

## **BLEKINGE TEKNISKA HÖGSKOLA** karlskrona, sweden

**May 14, 2015**

ET1446: SOFTWARE DEVELOPMENT FOR TELECOMMUNICATION SYSTEMS   
  
*This document is a superset of the Project Proposal document in the context of building an enterprise chat system for XtremeCompany.*

PROJECT specification V1.3

### **Group: SWAT Kats**

|  |  |
| --- | --- |
| Sokratis Papadopoulos | Jibraan Singh Chahal |
| Johannes Grohmann | Jyoti |
| David Alarcón Prada | Akanksha Gupta |
| Gautam Vij | Rohit Raghav |
| Inanc Gurkan | |

**Publication Date: May 14, 2015 Version Number: V1.3**

# PROJECT SPECIFICATION

## **1. Preface**

This document contains a project specification to create a chat client called “**LetsTalk**” for XtremeSecurity. The document is specifying important aspects which are to be executed by our team “**SWAT Kats**”.

In the following we will first provide a glossary with a list of abbreviation in chapter 2. After that, the environment as well as the problem are described in section 3. Section 4 contains our solution for the described problem, while section 5 states some limitations that our product will not include. In addition, a rough time plan is given in section 6 and a project organization in section 7. Then it is configuration management at section 8 and progress tracking in section 9, while quality control is presented in section 10 and risk management in section 11. Finally we present the system release plan and we refer to the references.

## Version history of this document:

***Release v1.3 on 2015-05-14***

- Risk name changed and two sub-risks and described in more detail in section 11

- Project plan picture updated

***Release v1.2 on 2015-05-05***

- Modified section 8.2 providing more detailed descriptions on how we build the code

- Modified section 8.3 elaborating on how we document the release

- Added a table that shows the identified risks together with occurrence probability and effects

- Added group meetings in project plan picture at the end of section 6

***Release v1.1 on 2015-04-30***

- Text modification: removed unnecessary text on Documentation plan

***Release v1.0 on 2015-04-27***

- Initial release

## **2. Glossary and abbreviations**

In this section are defined technical terms that are in this document in alphabetical order.

* **GUI**: Graphical User Interface
* **IP**: Internet Protocol.
* **Metadata**: information about a certain item’s content
* **WBS**: Work Breakdown Structure.

## **3. Background**

We are required to design a communicating tool among the employees of the company XtremeSecurity and satisfy the specific needs of its travelling salesmen.

The employees will be able to communicate over an encrypted network and share binary files as needed. The main purpose is to provide our clients with the means of fast, efficient and reliable communication over the distance with the goal to help them increase their time utility.

The salesmen often require access to the latest marketing materials, white papers, company brochures and user manuals in order to showcase the products to potential customers. Proper update of new plans in the respective sector of sales is the top priority for them to present best plan to their clients. The work office of XtremeSecurity is situated in Stockholm and are supposed to travel to different destinations to meet with their clients. They are required to convince their clients over some package and may need some details from their center office. Support Engineers out on the field may also need more data from their headquarters to show it to their clients. Our product “**LetsTalk**” will resolve the issue of carrying around the updated soft copy of various packages to be shown to their clients.

## **4. Proposed sOlution**

The project aims to provide the customer with a secure fully functional communication system. The customer for the project is a company, which wants its employees to properly communicate and work together. We will provide all the various collaboration tools that allow its employees to have secure communication from different geographic locations.

The communication involves text messages and exchange binary files of arbitrary size to access certain documents such as the latest marketing materials, white papers, company brochures and user manuals in order to showcase the products to potential customers. The customer also needs on field engineers to be able to securely download patches and service packs to resolve customer issues on location.

A central database is required to store user information metadata associated with each sent or received message, so that every user can see the date and time a message was sent, the name of the sender and recipient, if the message was read by the recipient and when it was read and can browse the history of their conversations with various peers. To provide fast service the messages should be sent to offline users as e-mails to reduce the delay. User can see a message status (read/unread, message time stamp). A call record, timestamped information about the sender and receiver of messages, their corresponding IP addresses, along with metadata such as message size, type (message or file) should be kept in a SQL database and only authorized users (administrators) should be able to access it. The customer requirement also includes an administrative role: delete a user, block and unblock a user, emergency broadcast message like some crash problem. The administrator can view statistics of records graphically as tables/graphs (GUI) available on main database server (SQL).

