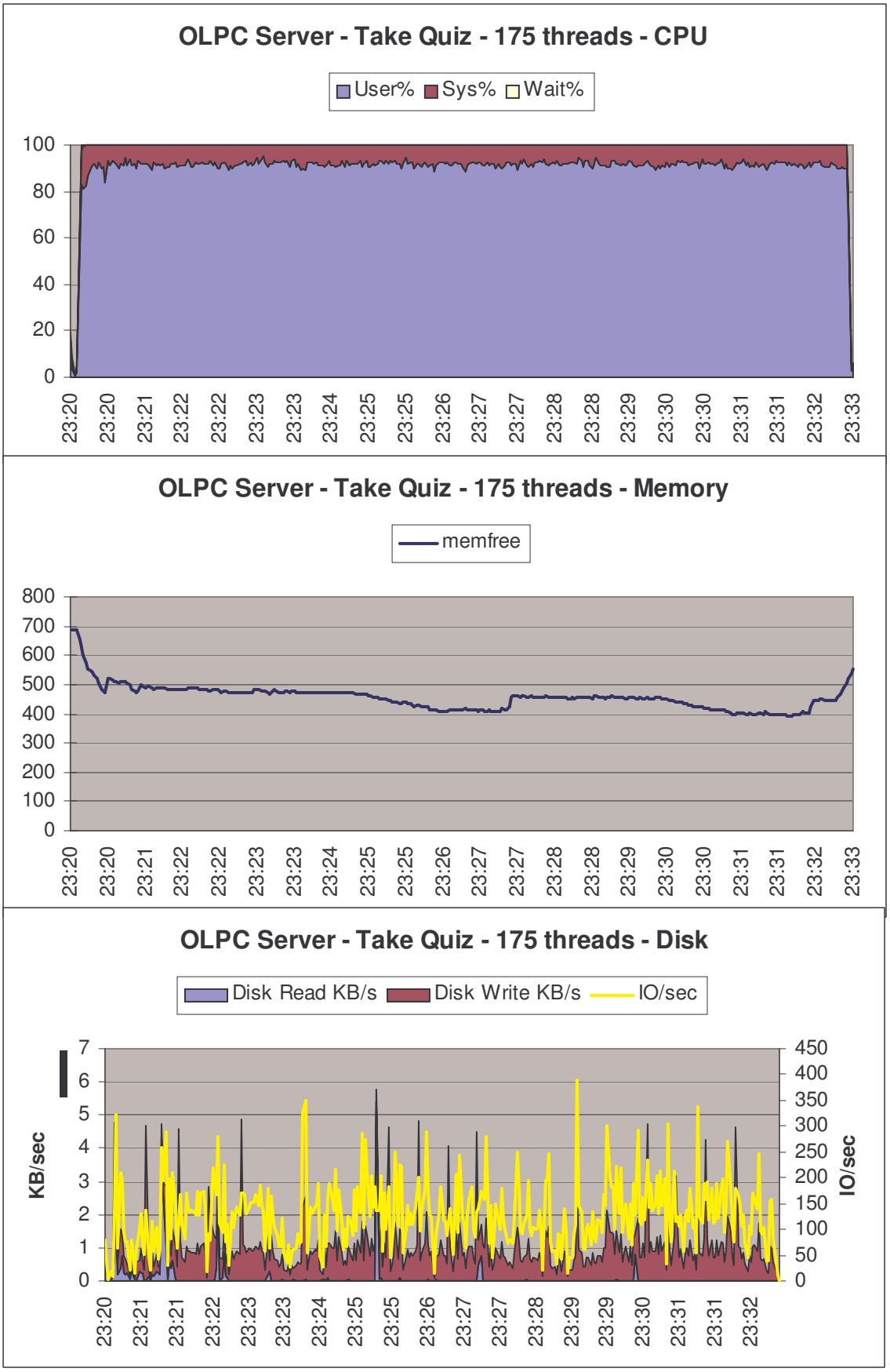


Figure 8.l Nmon Graphs - OLPCorps SolidLogic – Take Quiz

8.2.2 Comparison of JMeter Test Results

The JMeter test results for the other 5 machines can be found in Appendix J. Looking at the graphs in the other test results, similarities can be seen. For the test case where users access a resource in the course, a lot of memory was being used. The Dell workstation had the largest amount of memory and its free memory decreased from 2750MB to 1000MB at one point. For the rest of the machines, memory stayed very low throughout the testing and came close to 0 at some points. There were always a large number of disk reads/writes for the test case where users posted to the forum. The total average response time was always the highest for the quiz test case. Also, notice the XO was only able to support 6 accessing the 2MB resource at the same time. This was due to the XO only having 80MB of memory free to begin with and there was no hard disk. Data was stored on a secure digital card.

8.2.3 Discussion

Figure 8.m consists of data from the test results. For each test machine, it shows the maximum number of users that were able to successfully complete the test case with no errors and the total average response time for each user. As we can see, 125 users took an average of 164 seconds to log in with the School Server running on the XO. In other words, if 125 users try to log in to Moodle over a period of 1 minute, each user will have to wait close to 3 minutes before seeing the next page. Although there was no error for everyone, the total average response time might not be acceptable to most

Test Machine	Test Case	Number of Users	Total Average Response Time (ms)
XO	Log In	125	163868
	Access Course	125	276929
	Access Resource	6	51674
	Post to Forum	125	1298354
	Upload Homework	100	426129
	Take Quiz	100	2193693
FitPC	Log In	150	94795
	Access Course	150	180032
	Access Resource	125	343659
	Post to Forum	125	509705
	Upload Homework	125	242833
	Take Quiz	125	596779
FitPC2	Log In	200	66711
	Access Course	175	100757
	Access Resource	175	354608
	Post to Forum	175	448456
	Upload Homework	175	198273
	Take Quiz	175	496532
OLPCorps SolidLogic	Log In	225	65528
	Access Course	150	83638
	Access Resource	150	251650
	Post to Forum	150	347281
	Upload Homework	150	157451
	Take Quiz	150	426507
P4 Desktop	Log In	450	22163
	Access Course	300	32413
	Access Resource	175	167049
	Post to Forum	175	109487
	Upload Homework	225	51454
	Take Quiz	175	106769
Dell Xeon Workstation	Log In	825	11385
	Access Course	525	15746
	Access Resource	325	33960
	Post to Forum	250	56939
	Upload Homework	350	25323
	Take Quiz	250	52044

Figure 8.m Comparing the Test Results

users. What total average response time is acceptable or unacceptable will not be discussed here.

The purpose of this project was to create a method to help determine what hardware is ideal for use given the total number of students. Or, vice versa, given a system that could potentially act as the server, we can use the same method to find out how many users it is capable of supporting. Functional testing using Selenium ensured each system in the experiments was working correctly. Load testing using Apache JMeter provided us the data in Figure 8.m. Nmon for Linux verified the correctness of this data by logging and presenting consistent patterns of CPU usage, free memory, and disk I/O activities across different test cases. The different hardware systems used in the experiments, ranging from the XO itself to a powerful workstation, presented data across a spectrum. This array of data allowed us to see the relationship between increased hardware performance and the maximum number of users a system can handle. The method of testing presented in this paper helps to solve the stated problem. For example, if the FitPC is being considered for deployment, we can claim it can support up to 150 users. The number would increase to 175 for the FitPC2 along with decreased total average response times. However, as a safety measure, we recommend reducing the total number of users in the table by 20% as this will help ensure the School Server hardware can really support the load. That would bring the maximum number of users supported by the FitPC2 down to 140.

The test results were presented in this chapter. The functional test results ensured that the system was working according to the specified requirements. After this was done, the system was ready for the load testing. Server-side resources were monitored while tests were being executed. Analysis of the test results and resource usage was performed afterward. The next chapter is the concluding chapter describing the outcome of this project and recommendations for future work that could be done to enhance this project.

9. Conclusion and Future Work

This chapter presents the conclusion of this project, along with a section describing possible future work that could be performed to extend and enhance testing of the OLPC School Server.

9.1 Conclusion

The OLPC School Server was designed to be installed on generic low-end servers with limited hardware resources. The limitations put a cap on how many students can be supported simultaneously when they are all trying to access the server at a given point in time. The main goal of this project was to design, write, and execute a test plan on a set of representative computer systems in order to help determine what hardware combination would be needed given the total number of students in a school.

We wrote a test plan for carrying out functional and load tests against the OLPC School Server. In the process of writing the test plan, certain decisions had to be made, including what machines to test against, what tools to use for testing, what test cases to include, what a sample course should contain, and the estimated ramp-up time of users. Every one of these decisions were crucial and affected the overall results of this project. We ran into several technical issues during the project such as setting up the network connections, obtaining initial access to Moodle, generating and uploading Moodle user accounts in bulk, bypassing an unexpected session key error, and installing tools for server-side resource monitoring. All the technical problems

and solutions were documented in this paper, which will benefit any contributor who plans on doing future work relevant to this project.

Once the test plan was written and the test environment was set up, the tests were executed. Results were logged and resources were monitored. A detailed description and analysis of the results were provided in the previous chapter, along with a comparison of the results from all the test machines.

Given an estimated total number of students at a future deployment site, we wanted to know what hardware combination would be ideal for use as the School Server. The hardware selection process depends on the number of XOs in the school, the cost of hardware, power requirements of the hardware, and the availability of electricity at the deployment site. This project provided a formal process to set up, test, and measure the capabilities of potential machines. The outcome of this project included a test plan, test scripts, and experimental results from executing these tests against a set of test machines. The testing process can be easily replicated with a different set of test machines for future OLPC deployments.

9.2 Future Work

There is no easy formula to calculate the maximum number of concurrent users a system is capable of supporting. That number highly depends on the hardware, software, and network combination involved. There are many performance factors that lie within each category. Having a faster processor will help to a certain extent, but it is usually the amount of memory installed that is the deciding factor. Performance also depends on the

configurations of the different software running on the system. Work can be done to benchmark each component of the software stack – operating system, web server, database server, PHP engine and accelerator, for example. Further work could involve tuning variables in the configurations one-by-one, executing tests against the system, and analyzing the effects of a single change. The work done in this project provided a process and a set of tests that can act as a starting point for a much bigger project and thus, only barely scratched the surface of what can be done.

10. References

[1] One Laptop Per Child http://laptop.org/en/vision/index.shtml Last accessed on April 8, 2010.

[2] One Laptop Per Child: Deployments http://wiki.laptop.org/go/Deployments Last accessed on November 18, 2010.

[3] One Laptop Per Child: School Server http://wiki.laptop.org/go/XS Last accessed on April 8, 2010.

[4] About Moodle http://docs.moodle.org/en/About_Moodle Last accessed on April 8, 2010.

[5] Moodle Statistics http://moodle.org/stats/ Last accessed on August 19, 2010.

[6] Moodle Developers http://moodle.org/mod/cvsadmin/view.php?cid=1 Last accessed on August 19, 2010.

[7] Martin Langhoff Brings Change to XS-ive OLPC School Server http://www.olpcnews.com/hardware/school_servers/martin_langhoff_brin_1.html Last accessed on April 8, 2010.

[8] One Laptop Per Child: Communication Channels http://wiki.laptop.org/go/Communication_channels Last accessed on August 19, 2010.

[9] One Laptop Per Child: Deployment Guide http://wiki.laptop.org/go/Deployment_Guide Last accessed on April 8, 2010.

[10] Kaner, C., Falk, J., Nguyen H. (1999) *Testing Computer Software, 2nd Edition*. John Wiley & Sons, Inc.

[11] McAfee glitch causes trouble in RI emergency rooms http://ww.abc6.com/Global/story.asp?S=12351281 Last accessed on April 22, 2010.

[12] Performance vs. load vs. stress testing http://agiletesting.blogspot.com/2005/02/performance-vs-load-vs-stress-testing.html Last accessed on August 19, 2010.

[13] Moodle Roadmap http://docs.moodle.org/en/Roadmap Last accessed on November 18, 2010.

[14] Moodle Testing Group has been established. (9/13/2006) http://moodle.org/mod/forum/discuss.php?d=53918 Last accessed on April 8, 2010.

[15] Moodle Quality Assurance http://docs.moodle.org/en/Quality_assurance Last accessed on April 8, 2010.

[16] Using Moodle: Testing and QA http://moodle.org/mod/forum/view.php?id=56 Last accessed on April 8, 2010.

[17] Selinium for testing Moodle (8/4/2006) http://moodle.org/mod/forum/discuss.php?d=51161 Last accessed on April 8, 2010.

[18] Moodle Selenium Test (4/10/2006) http://moodle.org/mod/forum/discuss.php?d=43569 Last accessed on April 8, 2010.

[19] Using automated functional / regression testing tool? (3/7/2008) http://moodle.org/mod/forum/discuss.php?d=92050 Last accessed on April 8, 2010.

[20] Doubt on how to help with functional tests (12/19/2009) http://moodle.org/mod/forum/discuss.php?d=140271 Last accessed on April 8, 2010.

[21] Moodle participation in the GHOP project http://code.google.com/p/google-highly-open-participation-moodle/w/list Last accessed on April 8, 2010.

[22] Tool to estimate server's maximum concurrent users (8/9/2008) http://moodle.org/mod/forum/discuss.php?d=103040 Last accessed on April 9, 2010.

[23] Performance perspectives - a little script http://moodle.org/mod/forum/discuss.php?d=57028 Last accessed on April 9, 2010.

[24] JMeter Scripts (3/3/2009) http://moodle.org/mod/forum/discuss.php?d=119443 Last accessed on April 9, 2010.

[25] Moodle Performance http://docs.moodle.org/en/Performance Last accessed on April 9, 2010.

[26] Moodle Development: Performance and Scalability http://docs.moodle.org/en/Development:Performance_and_scalability Last accessed on April 9, 2010.

[27] Moodle User site capacities http://docs.moodle.org/en/User_site_capacities Last accessed on April 9, 2010.

[28] EDUCAUSE: An Open Source LMS for a Mission-Critical, Enterprise-Level Application: Are We There Yet? (3/31/2008) http://www.educause.edu/Resources/AnOpenSourceLMSforaMissionCrit/162843 Last accessed on April 9, 2010.

[29] Dr. Wen Hao Chuang's Home Page (SFSU) http://userwww.sfsu.edu/~wchuang/ Last accessed on April 9, 2010.

[30] Moodle 1.7 Quiz Handles 300 Simultaneous Users on One Server (4/5/2007) http://moodle.org/mod/forum/discuss.php?d=68579 Last accessed on April 9, 2010.

[31] One Laptop Per Child: Schoolserver Testing http://wiki.laptop.org/go/Schoolserver_Testing Last accessed on April 8, 2010.

[32] One Laptop Per Child: XS Installing Software http://wiki.laptop.org/go/XS_Installing_Software Last accessed on April 8, 2010.

[33] One Laptop Per Child: XS Moodle Design http://wiki.laptop.org/go/XS_Moodle_design Last accessed on April 8, 2010.

[34] Re: [Server-devel] Populating the Moodle db with users http://www.mail-archive.com/server-devel@lists.laptop.org/msg03580.html Last accessed on May 2, 2010.

[35] SeleniumHQ http://seleniumhq.org/ Last accessed on April 10, 2010.

[36] Apache JMeter http://jakarta.apache.org/jmeter/ Last accessed on April 10, 2010.

[37] 20 Linux System Monitoring Tools Every SysAdmin Should Know http://www.cyberciti.biz/tips/top-linux-monitoring-tools.html Last accessed on April 10, 2010.

[38] nmon for Linux http://nmon.sourceforge.net/pmwiki.php Last accessed on August 19, 2010.

[39] Concurrency
http://en.wikipedia.org/wiki/Concurrency_(computer_science) Last accessed on April 10, 2010.

[40] Web: La Selva http://www.dailymotion.com/video/xby2uj_web-la-selva_tech Last accessed on August 8, 2010.

[41] XS: OLPC School Server http://www.slideshare.net/sverma/xs-olpc-school-server Last accessed on November 20, 2010

[42] XS Release Notes http://wiki.laptop.org/go/XS_Release_Notes#XS_0.4 Last accessed on November 20, 2010

APPENDIX A - Test Plan for OLPC School Server

Test Plan for
XS School Server for 'One Laptop Per Child' Project

Contents

Document History

Date	Name	Description
Mar 17, 2010	Benjamin Tran	Original Document (Draft)
Apr 10, 2010	Benjamin Tran	Revised & Added Test Cases
Apr 23, 2010	Prof. Petkovic	Reviewed Test Plan, Provided Comments
Apr 29, 2010	Benjamin Tran	Added Test Objectives, Organized Test Cases

1. Introduction

The One Laptop Per Child Association, Inc. (OLPC) oversees the development, construction and deployment of an affordable, educational laptop (XO) for use in the developing countries. The low-cost laptop helps to revolutionize the way we educate the world's children. OLPC's mission is "to provide educational opportunities for the world's poorest children by providing each children with a rugged, low-cost, low-power, connected laptop with content and software designed for collaborative, joyful, self-empowered learning." In order to achieve the key goals of cost reduction, lower power consumption, and a smaller footprint, there are many hardware and software limitations to the XO compared to a standard laptop one can find a retail store.

The School Server (XS) is a Linux-based OS that provides networking infrastructure, services, and education and discovery tools to the XO laptops. Moodle is a free web-based course management system that educators use to create effective online learning sites. Due to the cost and limited hardware resources, there exist uncertainties in areas such as performance and reliability. With the limited resources, how many users can reasonably connect to the server and use Moodle? What hardware combination (XO-1, XO-1.5, fit-PC, etc.) would be best for use with the XS given the total number of students? The purpose of this project is to perform functional and performance/load testing against the School Server loaded on different systems. The outcome of this project will be a systematic analysis of the performance of the XS under different load conditions and hardware.

More information about the School Server can be found here. The Moodle project's website is located here.

2. Test Objectives

The objectives of the current testing effort are to ensure major components of the School Server, specifically Moodle, work as intended and to find out what hardware would be suitable for a school of a given size. Aside from the operating system, the School Server consists of major software programs such as Apache HTTP Server, PostgreSQL database management system, PHP: Hypertext Preprocessor engine, and Moodle course management system. It is critical to ensure that the system is functioning according to specifications. Also, due to constraints such as hardware resources, network availability, and limited electricity, it is crucial to know what hardware is needed to support a school of a given size.

3. Testing Environment

The URL of the test server will be http://schoolserver.

3.1 Hardware

- The School Server can be installed on any computer. This will be the machine under test. Testing will include, at a minimum, the following systems:
  - Regular Desktop
  - XO 1.0
  - FitPC
- Another computer is needed for test execution. This computer needs to be Java-enabled.
- A switch and network cables are needed to connect the computers together. Alternatively, a crossover cable can be used in place of a switch.

3.2 Software

- OLPC School Server 0.6 (released October 7, 2009)
- Apache JMeter 2.3.4 (released June 21, 2009)
- Selenium RC 1.0.3 (released February 23, 2010)
- Firefox Browser (latest version)

4. Features to be Tested

Testing will include a set of acceptance tests, consisting of basic UI and functional tests, and a set of performance tests.

- UI - Moodle is available to users. Users are able to log in and see relevant links/features.
- Admin - The 'admin' is able to log in, create/delete user(s), create/edit/delete course(s), and grant/revoke permissions to access the courses.
- User - The 'user' is able to log in and access course contents but has limited capabilities compared to the 'admin.'
- Performance - The School Server remains available and performs at an acceptable level as the number of users and load vary.

5. Features Not to be Tested

These items may be tested at the beginning of the test set up, but will not be revisited unless they are changed.

- First user to log in to Moodle has 'Course Creator' role - After the initial installation of the School Server, the first XO to successfully log in to Moodle will be granted the 'Course Creator' role and have access to administration options. For testing purposes, the special 'admin' account, which is disabled by default, will be enabled via an existing script and the password will be obtained from the /etc/moodle/adminpw file. An account with 'Course Creator' role will be created as part of the setup.

- XO registration with School Server - Each XO can register itself with a School Server. After registration, the 'Register' feature disappears. For testing purposes, the 'admin' account will be used to create a list of Moodle user accounts in bulk instead. This is done via a CSV file upload after logging in to Moodle as 'admin.'
- School Identity Manager - Each laptop registers with the School Server via the Identity Manager. For the same reason as above, this will not be tested.
- XO automatically logs in to Moodle - Each XO automatically logs in to Moodle by sending its credentials to the School Server. Testing will simulate the logging in of many user accounts. No XO will be involved and thus, the automatic log in will not be tested.
- Correct rendering of the Moodle UI in the XO's browser - The XO's embedded browser uses the same Gecko rendering engine and JavaScript language support as Firefox. It is simpler and is not directly compatible with Firefox add-ons though. Selenium tests will be executed against Firefox on a non-XO computer.

6. Test Approach

Automated Selenium tests will be used to perform basic checks of the UI and functionalities within Moodle. JMeter tests will be used to perform load testing of the School Server as a whole.

6.1 Setup of Hardware

- The machine under test should be connected to the switch.
- The computer from which tests will be executed should also be connected to the switch. Verify connectivity by trying to access the test server URL as listed in Section 2 above.

6.2 Setup of Software

- The School Server software should be installed on the machine under test.
    - The server domain name should be set via the following script: /etc/sysconfig/olpc-scripts/domain_config
        - Reboot so the hostname change takes effect: shutdown -r now
        - Confirm with either of the following commands: cat /etc/sysconfig/network or hostname -f
    - Enable the Moodle admin account: sudo -u apache php /var/www/moodle/web/local/scripts/adminuser-enable.php
    - Find the Moodle admin password: cat /etc/moodle/adminpw
    - Log in to Moodle with the special 'admin' account.
        - For functional testing:
            - Create an account with 'Course Creator' role.
            - Create a user account.
        - For load testing:
            - Create an account with 'Course Creator' role.

        - Create a sample course that contains the following: lectures/PDFs, discussion forum, text resource, link to upload homework, and a quiz.
        - Run the 'create_users_csv.pl' script to generate two CSV files: one for creating bulk user accounts in Moodle and one for CSV Data Set variables in JMeter.
        - Log in to Moodle with the special 'admin' account and create multiple users via the Upload users feature.
- JMeter, Selenium RC, and Firefox should be installed on the computer from which tests will be executed.

6.3 Test Cases

XSF = School Server Functional

XSP = School Server Performance

Functional Testing - School Server/Moodle

Test ID	Test Description	Preconditions	Step Description	Postconditions	Test Status	Notes
XSF001	The Moodle homepage for logging in is up and available.	N/A	1. Go to http://schoolserver/	• The Moodle homepage loads and shows the username/password boxes and 'Login' button.		
XSF002	A user cannot log in with incorrect credentials.	N/A	1. Go to http://schoolserver/ 2. Enter invalid username and password. 3. Click button to log in.	• User cannot log in. • Error message is displayed.		

Functional Testing - CourseCreator/Teacher

Test ID	Test Description	Preconditions	Step Description	Postconditions	Test Status	Notes
XSF003	The teacher is able to log in to the system.	• Teacher account exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log	• User is logged in with the teacher account. • User sees		

			in.	page containing 'My courses' section.		
XSF004	The teacher has the ability and can successfully add a new course.	• Teacher account has course creator role. • Course categories exist.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click 'Courses' link in 'Site Administration' section. 5. Click 'Add/edit courses' link. 6. Click on a course category. 7. Click 'Add a new course' button. 8. Enter full name and short name for course. 9. Click 'Save changes' button.	• New course is created. • User sees page with full name and short name of new course.		
XSF005	The teacher can adjust the settings of an existing course.	• A course exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Settings' link in 'Administration' section. 6. Change course full name. 7. Click 'Save changes' button.	• New course full name is shown on main page of course.		
XSF006	The teacher can add a new topic to the News forum.	• A course exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'News forum' link. 6. Click 'Add a new topic' button. 7. Enter subject and	• New topic is added. • User sees new topic in News forum.		

			message. 8. Click 'Post to forum' button.			
XSF007	The teacher can delete an existing topic from the News forum.	• A course exists. • A topic exists in the course's News forum.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'News forum' link. 6. Click on a topic. 7. Click 'Delete' link. 8. Confirm delete by clicking 'Yes' button.	• Topic deleted from forum.		
XSF008	The teacher is able to add a new event.	• A course exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'New Event...' link in 'Upcoming Events section. 6. Click 'OK' button. 7. Enter details of event. 8. Click 'Save changes' button.	• New event is created.		
XSF009	The teacher is able to delete an existing event.	• A course exists. • An event exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click on the event in 'Upcoming Events' section. 6. Click 'X' icon. 7. Click 'Delete' button to confirm.	• Event is deleted.		
XSF010	The teacher has the ability and can successfully	• A course exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username	• User sees the text page.		

	add a resource (such as a label, text page, or web page) to the weekly outline.		and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Turn editing on' button. 6. Under 'Add a resource...' dropdown for any week in the outline, select 'Compose a text page.' 7. Enter a name and some text. 8. Click 'Save and return to course' button. 9. Click 'Turn editing off' button. 10. Click on name of text page.			
XSF011	The teacher has the ability and can successfully add an activity (such as a forum, quiz, or glossary) to the weekly outline.	• A course exists.	1. Go to http://schoolserver/ 2. Enter teacher account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Turn editing on' button. 6. Under 'Add an activity...' dropdown for any week in the outline, select 'Forum.' 7. Enter name and introduction. 8. Click 'Save and return to course' button. 9. Click 'Turn editing off' button. 10. Click on name of forum.	• User sees new forum.		

Functional Testing - User/Student

Test ID	Test Description	Preconditions	Step Description	Postconditions	Test Status	Notes
XSF012	The student	• Student	1. Go to	• User is		

	is able to log in to the system.	account exists.	http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in.	logged in with the student account. • User sees page containing 'My courses' section.		
XSF013	The student is able to enroll in an existing course created by a teacher.	• A course exists.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click 'All courses' button. 5. Click on a course. 6. Click 'Yes' to confirm enrollment.	• User is enrolled. • User sees weekly outline of course.		
XSF014	The student can access the Participants page of an enrolled course.	• Student is enrolled in a course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Participants' link in 'People' section.	• User is able to access the page.		
XSF015	The student can access the News forum of an enrolled course.	• Student is enrolled in a course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course.	• User is able to access forum.		

			5. Click 'News forum' link.			
XSF016	The student is able to access a resource added by the teacher.	• Student is enrolled in a course. • A resource exists in the course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click on a resource in weekly outline.	• User is able to access the resource.		
XSF017	The student is able to access forum created by the teacher and add a topic.	• Student is enrolled in a course. • Teacher created a forum.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click on a forum in weekly outline. 6. Click on 'Add a new discussion topic' button. 7. Enter subject and message. 8. Click “Post to forum' button.	• New topic is created in forum.		
XSF018	The student can access the Grades page of an enrolled course.	• Student is enrolled in a course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Grades' link in 'Administratio	• User is able to access the page.		

			n' section.			
XSF019	The student can access the Profile page of an enrolled course.	• Student is enrolled in a course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Profile' link in 'Administration' section.	• User is able to access the page.		
XSF020	The student can modify his/her Profile page such as adding a description.	• Student is enrolled in a course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click 'Profile' link in 'Administration' section. 6. Click 'Edit profile' tab. 7. Add a description. 8. Click 'Update profile' button.	• Profile is saved. • New description is shown.		

Load Testing - User/Student

Number of Threads: range from 25-1000 with variable increments

Test ID	Test Description	Preconditions	Step Description	Postconditions	Test Status	Notes
XSP001	Student logs in to the School Server.	• Student accounts exist.	1. Go to http://schoolserver/ 2. Enter student account's username and password.	• User is logged in with the student account. • User sees page		

			3. Click button to log in.	containing 'My courses' section.		
XSP002	Student accesses an enrolled course.	• Students are enrolled in a course.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course.	• User sees weekly outline.		
XSP003	Student accesses a text page resource in the enrolled course.	• Students are enrolled in a course. • A text page resource exists.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click on text resource in weekly outline.	• User is able to access the text page resource.		
XSP004	Student accesses a forum in the enrolled course and replies to an existing thread.	• Students are enrolled in a course. • A forum exists and has topics.	1. Go to http://schoolserver/ 2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click on a forum in weekly outline. 6. Click on topic in forum. 7. Click 'Reply' link. 8. Enter message. 9. Click 'Post to forum' button.	• Message is posted as a reply to the topic.		
XSP005	Student uploads a homework	• Students are enrolled	1. Go to http://schoolserver/	• File is successfully uploaded.		

	assignment to the enrolled course.	in a course. • A link exists where students can go upload a file, which is the assignment.	2. Enter student account's username and password. 3. Click button to log in. 4. Click on a course. 5. Click on the link to upload homework. 6. Select a file on the local computer. 7. Click button to upload the file.			
XSP006	Student takes a quiz in the enrolled course.	• Students are enrolled in a course. • A quiz exists.	8. Go to http://schoolserver/ 9. Enter student account's username and password. 10. Click button to log in. 11. Click on a course. 12. Click on a quiz in weekly outline. 13. Click 'Attempt quiz now' button. 14. Answer quiz. 15. Click 'Submit all and finish' button. 16. Click 'OK' button to confirm submission.	• User sees quiz results.		

7. Test Deliverables

- The School Server release notes are available here.
- Development will send an announcement to all involved as soon as a new release is available for testing. Mailing lists can be found here.
- Any issues found during testing will be reported to the OLPC Development Site (http://dev.laptop.org/).
- All data collected from executing automated load simulation tests against different hardware will be published in a separate document.

8. Schedule

Tests will be written and testing will be performed by August 31, 2010.

9. Risks, Contingencies, and Dependencies

- The School Server needs to be successfully installed and configured on the machines under test.
- The special 'admin' account can be enabled and the password is available.
- The feature to create multiple user accounts in bulk via CSV upload needs to be functional.
- The different machines/hardware for testing need to be available.

10. Reference Materials

OLPC
OLPC School Server
Moodle

Moodle Docs- Upload Users
XO 1.0 Specifications
XO 1.5 Specifications
FitPC
Apache JMeter
SeleniumHQ

APPENDIX B – Installing and Using Selenium

This section provides brief information on the installation and usage of Selenium. More detailed documentation on Selenium and its usage can be found on its website.

Installing Selenium IDE

Selenium IDE comes in the form of a .XPI file, which is pronounced “zippy” and derived from XPInstall. It is a ZIP file that contains an install script and a number of data files. This installer module is for installing Mozilla extensions to add functionality to the main application, which will be Mozilla Firefox in this case. Use the Firefox browser to download and install Selenium IDE.

Using Selenium IDE

Go to the 'Tools' menu in the Firefox browser and select 'Selenium IDE.' A small window similar to Figure 8 should appear, with the red record button active. It will start recording your actions in the browser. Use the browser to access any web page. Click on some links on the web page. Go back to the Selenium IDE extension and click the red record button to stop it from recording. The actions taken in the browser earlier should have been recorded, causing values to appear in the table in the extension. Click one of the green play buttons and watch the recorded actions take place in the browser. Many test cases can be recorded to form a test suite. The test case or entire test suite can be exported to a major programming language via the 'File' menu.

The exported test script can be executed with Selenium Remote Control against many different operating systems and browsers.

Installing Selenium Remote Control

A ZIP file containing all the files required for Selenium Remote Control can be downloaded from the Selenium website for free. The version number of Selenium Remote Control used for this project was 1.0.3 and was released on February 23, 2010. Download the ZIP file and uncompress it anywhere on your machine. It should contain a server folder and client drivers for different programming languages.

Starting Selenium Server

The Selenium Server can be started via the command-line interface. Open up a console or command prompt and browse to the directory that contains the Selenium Server JAR file.

To start the Selenium Server, use the following command:

java -jar selenium-server.jar

It should be similar to what is shown in Figure 10. Control-C can be used to shut down the server.

APPENDIX C – Installing and Using JMeter

This section provides brief information on the installation and usage of Apache JMeter. More detailed documentation on JMeter and its usage can be found on its website.

Installing JMeter

A ZIP or TGZ file containing all the files required for JMeter can be downloaded from the Apache JMeter website for free. The version number of JMeter used for this project was 2.3.4. Download the compressed file and uncompress it anywhere on your machine. Similar to the Selenium Server, JMeter comes in the form of a JAR file and does not require any installation.

Running JMeter

JMeter can be started via the command-line interface. Open up a console or command prompt and browse to the directory that contains the JMeter JAR file. This is the bin\ directory under the folder where the files were expanded to.

To start JMeter, use the following command:

jmeter

The JMeter window should appear and it should be similar to what is shown in Figure 11.

Using JMeter

In JMeter, the Test Plan contains one or more Thread Group. A Thread Group contains all the information necessary to run a test. The number of threads for the test and how fast they are started can be set inside the Thread

Group. A thread can be though of as a user. In the Thread Group, samplers can be added. A sampler is another name for a request. There are different types of samplers such as HTTP, FTP, JDBC, and more. A listener can be added to report the data and results of a test. Think of the listener as something that listens on the wire when the requests and responses go back and forth between the client and the server. Other configuration elements such as timers, cookie manager, pre-processor, and post-processor can be added. After adding and setting the elements in a Thread Group, select 'Start' from the 'Run' menu and the test will execute.

JMeter also provides a HTTP Proxy Recorder for recording all the requests made when browsing a web site. This eliminates the need to manually add each sampler to a Thread Group. Right-click on the Work Bench and add a Non-Test Element > HTTP Proxy Recorder. Set the Target Controller to either record the requests to the Work Bench or to an existing Thread Group. It is possible to add URL patterns to include or exclude. Click the 'Start' button when ready. Open any web browser and set the proxy to 'localhost' and use the same port as specified in JMeter HTTP Proxy Server (default: 8080). Then use the browser as usual. Requests should be recorded to JMeter. When done, revert the proxy setting in the browser and click 'Stop' in JMeter.

APPENDIX D – Installing and Using nmon for Linux

This section provides brief information on the installation and usage of nmon for Linux. More detailed documentation on nmon for Linux and its usage can be found on its website.

Installing nmon for Linux

No installation is necessary. Download and execute the single binary file for the underlying operating system and platform. The source code consists of one file that contains approximately 5,000 lines and one makefile exists if one wants to compile it.

Running nmon for Linux

Nmon for Linux can be started via the command-line interface. Open up a console or command prompt and browse to the directory that contains the nmon executable. Enter the name of the nmon executable to start it in interactive mode. Nmon for Linux offers many command-line options. Those are documented on the project website. The following command-line options were used for this project:

- -F <filename> : spreadsheet output format
- -t : include top processes in output
- -s <seconds> : between snapshots
- -c <number> : of refreshes
- -p : output the background process ID

The following nmon command was used in this project:

./nmon -F output.csv -t -s 2 -c 9999999 –p

APPENDIX E - Perl Script to Generate Test Data

```
#!/usr/bin/perl
use strict;

# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 12, 2010
#
# Description: This script generates a CSV file that can be used to create
# Moodle accounts via its 'Upload users' feature. For upload file format,
# see http://docs.moodle.org/en/Upload_users
# It also generates a CSV file that can be used as the CSV Data Set for
# dynamic data when load testing is performed with JMeter.
#
# Input:
#   - the 1st argument must be the number of users to generate
#   - any succeeding arguments are optional and must be in the form of key=value
# example: course1=CF101
#
# Output:
# A CSV file containing the following fields (and any additional fields
# specified on the command-line):
# username,password,firstname,lastname,email
# Another CSV file containing the 5 fields from above and another field
# consisting of a random string that can be used as input for any reason.

my ($moodle_csv_file, $jmeter_csv_file);
my ($number_of_users, $counter);
my ($username_prefix, $password_prefix,
      $firstname_prefix, $lastname_prefix, $email_domain);
my (@addfield, @addvalue);
my ($random_string, $random_string_length);

# settings
$moodle_csv_file = "moodle_accounts.csv";
$jmeter_csv_file = "jmeter_data_set.csv";
$username_prefix = "user";
$password_prefix = "password";
$firstname_prefix = "firstname";
$lastname_prefix = "lastname";
$email_domain = "example.org";
$random_string_length = 10;

# check command-line arguments
if ($#ARGV < 0 || $ARGV[0] !~ /^\d+$/) {
  die "The first command-line argument must be the # of users to generate.";
}
$number_of_users = shift(@ARGV);
if ($#ARGV >= 0) {
```

```
  foreach my $arg (@ARGV) {
    if ($arg =~ /(.*)=(.*)/) {
      push(@addfield, $1);
      push(@addvalue, $2);
    }
  }
}

# output CSV files
open(MOODLECSV, ">$moodle_csv_file") or die "failed to open $moodle_csv_file:
$!\n";
open(JMETERCSV, ">$jmeter_csv_file") or die "failed to open $jmeter_csv_file:
$!\n";
# print first record (moodle csv only)
print MOODLECSV "username,password,firstname,lastname,email";
foreach my $field (@addfield) { print MOODLECSV ",$field"; }
# print records
while ($counter != $number_of_users) {
  $counter += 1;
  my $record = "$username_prefix$counter,$password_prefix$counter,";
  $record .= "$firstname_prefix$counter,$lastname_prefix$counter,";
  $record .= "$username_prefix$counter\@$email_domain";
  print MOODLECSV "\n$record";
  foreach my $value (@addvalue) { print MOODLECSV ",$value"; }
  $random_string = &get_random_string($random_string_length);
  print JMETERCSV "$record,$username_prefix$counter-$random_string\n";
}
close(MOODLECSV) or die "failed to close $moodle_csv_file: $!\n";
close(JMETERCSV) or die "failed to close $jmeter_csv_file: $!\n";

sub get_random_string
{
  my $str_length = shift;
  my @chars = ('a'..'z','A'..'Z','0'..'9','_');
  my $random_string;

  foreach (1..$str_length) { $random_string .= $chars[rand @chars]; }
  return $random_string;
}
```

APPENDIX F – Selenium Scripts for Setting/Cleaning Up Test Environment

setup_for_selenium_tests.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: This script sets up Moodle for running Selenium tests.
# It does the following only when it is necessary:
#   - adds teacher account
#   - assigns teacher as course creator
#   - adds student account
#   - add course category

from selenium import selenium
import unittest, time, re

class AdminSetup(unittest.TestCase):

  admin_username = "admin"
  admin_password = "test123"
  teacher_username = "teacher1"
  teacher_password = "password"
  teacher_firstname = "Teacher"
  teacher_lastname = "One"
  teacher_email = "teacher1@example.org"
  teacher_name = teacher_firstname + " " + teacher_lastname
  student_username = "student1"
  student_password = "password"
  student_firstname = "Student"
  student_lastname = "One"
  student_name = student_firstname + " " + student_lastname
  student_email = "student1@example.org"
  course_category = "Computer Science"

  def setUp(self):
    self.verificationErrors = []
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
    self.selenium.start()

  def return_to_homepage(self):
    sel = self.selenium
    sel.click("link=XS")
    sel.wait_for_page_to_load("60000")

  def test_admin_setup(self):
    sel = self.selenium
    print
```

```
print "----------------------"

# weird, first login attempt does not work, clears the fields
sel.open("/moodle/login/index.php")
sel.type("username", self.admin_username)
sel.type("password", self.admin_password)
sel.click("//input[@value='Login']")
sel.wait_for_page_to_load("60000")
if sel.is_element_present("username"):
  sel.type("username", self.admin_username)
  sel.type("password", self.admin_password)
  sel.click("//input[@value='Login']")
  sel.wait_for_page_to_load("60000")

create_teacher_account = True
create_student_account = True

# check if accounts already exist
print "check if accounts exist already"
sel.click("link=Users")
sel.click("link=Accounts")
sel.click("link=Browse list of users")
sel.wait_for_page_to_load("60000")
if sel.is_element_present("link=" + self.teacher_name):
  print " teacher account exists"
  create_teacher_account = False
if sel.is_element_present("link=" + self.student_name):
  print " student account exists"
  create_student_account = False
self.return_to_homepage()

# add teacher account
if create_teacher_account:
  print "create teacher account"
  sel.click("link=Users")
  sel.click("link=Accounts")
  sel.click("link=Add a new user")
  sel.wait_for_page_to_load("60000")
  sel.type("id_username", self.teacher_username)
  sel.type("id_newpassword", self.teacher_password)
  sel.type("id_firstname", self.teacher_firstname)
  sel.type("id_lastname", self.teacher_lastname)
  sel.type("id_email", self.teacher_email)
  sel.type("id_city", "San Francisco")
  sel.select("id_country", "label=United States")
  sel.click("id_submitbutton")
  sel.wait_for_page_to_load("60000")
  try: self.failUnless(sel.is_element_present("link=" + self.teacher_name))
  except AssertionError, e: self.verificationErrors.append(str(e))
  self.return_to_homepage()
```

```
        # assign teacher as course creator
        print "assign teacher as course creator"
        sel.click("link=Users")
        sel.click("link=Permissions")
        sel.click("link=Assign system roles")
        sel.wait_for_page_to_load("60000")
        sel.click("link=Course creator")
        sel.wait_for_page_to_load("60000")
        sel.add_selection("addselect", "label=" + self.teacher_name + ", " +
self.teacher_email)
        sel.click("add")
        sel.wait_for_page_to_load("60000")
        sel.click("//input[@value='Assign roles in System']")
        sel.wait_for_page_to_load("60000")
        try: self.failUnless(sel.is_element_present("link=" + self.teacher_name))
        except AssertionError, e: self.verificationErrors.append(str(e))
        self.return_to_homepage()

      # add student account
      if create_student_account:
        print "create student account"
        sel.click("link=Users")
        sel.click("link=Accounts")
        sel.click("link=Add a new user")
        sel.wait_for_page_to_load("60000")
        sel.type("id_username", self.student_username)
        sel.type("id_newpassword", self.student_password)
        sel.type("id_firstname", self.student_firstname)
        sel.type("id_lastname", self.student_lastname)
        sel.type("id_email", self.student_email)
        sel.type("id_city", "San Francisco")
        sel.select("id_country", "label=United States")
        sel.click("id_submitbutton")
        sel.wait_for_page_to_load("60000")
        try: self.failUnless(sel.is_element_present("link=" + self.student_name))
        except AssertionError, e: self.verificationErrors.append(str(e))
        self.return_to_homepage()

      # add course category
      sel.click("link=Courses")
      sel.click("link=Add/edit courses")
      sel.wait_for_page_to_load("60000")
      if sel.is_element_present("link=" + self.course_category):
        print "course category exists"
      else:
        print "add course category"
      sel.click("//input[@value='Add new category']")
      sel.wait_for_page_to_load("60000")
      sel.type("id_name", self.course_category)
      sel.click("id_submitbutton")
      sel.wait_for_page_to_load("60000")
      try: self.failUnless(sel.is_text_present("No courses in this category"))
      except AssertionError, e: self.verificationErrors.append(str(e))
```

