SMS Spamming App

Software Requirements Specification

Version 2.0

17-01-2019

Prepared By
Ibrahim Irfan
Bilal Ahmed

Revision History

Date	Description	Author	Comments
16-01-19	Version 1	Ibrahim Irfan	1st version of SRS
17-01-19	Version 2	Bilal Ahmed	2nd version of SRS

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date
	Dr. Syed Jawwad	HOD, Lecturer	17-01-19
	Mr. Usman Ahmed	Lecturer	17-01-19

Table of Contents

1. Introduction

This document presents a detailed description of the SMS Spamming Application (an Android based app) which is to be used by most Android smartphone users to protect their Android devices from any harmful spams. The SMS Spamming Application will be a mobile based app exclusively for devices build with the Android operating system. Spamming has become a major problem in today's world of smartphones, these spams usually infect a user's device and the makers of these spams can easily steal confidential information from a user's device without even the user knowing. This project will provide Android smartphone users with a ability to keep their important data that is stored on their Android device safe and secure.

1.1 Purpose

The purpose of Software Requirement Specification (SRS) is to specify the functionality, performance and interface requirements of this SMS Spamming App project. This document explains what the system will do and the constraints under which it will operate. This Software Requirement Specification document is intended for stakeholders and developers of the project. The purpose of system is to develop an Android based app exclusively for Android smartphones that will provide the users with the means to protect their Android devices from incoming harmful spams. These spams can be known or unknown. Known spams are those that come through promotional SMS ads and unknown can be part of a casual chat between two individuals.

1.2 Scope

The scope of this project is to develop an Android based application designed for exclusively for Android smartphones that will provide users with the means block or prevent any kind of spam from infecting their device. In comparison to previously developed applications with similar functionalities, this app will be reliable and free of cost. This app will also be more efficient and quicker than previous apps with similar functionalities. The app will be developed using Python and Java programming languages and will be able to run on any smartphone that is equipped with version 4.4 of Android OS or above.

1.3 Definitions, Acronyms, and Abbreviations

- **Android:** A Linux based open source operating system developed by Google Inc. for smartphones.
- **Android application:** An app developed using Python (occasionally) and Java exclusively for
- Android devices.
- **Short Message Service (SMS):** A type of text message component for mobile devices.
- **Spam:** Irrelevant or unsolicited messages sent over the Internet, typically to a large number of users, for the purposes of advertising, phishing, and spreading malware.

1.4 Tools

The application will run on Android Operating System and it is implemented on Python IDE which supports Python scripting language as well as Java IDE which supports Java classes. These development tools are preferred because they are the best application developing engines. They are portable and cross platform which means that the same code, developed via Python IDE and Java IDE, can be ported on many platforms with minimal modifications.

1.5. Overview

The next section, General Description, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next section. The third section, Specific Requirements, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product. Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

2. General Description

This section describes the functions, aims and objectives of the projects. It also includes the constraints, requirements, as well as assumptions and dependencies of the project.

2.1 Product Perspective

Spamming Application is an Android based application developed exclusively for smartphones built with the Android operating system. While the original application is built for Android, there will also be some Machine Learning involved for the purpose of successfully detecting spam messages. The primary scope of this project includes developing an Android app that will provide an Android user with the means to detect spams and block them from harming his or her device.

2.2 Product Functions

- User receives SMS message.
- App then detects SMS.
- It gives user the option to save the spam message or block it.
- App blocks the spam and send it to spam filter instantly.

2.3 User Characteristics

Android users – These users possess an android smartphone and constantly receive messages without knowing if it is a spam or not.

2.4 General Constraints

1. System will need only SMS Spam messages for functionality.
2. Application should be able to run on any version of Android 4.4 and above.
3. Application must detect SMS Spam messages.
4. System must performed accurate results.
5. System must be fully and well trained.
6. Dataset must be categorized according to classes being made.

7. Application must be able to detect any SMS Spam and then provide the user with the option to store it or block it.

2.5 Assumptions and Dependencies

We will assume that the data sets will be in CSV format. Dependencies include training data sets, that is before the system can begin processing and display any kind of output, the development team will need to provide input in the form of data sets that include known spams (ads) and unknown spams (casual conversations between two individuals) so that the system can perform data training.

3. Specific Requirements

This section contains the complete functionalities of the system, use cases, functional and non-functional requirements, external interface requirements, inverse interfaces as well as design constraints.

1) Functionalities

This sub-section contains the requirements for the SMS Spamming Application. It also includes what the system will do and what it won't do. This section will also describe the features and characteristics of the proposed system.

Table1: Searching for Spams

Introduction	
Input	The user clicks the search button to search for spams
Processing	1. First the user goes to first tab in the Main activity, and click Search for Spams 2. Once user reaches search page, user clicks the Search button 3. Once user has clicked the search, starts to detect Spams 4. Once the system has detect spams, the use case terminates.
Output	Spamming App displays SMS spam messages stored in phone
Error Handling	No SMS detected Display error message

Table2: Identifying Spams

Introduction	
Input	The user begins to identify spams available on the device
Processing	1. First, the user identifies SMS spam 2. Once the user has identified a spam, he or she chooses to either save SMS or block it 3. Once the user has blocked the spam, the system blocks and deletes the spam 4. Once the system has deleted the spam, the use case terminates.
Output	Spamming App starts detecting spams
Error Handling	No SMS detected Display error message

Table 3: Detection of Spams

Introduction	
Input	System starts detecting spams once search is complete
Processing	1. First the user chooses to delete spam 2. Once the user has chosen to delete the spam, App then sends spam to spam filter 3. Once the app has sent the spam to the spam filter, the use case terminates.
Output	Spamming App displays spams to the user
Error Handling	No SMS detected Display error message

Table 4: Giving user option to block spam

Introduction	
Input	The system gives user the option to block detected spams
Processing	1. First the user chooses to block detected spam 2. Once the user has chosen to block the detected spams, App then blocks the selected spams 3. Once the app has blocked the selected spams, the use case terminates.
Output	User receives message to block spams
Error Handling	No SMS detected Display error message

Table 5: Blocking Selected Spams

Introduction	
Input	User chooses to block selected spams
Processing	1. First the user chooses to block the detected spam 2. Once the user has blocked the selected spams, App then blocks the selected spams 3. Once the App has blocked the selected spams, the use case terminates.
Output	Selected spams are sent to the system
Error Handling	No SMS detected Display error message

Table 6: Deleting chosen spams

Introduction	
Input	User blocks selected spams
Processing	1. First the user chooses to block the selected spams 2. Once the user has chosen to block the selected spams, App then blocks the selected spams 3. Once the app has blocked the spams, the use case terminates.
Output	Spamming App deletes spams selected by the user
Error Handling	No SMS detected Display error message

3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface is the most important part of the system as it is used to interact with the users of the system. The first thing which should be considered while designing a user interface is that a GUI must be user friendly. The user interface for SMS Spamming Application shall be Xml, Python, and Java. A first-time user of the mobile application should see the main page when he/she opens the application. If the user is not a first-time user, he/she should be able to see the search page directly when the application is opened. Every user should have a profile page where they can edit their phone number.

3.1.2 Hardware Interfaces

Since the application is Android based and only runs on Android smartphones, all users must have a working Android smartphone with an active phone number. The Android version should be 4.4 or above. If a user does not have an Android version of 4.4 or above on his or her Android device, he or she will be unable to run the application. The app is not very heavy and will download and install instantly depending on the strength of the user’s Internet connection. Once the app is installed, the user will receive weekly notifications on future updates.

3.1.3 Software Interfaces

1) The SMS Spamming application will search for any SMS spam messages in phone’s database.
2) The SMS spamming application will detect spams and give the user with the option to either save the SMS or block it.

3) The SMS spamming application will then send the spam to the spam filter
4) The SMS spamming application will notify the user of incoming anonymous SMS messages.

3.1.4 Communications Interfaces

The SMS Spamming application shall communicate through user's current phone number. There is no need for any internet connection for the app to run on the Android smartphone.

