Master's Degree in Informatics Engineering

Final Report

Integration of Social Network and Cloud Storage services in an Android application

Department of Informatics Engineering
Faculty of Sciences and Technology

University of Coimbra

Paulo Jorge Pires dos Santos
pjpires@student.dei.uc.pt

Supervisors
Frederico Lopes
Jorge Granjal

June 25, 2014

Abstract

This internship is inserted in the Rich Communication Services (RCS) initiative by GSMA – the Groupe Speciale Mobile Association. The RCS initiative is an ambitious project for the mobile operators to fight back the Over the Top (OTT) applications, currently proliferating in the telecommunications market. WIT Software develops an RCS product for numerous mobile operators, and the objective of this internship is to connect its product with social networks and cloud storage services.

The outcome of this internship introduces the integration of social networks and cloud storage services in the Android version of WIT's RCS Apps. Sharing and saving files from/to cloud storage services, enriching the application contacts with pictures and information from social networks and also display their latest status updates are some of the features under this internship's scope. Alerting the users of their friends' birthdays and to allow the backup and restore of the application conversations are other features to be implemented.

Additionally, during the internship two other features were embraced: sharing current location in real time with other users in two different modes – walking and driving – and the ability to share and view ephemeral multimedia messages, which vanish without leaving a trail after displayed.

Keywords: Social Networks, Cloud Storage, Contact Enrichment, Active Synchronization, Android, Ephemeral Messaging, Real-Time Location, Rich Communication Services

Acknowledgments

First, I would would like to express my gratitude to both supervisors, Frederico Lopes and Jorge Granjal for their time, patience and support along the whole internship. I also would like to thank the whole Android team of the RCS product in WIT Software for their invaluable help whenever I needed.

Next, a big thanks to all my friends and colleagues on a five-year adventure, which is now reaching its end. It was an incomparable experience, full of sleepless nights but also full of team spirit and joy everywhere, which always got us the desired results, in the end.

Last, I would like to thank all my family, to my parents in particular, which always stood there for me, supporting and guiding throughout the challenges. A very special thanks to them, for their continuous effort in giving me the best they can.

Contents

List of Tables

List of Figures

Acronyms

API	Application Programming Interface
CEO	Chief Executive Officer
CTO	Chief Technical Officer
DEI	Department of Informatics Engineering
DoD	Definition of Done
GPS	Global Positioning System
GSMA	Groupe Speciale Mobile Association
IMS	IP Multimedia Subsystem
JNI	Java Native Interface
MIME Type	Multipurpose Internet Mail Extension Type
MMS	Multimedia Messaging Service
MNO	Mobile Network Operator
OS	Operating System
OTT	Over The Top
PoC	Proof of Concept
RCS	Rich Communication Services
RCS-e	Rich Communication Services – enhanced
SMS	Short Message Service
UI	User Interface
UX	User Experience
VoIP	Voice over IP

Glossary

API An Application Programming Interface is a group of methods or access points that a software provides so that external parties can make use of their functionalities with no need to know the intrinsic processes

Burndown Chart Burndown Chart is a graphical representation of a Team's outcome at a given point of a Sprint. It plots the evolution of the remaining points – Sprint's User Stories still opened – corresponding to the Tasks still opened inside each User Stories

Contact Enrichment Contact Enrichment, in the scope of this project, consists in gathering information about the contacts in the user's mobile phone from social networks, and use that content to update the contact profiles in the user's smartphone

DoD The Definition of Done is a document that states the requirements that must be met before a Backlog User Story is marked as done

GSMA The GSM Association (GSMA) is a group of mobile operators and companies related to the mobile market, created in 1995, to define standards for mobile communication systems

Impediment In Scrum, an impediment is something required for the Team Members to finish a Task that is outside of their scope in the project. Impediments are assigned to the Scrum Master so that he can solve them as soon as possible

JNI Java Native Interface is a framework which allows Java code running in a Java virtual machine to make calls or be called from native code (such as C or C++)

joyn joyn is the brand created by GSMA to identify the applications that comply with their specifications to deliver Rich Communication Services to the end user

MNO A Mobile Network Operator is both a provider and manager of the technologies and infrastructures required to supply mobile communication services for their users, such as messaging or telephony

Mockup Regarding this project, a Mockup is both a graphical illustration of a feature's use cases and a proposal for the User Interface (UI)/User Experience (UX) of that feature in the current product

OTT Application Over The Top applications refers to applications that deliver multimedia content over the Internet, like audio and video calls between two end users

PJSIP PJSIP is an open source library which provides multimedia communications, implemented under the standards defined by a set of communication protocols designed for multimedia networks, such as SIP and RTP

Planning Poker Planning Poker is an activity performed by the Team Members in order to estimate the effort for each User Story in the Product Backlog, based on their Velocity

PoC A Proof of Concept is a demonstration of a certain feature's feasibility inside a product, which is obtained through its successful implementation

Product Backlog Product Backlog is an artifact that contains all the ideas for the project, in the form of User Stories. These stories are organized by the priority given by the Product Owner, and are also assigned an effort that is the estimate given by the Team Members, as a result of the Planning Poker activity

Product Owner In Scrum the Product Owner is the person responsible for defining the User Stories that compose the Product Backlog and assign them a priority in order to conduct the Scrum Team's work. The priority of those User Stories can be changed by the Product Owner in order to maximize the value of the work by the delivery date

RCS Rich Communication Services is a global initiative by GSM Association (GSMA) to compete the Over the Top applications that are penetrating the mobile communications market at a fast pace

Redmine Redmine is a web platform for project management that includes various tools to check the project state, such as charts, task boards, etc. This tool was used in the project because it is widely used inside WIT Software and has support for the Scrum framework

Scrum Scrum is an iterative and incremental agile development process/framework. This process is task priority-oriented, meaning that it seeks to fulfill tasks with higher priority first

