Faculty of Natural & Mathematical Sciences

Group Project - 7CCSMGPR
Deadline Fighters

Preliminary report - February 8, 2019

Sivaranjani Subramanian
Ka Hang Jacky Au Yeung
Lili Chen
Letao Wang
Qilin Zhou
Georgii Fidarov

1 Aim

Our team, *Deadline Fighters*, aim to develop a multi-host file synchronizer which can:

- allow upload and download of files from a central server (hub) through client applications with necessary authorization.
- automatically synchronize changes made by client applications unto the corresponding copy of the file in the server.
- handle all possible conflict/non-conflict scenarios of file synchronization (ie. combinations of Create, Edit, Rename, Delete) involving multiple client applications.
- enable file sychronization through both desktop (all platform) and mobile (Android) clients.

2 Strategy

High level workflow

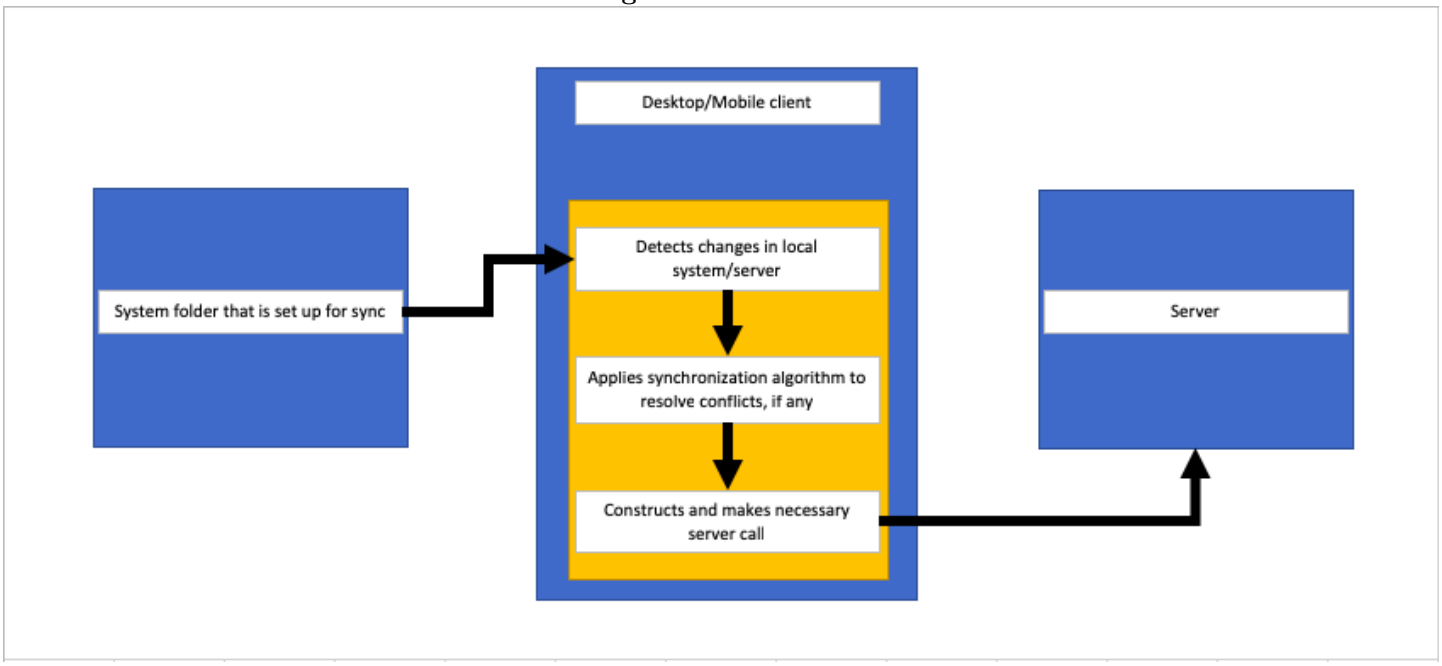

Figure 1: Flow of control between client and server

In our Agile workflow, we will start by building a basic prototype of the desktop client. This prototype will perform functionalities of Create, Edit, Rename and Delete for non-conflicting files for single user. It will also have a *Upload* and *Download* button to manually invoke synchronization. Then we will extend this prototype to handle conflict scenarios. The potential conflict scenarios and their resolution are stated in table 1.

