GROUP C

UNIVERSITY OF TEXAS AT DALLAS | 800 W CaMPBELL RD RICHARDSON TX 75080

MIDTERM REPORT

CLOUD COMPUTING PROJECT



Table of Contents

[Group Information 2](#_Toc433206534)

[Details of team members 2](#_Toc433206535)

[Problem Statement 3](#_Toc433206536)

[Design 4](#_Toc433206537)

[Ceph 4](#_Toc433206538)

[GFS 4](#_Toc433206539)

[Implementation 5](#_Toc433206540)

[Ceph Directory Operations 5](#_Toc433206541)

[Installation Guide 6](#_Toc433206542)

[Master 6](#_Toc433206543)

[Individual contributions 7](#_Toc433206544)

[Ekal Golas 7](#_Toc433206545)

[References 8](#_Toc433206546)

# Group Information

Group: C

Project Name: Directory Structure Maintenance

Team Members: 4

TA: Yongtao Huang

## Details of team members

1. Ekal Golas – exg140230
2. Jayakarthigayan Sridharan – jxs143730
3. Ketan Joshi – kkj140030
4. Sahith Katukuri – sxk145130

# Problem Statement

The project problem statement (Yen, Cloud Computing, 2015) is stated as:-

Compare different methods in implementing directory files, including three solutions

* Solution 1: Use a centralized server to store the entire directory
* Solution 2: Treat directory files as regular files, but may merge a subtree of directories into one file, with a fixed number of levels (the fixed number of levels is configurable)
* Solution 3: Ceph solution

For the midterm report, the progress covers the following:-

* Complete the basic directory maintenance systems
  + Implement all three systems in memory without replication and accept a single request at a time
    - For Ceph, do not consider dynamic load partitioning, but develop the mechanism to decide which partitioning is the best for the system
    - For HDFS, same as Ceph, except that there is no partitioning
    - For Solution 2, Yongtao provides the file system to host the directory files
  + Support create, delete, ls commands
* Implement the basic client
  + Generate the basic directory system on three maintenance systems
  + Generate a mix of client requests for accessing the directories
  + Submit the commands to the three directory management systems
* Support replication
  + Provide replication and master/slave update for HDFS
  + Ceph is the same, except that there are multiple partitions
  + For Solution 2, the system already supports replication

# Design

As discussed in the problem statement (Yen, Cloud Computing, 2015), the design for the different solutions and the overall implementation is stated as follows:-

## Overall Design

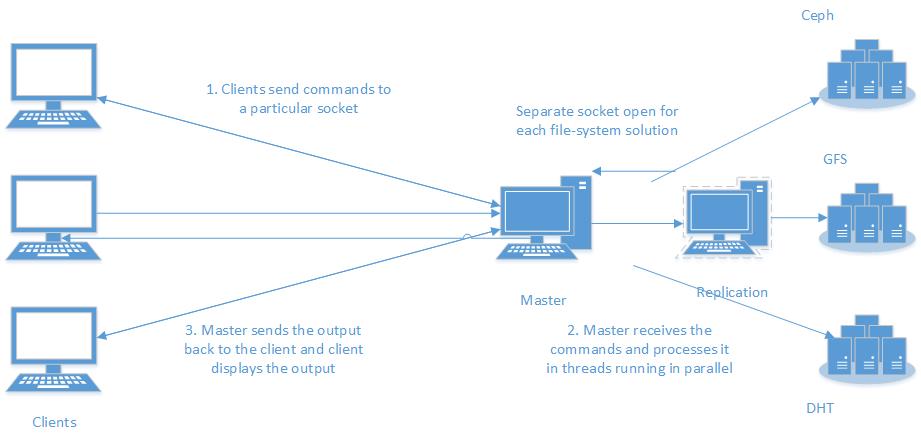
There are three components to the system, each represented as packages which represent the following:-

1. Client – All classes and packages inside it represents the client functionality and implementation
   1. Command generation
   2. Cache
   3. Communication with the master by socket programming
2. Master – All classes and packages inside it represents the master`s functionality, dependencies and implementation
   1. Ceph solution
   2. GFS solution
   3. DHT solution
   4. Communication with multiple clients with socket programming and multithreading
   5. Replication
3. Commons – A package required by both client and master to implemented common utility methods and class representations
   1. Directory representation
   2. Code provided by the TA for communication (danei, 2015)
   3. Configuration and other utilities

As per the above overall design, the client-server architecture is stated as follows:-