3.2 Functional Requirements

3.2.1 Functional Requirement 1

Download mobile application: A user should be able to download the mobile application through either an application store or similar service on the mobile phone. The application should be free to download. The app will not be very heavy and should download instantly depending on the strength of the user's Internet connection. Once the app has been downloaded and installed, the user should have the ability to detect and block spams from his or device.

3.2.2 Functional Requirement 2

Download and notify users of new releases: When a new/updated version or release of the software is released, the user should check for these manually. The download of the new release should be done through the mobile phone in the same way as downloading the mobile application. The user will also receive notifications from the app and will be given the option to update the app to a newer version if he or she wants to get the latest features of the application.

3.2.3 Functional Requirement 3

User registration - Mobile application: Given that a user has downloaded the mobile application, then the user should be able to register through the mobile application. The user must provide name and phone number to complete his or her registration. Once the user has completed his or her registration, he or she will be directed to the login page where the user will have to enter his or her full name and current phone number to access app's features.

3.2.4 Functional Requirement 4

User log-in - Mobile application: Given that a user has registered, then the user should be able to log in to the mobile application. The log-in information will be stored on the phone and in the future the user should be logged in automatically. After the user has logged in, he or she will automatically be directed to the main page where they can search for any spams located on his or her device. The user will also be given the option to edit his or her profile when he or she sees fit.

3.2.5 Functional Requirement 5

Retrieve password: Given that a user has registered, then the user should be able to retrieve his/her password by email. He/she should also be able to search for any SMS spams on his or her Android smartphone. The user can also retrieve his or her account by providing his or her email or phone number. Once the user provides his or her email or phone number, he or she will receive verification code which he can enter to retrieve his or her account.

3.2.6 Functional Requirement 6

Mobile application – Search: Given that a user is logged in to the mobile application, then the first page that is shown should be the search page. The user should be able to search for a spam. Once the search begins, the system will start identifying and detecting spams which the user can later block and delete if he or she sees fit.

3.2.7 Functional Requirement 7

Spam Blocking: Once the user has detected any spams, he or she will be given the option to store it in the phone's database or to block and delete it. If the user chooses to block the spams that he or she has selected, the app will then instantly delete the spam and send it to the spam filter.

3.3 Use Cases

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. It can also summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. It demonstrates the relationship between all the actors involved in a Software Project. It contains 4 components:

- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.

- The use cases, which are the specific roles played by the actors within and around the system.
- The relationships between and among the actors and the use cases.

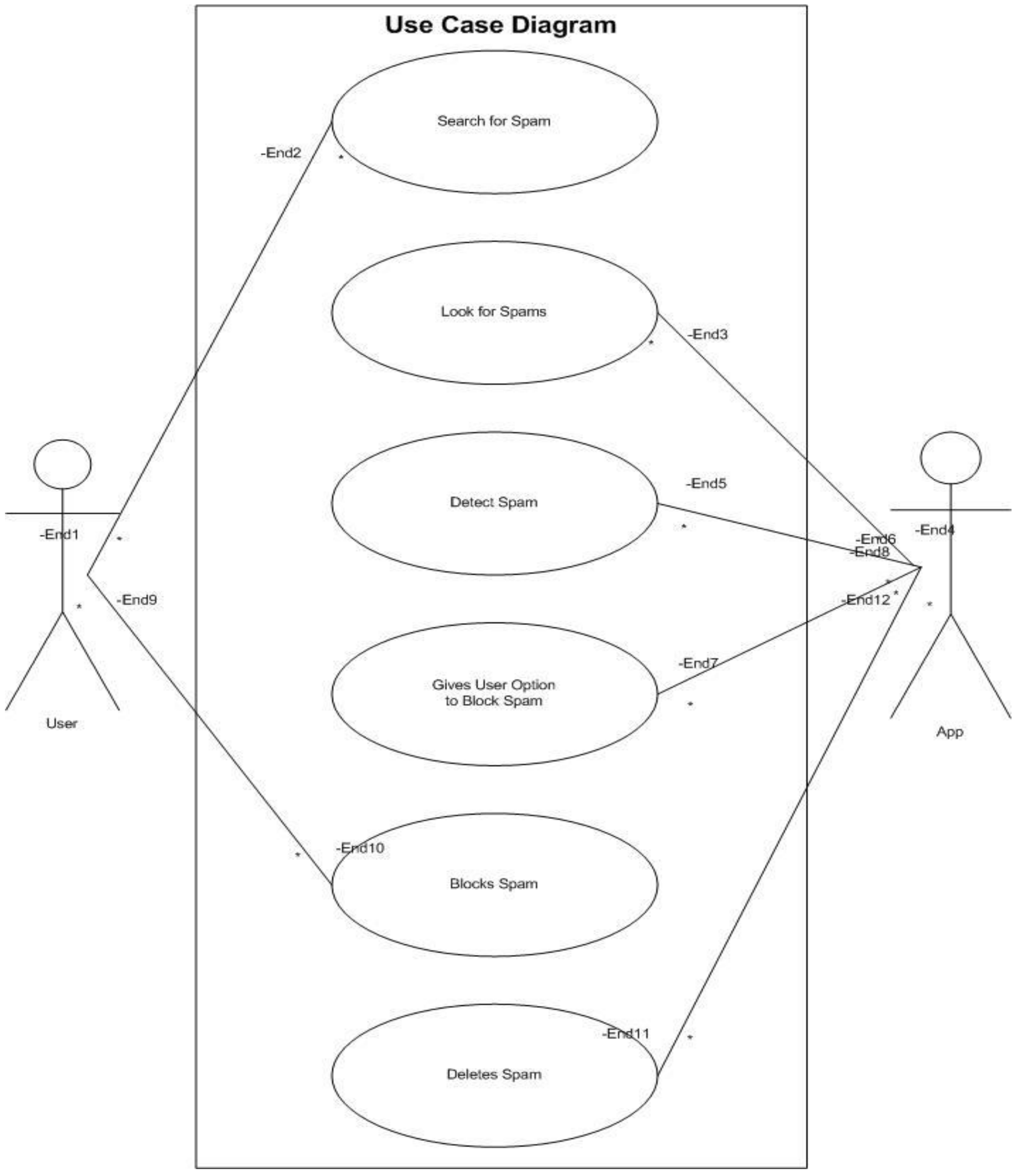

Figure 3.1: Use Case of App

In this diagram, there are two actors, one is the user and the other is the app. In the first use case, the user clicks the search button to search for spams. Once the user has clicked the search button, the app begins to look for any spams on the device. Once the search is complete, the app then starts to detect spams. After detecting spams, the app gives the user the option to delete the spams from his or her device. If the user chooses to block the selected spams, the app then deletes those spams from the device.

3.3.1 Use Case #1

Table 7: Use Case #1

ID	1
Description	To Search for an SMS Spam
Actors	User
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user goes to first tab in the Main activity, and clicks search for spams 2. Once the user reaches the search page, user clicks the Search Button 3. Once the user clicks the search button, the system starts to detect spams 4. Once the system finishes spam detection, the use case terminates.
Alternate Steps	None
Exceptions	User’s Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

3.3.2 Use Case #2

Table 8: Use Case #2

ID	2
Description	To identify and look for any Spams on the device
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user searches for spams by clicking the search button 2. Once the search button is clicked, the system starts to search for spams 3. Once the search is complete, the system starts detecting spams 4. Once detection is complete the use case terminates.
Alternate Steps	None
Exceptions	User’s Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

3.3.3 Use Case #3

Table 9: Use Case #3

ID	3
Description	Once search is complete, system detects spams on the device
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user searches for spams by clicking the search button 2. Once the search is complete, the system then starts detecting spams. 3. Once detection is complete, the system returns results in the form of the spams that were detected. 4. Use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

3.3.4 Use Case #4

Table 10: Use Case #4

ID	3
Description	Once a spam is detected, give user the option to block it
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user chooses to block the spam from his or her device 2. Once the user has chosen to block the selected spams, the system then blocks the spams from harming the device 3. Once the system has blocked the selected spams, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

3.3.5 Use Case #5

Table 11: Use Case #5

ID	3
Description	Once the system has detected a spam, user blocks it
Actors	User
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user chooses to block the spam from his or her device 2. Once the user has chosen to block the selected spams, the system then blocks the spams from harming the device 3. Once the system has blocked the selected spams, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

3.3.6 Use Case #6

Table 12: Use Case #6

ID	6
Description	Once user chooses to delete the spam, system blocks it
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user chooses to block spam from his or her device 2. Once the user has chosen to block the selected spam, the system then blocks the spam from harming the device 3. Once the system blocks the spam, it then sends the blocked spam to spam filter 4. Once the system deletes the spam and sends it to the spam filter, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

3.4 Non-Functional Requirements

The Non-functional requirement of the system includes following quality attributes that affect run-time behavior, system design, and Player experience, performance, security, portability, maintainability, reliability as well as availability.