Once the conflict resolution algorithm implementation is stable, we will develop the mobile client prototype followed by testing conflict resolution using mobile-desktop client pair. After satisfactory testing, we will move to the final phase of introducing automatic file sync using *file system notification* and *server polling*.

Table 1: Conflict scenarios and resolutions

No.	Operation 1	Operation 2	Resolution
1	Create	Create	Upload both files, one with the original name and one with a prefix (eg. (1) or 'copy')
2	Create	Edit	Upload edit changes to the original file and upload new file with a prefix (eg. (1) or 'copy')
3	Create	Rename	Rename is applied first followed by upload of new file
4	Create	Delete	Delete file first and then upload the newly created file
5	Edit	Edit	Possible strategies: • Send server error for both requests • Upload two copies of the file, each with different edits
6	Edit	Rename	Rename original file and upload document with edited changes as a new file
7	Edit	Delete	Delete original file and upload document with edited changes as a new file
8	Rename	Rename	Rename original file on server to name1 and upload another copy of the file with name2
9	Rename	Delete	Delete original file and upload another copy of the file with the new name
10	Delete	Delete	No conflict

Operation 1 and 2 are performed on/for files with the same name by same/different users from different devices.

3 Tools/Technology

At first, we use GitHub for code hosting, GitHub is a hosting platform for open source and proprietary software projects. It benefits for the team work projects because every member in the team can modify code and pull requests.

We need to write documents use LaTex , Overleaf is an online LaTex editor that we make use of.

Before coding, we apply Mockingbot to design user interface and set standards for what the clients will look like.

For the server, we use Amazon S3, it is a simple cloud storage service. The main reason we use that instead of virtual machine is security, Amazon S3 uses the Advanced Encryption Standard for 256 bits, AES can quickly encrypt and decrypt in software and hardware. When accessKeyID and secretKeyID are exposed to public, the Amazon S3 will send an email to user.

For the desktop client, we plan to use JavaScript/html/CSS language and the editor is Visual studio code. The structure we use is electron because it integrates *Chromium* and *Node.js* very well together, interfaces and algorithms can be combined in an orderly manner.

For the mobile client, we use Java on Android studio platform. And for the database, we use Structured Query Language on MySQL.

At the step of web debugging, we decide to use Charles. Charles is a HTTP proxy server, when the users send requests to server or get responses from server, Charles can monitor all data.

Testing is a critical part, we decide to use unit test and we preliminarily use Mocha and Junit. Mocha is a testing tool for JavaScript, it is more flexible than other test frameworks. Junit is the most basic testing structure for Java, we should apply it to test mobile client.

4 Team protocol

The main method the team choose is that agile software developing, which means that the development is iterative and step-by-step. The project divided into several individual small projects and during developing these small projects, it goes through the requirement, plan, design, development, tests, deploy, review and then go to the next small project.

There are two types of meeting that the team will be held. The first type is held every week which all members should attend. During this meeting all the team members review the work done last week finding the potential problems and planning for the next week. It usually lasts for 1 hour.

The other type is held between 1-2 days and this is not for all members but between the members who have the same tasks assigned from last week. This meeting usually does not involve the decision making but the attendance review what they have do after last daily meeting. It lasts for not longer than 5 minutes.

All team members should push at least one commit every two days. Once a member pulls request on the git, at least 2 members should view changes and approve before it merges on the master branch of the repository. If some members request changes, the member who pulls request should be fixed or give a satisfactory explanation to these changes.

5 What we have done within recent weeks

Totally we have 10 meetings within 2 weeks. We have the first meeting to know each other after the team members are allocated and discussed the group name and choose the team coordinator and registered to the GitHub by using Git as required. Analysed what should be achieved in the final and distributed task to each member to looking through the details about the server, unite test, mobile development, desktop development, file sync and conflict resolution on the second meeting.

After that, we make decisions of each section within a day. We choose to use electron framework to develop the desktop application and java to develop the Android application. Start working with Amazon AWS as service and make it work within a day in the next meeting.