## Client-Server architecture

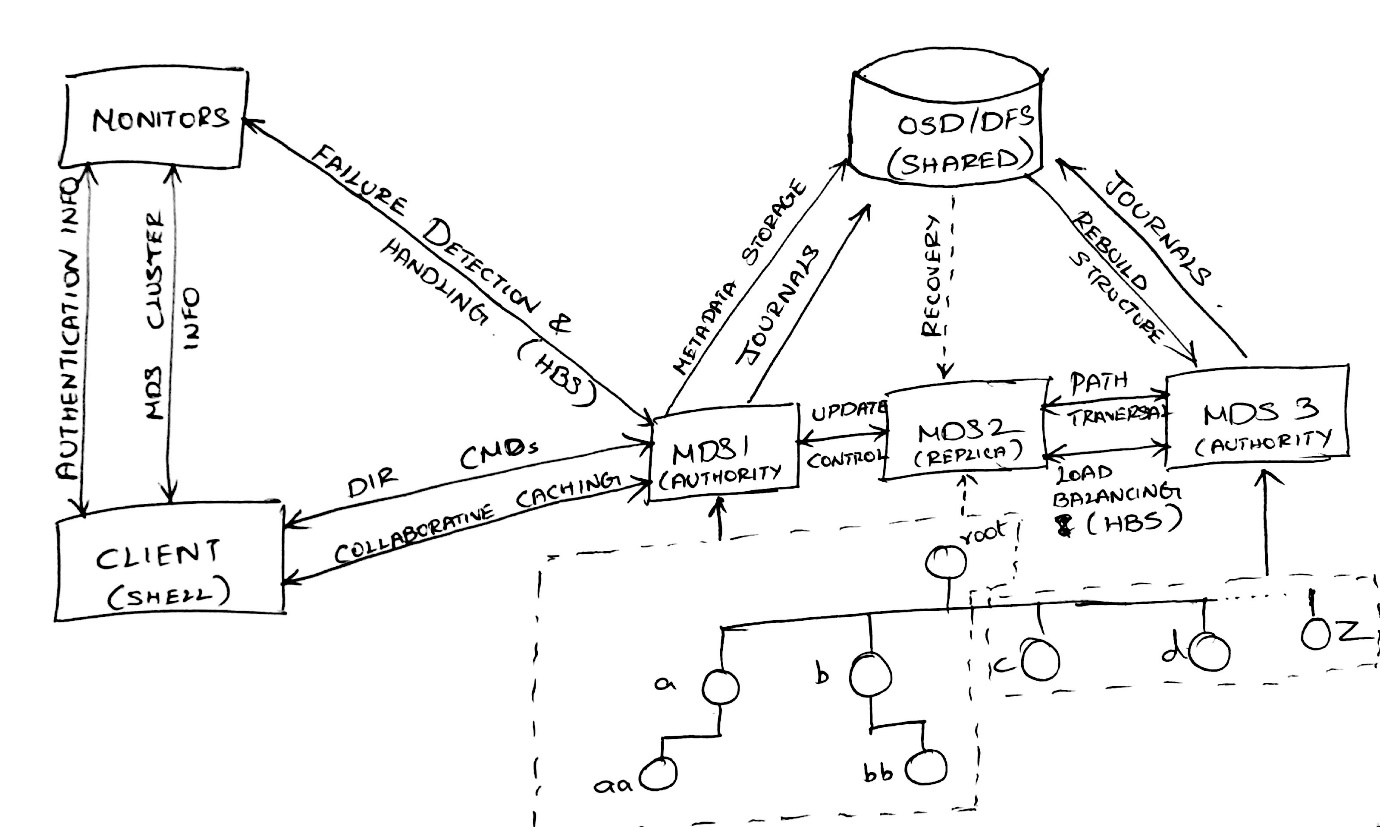
The following diagram depicts the design of client and the master components and the communication between them:-



As stated in the diagram, the different solutions are designed as follows:-

## Ceph

A design was developed using the Ceph paper (University of California, 2006). The design diagram is stated as follows:-

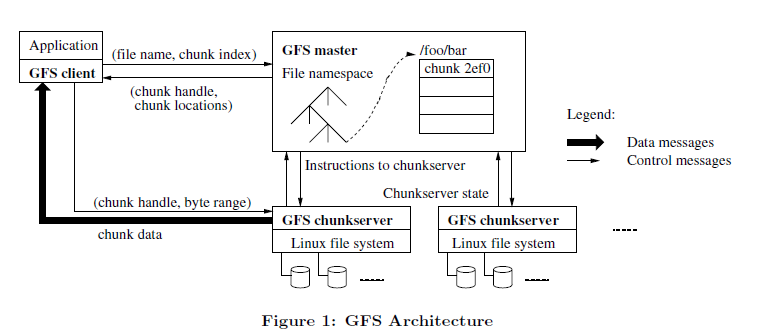


This design was implemented in the following manner:-

TODO

## GFS

A design was developed using the GFS paper (Google, 2003) and the class notes (Yen, Bigdata, 2015). The design diagram is stated as follows:-



The design was implemented in the following manner:-

TODO

## DHT

A design was developed using the DHT filesystem code (Yinzi, 2015) provided by the TA and class notes (Yen, Bigdata, 2015). The design is stated as follows:-