```
    self.return_to_homepage()

    def tearDown(self):
      self.selenium.stop()
      self.assertEqual([], self.verificationErrors)

if __name__ == "__main__":
  unittest.main()
```

setup_for_jmeter_tests.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: This script sets up Moodle for running JMeter tests.
# It does the following only when it is necessary:
#   - adds teacher account
#   - assigns teacher as course creator
#   - add course category
#   - restore OLPC sample course
#   - upload users via CSV

from selenium import selenium
import unittest, time, re

class SetupForJmeter(unittest.TestCase):

  admin_username = "admin"
  admin_password = "test123"
  teacher_username = "teacher1"
  teacher_password = "password"
  teacher_firstname = "Teacher"
  teacher_lastname = "One"
  teacher_email = "teacher1@example.org"
  teacher_name = teacher_firstname + " " + teacher_lastname
  course_category = "Computer Science"
  course_full_name = "Dummy Course"
  course_short_name = "DC001"

  zip_file_name = "olpc_sample_course.zip"
  path_for_zip_file = "C:\\Documents and Settings\\btran\\Desktop\\Tests\\" +
zip_file_name
  path_for_csv_file = "C:\\Documents and
Settings\\btran\\Desktop\\Tests\\moodle_accounts.csv"
  sample_user = "firstname1 lastname1"

  def setUp(self):
    self.verificationErrors = []
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
```

```
  self.selenium.start()

def return_to_homepage(self):
  sel = self.selenium
  sel.click("link=XS")
  sel.wait_for_page_to_load("60000")

def return_to_course_page(self, name):
  sel = self.selenium
  sel.click("link=" + name)
  sel.wait_for_page_to_load("60000")

def test_setup_for_jmeter(self):
  sel = self.selenium
  print

  # weird, oftentimes first login attempt does not work, clears the fields
  print "ADMIN"
  sel.open("/moodle/login/index.php")
  sel.type("username", self.admin_username)
  sel.type("password", self.admin_password)
  sel.click("//input[@value='Login']")
  sel.wait_for_page_to_load("60000")
  if sel.is_element_present("username"):
    sel.type("username", self.admin_username)
    sel.type("password", self.admin_password)
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
  try: self.failUnless(sel.is_text_present("You are logged in as"))
  except AssertionError, e: self.verificationErrors.append(str(e))

  create_teacher_account = True

  # check if accounts already exist
  print " - check if account exist already"
  sel.click("link=Users")
  sel.click("link=Accounts")
  sel.click("link=Browse list of users")
  sel.wait_for_page_to_load("60000")
  if sel.is_element_present("link=" + self.teacher_name):
    print " - teacher account exists"
    create_teacher_account = False
  self.return_to_homepage()

  # add teacher account
  if create_teacher_account:
    print " - create teacher account"
    sel.click("link=Users")
    sel.click("link=Accounts")
    sel.click("link=Add a new user")
    sel.wait_for_page_to_load("60000")
    sel.type("id_username", self.teacher_username)
    sel.type("id_newpassword", self.teacher_password)
```

```
        sel.type("id_firstname", self.teacher_firstname)
        sel.type("id_lastname", self.teacher_lastname)
        sel.type("id_email", self.teacher_email)
        sel.type("id_city", "San Francisco")
        sel.select("id_country", "label=United States")
        sel.click("id_submitbutton")
        sel.wait_for_page_to_load("60000")
        try: self.failUnless(sel.is_element_present("link=" + self.teacher_name))
        except AssertionError, e: self.verificationErrors.append(str(e))
        self.return_to_homepage()

        # assign teacher as course creator
        print " - assign teacher as course creator"
        sel.click("link=Users")
        sel.click("link=Permissions")
        sel.click("link=Assign system roles")
        sel.wait_for_page_to_load("60000")
        sel.click("link=Course creator")
        sel.wait_for_page_to_load("60000")
        sel.add_selection("addselect", "label=" + self.teacher_name + ", " +
self.teacher_email)
        sel.click("add")
        sel.wait_for_page_to_load("60000")
        sel.click("//input[@value='Assign roles in System']")
        sel.wait_for_page_to_load("60000")
        try: self.failUnless(sel.is_element_present("link=" + self.teacher_name))
        except AssertionError, e: self.verificationErrors.append(str(e))
        self.return_to_homepage()

    # add course category
    sel.click("link=Courses")
    sel.click("link=Add/edit courses")
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("link=" + self.course_category):
      print " - course category exists"
    else:
      print " - add course category"
      sel.click("//input[@value='Add new category']")
      sel.wait_for_page_to_load("60000")
      sel.type("id_name", self.course_category)
      sel.click("id_submitbutton")
      sel.wait_for_page_to_load("60000")
      try: self.failUnless(sel.is_text_present("No courses in this category"))
      except AssertionError, e: self.verificationErrors.append(str(e))
      self.return_to_homepage()

    # log out
    sel.click("link=Logout")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("Login to the site"))
    except AssertionError, e: self.verificationErrors.append(str(e))
```

```
# log in as teacher
print
print "TEACHER"
sel.type("username", self.teacher_username)
sel.type("password", self.teacher_password)
sel.click("//input[@value='Login']")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present("My courses"))
except AssertionError, e: self.verificationErrors.append(str(e))

# create course
if sel.is_element_present("link=" + self.course_full_name):
  print " - dummy course exists"
  sel.click("link=" + self.course_full_name)
else:
  print " - create dummy course"
  sel.click("link=Courses")
  sel.click("link=Add/edit courses")
  sel.click("link=" + self.course_category)
  sel.wait_for_page_to_load("60000")
  sel.click("xpath=//input[@value='Add a new course']")
  sel.wait_for_page_to_load("60000")
  sel.type("id_fullname", self.course_full_name)
  sel.type("id_shortname", self.course_short_name)
  sel.click("id_submitbutton")
  sel.wait_for_page_to_load("60000")
  self.assertEqual("Edit course settings", sel.get_title())
  try: self.failUnless(sel.is_text_present(self.course_full_name))
  except AssertionError, e: self.verificationErrors.append(str(e))
  try: self.failUnless(sel.is_text_present(self.course_short_name))
  except AssertionError, e: self.verificationErrors.append(str(e))
  sel.click("//input[@value='Click here to enter your course']")
  sel.wait_for_page_to_load("60000")
  try: self.failUnless(sel.is_text_present("Weekly outline"))
  except AssertionError, e: self.verificationErrors.append(str(e))

# restore course from backup/zip file
name = "Testing OLPC XS"
if sel.is_element_present("link=" + name):
  print " - olpc sample course exists"
  sel.click("link=" + name)
else:
  print " - restore olpc sample course"
  sel.click("link=Restore")
  sel.wait_for_page_to_load("60000")
  sel.click("//input[@value='Upload a file']")
  sel.wait_for_page_to_load("60000")
  sel.type("userfile", self.path_for_zip_file)
  sel.click("save")
  sel.wait_for_page_to_load("60000")
  try: self.failUnless(sel.is_text_present("File uploaded successfully"))
  except AssertionError, e: self.verificationErrors.append(str(e))
  sel.click("link=Restore")
```

```
  sel.wait_for_page_to_load("480000")
  sel.click("//input[@value='Yes']")
  sel.wait_for_page_to_load("480000")
  sel.click("//input[@value='Continue']")
  sel.wait_for_page_to_load("480000")
  sel.click("//input[@value='Continue']")
  sel.wait_for_page_to_load("480000")
  sel.click("//input[@value='Restore this course now!']")
  sel.wait_for_page_to_load("480000")
  try: self.failUnless(sel.is_text_present("Restore completed successfully"))
  except AssertionError, e: self.verificationErrors.append(str(e))
  sel.click("//input[@value='Continue']")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present("Weekly outline"))
except AssertionError, e: self.verificationErrors.append(str(e))
print " (" + sel.get_location() + ")"

name = "CSC895"
thread_title = "Discuss OLPC here"
print " - get olpc forum link/id"
sel.click("link=OLPC forum")
sel.wait_for_page_to_load("60000")
print " (" + sel.get_location() + ")"
print " - get discussion thread link/id"
sel.click("link=" + thread_title)
sel.wait_for_page_to_load("60000")
print " (" + sel.get_location() + ")"
print " - get reply link/id"
sel.click("link=Reply")
sel.wait_for_page_to_load("60000")
print " (" + sel.get_location() + ")"
self.return_to_course_page(name)

print " - get resource link/id"
sel.click("link=Text Page Resource")
sel.wait_for_page_to_load("480000")
print " (" + sel.get_location() + ")"
self.return_to_course_page(name)

print " - get homework upload link/id"
sel.click("link=Homework Assignment")
sel.wait_for_page_to_load("60000")
print " (" + sel.get_location() + ")"
self.return_to_course_page(name)

print " - get quiz link/id"
sel.click("link=Chapter 6 Quiz")
sel.wait_for_page_to_load("60000")
print " (" + sel.get_location() + ")"
self.return_to_course_page(name)

# log out
sel.click("link=Logout")
```

```
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("Login to the site"))
    except AssertionError, e: self.verificationErrors.append(str(e))

    # log in as admin
    print
    print "ADMIN"
    sel.type("username", self.admin_username)
    sel.type("password", self.admin_password)
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")

    # upload users via CSV file
    sel.click("link=Users")
    sel.click("link=Accounts")
    sel.click("link=Browse list of users")
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("link=" + self.sample_user):
      print " - no need to upload CSV file"
    else:
      print " - upload users via CSV file"
      self.return_to_homepage()
      sel.click("link=Users")
      sel.click("link=Accounts")
      sel.click("link=Upload users")
      sel.wait_for_page_to_load("60000")
      sel.type("id_userfile", self.path_for_csv_file)
      sel.click("id_submitbutton")
      sel.wait_for_page_to_load("60000")
      sel.type("id_city", "San Francisco")
      sel.click("id_submitbutton")
      sel.wait_for_page_to_load("480000")
      for i in range(60):
        try:
          if sel.is_text_present("Users created:"):
            break
        except: pass
        time.sleep(1)
      else: self.fail("time out")
      sel.click("//input[@value='Continue']")
      sel.wait_for_page_to_load("240000")
      sel.click("link=Browse list of users")
      sel.wait_for_page_to_load("60000")
      try: self.failUnless(sel.is_element_present("link=" + self.sample_user))
      except AssertionError, e: self.verificationErrors.append(str(e))
    self.return_to_homepage()

  def tearDown(self):
    self.selenium.stop()

if __name__ == "__main__":
  unittest.main()
```

clean_forum.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: This script restores the forum to its original state.

from selenium import selenium
import unittest, time, re

class clean_forum(unittest.TestCase):

  username = "teacher1"
  password = "password"

  def setUp(self):
    self.verificationErrors = []
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
    self.selenium.start()

  def test_clean_forum(self):
    sel = self.selenium
    print "TEACHER - clean forum"
    sel.open("/moodle/login/index.php")
    sel.type("username", self.username)
    sel.type("password", self.password)
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("username"):
      sel.type("username", self.username)
      sel.type("password", self.password)
      sel.click("//input[@value='Login']")
      sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("My courses"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    sel.click("link=Testing OLPC XS")
    sel.wait_for_page_to_load("60000")

    thread_title = "Discuss OLPC here"
    sel.click("link=OLPC forum")
    sel.wait_for_page_to_load("60000")
    sel.click("link=" + thread_title)
    sel.wait_for_page_to_load("60000")
    sel.click("link=Delete")
    sel.wait_for_page_to_load("60000")
    sel.click("//input[@value='Yes']")
    sel.wait_for_page_to_load("60000")
    sel.click("//input[@value='Add a new discussion topic']")
    sel.wait_for_page_to_load("60000")
    sel.type("id_subject", thread_title)
```

```
    sel.type("id_message", "Please post here.")
    sel.click("id_submitbutton")
    sel.wait_for_page_to_load("60000")
    for i in range(60):
      try:
        if thread_title == sel.get_text("link=" + thread_title): break
      except: pass
      time.sleep(1)
    else: self.fail("time out")
    print " - get discussion thread link/id"
    sel.click("link=" + thread_title)
    sel.wait_for_page_to_load("60000")
    print " (" + sel.get_location() + ")"
    print " - get reply link/id"
    sel.click("link=Reply")
    sel.wait_for_page_to_load("60000")
    print " (" + sel.get_location() + ")"

  def tearDown(self):
    self.selenium.stop()
    self.assertEqual([], self.verificationErrors)

if __name__ == "__main__":
  unittest.main()
```

cleanup.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: This script cleans up Moodle.

from selenium import selenium
import unittest, time, re

class AdminCleanup(unittest.TestCase):

  admin_username = "admin"
  admin_password = "test123"
  teacher_name = "Teacher One"
  student_name = "Student One"
  course_category = "Computer Science"

  def setUp(self):
    self.verificationErrors = []
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
    self.selenium.start()

  def return_to_homepage(self):
```

```
    sel = self.selenium
    sel.click("link=XS")
    sel.wait_for_page_to_load("60000")

  def test_admin_cleanup(self):
    sel = self.selenium
    print
    print "----------------------"

    # weird, first login attempt does not work, clears the fields
    sel.open("/moodle/login/index.php")
    sel.type("username", self.admin_username)
    sel.type("password", self.admin_password)
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("username"):
      sel.type("username", self.admin_username)
      sel.type("password", self.admin_password)
      sel.click("//input[@value='Login']")
      sel.wait_for_page_to_load("60000")

    # remove ALL accounts EXCEPT OLPC Admin
    print "remove all accounts EXCEPT OLPC Admin"
    sel.click("link=Users")
    sel.click("link=Accounts")
    sel.click("link=Bulk user actions")
    sel.wait_for_page_to_load("60000")
    if not sel.is_text_present("All users (1)"):
      sel.click("id_addall")
      sel.wait_for_page_to_load("60000")
      sel.add_selection("id_susers", "label=Admin OLPC")
      sel.click("id_removesel")
      sel.wait_for_page_to_load("60000")
      sel.select("id_action", "label=Delete")
      sel.click("id_doaction")
      sel.wait_for_page_to_load("60000")
      try: self.failUnless(sel.is_text_present("Confirmation"))
      except AssertionError, e: self.verificationErrors.append(str(e))
      sel.click("//input[@value='Yes']")
      sel.wait_for_page_to_load("120000")
      for i in range(60):
        try:
          if "All users (1)" == sel.get_text("//select[@id='id_ausers']/option[1]"):
break
        except: pass
        time.sleep(1)
      else: self.fail("time out")
    self.return_to_homepage()

    # remove course category and its courses (xpath: course category must be 2nd
on the list)
    sel.click("link=Courses")
    sel.click("link=Add/edit courses")
```

```
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("link=" + self.course_category):
      print "remove course category"
      sel.click("//td[@id='middle-column']/div/table/tbody/tr[3]/td[3]/a[2]/img")
      sel.wait_for_page_to_load("60000")
      try: self.failUnless(sel.is_text_present("Delete category: " +
self.course_category))
      except AssertionError, e: self.verificationErrors.append(str(e))
      if sel.is_element_present("id=id_fulldelete"):
        sel.select("id_fulldelete", "label=Delete all - cannot be undone")
      sel.click("id_submitbutton")
      sel.wait_for_page_to_load("60000")
      try: self.failUnless(sel.is_text_present("Deleted course category " +
self.course_category))
      except AssertionError, e: self.verificationErrors.append(str(e))
    self.return_to_homepage()

  def tearDown(self):
    self.selenium.stop()
    self.assertEqual([], self.verificationErrors)

if __name__ == "__main__":
  unittest.main()
```

APPENDIX G - Selenium Tests

tests_suite_olpc.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: Selenium test suite for XS Moodle

import unittest
import setup_for_selenium_tests
import tests_homepage
import tests_teacher
import tests_student
import cleanup
import sys

def suite():
  return unittest.TestSuite((\
    unittest.makeSuite(setup_for_selenium_tests.AdminSetup),
    unittest.makeSuite(tests_homepage.TestHomepage),
    unittest.makeSuite(tests_teacher.TestTeacher),
    unittest.makeSuite(tests_student.TestStudent),
    unittest.makeSuite(cleanup.AdminCleanup),
  ))

if __name__ == "__main__":
  result = unittest.TextTestRunner(verbosity=2).run(suite())
  sys.exit(not result.wasSuccessful())
```

tests_homepage.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: Selenium tests for XS Moodle - Homepage

from selenium import selenium
import unittest, time, re

class TestHomepage(unittest.TestCase):

  def setUp(self):
    self.verificationErrors = []
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
```

```
    self.selenium.start()

  def test_homepage(self):
    sel = self.selenium
    print
    print "-----------------------"

    # XSF001
    print "testHomepage - check for username/password fields, login button"
    sel.open("/moodle/login/index.php")
    sel.type("username", "baduser")
    sel.type("password", "wrongpassword")
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_element_present("username"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_element_present("password"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_text_present("Invalid login"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_element_present("//input[@value='Login']"))
    except AssertionError, e: self.verificationErrors.append(str(e))

    # XSF002
    print "testInvalidLogin - log in with invalid credentials, check error message"
    sel.open("/moodle/login/index.php")
    sel.type("username", "baduser")
    sel.type("password", "wrongpassword")
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_element_present("username"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_element_present("password"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_text_present("Invalid login"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_element_present("//input[@value='Login']"))
    except AssertionError, e: self.verificationErrors.append(str(e))

  def tearDown(self):
    self.selenium.stop()
    self.assertEqual([], self.verificationErrors)

if __name__ == "__main__":
  unittest.main()
```

tests_teacher.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: Selenium tests for XS Moodle - Teacher

from selenium import selenium
import unittest, time, re

class TestTeacher(unittest.TestCase):

  username = "teacher1"
  password = "password"
  course_category = "Computer Science"
  course_full_name = "Applied Research Project"
  course_short_name = "CSC895"
  course_new_full_name = "Testing OLPC XS"

  def setUp(self):
    self.verificationErrors = []
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
    self.selenium.start()

  def return_to_course_page(self):
    sel = self.selenium
    sel.click("link=" + self.course_short_name)
    sel.wait_for_page_to_load("60000")

  def test_teacher(self):
    sel = self.selenium
    print
    print "----------------------"

    # XSF003 - weird, 1st login attempt doesn't work, clears the fields instead
    print "testTeacherLogin - log in as teacher"
    sel.open("/moodle/login/index.php")
    sel.type("username", self.username)
    sel.type("password", self.password)
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("username"):
      sel.type("username", self.username)
      sel.type("password", self.password)
      sel.click("//input[@value='Login']")
      sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("My courses"))
    except AssertionError, e: self.verificationErrors.append(str(e))

    # XSF004
```

```
print "testCreateNewCourse - create new course"
sel.click("link=Courses")
sel.click("link=Add/edit courses")
sel.click("link=" + self.course_category)
sel.wait_for_page_to_load("60000")
sel.click("xpath=//input[@value='Add a new course']")
sel.wait_for_page_to_load("60000")
sel.type("id_fullname", self.course_full_name)
sel.type("id_shortname", self.course_short_name)
sel.click("id_submitbutton")
sel.wait_for_page_to_load("60000")
self.assertEqual("Edit course settings", sel.get_title())
try: self.failUnless(sel.is_text_present(self.course_full_name))
except AssertionError, e: self.verificationErrors.append(str(e))
try: self.failUnless(sel.is_text_present(self.course_short_name))
except AssertionError, e: self.verificationErrors.append(str(e))
sel.click("//input[@value='Click here to enter your course']")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present("Weekly outline"))
except AssertionError, e: self.verificationErrors.append(str(e))

# XSF005
print "testCourseSettings - change course full name"
sel.click("link=Settings")
sel.wait_for_page_to_load("60000")
sel.type("id_summary", "")
sel.type("id_fullname", self.course_new_full_name)
sel.click("id_submitbutton")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present(self.course_new_full_name))
except AssertionError, e: self.verificationErrors.append(str(e))

subject = "Test Thread"
message = "this is a test message."

# XSF006
print "testAddTopicNewsForum - add new topic in News Forum"
sel.click("link=News forum")
sel.wait_for_page_to_load("60000")
sel.click("//input[@value='Add a new topic']")
sel.wait_for_page_to_load("60000")
sel.type("id_subject", subject)
sel.type("id_message", message)
sel.click("id_submitbutton")
sel.wait_for_page_to_load("60000")
for i in range(60):
  try:
    if "General news and announcements" == sel.get_text("intro"): break
  except: pass
  time.sleep(1)
else: self.fail("time out")
sel.click("link=" + subject)
sel.wait_for_page_to_load("60000")
```

```
try: self.failUnless(sel.is_text_present(message))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

# XSF007
print "testDeleteTopicNewsForum - delete topic in News Forum"
sel.click("link=News forum")
sel.wait_for_page_to_load("60000")
sel.click("link=" + subject)
sel.wait_for_page_to_load("60000")
sel.click("link=Delete")
sel.wait_for_page_to_load("60000")
sel.click("//input[@value='Yes']")
sel.wait_for_page_to_load("60000")
for i in range(60):
  try:
    if "General news and announcements" == sel.get_text("intro"): break
  except: pass
  time.sleep(1)
else: self.fail("time out")
try: self.failUnless(sel.is_text_present("(No news has been posted yet)"))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

event_name = "Test Event"

# XSF008
print "testAddNewEvent - add new event to calendar"
sel.click("link=New Event")
sel.wait_for_page_to_load("60000")
sel.click("//input[@value=' OK ']")
sel.wait_for_page_to_load("60000")
sel.type("eventname", event_name)
sel.click("//input[@value='Save changes']")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_element_present("//input[@value='Export calendar']"))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()
try: self.failUnless(sel.is_text_present("Upcoming Events"))
except AssertionError, e: self.verificationErrors.append(str(e))
try: self.failUnless(sel.is_element_present("link=" + event_name))
except AssertionError, e: self.verificationErrors.append(str(e))

# XSF009
print "testDeleteEvent - delete event from calendar"
sel.click("link=" + event_name)
sel.wait_for_page_to_load("60000")
sel.click("//img[@alt='Delete event']")
sel.wait_for_page_to_load("60000")
sel.click("//input[@value=' Delete ']")
sel.wait_for_page_to_load("60000")
for i in range(60):
  try:
```

```
    if sel.is_element_present("//input[@value='Export calendar']"): break
  except: pass
  time.sleep(1)
else: self.fail("time out")
self.return_to_course_page()
try: self.failUnless(sel.is_text_present("Upcoming Events"))
except AssertionError, e: self.verificationErrors.append(str(e))
try: self.failUnless(sel.is_text_present("no upcoming"))
except AssertionError, e: self.verificationErrors.append(str(e))

text_name = "OLPC Mission Statement"
text_body = ("\"to provide educational opportunities for the world's poorest "
                "children by providing each children with a rugged, low-cost, "
                "low-power connected laptop with content and software designed "
                "for collaborative, joyful, self-empowered learning.\"")

# XSF010
print "testAddResource - compose a text page"
sel.click("//input[@value='Turn editing on']")
sel.wait_for_page_to_load("60000")
sel.select("ressection0_jump", "label=Compose a text page")
sel.wait_for_page_to_load("60000")
for i in range(60):
  try:
    if sel.is_text_present("Adding a new Resource"): break
  except: pass
  time.sleep(1)
else: self.fail("time out")
sel.type("id_name", text_name)
sel.type("id_alltext", text_body)
sel.click("id_submitbutton2")
sel.wait_for_page_to_load("60000")
sel.click("//input[@value='Turn editing off']")
sel.wait_for_page_to_load("60000")
sel.click("link=" + text_name)
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present(text_body))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

forum_name = "OLPC Forum"
forum_intro = "This forum is for discussing OLPC topics"

# XSF011
print "testAddActivity - create new forum"
sel.click("//input[@value='Turn editing on']")
sel.wait_for_page_to_load("60000")
sel.select("section0_jump", "label=Forum")
for i in range(60):
  try:
    if sel.is_text_present("Adding a new Forum"): break
  except: pass
  time.sleep(1)
```

```
    else: self.fail("time out")
    sel.type("id_name", forum_name)
    sel.type("id_intro", forum_intro)
    sel.click("id_submitbutton2")
    sel.wait_for_page_to_load("60000")
    sel.click("//input[@value='Turn editing off']")
    sel.wait_for_page_to_load("60000")
    sel.click("link=" + forum_name)
    for i in range(60):
      try:
        if forum_intro == sel.get_text("intro"): break
      except: pass
      time.sleep(1)
    else: self.fail("time out")
    self.return_to_course_page()

    # Log out
    sel.click("link=Logout")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("Login to the site"))
    except AssertionError, e: self.verificationErrors.append(str(e))

  def tearDown(self):
    self.selenium.stop()
    self.assertEqual([], self.verificationErrors)

if __name__ == "__main__":
  unittest.main()
```

tests_student.py

```
# Copyright 2010 by Benjamin Tran. All Rights Reserved.
#
# Author: Benjamin Tran
# Date: April 25, 2010
#
# Description: Selenium tests for XS Moodle - Student

from selenium import selenium
import unittest, time, re

class TestStudent(unittest.TestCase):

  username = "student1"
  password = "password"
  student_name = "Student One"
  course_short_name = "CSC895"
  course_full_name = "Testing OLPC XS"

  def setUp(self):
    self.verificationErrors = []
```

```
    self.selenium = selenium("localhost", 4444, "*chrome",
"http://schoolserver.example.org/")
    self.selenium.start()

  def return_to_course_page(self):
    sel = self.selenium
    sel.click("link=" + self.course_short_name)
    sel.wait_for_page_to_load("60000")

  def test_student(self):
    sel = self.selenium
    print
    print "----------------------"

    # XSF012 - weird, 1st login attempt doesn't work, clears the fields instead
    print "testStudentLogin - log in as student"
    sel.open("/moodle/login/index.php")
    sel.type("username", self.username)
    sel.type("password", self.password)
    sel.click("//input[@value='Login']")
    sel.wait_for_page_to_load("60000")
    if sel.is_element_present("username"):
      sel.type("username", self.username)
      sel.type("password", self.password)
      sel.click("//input[@value='Login']")
      sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("My courses"))
    except AssertionError, e: self.verificationErrors.append(str(e))

    # XSF013
    print "testEnrollCourse - enroll in a course"
    sel.click("link=" + self.course_full_name)
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("You are about to enrol yourself as a
member of this course."))
    except AssertionError, e: self.verificationErrors.append(str(e))
    sel.click("//input[@value='Yes']")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("Weekly outline"))
    except AssertionError, e: self.verificationErrors.append(str(e))

    # XSF014
    print "testParticipantsPage - access Participants page"
    sel.click("link=Participants")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("All participants:"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    self.return_to_course_page()

    # XSF015
    print "testNewsForum - access News forum"
    sel.click("link=News forum")
    sel.wait_for_page_to_load("60000")
```

```
try: self.failUnless(sel.is_text_present("General news and announcements"))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

text_name = "OLPC Mission Statement"
text_body = ("\"to provide educational opportunities for the world's poorest "
             "children by providing each children with a rugged, low-cost, "
             "low-power connected laptop with content and software designed "
             "for collaborative, joyful, self-empowered learning.\"")

# XSF016
print "testTextPage - access text page"
sel.click("link=" + text_name + " Resource")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present(text_body))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

forum_name = "OLPC Forum"
forum_intro = "This forum is for discussing OLPC topics"
subject = "OLPC Rocks"
message = "this is a test"

# XSF017
print "testPostForum - access forum and post topic"
sel.click("link=" + forum_name)
sel.wait_for_page_to_load("60000")
sel.click("//input[@value='Add a new discussion topic']")
sel.wait_for_page_to_load("60000")
sel.type("id_subject", subject)
sel.type("id_message", message)
sel.click("id_submitbutton")
sel.wait_for_page_to_load("60000")
for i in range(60):
  try:
    if sel.is_text_present(forum_intro): break
  except: pass
  time.sleep(1)
else: self.fail("time out")
try: self.failUnless(sel.is_element_present("link=" + subject))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

# XSF018
print "testGradesPage - access grades page"
sel.click("link=Grades")
sel.wait_for_page_to_load("60000")
try: self.failUnless(sel.is_text_present("User report - " + self.student_name))
except AssertionError, e: self.verificationErrors.append(str(e))
self.return_to_course_page()

# XSF019
print "testProfilePage - access profile page"
```

```
    sel.click("link=Profile")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("Edit profile"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_text_present("First access:"))
    except AssertionError, e: self.verificationErrors.append(str(e))

    profile_description = "O-L-P-C"

    # XSF020
    print "testEditProfile - add profile description"
    sel.click("link=Edit profile")
    sel.wait_for_page_to_load("60000")
    sel.type("id_description", profile_description)
    sel.click("id_submitbutton")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present(profile_description))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_text_present("Edit profile"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    try: self.failUnless(sel.is_text_present("First access:"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    self.return_to_course_page()

    # log out
    sel.click("link=Logout")
    sel.wait_for_page_to_load("60000")
    try: self.failUnless(sel.is_text_present("Login to the site"))
    except AssertionError, e: self.verificationErrors.append(str(e))
    def tearDown(self):
    self.selenium.stop()
    self.assertEqual([], self.verificationErrors)