The project will establish a secure chat for authorized employees .The administrative access will be granted by authentication based on username/e-mail address and password. To ensure the security of confidential client data the chat messages will be encrypted. File exchange with live-status (used/remaining time for upload/download) will be provided. A user can temporarily block a user from the address book and an address book for the users will be displayed with associated status (offline, available, idle, busy: available & busy are handled manually by user, whereas offline & idle by the system).

## **5. Limitations**

Although the chat messages are sent instantaneously, this is no real-time client. Since messages and files are usually transmitted over the internet, delays of various lengths can occur. Furthermore in extreme cases (e.g. internet connection lost), messages or files can also be lost, transmission is never guaranteed and transmission time may vary.

The client is not supposed to support any acoustic support or emoticons in the chat. Only private chats, are supported (no group-chats or broadcast options).

## **6. Time plan**

After describing the background, our proposed solution and its limitations it is time to present the time plan of the project. We have developed a WBS structure that shows the whole picture of the project, broken into small parts. Also, we have written down all the milestones and tollgates that this project will follow within the time budget.

WBS LEVEL **3**

WBS LEVEL **2**

WBS LEVEL **1**

|  |  |  |  |
| --- | --- | --- | --- |
| **MILESTONE (blue) / TOLLGATE (green)** | **Deliverable** | **Receiver** | **Time** |
| Deliver initial project proposal | Project Proposal | CEO | **13 April** |
| Deliver updated project proposal & project plan | Project Proposal, Project Plan | CEO | **20 April** |
| **Start line** of Software development | Split of work | Dev Team | **20 April** |
| Deliver project specification & SRS | Project Specification & SRS | CEO/Customer | **27 April** |
| Deliver design document | Design Document | CEO/Customer | **4 May** |
| GUI implementation | GUI of Software | Dev Team | 20 April - 5 May |
| Texting implementation | Texting of Software | Dev Team | 20 April - 5 May |
| File Transfer implementation | File Transfer of Software | Dev Team | 20 April - 5 May |
| Address Book implementation | Address Book of Software | Dev Team | 20 April - 5 May |
| Statistics Implementation | Statistics of Software | Dev Team | 20 April - 5 May |
| Database Implementation | Database of Software | Dev Team | 20 April - 5 May |
| Connecting and constructing Alpha Version | Package of Alpha Version | Dev Team | **7 May** |
| Meeting with CEO for product demo, providing basic functionality of alpha version | Software | CEO | **10 May** |
| Deliver Acceptance Test Plan and the Alpha Version of the software | Alpha version & instructions, Acceptance Test Plan | CEO/Customer | **11 May** |
| Software debugging (each part is being debugged by the people responsible) | Debugging results | Dev Team, CEO/Customer | **18 May** |
| Beta version of our software accompanied with a full documentation | Beta version & Full documentation | CEO/Customer | **18 May** |
| Working on issues, constructing a stable final release | Stable software | Dev Team | **25 May** |
| Final Release of Chat → Final version of the program. Show to CEO/Customer.  Explain the support provided. | **Final Release** | CEO/Customer | **28-May** |

## 

## 

## 

## **7. PROJECT ORGANIZATION**

## **8. CONFIGURATION MANAGEMENT** 8.1 version management

## A version control system is a repository of files, often the files for the source code of computer programs, with monitored access. There are a lot of possibilities for new feature upgrades in future in different components of the system. Since we have different members of our team working on different components, it is very important to record the changes and build operations done by different developers from our team. Git system, a distributed revision control system is used as a support for distributed, non-linear workflows. This system is used as a part of learning phase for this course and to record all the changes made in the development process to ensure smooth working and avoid any interference between each other.