Scrum Master Scrum Master is the responsible to help both Product Owner and Team Members. The Scrum Master helps the Product Owner to manage the Product Backlog, and helps the Development Team to mark all the User Stories in a Sprint as done, according to the Team's Definition of Done (DoD), by removing impediments and keeping them focused and productive

Sprint A development Sprint is a fixed implementation period, where the Team Members focus their effort in developing the features that match the User Stories they compromised to. Sprint duration is usually between one week to one month; in this project, two weeks Sprints took place

Sprint Backlog Sprint Backlog is composed by the User Stories that are selected by the Team Members to be accomplished along a Sprint. Each of these User Stories is divided into smaller Tasks that must detail the steps the Team must take in order to fulfill a story

Sprint Meeting Sprint Meeting comprises two Scrum activities at once in our project: Sprint planning and Sprint reviewing. First consists of planning the next Sprint, deciding which User Stories will be performed. Its counterpart is Sprint reviewing, which stands for a brief review on the previous Sprint outcome

System Test System Tests are made after the integration of the developed features in the main product, and are run in order to check for inconsistency on the system's behavior after this integration

Task A Task is a smaller assignment, subset of a User Story, which states something that must be implemented as part of that functionality. The tasks for each User Story may be defined when that story is assigned to a development Sprint, defining specifically what needs to be done to mark the story as closed

Team Member The Team Members are responsible to create the incremental changes on the product that match the User Stories assigned to each Sprint. Team Members forecast the User Stories that they can deliver at the end of the Sprint, and also decide how they will be implemented

Unit Test Unit Testing consists of testing an application's methods or functions individually, by using an input test and checking the output result to see whether it matches the expected outcome or not

User Story	A User Story defines a possible user action with the product in order to state a project requirement in the everyday language. A User Story can comprise multiple Tasks that define meticulously what must be created by the development Team Members in order to mark the story as done
Velocity	The Velocity of a Development Team is given by the sum of the effort points of the finished User Stories inside a Sprint. Velocity is used at a new Sprint's beginning, when of the selection of the User Stories that the Team will compromise to deliver

1 Introduction

This introductory chapter provides an overview on what are the contents of this project and, particularly, this report. The **first section** explains the motivation behind the work hereby proposed. **Second section** introduces the context of this project and how it took place in WIT Software. The **third section** exposes the objectives of this internship. **Last section** details the remaining document's structure.

1.1 Motivation

We are now entering the mobile devices era, and the communication solutions diversified a lot. In the last years, many applications providing communications over the Internet appeared, mainly because of their minimal production requirements and deployment costs. Communicating is part of life, so these applications are always very likely to succeed if they are able to catch the users' interest. In reality, people started to engage in this applications because of their simplicity, and the fact that most of them are cost-free. Commonly called Over The Top (OTT) applications, they revealed to be a real concern for Mobile Network Operators (MNOs), by proliferating in the telecommunications market and taking away a good share of the operators revenue worldwide.

Most Over the Top (OTT) applications are based on a free/*freemium*[1] business model. They take advantage of the latest years improvements on cellular networks, and make use of an IP-based environment to deliver rich multimedia content across a variety of devices at lower or no costs to the end-user. They provide a whole range of services such as voice, messaging and even games, which are far more attractive than what the common MNO offers. The impact of these applications in the telecommunications area started to be noticed and are predicted as potentially dangerous for the MNOs revenue [1]. **Figure 1.1** illustrates the Short Message Service (SMS) revenue forecast from 2010 to 2017 in North America alone [2], which shows a decay starting from 2011. This is the particular case of the North American market where, in opposition to the general situation worldwide, SMS is already entering the declining phase, with both revenue and traffic getting lower year after year. Still, it is an indicator of what is yet to come, at a global scale. Another report introduces the remarkable growth of the OTT messaging traffic, that is supposed to surpass SMS traffic in 2013 [3].

[1] Free applications often use advertisement inside the application as their profit source. On the other hand, the income for an application based on a freemium business model relies on paid advanced options or functionalities, while the application by itself is also free. See `http://mashable.com/category/freemium/` for more information about the *fremium* business model.

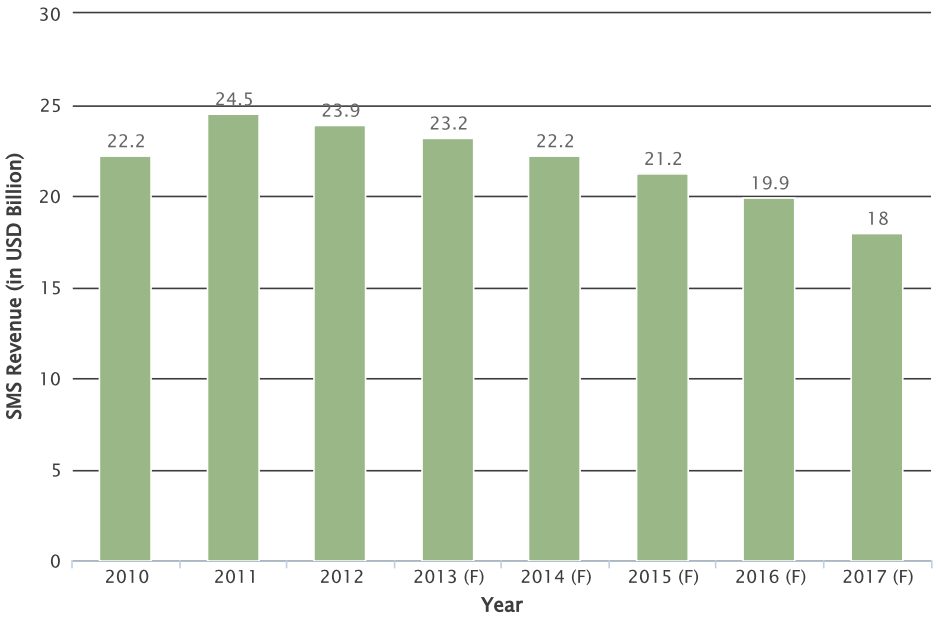

Figure 1.1: SMS Revenue in North America in USD Billion, 2010-2017[2] [2].