if __name__ == "__main__":
  unittest.main()
```

APPENDIX H – System Information of Test Machines

XO Laptop

Disk /dev/mmcblk0: 8166 MB, 8166309888 bytes
212 heads, 51 sectors/track, 1475 cylinders
Units = cylinders of 10812 * 512 = 5535744 bytes

processor: 0
vendor_id: AuthenticAMD
cpu family: 5
model: 10
model name: Geode(TM) Integrated Processor by AMD PCS
stepping: 2
cpu MHz: 431.235
cache size: 128 KB

MemTotal: 235728 kB

FitPC

Disk /dev/sda: 64.1 GB, 64105742336 bytes
255 heads, 63 sectors/track, 7793 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-0: 53.2 GB, 53217329152 bytes
255 heads, 63 sectors/track, 6469 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-1: 2147 MB, 2147483648 bytes
255 heads, 63 sectors/track, 261 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

processor: 0
vendor_id: AuthenticAMD
cpu family: 5
model: 10
model name: Geode(TM) Integrated Processor by AMD PCS
stepping: 2
cpu MHz: 499.900
cache size: 128 KB

MemTotal: 494248 kB

FitPC2

Disk /dev/sda: 64.0 GB, 64023257088 bytes
255 heads, 63 sectors/track, 7783 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-0: 53.1 GB, 53116665856 bytes
255 heads, 63 sectors/track, 6457 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-1: 2147 MB, 2147483648 bytes
255 heads, 63 sectors/track, 261 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

processor: 0
vendor_id: GenuineIntel
cpu family: 6
model: 28
model name: Intel(R) Atom(TM) CPU Z510 @ 1.10GHz
stepping: 2
cpu MHz: 1100.000
cache size: 512 KB

MemTotal: 1024920 kB

OLPCorps SolidLogic

Disk /dev/sda: 160.0 GB, 160041885696 bytes
255 heads, 63 sectors/track, 19457 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-0: 149.1 GB, 149149450240 bytes
255 heads, 63 sectors/track, 18133 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-1: 2147 MB, 2147483648 bytes
255 heads, 63 sectors/track, 261 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

processor: 0
vendor_id: CentaurHauls
cpu family: 6
model: 13
model name: VIA C7 Processor 1000MHz
stepping: 0
cpu MHz: 797.976
cache size: 128 KB

MemTotal: 968304 kB

P4 Desktop

Disk /dev/sda: 40.0 GB, 40020664320 bytes
255 heads, 63 sectors/track, 4865 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-0: 29.1 GB, 29125246976 bytes
255 heads, 63 sectors/track, 3540 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-1: 2147 MB, 2147483648 bytes
255 heads, 63 sectors/track, 261 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

processor: 0
vendor_id: GenuineIntel
cpu family: 15
model: 2
model name: Intel(R) Pentium(R) 4 CPU 3.06GHz
stepping: 7
cpu MHz: 3066.564
cache size: 512 KB

processor: 1
vendor_id: GenuineIntel
cpu family: 15
model: 2
model name: Intel(R) Pentium(R) 4 CPU 3.06GHz
stepping: 7
cpu MHz: 3066.564
cache size: 512 KB

MemTotal: 1033648 kB

Dell Xeon Workstation

Disk /dev/sda: 250.0 GB, 250000000000 bytes
255 heads, 63 sectors/track, 30394 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-0: 239.1 GB, 239108882432 bytes
255 heads, 63 sectors/track, 29069 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/dm-1: 2147 MB, 2147483648 bytes
255 heads, 63 sectors/track, 261 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

processor: 0
vendor_id: GenuineIntel
cpu family: 15
model: 4
model name: Intel(R) Xeon(TM) CPU 2.80GHz
stepping: 1
cpu MHz: 2792.995
cache size: 1024 KB

processor: 1
vendor_id: GenuineIntel
cpu family: 15
model: 4
model name: Intel(R) Xeon(TM) CPU 2.80GHz
stepping: 1
cpu MHz: 2792.995
cache size: 1024 KB

processor: 2
vendor_id: GenuineIntel
cpu family: 15
model: 4
model name: Intel(R) Xeon(TM) CPU 2.80GHz
stepping: 1
cpu MHz: 2792.995
cache size: 1024 KB

processor: 3
vendor_id: GenuineIntel
cpu family: 15
model: 4
model name: Intel(R) Xeon(TM) CPU 2.80GHz
stepping: 1
cpu MHz: 2792.995
cache size: 1024 KB

MemTotal: 3113380 kB

APPENDIX I – Results from Ramp-Up Time Experiment

The following are results from an experiment performed to verify the effects of what a larger ramp-up time would have on the system. The 'Log In' test (test case ID XSP001) was executed on the FitPC2 machine and the OLPCorps SolidLogic machine. Results confirmed that if we increase the ramp-up time, the server can then handle more requests at a slower average response time with no errors.

FitPC2

Threads	Ramp-Up Time	Avg Response Time	Error Rate
200	60s	66711ms	0.00%
225	60s	70633ms	2.67%
250	60s	69566ms	10.80%

Threads	Ramp-Up Time	Avg Response Time	Error Rate
200	120s	22960ms	0.00%
225	120s	34120ms	0.00%
250	120s	40995ms	0.00%
275	120s	51694ms	0.00%
300	120s	59167ms	0.00%
325	120s	70707ms	1.23%
350	120s	72012ms	8.57%

OLPCorps SolidLogic

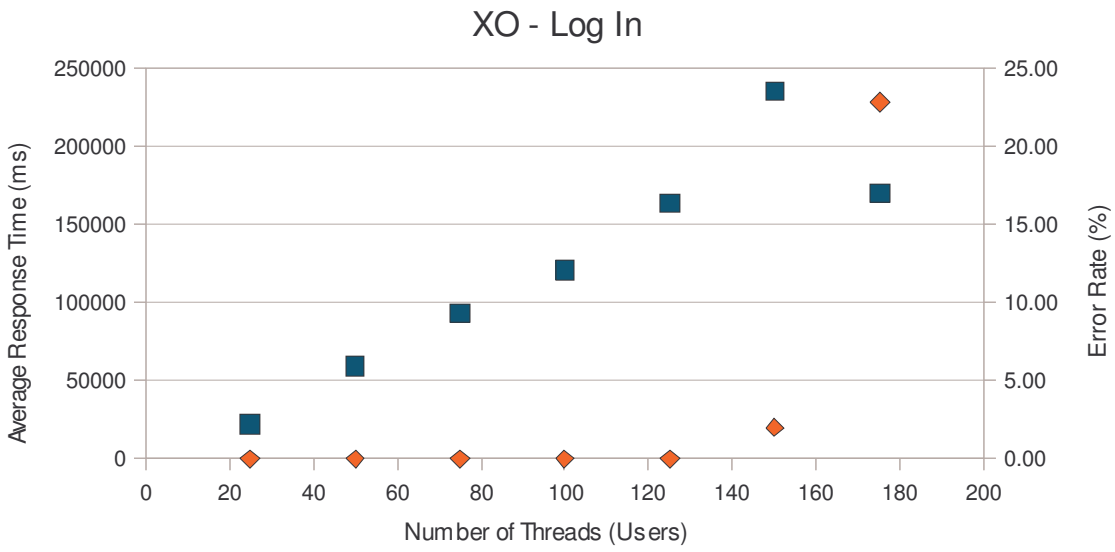
Threads	Ramp-Up Time	Avg Response Time	Error Rate
225	60s	65528ms	0.00%
250	60s	64776ms	6.80%
275	60s	63921ms	15.27%

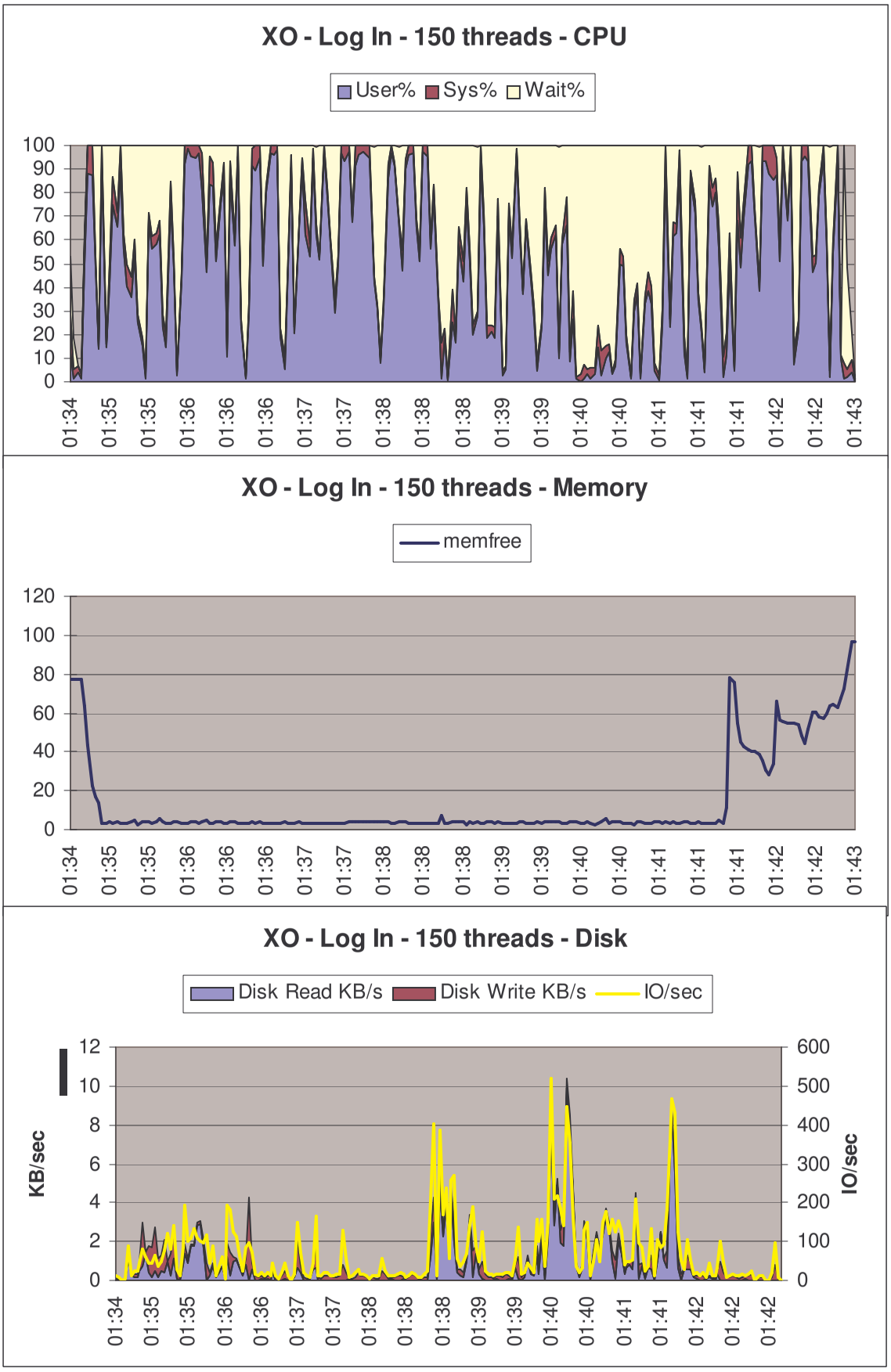
Threads	Ramp-Up Time	Avg Response Time	Error Rate
225	120s	30289ms	0.00%
250	120s	39590ms	0.00%
275	120s	44772ms	0.00%
300	120s	52334ms	0.00%
325	120s	63917ms	1.85%
350	120s	66111ms	9.14%

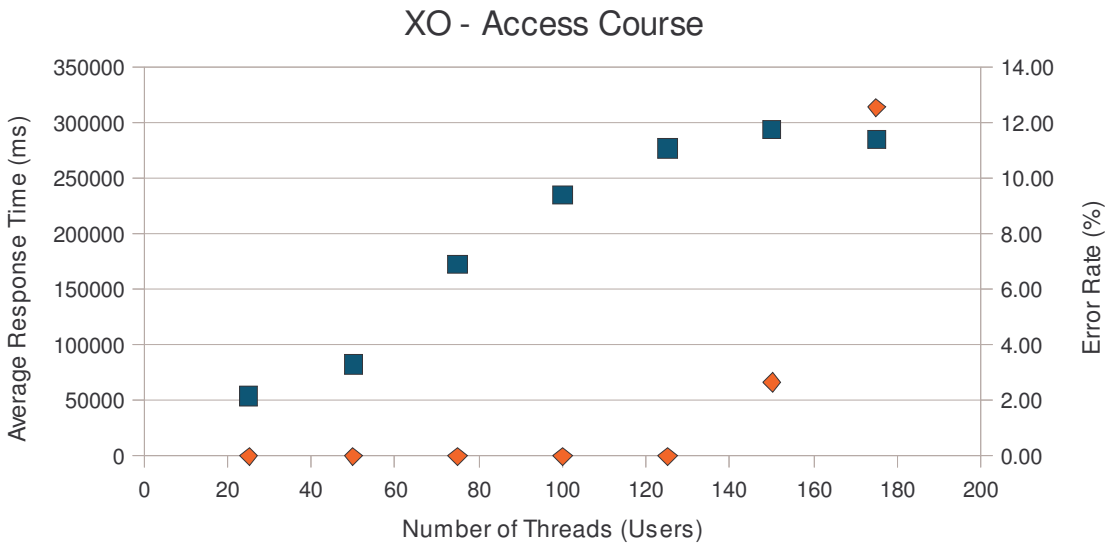
APPENDIX J – Graphs from Experimental Results

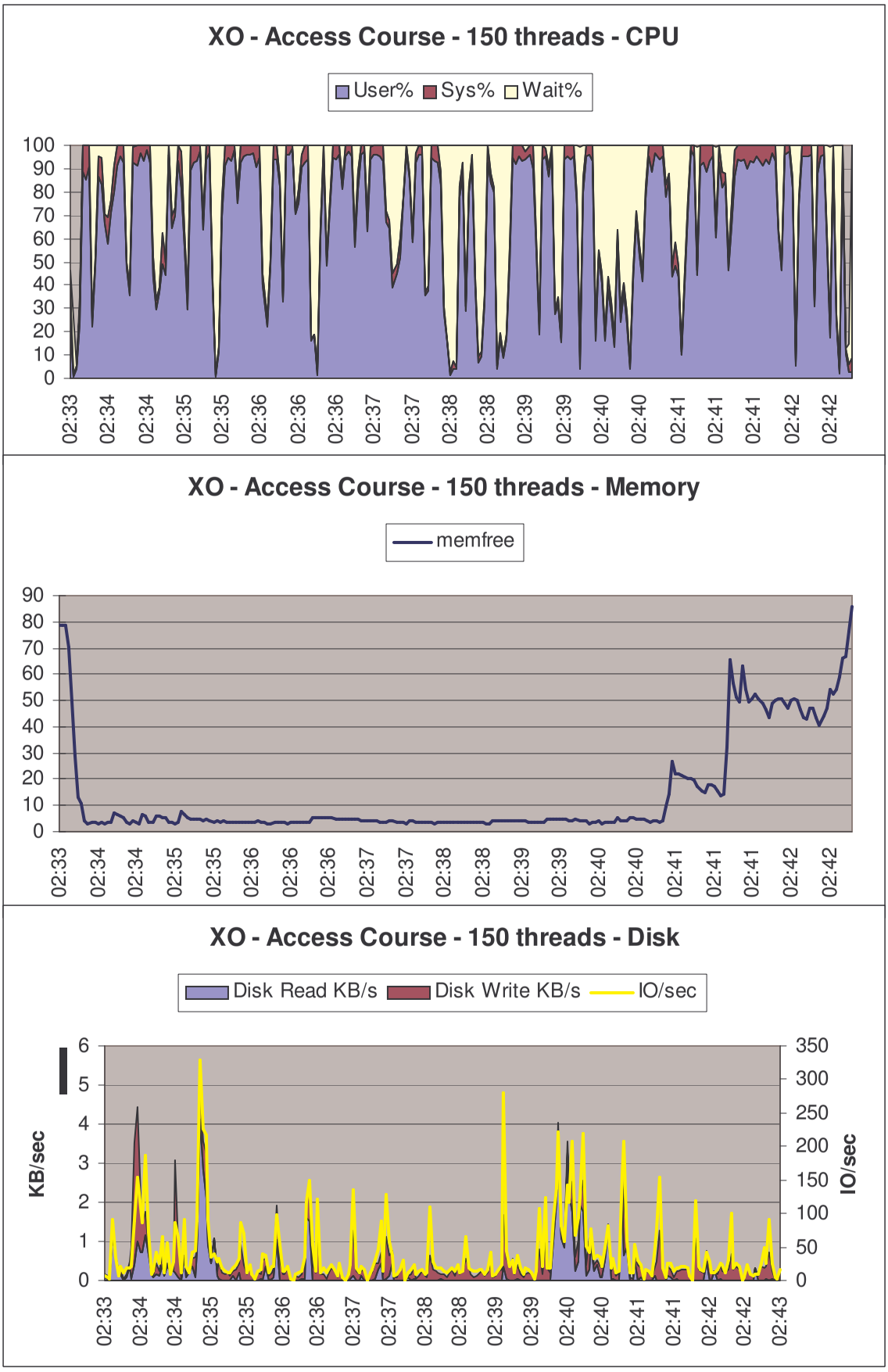
The following pages contain graphs from running the JMeter tests against the following five machines: XO, FitPC, FitPC2, P4 Desktop, and Dell Xeon Workstation. The graphs appear in the following order: number of threads/users versus total average response time, CPU utilization, free memory, and disk data rate. The graphs of the server resources correspond to the test run where the number of threads caused the error rate to become non-zero.

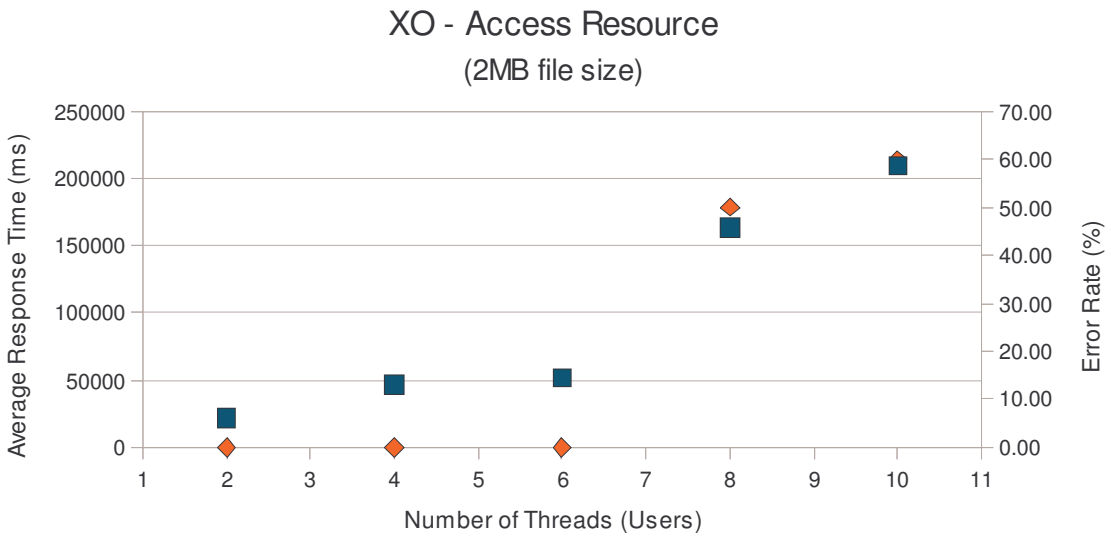
J.1 XO Laptop

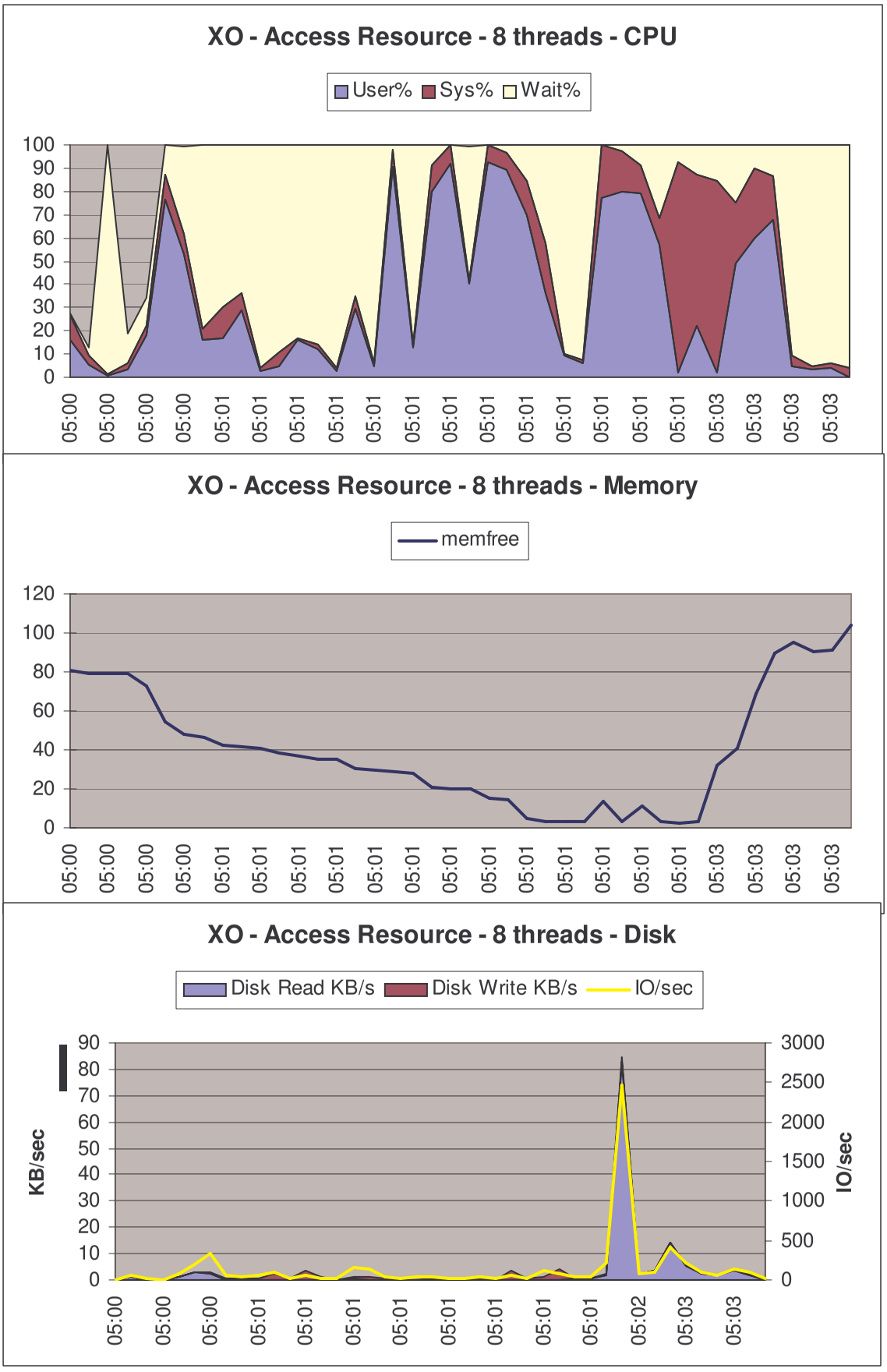

XO - Log In
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
250000
200000
150000
100000
50000
0
25.00
20.00
15.00
10.00
5.00
0.00
0
20
40
60
80
100
120
140
160
180
200

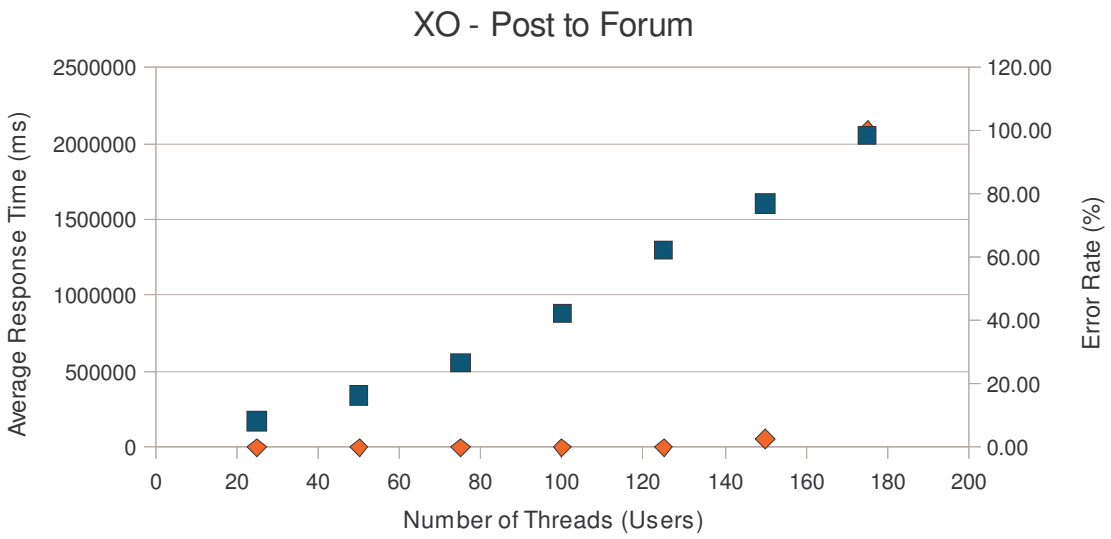

XO - Log In - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
01:34
01:35
01:35
01:36
01:36
01:36
01:37
01:37
01:38
01:38
01:38
01:39
01:39
01:40
01:40
01:41
01:41
01:41
01:42
01:42
01:43
XO - Log In - 150 threads - Memory
memfree
120
100
80
60
40
20
0
01:34
01:35
01:35
01:36
01:36
01:36
01:37
01:37
01:38
01:38
01:38
01:39
01:39
01:40
01:40
01:41
01:41
01:41
01:42
01:42
01:43
XO - Log In - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
12
10
8
6
4
2
0
IO/sec
600
500
400
300
200
100
0
01:34
01:35
01:35
01:36
01:36
01:37
01:37
01:38
01:38
01:38
01:39
01:39
01:40
01:40
01:41
01:41
01:42
01:42
01:42

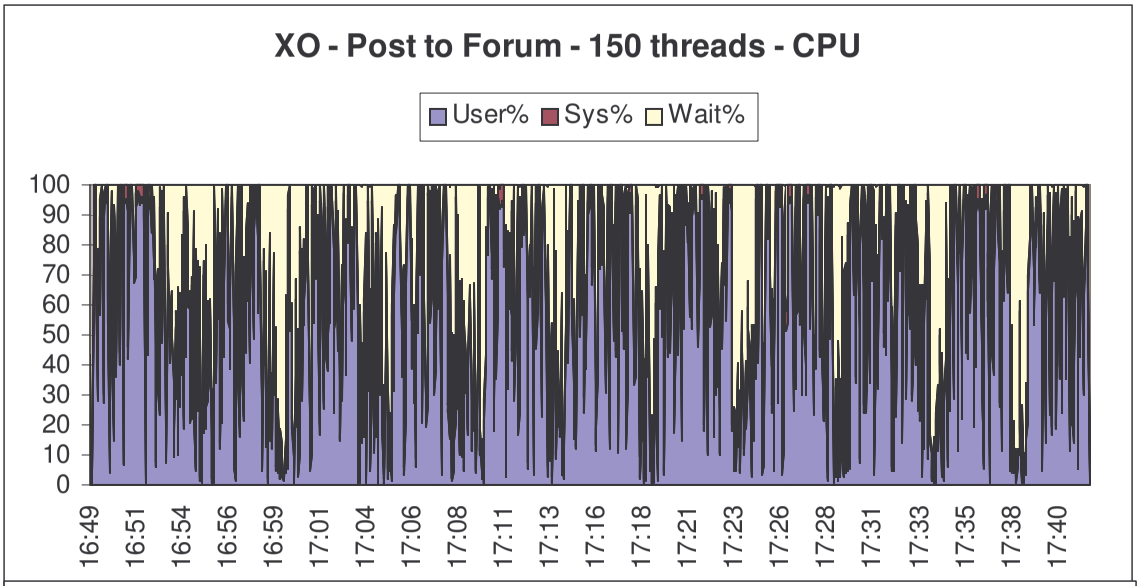

XO - Access Course
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
350000
300000
250000
200000
150000
100000
50000
0
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
20
40
60
80
100
120
140
160
180
200

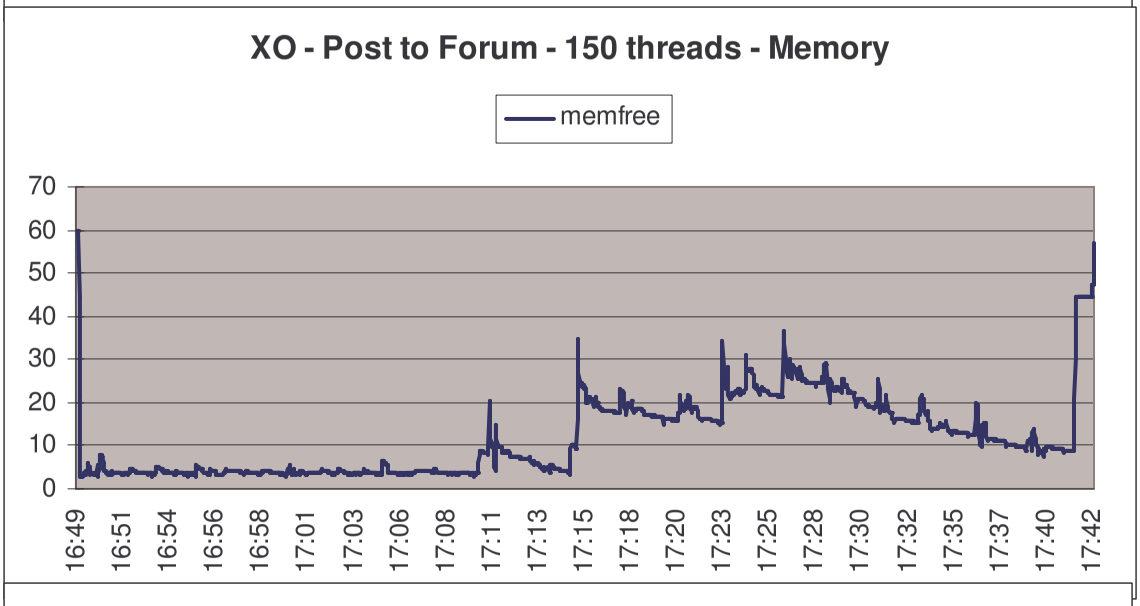

XO - Access Course - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
02:33
02:34
02:34
02:35
02:35
02:36
02:36
02:36
02:37
02:37
02:38
02:38
02:39
02:39
02:40
02:40
02:41
02:41
02:41
02:42
02:42
XO - Access Course - 150 threads - Memory
memfree
90
80
70
60
50
40
30
20
10
0
02:33
02:34
02:34
02:35
02:35
02:36
02:36
02:36
02:37
02:37
02:38
02:38
02:39
02:39
02:40
02:40
02:41
02:41
02:41
02:42
02:42
XO - Access Course - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
6
5
4
3
2
1
0
IO/sec
350
300
250
200
150
100
50
0
02:33
02:34
02:34
02:35
02:35
02:36
02:36
02:37
02:37
02:38
02:38
02:39
02:39
02:40
02:40
02:41
02:41
02:42
02:42
02:43

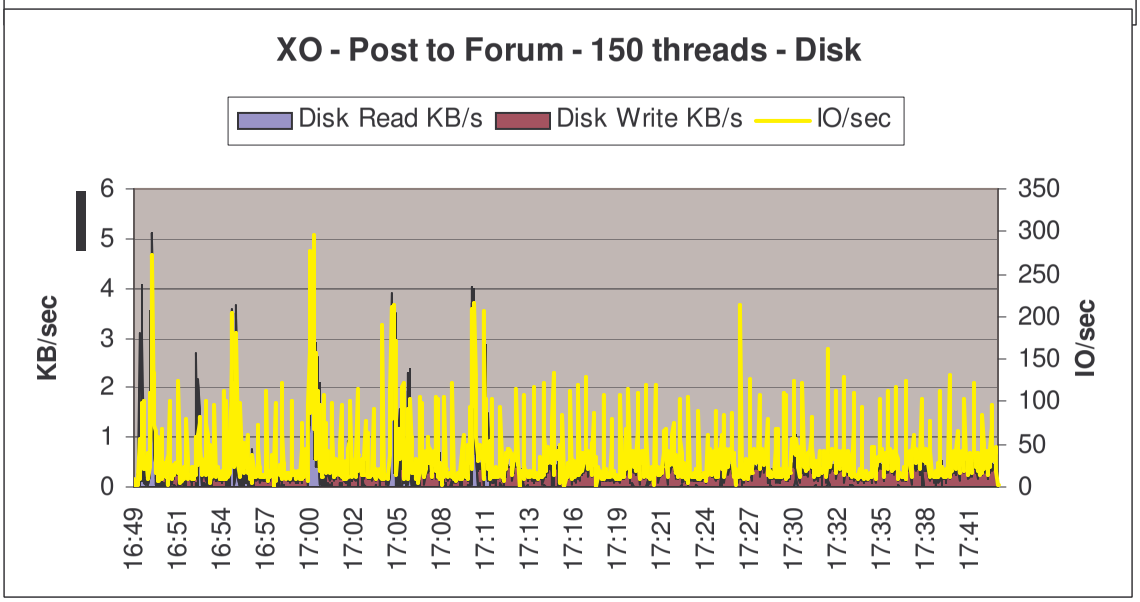

XO - Access Resource
(2MB file size)
Average Response Time (ms)
250000
200000
150000
100000
50000
0
Error Rate (%)
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
1
2
3
4
5
6
7
8
9
10
11
Number of Threads (Users)

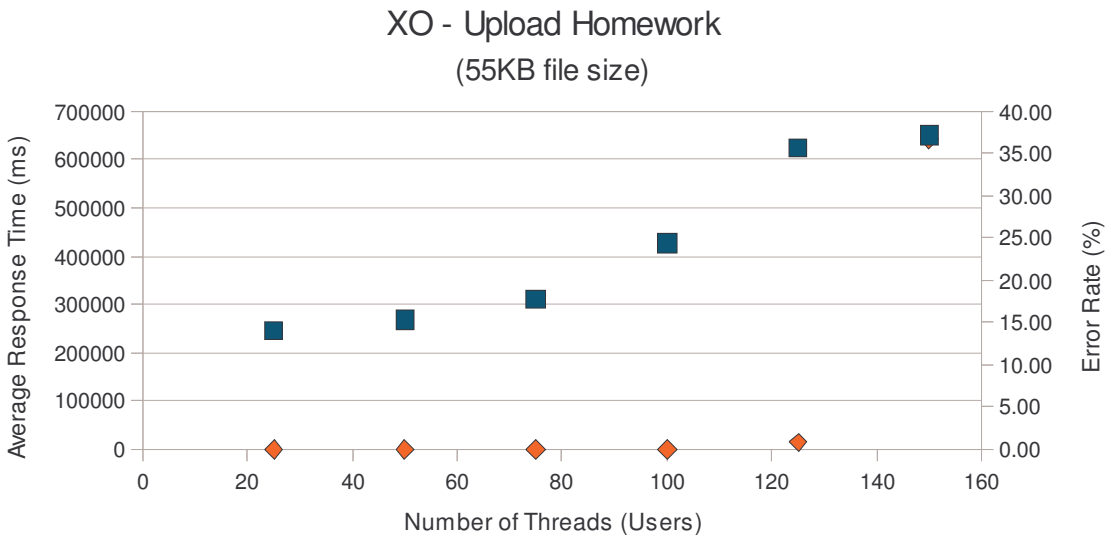

XO - Access Resource - 8 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
05:00
05:00
05:00
05:00
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:03
05:03
05:03
05:03
XO - Access Resource - 8 threads - Memory
memfree
120
100
80
60
40
20
0
05:00
05:00
05:00
05:00
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:03
05:03
05:03
05:03
XO - Access Resource - 8 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
90
80
70
60
50
40
30
20
10
0
3000
2500
2000
1500
1000
500
0
IO/sec
05:00
05:00
05:00
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:01
05:02
05:03
05:03

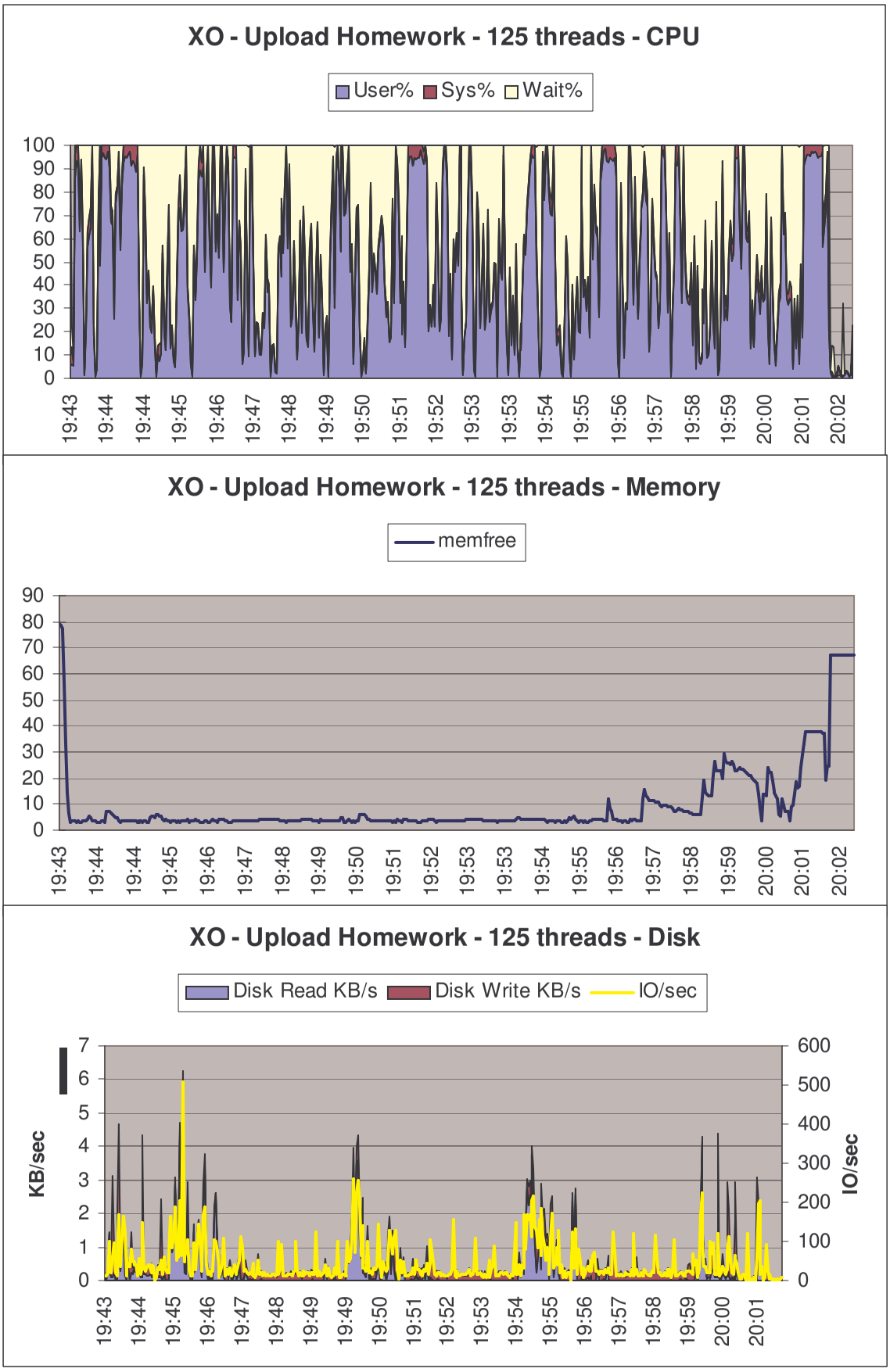

XO - Post to Forum
Average Response Time (ms)
2500000
2000000
1500000
1000000
500000
0
120.00
100.00
80.00
60.00
40.00
20.00
0.00
Error Rate (%)
0
20
40
60
80
100
120
140
160
180
200
Number of Threads (Users)

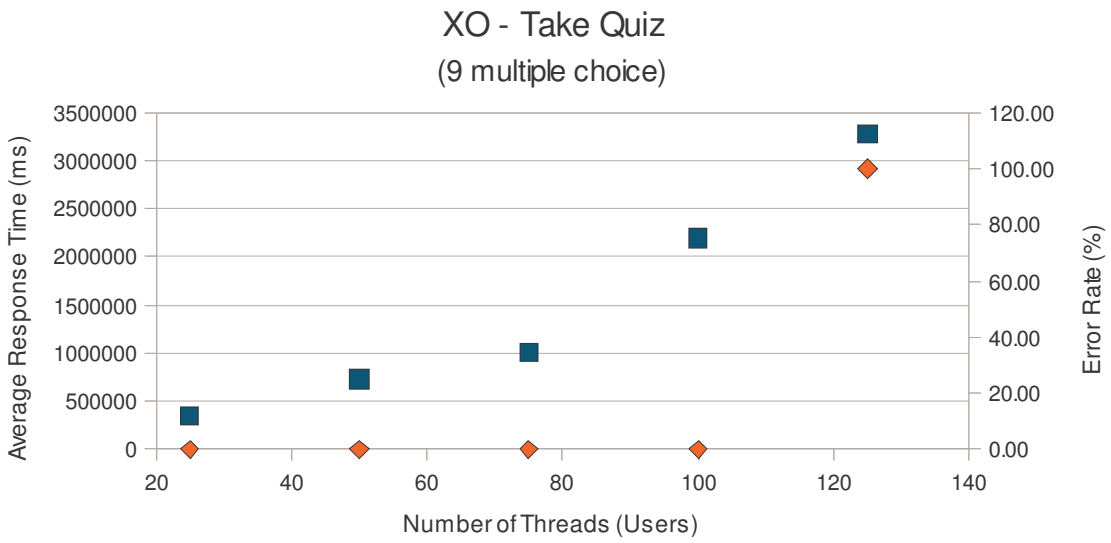

XO - Post to Forum - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
16:49
16:51
16:54
16:56
16:59
17:01
17:04
17:06
17:08
17:11
17:13
17:16
17:18
17:21
17:23
17:26
17:28
17:31
17:33
17:35
17:38
17:40

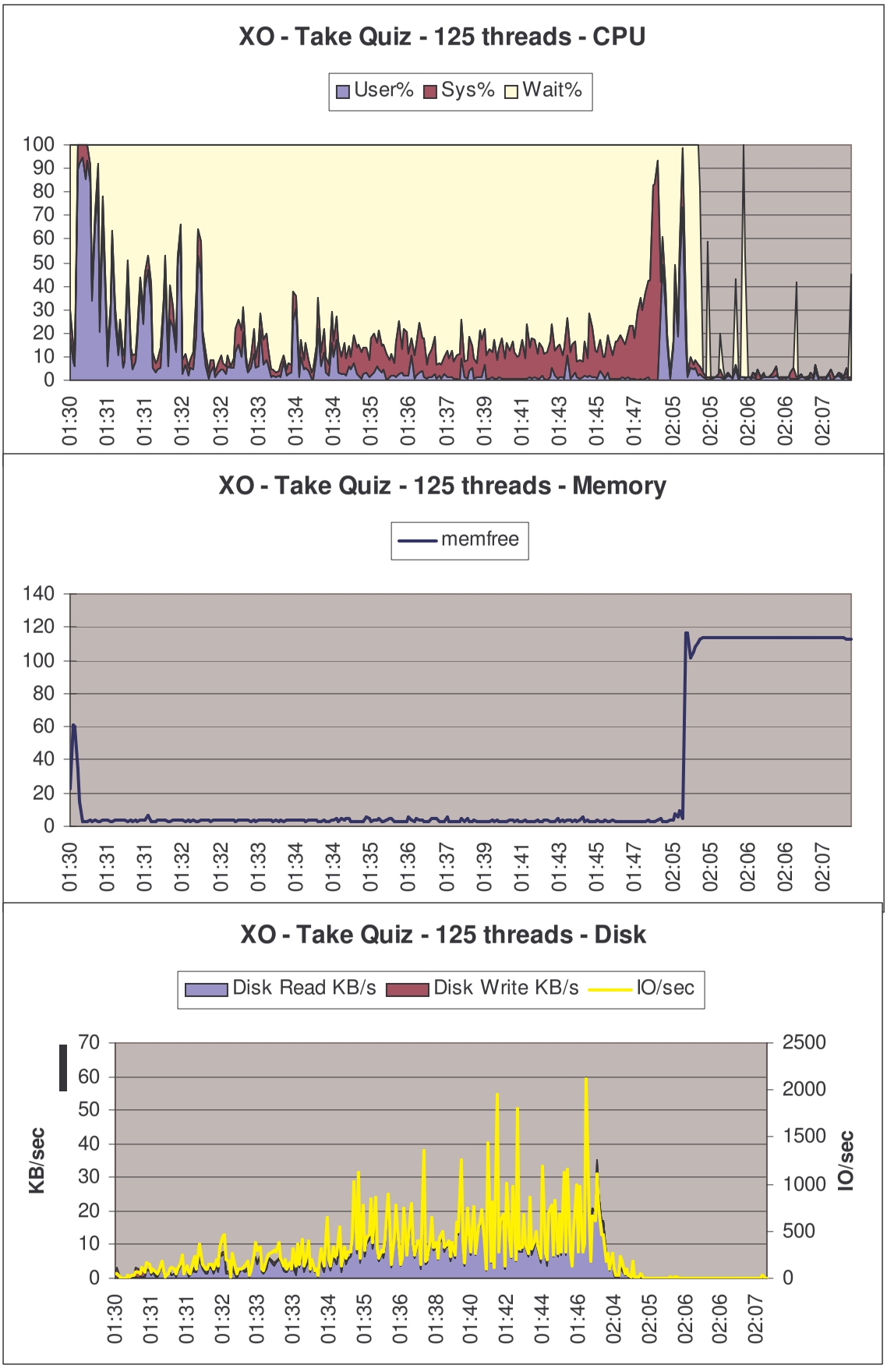

XO - Post to Forum - 150 threads - Memory
memfree
70
60
50
40
30
20
10
0
16:49
16:51
16:54
16:56
16:58
17:01
17:03
17:06
17:08
17:11
17:13
17:15
17:18
17:20
17:23
17:25
17:28
17:30
17:32
17:35
17:37
17:40
17:42


XO - Post to Forum - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
6
5
4
3
2
1
0
350
300
250
200
150
100
50
0
16:49
16:51
16:54
16:57
17:00
17:02
17:05
17:08
17:11
17:13
17:16
17:19
17:21
17:24
17:27
17:30
17:32
17:35
17:38
17:41

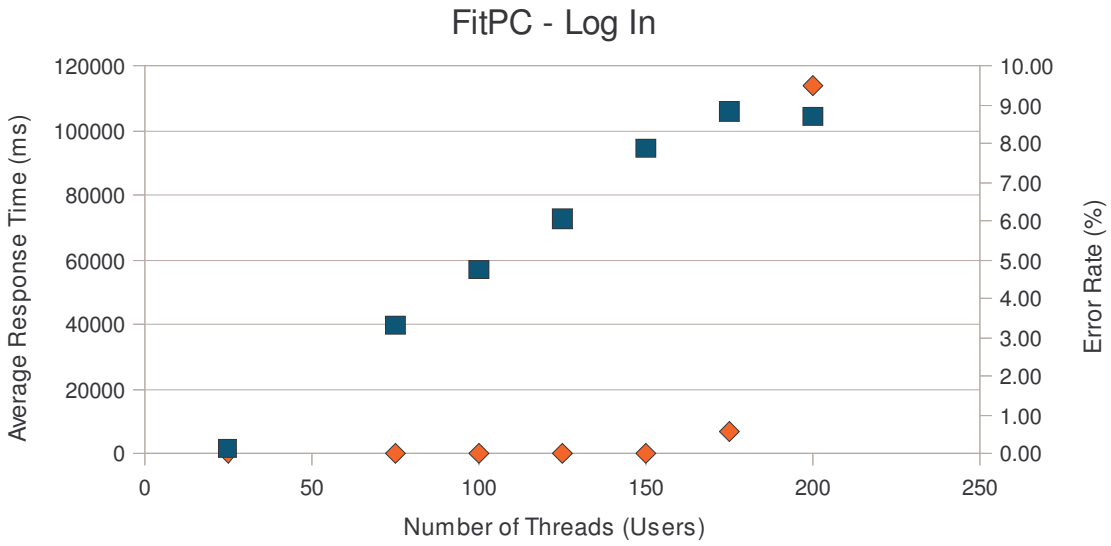

XO - Upload Homework
(55KB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
700000
600000
500000
400000
300000
200000
100000
0
40.00
35.00
30.00
25.00
20.00
15.00
10.00
5.00
0.00
0
20
40
60
80
100
120
140
160

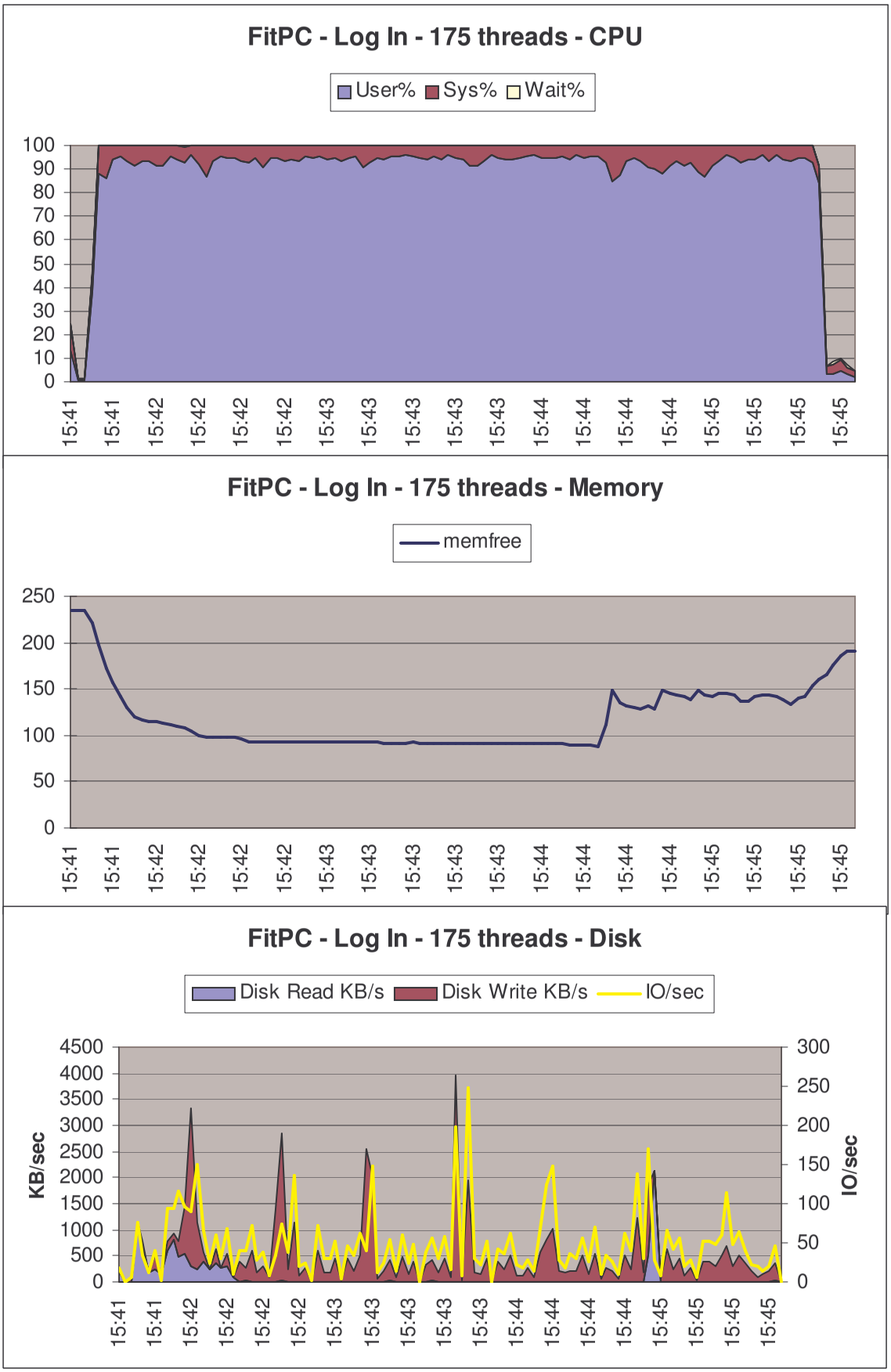

XO - Upload Homework - 125 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
