**CSC 440 -001 Software Engineering**

**Software Requirements Specification**

**Team 1 Monday, 19th October 2015**

Contents

[**1.** Document information 3](#_Toc432703264)

[**1.1.** Prepared by 3](#_Toc432703265)

[**1.2.** Document responsibilities 3](#_Toc432703266)

[**1.3.** Revision history 3](#_Toc432703267)

[**2.** Introduction 4](#_Toc432703268)

[**2.1.** Purpose 4](#_Toc432703269)

[**2.2.** Scope 4](#_Toc432703270)

[**2.3.** Definitions, acronyms, and abbreviations 4](#_Toc432703271)

[**2.4.** References 4](#_Toc432703272)

[**2.5.** Overview 4](#_Toc432703273)

[**3.** Overall description 5](#_Toc432703274)

[**3.1.** Product perspective 5](#_Toc432703275)

[**3.2.** Product functions 5](#_Toc432703276)

[**3.3.** User characteristics 5](#_Toc432703277)

[**3.4.** General constraints 5](#_Toc432703278)

[**3.5.** Assumptions and dependencies 5](#_Toc432703279)

[**4.** Specific requirements 6](#_Toc432703280)

[**4.1.** External interface requirements 6](#_Toc432703281)

[**4.2.** Functional requirements 6](#_Toc432703282)

[**4.3.** Performance requirements 6](#_Toc432703283)

[**4.4.** Design constraints 7](#_Toc432703284)

[**4.5.** Software system attributes 7](#_Toc432703285)

[**4.6.** Other requirements 7](#_Toc432703286)

[**5.** Project management 7](#_Toc432703287)

[**5.1.** Project estimation 7](#_Toc432703288)

[**5.2.** Risk assessment 7](#_Toc432703289)

[**6.** Future system improvements 7](#_Toc432703290)

[**7.** Appendix: 8](#_Toc432703291)

## Document information

### Prepared by

Erik Pratt

Mahamadou TRAORE

kyle marcum

Sidiya Sidiya

### Document responsibilities

**Use case diagram**

Mahamadou Traore

Kyle Marcum

Erik Pratt

Sidiya Sidiya

**Activity diagram**

Erik Pratt

**Slides preparation**

Sidiya Sidiya

### Revision history

09/29/2015 Initial version (V1.0)

10/6/2015 second version

10/15/2015 final version

|  |  |  |
| --- | --- | --- |
| Date | Version | Students |
| 09/29/2015 | Initial version (V1.0) | Erik Pratt  Mahamadou Traore |
| 10/6/2015 | second version | Sidiya Sidiya  Kyle Marcum |
| 10/15/2015 | final version | Sidiya Sidiya  Kyle Marcum  Erik Pratt  Mahamadou Traore |

## Introduction

### Purpose

The Purpose of our project is to make a connection between a server and a client using the HTTP 1.1 protocol .We will also design and implement a web file generator in Java Programming Language.

### Scope

Java Language Programming will be used to implement a server which will be able to handle HTTP requests from a client and generate HTTP responses. All HTTP requests will be over TCP connection.

### Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| Term/Abbreviation | Explanation/Definitions |
| Server | a software program, or the **computer** on which that program runs, that provides a specific kind of service to client software running on the same **computer** or other computers on a network |
| Client | A **client** is a piece of **computer** hardware or software that accesses a service made available by a server. In our project the Basic Web Browser is the client |
| HTTP: Hyper Text Transfer Protocol | is an application **protocol** for distributed, collaborative, hypermedia information systems |
| SRS: Software Requirements Specifications | is a document that captures complete description about how the system is expected to perform |
| UDP: User Datagram Protocol | is a transport layer **protocol** defined for use with the IP network layer **protocol.** |
| TCP: Transmission Control Protocol | is a core **protocol** of the Internet **protocol** suite. It originated in the initial network implementation in which it complemented the Internet **Protocol** (IP). |
| WASCE | WebSphere Application Server Community Edition |

### References

* Foundation of Security, *Neil Daswani, Christoph Kern* - for a sample HTTP java implementation
* cacoo.com - for the UML collaboration
* google search - for the IEEE code for Ethernet
* Wikipedia

### Overview

The HTTP server will be programmed in java, the files will be randomly generated, and TCP will be used for the network communications.