3.4.1 Performance

All components of the system are performing well and correctly. The perceived response is immediate. The system does not waste valuable resources. It does not take initial load time more than 10 to 12 seconds

3.4.2 Reliability

The system can perform its intended functions and operations in a system's environment without experiencing any failure, except Operating System error. All errors are handled in a graceful manner. The software is a very reliable, one because there is no chance of system failure as there are only limited user which can access the system. The system is 90% reliable.

3.4.3 Availability

As our application is an open source application once, it will connect to the server then it will available for everyone at any time. Once the software is installed, it is always available unless and until the software would uninstall.
METER: Measurements obtained from 1000 hours of usage during testing.
MUST: More than 98% of the time.
PLAN: More than 99% of the time.
WISH: 100% of the time.

3.4.4 Security

The app should be able to keep any user's Android smartphone safe and secure from spams whether they are known spams or unknown spams.

3.4.5 Maintainability

The system can be updated and modified in case of any defects. It has ability to change the system components to meet new functionality. It can easily adapt new features and customization. All upgrades can be simply and safely performed. The application should be easy to extend. The code should be written in a way that it favors implementation of new functions

3.4.6 Portability

The system currently runs on Android Operating Systems. It is portable, as it has ability to reuse features and source code to any other Operating System as well. The application should be portable with Android.

3.5 Inverse Requirements

1. The app should detect Spams.
2. The system should only detect trained datasets.
3. The system should detect non-PDF formats.

3.6 Design Constraints

All of the classes and relations are designed to make the system as flexible and extendable as possible. The system design must be user friendly that’s why every person can easily understand and use the system.

4. Analysis Models

4.1 Sequence Diagrams

A Sequence Diagram (SD) is an interaction diagram that shows how object operate with one another and in what order. It is a construct of a message sequence chart. In this, we identify how the functions are carried out in the system. So this can help in making the different functionalities in the application. We carried out sequence diagram in each use case.

1. Search for Spams
User clicks the search button and starts searching for any spams on his or her Android device.

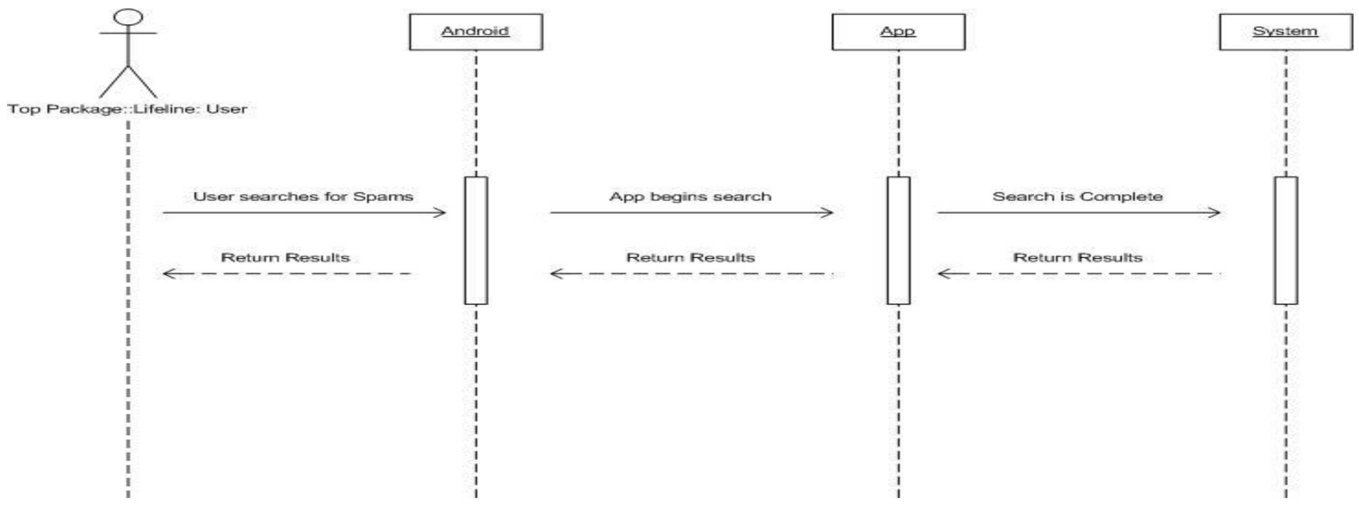

Figure 4.1: Searching for Spams

In this diagram there is an actor which is the user. The user clicks the search button to search for spams. The app then begins the search. Once the search is complete, the app will then return the results to the user in the form of detected spams on the Android device.

2. Identify and look for spams
User has the app look for any available spams on the Android device.

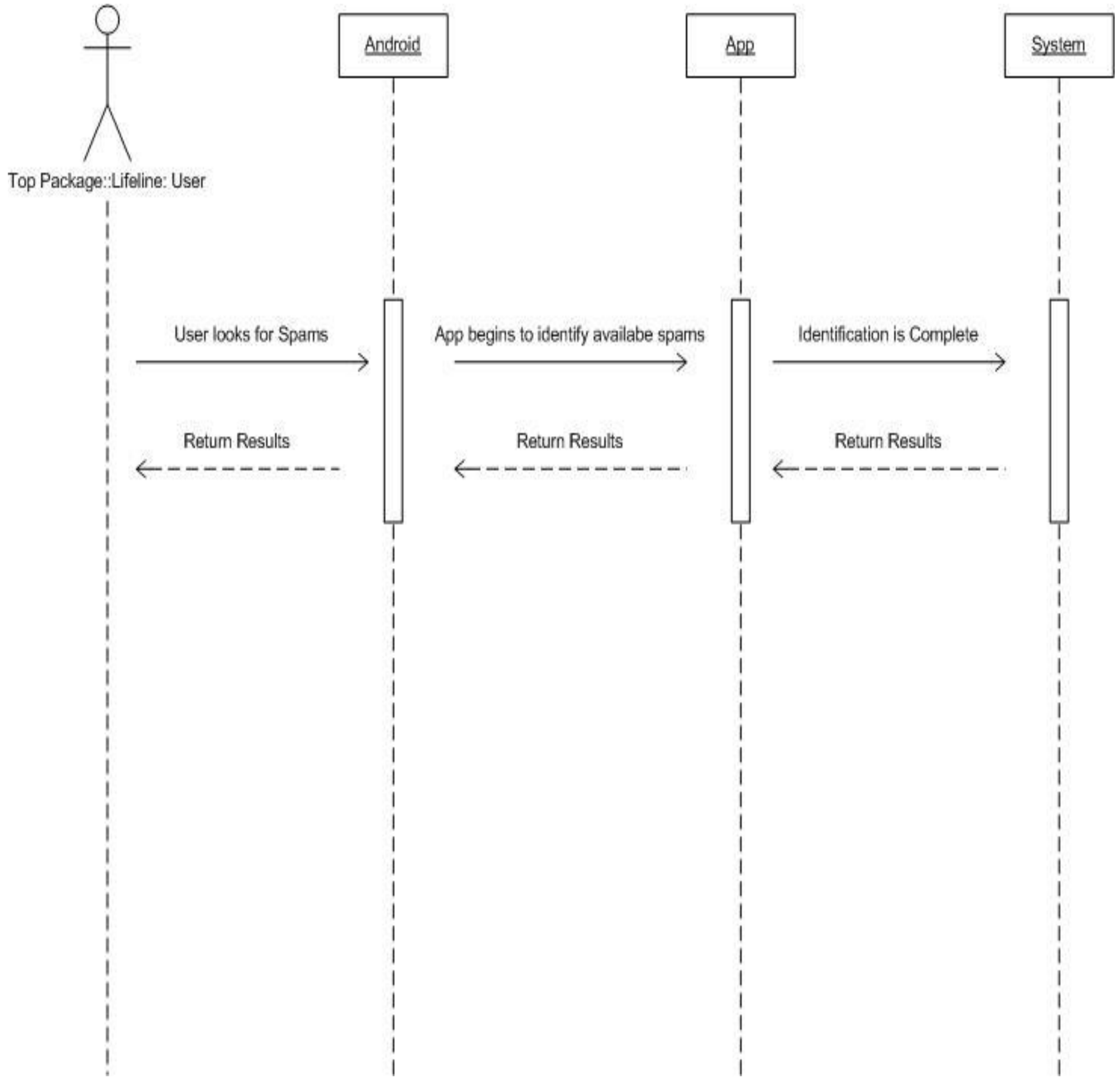
Top Package::Lifeline: User
Android
App
System
User looks for Spams
App begins to identify availabe spams
Identification is Complete
Return Results
Return Results
Return Results