Considering these applications a serious threat, GSMA – a group that comprises the major mobile operators and brands from the telecommunications industry – decided to face the problem by creating a service able to provide similar features to those that one may find in OTT applications. The Rich Communication Services (RCS) initiative was created to provide a variety of "rich content" in addition to the traditional messaging and voice services, like files or media sharing during a call, or sharing the current location inside a conversation. The adoption of such features by the mobile operators marks the transition from the circuit-switched communications to an IP-based communication environment [4], while mobile devices continuously get support for better and faster network connections. The key selling point of this idea over the OTT applications is the interoperability between different Mobile Network Operator (MNO)s, which means that assuming all operators would deploy such a solution, these features would be available regardless of one's carrier, such as the simple SMS and call services. Additionally, the security and privacy of the user's data is another advantage of the MNOs services, because they control the network infrastructures and operate in the telecommunications field for several years now, which allows them to provide strong solutions regarding the security and privacy of their networks.

The RCS initiative is specified by GSMA in a document that has iterated over five main releases until now, adding new features and defining the technical details to deliver rich multimedia content to the end user. The current version of this document is the RCS 5.1 [5].

This initiative is promoted by GSMA under the brand "joyn", as an engaging new experience

[2]F stands for Forecast.

of communicating with each other. To deploy a joyn-branded product it needs to comply with the some requirements at the User Interface (UI)/User Experience (UX), so that it can get an approval from GSMA. These UI/UX guidelines are present in the latest version of joyn product definition – joyn Blackbird [6] – which is based on the RCS 5.1.

Finally, when a MNO decides to introduce a joyn-branded application, they will surely look for an application that, beyond compliant with joyn specifications, also has unique features that bring in some extra value over the applications provided by their competitors (in this particular case, other MNOs and OTT applications). Therefore, one can assume when an application presents itself with a good User Interface and User Experience, a robust implementation and adds extra features over the competition is has a great potential to get selected over the others.

Social Networks and Cloud Storage Services

The particular need to introduce social networks is revealed by the astonishing numbers collected during the State of the Art, along the first semester of this internship. A significant number of people are using these services on a daily basis, which means that a successful integration on the product could bring in some attention from the end users. Additionally, there are some OTT applications successfully making use of the social wave that the world entered, thus testing the end users for adherence to such functionalities is a must. The proposed features will allow the application to use information from the user's social network accounts to enhance his experience inside the application: filling missing details in contacts, like their birthday or display photo, or even see the latest picture that a friend uploaded on his profile will be useful tools for the user when inside the joyn application.

On the other hand, the cloud storage services are widely used, and they are still a growing market. People tend to forget and lose their digital documents very often, and cloud storage services revealed very useful since they represent a place that is expected to be secure and accessible anywhere, anytime. The integration of the application with such services will prove to be very handy, allowing its users to share and save contents from the joyn application directly to their cloud accounts, or even backup their conversations from an old device and restore them in a new one.

Additionally, the integration with social network and cloud storage services are still an innovation when comparing to the remaining joyn solutions, as none of WIT Software's competitors provides such features. Therefore, they are also an advantage for the company when it comes to their direct competitors.

Ephemeral Messaging and Real-Time Location

During the State of the Art analysis, two other mobile application experiences were found very interesting: ephemeral messaging and location share in real-time. Some applications like Snapchat and Confide brought a new way of communicating through volatile, short-term, multimedia and text messages, which easily conquered the interest of the users. On the other hand, applications like Glympse and the Apple's proprietary Find My Friends system allow the users to share and view the location of multiple people at a time, providing another way to stay in touch.

These two features were further analyzed during the State of the Art, detailed in **Chapter 2** of this document. As a result of this analysis, both the trainee and WIT Software considered the unique experience provided by these applications to be valuable in a messaging application as the one were the internship was integrated. Later along the road both features were implemented by the trainee in the Android version of WIT's RCS product.

1.2 Context

This internship took place in WIT Software headquarters, in Coimbra. The company focuses in developing communication solutions, from mobile and TV applications to network infrastructures software. The Android application in which the project's final solution was integrated is in charge of the WIT Software's RCS team, where the trainee worked.

The whole project was created to be part of the current joyn solution for Android. This application is continuously improving, in order to match the requirements of the joyn specifications that GSMA releases periodically, either to introduce new features or to change the existing ones.

During the project the trainee was assigned a supervisor from the Department of Informatics Engineering (DEI), Professor Jorge Granjal, and another from WIT Software, Engineer Frederico Lopes. Both of them contributed a lot to this work by providing guidance and clarifying doubts that occurred along the way. They were also responsible for revising the contents of this report, providing feedback and helping whenever it was needed.

Two other trainees from DEI worked in same RCS team, very close to the trainee hereby reporting his internship. The work of the three trainees is under the supervision of the same person in WIT Software, as mentioned before, Engineer Frederico. All three were grouped in a small team, as described ahead in **Section 3.1**.

1.3 Objectives

The objectives of this internship may be analyzed under two different scopes. The first comprises the company's goals – the product objectives – while the second is related to the trainee internship goals – personal objectives. Both of them are detailed in the subsections below.

1.3.1 Product Objectives

A successful integration with social network and cloud storage services brings added value to the current joyn solution developed by WIT Software. At the moment, none of the features to be developed along the internship are part of the RCS specifications by GSMA, so they would probably be distinctive when comparing to other joyn applications. This is very useful for the company when it comes to show the product, once it will give them an advantage over their competitors.

