CS673S15 Software Engineering
Project 12 - Communication Tool
Project Proposal and Planning

Your project Logo here if any

Team Member	Role(s)	Signature	Date
Oscar Ingham	Environment and Integration Leader	Oscar Ingham	02/08/2015
Yuyan Zhang	QA leader	yuyan zhang	02/07/2015
Shishuang Shu	Requirement leader	Shishuang Shu	02/07/2015
Xinzhu Xu	Testing	Xinzhu Xu	02/07/2015
William Wilson	Team Leader / Configuration	William Wilson	02/09/2015

Revision history

Version	**Author**	**Date**	**Change**

1. Overview

Nowadays, we do most of our business or projects in groups which need collaborate as well as communication. No matter you are a leader of company, or you are a member of a group project, the effective communication is a key factor that will influence the collaborate and outcome of every business or work. We can list some examples of communication tools, such as an email hosting provider, a phone call system, SMS and so on. These communication tools do help people communicate with each other in many ways.

However, each of these tools has its own shortcomings. For the email service as most of the company use, imagine that everyday a staff may receive hundreds of mails from his boss, his managers, his colleagues, his customers, and even his family, with the content of notification, project progress, or even complaint or spam advertisement, which makes the communication inconvenient. Also the email communication is not rapid because you have to log in to your email account to get emails. For the phone call, though bringing the advantage of direct and quick communication, it cannot store information that user shall rely on for reference later. For the SMS, it always does in one-to-one communication, which can not apply to group conversation. Besides, since most of the project work should be done in the office, people use computers much more than cellphones, making this service impropriate and informal for office work.

By analyzing advantage and disadvantage of these communication tools, we arrive at a conclusion that we should have a new tool to facilitate the communication more effectively and conveniently. The way we can achieve this is to take advantage of different communication tools and make a right combination of them.

The basic function of this communication tool is that user can send messages to whoever he wants or to a group instantly with no latency, and also the record of each conversation will be kept for further reference. Files sharing and post publishing are also good function for group projects. Searching information from conversation and files is included. User management is synced to make it appropriate for project management. Further function details will be discussed in the high level requirements.

This tool is not only for people who work in group projects to use, but also for daily communication meaning that the usage of this tool doesn't have the limitation in work. This tool can also do a favor when people want to chat or leave a message to their friends and family.

2. Related Work

There are several popular peer-to-peer messaging platforms available on the market today. Some services (i.e. skype) are geared mostly towards end-users for personal communication, while several (i.e. Slack, HipChat) are marketed towards businesses as a a faster, collaborative alternative to e-mail. These products aim to be the primary communications channel for teams working together on a project. These products organize discussion topics via multiple channels, including group chats and private one-on-one communication.

Many of these tools offer tight integration with other popular third-party services, including GitHub, Pivotal Tracker, and JIRA. These services allow event notifications to appear and be discussed in a centralized location.

Most of these services are hosted on third-party servers and offer SSL-encryption to protect the users.In addition to text-based chat, some of these services (i.e. HipChat) also offer voice and video-based communication services. Also, many of these services offer a number of cross-platform clients, allowing the user to participate in discussions from any operating system, mobile device, or web browser.

Other services:

a. Slack
b. Skype
c. HipChat
d. Flowdock
e. Campfire

3. Proposed High level Requirements

a. Functional Requirements
    i. Essential Features
        1. Low-latency communication between users
        2. Allow private conversation and group conversation
        3. Always available (no data loss)
    ii. Desirable Features
        1. User profiles

a. User registration?
2. @mentions for off-line notifications
3. search(record of the chatting conversation as well as the files we upload and post and so on)
4. archive
5. grouping
6. encrypted chats
7. See whether users are online or offline
8. Identity management (user management)
9. emoticons

iii. Optional Features
1. File sharing
2. Send voice message
3. Comments on posts

b. Nonfunctional Requirements
i. User-friendly
ii. Designed well
iii. Secure

c. Implemented Features (to be completed at the end of project)

4. Management Plan

(For more detail, please refer to SPMP document for encounter example)

a. Process Model

i. We will be using an iterative and incremental development process. It will be “Agile-Lite”. There will be an initial planning phase, and then three iterations. The goal of each iteration will be to produce working, well tested software.

b. Objectives and Priorities

i. Create a project communication tool that will facilitate communication between team members on a software project.
ii. Develop a tool that can be used in the academic setting for group-based class projects or research projects
iii. Complete iterations on time. Each iteration must produce quality software that has been tested and documented.

c. Risk Management (need update constantly)

i. Insufficient python skills. Oscar is the only team member with sufficient experience using python. The other team members will need to learn python for this project to succeed.
ii. Time management will be critical. All five team members are either full time students or have full time jobs. Team members will need to manage their time wisely, and help one another when necessary.

d. Monitoring and Controlling Mechanism

i. We will two meetings per week. One meeting after class on Wednesday and one meeting at 4:00pm on Saturday using Google Hangouts. Pivital Tracker will be used to create, assign, and track tasks. GitHub will be used to manage revision changes.

e. Schedule and deadlines (need update constantly)

i. Complete rough draft of this SPPP document by 11 Feb 2015

5. Quality Assurance Plan

(For more detail, please refer to SQAP document for encounter example)

a. Metrics

i. Definition

a） product metrics

Complexity: the number of lines of code (LOC), number of features of production, cost of time
Defect density: number of defects relative to the software size
Mean time of failure: (according to our communication tool) interruption frequency, signal emission failure
Customer problems: total number of problems encountered by customer while using the product
Customer satisfaction: the survey from professor, and other classmates

b） process metrics

Project milestones: time of breakthroughs in difference phrases, to determine whether project is on schedule.
Rate of test case execution & number of passed test cases: determining whether testing is proceeding on schedule.
Defect detection rate: number of defect detected per period of time
Defect resolution: number of solved defect compared with detects

ii.Results (to be completed at the end of each iteration)

b. Standards and tools

(e.g. documentation standard, coding standard, tools to be used etc.)