|  |  |
| --- | --- |
| GUI LAYOUT | |
| Login interface | Sokratis Papadopoulos |
| Basic GUI for the messages, user/admin statistics, admin tools | David Alarcon Prada |
| GUI for other users’ status in address book, the user account configuration (change password etc.) | Sokratis Papadopoulos |
| GUI for address book, user status, file upload/download | David Alarcon Prada |
| TEXTING / FILE TRANSFER | |
| Data encryption & Connection Features | Johannes Grohmann |
| Live message update and Message History | Akanksha Gupta |
| File upload/download feature | Jyoti |
| Connecting the email to chat (for offline) | Johannes Grohmann |
| Developing the message status function | Akanksha Gupta |
| DATABASE AND STATISTICS | |
| Login tables and queries | Jibraan Chahal |
| User account configuration tables and queries | Rohit Raghav |
| Group Chat tables and queries | Gautam Vij |
| Users activity analysis, admin features | Inanc Gurkan |
| Single Chat tables and queries | Jibraan Chahal |
| File transfer tables and queries | Gautam Vij |
| Address book tables and queries | Rohit Raghav |
| Block list queries | Inanc Gurkan |

## 8.2 system building

## A method of building in which prefabricated components are used to speed the construction of buildings. Main focus would be put in building a decentralized structure of different functionalities that are to be implemented. Smart build operations will be used to for minimal compilation using "make" operations. We will be writing our own build scripts for building the jar file. All file paths to be compiled will be defined in the makefile and an executable file will be provided both for the server and the client. These files will be stored and collected from history of git version management system.

## The application will be developed in Java that will be platform independent i.e windows or any other operating system equipped with javac compiler. At the end of the development of application, "Junit" will be used to run automation test and to confirm the working of all the functionalities. All the individual XML files generated by the JUnit task will be used to provide a browsable report of the test cases results. We have used Microsoft Word for document generation. The approach defined might change depending upon the allowed features of the tools or methods mentioned.

## 8.3 release management

## Our application will have a product release on 28th of May. The release document will list the components and subparts of the components of the product along with the version of the component used in the bundle. The release will be tested using the tests listed in testing plan. Checklist will contain the functionalities mentioned in the project proposal which will be verified by the same tests specified. All libraries and the compilers used will be properly documented along with the build information and other tools configuration used. One executable file or an installer file will be provided during the release of the application.

## **9. PROGRESS TRACKING**

Due to the lack of a project manager, all progresses of all tasks have to be supervised by all team-members themselves. Since every task has a responsible team member, this member is also responsible for the *delivery of the task on time.*

Communication is crucial on this topic. It is very important to communicate upcoming delays or resource contentions with all relevant team members, in order to be able to react properly.

The weekly meetings perform a key role here. In this meetings, every team member has to report the progress of his/her assigned task to the plenum. Any problems and/or delays shall be reported and discussed by the team. If necessary additional resources can be assigned to the task, in order to finish in time. Although it is in the responsibility of the assigned team-member to report delays, problems or resource contention, it is the authority of the whole team to deal with upcoming problems.

Delayed or missed deadlines shall also be included in the Project-Libre-file in order to communicate them to all team members, as well as the CEO in severe cases.

## **10. QUALITY CONTROL**

The successful outcome of virtual anything requires good, up-front planning. And quality is no exception. Controlling the quality of product throughout development is imperative to ensuring that the end deliverable is going to be at a level that maximizes the enhancements and minimizes any issues. Following we explain how is this going to be the case in our program.

Essential is the communication not only among the team but also the establishment of a great connection with the CEO & customer. Having a constant communication with them will lead us to useful feedback that will perfectly adjust the product into customer needs. So, we value communication as a great principle for achieving the best possible quality of our product and it will be the best way to build exactly the product that customer wants. Alpha and Beta versions of our product will boost their feedback and will lead us to develop exactly the product they demand.

As for internal work, as developing the software, weekly meetings can lead the control checking. Each member has been assigned with a part of the project and he will inform and present to the rest of the group for his progress on it. That way, everyone can check, feedback and suggest ways to overcome problems that will may occur. Plus, we will be constantly checking if whatever was promised in proposal will be met on the final release, point by point.

Regarding software quality control, we are planning to use JUnit that will support us in accomplishing a great testing period for our software. In that period, we will also be able to clear out the code, break down longer functions, making it easily understandable by everyone.