Figure 4.2: Identification of Spams

In this diagram, there is an actor which is the user. Once the search is complete, the user has the app identify SMS. The app then starts to identify the spams. Once identification is complete, the app will then return the results in the form of detected spams on the Android device.

3. Detection of Spams on the device
User has the app detect spams on the device.

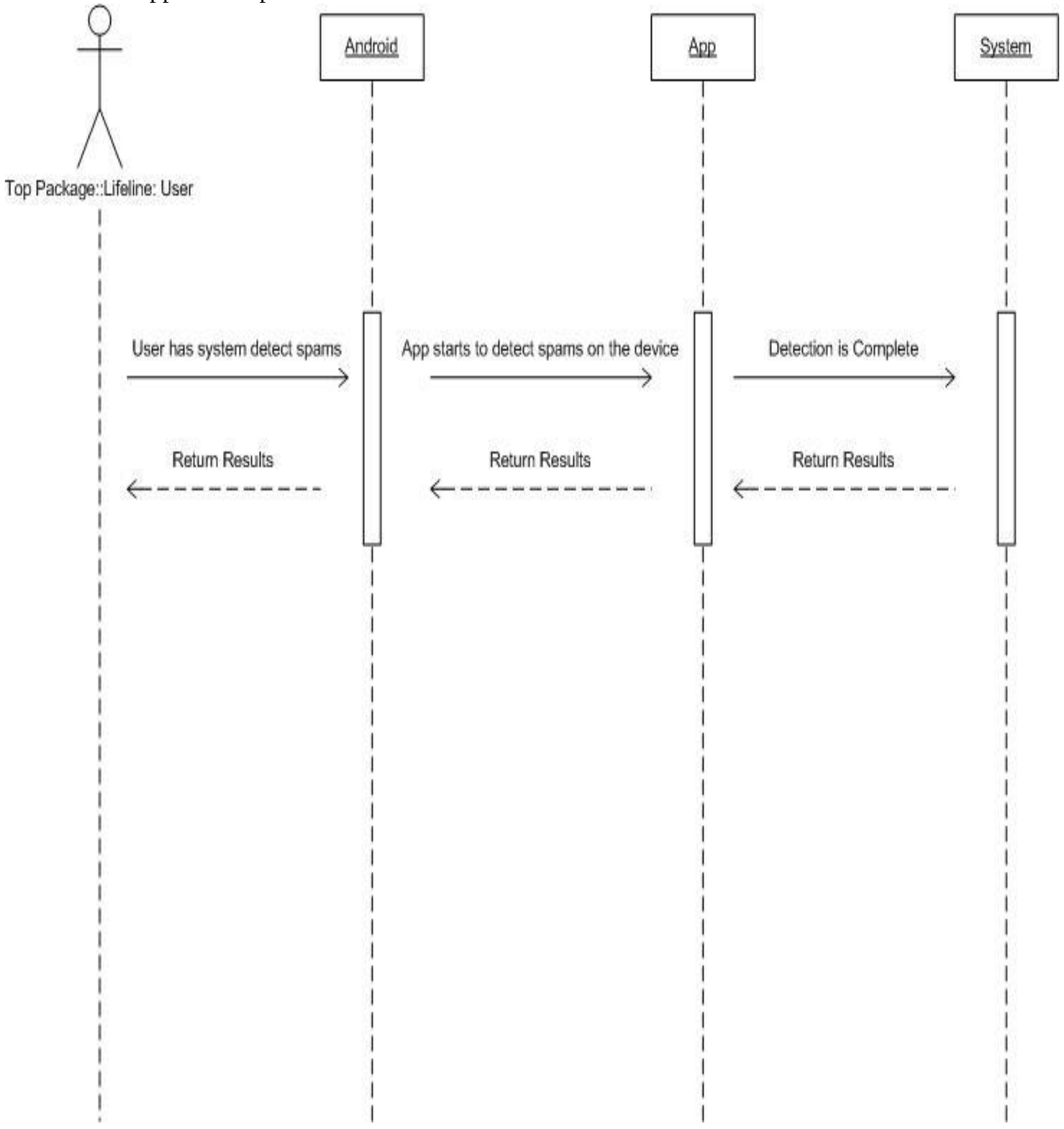
Android
App
System
Top Package::Lifeline: User
User has system detect spams
App starts to detect spams on the device
Detection is Complete
Return Results
Return Results
Return Results

Figure 4.3: Detection of Spams

In this diagram, there is an actor which is the user. Once identification is complete, the user will have the app detect spams located on the device. The app then begins spam detection. Once detection is complete, it will then return the results to the user.

4. Option for blocking spam is given
User is given the option to block or save the SMS spam by the app.

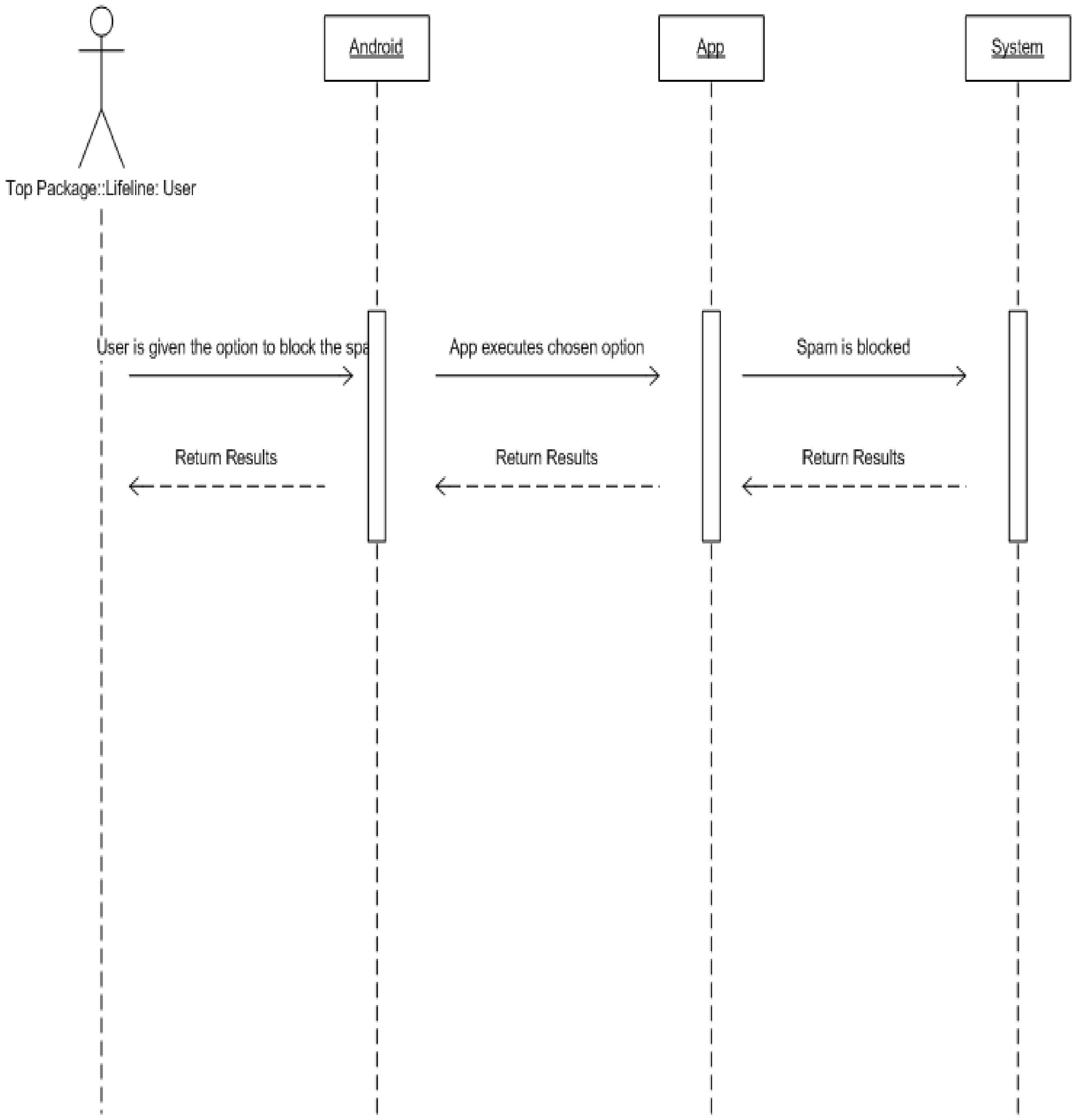
Top Package::Lifeline: User
Android
App
System
User is given the option to block the spa
App executes chosen option
Spam is blocked
Return Results
Return Results
Return Results