Documentation standard: all documentation should follow the standard is described by professor(all the forms can be found on sppp).
Coding standard: the standard for python will be found at https://www.python.org. the standard for javascript will be found at http://www.w3schools.com/js/. the standard for database will be found at http://www.postgresql.org/.
Tools：
IDE: Sublime Text 2
VM: Virtual Box
OS(s): Ubuntu Server 14.04.1 LTS
Persistence Storage: MySQL Community Server 5.6.23
PM Tool(s): Slack & PivotalTracker
Web Programming Language: Python 2.7.9
Web Framework: Django 1.7
VCS: Git 2.2.2
Modelling Tool: Visual Paradigm 12

c. Inspection/Review Process

(e.g. describe what are subject to review, when to conduct review, who do the reviews and how ?)

Software requirement reviews: requirement leader
Architecture design reviews: design leader
Detail design reviews:design leader
Test plan reviews: QA leader
Verification and validation reviews: QA leader

d. Testing

(e.g. who, when and what type of testing to be performed? How to keep track of testing results?)

A separate document about testing result should be linked here.

Regression testing : if there is new part added into original part(eg. new functionality added), retesting of the whole part should be launched
Unit testing: test individual unit of source code
Integration testing: when assembling developed parts together, we should give integration testing to make sure these parts work together correctly.
Functional testing: a type of black box testing, we should provide the required inputs(eg.user profile, messages, operations) , and compare the actual outputs with the planned output.
Nonfunctional testing : testing perfomance, load/stress,usability, recoverability, compatibility, security,installation, serviceability.
Acceptance testing: test if all the requirements in plan are met

e. Defect Management

(e.g. describe the criteria of defect, also in terms of severity, extend, priority, etc. The tool used to management defect, actions or personnel for defect management)

a) standards
Criteria of severity:
*Critical: cause the application to crash with significant frequency and more than two requirements not to be met
*Serious:cause at least one planned requirement to be unmet, and no alternative methods.
*Trival: the defect does not damage the usability of the system and the desired results can be easily obtained by working around the defects
*Medium: the defect that is improving system where the changes are related to thelook and field of the application
Criteria of priority:determine the order to fix the defect
*high:fix as soon as possible
*medium:should be fixed in its own course of development activities
*low:can be fixed after more serious defects fixed

b)defect types

Documentation: Documents, comments, or messages are not understandable or wrong
Syntax/Static: some code defects can be detected when compiling
Interface: Incorrect design or misuse of interfaces (class, procedure, or data type interface is incomplete or wrong or used in an inappropriate way, objects are invisible, etc.)
Data: wrong variable value, violate the variable structure
System: Problem with timing, synchronization, network, hardware or the like
Environment: Defect in development environment or support systems

c)tools

For every iteration process, the team members who find the defects should update it to the GitHub.
Define defects: any obstacles for user including program crash, data corruption, display of an error message.
Use "Pivotal Tracker" to track the bugs in a user story and discuss how to improve the software. After a bug is modified, it should be committed on the GitHub with a clear 'commit name'. For example, "Modified bug1 in View2(Problem 3 is solved)."
At the end of iteration, the testing and QA leader should exactly write the total number of bugs and reason.

6. Configuration Management Plan

(For more detail, please refer to SCMP document for encounter example)

a. Configuration items and tools

Git/Github Slack PivotalTracker GoogleDocs
Individual requirements documents, software model

b. Change management and branch management

Ensure that the change management process is agreed upon, communicated to, and adhered to by team members and provide checks and balances so that no single body has the power to approve, release or update changes to the production environment without formal approval.

c. Code commit guidelines

Commits should be as concise as possible and easy to review and merge
Everyone should comment their code. Particularly, if someone change shared common code, you should also make at least a best-effort attempt to make sure all of the engine/backend code stays working. Try to assure that ScummVM compiles with every commit. In case of regressions, this helps to track down the commit introducing the regression.

7. References

a. https://www.hipchat.com/
b. https://slack.com/

c. https://www.flowdock.com/
d. http://www.ask.com/business-finance/examples-communication-tools-68df9290603d6786
e. https://www.americanexpress.com/us/small-business/openforum/articles/8-great-communications-tools-for-small-business/
f. http://istqbexamcertification.com/what-is-the-difference-between-severity-and-priority/
g. http://www.cs.kent.edu/~jmaletic/cs63901/forms/DefectTypes.pdf
h. http://wiki.scummvm.org/index.php/Commit_Guidelines
i. Eric Braude, Michael E. Bernstein. Software Engineering: Modern Approaches (2rd Edition). Wiley. (ISDN:978-0-471-69208-9)

8. Glossary

Hipchat Flowdot Scrum