1.3.2 Personal Objectives

When it comes to the trainee objectives, the experience to be acquired by working on a big development team and using cutting edge technologies is both a challenge and a privilege, which turns the expectations high. The main goals for this internship are to accomplish all the tasks in the project efficiently, to get knowledge in the mobile development field and to be part of a real company, while applying the concepts learned along the last five years in college.

1.4 Document Structure

The structure of the remaining document is the following. **Chapter two** presents the State of the Art in terms of competitors to the joyn product developed by WIT Software, and covers the major cloud storage and social network services. The **third chapter** introduces the approach to the implementation of the desired functionalities on the project – the methodology and the planning for both semesters, compared to the real outcome. **Fourth chapter** covers the identified requirements for the project. The **fifth chapter** presents the architecture designed and implemented to create all the features. **Chapter six** analyzes the implementation stage – specifically, the problems and solutions found along the way – and the validation of all the work done along the internship. **Last chapter** concludes this report by reviewing the overall implementation and what could be improved, and accounts the future work as well.

2 State of the Art

This chapter overviews the major communication applications in the mobile market. It also supplies information on social network and cloud storage services, since they are the base of this internship.

First section presents information on competitor applications to joyn – the so-called OTT applications –and the direct competitors to WIT Software's RCS product. This competitors analysis was an extensive study, regarding 25 communication applications. In the **Appendix A** this analysis is presented with all the collected data. That information is summarized in the following pages[1].

Sections 2.2 and **2.3** provide an analysis on the major cloud storage and social network services. This analysis presents an explanation on those services and why the integration with them is important nowadays. At the end, a brief note on the chosen services to integrate in the joyn app is given.

Last section of this chapter mentions three other applications which were part of the State of the Art analysis, but not a real match for the competitors or pure social network applications. However, these applications had a great impact on the internship, once they lead to some development around their base concepts.

2.1 Competitors

Over the Top applications are proliferating in the communications market, as a study conducted by Informa Telecoms & Media, in April, 2013 proves [3]. Based on the daily traffic for both OTT messages and SMS, these analysts concluded that Over the Top messaging has already surpassed SMS, and predicted that it would be twice the SMS traffic by the end of 2013. In **Table 2.1**, it is possible to analyze the current values for the user count, the average messages a user sends per day and the daily traffic on both messaging types. Although SMS has a huge number of users accumulated along the past 20 years, in 5 years or so OTTs are providing a very strong response.

[1] **Note:** The information presented in this chapter is valid as of February 22nd, 2014, when it was finalized.
[2] Values presented for December are based on Informa's forecast in study from April, 2013.

Type	User count	Daily average	April 2013	December 2013[2]
SMS	3,5Billion	5,0	17,6Billion	19,5Billion
OTT	586,3Million	32,6	19,1Billion	41,0Billion

Table 2.1: Comparison between Over the Top messaging and SMS: user count, daily messages per user and daily traffic in April and December, 2013 [3].

The threat is real, and it concerns Mobile Operators all over the world. Regarding the SMS traffic alone, worldwide, a study performed by Portio Research, Ltd. [7] points that it is now in stagnation, and it will enter the decline phase in 2015. OTT applications depend on network coverage, which still affects some markets in favor of the standard messaging and voice services, either because the network coverage is weak or the plans offered by the Mobile Network Operators are too expensive. Still, in the number one market for SMS – the U.S. – the changes are already visible and the trend is to get worse.

2.1.1 Over The Top Applications

As mentioned before, a total of twenty-five OTTs were analyzed in detail for this State of the Art study. As it would be too exhaustive to present them all, in the following pages, seven OTT applications are presented in detail. These applications are presented among all the twenty-five analyzed both because of their distinctive features and their user count. For the sake of simplicity, only a subset of all the analyzed features is presented, regarding each of the applications: the features selected to present here were the most significant for a normal user, in order to provide a fair head-to-head between the OTT applications. A description about the history of each application is given next, and some of their features and user count are highlighted as well. The current rating from Google Play Store is also provided, since it is the main application's store for the Android platform. Later, in **Table 2.2** the differences between these applications are displayed, comparing the features that each of them supports. **Table 2.3** introduces the platforms supported by each of these applications.

Facebook Messenger

User Count: 874 Million **monthly active** users[3] [8]
Rating: 4,3/5 (rated by **1 128 850** users)

Facebook Messenger focuses on instant messaging for users from the Facebook social network. The first release of the mobile app from Facebook Inc. rolled out on August 2011. The messaging application allows people to text with their friends independently of the platform they are accessing from. Currently, it is available for Android, iOS and BlackBerry mobile operating systems. A desktop version is also available for Windows. In December 2012, Facebook Messenger added the ability to use the application without a Facebook account in some

countries [9]. Instead of their Facebook login credentials, users could provide a name and a phone number, as most OTT applications require. Additionally, along the way the company introduced a feature that let the users send SMS; however, given the low adherence by the users, Facebook took a step backwards and removed the feature [10].

Key Features:

- 1-to-1 Voice over IP (VoIP) calls
- Group Chat
- Chat and Group Chat File Transfer
- Voice Messages
- Location Share

Distinctive Feature:

- Chat Heads (Android version)

When the application is connected and a friend sends a message to the user, a bubble pops up in front of the screen with the message, letting the user tap it in order to open a compact conversation window. From there, the user can reply back directly or open the full conversation inside the application.