**11. RISK MANAGEMENT**

Change IN SOFTWARE

After an appropriate project proposal and workload assigned; technologies to be used were selected. Change of software development approaches risk can either be offered by the developers, what is 'dev team sourced'(a); or forced by the owner of the technology, what is 'outsourced'(b).

The technology that is being used may not be enough satisfy developers' desires. Technologies that have disadvantages and cause limitations are offered to change by developers. As for the software factor; the technology with changes -a big update in Java, MySQL or Connector Libraries in our case- may affect the software. The effect is proportional to the changes made in the areas of the technology that are being used in the project. Small effects would be compatibility problems. Yet, we have stated that the customers are going to use the software in their PCs without encountering any compatibility problem.

a) Changing the Software Used; Using a New Software (Dev Team sourced risk)

Foreseeing this risk caused by developers is though; as the technologies used were selected by the same Dev Team. In other words; if the Dev Team needs a change in technologies, this will be the result of bad experience with the technology right in this project. The technology that is subjected to change may/will affect the project depending on how far is the project developed, and the number of dependencies used the technology yet. This is a risk that affects project crucially. Team SWAT-KATS are prepared for this risk. Technologies were studied well before the start of development and some technologies changed (such as FTP). Brainstorming sessions and team meetings were handy lowering the impact and the probability of happening of this risk. The risk will have an impact on project as the project is developed in parallel, each part is dependent to one another; yet SWAT-KATS have managed to lower the probability.

b) Changes in Used Software (Outsourced risk)

This outsourced risks means, any changes or updates in the technologies that are being used may affect the project. This risk is only dependent to the owners of the technology. This type of risks are not meant to happen in this project's timeline; as the project is planned to finish around 2 months and no crucial changes expected in the technologies that are used. As stated before; Java is being used in this project and any changes in Java (by Oracle) have the probability to affect the project even though it is an "improvement". Technology-sourced risks have minor effects in this project as it can be fixed quickly. The possibility of happening is low. So, this risks is not considered as a "dangerous" risk nor "likely to happen".

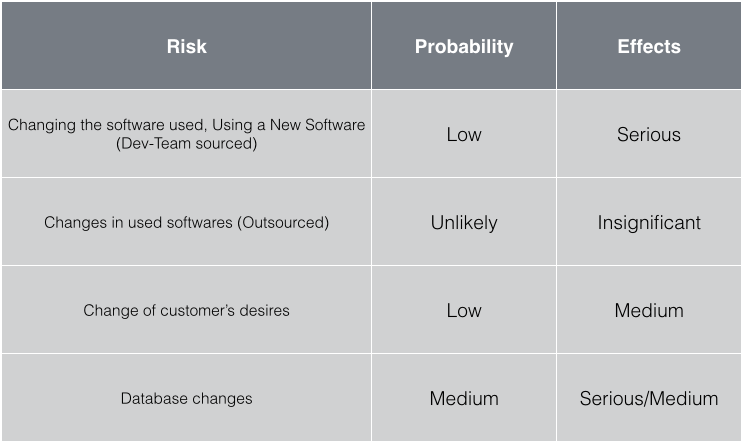
Although we have lowered the probability, a few was encountered. We have been successful to manage this type of events throughout the project in a state what has too small effect to be stated as a risk.

Change OF CUSTOMER DESIRES

Software projects are subject to change in anytime. A good software is improvable. The project is being planned and developed with fully understanding of that statements; yet, change of customer's desires stands as a problem. These changes affects project plan and the project structure. Analysing customer's needs is crucial for avoiding this risk. It is hard to foresee the impact of this risk to the project but it can be stated that it will affect the project. Misunderstanding and/or miscommunication between client and the team leads to this problem. SWAT-KATS analysed the user needs and proposed improvements -as additional functions. The project structure is rigid at this moment. All pros and cons and limits of technologies were discussed, and then comes the rigid structure.  
  
This risk is not likely to happen, and if it happens; the impact won't be crucial because of the maturity of the structure. By "rigid", it doesn't mean it is not "changeable" but many scenarios have been thought of. Thus, this risk is not considered as a "dangerous" risk.