TODO

# Implementation

## Client

The client is launched by calling the Client class. It defines a main method which carries out the client operations. After the client class is called and the client is started, the process flow is implemented as:-

1. As the client starts, it calls the AppConfig class in [commons](#_Commons) to load configuration
2. Gets the master`s IP address and port to communicate, and creates a socket on that
3. Uses CommandGenerator class to generate commands to the server.
   1. CommandGenerator class gets the directory structure from the file in the configuration
   2. Gets all possible paths in the directory structure as an array
   3. Randomly shuffles the list of command operations defined in CommandsSupported enum in [commons](#_Commons) and the array of all possible paths
   4. Uses Zipf distribution provided by Apache`s math library to get a weighted distribution of a preconfigured number of commands that will be sent to the server
4. Client obtains the commands and serializes the data using the Message class in [commons](#_Commons)
5. executes them sequentially and displays the output to the console using the OutputFormatter class in [commons](#_Commons)

## Commons

This component is defines utilities and classes used by both client and master. It contains the following:-

1. The code provided by the TA to implement communication between client and server using wrapper over TCP
2. The Directory class

## Ceph Directory Operations

# Installation Guide

## Master

# Individual contributions

## Ekal Golas

* Week 1 Tasks: Sep 13, 2015 to Sep 19, 2015
  + Read papers and Worked on Design
  + Downloading and making bootable pen drives for installation of OS and VMs
  + Created GitHub repository, tracked machine details and documentation
* Week 2 Tasks: Sep 20 to Sep 26
  + Helped in discussions for Ceph design
  + Created basic class and packages for initial implementation
  + Added DHT-fs to the project
  + Implemented commands: ls, mkdir, touch
* Week 3 Tasks: Sep 27 to Oct 3
  + Helped in discussions for Ceph design
  + Testing client and master communication
  + Debugging issues
  + Modified client and server to read configuration from a file
  + Added unit tests
* Week 4 Tasks: Oct 4 to Oct 10
  + Implemented basic structure to provide all the command operations
  + Read about Zipf
  + Added implementation to print command output in table format
  + Integrated the socket programming utilities provided by TA to the project code
* Week 5 Tasks: Oct 11 to Oct 17
  + Read about Zipf
  + Implemented Zipf distribution for the clients
  + Implemented command generation for clients in a configurable way
  + Tested and added unit tests for the same
* Week 6 Tasks: Oct 18 to Oct 24
  + Updated code to use correct version of dht-fs
  + Worked on project report

# References

Apache. (2012, May 13). *Log4j*. Retrieved from Logging: https://logging.apache.org/log4j/1.2/download.html

Apache. (2014, Feb 17). *IO*. Retrieved from Commons: https://commons.apache.org/proper/commons-io/download\_io.cgi

Apache. (2014, July 18). *Logging*. Retrieved from Commons: http://commons.apache.org/proper/commons-lang/download\_lang.cgi

Apache. (2015, May 18). *Lang*. Retrieved from Commons: http://commons.apache.org/proper/commons-lang/download\_lang.cgi

Apache. (2015, Sep 11). *Math*. Retrieved from Commons: http://commons.apache.org/proper/commons-math/download\_math.cgi

Apache. (2015, Aug 2). *Pool*. Retrieved from Commons: https://commons.apache.org/proper/commons-pool/download\_pool.cgi

danei. (2015). *Request-response interface*. Retrieved from Bitbucket: https://bitbucket.org/danei/iocontrol/

Google. (2003). *Google File System.* Retrieved from http://static.googleusercontent.com/media/research.google.com/en//archive/gfs-sosp2003.pdf

Google. (2009, June 3). *Collections*. Retrieved from Maven: http://central.maven.org/maven2/com/google/collections/google-collections/1.0-rc2/google-collections-1.0-rc2.jar

Google. (2010, June 1). *Guava*. Retrieved from Maven: http://central.maven.org/maven2/com/google/guava/guava/r05/guava-r05.jar

*Ini4j*. (n.d.). Retrieved from Sourceforge: http://downloads.sourceforge.net/project/ini4j/ini4j-bin/0.5.5-SNAPSHOT/ini4j-0.5.5-SNAPSHOT-bin.zip

University of California. (2006). *Ceph: A Scalable, High-Performance Distributed File System.* Retrieved from http://www.ssrc.ucsc.edu/Papers/weil-osdi06.pdf

Yen, I. L. (2015). *Bigdata.* Retrieved from Class notes: E-learning

Yen, I. L. (2015, 10 12). *Cloud Computing.* Retrieved from I-Ling`s home page: http://www.utdallas.edu/~ilyen/course/cloud/for15f/term-proj-midrep.pdf

Yinzi. (2015). *DHT file system*. Retrieved from Bitbucket: https://bitbucket.org/utdcloudlab/dht-fs