19:43
19:44
19:44
19:45
19:46
19:47
19:48
19:49
19:50
19:51
19:52
19:53
19:53
19:54
19:55
19:56
19:57
19:58
19:59
20:00
20:01
20:02
XO - Upload Homework - 125 threads - Memory
memfree
90
80
70
60
50
40
30
20
10
0
19:43
19:44
19:44
19:45
19:46
19:47
19:48
19:49
19:50
19:51
19:52
19:53
19:53
19:54
19:55
19:56
19:57
19:58
19:59
20:00
20:01
20:02
XO - Upload Homework - 125 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
7
6
5
4
3
2
1
0
600
500
400
300
200
100
0
IO/sec
19:43
19:44
19:45
19:46
19:47
19:48
19:49
19:49
19:50
19:51
19:52
19:53
19:54
19:55
19:56
19:57
19:58
19:59
20:00
20:01

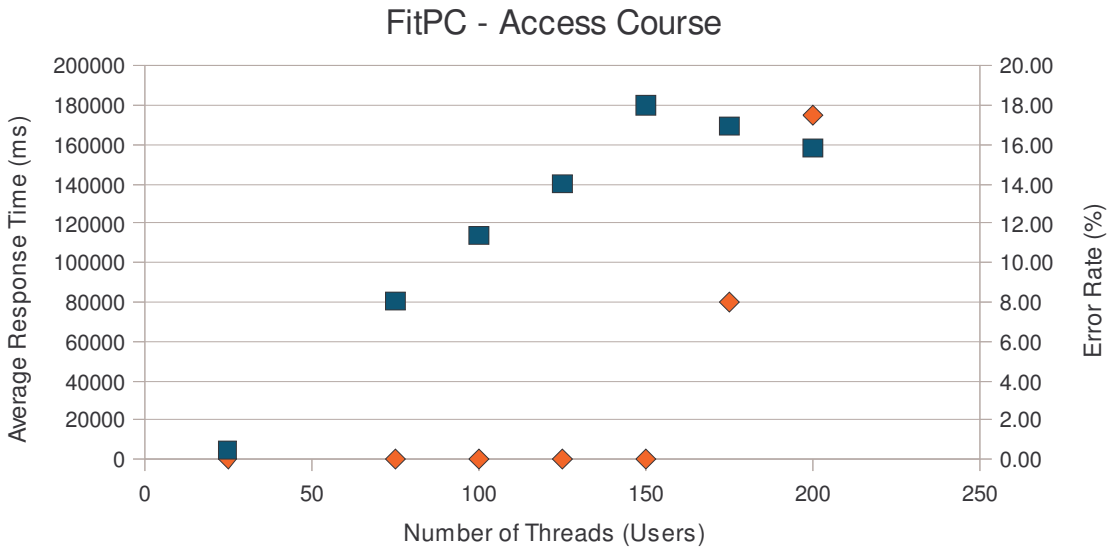

XO - Take Quiz
(9 multiple choice)
Average Response Time (ms)
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
120.00
100.00
80.00
60.00
40.00
20.00
0.00
Error Rate (%)
20
40
60
80
100
120
140
Number of Threads (Users)

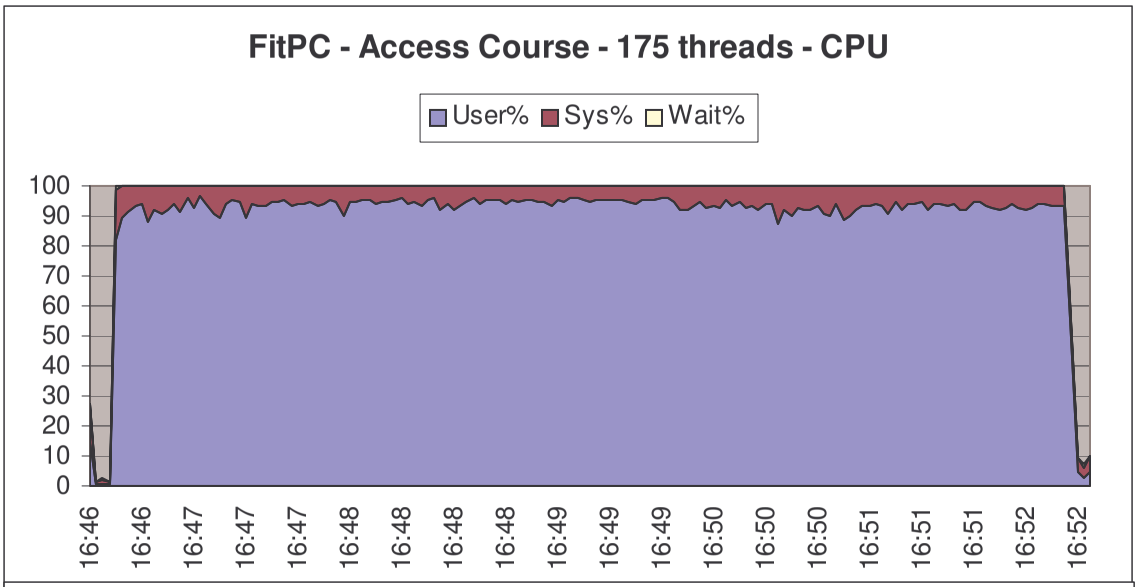

XO - Take Quiz - 125 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
01:30
01:31
01:31
01:32
01:32
01:33
01:34
01:34
01:35
01:36
01:37
01:39
01:41
01:43
01:45
01:47
02:05
02:05
02:06
02:06
02:07
XO - Take Quiz - 125 threads - Memory
memfree
140
120
100
80
60
40
20
0
01:30
01:31
01:31
01:32
01:32
01:33
01:34
01:34
01:35
01:36
01:37
01:39
01:41
01:43
01:45
01:47
02:05
02:05
02:06
02:06
02:07
XO - Take Quiz - 125 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
70
60
50
40
30
20
10
0
IO/sec
2500
2000
1500
1000
500
0
01:30
01:31
01:31
01:32
01:33
01:33
01:34
01:35
01:36
01:38
01:40
01:42
01:44
01:46
02:04
02:05
02:06
02:06
02:07

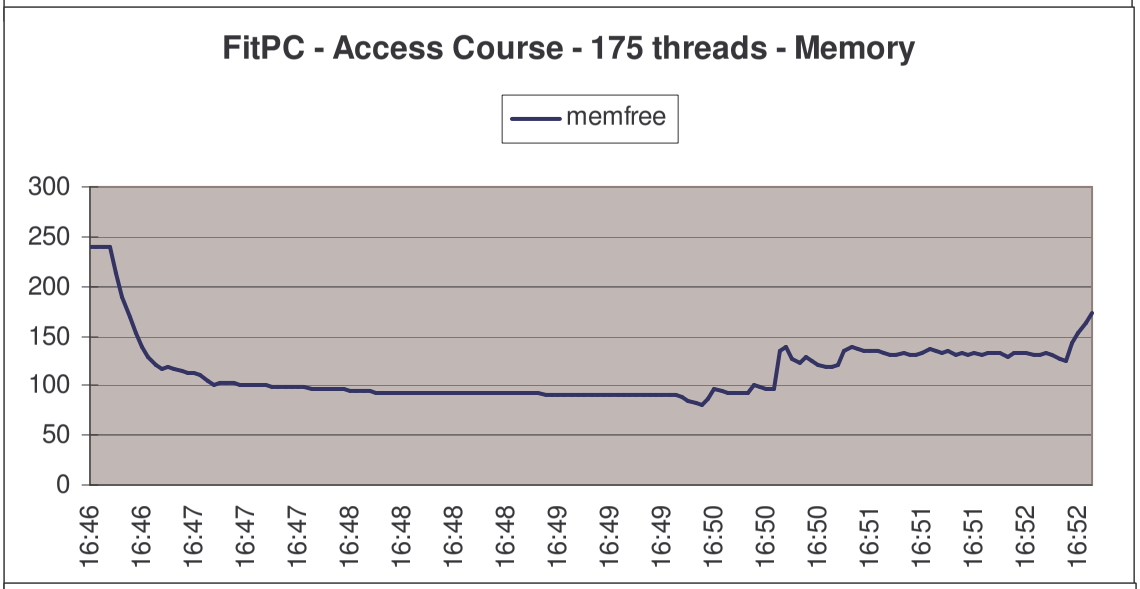
J.2 FitPC

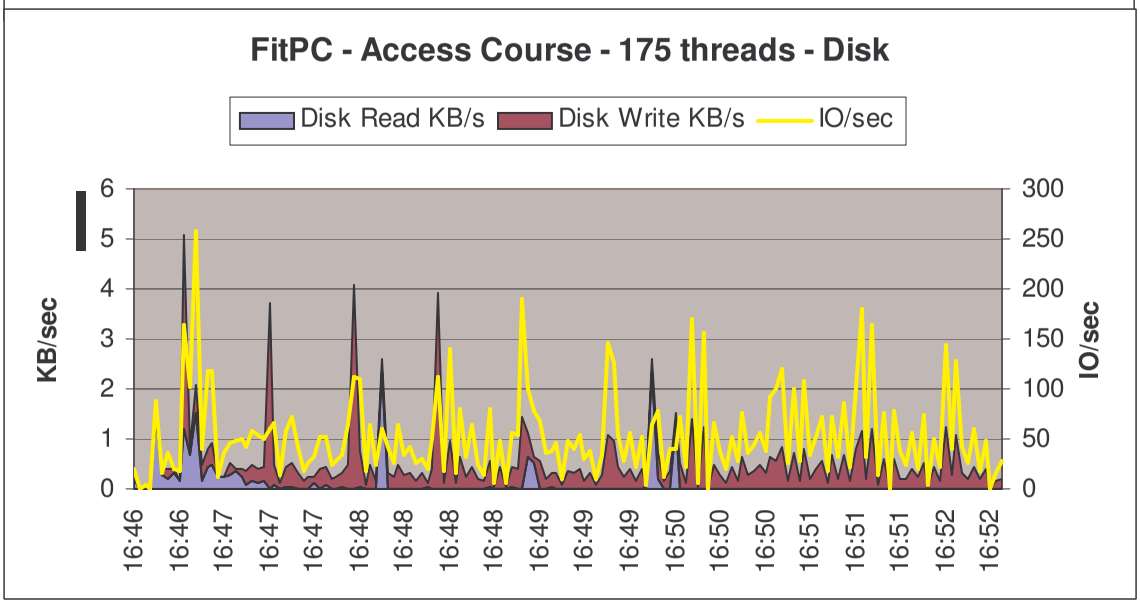

FitPC - Log In
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
120000
100000
80000
60000
40000
20000
0
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
0
50
100
150
200
250

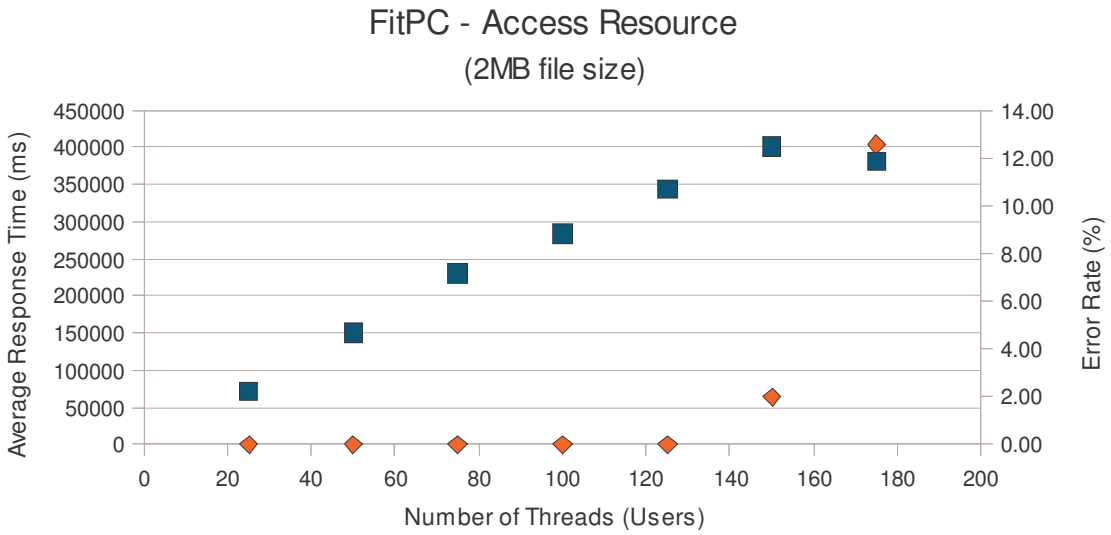

FitPC - Log In - 175 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
15:41
15:41
15:42
15:42
15:42
15:42
15:43
15:43
15:43
15:43
15:43
15:44
15:44
15:44
15:44
15:45
15:45
15:45
15:45
FitPC - Log In - 175 threads - Memory
memfree
250
200
150
100
50
0
15:41
15:41
15:42
15:42
15:42
15:42
15:43
15:43
15:43
15:43
15:43
15:44
15:44
15:44
15:44
15:45
15:45
15:45
15:45
FitPC - Log In - 175 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
4500
4000
3500
3000
2500
2000
1500
1000
500
0
300
250
200
150
100
50
0
15:41
15:41
15:42
15:42
15:42
15:42
15:43
15:43
15:43
15:43
15:43
15:44
15:44
15:44
15:44
15:45
15:45
15:45
15:45

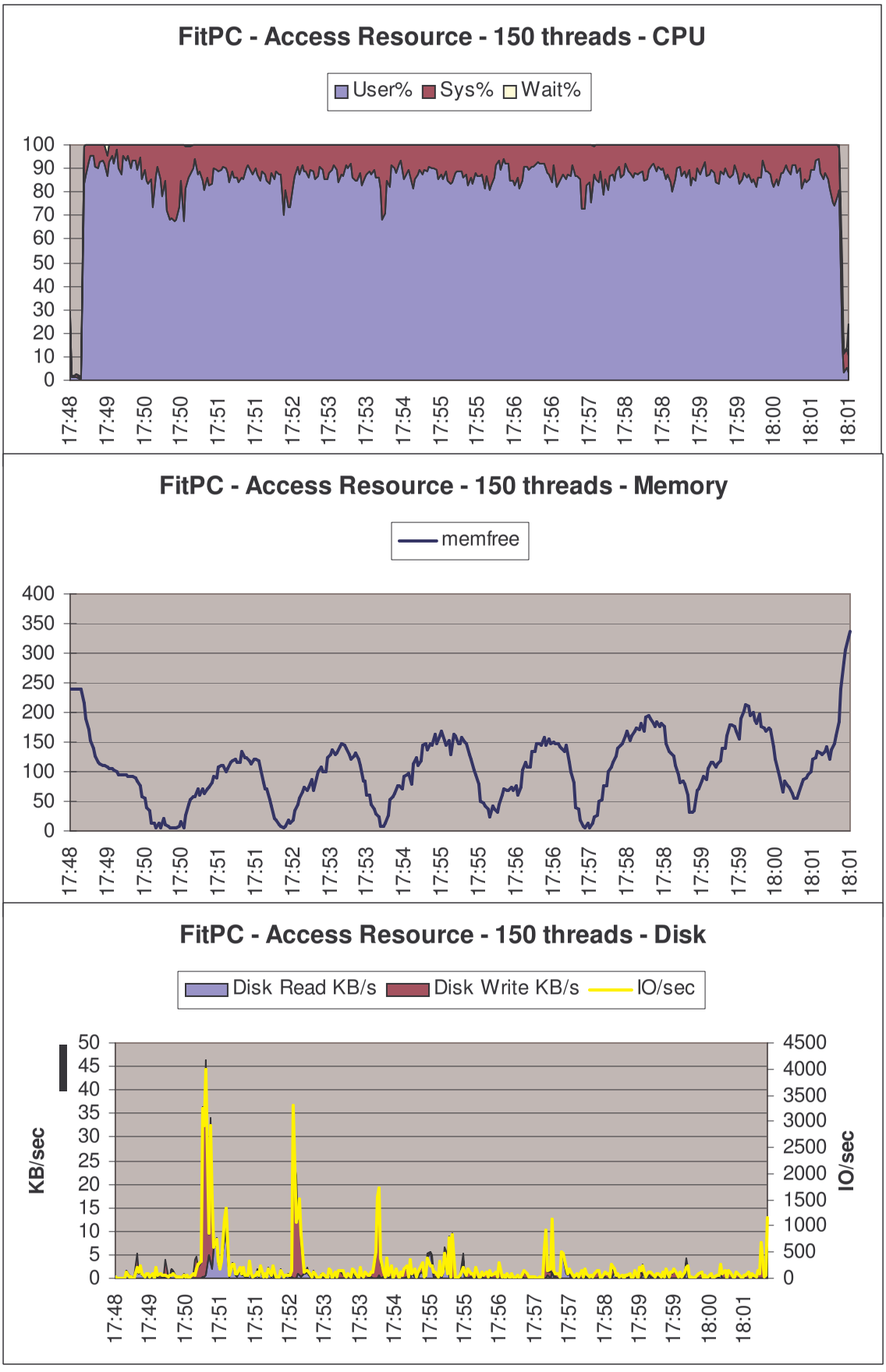

FitPC - Access Course
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
200000
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
20.00
18.00
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250

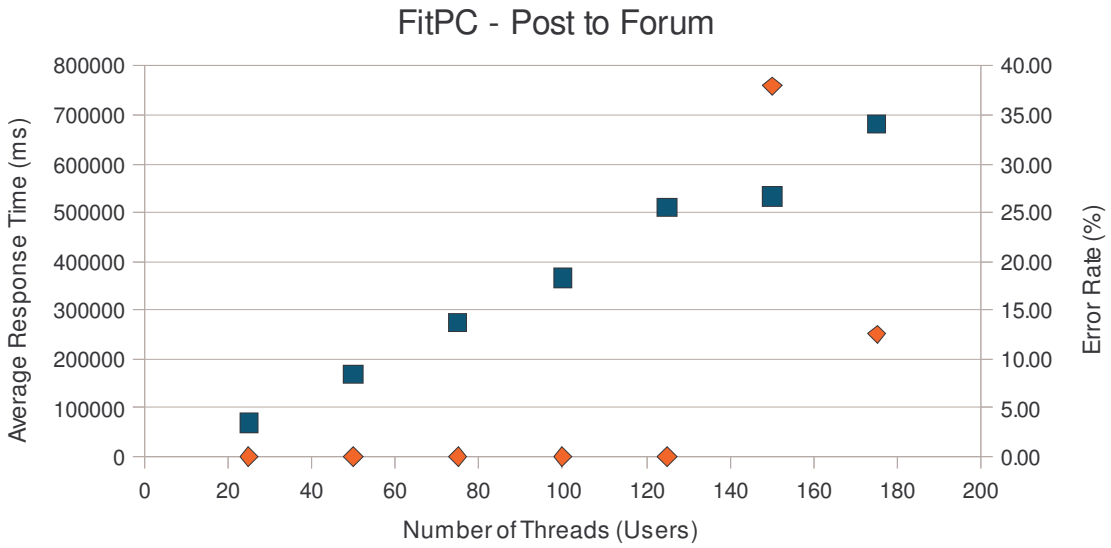

FitPC - Access Course - 175 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
16:46
16:46
16:47
16:47
16:47
16:48
16:48
16:48
16:48
16:49
16:49
16:49
16:50
16:50
16:50
16:51
16:51
16:51
16:52
16:52

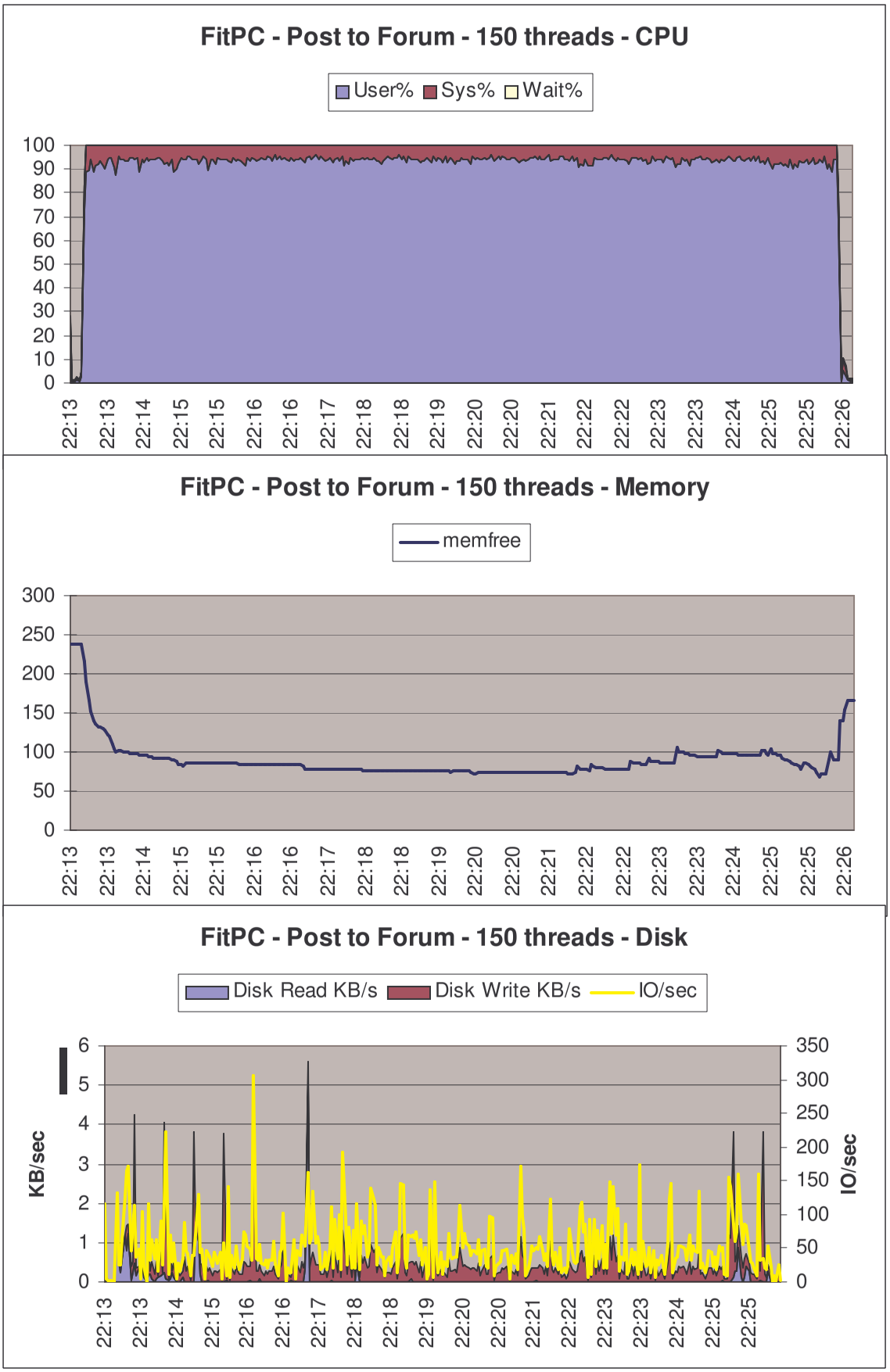

FitPC - Access Course - 175 threads - Memory
memfree
300
250
200
150
100
50
0
16:46
16:46
16:47
16:47
16:47
16:48
16:48
16:48
16:48
16:49
16:49
16:49
16:50
16:50
16:50
16:51
16:51
16:51
16:52
16:52

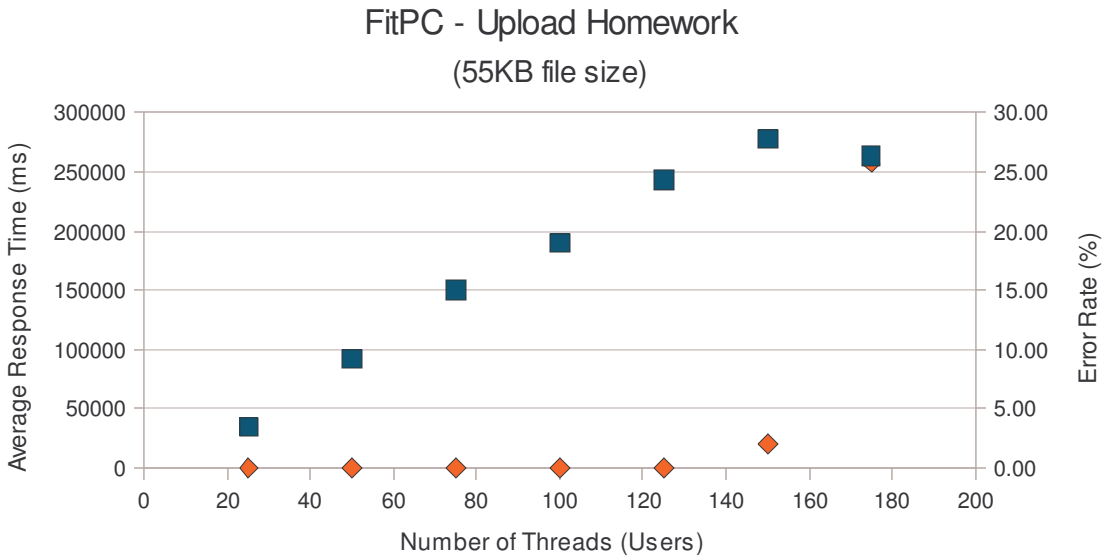

FitPC - Access Course - 175 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
6
5
4
3
2
1
0
300
250
200
150
100
50
0
16:46
16:46
16:47
16:47
16:47
16:48
16:48
16:48
16:48
16:49
16:49
16:49
16:50
16:50
16:50
16:51
16:51
16:51
16:52
16:52

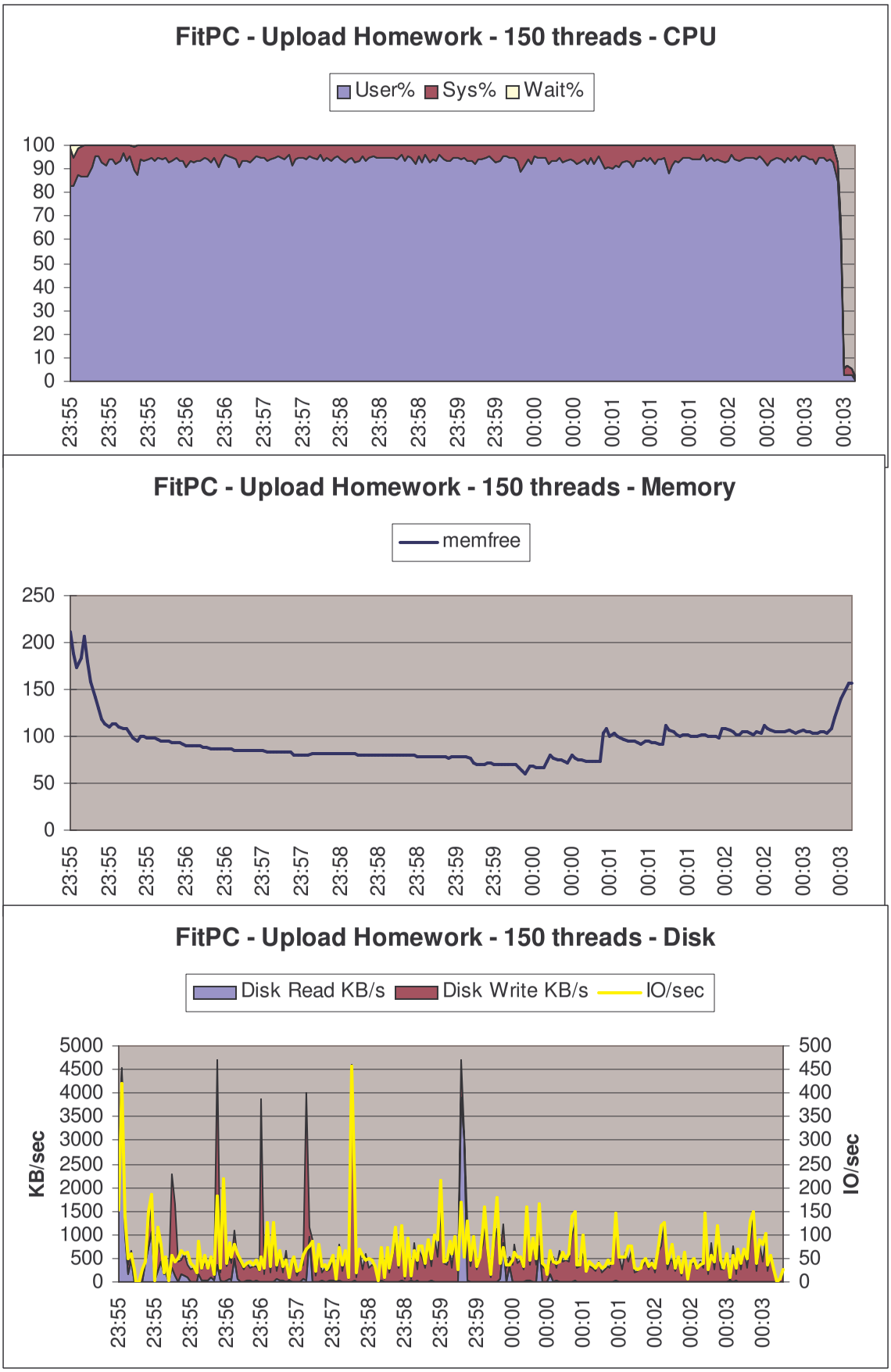

FitPC - Access Resource
(2MB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
20
40
60
80
100
120
140
160
180
200

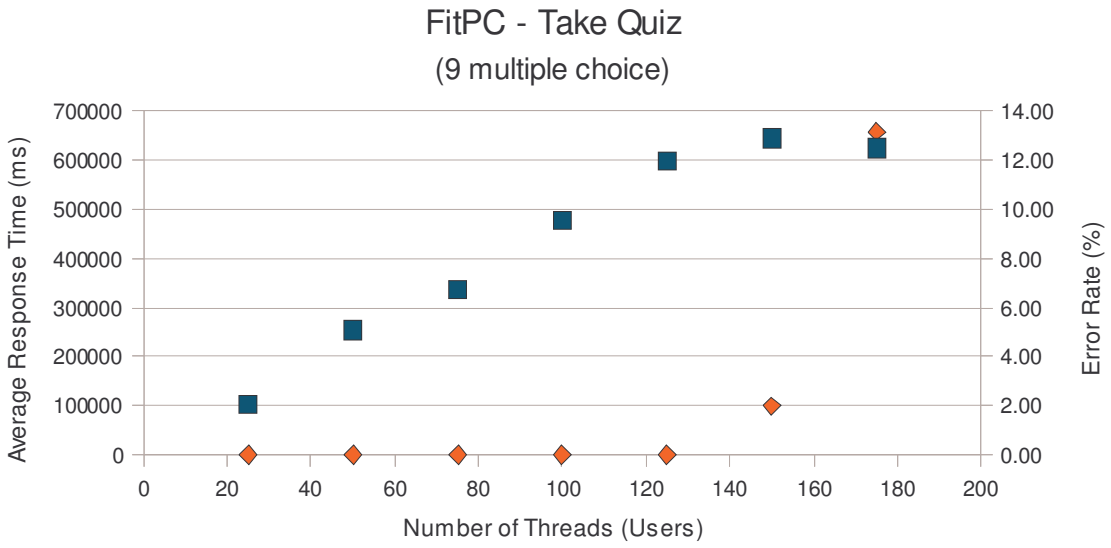

FitPC - Access Resource - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
17:48
17:49
17:50
17:50
17:51
17:51
17:52
17:53
17:53
17:54
17:55
17:55
17:56
17:56
17:57
17:58
17:58
17:59
17:59
18:00
18:01
18:01
FitPC - Access Resource - 150 threads - Memory
memfree
400
350
300
250
200
150
100
50
0
17:48
17:49
17:50
17:50
17:51
17:51
17:52
17:53
17:53
17:54
17:55
17:55
17:56
17:56
17:57
17:58
17:58
17:59
17:59
18:00
18:01
18:01
FitPC - Access Resource - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
50
45
40
35
30
25
20
15
10
5
0
IO/sec
4500
4000
3500
3000
2500
2000
1500
1000
500
0
17:48
17:49
17:50
17:51
17:51
17:52
17:53
17:53
17:54
17:55
17:55
17:56
17:57
17:57
17:58
17:59
17:59
18:00
18:01

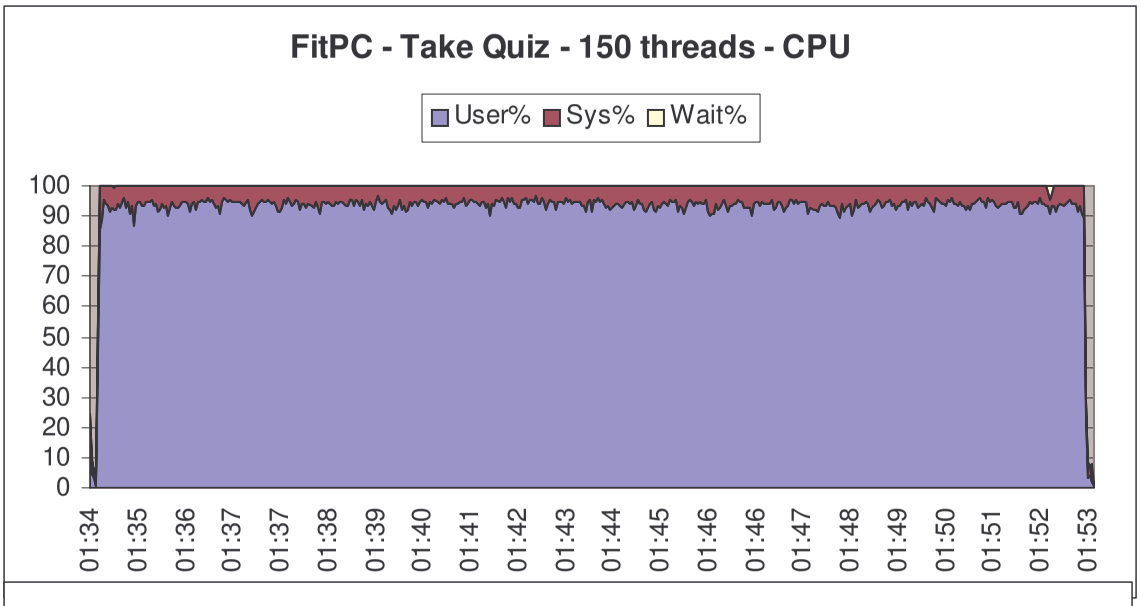

FitPC - Post to Forum
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
800000
700000
600000
500000
400000
300000
200000
100000
0
40.00
35.00
30.00
25.00
20.00
15.00
10.00
5.00
0.00
0
20
40
60
80
100
120
140
160
180
200

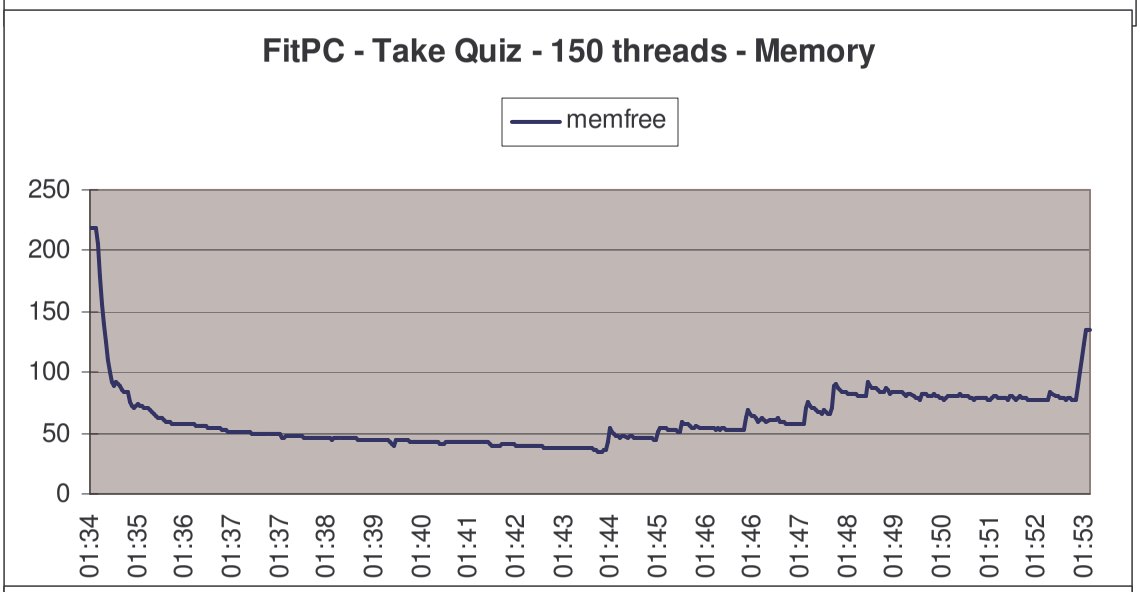

FitPC - Post to Forum - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
22:13
22:13
22:14
22:15
22:15
22:16
22:16
22:17
22:18
22:18
22:19
22:20
22:20
22:21
22:22
22:22
22:23
22:23
22:24
22:25
22:25
22:26
FitPC - Post to Forum - 150 threads - Memory
memfree
300
250
200
150
100
50
0
22:13
22:13
22:14
22:15
22:15
22:16
22:16
22:17
22:18
22:18
22:19
22:20
22:20
22:21
22:22
22:22
22:23
22:23
22:24
22:25
22:25
22:26
FitPC - Post to Forum - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
6
5
4
3
2
1
0
350
300
250
200
150
100
50
0
IO/sec
22:13
22:13
22:14
22:15
22:16
22:16
22:17
22:18
22:18
22:19
22:20
22:20
22:21
22:22
22:23
22:23
22:24
22:25
22:25

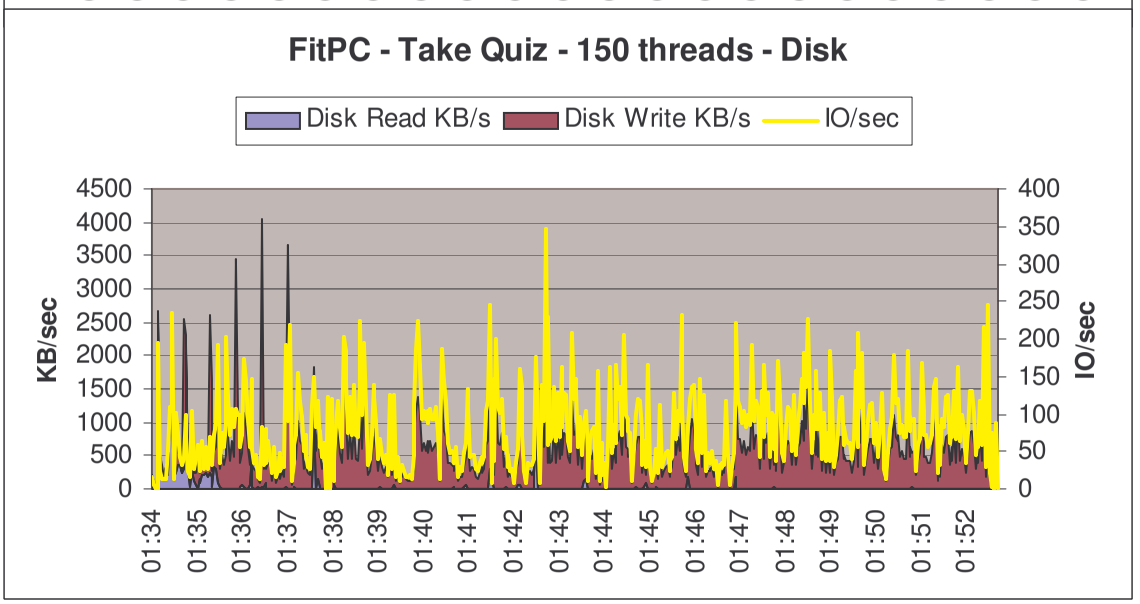

FitPC - Upload Homework
(55KB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
300000
250000
200000
150000
100000
50000
0
30.00
25.00
20.00
15.00
10.00
5.00
0.00
0
20
40
60
80
100
120
140
160
180
200


FitPC - Upload Homework - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
23:55
23:55
23:55
23:56
23:56
23:57
23:57
23:58
23:58
23:58
23:59
23:59
00:00
00:00
00:01
00:01
00:01
00:02
00:02
00:03
00:03
FitPC - Upload Homework - 150 threads - Memory
memfree
250
200
150
100
50
0
23:55
23:55
23:55
23:56
23:56
23:57
23:57
23:58
23:58
23:58
23:59
23:59
00:00
00:00
00:01
00:01
00:01
00:02
00:02
00:03
00:03
FitPC - Upload Homework - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
IO/sec
500
450
400
350
300
250
200
150
100
50
0
23:55
23:55
23:55
23:56
23:56
23:57
23:57
23:58
23:58
23:59
23:59
00:00
00:00
00:01
00:01
00:02
00:02
00:03
00:03

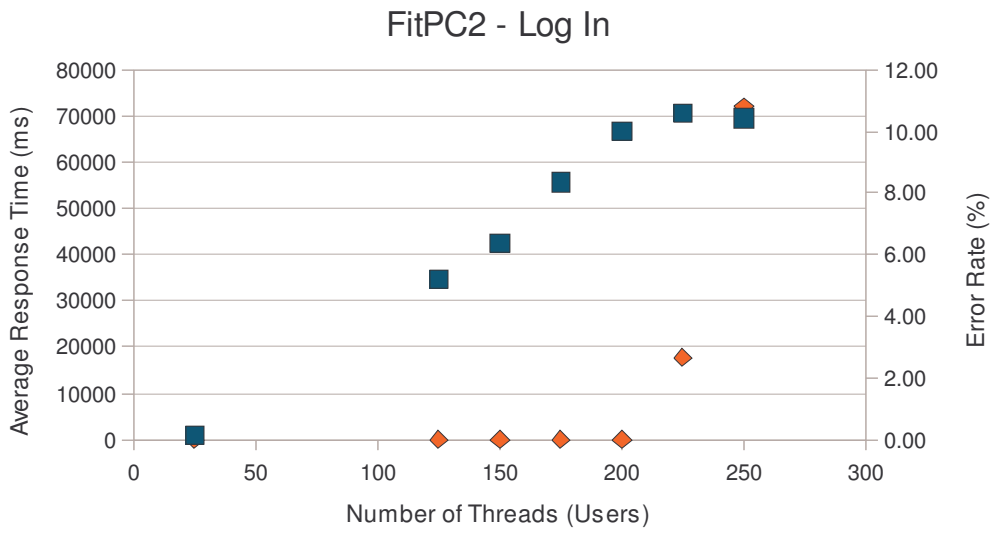

FitPC - Take Quiz
(9 multiple choice)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
700000
600000
500000
400000
300000
200000
100000
0
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
20
40
60
80
100
120
140
160
180
200

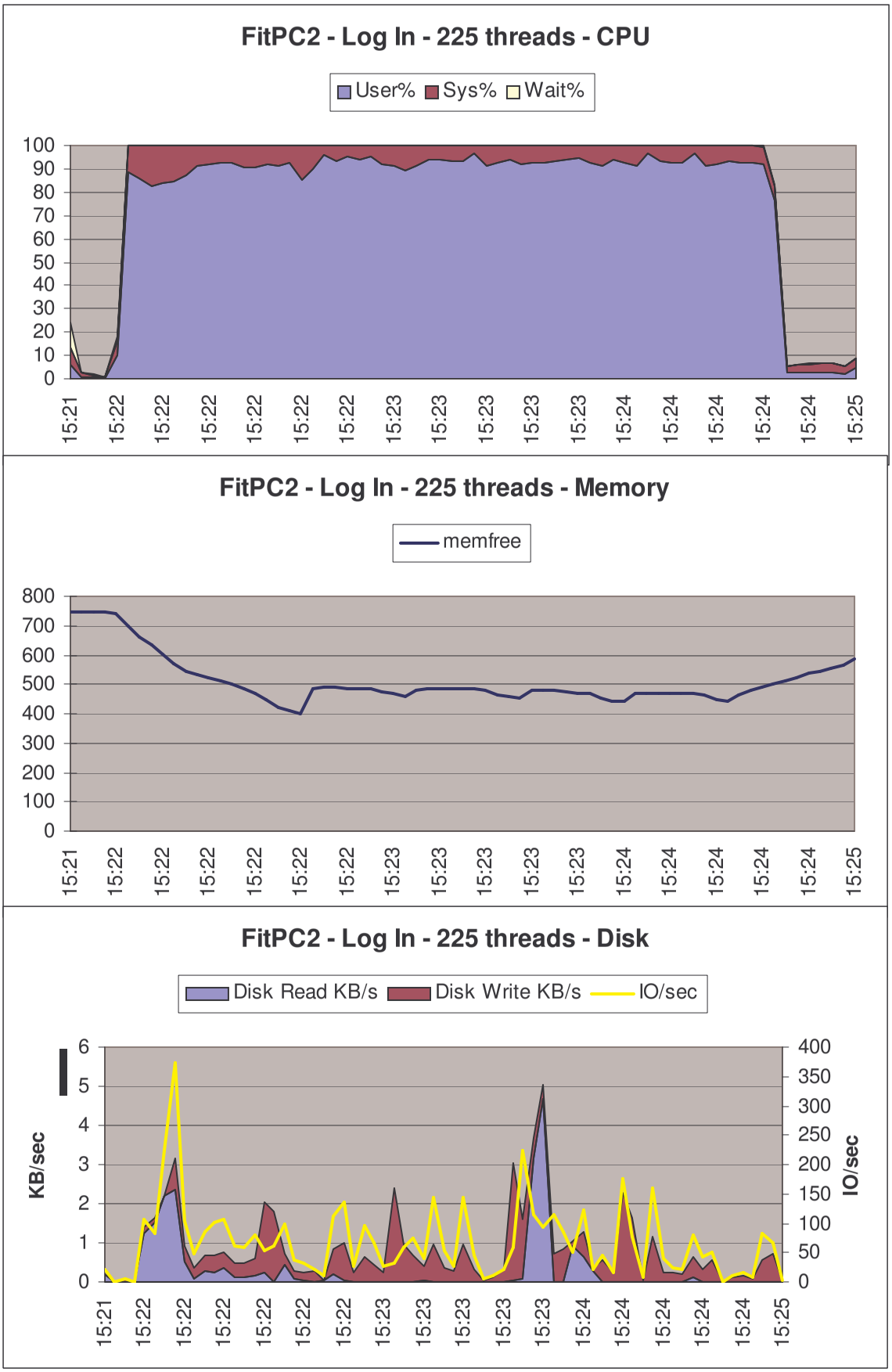

FitPC - Take Quiz - 150 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
01:34
01:35
01:36
01:37
01:37
01:38
01:39
01:40
01:41
01:42
01:43
01:44
01:45
01:46
01:46
01:47
01:48
01:49
01:50
01:51
01:52
01:53

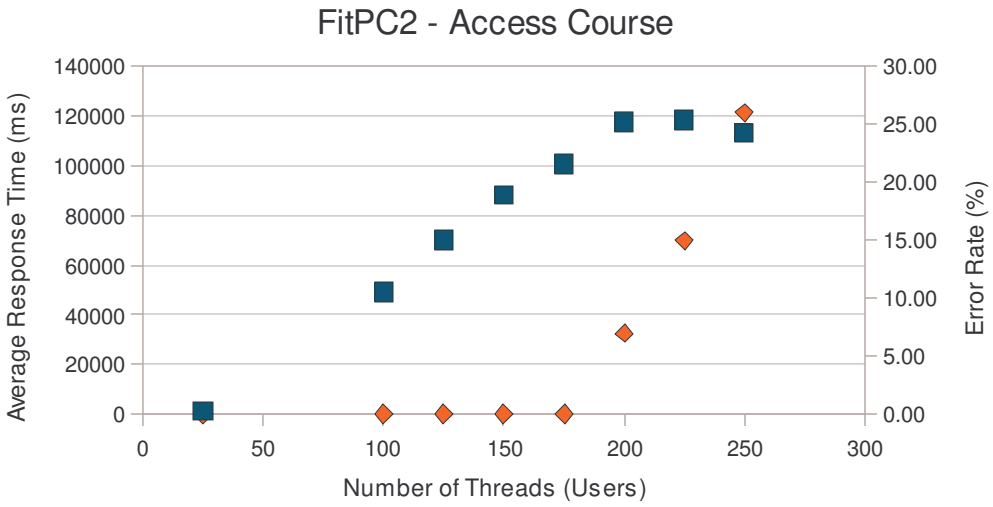

FitPC - Take Quiz - 150 threads - Memory
memfree
250
200
150
100
50
0
01:34
01:35
01:36
01:37
01:37
01:38
01:39
01:40
01:41
01:42
01:43
01:44
01:45
01:46
01:46
01:47
01:48
01:49
01:50
01:51
01:52
01:53

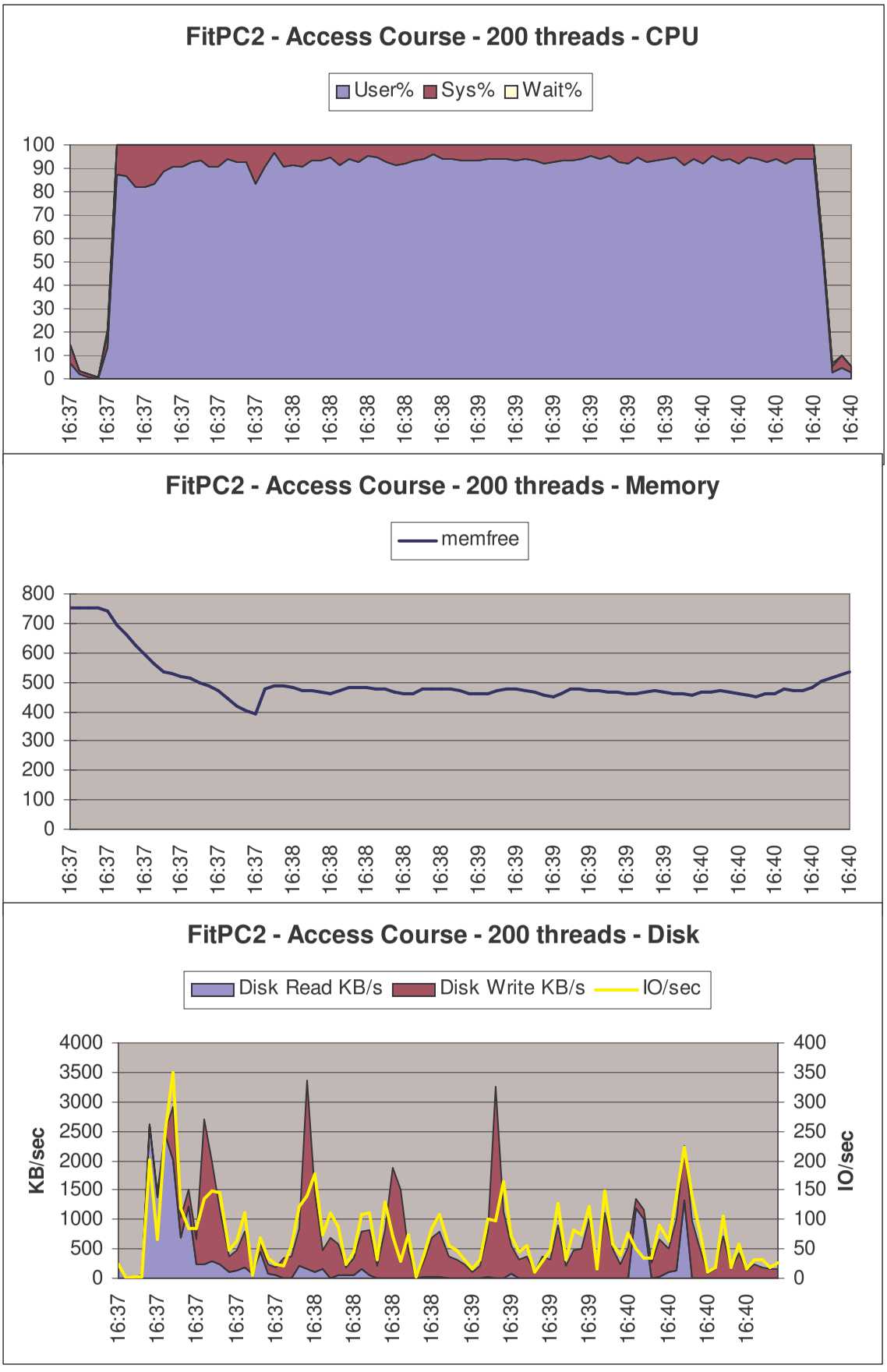

FitPC - Take Quiz - 150 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
4500
4000
3500
3000
2500
2000
1500
1000
500
0
400
350
300
250
200
150
100
50
0
01:34
01:35
01:36
01:37
01:38
01:39
01:40
01:41
01:42
01:43
01:44
01:45
01:46
01:47
01:48
01:49
01:50
01:51
01:52

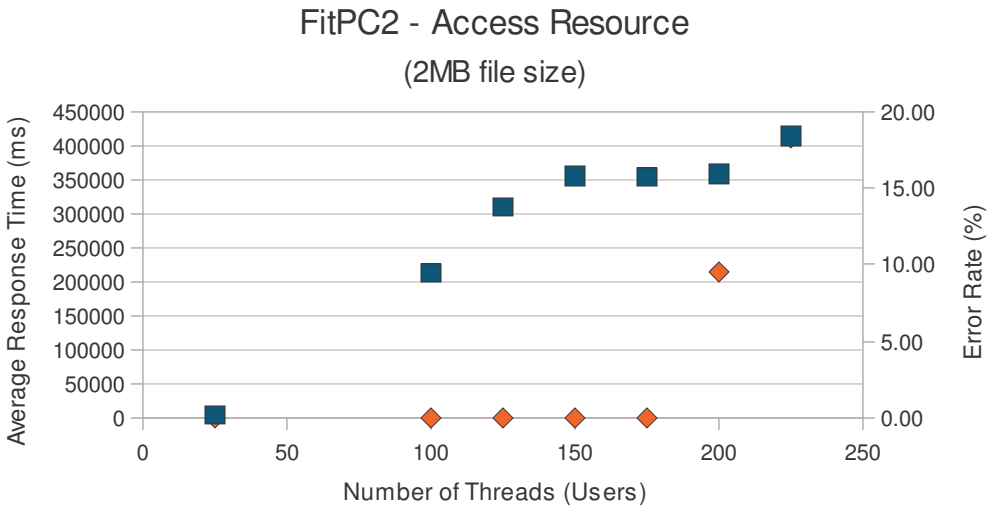
J.3 FitPC2

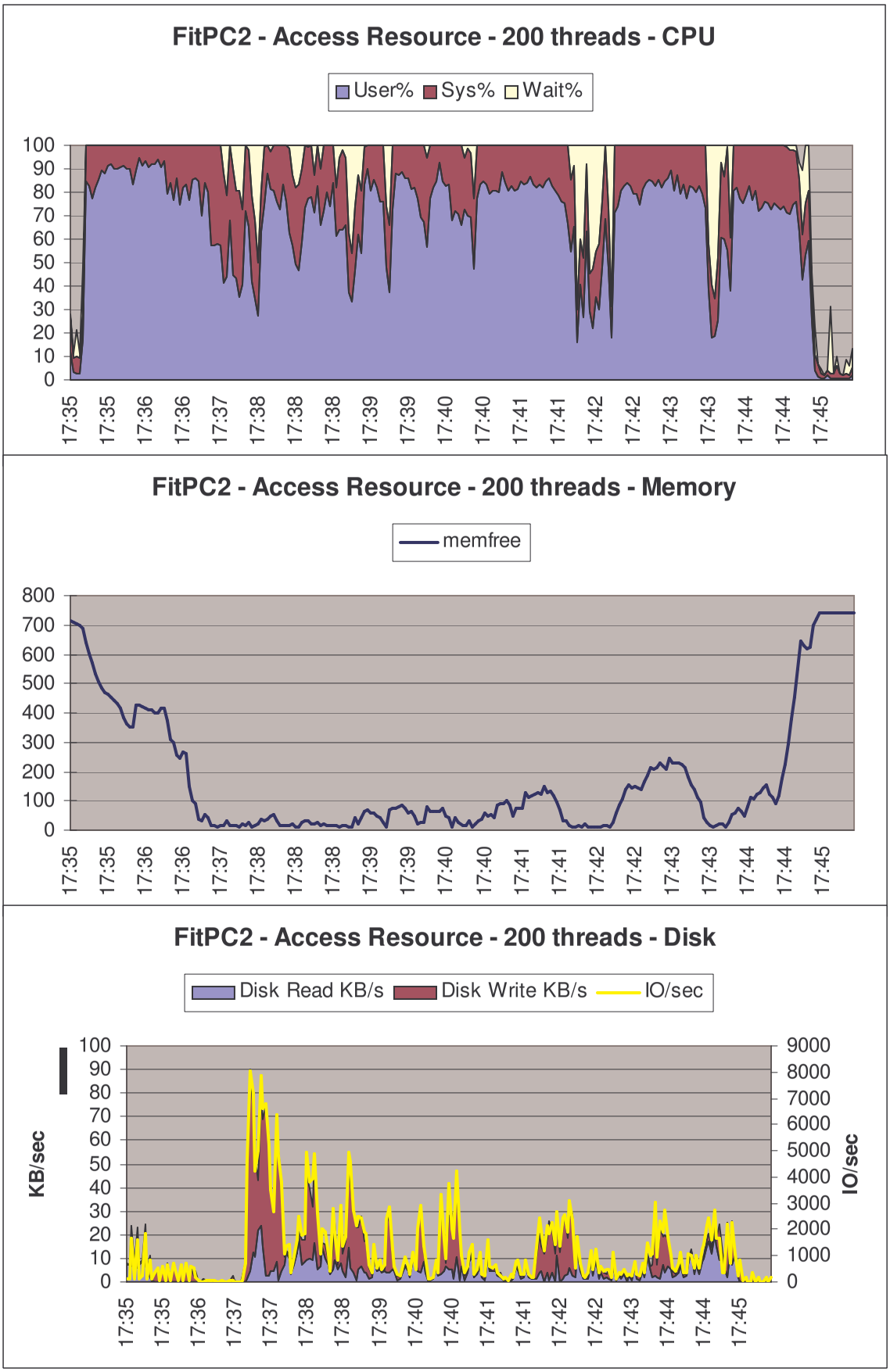

FitPC2 - Log In
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
80000
70000
60000
50000
40000
30000
20000
10000
0
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250
300

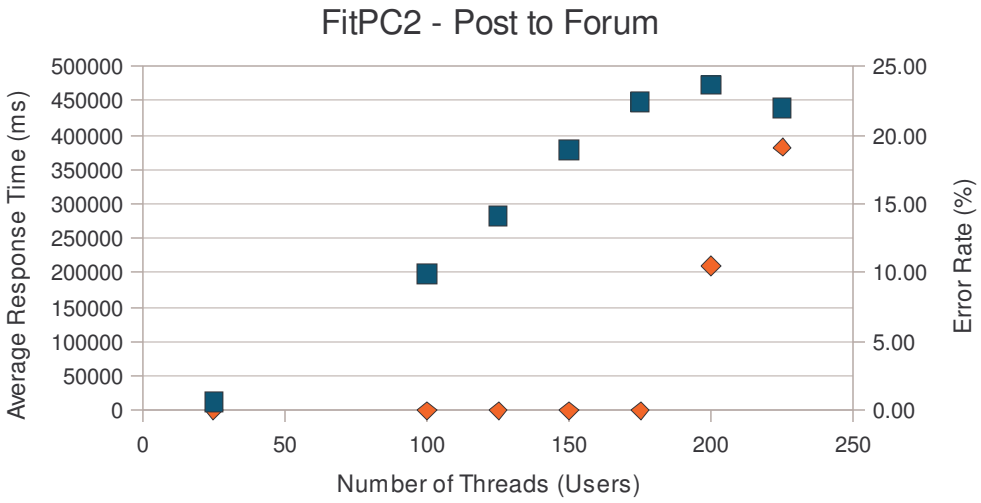

FitPC2 - Log In - 225 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
15:21
15:22
15:22
15:22
15:22
15:22
15:22
15:23
15:23
15:23
15:23
15:24
15:24
15:24
15:24
15:24
15:25
FitPC2 - Log In - 225 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
15:21
15:22
15:22
15:22
15:22
15:22
15:22
15:23
15:23
15:23
15:23
15:24
15:24
15:24
15:24
15:24
15:25
FitPC2 - Log In - 225 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
6
5
4
3
2
1
0
400
350
300
250
200
150
100
50
0
15:21
15:22
15:22
15:22
15:22
15:22
15:22
15:23
15:23
15:23
15:23
15:24
15:24
15:24
15:24
15:24
15:25

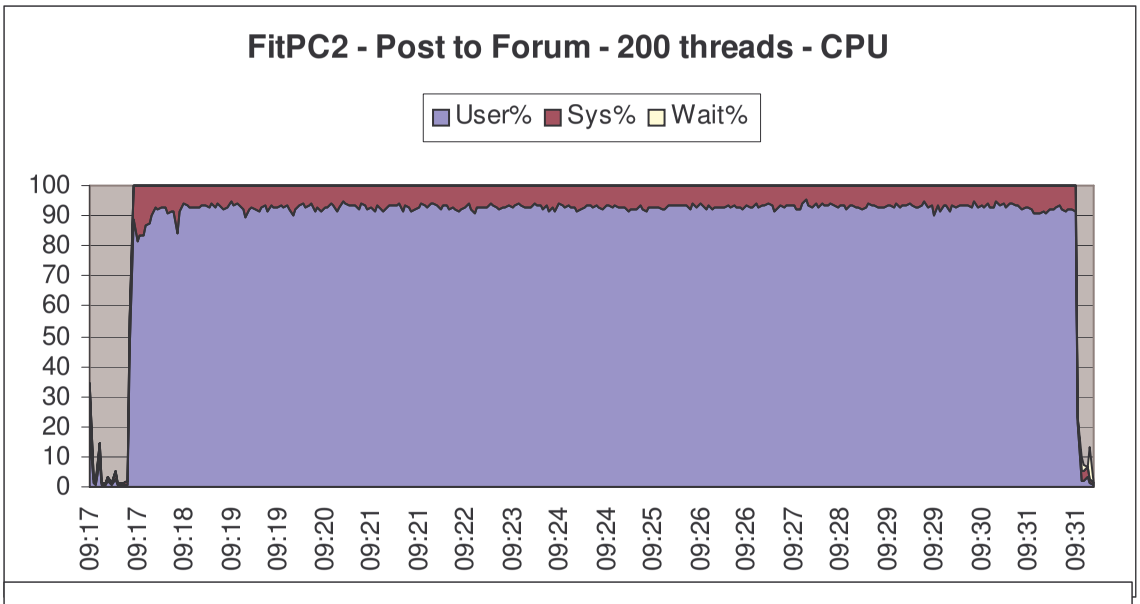

FitPC2 - Access Course
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
140000
120000
100000
80000
60000
40000
20000
0
30.00
25.00
20.00
15.00
10.00
5.00
0.00
0
50
100
150
200
250
300

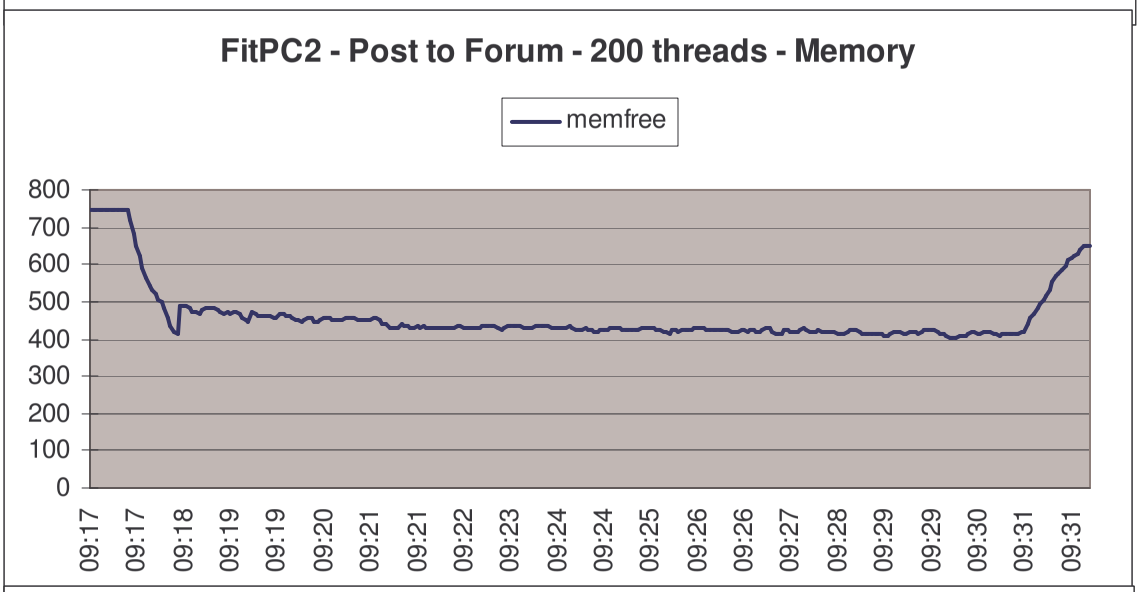

FitPC2 - Access Course - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
16:37
16:37
16:37
16:37
16:37
16:37
16:38
16:38
16:38
16:38
16:38
16:39
16:39
16:39
16:39
16:39
16:39
16:40
16:40
16:40
16:40
16:40
FitPC2 - Access Course - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
16:37
16:37
16:37
16:37
16:37
16:37
16:38
16:38
16:38
16:38
16:38
16:39
16:39
16:39
16:39
16:39
16:39
16:40
16:40
16:40
16:40
16:40
FitPC2 - Access Course - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
4000
3500
3000
2500
2000
1500
1000
500
0
IO/sec
400
350
300
250
200
150
100
50
0
16:37
16:37
16:37
16:37
16:37
16:38
16:38
16:38
16:38
16:39
16:39
16:39
16:39
16:40
16:40
16:40
16:40

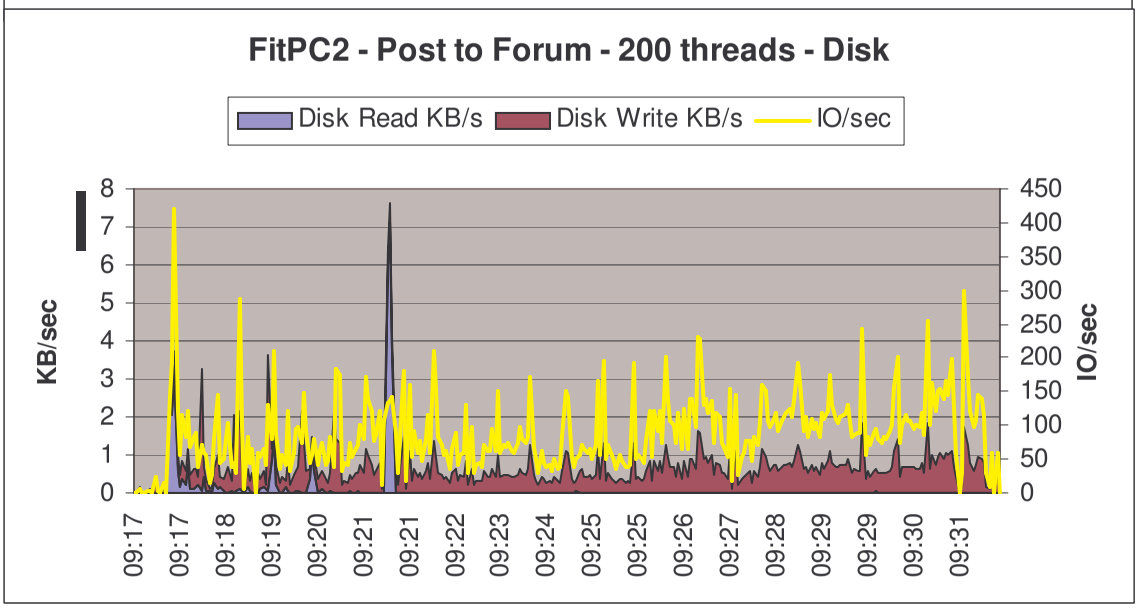

FitPC2 - Access Resource
(2MB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
20.00
15.00
10.00
5.00
0.00
0
50
100
150
200
250

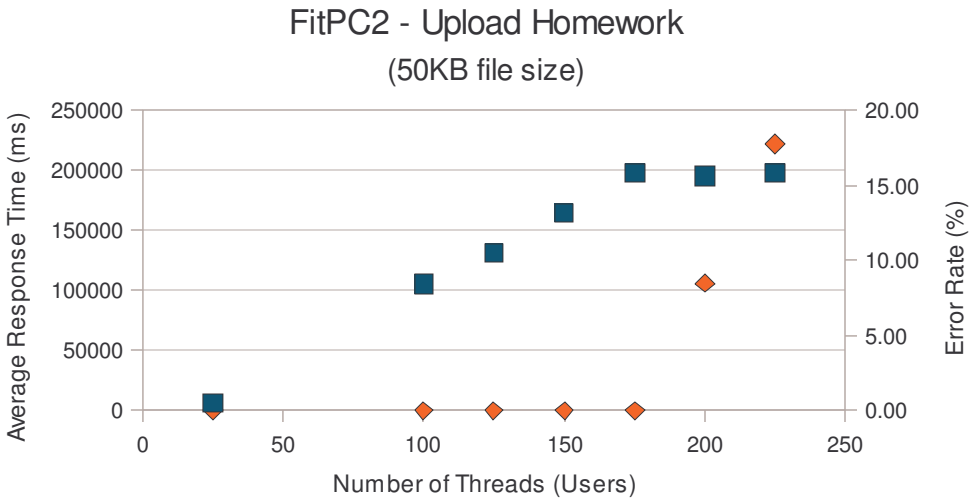

FitPC2 - Access Resource - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
17:35
17:35
17:36
17:36
17:37
17:38
17:38
17:38
17:39
17:39
17:40
17:40
17:41
17:41
17:42
17:42
17:43
17:43
17:44
17:44
17:45
FitPC2 - Access Resource - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
