|  |
| --- |
| University of Pretoria |
| Project Management Information |
| COS301 - Zeon |

|  |
| --- |
| Zenadia Groenewald, Bernhard Muller, Lecton Ramasila  10-18-2014 |

Introduction

This document serves as an explanation of the software development process undertaken in the creation of the Stream2Me project, as well as how it was implemented; this document also includes information pertaining to the profiles of team members, plans for issue management, information about the project’s status over the course of the development process, the functionality that the team was unable to implement, and finally the discussion of the main risks and challenges faced in the development of the StreamMe project.

Software development process

The software development process followed for the Stream2Me project focused on the completion and efficiency of individual modules within the system, which would later be integrated into the project to form the whole. This was mainly to ensure that features were not only complete, but that they worked efficiently and correctly as separate modules which could then be integrated into the main system with minimal changes.

This particular method also allowed for on-the-fly development and fixing of individual modules and features; further aided by the use of version control by the GitHub repository.

First it was important to ensure that the core functionality of the project was working and to the clients’ description; as such, the first modules created were those performing the video/image streaming, the audio recording and streaming, and finally the message sending.

Gradually, however, these features had to be adjusted throughout the project’s progression, as not only were the first prototypes slow and expensive to the system, but they were also an ineffective solution in comparison to what the Stream2Me project presents in its completed state.

Following the core functionality, it was important to create a sound, stable and user-friendly interface for both the desktop and mobile application. This was a challenging feat, as the extent to which we could develop the interface was largely limited by the development environment. This aspect was not only important for the clients, but for the ease-of-use of the end-users as well.

With all the core aspects of the project, it was important to create a suitable back-end in order to manage the distribution of messages, as well as managing the database that contained the various users’ information, such as e-mail addresses, passwords, etc. The back-end was also partly responsible for the security features required by the front-end of the system, such as the salt key encryption.

Team profile

The profile of the team and each of the member is as follows:

Bernhard Müller:

|  |  |
| --- | --- |
| First Names: | Bernhard Wilhelm |
| Surname: | Müller |
| Studies: | BSc Computer Science |
| Responsibilities: | Desktop application, security layer, back-end functionality, front-end functionality. |

Zenadia Groenewald:

|  |  |
| --- | --- |
| First Names: | Zenadia |
| Surname: | Groenewald |
| Studies: | BSc Computer Science |
| Responsibilities: | Desktop application, interface design, database, front-end functionality, documentation. |

Lecton Ramasila:

|  |  |
| --- | --- |
| First Names: | Lecton |
| Surname: | Ramasila |
| Studies: | BSc Computer Science |
| Responsibilities: | Mobile application, interface design, front-end functionality, back-end functionality. |

Issue management

Project issues and development issues were handled in a manner best suited to our design methodology, that being to develop the system in the most agile and flexible way as possible.

If an issue arises with the program, the team looks at which branch of the project on the repository is furthest ahead and, depending on the issue, consult whoever has the most expertise and/or knowledge about that specific problem.

Once the issue is identified and a course of action is established, the erroneous version of the project or module is handled first, while the latest version of the system is set aside to be dealt with later; this process accompanies a report stating just how far each phase of the project is, along with any errors or challenges faced along the way.

Project status over time

The project’s status and progress is monitored with the assistance of the GitHub repository, which provides version control and also allows one to extrapolate various statistics and effectively monitor a project’s progress.

Versions were determined by significant changes made to the system, such as the addition of new functionality, or the repair or large errors or bugs within the system; essentially, every time a significant change was made to the system.

Functionality not implemented

There are few features that have been excluded from the project’s scope and functionality, mostly for reasons such as time constraints or dispersed workflow.

The following is a list of the excluded and/or unimplemented features of the system:

* Screen manipulation: this would allow a user to receive a video feed of the sender’s desktop and allow them to not only see what the sender is doing on their computer, but also to be able to manipulate the images and perform actions as though it were their own desktop. This was a desired feature, but not vital to the project’s success. This feature was omitted due to time constraints.

Main risks and challenges

The risks and challenges in the making of the Stream2Me project were thankfully few; the main risks involved in the creation of this system, especially with the methodology used, was that the system may not have been finished before the designated due date and it might have not been to the clients’ specifications.

The largest challenged were to get the mobile and desktop applications to function simultaneously and from the same server. Ensuring that messages were transmitted quickly, efficiently, and intact was a particularly daunting challenge; this was mostly due to the heavy role of the security layer and partly due to the fact that every aspect of the project needed to utilise open-source solutions.

Another challenge faced was that of constructing the program into a simple executable application – predominantly because build technologies such as Maven and Gradle did not support many of the features we had included in the project, such as the use of JOGL (Java OpenGL) and Netty, among others. This forced the project to be built from an IDE (namely NetBeans), which is not an ideal solution.

The largest challenge was creating a project from a seemingly modest set of specifications and transforming it into something significant and relevant in our field of study. Because the restrictions on the project had already limited it in many way, it was challenging to create it in a way that made it noteworthy and still performed as required. The restrictions, however, also allowed the project to become more relevant in the field of computer science, mostly because of the functionality that had to be written from scratch and the complexity of the system as a whole.