Figure 4.4: Option to block the spam is given

In this diagram there is an actor which is the user. Once the app has detected spams located on the Android device, it will give the user the option to delete the spam. If the user chooses to delete the spam, the app will then block the spams that are selected by the user.

5. User blocks the selected spams
Once user has chosen the selected spams, he or she blocks those spams.

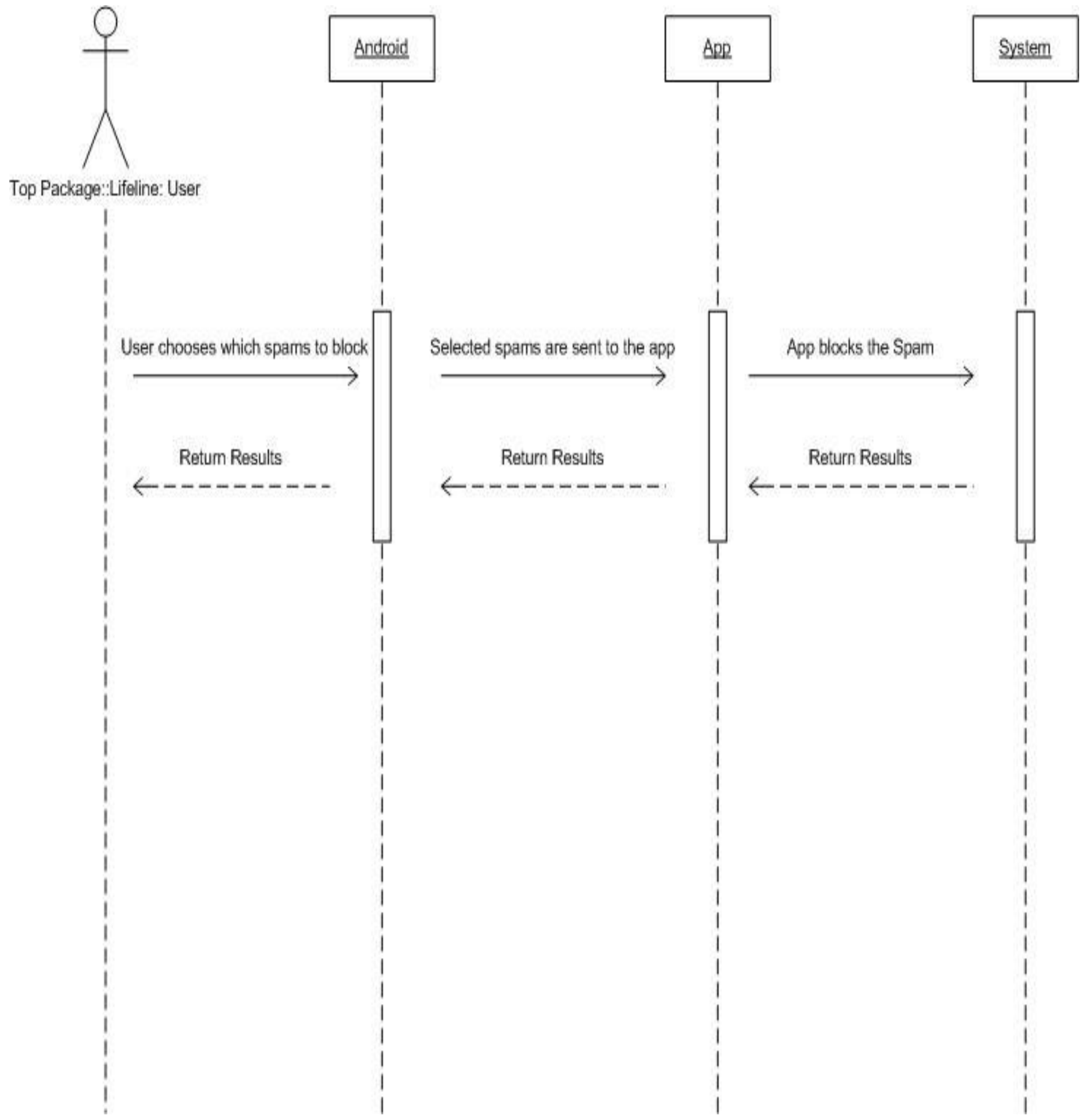
Top Package::Lifeline: User
Android
App
System
User chooses which spams to block
Selected spams are sent to the app
App blocks the Spam
Return Results
Return Results
Return Results

Figure 4.5: User blocks selected spams

In this diagram there is an actor which is the user. Once the user is given the option to store or delete a spam, he or she chooses to block it. The app then recognizes the selected spams and then deletes them from the device.