17:35
17:35
17:36
17:36
17:37
17:38
17:38
17:38
17:39
17:39
17:40
17:40
17:41
17:41
17:42
17:42
17:43
17:43
17:44
17:44
17:45
FitPC2 - Access Resource - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
100
90
80
70
60
50
40
30
20
10
0
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
17:35
17:35
17:36
17:37
17:37
17:38
17:38
17:39
17:40
17:40
17:41
17:41
17:42
17:42
17:43
17:44
17:44
17:45

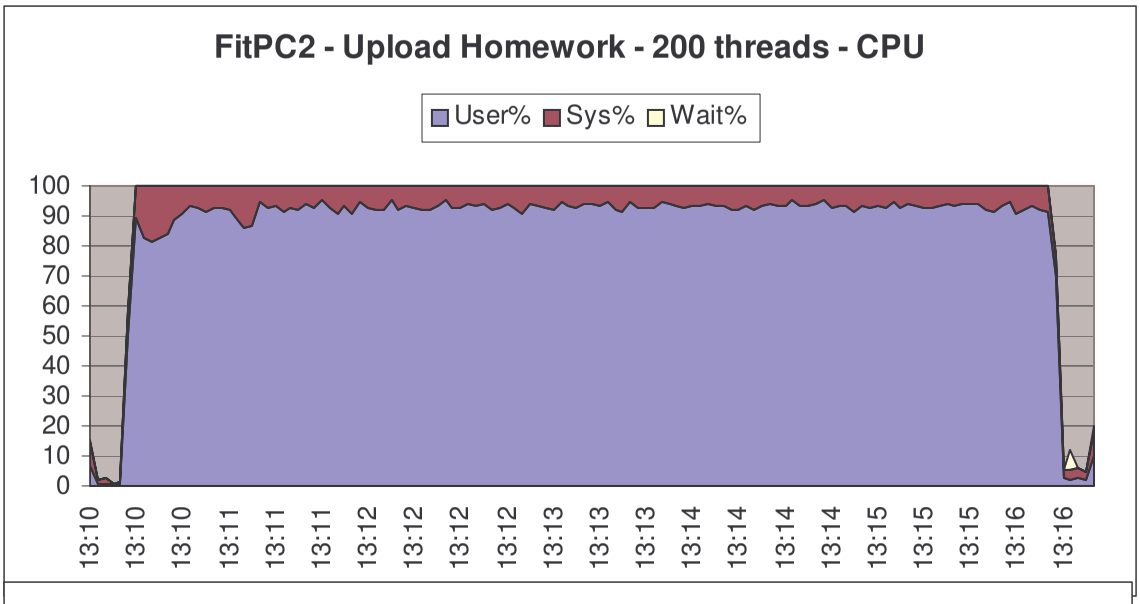

FitPC2 - Post to Forum
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
500000
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
25.00
20.00
15.00
10.00
5.00
0.00
0
50
100
150
200
250

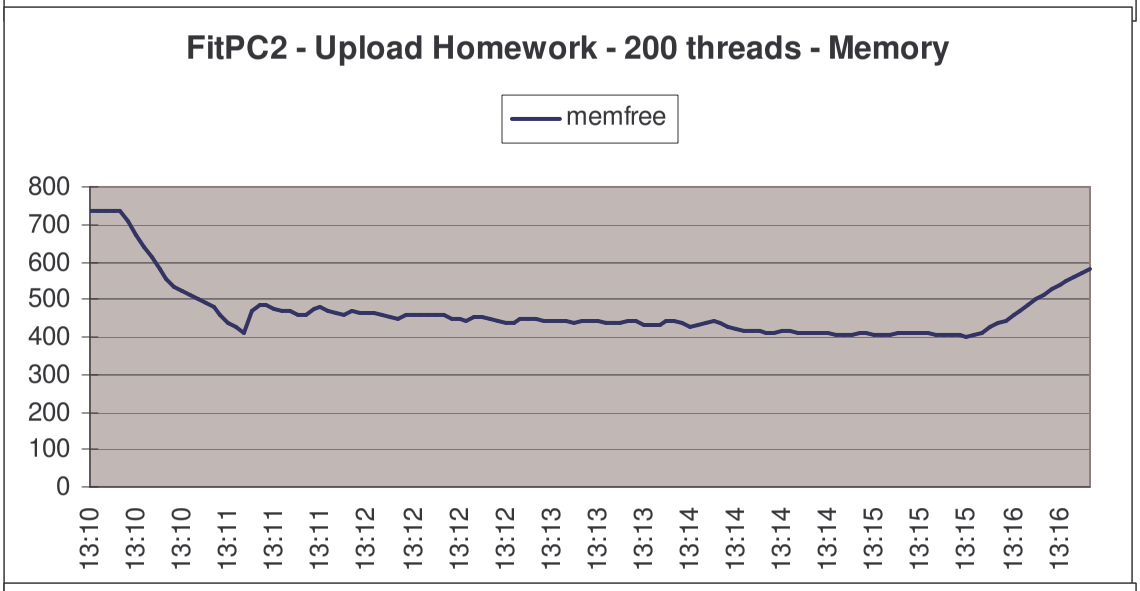

FitPC2 - Post to Forum - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
09:17
09:17
09:18
09:19
09:19
09:20
09:21
09:21
09:22
09:23
09:24
09:24
09:25
09:26
09:26
09:27
09:28
09:29
09:29
09:30
09:31
09:31

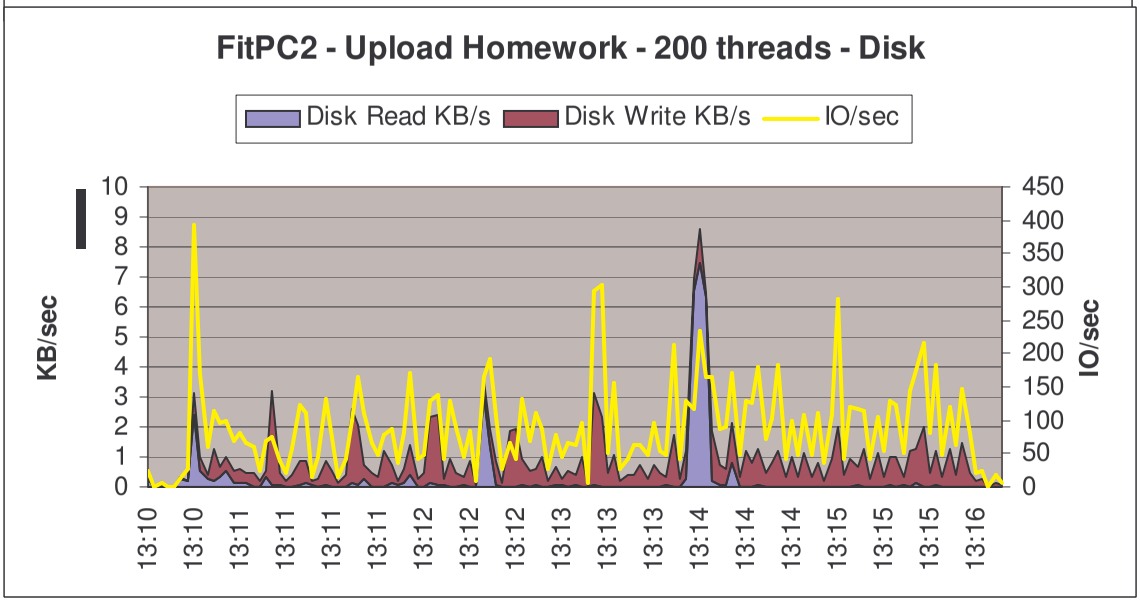

FitPC2 - Post to Forum - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
09:17
09:17
09:18
09:19
09:19
09:20
09:21
09:21
09:22
09:23
09:24
09:24
09:25
09:26
09:26
09:27
09:28
09:29
09:29
09:30
09:31
09:31

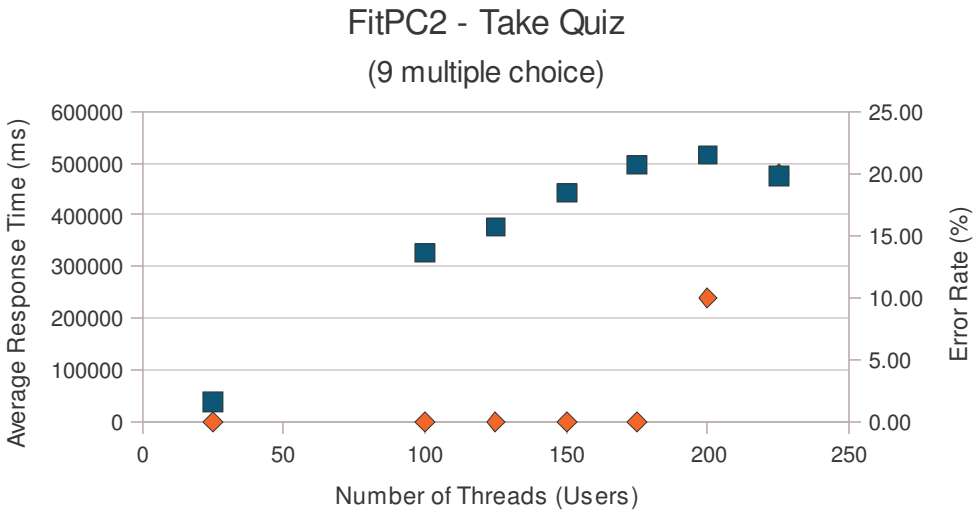

FitPC2 - Post to Forum - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
8
7
6
5
4
3
2
1
0
450
400
350
300
250
200
150
100
50
0
09:17
09:17
09:18
09:19
09:20
09:21
09:21
09:22
09:23
09:24
09:25
09:25
09:26
09:27
09:28
09:29
09:29
09:30
09:31

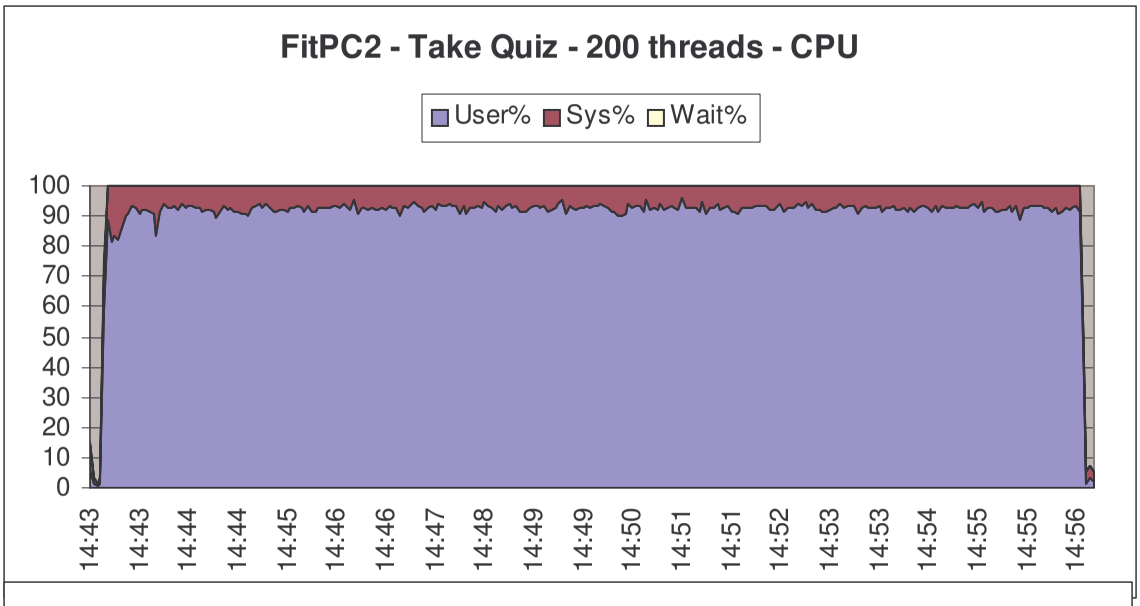

FitPC2 - Upload Homework
(50KB file size)
Average Response Time (ms)
250000
200000
150000
100000
50000
0
20.00
15.00
10.00
5.00
0.00
Error Rate (%)
0
50
100
150
200
250
Number of Threads (Users)

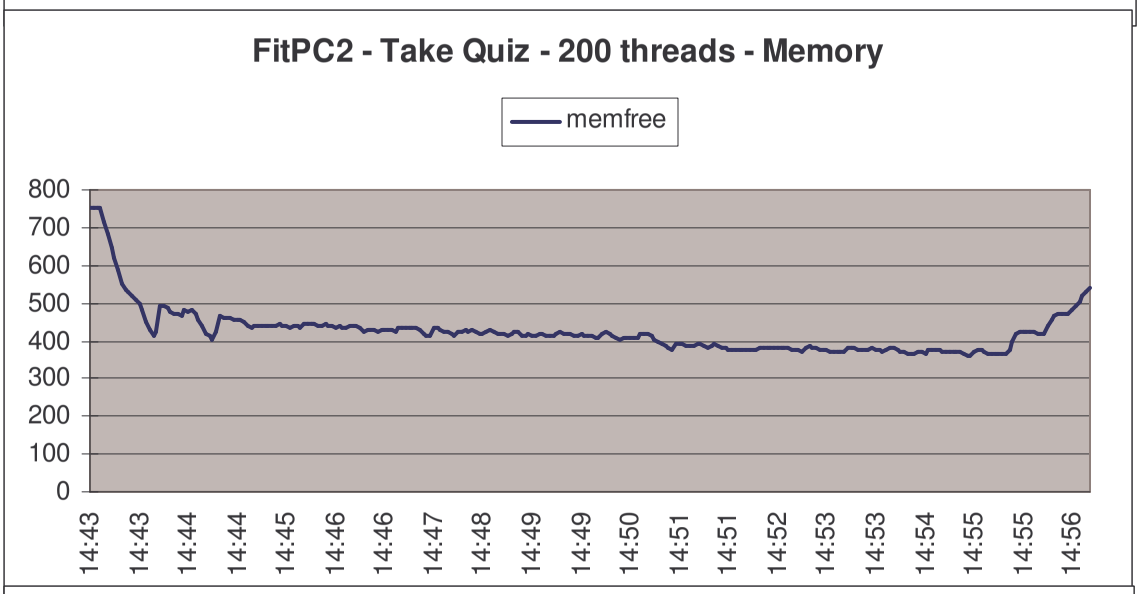

FitPC2 - Upload Homework - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
13:10
13:10
13:10
13:11
13:11
13:11
13:12
13:12
13:12
13:12
13:13
13:13
13:13
13:14
13:14
13:14
13:14
13:15
13:15
13:15
13:16
13:16

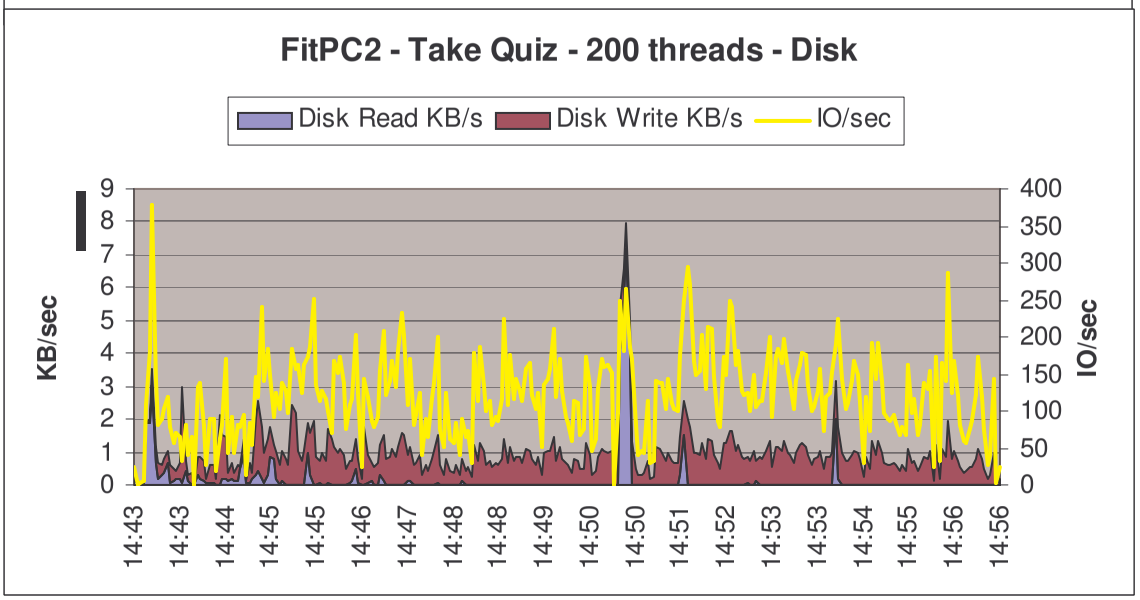

FitPC2 - Upload Homework - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
13:10
13:10
13:10
13:11
13:11
13:11
13:12
13:12
13:12
13:12
13:13
13:13
13:13
13:14
13:14
13:14
13:14
13:15
13:15
13:15
13:16
13:16


FitPC2 - Upload Homework - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
10
9
8
7
6
5
4
3
2
1
0
450
400
350
300
250
200
150
100
50
0
13:10
13:10
13:11
13:11
13:11
13:11
13:12
13:12
13:12
13:13
13:13
13:13
13:14
13:14
13:14
13:15
13:15
13:15
13:16

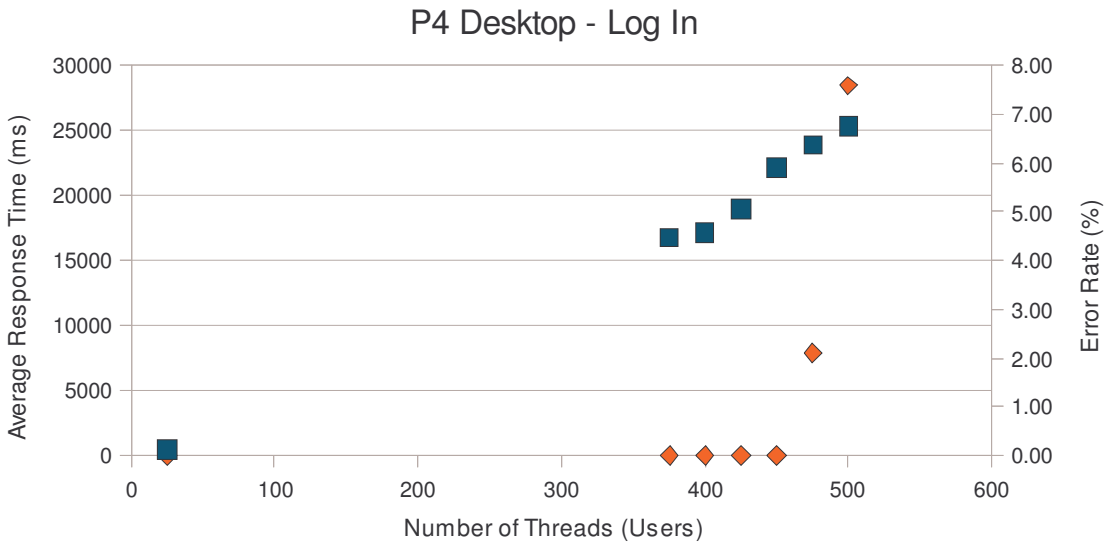

FitPC2 - Take Quiz
(9 multiple choice)
Average Response Time (ms)
600000
500000
400000
300000
200000
100000
0
25.00
20.00
15.00
10.00
5.00
0.00
Error Rate (%)
0
50
100
150
200
250
Number of Threads (Users)

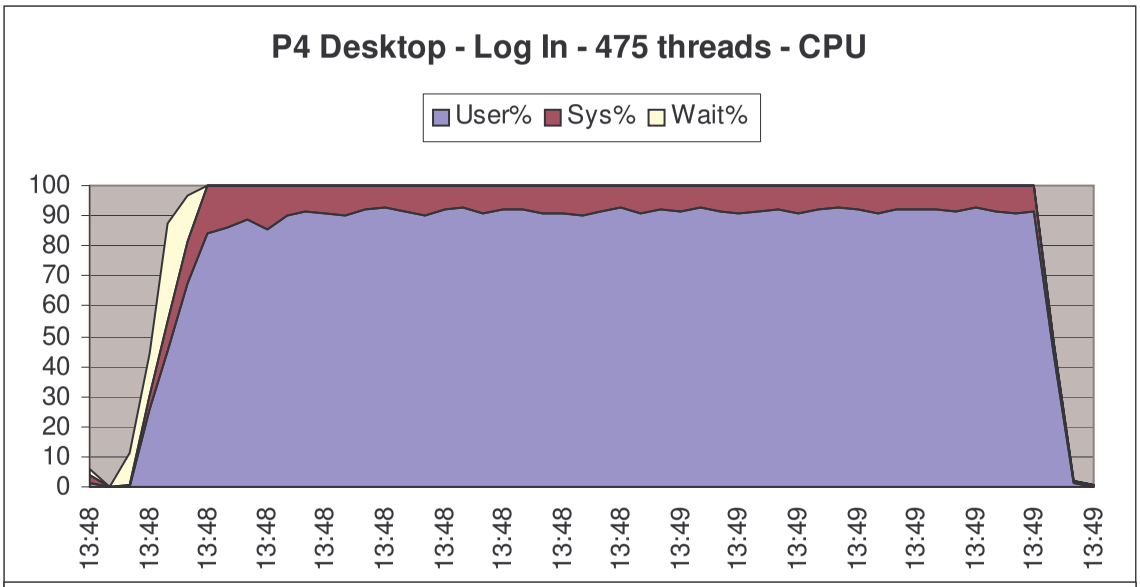

FitPC2 - Take Quiz - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
14:43
14:43
14:44
14:44
14:45
14:46
14:46
14:47
14:48
14:49
14:49
14:50
14:51
14:51
14:52
14:53
14:53
14:54
14:55
14:55
14:56

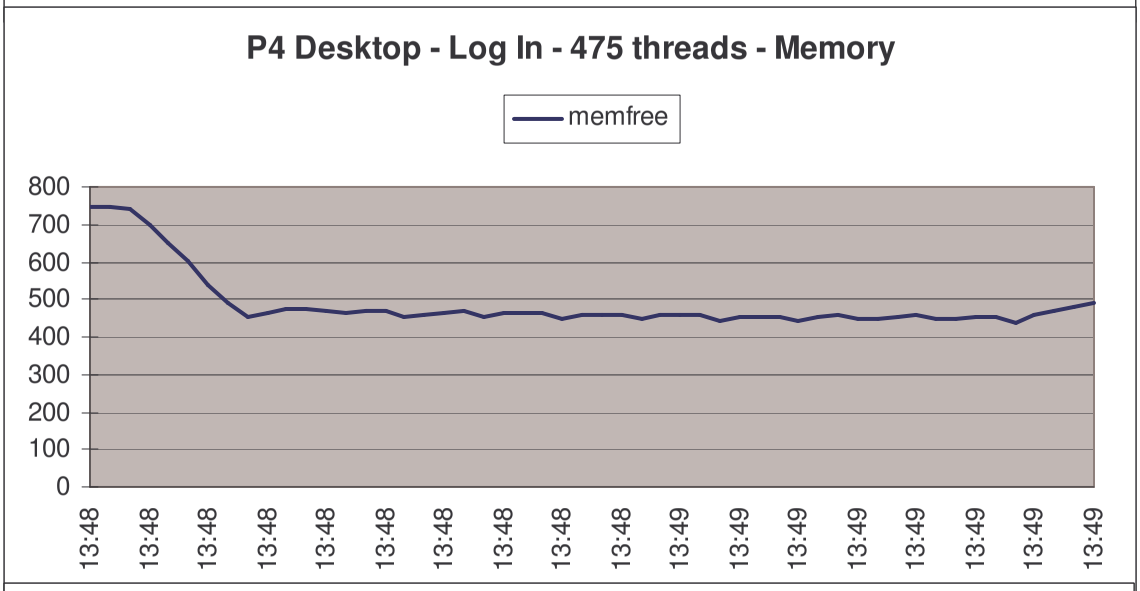

FitPC2 - Take Quiz - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
14:43
14:43
14:44
14:44
14:45
14:46
14:46
14:47
14:48
14:49
14:49
14:50
14:51
14:51
14:52
14:53
14:53
14:54
14:55
14:55
14:56

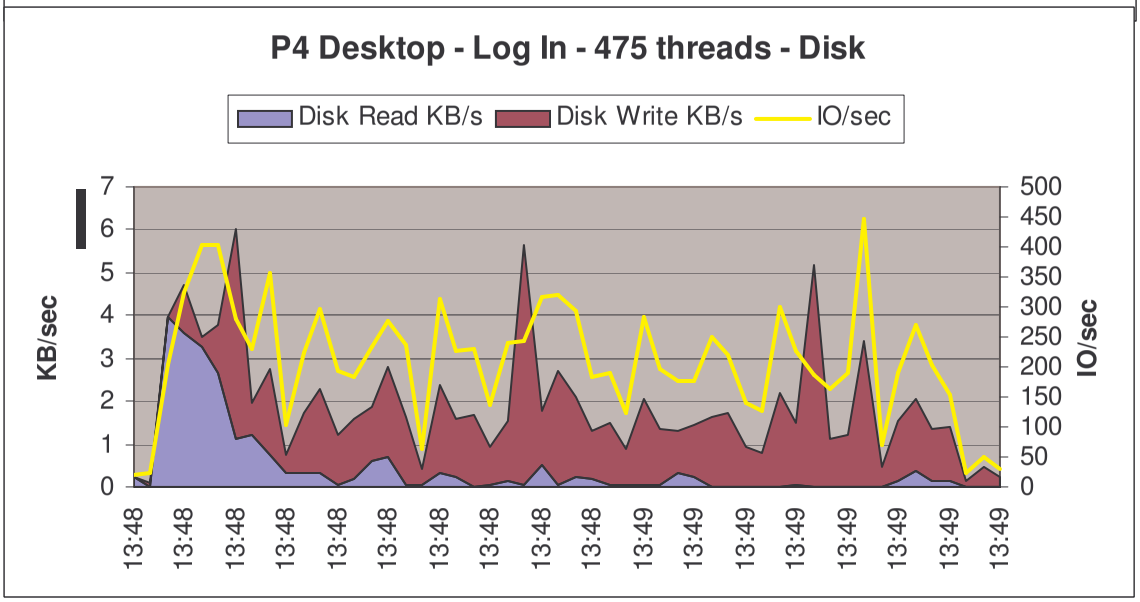

FitPC2 - Take Quiz - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
9
8
7
6
5
4
3
2
1
0
400
350
300
250
200
150
100
50
0
14:43
14:43
14:44
14:45
14:45
14:46
14:47
14:48
14:48
14:49
14:50
14:50
14:51
14:52
14:53
14:53
14:54
14:55
14:56
14:56

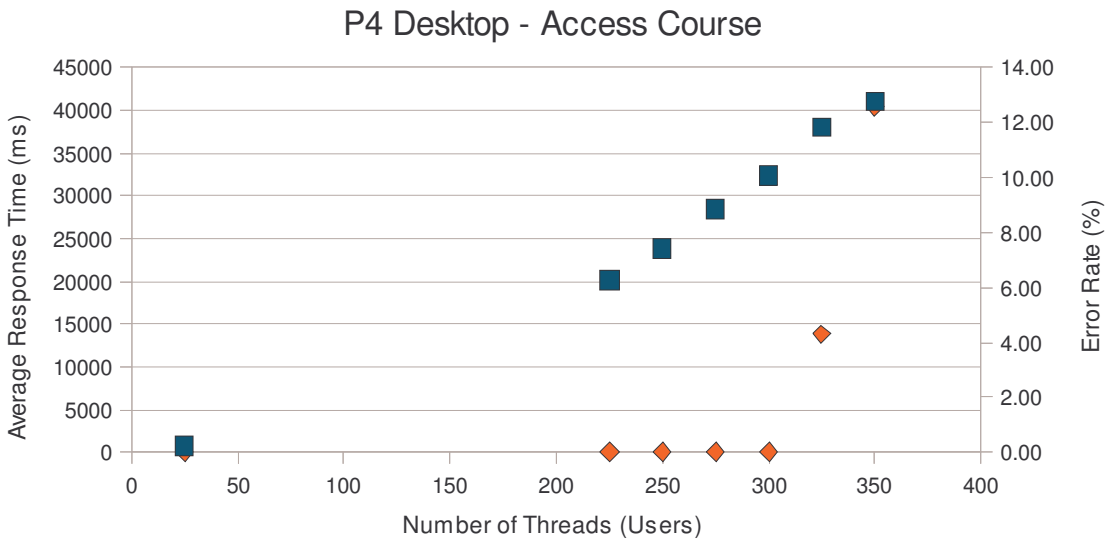
J.4 P4 Desktop

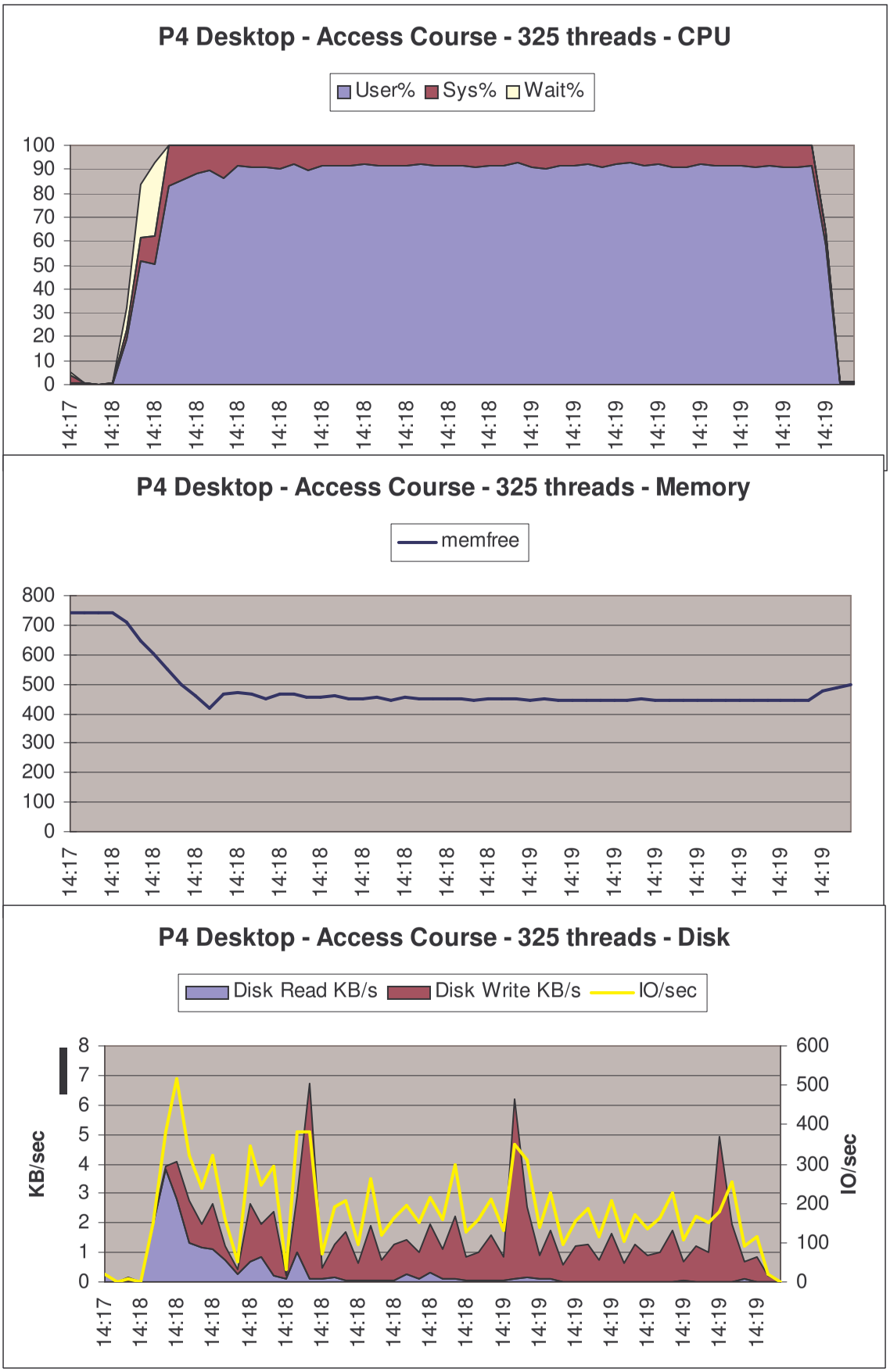


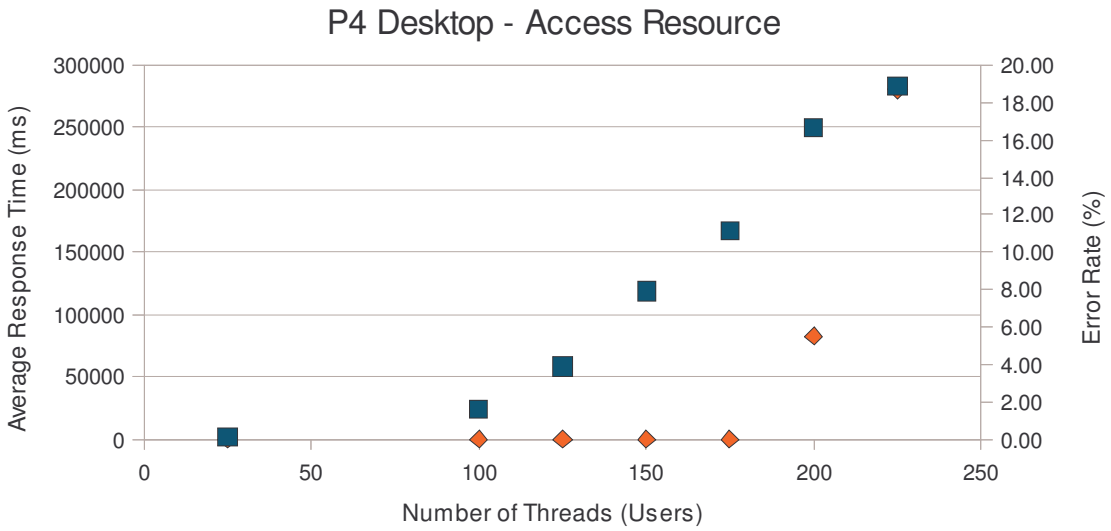

P4 Desktop - Log In - 475 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:49
13:49
13:49
13:49
13:49
13:49
13:49

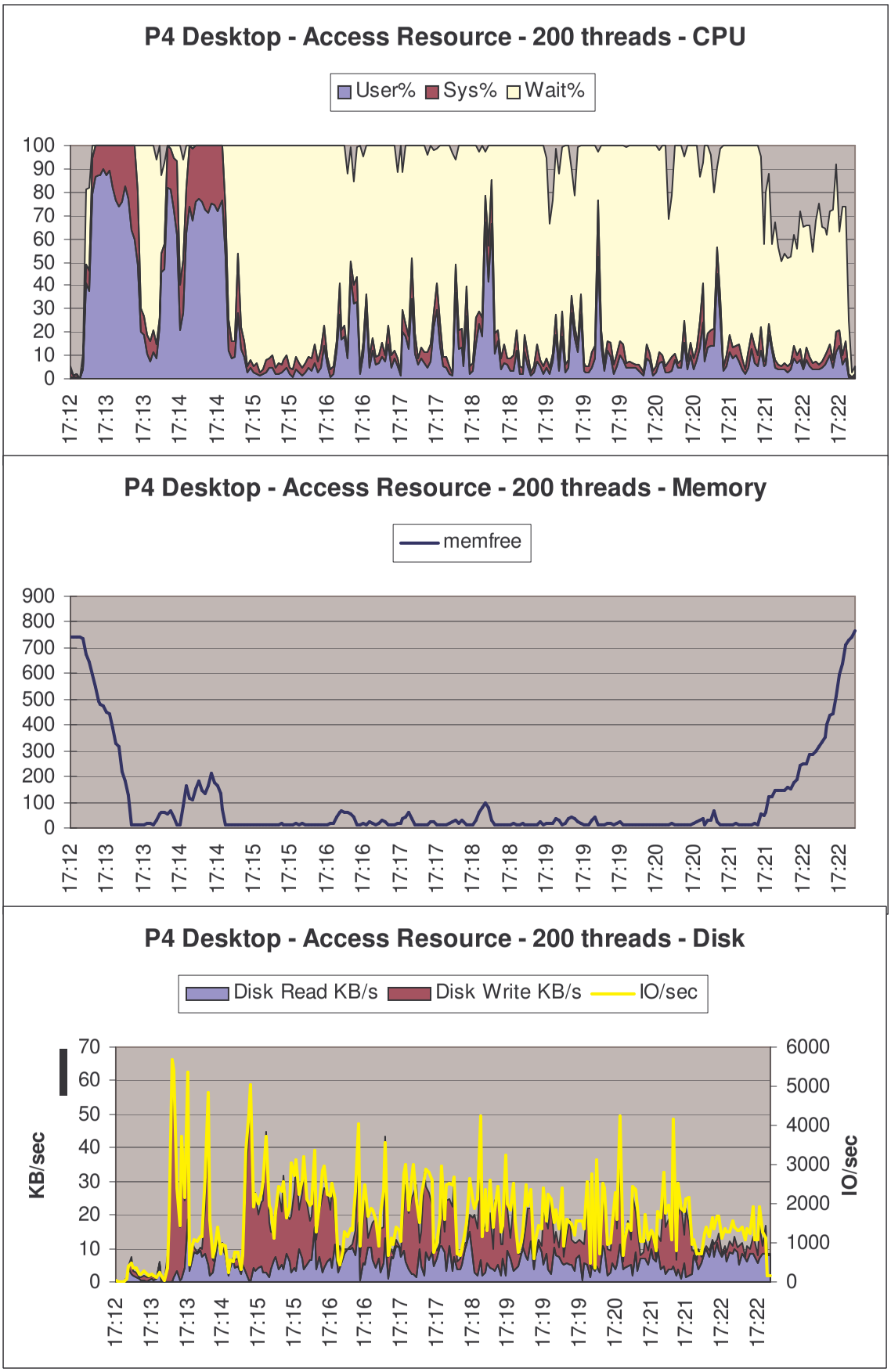

P4 Desktop - Log In - 475 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:49
13:49
13:49
13:49
13:49
13:49
13:49

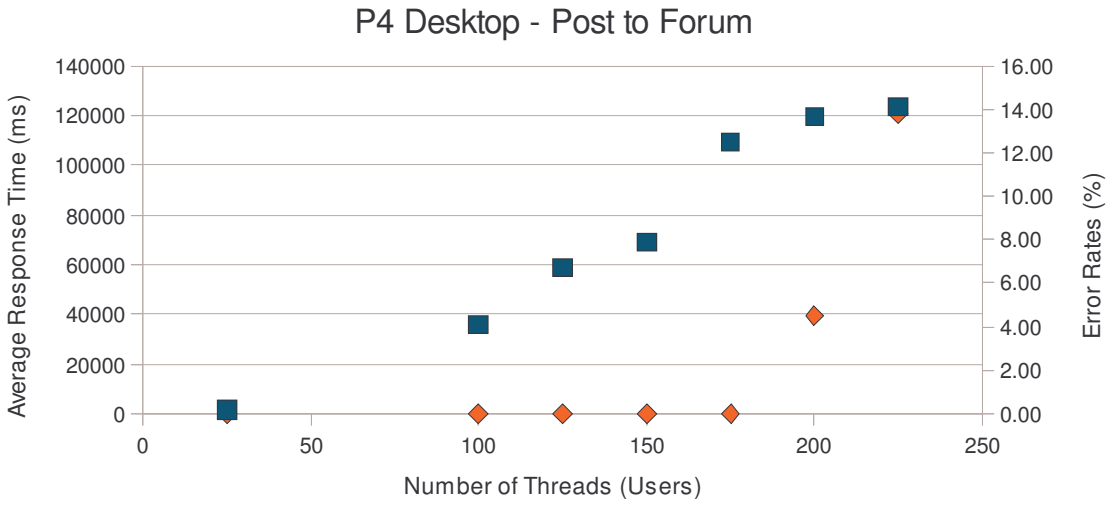

P4 Desktop - Log In - 475 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
7
6
5
4
3
2
1
0
500
450
400
350
300
250
200
150
100
50
0
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:48
13:49
13:49
13:49
13:49
13:49
13:49
13:49

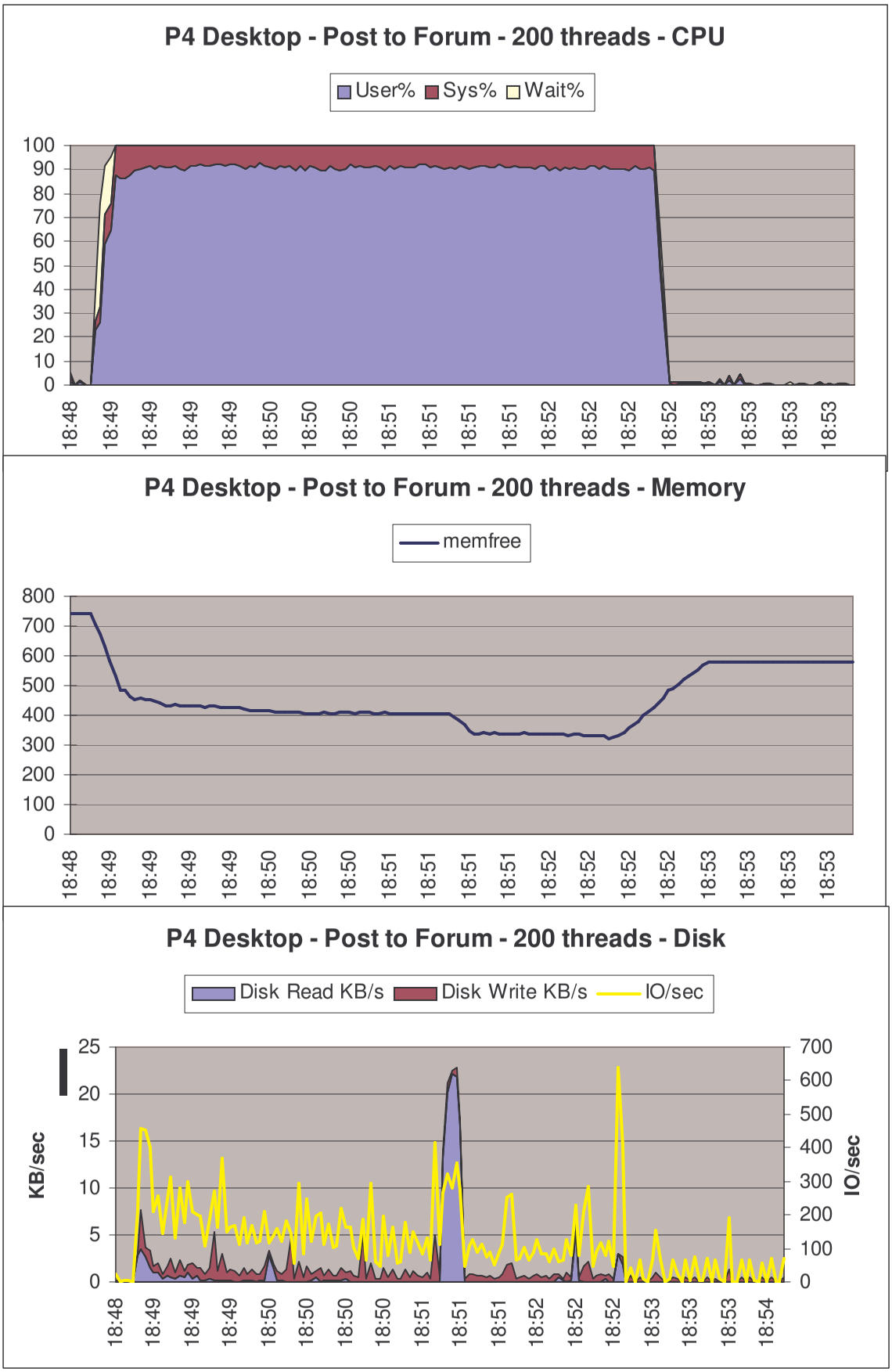

P4 Desktop - Access Course
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250
300
350
400

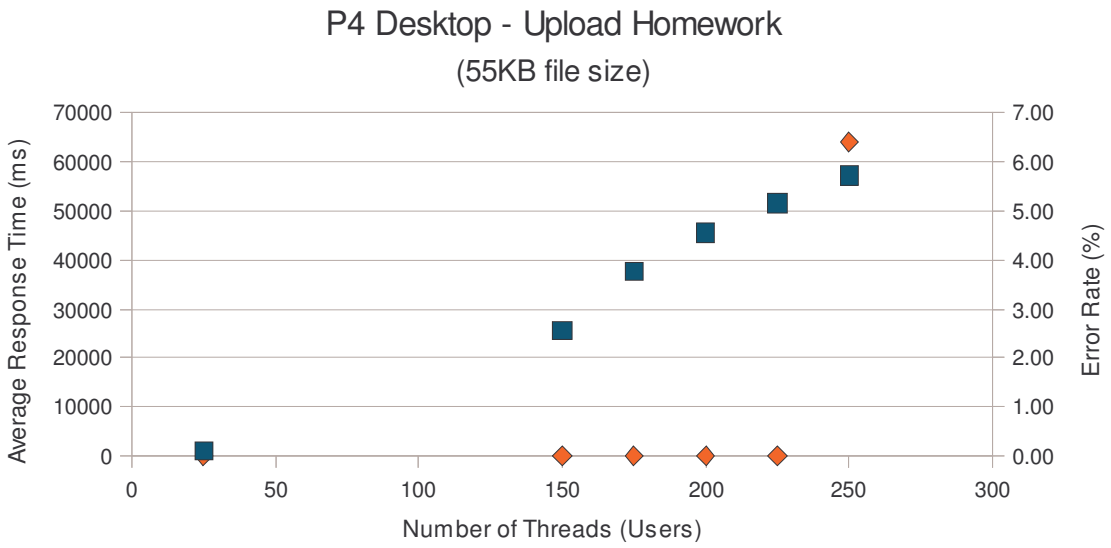

P4 Desktop - Access Course - 325 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
14:17
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:19
14:19
14:19
14:19
14:19
14:19
14:19
14:19
P4 Desktop - Access Course - 325 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
14:17
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:19
14:19
14:19
14:19
14:19
14:19
14:19
14:19
P4 Desktop - Access Course - 325 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
8
7
6
5
4
3
2
1
0
IO/sec
600
500
400
300
200
100
0
14:17
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:18
14:19
14:19
14:19
14:19
14:19
14:19
14:19
14:19

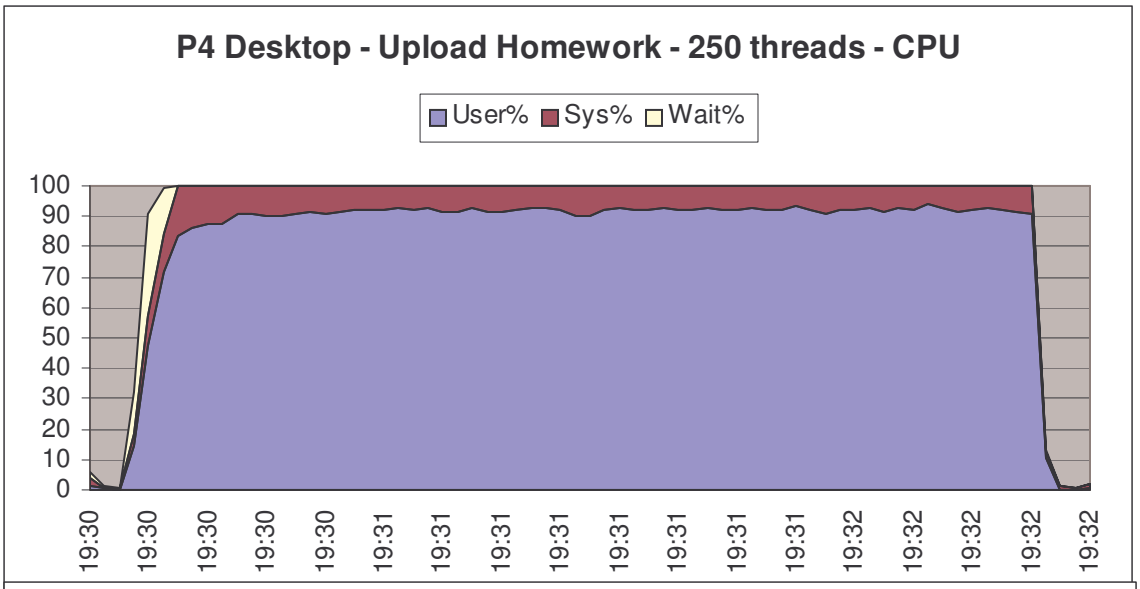

P4 Desktop - Access Resource
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
300000
250000
200000
150000
100000
50000
0
20.00
18.00
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250

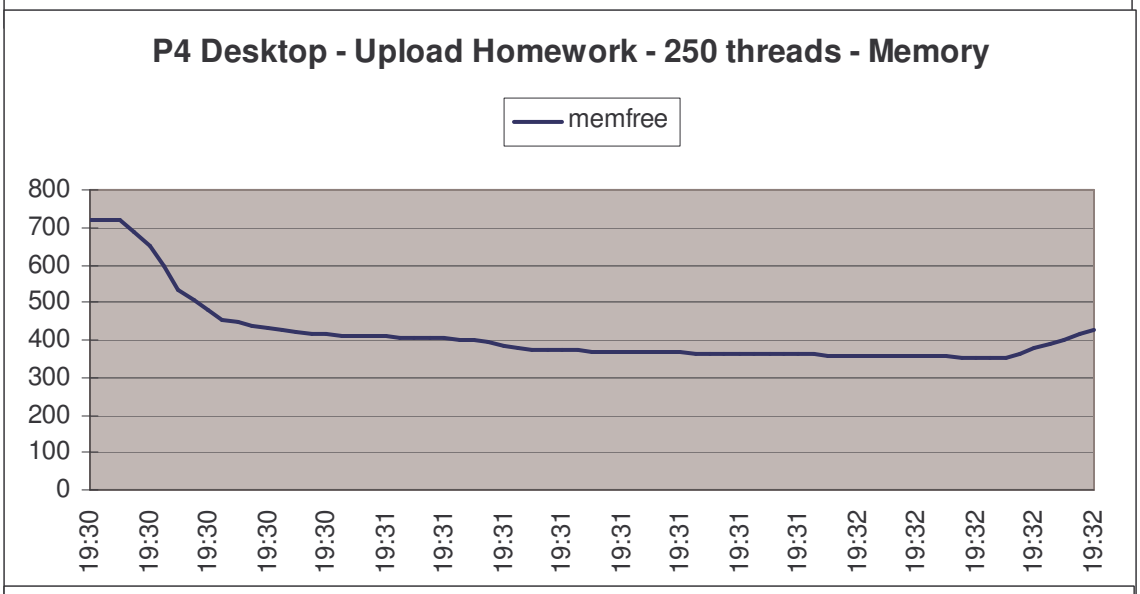

P4 Desktop - Access Resource - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
17:12
17:13
17:13
17:14
17:14
17:15
17:15
17:16
17:16
17:17
17:17
17:18
17:18
17:19
17:19
17:19
17:20
17:20
17:21
17:21
17:22
17:22
P4 Desktop - Access Resource - 200 threads - Memory
memfree
900
800
700
600
500
400
300
200
100
0
17:12
17:13
17:13
17:14
17:14
17:15
17:15
17:16
17:16
17:17
17:17
17:18
17:18
17:19
17:19
17:19
17:20
17:20
17:21
17:21
17:22
17:22
P4 Desktop - Access Resource - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
70
60
50
40
30
20
10
0
6000
5000
4000
3000
2000
1000
0
IO/sec
17:12
17:13
17:13
17:14
17:15
17:15
17:16
17:16
17:17
17:17
17:18
17:19
17:19
17:19
17:20
17:21
17:21
17:22
17:22

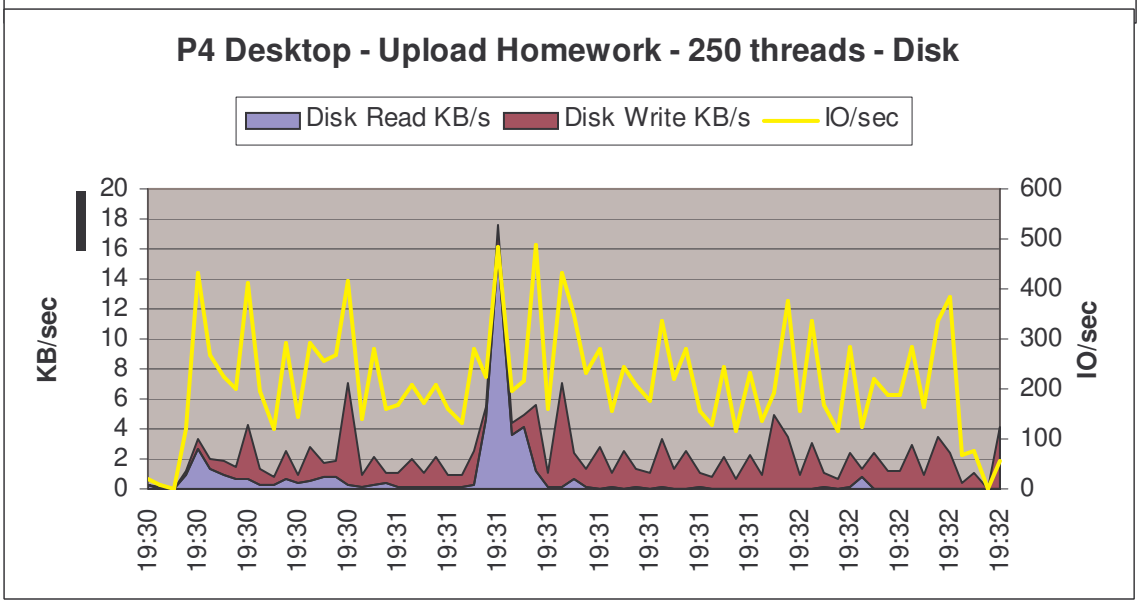

P4 Desktop - Post to Forum
Average Response Time (ms)
Error Rates (%)
Number of Threads (Users)
140000
120000
100000
80000
60000
40000
20000
0
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250

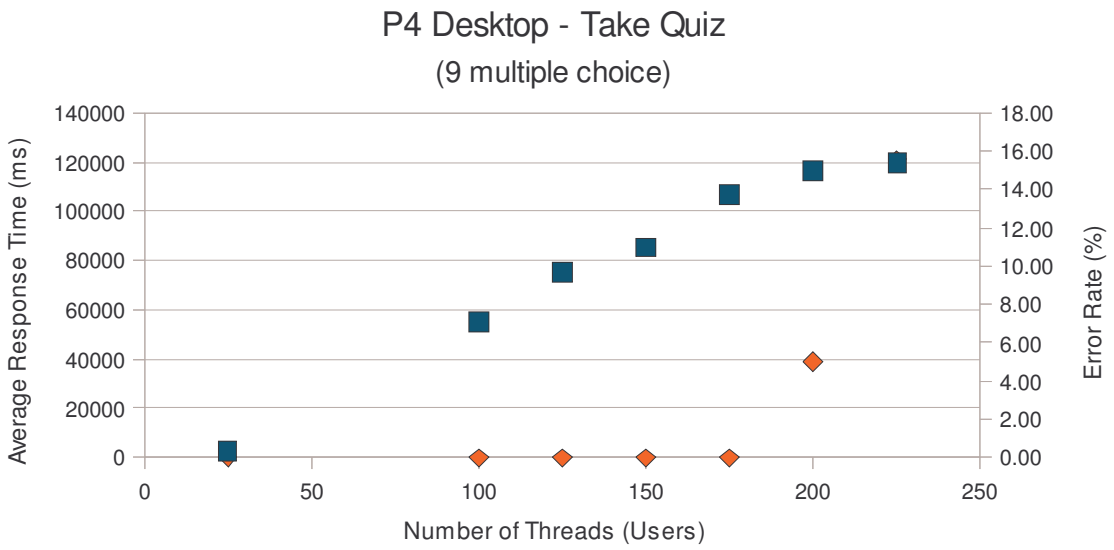

P4 Desktop - Post to Forum - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
18:48
18:49
18:49
18:49
18:49
18:50
18:50
18:50
18:51
18:51
18:51
18:51
18:52
18:52
18:52
18:52
18:53
18:53
18:53
18:53
P4 Desktop - Post to Forum - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
18:48
18:49
18:49
18:49
18:49
18:50
18:50
18:50
18:51
18:51
18:51
18:51
18:52
18:52
18:52
18:52
18:53
18:53
18:53
18:53
P4 Desktop - Post to Forum - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
25
20
15
10
5
0
700
600
500
400
300
200
100
0
18:48
18:49
18:49
18:49
18:50
18:50
18:50
18:50
18:51
18:51
18:51
18:52
18:52
18:52
18:53
18:53
18:53
18:54

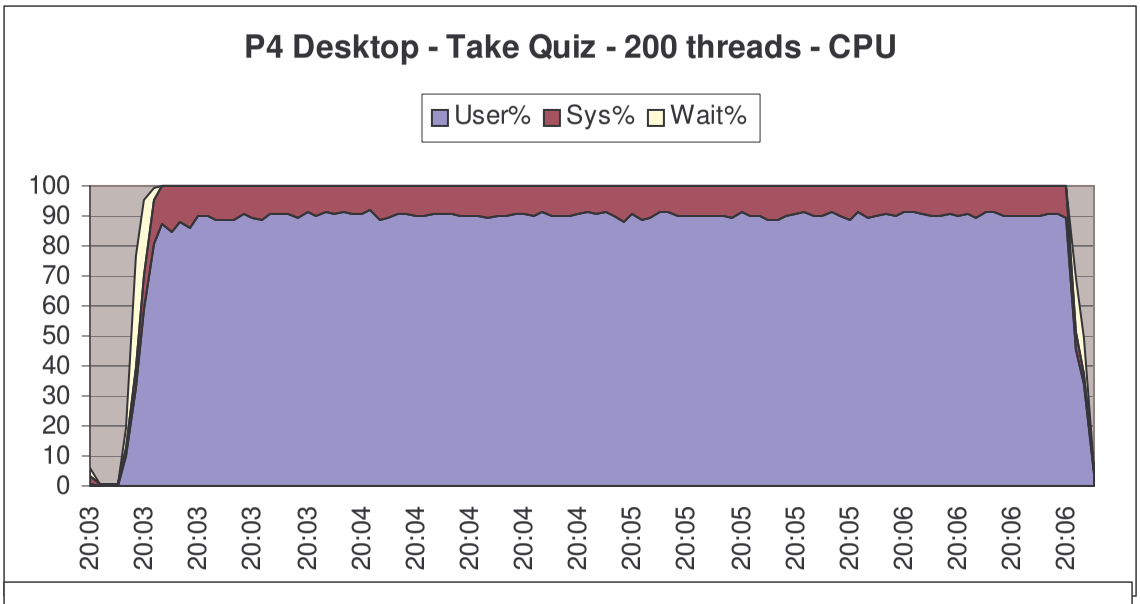

P4 Desktop - Upload Homework
(55KB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
70000
60000
50000
40000
30000
20000
10000
0
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
0
50
100
150
200
250
300

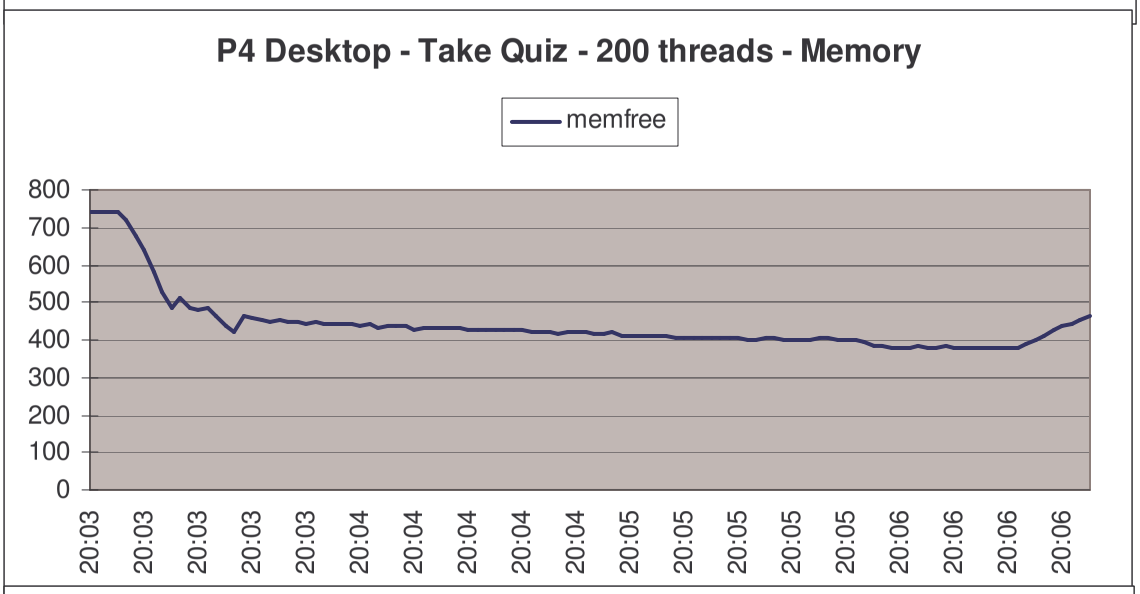

P4 Desktop - Upload Homework - 250 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
19:30
19:31
19:32

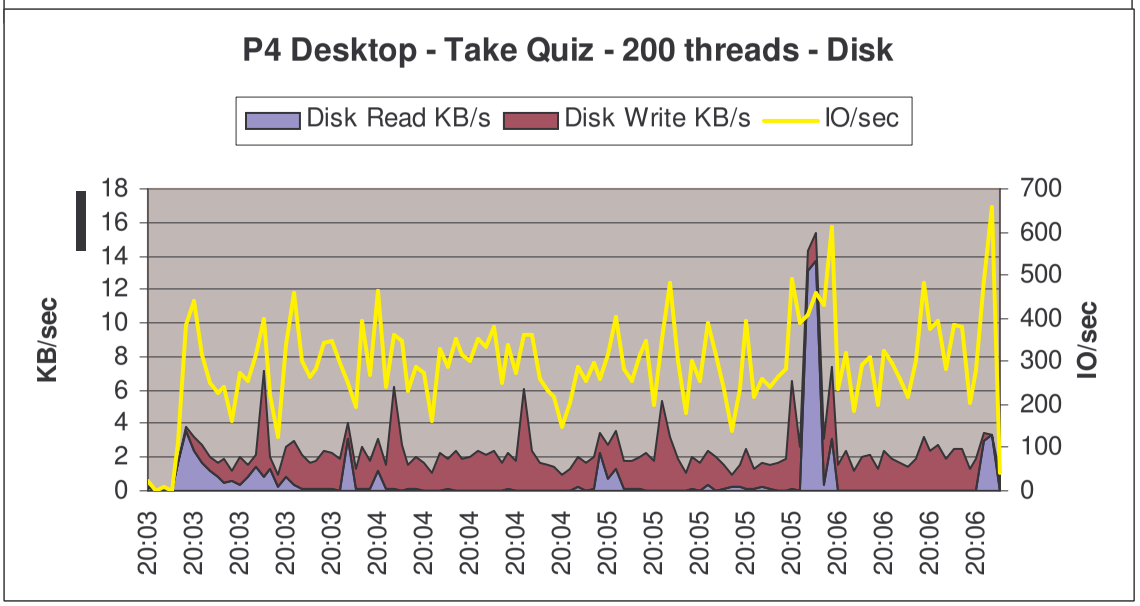

P4 Desktop - Upload Homework - 250 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
19:30
19:31
19:32


P4 Desktop - Upload Homework - 250 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
20
18
16
14
12
10
8
6
4
2
0
600
500
400
300
200
100
0
19:30
19:31
19:32

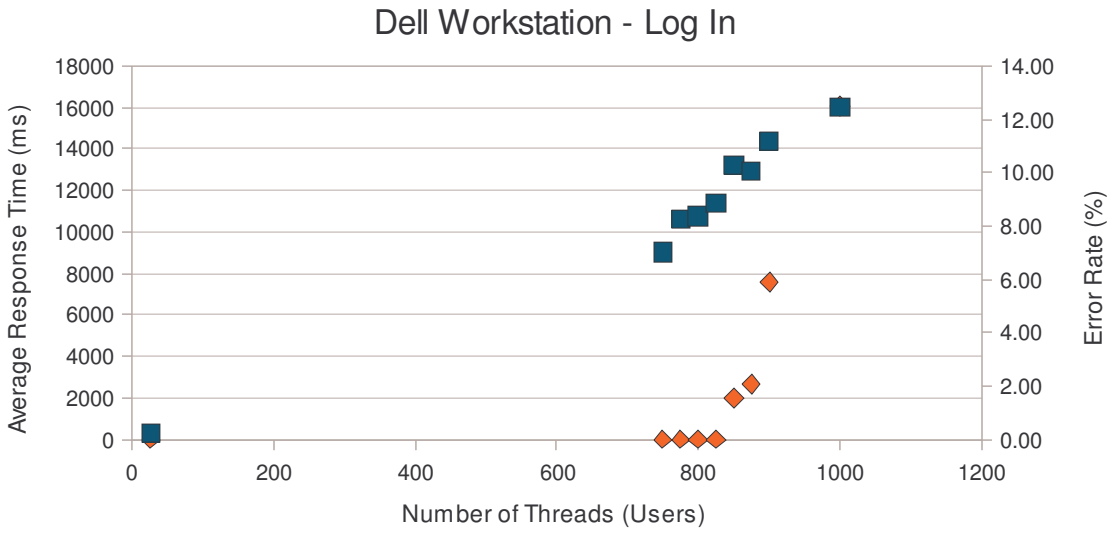

P4 Desktop - Take Quiz
(9 multiple choice)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
140000
120000
100000
80000
60000
40000
20000
0
18.00
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250

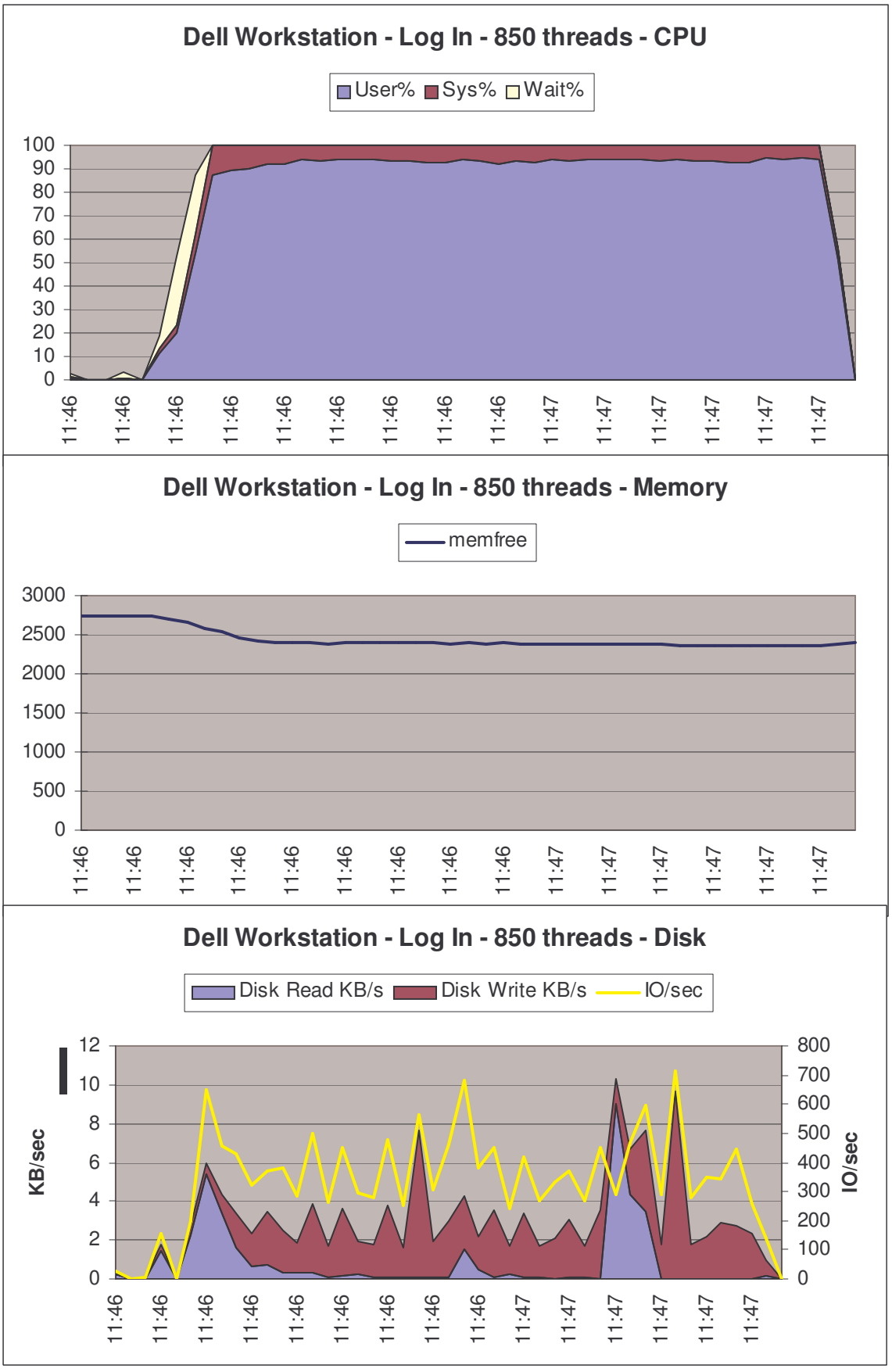

P4 Desktop - Take Quiz - 200 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
20:03
20:03
20:03
20:03
20:03
20:04
20:04
20:04
20:04
20:05
20:05
20:05
20:05
20:06
20:06
20:06
20:06

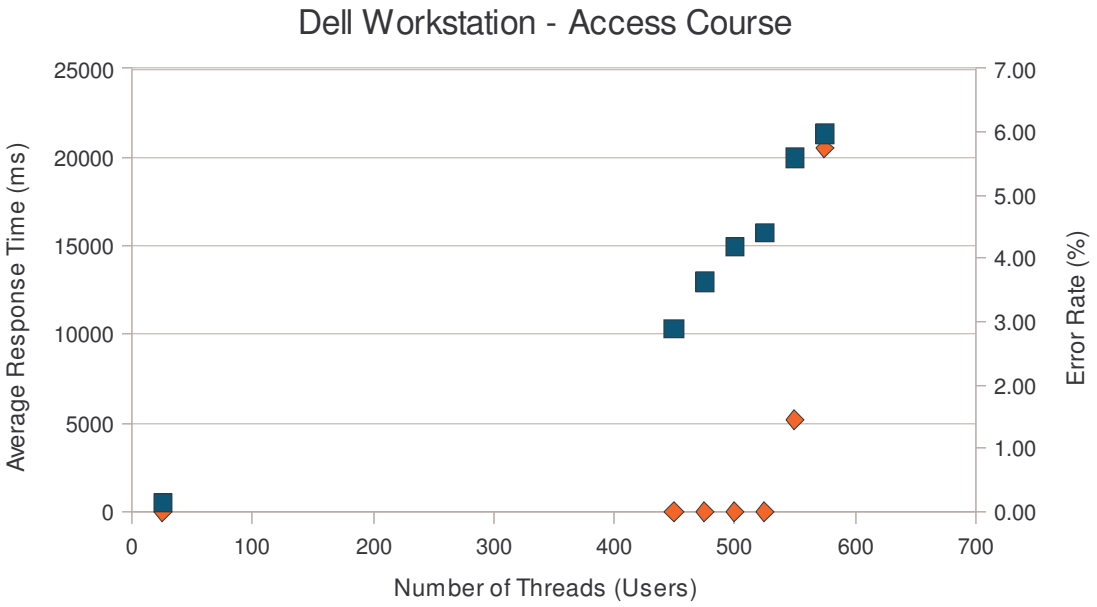

P4 Desktop - Take Quiz - 200 threads - Memory
memfree
800
700
600
500
400
300
200
100
0
20:03
20:03
20:03
20:03
20:03
20:04
20:04
20:04
20:04
20:05
20:05
20:05
20:05
20:06
20:06
20:06
20:06

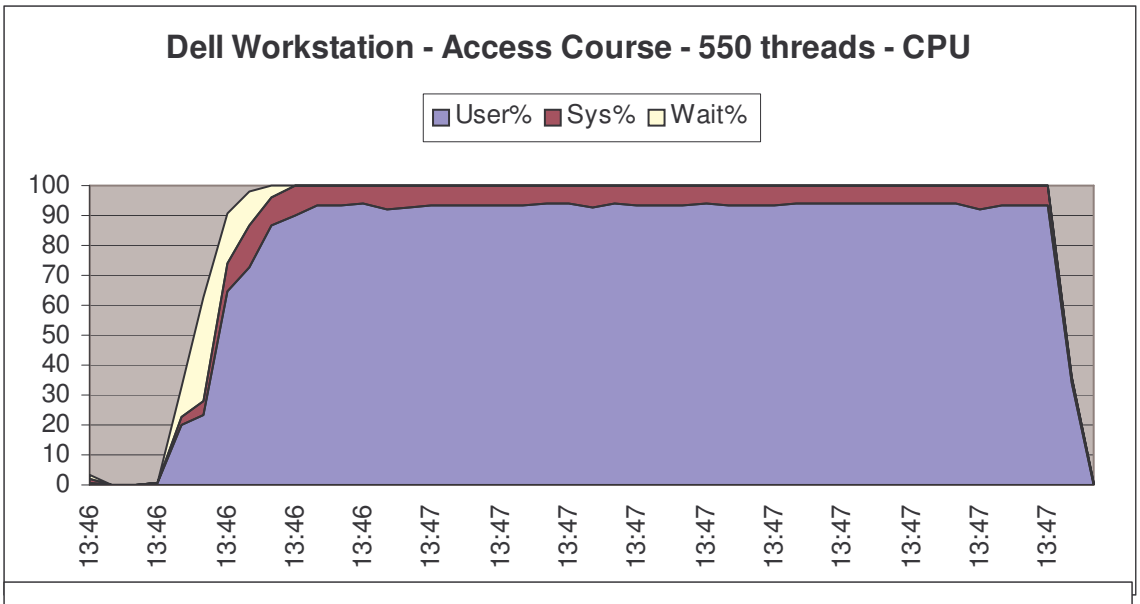

P4 Desktop - Take Quiz - 200 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
18
16
14
12
10
8
6
4
2
0
700
600
500
400
300
200
100
0
20:03
20:03
20:03
20:03
20:03
20:04
20:04
20:04
20:04
20:05
20:05
20:05
20:05
20:06
20:06
20:06
20:06

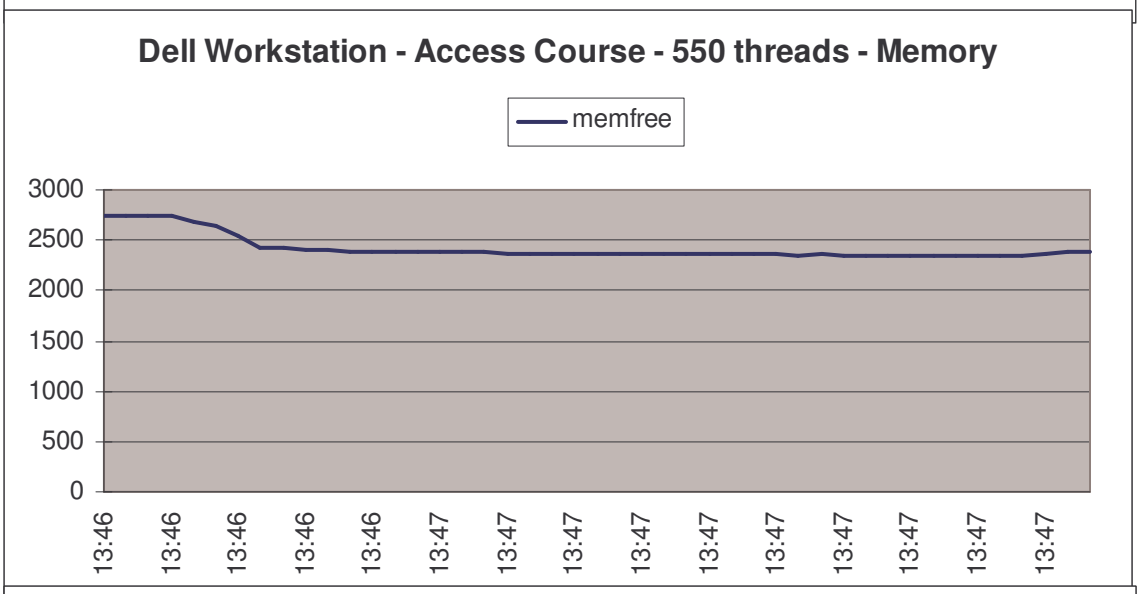
J.5 Dell Xeon Workstation

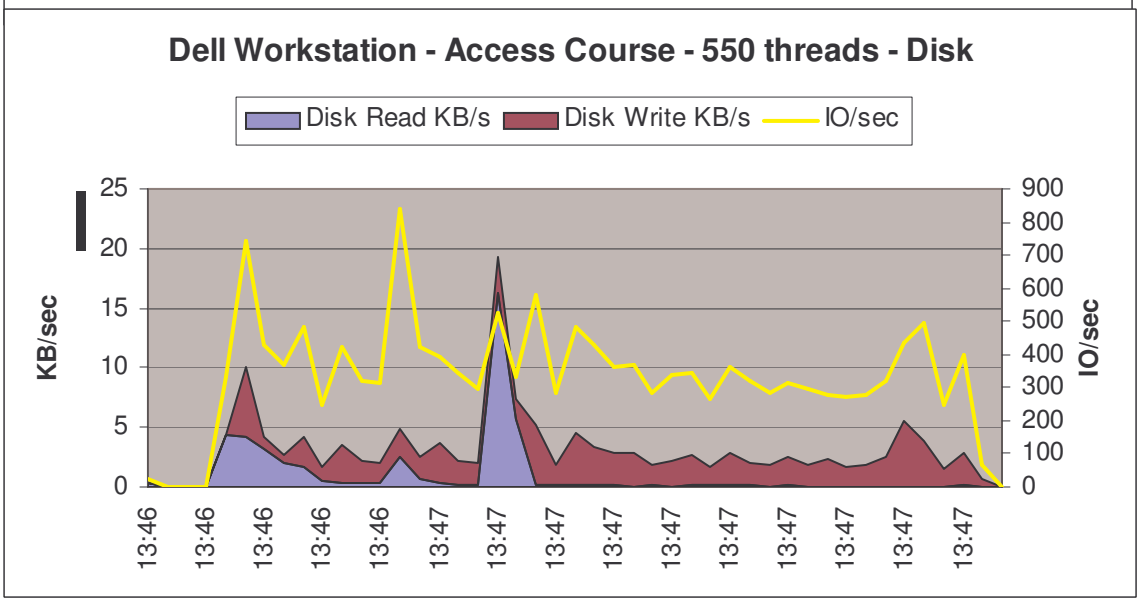


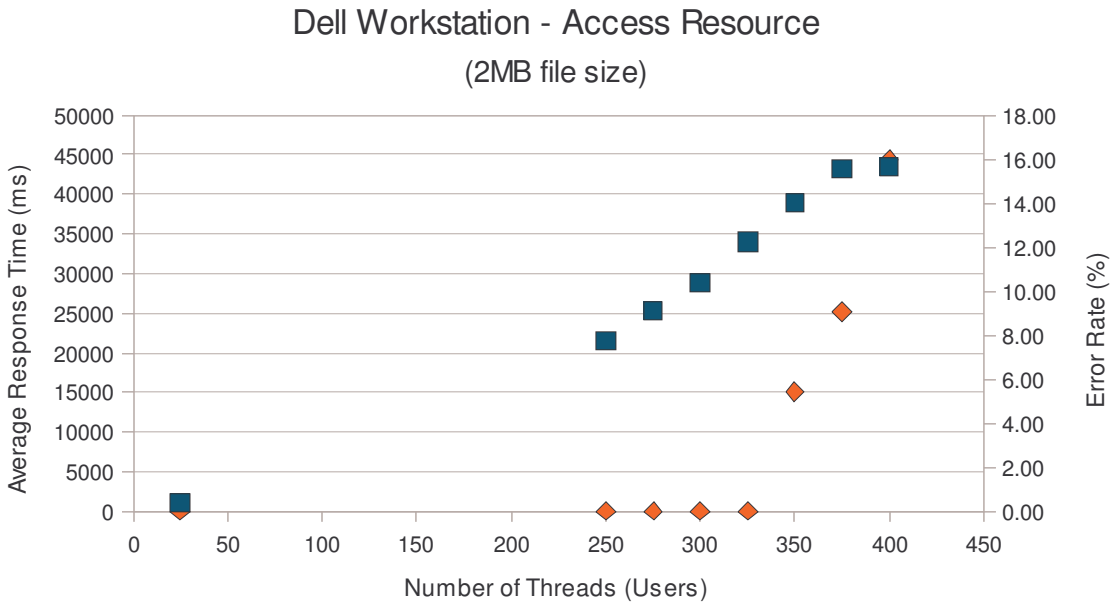

Dell Workstation - Log In - 850 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:47
11:47
11:47
11:47
11:47
11:47
Dell Workstation - Log In - 850 threads - Memory
memfree
3000
2500
2000
1500
1000
500
0
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:47
11:47
11:47
11:47
11:47
11:47
Dell Workstation - Log In - 850 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
12
10
8
6
4
2
0
800
700
600
500
400
300
200
100
0
IO/sec
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:46
11:47
11:47
11:47
11:47
11:47
11:47

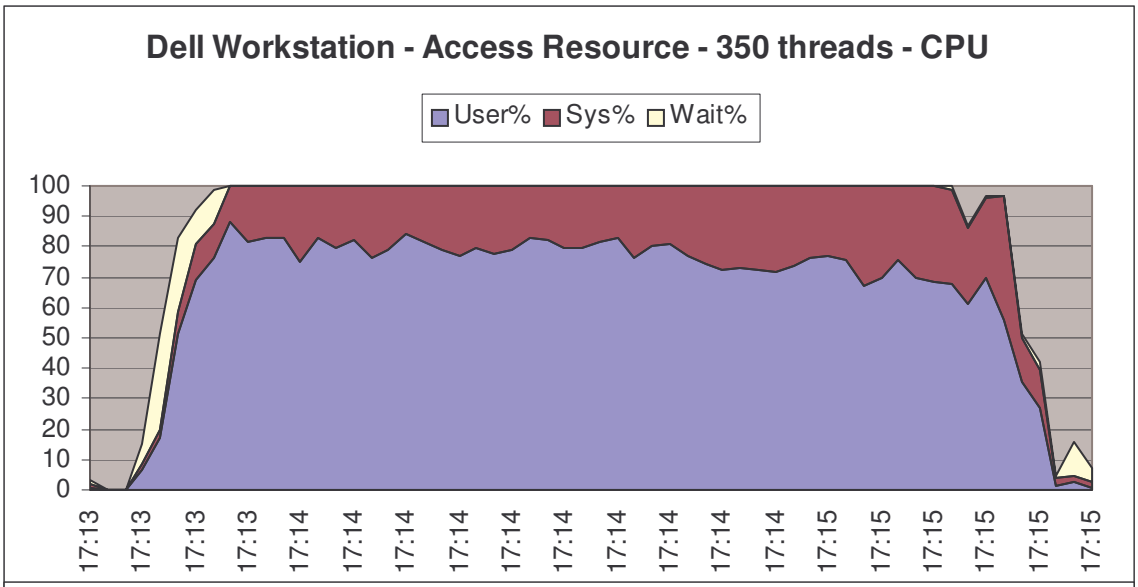

Dell Workstation - Access Course
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
25000
20000
15000
10000
5000
0
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
0
100
200
300
400
500
600
700

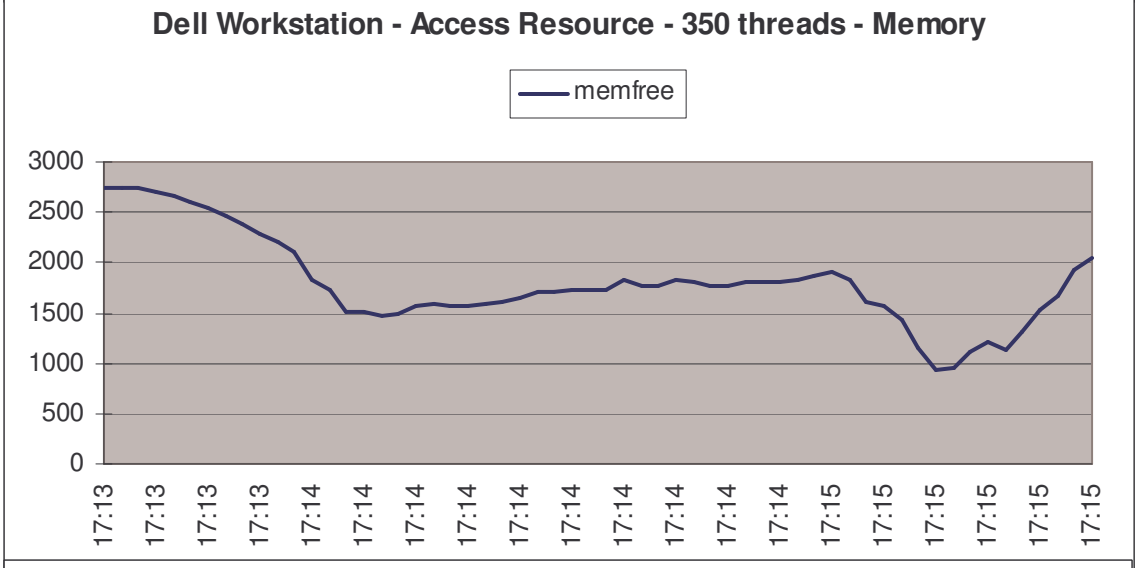

Dell Workstation - Access Course - 550 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
13:46
13:46
13:46
13:46
13:46
13:47
13:47
13:47
13:47
13:47
