Project Report
Group Badgers

Java and C# in depth, Spring 2013

Thomas Frick (03-150-927)
Matthias Ganz (04-862-850)
Philipp Rohr (04-397-030)

April 4, 2013

1 The Full Model

After we did one more experiment that is explained in the appendix I could could examine the measured values and made some observations that are described in this section. Based on those observations I built the full model that will be the basis of a mean value analysis shown further in the document.

1.1 Definitions

blablab

1.2 Observations

This is a table (table 1) measured throughout the and a footnote [1]

N	R_{puts}	$R_{retrieves}$	R_{meas}	X_{meas}	X_{calc}	R_{calc}	Z_{calc}
32	69	112	181	176.5	176.8	181	0
64	72	113	185	344.7	345.9	186	1
96	81	115	196	486.4	489.8	197	1
128	98	119	217	586.8	589.9	218	1
160	124	130	254	628.3	629.9	255	1
192	161	156	317	603.9	605.7	318	1
224	213	204	417	536.4	537.2	418	1
256	251	230	481	531.2	532.2	482	1

Table 1: Measured (on client side) and calculated data of the whole system.

1.3 The Model

and a figure 1. fancy foobar

[1]Described in the appendix **??**

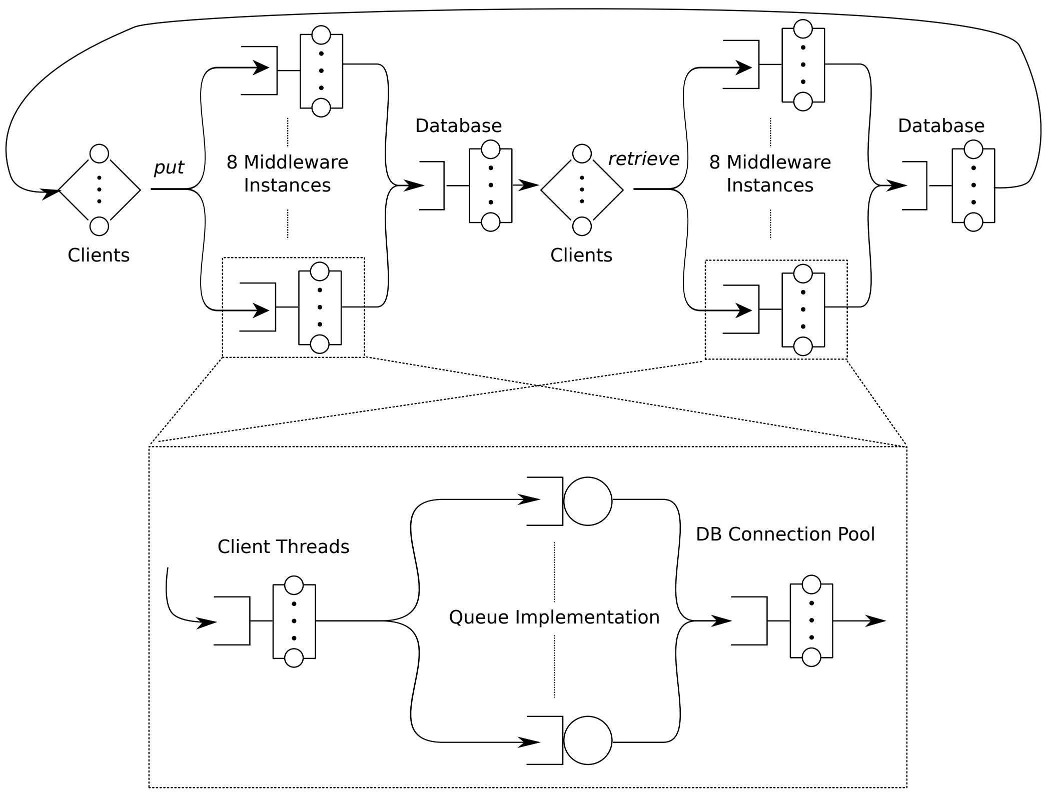

Figure 1: Full model of the system.

citation[1].
some math

$$\mu(n) = \begin{cases} n/S & \text{if } n = 1, 2, ..., m-1 \\ m/S & \text{if } n = m, m+1, ..., \infty \end{cases}$$

2 Introduction

This document describes the design and implementation of the *Personal Virtual File System* of group *Badgers*. The project is part of the course *Java and C# in depth* at ETH Zurich. The following sections describe each project phase, listing the requirements that were implemented and the design decisions taken. The last section describes a use case of using the *Personal Virtual File System*.

3 VFS Core

Give a short description (1-2 paragraphs) of what VFS Core is.

3.1 Requirements

Describe which requirements (and possibly bonus requirements) you have implemented in this part. Give a quick description (1-2 sentences) of each requirement. List the software elements (classes and or functions) that are mainly involved in implementing each requirement.

3.2 Design

Give an overview of the design of this part and describe in general terms how the implementation works. You can mention design patterns used, class diagrams, definition of custom file formats, network protocols, or anything else that helps understand the implementation.

4 Quick Start Guide

4.1 the eclipse project

The project requires to be compiled with JAVA 7. It also depends on the maven plugin which pulls in all the required libraries.

4.2 Command line client

The command line client allows the usage of the VFS core and is mainly intended to test the basic functionalities. The console runs either in management mode or in filesystem mode. The management mode is entered automatically when starting the command line client. It allows creating and disposing virtual disks. The filesystem mode is entered as soon as a virtual disk is opened.

4.2.1 startup

The command line client can be started as follows:

```
java -jar VFSCore.jar ch.eth.jcd.badgers.vfs.ui.VFSConsole
```

or by starting `ch.eth.jcd.badgers.vfs.ui.VFSConsole` in eclipse.

This gives a console prompt where the following commands can be used in.

4.2.2 commands

Following commands can be used with the command line client in management mode:

- **create c:\path\to\disk.bfs 1024** creates virtual disk with a maximum quota of 1024 megabytes on the host system. The file may grow up to 1024 megabytes. **TODO**: more parameters are needed (encryption, compression, password if there is encryption)
- **dispose c:\path\to\disk.bfs** deletes the given virtual disk
- **open c:\path\to\disk.bfs** opens filesystem mode for the given virtual disk
- **exit** exits the console program

follwing commands can be used in filesystem mode:

- **ls** lists the contents of the current directory
- **pwd** shows the path to the current directory
- **cd dst** changes current directory to *dst* which must be either a child directory of the current path or ".."
- **mkdir dirName** creates a new directory *dirName* in the current path
- **mkfile fileName** creates a new empty file *fileName* in the current path - this is rather not usefull, as the "import" creates a file with content
- **rm file** deletes the entry denoted as *file*, it must be a child of the current path
- **cp src dst** copies the *src* file to *dst* as a child of the current path
- **mv src dst** moves the *src* file to *dst*
- **import ext_src dst** imports a *ext_src* from the host system to *dst*
- **export src ext_src** exports a *src* file to the host system *ext_dst*
- **find searchString** lists all filesystem entries below the current entry containing *searchString*
- **close** closes the filesystem mode, from now on management mode commands can be executed

5 Glossary

VFS core The main Java library, that handles all the interaction with virtual disks and importing/exporting/storing files. It is used by the command line client and the gui.

Virtual Disk A virtual disk denotes a container file that is stored on the host file system. A virtual disk can be opened with the software that is developed during this project and stores the actual files. The file extension of the virtual disk is "*.bfs".

References

[1] A. Lempel and J. Ziv. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory 23, 337-343*, 1997.