## Database changes The project is developed in parallel, following the work plan with respect to the workload assigned. Database has a crucial part. Database needs to represent all information in an efficient way. A not well-structured database still can do its part, but difficulties will occur in all other areas as database is dependent to every other area, vice versa. Changes in database will affect whole project in different ways. It may improve or worsen the different areas, but it is easily seen that; changes in database will lead to changes in project which means it will cost time. Its cost is only time in a good scenario, on the other hand; it may have a huge impact on project.

## This risk is being handled with brainstorming sessions and team meetings where every member can state their thoughts. The database structure is decided and it is not very likely to change as it contains point of view of every member. Database changes may or may not be crucial, but we have trust in the project structure.

****

**12. System release plan**

## 12.1 testing plan

|  |  |  |
| --- | --- | --- |
| Requirement | Time | Test |
| Server Connection | 2nd May 2015 | Access to information on central server |
| User Login | 4th May 2015 | User should be on home page after login. |
| Text Messages Exchange | 7th May 2015 | Messages should appear on both users’ screen. |
| Address Book Update | 10th May 2015 | If a contact is added, then it should appear in user’s address book and they should be able to chat with each other. If a contact is blocked then it should not appear in address book and they should not be able to chat with each other. |
| Files Exchange | 10th May 2015 | Users must be allowed to exchange files with each other and the GUI must show statistics on the used and remaining time for upload/download and a time estimate for completion. |
| Encryption | 14th May 2015 | Encrypted password , messages should be saved to the database |
| Online Status | 16th May 2015 | Status of user should be changed and other contacts should see the same status which he sets. |
| Offline Chat | 20th May 2015 | Test in a scenario with the unread message and offline user to check the email notification service. |

## 12.2 packaging plan

We will provide two .tar packages. One for the server to be distributed to the administrator and the other will be for the clients. Each package can be extracted in the working directory. Each package will have an executable .jar file. Along with it our package will have different sections- bin, lib, src, doc with data as an additional in server package.

* **Bin**: It will contain all the binary executables of the source files.
* **Data** (server only): This will contain the .db files for the database used by the clients for old chat messages, files, address book etc.
* **Lib**: It will contain all the java libraries used up in all the source files.
* **Src**: It will comprise of all the java source files along with image or audio files for GUI.
* **Docs**: In the docs/ directory we have the documents that we produced in our project (e.g., the Project Libre plan, the SRS, the design doc, the acceptance test plan).

We will submit the final release by itslearning & git server on **18th of May**.

## 12.3 documentation plan

### 12.3.1 Installation documentation

We will provide a PDF document where the software responsible of XtremeSecurity can find information about how install & configure the program in each employees’ computers.

Because we are going to provide the program through .jar file in order to let execute the software in all Operative Systems as be possible, we will explain how to install the JVM corresponding to run the program. All explanations will be written with clear sentences and will be attached with useful images.

### 12.3.2 User documentation

We will make a PDF manual where users and system administrators can find information about how use the program.

It will consist of an easy guide, which will contain screenshots about different options available clicking different buttons. Thereby users and systems administrators will be able to learn to use the tool by themselves.

### 12.3.3 Developer documentation

### The main code with English comments in all methods in order to understand the code and to develop future extensions. Then we will generate a Javadoc document, and as developer documentation we will include all the documents delivered in checkpoints.

### A PDF will be provided with the full explanation of the working and structure of the Database. It will contain the graphical representation of the tables and their connections with each other.

### 12.3.4 Documentation time schedule

|  |  |  |  |
| --- | --- | --- | --- |
| MILESTONE (blue) / TOLLGATE (green) | Deliverable | Date | Dependent Upon |
| Alpha version documentation | Installation & User documentation | 14 May (Checkpoint 4) | Alpha version of the code |
| Beta version  documentation | Installation, User & Developer documentation | 20 May (Checkpoint 5) | Beta version of the code |
| Final release  documentation | All documentation complete and ready to ship | 29 May (Final Checkpoint) | Documentation complete. Approved for release. |

## **13. references**

* <http://oss-watch.ac.uk/resources/versioncontrol>
* <https://ant.apache.org/manual/Tasks/junitreport.html>
* <http://www.istc.org.uk/wp-content/uploads/2011/08/Beth-DOCPLN_TPL.doc>