6. Deleting selected spams
Once the user has blocked selected spams, the App deletes those spams.

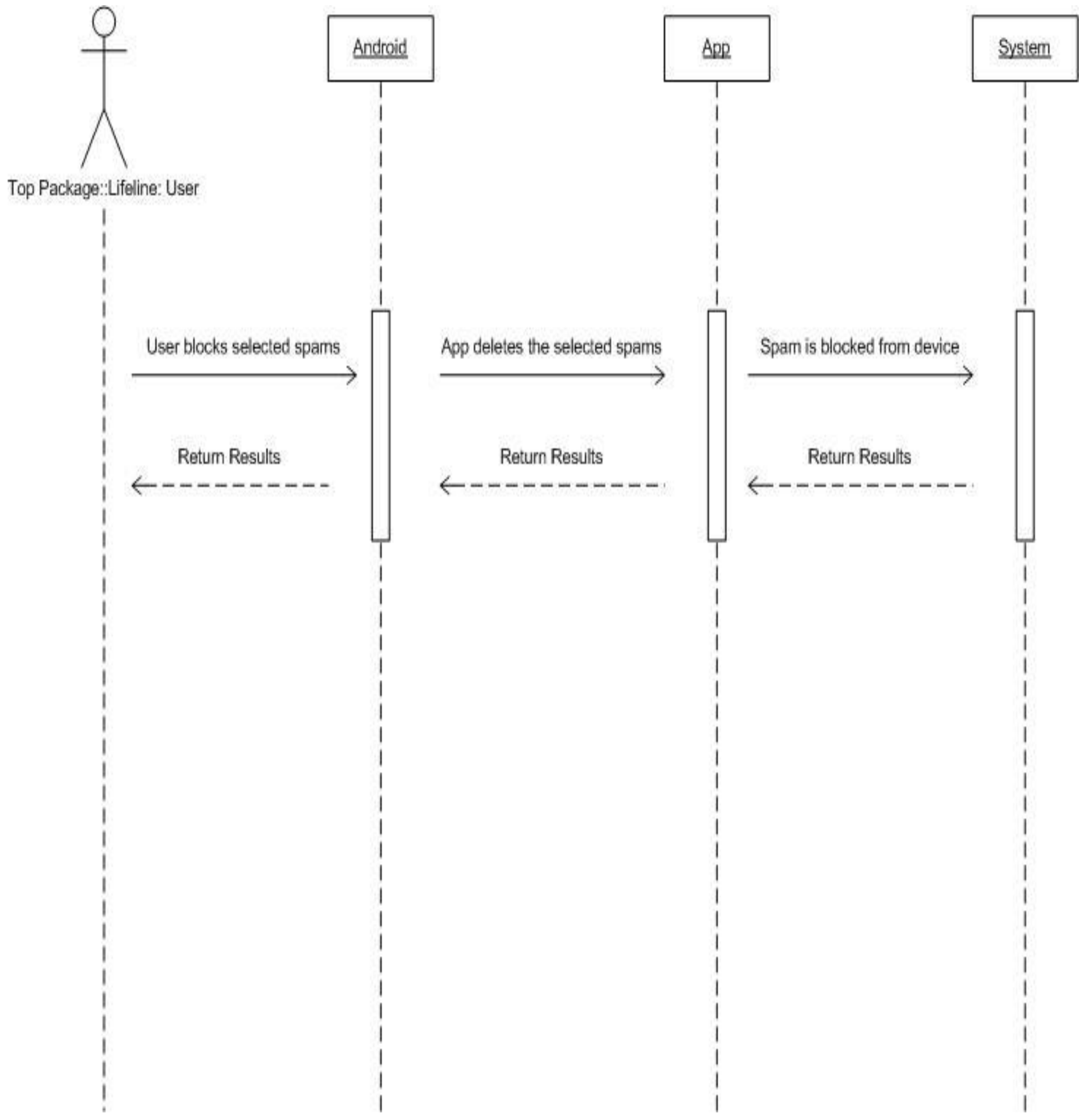
Android
App
System
Top Package::Lifeline: User
User blocks selected spams
App deletes the selected spams
Spam is blocked from device
Return Results
Return Results
Return Results

Figure 4.6: Deleting selected spams

In this diagram there is an actor which is the user. Once the user is given the option to store or delete a spam, he or she chooses to block it. The app then recognizes the selected spams and then deletes them from the device.

4.2 Component Diagram

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

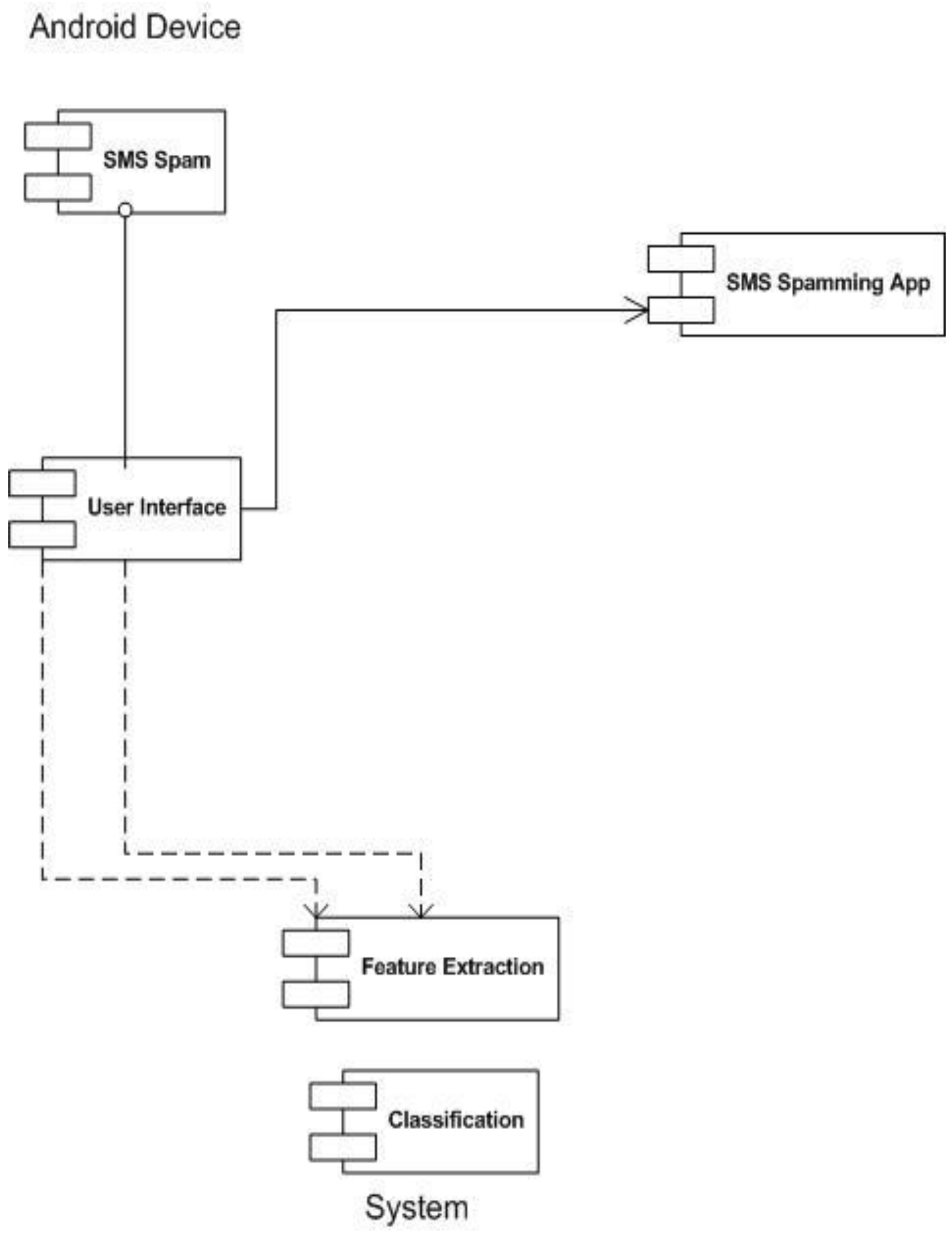

Figure 4.2: Composite Diagram of App

In this diagram there are 5 components, the first one is the SMS spam located on the device. The second is the user interface which is what the user will see on his or her smartphone. The third is the SMS Spamming App which is the software installed on the user’s device. The fourth and fifth are feature extraction and classification that are functions performed by the software.

4.3 Deployment Diagram

Deployment diagrams have several valuable applications.

- Show which software elements are deployed by which hardware elements.
- Illustrate the runtime processing for hardware.
- Provide a view of the hardware system's topology.