13:47
13:47
13:47
13:47

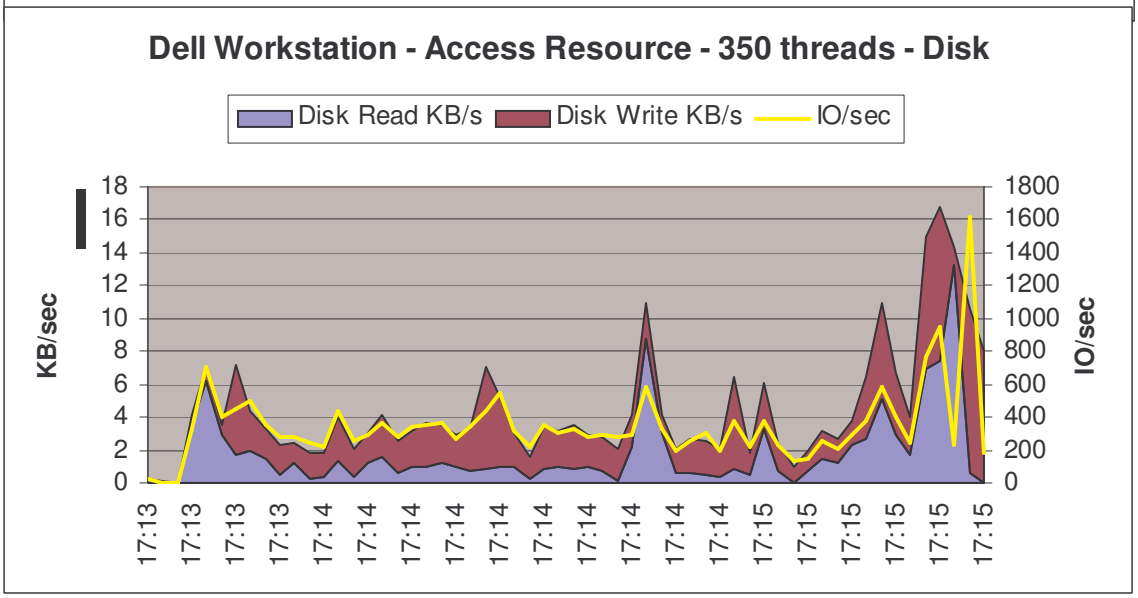

Dell Workstation - Access Course - 550 threads - Memory
memfree
3000
2500
2000
1500
1000
500
0
13:46
13:46
13:46
13:46
13:46
13:47
13:47
13:47
13:47
13:47
13:47
13:47
13:47
13:47

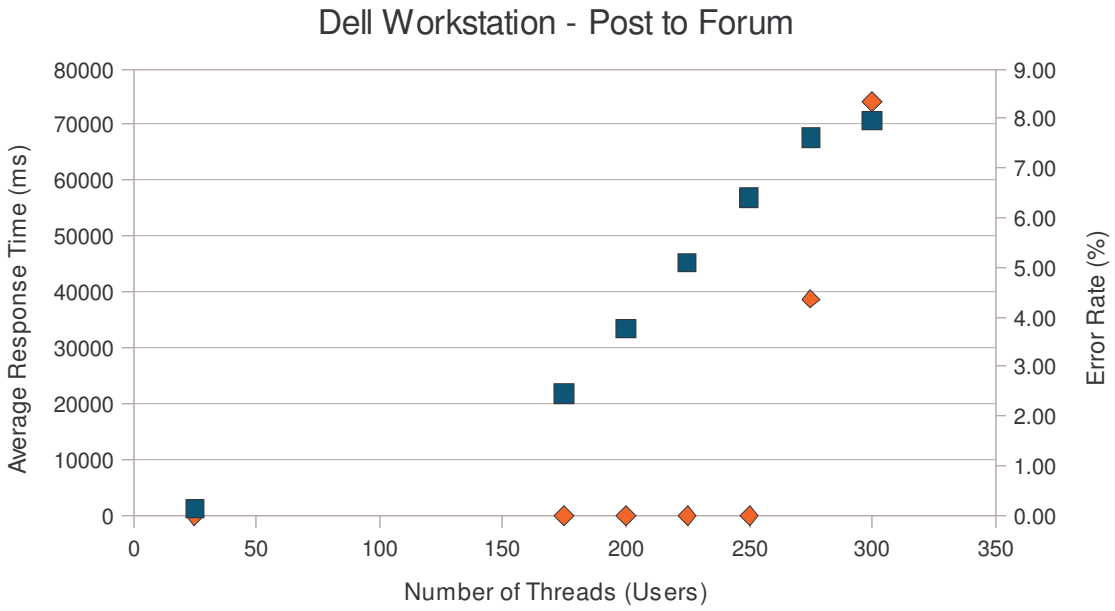

Dell Workstation - Access Course - 550 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
25
20
15
10
5
0
900
800
700
600
500
400
300
200
100
0
13:46
13:46
13:46
13:46
13:46
13:47
13:47
13:47
13:47
13:47
13:47
13:47
13:47
13:47

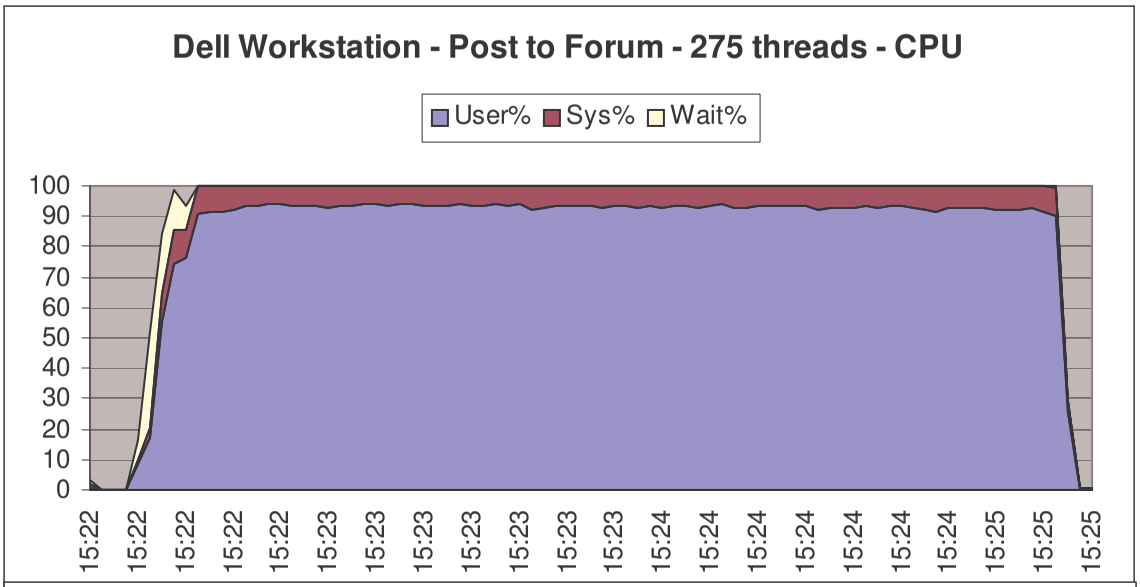

Dell Workstation - Access Resource
(2MB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
18.00
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250
300
350
400
450

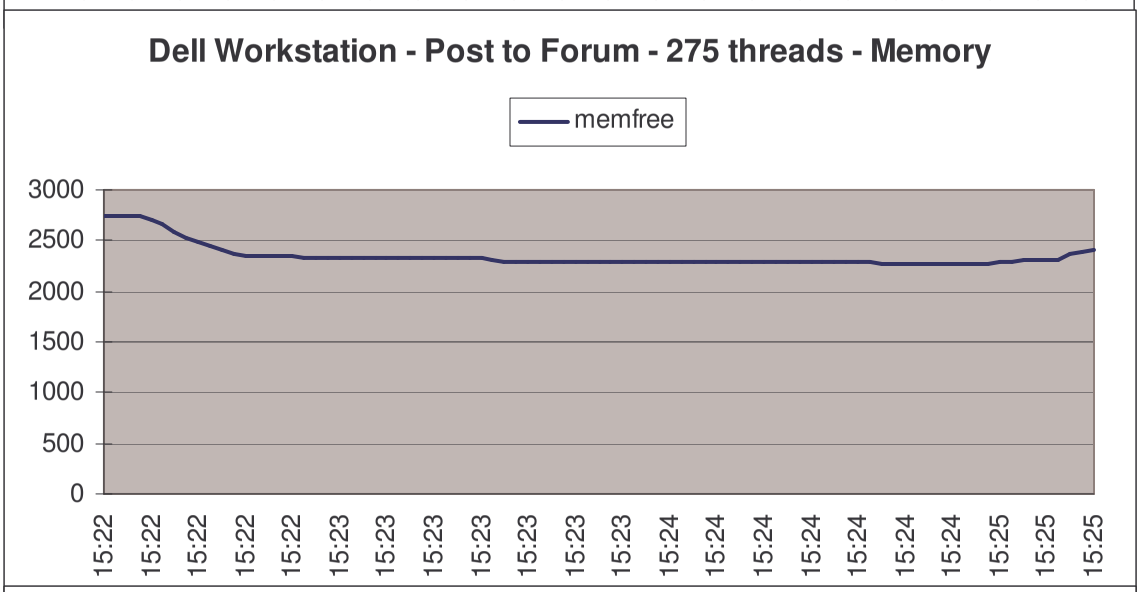

Dell Workstation - Access Resource - 350 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
17:13
17:13
17:13
17:13
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:15
17:15
17:15
17:15
17:15
17:15

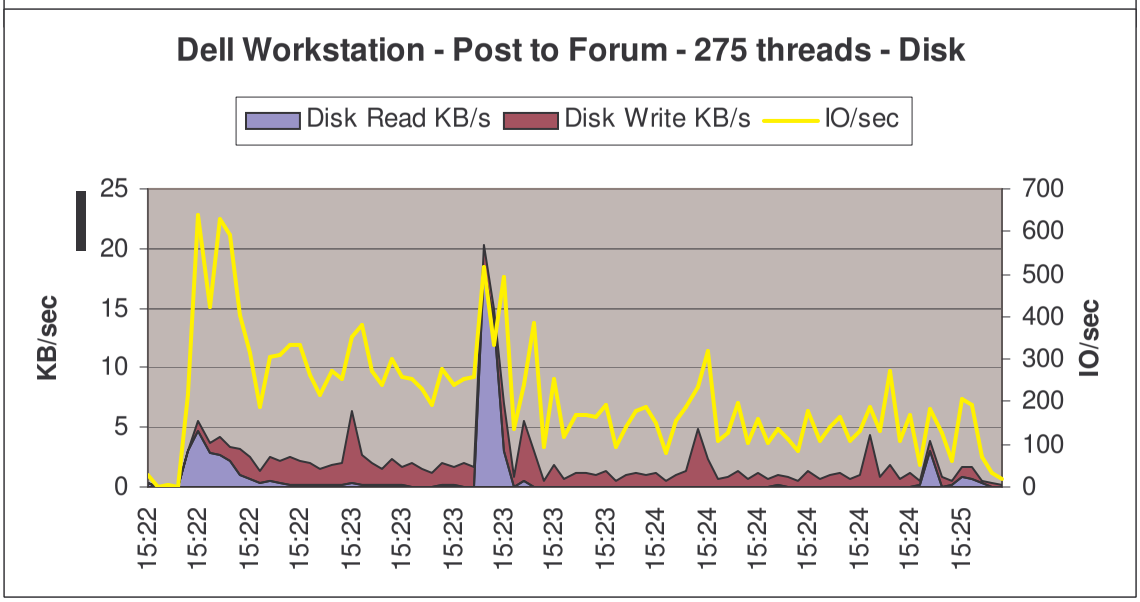

Dell Workstation - Access Resource - 350 threads - Memory
memfree
3000
2500
2000
1500
1000
500
0
17:13
17:13
17:13
17:13
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:15
17:15
17:15
17:15
17:15
17:15

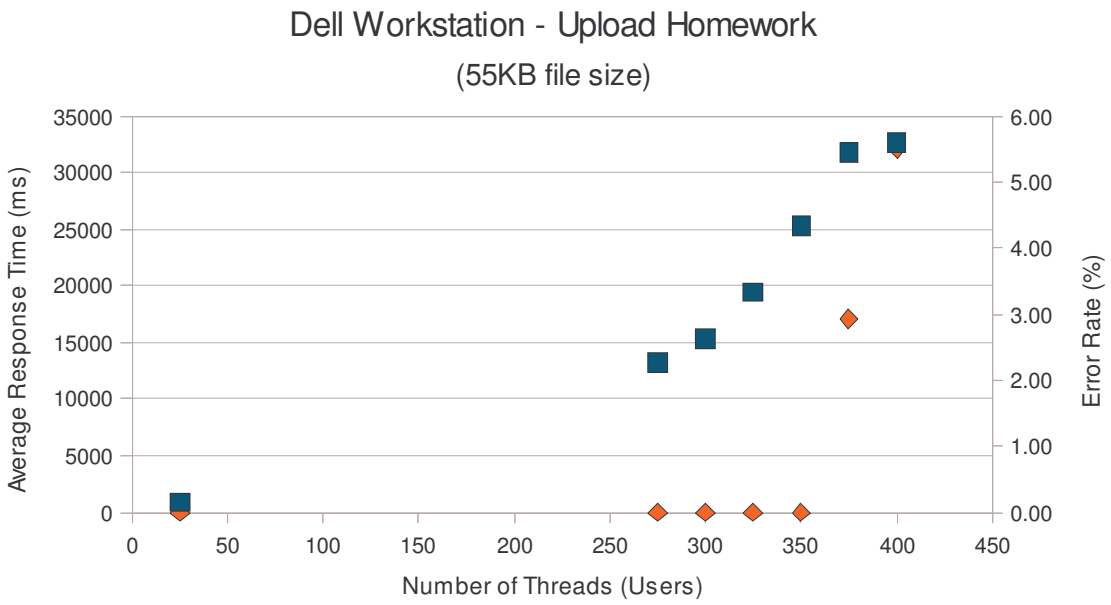

Dell Workstation - Access Resource - 350 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
18
16
14
12
10
8
6
4
2
0
1800
1600
1400
1200
1000
800
600
400
200
0
17:13
17:13
17:13
17:13
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:14
17:15
17:15
17:15
17:15
17:15
17:15

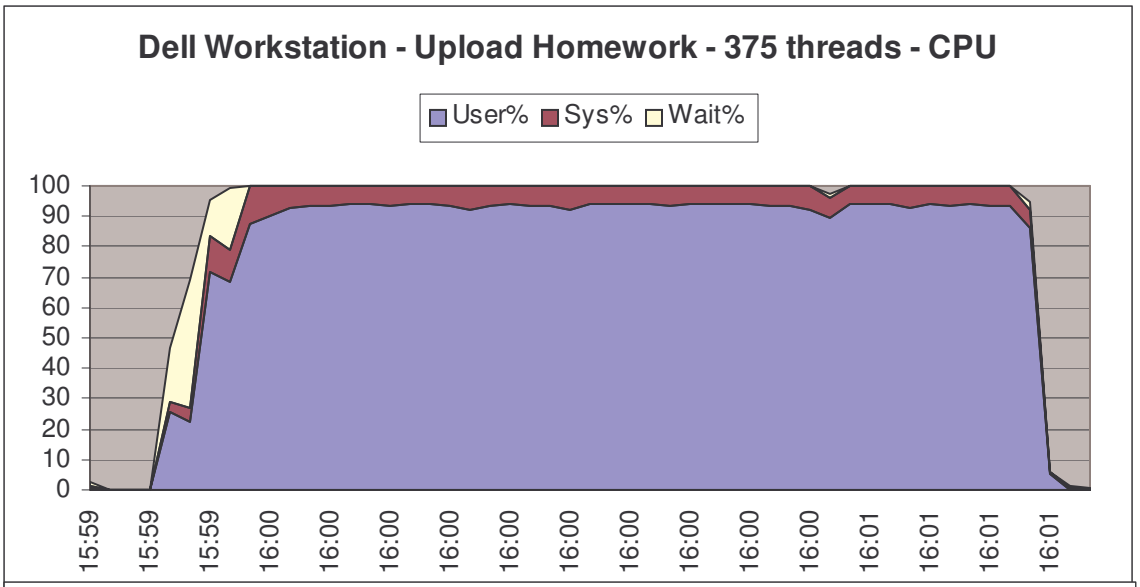

Dell Workstation - Post to Forum
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
80000
70000
60000
50000
40000
30000
20000
10000
0
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
0
50
100
150
200
250
300
350

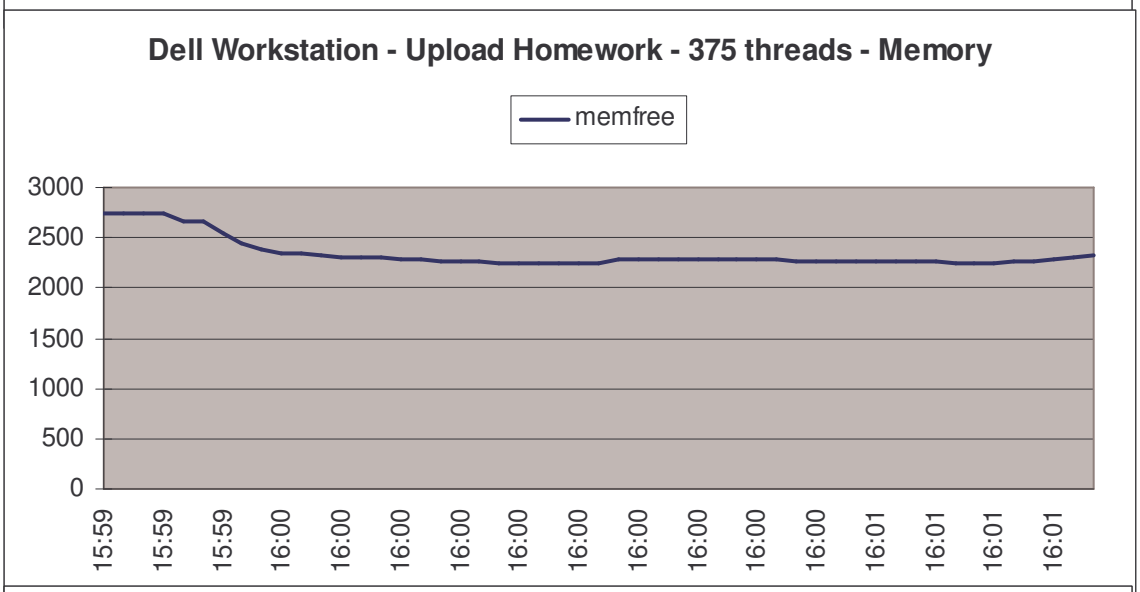

Dell Workstation - Post to Forum - 275 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
15:22
15:22
15:22
15:22
15:22
15:23
15:23
15:23
15:23
15:23
15:23
15:24
15:24
15:24
15:24
15:24
15:24
15:25
15:25
15:25

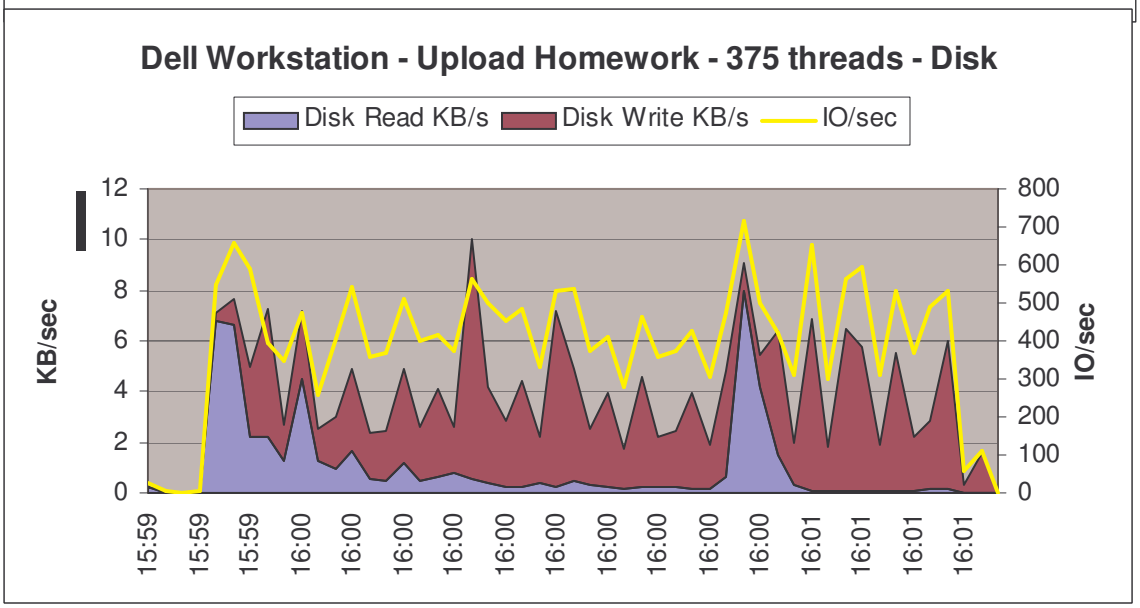

Dell Workstation - Post to Forum - 275 threads - Memory
memfree
3000
2500
2000
1500
1000
500
0
15:22
15:22
15:22
15:22
15:22
15:23
15:23
15:23
15:23
15:23
15:23
15:24
15:24
15:24
15:24
15:24
15:24
15:25
15:25
15:25

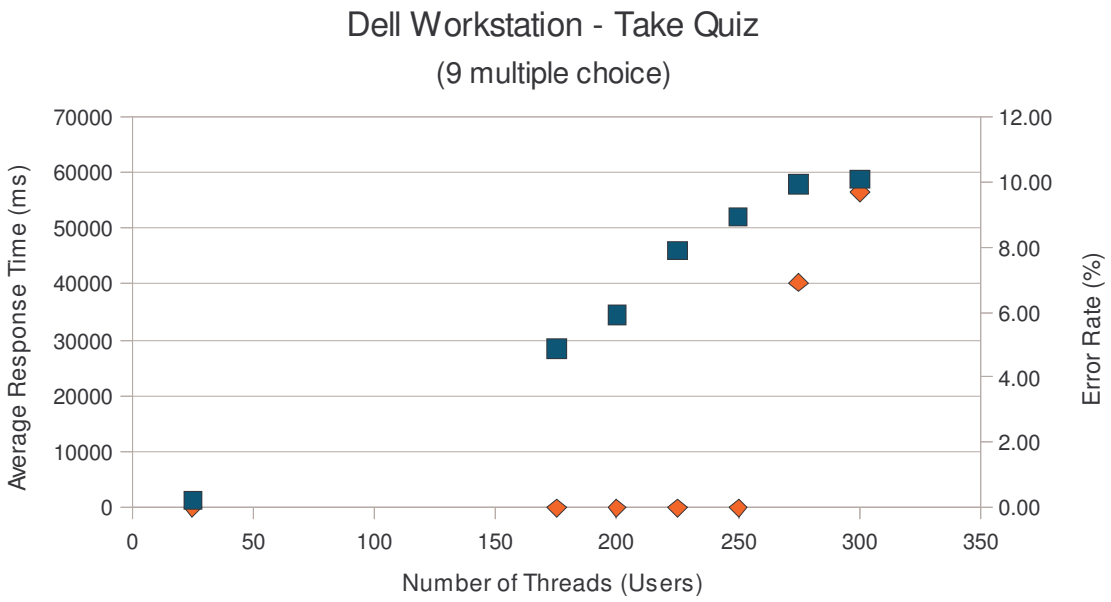

Dell Workstation - Post to Forum - 275 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
25
20
15
10
5
0
700
600
500
400
300
200
100
0
15:22
15:22
15:22
15:22
15:23
15:23
15:23
15:23
15:23
15:23
15:24
15:24
15:24
15:24
15:24
15:24
15:25

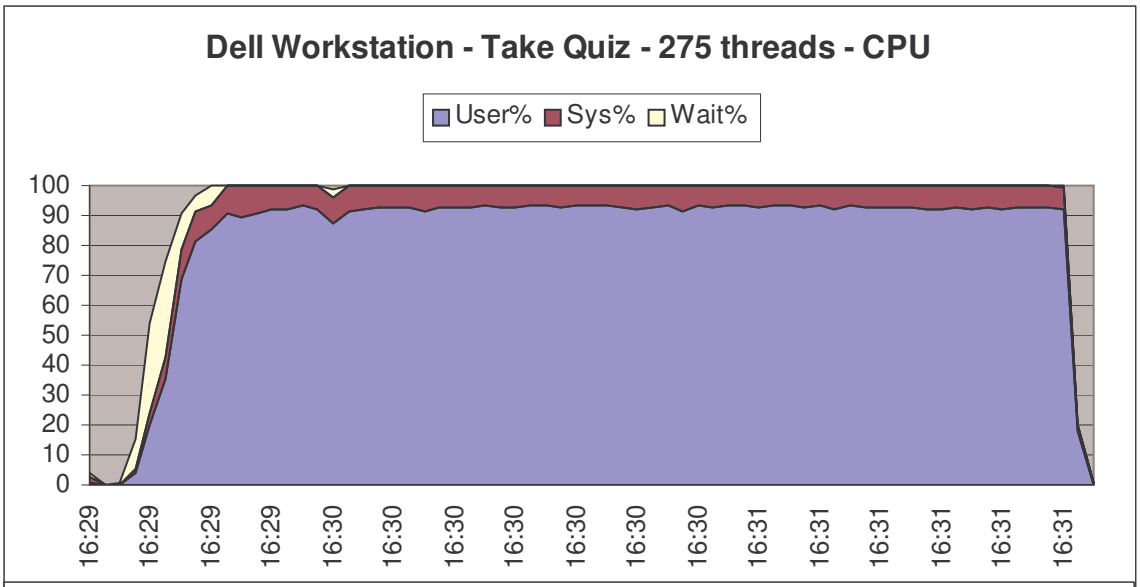

Dell Workstation - Upload Homework
(55KB file size)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
35000
30000
25000
20000
15000
10000
5000
0
6.00
5.00
4.00
3.00
2.00
1.00
0.00
0
50
100
150
200
250
300
350
400
450

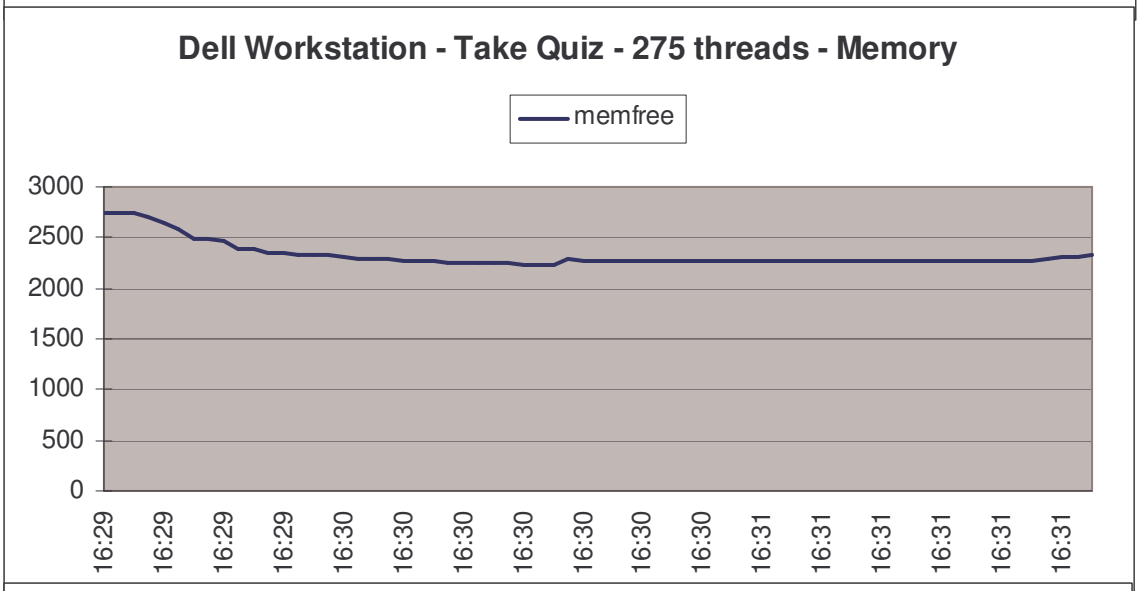

Dell Workstation - Upload Homework - 375 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
15:59
15:59
15:59
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:01
16:01
16:01
16:01

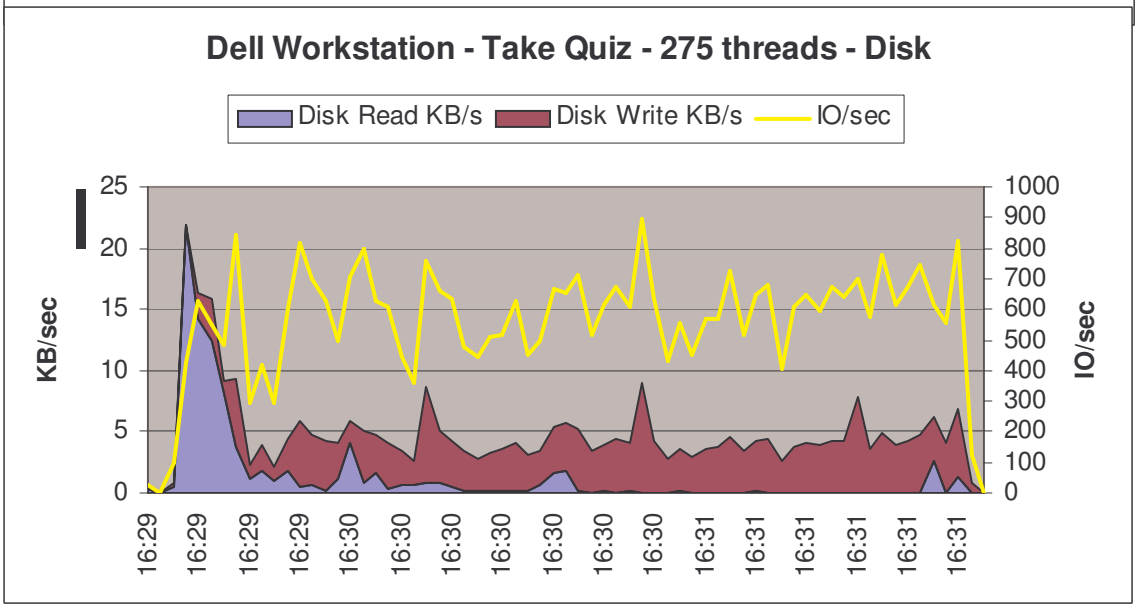

Dell Workstation - Upload Homework - 375 threads - Memory
memfree
3000
2500
2000
1500
1000
500
0
15:59
15:59
15:59
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:01
16:01
16:01
16:01


Dell Workstation - Upload Homework - 375 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
12
10
8
6
4
2
0
800
700
600
500
400
300
200
100
0
15:59
15:59
15:59
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:00
16:01
16:01
16:01
16:01


Dell Workstation - Take Quiz
(9 multiple choice)
Average Response Time (ms)
Error Rate (%)
Number of Threads (Users)
70000
60000
50000
40000
30000
20000
10000
0
12.00
10.00
8.00
6.00
4.00
2.00
0.00
0
50
100
150
200
250
300
350


Dell Workstation - Take Quiz - 275 threads - CPU
User%
Sys%
Wait%
100
90
80
70
60
50
40
30
20
10
0
16:29
16:29
16:29
16:29
16:30
16:30
16:30
16:30
16:30
16:30
16:31
16:31
16:31
16:31
16:31
16:31


Dell Workstation - Take Quiz - 275 threads - Memory
memfree
3000
2500
2000
1500
1000
500
0
16:29
16:29
16:29
16:29
16:30
16:30
16:30
16:30
16:30
16:30
16:31
16:31
16:31
16:31
16:31
16:31


Dell Workstation - Take Quiz - 275 threads - Disk
Disk Read KB/s
Disk Write KB/s
IO/sec
KB/sec
IO/sec
25
20
15
10
5
0
1000
900
800
700
600
500
400
300
200
100
0
16:29
16:29
16:29
16:29
16:30
16:30
16:30
16:30
16:30
16:30
16:31
16:31
16:31
16:31
16:31
16:31

APPENDIX K – JMeter Scripts

UsersLogIn.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="2.1">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="OLPC" enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables" elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments" testname="User Defined Variables" enabled="true">
        <collectionProp name="Arguments.arguments"/>
      </elementProp>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname="Users Log In" enabled="true">
        <elementProp name="ThreadGroup.main_controller" elementType="LoopController" guiclass="LoopControlPanel" testclass="LoopController" testname="Loop Controller" enabled="true">
          <boolProp name="LoopController.continue_forever">false</boolProp>
          <stringProp name="LoopController.loops">1</stringProp>
        </elementProp>
        <stringProp name="ThreadGroup.num_threads">75</stringProp>
        <stringProp name="ThreadGroup.ramp_time">60</stringProp>
        <longProp name="ThreadGroup.start_time">1271129323000</longProp>
        <longProp name="ThreadGroup.end_time">1271129323000</longProp>
        <boolProp name="ThreadGroup.scheduler">false</boolProp>
        <stringProp name="ThreadGroup.on_sample_error">stopthread</stringProp>
        <stringProp name="ThreadGroup.duration"></stringProp>
        <stringProp name="ThreadGroup.delay"></stringProp>
      </ThreadGroup>
      <hashTree>
        <ConfigTestElement guiclass="HttpDefaultsGui" testclass="ConfigTestElement" testname="HTTP Request Defaults" enabled="true">
          <elementProp name="HTTPsampler.Arguments" elementType="Arguments" guiclass="HTTPArgumentsPanel" testclass="Arguments" testname="User Defined Variables" enabled="true">
```

```
          <collectionProp name="Arguments.arguments"/>
        </elementProp>
        <stringProp
name="HTTPSampler.domain">schoolserver.example.org</stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp name="HTTPSampler.path"></stringProp>
      </ConfigTestElement>
      <hashTree/>
      <CookieManager guiclass="CookiePanel" testclass="CookieManager"
testname="HTTP Cookie Manager" enabled="true">
        <collectionProp name="CookieManager.cookies"/>
        <boolProp name="CookieManager.clearEachIteration">false</boolProp>
        <stringProp name="CookieManager.policy">rfc2109</stringProp>
      </CookieManager>
      <hashTree/>
      <CSVDataSet guiclass="TestBeanGUI" testclass="CSVDataSet"
testname="CSV Data Set Config" enabled="true">
        <stringProp name="delimiter">,</stringProp>
        <stringProp name="fileEncoding"></stringProp>
        <stringProp name="filename">jmeter_data_set.csv</stringProp>
        <boolProp name="quotedData">false</boolProp>
        <boolProp name="recycle">true</boolProp>
        <stringProp name="shareMode">All threads</stringProp>
        <boolProp name="stopThread">false</boolProp>
        <stringProp
name="variableNames">USER,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ADDR,
RANDOM_STRING</stringProp>
      </CSVDataSet>
      <hashTree/>
      <GaussianRandomTimer guiclass="GaussianRandomTimerGui"
testclass="GaussianRandomTimer" testname="Gaussian Random Timer"
enabled="true">
        <stringProp name="ConstantTimer.delay">300</stringProp>
        <stringProp name="RandomTimer.range">100.0</stringProp>
      </GaussianRandomTimer>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Log In" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
```

```
            <elementProp name="username" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">username</stringProp>
              <stringProp name="Argument.value">${USER}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="password" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">password</stringProp>
              <stringProp name="Argument.value">${PASSWORD}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="testcookies" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">testcookies</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/login/index.php</stringProp>
        <stringProp name="HTTPSampler.method">POST</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <ResultCollector                          guiclass="ViewResultsFullVisualizer"
testclass="ResultCollector" testname="View Results Tree" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
```

```
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="SummaryReport" testclass="ResultCollector"
testname="Summary Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
```

```
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector    guiclass="TableVisualizer"    testclass="ResultCollector"
testname="View Results in Table" enabled="true">
        <boolProp name="ResultCollector.error_logging">true</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
```

```
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="StatVisualizer" testclass="ResultCollector"
testname="Aggregate Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <threadCounts>true</threadCounts>
          </value>
        </objProp>
        <stringProp name="filename">fitpc_login_75.jtl</stringProp>
```

```
      </ResultCollector>
      <hashTree/>
     </hashTree>
   </hashTree>
  </hashTree>
</jmeterTestPlan>
```

