Deign Document: Basic Layer – File I/O

Hang Yuan (hyuan211@gmail.com)

Version: 0.2 (09/02/19)

0 Version Changes

This version of design removed auto-checker functionality because the checking function should be in Manager Layer rather than the Basic Layer based on its property.

Several directory manipulations were added to satisfy the need of user info separation.

New file types (*.df* and *.log*) were added.

1 Goals

The project will be based on our own designed database to store the email message instead of other existing standard libraries. Just like the other modules in each layer of this project, the "File I/O" will work as a functional module or library to the down layer (Manager Layer).

The File I/O module will provide other modules accesses to the file contained locally where store all the information of the system.

2 Design

The design for File I/O module will be included in three parts: (1) system design; (2) file type and format; (3) unit tests plan.

2.1 System Design

The module relies on `<cstdio>` to open, create or delete file and `<unistd.h>` and `<sys/*.h>` to manipulate directory. This module will take full responsibilities of creating, removing or accessing interfaces to the base file or directory.

Basic functions will be:

```
CreateFile   (const string &filename)
CreateDir    (const string &dirName)
DestoryFile (const string &fileName)
DestoryDir   (const string &dirName)
OpenFile     (const string &fileName)
CloseFile    ()
ResetFile    (const string &fileName)
ResetDir     (const string &dirName)
WriteFile    (size_t offset, size_t length, const void *data)
ReadFile     (size_t offset, size_t length, void *data)
AppendFile   (size_t length, const void *data)
```


More information about the type of file will be covered in next section.

Based on the ongoing development and project functionality requirements, more files might be added.

2.2 File Type and Format

This project uses the self-designed database to control the data collection, modification, and reading access. Thus, we design the specific type of data and corresponding format to support the data accesses.

2.2.1 File types

There are only two types maintained in the data directory: *.data*, *.sys*, *.df*, and *.log*.

.data: the file contains the project data, including the users' information and the email content (with email header) for each account. All data will be stored in plain text (we will consider adopt cipher text to improve the security in the future). This type of file should have many and frequent modifications and RD/WR manipulations due to data changed

.sys: will contains the system file, which cannot be modified at most time. This type of file only records the system configuration and actions for other modules to use. This type of file may not have many and frequent modifications or RD/WR manipulations because of its importance as a system file.

.df: the exact location of the email will be organized and stored in this kind of file.

.log: will contain all the information of system actions. Each module of layer will have its own log for better system maintenance.

2.2.2 File format

All files, no matter what the type, will contain a file header to store the basic file information. Different file with different purposes will have different usage which will be covered in other design documents.

2.3 Unit Test Plan

Unit tests will be run on the *UNIX* environment.

Unit tests for File I/O will cover nearly all possible cases that the module will meet. The unit test plan will include two parts:

(1) Valid and invalid input

(2) Edge cases:

a) Access issue

b) Lacking necessary file(s)

c) Creating file with existing filename

d) Opening nonexistent file

e) Reading nonexistent file

f) Closing nonexistent file

g) Deleting nonexistent file

All unit tests have been included in bl/basicIO.cpp.

A *Makefile* is provided to generate test result log.