On the next several meeting, we start working with the electron to develop the desktop application and achieve the upload file function. Also, designed a user interface of the desktop application by using MockingBot.
To start working the initial report, we arrange a meeting to analyse the initial report requirement and make a full plan over the whole development process and distribute into 6 sections to members to working with.

References

[1] GitHub, [online]Available at: `https://github.com/`(Accessed: 3 February 2019).

[2] ElectronJS, [online]Available at: `https://electronjs.org/`(Accessed: 3 February 2019).

[3] Amazon S3, [online]Available at: `https://aws.amazon.com/free/storage/?nc1=h_ls`(Accessed: 3 February 2019).

[4] Charles, [online]Available at: `https://www.charlesproxy.com/`(Accessed: 3 February 2019).

[5] Mocha, [online]Available at: `https://mochajs.org/`(Accessed: 3 February 2019).

[6] Junit, [online]Available at: `https://junit.org/junit5/`(Accessed: 3 February 2019).

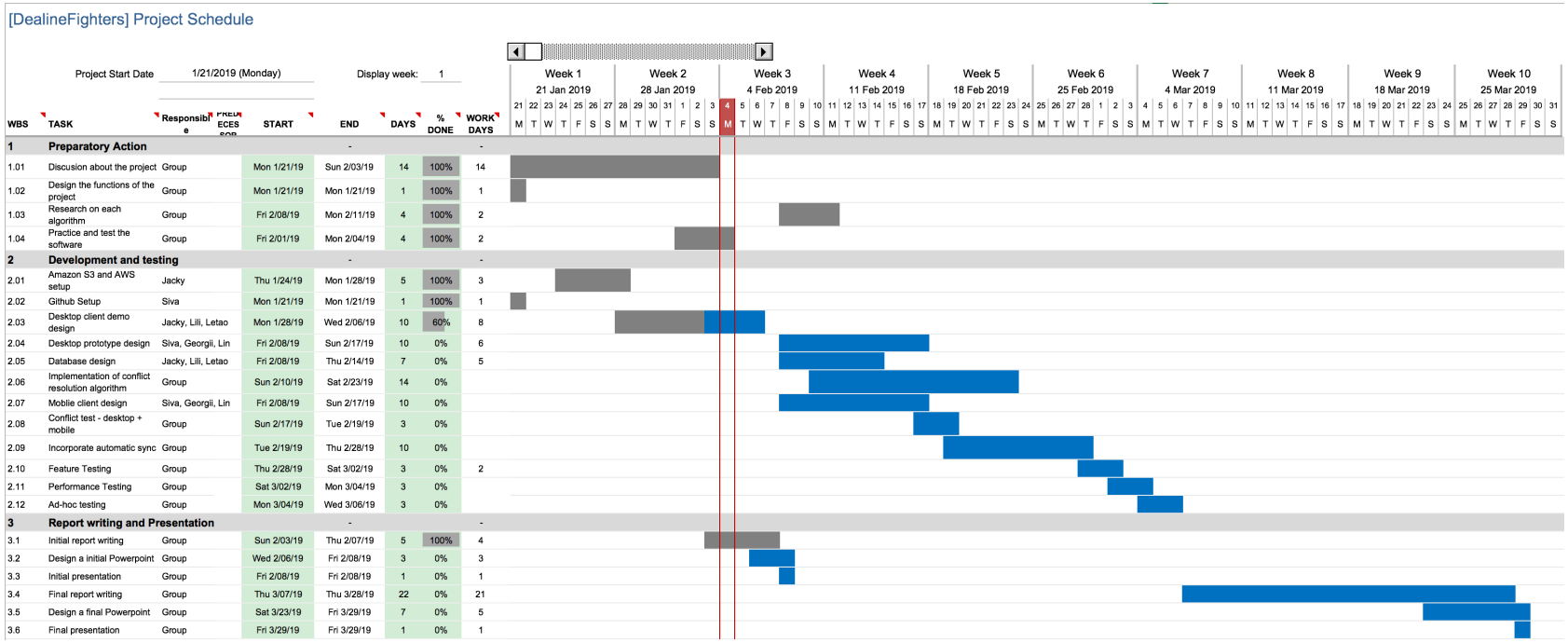

Figure 2: This is Gantt chart which illustrates the plan of our group within 10 weeks.