UsersAccessCourse.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="2.1">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="OLPC"
enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables"
elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments"
testname="User Defined Variables" enabled="true">
        <collectionProp name="Arguments.arguments">
          <elementProp name="COURSE_ID" elementType="Argument">
            <stringProp name="Argument.name">COURSE_ID</stringProp>
            <stringProp name="Argument.value">5</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup"
testname="Users Access Course" enabled="true">
        <elementProp name="ThreadGroup.main_controller"
elementType="LoopController" guiclass="LoopControlPanel"
testclass="LoopController" testname="Loop Controller" enabled="true">
          <boolProp name="LoopController.continue_forever">false</boolProp>
          <stringProp name="LoopController.loops">1</stringProp>
        </elementProp>
        <stringProp name="ThreadGroup.num_threads">75</stringProp>
        <stringProp name="ThreadGroup.ramp_time">60</stringProp>
        <longProp name="ThreadGroup.start_time">1271129323000</longProp>
        <longProp name="ThreadGroup.end_time">1271129323000</longProp>
        <boolProp name="ThreadGroup.scheduler">false</boolProp>
        <stringProp
name="ThreadGroup.on_sample_error">stopthread</stringProp>
```

```
      <stringProp name="ThreadGroup.duration"></stringProp>
      <stringProp name="ThreadGroup.delay"></stringProp>
    </ThreadGroup>
    <hashTree>
      <ConfigTestElement guiclass="HttpDefaultsGui"
testclass="ConfigTestElement" testname="HTTP Request Defaults"
enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" testname="User Defined Variables" enabled="true">
          <collectionProp name="Arguments.arguments"/>
        </elementProp>
        <stringProp
name="HTTPSampler.domain">schoolserver.example.org</stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp name="HTTPSampler.path"></stringProp>
      </ConfigTestElement>
      <hashTree/>
      <CookieManager guiclass="CookiePanel" testclass="CookieManager"
testname="HTTP Cookie Manager" enabled="true">
        <collectionProp name="CookieManager.cookies"/>
        <stringProp name="TestPlan.comments">enable cookies</stringProp>
        <boolProp name="CookieManager.clearEachIteration">false</boolProp>
        <stringProp name="CookieManager.policy">rfc2109</stringProp>
      </CookieManager>
      <hashTree/>
      <CSVDataSet guiclass="TestBeanGUI" testclass="CSVDataSet"
testname="CSV Data Set Config" enabled="true">
        <stringProp name="delimiter">,</stringProp>
        <stringProp name="fileEncoding"></stringProp>
        <stringProp name="filename">jmeter_data_set.csv</stringProp>
        <boolProp name="quotedData">false</boolProp>
        <boolProp name="recycle">true</boolProp>
        <stringProp name="shareMode">All threads</stringProp>
        <boolProp name="stopThread">false</boolProp>
        <stringProp
name="variableNames">USER,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ADDR,
RANDOM_STRING</stringProp>
      </CSVDataSet>
      <hashTree/>
```