## Overall description

### Product perspective (more details can be added)

This System will consists of two parts: one is a client that will request a specific file from the server via HTTP and a server that will send back the requested file to the client or “404 NOT FOUND “ if the requested file doesn’t exist.

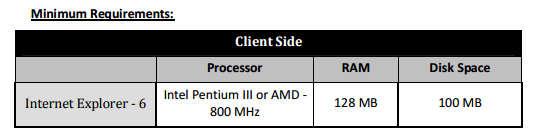
**Software Interface**

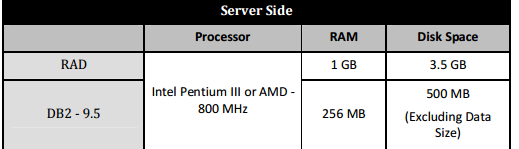
Client on Internet Web Browser, Operating System (any)

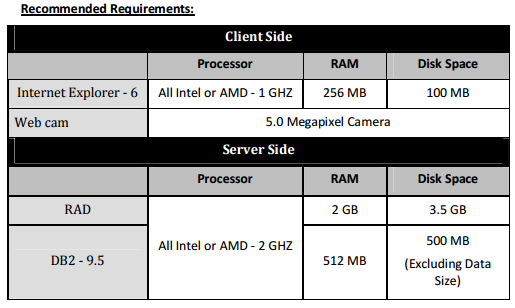
Web Server WASCE, Operating System (any)

Development End: OS (Windows), WebSphere (Web Server).

**Hardware Interface**

****

****

****

**Communication Interface**

Client (user) on Internet will be using HTTP protocol.

Client (system) on Internet will be using HTTP/HTTPS protocol.

### Product functions

With this product, the user will be able to request a specific file from server. The response will be vary depends on if the file is found or not.

The result will be 404 NOT FOUND if the specified file not found.

The result will include the requested file from the user when the file is found.

Concurrent HTTP request can be handle.

### User characteristics

The user will be able to request a file from the server by specify the name of that file and wait for the response.

The server (System) will be able to response with the requested file or with 404 Error.

### General constraints

* + Limited to HTTP Protocol.
  + This system is working for single server
  + File name is used to request the file
  + The size of the file is limited to 2MB at most.

### Assumptions and dependencies

We assume that the user knows the name of the requested file.

User will type the correct name of the requested file.

## Specific requirements

### External interface requirements

#### **User interfaces**

A client being a bare bones web browser

#### **Hardware interfaces**

Two X86-64 computers with a Ethernet or 802.11 NIC

#### **Software interfaces**

java ver 8 update 60

Java virtual Machine with standard libraries

the socket class will be used

#### **Communication interfaces**

TCP over IEEE 802.11 or 802.3

logging (basic I/O)

### Functional requirements

* + 1. **Overall activity diagram**

See Cacoo Link below

* + 1. **Use case diagram**

<https://cacoo.com/diagrams/N3kouMFEisLa65Ej/edit>

* + 1. **Detailed activity diagram**

### Performance requirements

• Response Time: 100ms

• Scalability: Ability to deal with growing work

• Platform: Light enough to be used with different platforms

• Proper Caching

### Design constraints

The server and client do not have proxy functions built in, also they do not have robust network error handling code built in. Connections are limited to TCP.

### Software system attributes

Multi-threaded and written in java with the sockets API.

### Other requirements

RAM usage needs to be light/efficient in order to handle multiple simultaneous connections. Proper multi-threading is needed as well.

Caching needs to be carefully done as well. Due to the cheapness of DDR3 it is tempting to cache everything but then there is little left for the system as a whole to use.

## Project management

### Project estimation

The current plan and estimation is to move the project into a beta state in about 5.5 weeks, from then there is about 1 week. to work through the Beta phase. Seeing as how testing is going to be done constantly throughout the alpha phase, this should be readily achievable.

### Risk assessment

The main risk is lack of proper communication and time management resulting in a rushed or poorly implemented product.

A less likely risk is a bug being found at the last minute, since testing is being done throughout the development process, this is a sufficiently remote possibility.

## Future system improvements

UDP support

SSL support

On the Fly Compression

Secured transfer (File encryption) - Using AES-NI when available

## Appendix:

1. Task partition
2. Work schedule (MS Project)
3. Weekly meeting agenda and minutes
4. Other communication records (such as emails, Wiki, Google Wave, Instant messaging, and so on)
5. Software configuration management

Git

Trello