Software Architecture Document

1. Introduction

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict its different aspects. It is intended to capture and convey the significant architectural decisions which have been made on the system.

2. Architectural Representation

A process view will be provided:
- to explain the system processes and how they communicate,
- to focus on the run-time behavior of the system,
- to address scalability and responsiveness.

A development view will be provided:
- to illustrate a system from programmer's perspective,
- to cover maintainability and security concerns.

A scenario (use-case) view will be provided:
- to identify architectural elements,
- to illustrate and validate the architectural design,
- to address learnability concern.

UML diagrams will be used systematically to represent the different views of the system. The goal of these diagrams is to represent our ideas and to let people understand them easily and without misinterpretation.

Sequence diagram will be used to describe the process view.

Package diagram will be used to describe the development view.

Use case diagram will be used to describe the scenario view.

3. Architectural Goals and Constraints

- This software is to be executed in mobile phones, since it is a mobile messaging application.
- The user must have a mobile phone number in order to login to the system, as login requires a mobile phone number.
- Every profile has their own data, such as profile name, profile photo and status. The user can change their profile data, if they adhere the constraints listed below:
  - Status must be at most 70 characters long.

- Profile photo must not be bigger than 250 MB.
- Profile name must be 10 characters or less.

- The user can have a directory of friends for easier messaging. In order to add a friend to the directory of friends, user should use the number of the friend and should assign a name to the number.
- The user must be able to send messages. Messages can be sent to friends in the directory.
- The user also should be able to receive messages.
- The user must be able to see the messages that have been sent or received as categorized according to the user whom have been communicated with, so the user should see the history of messaging.
- The user must get notification when a message has been received.
- Notification should contain the name of the user whom send the message and the message that been sent. If the user clicks at the notification, the messaging application will be opened and the user will see the chat history with the user whom has sent the message.
- Aside from sending messages, the user also must be able to sent files which are photos, documents, videos, audio,contact and location.

4. Scenario View

[This section lists use cases or scenarios from the use-case model if they represent some significant, central functionality of the final system, or if they have a large architectural coverage they exercise many architectural elements or if they stress or illustrate a specific, delicate point of the architecture.]

5. Process View

[This section describes the system's decomposition into lightweight processes (single threads of control) and heavyweight processes (groupings of lightweight processes). Organize the section by groups of processes that communicate or interact. Describe the main modes of communication between processes, such as message passing, interrupts, and rendezvous.]

6. Implementation View

[This section describes the overall structure of the implementation model, the decomposition of the software into layers and subsystems in the implementation model, and any architecturally significant components.]

8.1 Overview

[This subsection names and defines the various layers and their contents, the rules that govern the inclusion to a given layer, and the boundaries between layers. Include a component diagram that shows the relations between layers.]

8.2 Layers

1. Hardware, OS, COTS
2. Basic Elements
3. Distributed Virtual Machine

7. Size and Performance

Size restrictions, not on the application or its components themselves, but on the data that is being processed by Mobile Messaging Application an impact on the architecture and configuration of the system. Particular decisions can be taken depending on the specific infrastructure chosen for deployment.

8. Quality

8.1: Security

Since this is a messaging application, security must be the number one concern because messages contain personal data and personal data must not be exposed to third-parties whom have not granted access by the owner of the personal data.

8.2: Maintainability

This application requires maintainability as there will be:
- bugs that need bug fixing which involves searching for errors and correcting them to allow the software to run seamlessly.
- capacity enhancement needs as enhancing the software kill provide new features required by customers.
- replacement needs as replacing unwanted functionalities will improve adaptiveness and efficiency.
- security issues as fixing security vulnerabilities found in your proprietary code or third-party code, especially open source components.

8.3: Responsiveness

Since messaging is real-time and the people that message expects messaging to be like a call, mobile messaging application should have the ability to meet its objectives for response time or throughput.

8.4: Learnability

In 2020, messaging is quite the essential need of a human, so the mobile messaging application will be used by almost everybody. This means that, it should be easily understandable. Learnability will allow users to quickly become familiar with the

product to make good use of all of the product's features and capabilities.

8.5: Scalability

High traffic is expected from the mobile messaging application, so the system should be able to handle a growing amount of work by adding resources to the system.