Hangouts

User Count: 540 Million **monthly active** users[4] [11]
Rating: 3,4/5 (rated by 216 943 users)

Google Hangouts is an instant messaging and voice/video application developed by Google. It was announced and launched in 15 May 2013, during Google's I/O conference [12]. This application replaced others that Google launched in the past, such as Google Talk, Google+ Messenger and Hangouts feature of Google+, unifying messaging and calls in a single application. Google Hangouts is accessible on Android and iOS devices as an application, but it is also available to desktop platforms, as an extension to Google Chrome web browser.
Hangouts interacts with Google+ social network, using Google+ accounts to access contacts and circles. Much like on Facebook Messenger, the user can communicate through the mobile application or the web browser, independently of where the other party is.

Key Features:

- 1-to-1 Voice and Video over IP calls
- Voice and Video over IP Conferences (up to 10 participants)
- Group Chat
- Chat and Group Chat File Transfer

[3]Users of all Facebook mobile applications as of September 30, 2013.
[4]Users of Google+ Platform as of October 29, 2013.

Distinctive Feature:

- Integration with Google+

Hangouts allows their users to make use of the instant messaging service and Voice and Video over IP calls to connect to their friends over Google+. This works independently of whether the user friends are on the social network on a browser or in their mobile phones in the Hangouts application.

iMessage and FaceTime

User Count: 250 Million **active** users [13]
Rating: N/A[5]

In order to compare Apple's solutions against the other applications, both iMessage and FaceTime will be analyzed together, since one is dedicated to messaging and the other to voice and video calls.
Announced and released in 2011, iMessage is an instant messaging application for iOS and Mac OS X devices. iMessage is an integrated messaging system that makes use of packet-switched networks such as Wi-Fi, 3G or LTE to send instant messages and the traditional SMS/Multimedia Messaging Service (MMS), when network coverage is not available. In October 2012, Tim Cook, Apple's Chief Executive Officer (CEO), announced over 300 billion messages exchanged since the service rolled out [14] – a remarkable number given that the access to the application is restricted to Apple devices.
When it comes to FaceTime, it was announced and released for iPhone and 4th generation iPod Touch, in 2010. Later in that year, compatibility for Mac OS X was also announced. FaceTime is focused in Video over IP communications between any compatible Apple devices. Currently it is supported by Mac OS X, iPad, iPhone and iPod devices.

Key Features:

- 1-to-1 Voice and Video over IP calls
- Group Chat
- Chat and Group Chat File Transfer
- Location Share

Distinctive Feature:

- Multi-Device Sync

When a user configures two or more devices with the same Apple ID, iMessage syncs the conversations across all the devices through iCloud, which lets them read and answer their friends on each of the devices and get it all synchronized.

[5]Rating not considered for iMessage and FaceTime because both applications are not available for Android devices. Therefore, there is no rating available from Google Play Store.

LINE

User Count: 260 Million users [15]
Rating: 4,2/5 (rated by **1 240 992** users)

LINE is an instant messaging application that started in **2011**, in Japan. It began as an alternative communications system for Naver Corporation in Japan, as a response to an earthquake that damaged the nation telecommunications infrastructure [16]. The application was first released for iOS and Android, but nowadays it also supports BlackBerry and Windows Phone. The success of the application was big enough for the creation of a spin-off company named LINE Corporation, which is currently in charge of the application's development and support. LINE is widely used in East Asia and is based in an ecosystem of LINE applications and games that integrate with the messaging application, providing many additional features that improve the user experience and interaction with other users.

Key Features:

- 1-to-1 Voice and Video over IP calls
- Group Chat (up to 100 participants)
- Chat and Group Chat File Transfer
- VoiceMessages
- Location Share

Distinctive Feature:

- Timeline

The application allows users to make use of the in-app timeline, which allows them to post text, pictures and stickers visible to their contacts. It also lets users to like and comment in their friends' timelines, working as a simple social network.

Skype

User Count: 299 Million users [17]
Rating: 4,0/5 (rated by **1 482 834** users)

Skype is known for its VoIP and messaging services all over the world. It was founded in **2003** and the first beta version was released in August of that year. In 2005, eBay Inc. acquired Skype Technologies SA. In 2009, eBay sold Skype to a group of investors led by Silver Lake Partners [18]. In 2011, Microsoft announced the agreement with Skype to purchase the company along with all their technologies for **$8.5 Billion**.
In the end of 2012, Microsoft announced that they would retire their well-known Messenger application, giving the users the choice to migrate to Skype, with the option to merge accounts from both services.

Key Features:

- 1-to-1 Voice and Video over IP calls
- VoIP conferences (up to 25 participants)
- Video over IP conferences (up to 10 participants)[6]
- Group Chat (up to 300 participants)
- Chat File Transfer
- Location Share

Distinctive Feature:

- Mobile and landline calls worldwide

Skype lets users add credit to their accounts in order to perform VoIP calls, so that they can call to any of their non-Skype contacts. The prices of these calls are very competitive when compared to traditional phone/mobile calls.

Viber

User Count: 200 Million users [19]
Rating: 4,4/5 (rated by **1 511 350** users)

Viber is a mobile application based on instant messaging and VoIP calls. The application first released for iPhone as a voice call application, in December **2010**. Viber Media is the company behind the Viber development, and the former president of the iMesh file-sharing client, Talmon Marco, founded it. A first Android application released in 2011, but the company decided to open the access to everyone in July 2012. BlackBerry and Windows Phone support was also added in May 2012. High Definition Voice and instant messaging were introduced in Android and iOS platforms on July, when Viber reached 90 million users. Support for other operating systems like Symbian or Bada was also announced.
Currently, Viber is available in different languages and supports both mobile and desktop platforms, offering a serious alternative to Skype when it comes to VoIP calls.

Key Features:

- 1-to-1 VoIP calls
- Group Chat (up to 40 participants)
- Chat and Group Chat File Transfer
- Location Share

Distinctive Feature:

- Transparent Call Transfer between mobile and desktop

When a call is in progress, a user can opt to transfer the call over to other device, e.g., from the smartphone to the desktop application, while the other party does not notice this change.