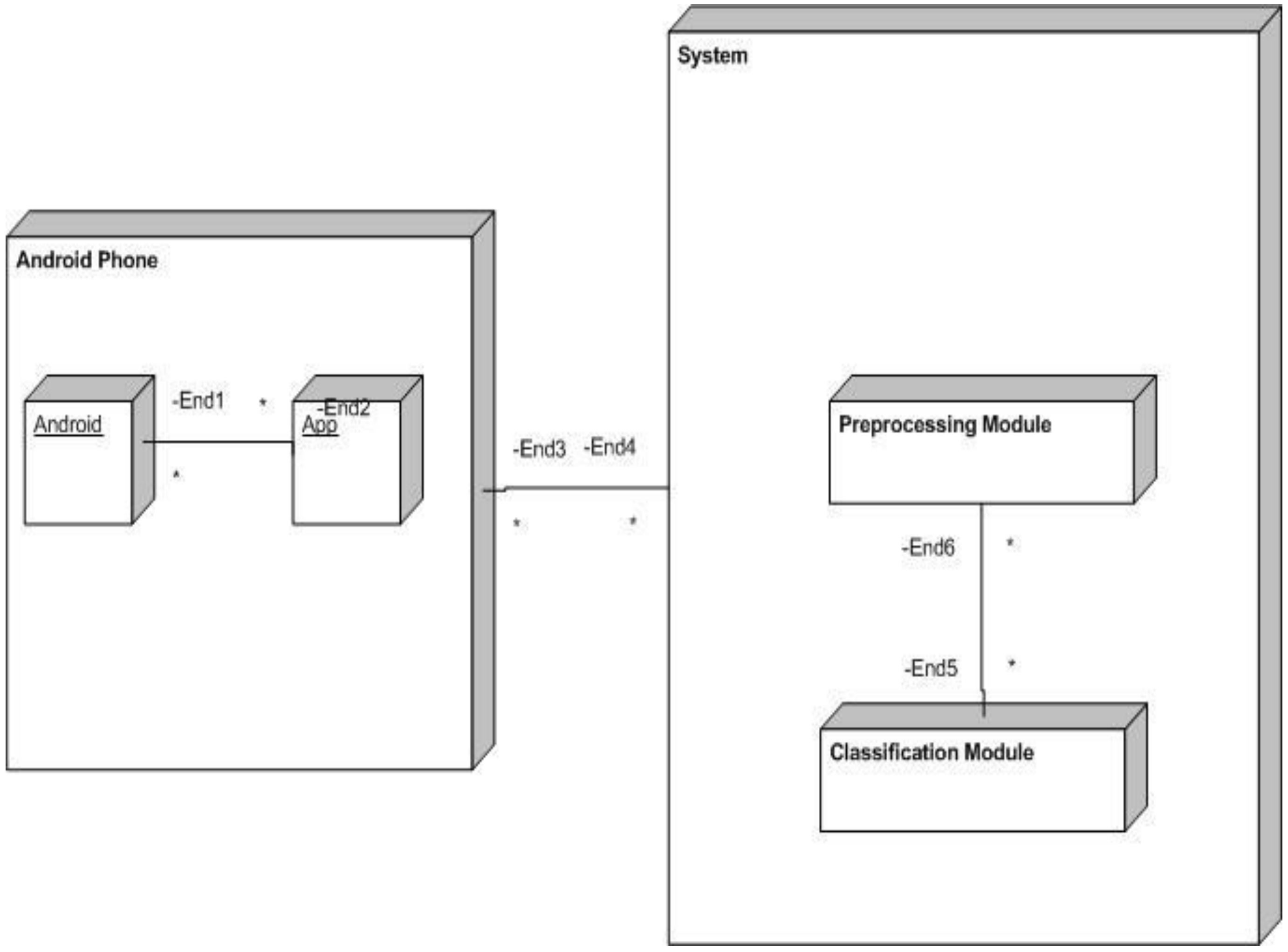

Figure 4.3: Deployment Diagram of App

In this diagram there are two main nodes which are the Android Phone and the system. In the Android phone, we have two sub-nodes called the Android operating system and the app while in the System, there are two modules called the pre-processing module and the classification module.

4.3Activity Diagram

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent.

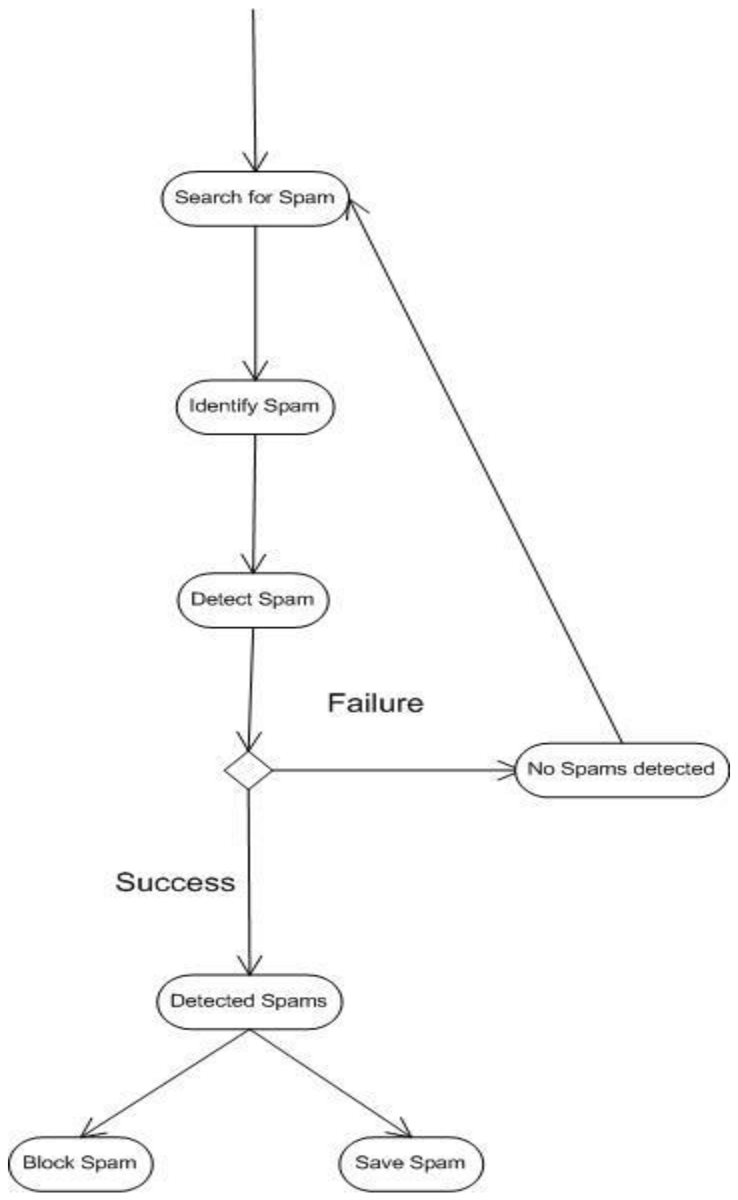

Figure 4.4: Activity Diagram

In this diagram we have seven action states. In the first one, the user clicks the search button to search for spams. Once the search begins, the system starts to identify spams. After identification, the system begins spam detection and when the detection process is finished, it returns the results. If there are no spams detected, the user will be redirected to the search page otherwise he or she will be see a list spams available on the device. The user will then have the option to store the spam or delete it.