```
      <GaussianRandomTimer guiclass="GaussianRandomTimerGui"
testclass="GaussianRandomTimer" testname="Gaussian Random Timer"
enabled="true">
        <stringProp name="ConstantTimer.delay">300</stringProp>
        <stringProp name="RandomTimer.range">100.0</stringProp>
      </GaussianRandomTimer>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Log In" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="username" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">username</stringProp>
              <stringProp name="Argument.value">${USER}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="password" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">password</stringProp>
              <stringProp name="Argument.value">${PASSWORD}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="testcookies" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">testcookies</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/login/index.php</stringProp>
        <stringProp name="HTTPSampler.method">POST</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
```

```
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Course" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${COURSE_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">id</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/course/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
```

```
      <ResultCollector guiclass="ViewResultsFullVisualizer"
testclass="ResultCollector" testname="View Results Tree" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="SummaryReport" testclass="ResultCollector"
testname="Summary Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>false</timestamp>
            <success>true</success>
```

```
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="TableVisualizer" testclass="ResultCollector"
testname="View Results in Table" enabled="true">
        <boolProp name="ResultCollector.error_logging">true</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
```

```
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="StatVisualizer" testclass="ResultCollector"
testname="Aggregate Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <threadCounts>true</threadCounts>
```

```
          </value>
        </objProp>
        <stringProp name="filename">fitpc_access_course_75.jtl</stringProp>
      </ResultCollector>
      <hashTree/>
    </hashTree>
  </hashTree>
</hashTree>
</jmeterTestPlan>
```

UsersAccessResource.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="2.1">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="OLPC"
enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables"
elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments"
testname="User Defined Variables" enabled="true">
        <collectionProp name="Arguments.arguments">
          <elementProp name="COURSE_ID" elementType="Argument">
            <stringProp name="Argument.name">COURSE_ID</stringProp>
            <stringProp name="Argument.value">5</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="RESOURCE_ID" elementType="Argument">
            <stringProp name="Argument.name">RESOURCE_ID</stringProp>
            <stringProp name="Argument.value">15</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup"
testname="Users View Text Page" enabled="true">
        <elementProp name="ThreadGroup.main_controller"
elementType="LoopController" guiclass="LoopControlPanel"
testclass="LoopController" testname="Loop Controller" enabled="true">
          <boolProp name="LoopController.continue_forever">false</boolProp>
          <stringProp name="LoopController.loops">1</stringProp>
```

```
      </elementProp>
      <stringProp name="ThreadGroup.num_threads">50</stringProp>
      <stringProp name="ThreadGroup.ramp_time">60</stringProp>
      <longProp name="ThreadGroup.start_time">1271129323000</longProp>
      <longProp name="ThreadGroup.end_time">1271129323000</longProp>
      <boolProp name="ThreadGroup.scheduler">false</boolProp>
      <stringProp
name="ThreadGroup.on_sample_error">stopthread</stringProp>
      <stringProp name="ThreadGroup.duration"></stringProp>
      <stringProp name="ThreadGroup.delay"></stringProp>
    </ThreadGroup>
    <hashTree>
      <ConfigTestElement                                guiclass="HttpDefaultsGui"
testclass="ConfigTestElement"       testname="HTTP       Request       Defaults"
enabled="true">
        <elementProp                                 name="HTTPsampler.Arguments"
elementType="Arguments"                               guiclass="HTTPArgumentsPanel"
testclass="Arguments" testname="User Defined Variables" enabled="true">
          <collectionProp name="Arguments.arguments"/>
        </elementProp>
        <stringProp
name="HTTPSampler.domain">schoolserver.example.org</stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp name="HTTPSampler.path"></stringProp>
      </ConfigTestElement>
      <hashTree/>
      <CookieManager       guiclass="CookiePanel"       testclass="CookieManager"
testname="HTTP Cookie Manager" enabled="true">
        <collectionProp name="CookieManager.cookies"/>
        <stringProp name="TestPlan.comments">enable cookies</stringProp>
        <boolProp name="CookieManager.clearEachIteration">false</boolProp>
        <stringProp name="CookieManager.policy">rfc2109</stringProp>
      </CookieManager>
      <hashTree/>
      <CSVDataSet          guiclass="TestBeanGUI"          testclass="CSVDataSet"
testname="CSV Data Set Config" enabled="true">
        <stringProp name="delimiter">,</stringProp>
        <stringProp name="fileEncoding"></stringProp>
        <stringProp name="filename">jmeter_data_set.csv</stringProp>
        <boolProp name="quotedData">false</boolProp>
        <boolProp name="recycle">true</boolProp>
        <stringProp name="shareMode">All threads</stringProp>
```

```
        <boolProp name="stopThread">false</boolProp>
        <stringProp
name="variableNames">USER,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ADDR,
RANDOM_STRING</stringProp>
      </CSVDataSet>
      <hashTree/>
      <GaussianRandomTimer guiclass="GaussianRandomTimerGui"
testclass="GaussianRandomTimer" testname="Gaussian Random Timer"
enabled="true">
        <stringProp name="ConstantTimer.delay">300</stringProp>
        <stringProp name="RandomTimer.range">100.0</stringProp>
      </GaussianRandomTimer>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Log In" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="username" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">username</stringProp>
              <stringProp name="Argument.value">${USER}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="password" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">password</stringProp>
              <stringProp name="Argument.value">${PASSWORD}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="testcookies" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">testcookies</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
```

```
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/login/index.php</stringProp>
        <stringProp name="HTTPSampler.method">POST</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Course" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${COURSE_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">id</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/course/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
```

```
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Resource" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${RESOURCE_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">id</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">moodle/mod/resource/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <ResultCollector guiclass="ViewResultsFullVisualizer"
testclass="ResultCollector" testname="View Results Tree" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
```

```
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector    guiclass="SummaryReport"    testclass="ResultCollector"
testname="Summary Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
```

```
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="TableVisualizer" testclass="ResultCollector"
testname="View Results in Table" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
```

```
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
           </value>
         </objProp>
         <stringProp name="filename"></stringProp>
       </ResultCollector>
       <hashTree/>
       <ResultCollector guiclass="StatVisualizer" testclass="ResultCollector"
testname="Aggregate Report" enabled="true">
         <boolProp name="ResultCollector.error_logging">false</boolProp>
         <objProp>
           <name>saveConfig</name>
           <value class="SampleSaveConfiguration">
             <time>true</time>
             <latency>true</latency>
             <timestamp>true</timestamp>
             <success>true</success>
             <label>true</label>
             <code>true</code>
             <message>true</message>
             <threadName>true</threadName>
             <dataType>true</dataType>
             <encoding>false</encoding>
             <assertions>true</assertions>
             <subresults>true</subresults>
             <responseData>false</responseData>
             <samplerData>false</samplerData>
             <xml>true</xml>
             <fieldNames>false</fieldNames>
             <responseHeaders>false</responseHeaders>
             <requestHeaders>false</requestHeaders>
             <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
             <assertionsResultsToSave>0</assertionsResultsToSave>
             <bytes>true</bytes>
             <threadCounts>true</threadCounts>
           </value>
         </objProp>
         <stringProp name="filename">fitpc_access_resource_50.jtl</stringProp>
       </ResultCollector>
       <hashTree/>
     </hashTree>
   </hashTree>
 </hashTree>
```

```
</jmeterTestPlan>
```

UsersPostForum.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="2.1">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="OLPC"
enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables"
elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments"
testname="User Defined Variables" enabled="true">
        <collectionProp name="Arguments.arguments">
          <elementProp name="COURSE_ID" elementType="Argument">
            <stringProp name="Argument.name">COURSE_ID</stringProp>
            <stringProp name="Argument.value">7</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="FORUM_ID" elementType="Argument">
            <stringProp name="Argument.name">FORUM_ID</stringProp>
            <stringProp name="Argument.value">20</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="DISCUSSION_ID" elementType="Argument">
            <stringProp name="Argument.name">DISCUSSION_ID</stringProp>
            <stringProp name="Argument.value">11</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="REPLY_ID" elementType="Argument">
            <stringProp name="Argument.name">REPLY_ID</stringProp>
            <stringProp name="Argument.value">556</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="USER_ID" elementType="Argument">
            <stringProp name="Argument.name">USER_ID</stringProp>
            <stringProp name="Argument.value">6162</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
```

```
    <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup"
testname="Users Post to Forum" enabled="true">
      <elementProp name="ThreadGroup.main_controller"
elementType="LoopController" guiclass="LoopControlPanel"
testclass="LoopController" testname="Loop Controller" enabled="true">
        <boolProp name="LoopController.continue_forever">false</boolProp>
        <stringProp name="LoopController.loops">1</stringProp>
      </elementProp>
      <stringProp name="ThreadGroup.num_threads">175</stringProp>
      <stringProp name="ThreadGroup.ramp_time">60</stringProp>
      <longProp name="ThreadGroup.start_time">1271129323000</longProp>
      <longProp name="ThreadGroup.end_time">1271129323000</longProp>
      <boolProp name="ThreadGroup.scheduler">false</boolProp>
      <stringProp
name="ThreadGroup.on_sample_error">stopthread</stringProp>
      <stringProp name="ThreadGroup.duration"></stringProp>
      <stringProp name="ThreadGroup.delay"></stringProp>
    </ThreadGroup>
    <hashTree>
      <ConfigTestElement guiclass="HttpDefaultsGui"
testclass="ConfigTestElement" testname="HTTP Request Defaults"
enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" testname="User Defined Variables" enabled="true">
          <collectionProp name="Arguments.arguments"/>
        </elementProp>
        <stringProp
name="HTTPSampler.domain">schoolserver.example.org</stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp name="HTTPSampler.path"></stringProp>
      </ConfigTestElement>
      <hashTree/>
      <CookieManager guiclass="CookiePanel" testclass="CookieManager"
testname="HTTP Cookie Manager" enabled="true">
        <collectionProp name="CookieManager.cookies"/>
        <stringProp name="TestPlan.comments">enable cookies</stringProp>
        <boolProp name="CookieManager.clearEachIteration">true</boolProp>
        <stringProp name="CookieManager.policy">default</stringProp>
      </CookieManager>
      <hashTree/>
```

```
      <CSVDataSet guiclass="TestBeanGUI" testclass="CSVDataSet"
testname="CSV Data Set Config" enabled="true">
        <stringProp name="delimiter">,</stringProp>
        <stringProp name="fileEncoding"></stringProp>
        <stringProp name="filename">jmeter_data_set.csv</stringProp>
        <boolProp name="quotedData">false</boolProp>
        <boolProp name="recycle">true</boolProp>
        <stringProp name="shareMode">All threads</stringProp>
        <boolProp name="stopThread">false</boolProp>
        <stringProp
name="variableNames">USER,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ADDR,
RANDOM_STRING</stringProp>
      </CSVDataSet>
      <hashTree/>
      <GaussianRandomTimer guiclass="GaussianRandomTimerGui"
testclass="GaussianRandomTimer" testname="Gaussian Random Timer"
enabled="true">
        <stringProp name="ConstantTimer.delay">300</stringProp>
        <stringProp name="RandomTimer.range">100.0</stringProp>
      </GaussianRandomTimer>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Log In" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="username" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">username</stringProp>
              <stringProp name="Argument.value">${USER}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="password" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">password</stringProp>
              <stringProp name="Argument.value">${PASSWORD}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="testcookies" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">testcookies</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
```

```
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
           </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/login/index.php</stringProp>
        <stringProp name="HTTPSampler.method">POST</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree>
        <RegexExtractor guiclass="RegexExtractorGui" testclass="RegexExtractor"
testname="Get Session Key" enabled="true">
          <stringProp name="RegexExtractor.useHeaders">true</stringProp>
          <stringProp
name="RegexExtractor.refname">SESSION_KEY</stringProp>
          <stringProp    name="RegexExtractor.regex">MoodleSessionTest(.*)=(.*);
path</stringProp>
          <stringProp name="RegexExtractor.template">$2$</stringProp>
          <stringProp
name="RegexExtractor.default">REGEX_FAILED</stringProp>
          <stringProp name="RegexExtractor.match_number">1</stringProp>
        </RegexExtractor>
        <hashTree/>
      </hashTree>
      <HTTPSampler    guiclass="HttpTestSampleGui"    testclass="HTTPSampler"
testname="Access Course" enabled="true">
        <elementProp                                  name="HTTPsampler.Arguments"
elementType="Arguments"                                guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
```

```
            <stringProp name="Argument.value">${COURSE_ID}</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
            <stringProp name="Argument.name">id</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="HTTPSampler.domain"></stringProp>
      <stringProp name="HTTPSampler.port"></stringProp>
      <stringProp name="HTTPSampler.connect_timeout"></stringProp>
      <stringProp name="HTTPSampler.response_timeout"></stringProp>
      <stringProp name="HTTPSampler.protocol"></stringProp>
      <stringProp name="HTTPSampler.contentEncoding"></stringProp>
      <stringProp
name="HTTPSampler.path">/moodle/course/view.php</stringProp>
      <stringProp name="HTTPSampler.method">GET</stringProp>
      <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
      <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
      <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
      <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
      <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
      <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
      <stringProp name="HTTPSampler.mimetype"></stringProp>
      <boolProp name="HTTPSampler.monitor">false</boolProp>
      <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
    </HTTPSampler>
    <hashTree/>
    <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Forum" enabled="true">
      <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
        <collectionProp name="Arguments.arguments">
          <elementProp name="id" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.value">${FORUM_ID}</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
            <stringProp name="Argument.name">id</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="HTTPSampler.domain"></stringProp>
      <stringProp name="HTTPSampler.port"></stringProp>
      <stringProp name="HTTPSampler.connect_timeout"></stringProp>
      <stringProp name="HTTPSampler.response_timeout"></stringProp>
```

```
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/forum/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="View Thread" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="d" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${DISCUSSION_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">d</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/forum/discuss.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
```

```
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Reply Thread" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="reply" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${REPLY_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">reply</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/forum/post.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Post Reply" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
```

```
<elementProp name="MAX_FILE_SIZE" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">MAX_FILE_SIZE</stringProp>
  <stringProp name="Argument.value">512000</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="subscribe" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">subscribe</stringProp>
  <stringProp name="Argument.value">1</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="timestart" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">timestart</stringProp>
  <stringProp name="Argument.value">0</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="timeend" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">timeend</stringProp>
  <stringProp name="Argument.value">0</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="course" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">course</stringProp>
  <stringProp name="Argument.value">${COURSE_ID}</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="forum" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">forum</stringProp>
  <stringProp name="Argument.value">0</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="discussion" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">discussion</stringProp>
  <stringProp name="Argument.value">${DISCUSSION_ID}</stringProp>
```

```
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="parent" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">parent</stringProp>
  <stringProp name="Argument.value">${REPLY_ID}</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="userid" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">userid</stringProp>
  <stringProp name="Argument.value">${USER_ID}</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="groupid" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">groupid</stringProp>
  <stringProp name="Argument.value">0</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="edit" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">edit</stringProp>
  <stringProp name="Argument.value">0</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="reply" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">reply</stringProp>
  <stringProp name="Argument.value">${REPLY_ID}</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="sesskey" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">sesskey</stringProp>
  <stringProp name="Argument.value">${SESSION_KEY}</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
```

```
          <elementProp                    name="_qf__mod_forum_post_form"
elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp
name="Argument.name">_qf__mod_forum_post_form</stringProp>
            <stringProp name="Argument.value">1</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
          </elementProp>
          <elementProp name="subject" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.name">subject</stringProp>
            <stringProp     name="Argument.value">Re:     This     is     a     test
thread</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
          </elementProp>
          <elementProp name="message" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.name">message</stringProp>
            <stringProp
name="Argument.value">${RANDOM_STRING}</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
          </elementProp>
          <elementProp name="format" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.name">format</stringProp>
            <stringProp name="Argument.value">0</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
          </elementProp>
          <elementProp name="submitbutton" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.name">submitbutton</stringProp>
            <stringProp name="Argument.value">Post to forum</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="HTTPSampler.domain"></stringProp>
      <stringProp name="HTTPSampler.port"></stringProp>
      <stringProp name="HTTPSampler.connect_timeout"></stringProp>
      <stringProp name="HTTPSampler.response_timeout"></stringProp>
      <stringProp name="HTTPSampler.protocol">http</stringProp>
```

```
        <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/forum/post.php</stringProp>
        <stringProp name="HTTPSampler.method">POST</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">true</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="View Thread Again" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="d" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">d</stringProp>
              <stringProp name="Argument.value">${DISCUSSION_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/forum/discuss.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
```

```
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <ResultCollector guiclass="ViewResultsFullVisualizer"
testclass="ResultCollector" testname="View Results Tree" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="SummaryReport" testclass="ResultCollector"
testname="Summary Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
```

```
            <time>true</time>
            <latency>true</latency>
            <timestamp>false</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="TableVisualizer" testclass="ResultCollector"
testname="View Results in Table" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
```

```
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="StatVisualizer" testclass="ResultCollector"
testname="Aggregate Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>
```

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessage>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <threadCounts>true</threadCounts>
          </value>
        </objProp>
        <stringProp name="filename">xo_post_forum_175.jtl</stringProp>
      </ResultCollector>
      <hashTree/>
    </hashTree>
  </hashTree>
</hashTree>
</jmeterTestPlan>

UsersUploadHomework.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="2.1">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="OLPC" enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables" elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments" testname="User Defined Variables" enabled="true">
        <collectionProp name="Arguments.arguments">
          <elementProp name="COURSE_ID" elementType="Argument">
            <stringProp name="Argument.name">COURSE_ID</stringProp>
            <stringProp name="Argument.value">7</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="ASSIGNMENT_ID" elementType="Argument">
            <stringProp name="Argument.name">ASSIGNMENT_ID</stringProp>
            <stringProp name="Argument.value">25</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
```

```
    <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup"
testname="Users Upload Homework" enabled="true">
      <elementProp name="ThreadGroup.main_controller"
elementType="LoopController" guiclass="LoopControlPanel"
testclass="LoopController" testname="Loop Controller" enabled="true">
        <boolProp name="LoopController.continue_forever">false</boolProp>
        <stringProp name="LoopController.loops">1</stringProp>
      </elementProp>
      <stringProp name="ThreadGroup.num_threads">75</stringProp>
      <stringProp name="ThreadGroup.ramp_time">60</stringProp>
      <longProp name="ThreadGroup.start_time">1271129323000</longProp>
      <longProp name="ThreadGroup.end_time">1271129323000</longProp>
      <boolProp name="ThreadGroup.scheduler">false</boolProp>
      <stringProp
name="ThreadGroup.on_sample_error">stopthread</stringProp>
      <stringProp name="ThreadGroup.duration"></stringProp>
      <stringProp name="ThreadGroup.delay"></stringProp>
    </ThreadGroup>
    <hashTree>
      <ConfigTestElement guiclass="HttpDefaultsGui"
testclass="ConfigTestElement" testname="HTTP Request Defaults"
enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" testname="User Defined Variables" enabled="true">
          <collectionProp name="Arguments.arguments"/>
        </elementProp>
        <stringProp
name="HTTPSampler.domain">schoolserver.example.org</stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp name="HTTPSampler.path"></stringProp>
      </ConfigTestElement>
      <hashTree/>
      <CookieManager guiclass="CookiePanel" testclass="CookieManager"
testname="HTTP Cookie Manager" enabled="true">
        <collectionProp name="CookieManager.cookies"/>
        <stringProp name="TestPlan.comments">enable cookies</stringProp>
        <boolProp name="CookieManager.clearEachIteration">false</boolProp>
        <stringProp name="CookieManager.policy">rfc2109</stringProp>
      </CookieManager>
      <hashTree/>
```

```
      <CSVDataSet guiclass="TestBeanGUI" testclass="CSVDataSet"
testname="CSV Data Set Config" enabled="true">
        <stringProp name="delimiter">,</stringProp>
        <stringProp name="fileEncoding"></stringProp>
        <stringProp name="filename">jmeter_data_set.csv</stringProp>
        <boolProp name="quotedData">false</boolProp>
        <boolProp name="recycle">true</boolProp>
        <stringProp name="shareMode">All threads</stringProp>
        <boolProp name="stopThread">false</boolProp>
        <stringProp
name="variableNames">USER,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ADDR,
RANDOM_STRING</stringProp>
      </CSVDataSet>
      <hashTree/>
      <GaussianRandomTimer guiclass="GaussianRandomTimerGui"
testclass="GaussianRandomTimer" testname="Gaussian Random Timer"
enabled="true">
        <stringProp name="ConstantTimer.delay">300</stringProp>
        <stringProp name="RandomTimer.range">100.0</stringProp>
      </GaussianRandomTimer>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Log In" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="username" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">username</stringProp>
              <stringProp name="Argument.value">${USER}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="password" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">password</stringProp>
              <stringProp name="Argument.value">${PASSWORD}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="testcookies" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">testcookies</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
```

```
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
           </elementProp>
          </collectionProp>
         </elementProp>
         <stringProp name="HTTPSampler.domain"></stringProp>
         <stringProp name="HTTPSampler.port"></stringProp>
         <stringProp name="HTTPSampler.connect_timeout"></stringProp>
         <stringProp name="HTTPSampler.response_timeout"></stringProp>
         <stringProp name="HTTPSampler.protocol">http</stringProp>
         <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
         <stringProp
name="HTTPSampler.path">/moodle/login/index.php</stringProp>
         <stringProp name="HTTPSampler.method">POST</stringProp>
         <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
         <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
         <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
         <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
         <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
         <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
         <stringProp name="HTTPSampler.mimetype"></stringProp>
         <boolProp name="HTTPSampler.monitor">false</boolProp>
         <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
       </HTTPSampler>
       <hashTree/>
       <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Course" enabled="true">
         <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
           <elementProp name="id" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.value">${COURSE_ID}</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
            <stringProp name="Argument.name">id</stringProp>
           </elementProp>
          </collectionProp>
         </elementProp>
         <stringProp name="HTTPSampler.domain"></stringProp>
         <stringProp name="HTTPSampler.port"></stringProp>
         <stringProp name="HTTPSampler.connect_timeout"></stringProp>
         <stringProp name="HTTPSampler.response_timeout"></stringProp>
         <stringProp name="HTTPSampler.protocol"></stringProp>
         <stringProp name="HTTPSampler.contentEncoding"></stringProp>
```

```
        <stringProp
name="HTTPSampler.path">/moodle/course/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Assignment" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${ASSIGNMENT_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">id</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">moodle/mod/assignment/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
```

```
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Upload Fle" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">id</stringProp>
              <stringProp name="Argument.value">${ASSIGNMENT_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="MAX_FILE_SIZE" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">MAX_FILE_SIZE</stringProp>
              <stringProp name="Argument.value">5242880</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="save" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">save</stringProp>
              <stringProp name="Argument.value">Upload this file</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/assignment/upload.php</stringProp
>
        <stringProp name="HTTPSampler.method">POST</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">true</boolProp>
```

```
        <stringProp name="HTTPSampler.FILE_NAME">C:\Documents and
Settings\btran\Desktop\Tests\homework-sample.jpg</stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD">newfile</stringProp>
        <stringProp name="HTTPSampler.mimetype">image/jpeg</stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <ResultCollector guiclass="ViewResultsFullVisualizer"
testclass="ResultCollector" testname="View Results Tree" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="SummaryReport" testclass="ResultCollector"
testname="Summary Report" enabled="true">
```

```
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>false</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="TableVisualizer" testclass="ResultCollector"
testname="View Results in Table" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
```

```
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector      guiclass="StatVisualizer"      testclass="ResultCollector"
testname="Aggregate Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
```

```
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <threadCounts>true</threadCounts>
          </value>
        </objProp>
        <stringProp name="filename">xo_upload_hw_75.jtl</stringProp>
      </ResultCollector>
      <hashTree/>
    </hashTree>
  </hashTree>
 </hashTree>
</jmeterTestPlan>
```

UsersTakeQuiz.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="2.1">
 <hashTree>
  <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="OLPC"
enabled="true">
    <stringProp name="TestPlan.comments"></stringProp>
    <boolProp name="TestPlan.functional_mode">false</boolProp>
    <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
    <elementProp name="TestPlan.user_defined_variables"
elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments"
testname="User Defined Variables" enabled="true">
      <collectionProp name="Arguments.arguments">
        <elementProp name="COURSE_ID" elementType="Argument">
          <stringProp name="Argument.name">COURSE_ID</stringProp>
          <stringProp name="Argument.value">5</stringProp>
          <stringProp name="Argument.metadata">=</stringProp>
        </elementProp>
        <elementProp name="QUIZ_ID" elementType="Argument">
          <stringProp name="Argument.name">QUIZ_ID</stringProp>
          <stringProp name="Argument.value">19</stringProp>
          <stringProp name="Argument.metadata">=</stringProp>
        </elementProp>
      </collectionProp>
    </elementProp>
    <stringProp name="TestPlan.user_define_classpath"></stringProp>
  </TestPlan>
```

```
   <hashTree>
     <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup"
testname="Users Take Quiz" enabled="true">
       <elementProp name="ThreadGroup.main_controller"
elementType="LoopController" guiclass="LoopControlPanel"
testclass="LoopController" testname="Loop Controller" enabled="true">
         <boolProp name="LoopController.continue_forever">false</boolProp>
         <stringProp name="LoopController.loops">1</stringProp>
       </elementProp>
       <stringProp name="ThreadGroup.num_threads">175</stringProp>
       <stringProp name="ThreadGroup.ramp_time">60</stringProp>
       <longProp name="ThreadGroup.start_time">1271129323000</longProp>
       <longProp name="ThreadGroup.end_time">1271129323000</longProp>
       <boolProp name="ThreadGroup.scheduler">false</boolProp>
       <stringProp
name="ThreadGroup.on_sample_error">stopthread</stringProp>
       <stringProp name="ThreadGroup.duration"></stringProp>
       <stringProp name="ThreadGroup.delay"></stringProp>
     </ThreadGroup>
     <hashTree>
       <ConfigTestElement guiclass="HttpDefaultsGui"
testclass="ConfigTestElement" testname="HTTP Request Defaults"
enabled="true">
         <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" testname="User Defined Variables" enabled="true">
           <collectionProp name="Arguments.arguments"/>
         </elementProp>
         <stringProp
name="HTTPSampler.domain">schoolserver.example.org</stringProp>
         <stringProp name="HTTPSampler.port"></stringProp>
         <stringProp name="HTTPSampler.connect_timeout"></stringProp>
         <stringProp name="HTTPSampler.response_timeout"></stringProp>
         <stringProp name="HTTPSampler.protocol"></stringProp>
         <stringProp name="HTTPSampler.contentEncoding"></stringProp>
         <stringProp name="HTTPSampler.path"></stringProp>
       </ConfigTestElement>
       <hashTree/>
       <CookieManager guiclass="CookiePanel" testclass="CookieManager"
testname="HTTP Cookie Manager" enabled="true">
         <collectionProp name="CookieManager.cookies"/>
         <stringProp name="TestPlan.comments">enable cookies</stringProp>
         <boolProp name="CookieManager.clearEachIteration">false</boolProp>
         <stringProp name="CookieManager.policy">rfc2109</stringProp>
       </CookieManager>
       <hashTree/>
```

```
      <CSVDataSet guiclass="TestBeanGUI" testclass="CSVDataSet"
testname="CSV Data Set Config" enabled="true">
        <stringProp name="delimiter">,</stringProp>
        <stringProp name="fileEncoding"></stringProp>
        <stringProp name="filename">jmeter_data_set.csv</stringProp>
        <boolProp name="quotedData">false</boolProp>
        <boolProp name="recycle">true</boolProp>
        <stringProp name="shareMode">All threads</stringProp>
        <boolProp name="stopThread">false</boolProp>
        <stringProp
name="variableNames">USER,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ADDR,
RANDOM_STRING</stringProp>
      </CSVDataSet>
      <hashTree/>
      <GaussianRandomTimer guiclass="GaussianRandomTimerGui"
testclass="GaussianRandomTimer" testname="Gaussian Random Timer"
enabled="true">
        <stringProp name="ConstantTimer.delay">300</stringProp>
        <stringProp name="RandomTimer.range">100.0</stringProp>
      </GaussianRandomTimer>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Log In" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="username" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">username</stringProp>
              <stringProp name="Argument.value">${USER}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="password" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">password</stringProp>
              <stringProp name="Argument.value">${PASSWORD}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="testcookies" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">testcookies</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
```

```
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
           </elementProp>
          </collectionProp>
         </elementProp>
         <stringProp name="HTTPSampler.domain"></stringProp>
         <stringProp name="HTTPSampler.port"></stringProp>
         <stringProp name="HTTPSampler.connect_timeout"></stringProp>
         <stringProp name="HTTPSampler.response_timeout"></stringProp>
         <stringProp name="HTTPSampler.protocol">http</stringProp>
         <stringProp name="HTTPSampler.contentEncoding">utf-8</stringProp>
         <stringProp
name="HTTPSampler.path">/moodle/login/index.php</stringProp>
         <stringProp name="HTTPSampler.method">POST</stringProp>
         <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
         <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
         <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
         <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
         <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
         <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
         <stringProp name="HTTPSampler.mimetype"></stringProp>
         <boolProp name="HTTPSampler.monitor">false</boolProp>
         <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
        </HTTPSampler>
        <hashTree/>
        <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Course" enabled="true">
         <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
           <elementProp name="id" elementType="HTTPArgument">
            <boolProp name="HTTPArgument.always_encode">false</boolProp>
            <stringProp name="Argument.value">${COURSE_ID}</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
            <boolProp name="HTTPArgument.use_equals">true</boolProp>
            <stringProp name="Argument.name">id</stringProp>
           </elementProp>
          </collectionProp>
         </elementProp>
         <stringProp name="HTTPSampler.domain"></stringProp>
         <stringProp name="HTTPSampler.port"></stringProp>
         <stringProp name="HTTPSampler.connect_timeout"></stringProp>
         <stringProp name="HTTPSampler.response_timeout"></stringProp>
         <stringProp name="HTTPSampler.protocol"></stringProp>
         <stringProp name="HTTPSampler.contentEncoding"></stringProp>
```

```
        <stringProp
name="HTTPSampler.path">/moodle/course/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Access Quiz" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.value">${QUIZ_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
              <stringProp name="Argument.name">id</stringProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol"></stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">moodle/mod/quiz/view.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">false</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">true</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
```

```
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Attempt Quiz" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="id" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">id</stringProp>
              <stringProp name="Argument.value">${QUIZ_ID}</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
        <stringProp
name="HTTPSampler.path">/moodle/mod/quiz/attempt.php</stringProp>
        <stringProp name="HTTPSampler.method">GET</stringProp>
        <boolProp name="HTTPSampler.follow_redirects">true</boolProp>
        <boolProp name="HTTPSampler.auto_redirects">false</boolProp>
        <boolProp name="HTTPSampler.use_keepalive">true</boolProp>
        <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
        <stringProp name="HTTPSampler.FILE_NAME"></stringProp>
        <stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
        <stringProp name="HTTPSampler.mimetype"></stringProp>
        <boolProp name="HTTPSampler.monitor">false</boolProp>
        <stringProp name="HTTPSampler.embedded_url_re"></stringProp>
      </HTTPSampler>
      <hashTree/>
      <HTTPSampler guiclass="HttpTestSampleGui" testclass="HTTPSampler"
testname="Submit Answers" enabled="true">
        <elementProp name="HTTPsampler.Arguments"
elementType="Arguments" guiclass="HTTPArgumentsPanel"
testclass="Arguments" enabled="true">
          <collectionProp name="Arguments.arguments">
            <elementProp name="resp10_" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
```

```
  <stringProp name="Argument.name">resp10_</stringProp>
  <stringProp name="Argument.value">43</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="resp11_" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">resp11_</stringProp>
  <stringProp name="Argument.value">49</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="resp12_" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">resp12_</stringProp>
  <stringProp name="Argument.value">53</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="resp13_" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">resp13_</stringProp>
  <stringProp name="Argument.value">55</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="resp14_" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">resp14_</stringProp>
  <stringProp name="Argument.value">62</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="resp15_" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">resp15_</stringProp>
  <stringProp name="Argument.value">65</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="resp16_" elementType="HTTPArgument">
  <boolProp name="HTTPArgument.always_encode">false</boolProp>
  <stringProp name="Argument.name">resp16_</stringProp>
  <stringProp name="Argument.value">51</stringProp>
  <stringProp name="Argument.metadata">=</stringProp>
  <boolProp name="HTTPArgument.use_equals">true</boolProp>
```

```
            </elementProp>
            <elementProp name="resp15_" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">resp15_</stringProp>
              <stringProp name="Argument.value">56</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="resp18_" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">resp18_</stringProp>
              <stringProp name="Argument.value">80</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="finishattempt" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">finishattempt</stringProp>
              <stringProp name="Argument.value">Submit all and
finish</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="timeup" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">timeup</stringProp>
              <stringProp name="Argument.value">0</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
            <elementProp name="questionids" elementType="HTTPArgument">
              <boolProp name="HTTPArgument.always_encode">false</boolProp>
              <stringProp name="Argument.name">questionids</stringProp>
              <stringProp
name="Argument.value">10,11,12,13,14,15,16,15,18</stringProp>
              <stringProp name="Argument.metadata">=</stringProp>
              <boolProp name="HTTPArgument.use_equals">true</boolProp>
            </elementProp>
          </collectionProp>
        </elementProp>
        <stringProp name="HTTPSampler.domain"></stringProp>
        <stringProp name="HTTPSampler.port"></stringProp>
        <stringProp name="HTTPSampler.connect_timeout"></stringProp>
        <stringProp name="HTTPSampler.response_timeout"></stringProp>
        <stringProp name="HTTPSampler.protocol">http</stringProp>
        <stringProp name="HTTPSampler.contentEncoding"></stringProp>
```

<stringProp
name="HTTPSampler.path">/moodle/mod/quiz/attempt.php?q=2&page=0
</stringProp>
<stringProp name="HTTPSampler.method">POST</stringProp>
<boolProp name="HTTPSampler.follow_redirects">true</boolProp>
<boolProp name="HTTPSampler.auto_redirects">false</boolProp>
<boolProp name="HTTPSampler.use_keepalive">true</boolProp>
<boolProp name="HTTPSampler.DO_MULTIPART_POST">true</boolProp>
<stringProp name="HTTPSampler.FILE_NAME"></stringProp>
<stringProp name="HTTPSampler.FILE_FIELD"></stringProp>
<stringProp name="HTTPSampler.mimetype"></stringProp>
<boolProp name="HTTPSampler.monitor">false</boolProp>
<stringProp name="HTTPSampler.embedded_url_re"></stringProp>
</HTTPSampler>
<hashTree/>
<ResultCollector guiclass="ViewResultsFullVisualizer"
testclass="ResultCollector" testname="View Results Tree" enabled="true">
<boolProp name="ResultCollector.error_logging">false</boolProp>
<objProp>
<name>saveConfig</name>
<value class="SampleSaveConfiguration">
<time>true</time>
<latency>true</latency>
<timestamp>true</timestamp>
<success>true</success>
<label>true</label>
<code>true</code>
<message>true</message>
<threadName>true</threadName>
<dataType>false</dataType>
<encoding>false</encoding>
<assertions>false</assertions>
<subresults>false</subresults>
<responseData>false</responseData>
<samplerData>false</samplerData>
<xml>false</xml>
<fieldNames>true</fieldNames>
<responseHeaders>false</responseHeaders>
<requestHeaders>false</requestHeaders>
<responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
<assertionsResultsToSave>0</assertionsResultsToSave>
<bytes>true</bytes>
<sampleCount>true</sampleCount>

```
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="SummaryReport" testclass="ResultCollector"
testname="Summary Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>false</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>false</dataType>
            <encoding>false</encoding>
            <assertions>false</assertions>
            <subresults>false</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>false</xml>
            <fieldNames>true</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <sampleCount>true</sampleCount>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="TableVisualizer" testclass="ResultCollector"
testname="View Results in Table" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
```

```
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
          </value>
        </objProp>
        <stringProp name="filename"></stringProp>
      </ResultCollector>
      <hashTree/>
      <ResultCollector guiclass="StatVisualizer" testclass="ResultCollector"
testname="Aggregate Report" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
            <message>true</message>
            <threadName>true</threadName>
            <dataType>true</dataType>
            <encoding>false</encoding>
```

```
            <assertions>true</assertions>
            <subresults>true</subresults>
            <responseData>false</responseData>
            <samplerData>false</samplerData>
            <xml>true</xml>
            <fieldNames>false</fieldNames>
            <responseHeaders>false</responseHeaders>
            <requestHeaders>false</requestHeaders>
            <responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>false</saveAssertionResultsFailureMessag
e>
            <assertionsResultsToSave>0</assertionsResultsToSave>
            <bytes>true</bytes>
            <threadCounts>true</threadCounts>
          </value>
        </objProp>
        <stringProp name="filename">fitpc_take_quiz_175.jtl</stringProp>
      </ResultCollector>
      <hashTree/>
    </hashTree>
  </hashTree>
</hashTree>
</jmeterTestPlan>
```