[6]Skype video conferences require a Premium account. One-to-one Video over IP is a free feature.

WhatsApp

User Count: 350 Million **monthly active** users [20]
Rating: 4,6/5 (rated by **4 482 688** users)

WhatsApp Messenger is an instant messaging OTT application that lets users share a variety of contents across different platforms, at the cost of **$0.99/year**, after a first year free of charges. The WhatsApp Inc. was founded in **2009** by two former *Yahoo!* workers, and their application is available to all major mobile operating systems nowadays.
This application is growing at an incredible pace. According to Jan Koum, WhatsApp CEO, in the end of October 2013 the application registered 350 Million users using the application at least once a month [20]. That means an increase of 50 Million users since August '13 [21], which demonstrates how the application is doing well.

Key Features:

- Group Chat (up to 50 participants)
- Chat and Group Chat File Transfer
- Voice Messages
- Location Share

Distinctive Feature:

- Ad free

The key selling point of WhatsApp is its creators vision about ads. They say that people are tired of advertisement everywhere, and they just want to provide something that works and saves people some money, making their lives easier. In addition, no ads means a cleaner layout and look, which WhatsApp creators appear to enjoy [22].

In **Table 2.2** the above cited applications are placed head-to-head, in order to understand what is behind their success in the mobile market. The features displayed in that table are now briefly explained:

- **Facebook authentication** Access to the service by logging in with a Facebook account.
- **One-to-one chat** Short messages exchange with a single contact, inside a chat session.
- **Group chat** Short messages exchange with multiple contacts in the same chat session, where every contact receives the messages sent by others.
- **File transfer One-to-one** Transfer of files directly from the application's chat window to the other contact.

- **Location share** Share the current location with the other party, based on the device's Global Positioning System (GPS) receiver.
- **IP voice calls** Voice calls over IP directly from inside the application.
- **IP video calls** Video calls over IP directly from inside the application.
- **Voice messages** Record and send a voice message while inside a chat session.
- **Stickers** Ability to send, receive and display stickers inside the chat session.
- **Stickers store** In-app store to purchase additional stickers, to use in the chat sessions.
- **Online status** Indication of the contacts which are currently available in the application.
- **Themes, skins** Different themes and/or skins in order to change the application's appearance.

Feature	FB Messenger	Hangouts	iMessage + FaceTime	Line	Skype	Viber	WhatsApp
Facebook Authentication	✓	x	x	✓	✓	x	x
1-to-1 Chat	✓	✓	✓	✓	✓	✓	✓
Group Chat	✓	✓	✓	✓	✓	✓	✓
File Transfer 1-to-1	✓	✓	✓	✓	✓	✓	✓
Location Share	✓	✓	✓	✓	x	✓	✓
IP Voice Calls	✓	✓	✓	✓	✓	✓	x
IP Video Calls	x	✓	✓	✓	✓	x	x
Voice Messages	✓	x	x	✓	x	x	✓
Stickers	✓	x	x	✓	x	✓	x
Stickers Store	✓	x	x	✓	x	x	x
Online Status	✓	✓	x	x	✓	✓	✓
Theme, skins	x	x	x	✓	x	x	✓

Table 2.2: Comparative table of each OTT application main features.

As one can infer from **Table 2.2**, there are some features in common to the seven OTT applications presented, like one-to-one and group chat, and file transfer under one-to-one chat. Then there are three or four features that most of the applications support, revealing that the features they provide are very balanced. Still, Line and Facebook Messenger have their credit on supporting almost all the features presented here, that are probably those which the common user looks for in a communications application.

The platforms supported by each of these applications is also a very important factor, which affects directly the number of users that will make use of them. In **Table 2.3** the supported platforms by each application are presented.

Platform	**FB Messenger**	**Hangouts**	**iMessage + FaceTime**	**Line**	**Skype**	**Viber**	**WhatsApp**
Android	✓	✓	x	✓	✓	✓	✓
iPhone	✓	✓	✓	✓	✓	✓	✓
iPad	✓	✓	✓	✓	✓	✓	x
iPod Touch	✓	✓	✓	✓	✓	✓	x
BlackBerry	✓	x	x	✓	✓	✓	✓
Windows Phone	x	x	x	✓	✓	✓	✓
Bada	x	x	x	x	x	✓	x
Symbian	x	x	x	✓	x	✓	✓
PC Win	✓	✓	x	✓	✓	✓	x
PC Mac	x	✓	✓	✓	✓	✓	x

Table 2.3: Comparative table of each OTT application supported platforms.

Based on the information provided in **Table 2.3** the clear winner is Viber, supporting all the platforms in the scope of this analysis. Second place goes to Line, which together with the wide features support may be the reason for their 260 Million users worldwide [15], and to be one of the key players in the Asian market. From another point of view, all applications support the iPhone, and only the iMessage +FaceTime combo does not support Android. In general, all the applications are pretty leveled.

2.1.2 joyn Developers

As the project will occur under WIT Software's joyn product, it was considered necessary to analyze the companies currently developing joyn solutions. This study allowed to get information on WIT Software's competitors, in order to get ideas for features that could be introduced regarding the scope of this project. However, none of the analyzed competitors has already any features regarding social networks and cloud storage, meaning that WIT Software will be ahead of them by the end of this project. The four main competitors to WIT Software in developing joyn products are introduced in the following pages, where information regarding each company and its respective joyn-branded product is presented. At the end of this section, two tables are displayed. The first (**Table 2.4**) presents the same features from **Table 2.2**, this time regarding WIT Software's product and their competitors. **Table 2.5** compares the platforms supported by each of the products under analysis.