4.5 Data Flow Diagram (DFD)

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

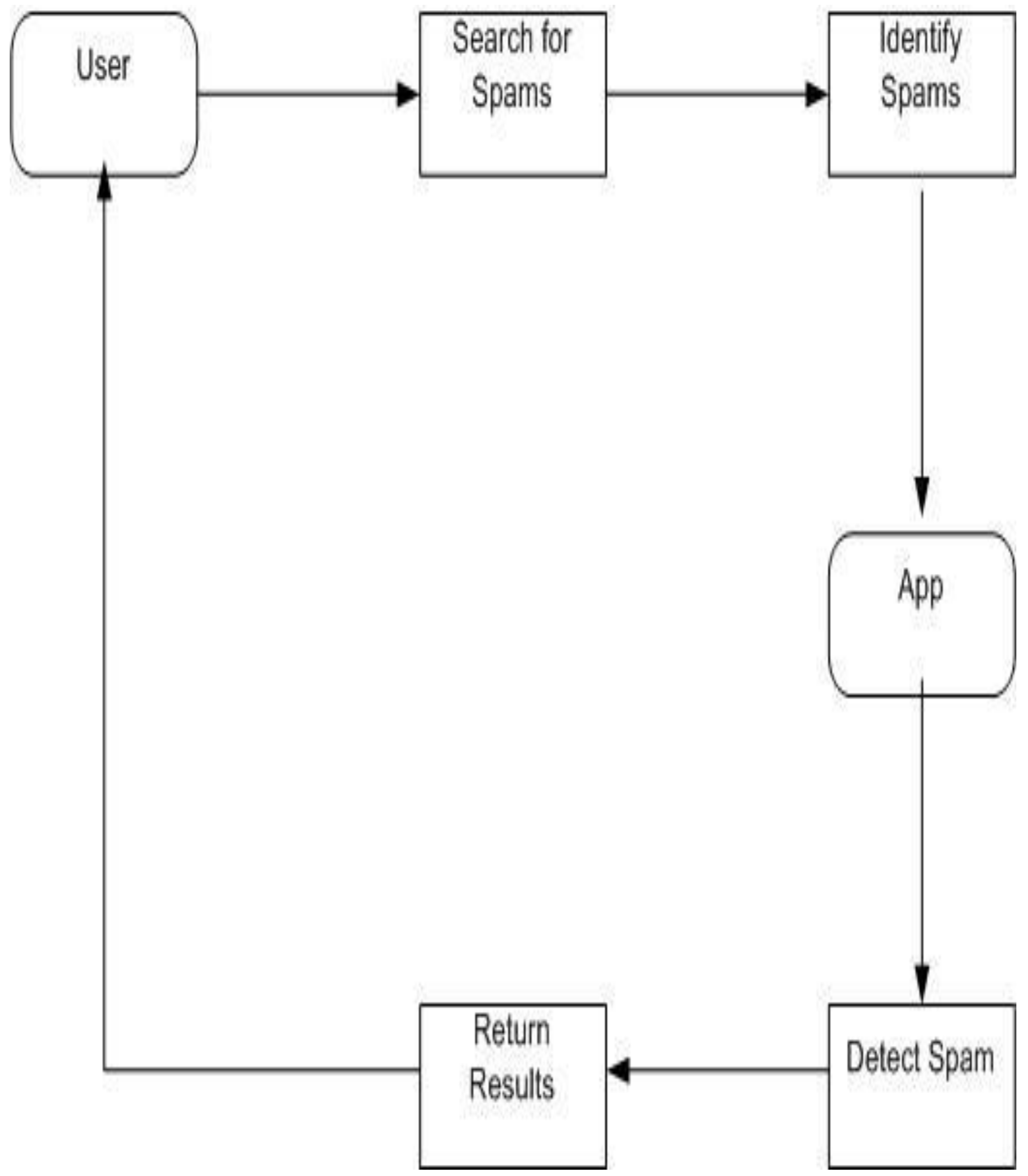

Figure 4.5: Data Flow Diagram

This diagram illustrates the flow of data for the system. The user clicks the search button to search for spams. The system then starts to identify spams. Once the identification process is finished, the system detects spams available on the device. After spam detection is complete, the system returns the results to the user.

4.6 State Transition Diagram

State machine diagram is a behaviour diagram which shows discrete behaviour of a part of designed system through finite state transitions.

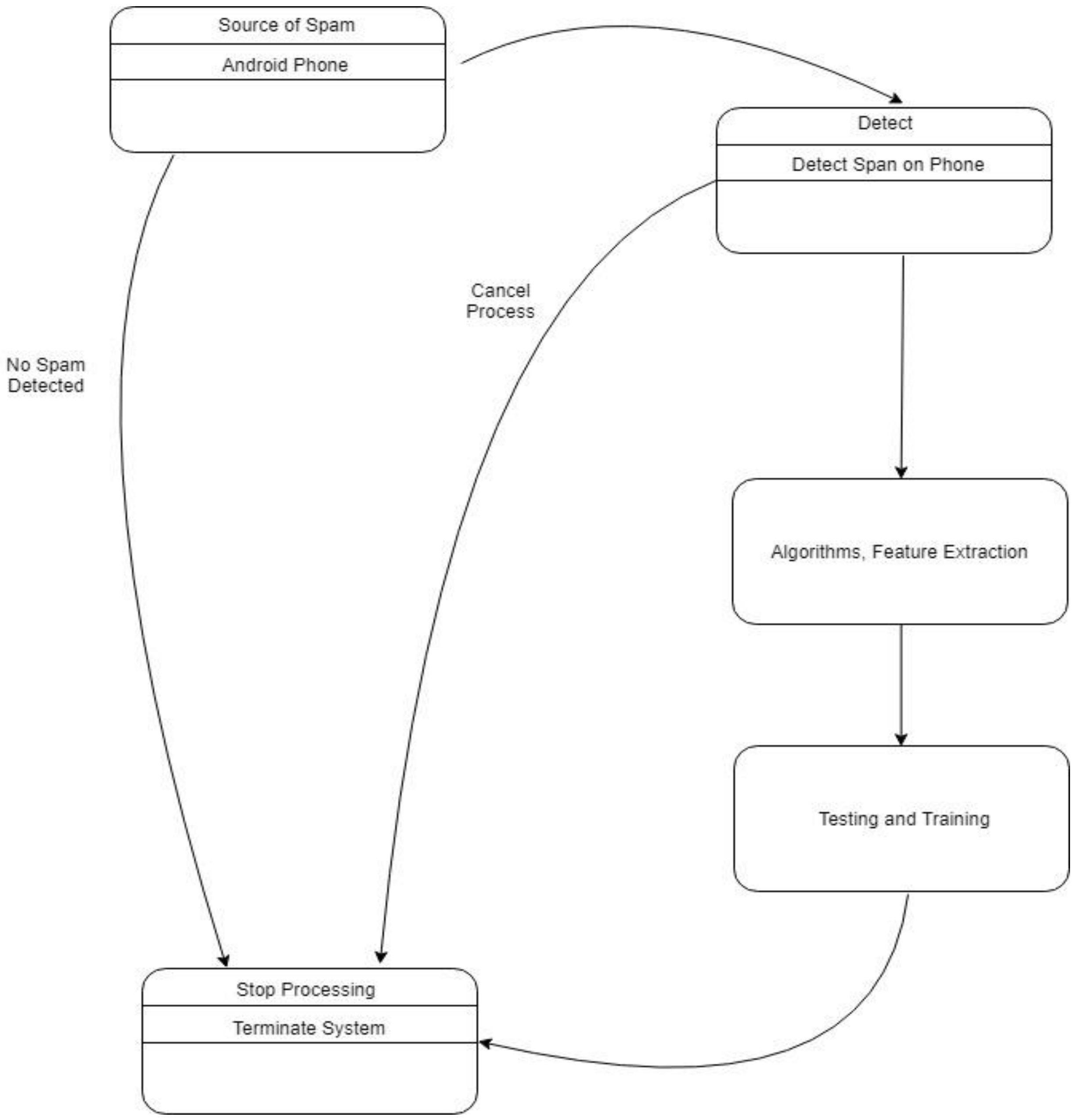

Figure 4.6: State Transition Diagram

This diagram illustrates the states that the system goes through. The first is the source of the spam which is the Android phone. The next one illustrates the detection process which will detect spams located on the device. The third one shows the feature extraction and algorithms that will be used. The fourth one shows the training and testing of data sets. And the last state illustrates that the process must be stopped or the system should be terminated if there is no spam detected or if the user manually stops the process.

References

[1]. https://www.slideshare.net/sinhaprita/final-srs-of-academic-a-webpage-based-android-app

[2]. https://clearbridgemobile.com/how-to-build-a-mobile-app-requirements-document/

[3]. https://www.citysdk.eu/wp content/uploads/2013/09/DELIVERABLE_WP4_TA_SRS_0.21.pdf

Appendices

Appendix 1:

Android studio	Android studio is official integrated development environment (IDE) for Google's android operating system, built on JetBrain's IntelliJ IDEA software and designed specifically for android development. It is available for download on windows. It is the replacement for the Eclipse android development tools (ADT) as primary IDE for native android application development
GUI	Graphic User Interface
Android App	An Android based application developed using Java and Python exclusively for Android Devices
Windows	Operating system
Scripting	A scripting language or script language is a programming language that supports the writing of scripts, programs written for a special runtime environment that can interpret and automate the execution of tasks which could alternatively be executed one-by-one by a human operator.
Graphics	Graphics are visual presentations of app
SRS	SRS Software Requirement Specification

UX	User experience for using a product.
User	User is the one who use the app to read listen and watch the stories of prophets
SMS	A type of text message component for mobile devices
Spam	Irrelevant or unsolicited messages sent over the Internet, typically to a large number of users, for the purposes of advertising, phishing, and spreading malware.
System	A system is a set of interacting or interdependent components forming an integrated whole or a set of elements and relationships which are different from Relationships of the set.