Jibe Mobile

Jibe Mobile is a company founded in **2005** and its headquarters are located in Mountain View, CA, USA. The founder and current CEO of Jibe Mobile is Amir Sarhangi, and he is responsible for the company's presence in the mobile industry, since he has more than 15 years of experience in telecommunications [23].

The current company's joyn product is called Jibe ON, which has accreditation for the Android platform conceded by GSMA, and valid until September 2014. The application accreditation level is joyn Hot Fixes, which corresponds to the Rich Communication Services – enhanced (RCS-e) v1.2 specification, by GSMA [24].

Jibe mobile announced in February 2013 that it would deliver a joyn-branded messaging application for MetroPCS Communications, an United States wireless services provider which was the fifth largest carrier in USA. MetroPCS already had a joyn branded client since 2012, created by Summit Tech, another joyn developer, which is also analyzed in this study. Meanwhile, T-Mobile purchased MetroPCS, although all the services provided by MetroPCS are still available.

Later, in October 2013, Jibe delivered another communications application for Sprint, called Messaging Plus. This application is available for both iPhone and Android phones, although it is not joyn branded. The application is able to make calls, instant messaging and content sharing.

Company website: `http://www.jibemobile.com`

Nable Communications

The company was founded in January **2003** and aims to develop solutions for network operators, providing IP Multimedia Subsystem (IMS) infrastructures and applications. Nable Headquarters are located in Seongnam-City, South Korea, and its CEO is Dae-Young Kim, which has a Masters Degree in Electronics Engineering [25].
The company is developing joyn products for both Android and iOS. The Android application is already joyn certified, since May 2013, and works for Android version 2.3 or higher. The iOS client is still waiting the accreditation process to finish, but allegedly works on iOS 4.3 and higher. Both versions are related to joyn Hot Fixes [24].
In 2009, Nable provided RCS solutions for three wireless operators in Korea – SKT, KT and LGU+. Later, in 2012, it provided the joyn service for the same operators. Still in 2012, the company was awarded with the "Communications Solutions Product of the Year", from Technology Marketing Corporation (TMC), for their Rich RCS Suite [26].

Company website: `http://www.nablecomm.com`

Neusoft

Neusoft is an IT services provider from China, founded in **1991**. The company Headquarters are located in Shenyang, China, and its CEO is Liu Jiren. They focus on RCS solutions for Mobile Network Operators. Neusoft was the first RCS client vendor to receive **full accreditation** from GSM Association (GSMA), in September 2012 [27].
Currently, Neusoft Mobile Solutions has an Android RCS-e client with joyn Hot Fixes accreditation by GSMA, awarded in February 2013. This client works for Android versions 2.3, 4.0 and 4.1. Another client by Neusoft is listed in the Rich Communication Services (RCS) client manufacturers list, for iOS platform. This client is still waiting GSMA accreditation, but according to the data provided, it is also targeted at joyn Hot Fixes [24].
Neusoft provides RCS-e solutions for the major mobile platforms, such as Android, iOS, Windows Phone and Symbian. Their client also exists for PC and Mac operating systems [28].

Company website: `http://www.neusoft.com`

Summit Tech

Summit Tech Communications is a company that nowadays focuses in the development of RCS-based applications for various platforms. The company Headquarters are located in Montreal, Canada. The company CEO is also its president and co-founder, Mr. Alido Di Giovanni. It was founded in **1996** and since then tries to deploy innovative IP solutions, such as the RCS applications [29].

The company has currently two joyn applications accredited by GSMA. The Android version of the application runs on 2.2, 2.3, 4.0 and 4.1 and the accreditation was granted in March 2013. Another RCS product by Summit Tech is listed in GSMA accredited clients list, and it is the joyn client developed for MetroPCS in 2012, available for Android devices with version 2.3 or higher [24]. This application was the first RCS 5.0 application commercialized in North America [30].

Summit Tech's Rich Communication Services (RCS) client is available for both mobile and desktop platforms. The mobile clients are available for Android, iOS and Windows Phone. For the desktop platform, Summit has versions of their RCS client for Windows, Mac OS, Linux and Firefox OS.

Company website: `http://www.richcommunicationsuite.com`

In **Table 2.4** there is an head-to-head between these four applications and WIT Software's joyn solution. It was a difficult task to gather information on the features that each application provides, since their objective is to share as less as possible regarding how their applications work. The information was collected mainly from the companies websites, therefore it may not be updated because these products are constantly innovating, to present cutting edge features to the users. The presence of a dash (" – ") in the table means that there is no information regarding that feature for the respective application.

Feature	WIT Software	Jibe Mobile	Nable Comm.	Neusoft	Summit Tech
Facebook Authentication	x	-	-	-	-
1-to-1 Chat	✓	✓	✓	✓	✓
Group Chat	✓	✓	✓	✓	✓
File Transfer 1-to-1	✓	✓	✓	✓	✓
Location Share	✓	✓	✓	✓	✓
IP Voice Calls	✓	✓	✓	✓	✓
IP Video Calls	✓	✓	✓	✓	✓
Voice Messages	✓	-	-	-	-
Stickers	✓	x	x	x	x
Stickers Store	x	x	x	x	x
Online Status	x	x	✓	✓	-
Theme, skins	✓	x	x	x	x

Table 2.4: WIT Software's joyn product features comparing to its main competitors.

As it would be expected, the analyzed applications are very tied up regarding most of the features considered, since most of them make part of the current joyn specification, joyn Blackbird [6], which is what all these applications are seeking to fulfill. On the other end, there are still some features that one or two stand out. Namely, WIT Software offers Stickers, Voice Messages and different themes inside the application, which is better than the competition, given the information that could be collected.

Table 2.5 introduces the supported platforms by each of the competitor applications and also by WIT Software current solution. Once again, the information regarding the supported platforms was collected based on each company's website, which may be outdated, given the companies need to target at as much platforms as possible, to have the potential to reach more users.

Platform	**WIT Software**	**Jibe Mobile**	**Nable Comm.**	**Neusoft**	**Summit Tech**
Android	✓	✓	✓	✓	✓
iPhone	✓	✓	✓	✓	✓
iPad	✓	✓	✓	✓	✓
iPod Touch	✓	✓	✓	✓	✓
BlackBerry	x	x	x	x	x
Windows Phone	x	x	x	✓	✓
Bada	x	x	x	x	x
Symbian	x	x	x	✓	x
PC Win	✓	x	x	✓	✓
PC Mac	x	x	x	✓	✓

Table 2.5: Comparative table of each OTT application supported platforms.

Comparing the platforms that each of the WIT Software competitors support, it is easy to notice a concern on the two bigger mobile platforms nowadays, Android and iOS [31]. BlackBerry OS and Samsung's Bada operating systems are still left behind by all the companies, while some already present support for desktop operating systems, more precisely, Windows and Mac OS. Based on the information collected at the State of the Art writing, only two of the companies supported the Windows Phone platform.

2.2 Cloud Storage Services

The proposal for this project regarding the integration with cloud storage services encompasses three different services: Dropbox, OneDrive and Google Drive. In order to explain the motivation behind the integration with these specific services, a global analysis about the main cloud storage services was conducted. The following pages of this section overview five cloud storage providers. In the end, some conclusions are presented regarding the providers chosen to integrate in WIT Software's joyn product.

Box

Box was initially a college project, and it started in **2005**. In the next year, it was officially launched to the public domain. Aaron Levie is the current CEO of Box, and he was also a co-founder [32]. Box intends to connect people devices and networks, providing a cloud storage solution. Along its almost ten-year history, Box has raised a series of funds that allowed it to purchase small companies in order to enhance its services. The company follows a fremium business model, providing free accounts with up to 10 GB of storage for personal usage. As with many cloud storage providers, this company is already targeting at business, and offers specific plans for the enterprise environment. Apart from the website and desktop versions, the service provides applications for different mobile operating systems, namely Android, iOS and Windows Phone [33].

User Count: 15 Million users [34]

Key Features:

- Free accounts with storage up to 10 GB
- Public share links allow sharing without the need of an account
- Shared folders within multiple Box users
- Google Docs and Gmail integration
- 256-bit AES encryption for stored files
- Mobile clients (Android, iOS, Windows Phone)

Dropbox

Dropbox was created in **2007** by Drew Houston and Arash Ferdowsi, which are its current CEO and Chief Technical Officer (CTO), respectively. The idea of creating an online storage service came from one of its founders frustration to work on multiple computers, with the need for data accessible across them [35]. The company is expanding very well, and along the way it also acquired some smaller companies, in order to mitigate their problems and offer a better service to the end user. Dropbox provides free accounts starting from 2 GB of storage. It also offers a "Pro" account, which provides storage

up to 100 GB for $9.99 per month. Recently, Dropbox also introduced a service focused into business, which offers solutions for teams with up to 5 persons. Dropbox provides mobile access with applications for Android, iOS, BlackBerry OS and Amazon's Kindle Fire [36].

User Count: 200 Million users [37]

Key Features:

- Free accounts with storage up to 2 GB
- Public share links allow sharing without the need of an account
- Referrals links allow people to get extra user space by inviting friends to use the service
- Automatic uploads of media content for the user's Dropbox from mobile devices
- 256-bit AES encryption for stored files
- Mobile clients (Android, iOS, BlackBerry)

Google Drive

Google Drive is part of a variety of services provided by Google, offering a secure place in the cloud to store files, since it was introduced in April, **2012** [38]. Users access the service by logging in with their Google accounts. Google Drive integrates with Google Docs service, allowing to edit files directly from an user's Drive in that platform. A free account over Google Drive is available with up to 15 GB of storage, and the paid accounts go from 25 GB up to 16 TB of storage. Along with the web version, where the account can be accessed, there is also a desktop client for Windows and Mac OS and mobile clients for the major operating systems. An user's Drive is used for two additional services provided by Google: Gmail, in order to save sent and received attachments and Google+ Photos, in order to store photos with an higher resolution than 2048x2048 pixels.

User Count: 120 Million users [39]

Key Features:

- Free accounts with storage up to 15 GB
- Allows to edit files from an user's Drive directly in Google Docs
- Public share links allow sharing without the need of an account
- Mobile clients (Android, iOS)

Mega

The appearance of Mega service is associated with the end of Megaupload, a file-hosting service that was shutdown in January, **2012**. Kim Dotcom was the creator and is currently Mega's CEO, which was made public in January 2013, to "celebrate" the end of Megaupload [40]. The service focuses on a different approach regarding

the file encryption: the encryption process takes place locally, before the upload of the files. This job is done through Javascript inside the web browser, where the files can be uploaded [41]. The free plan of Mega service holds a storage of up to 50 GB, while the "Pro" plans go from 500 GB to 4 TB of storage.

User Count: 5 Million users [42]

Key Features:

- Free accounts with storage up to 50 GB
- File encryption controlled by the client, the encryption process takes place locally
- Mobile clients (Android, iOS)

OneDrive

OneDrive is a cloud storage service provided by Microsoft. When it was released in **2007**, a free account had support for 5 GB of storage, but since then it was upgraded to 7 GB of storage in free accounts (plus 3 free GB for students, during a year [43]). Recently, Microsoft provided an integrated solution for editing documents from an user's OneDrive account in their Office web apps, which competes directly with Google Docs. The service can be accessed through the web browser, the desktop client or one of the supported mobile applications (Android, iOS and Windows Phone). Users sign in with their Windows Live ID, which is also a similar procedure to what Google offers with Google Drive [44].

User Count: 250 Million users [44]

Key Features:

- Free accounts with storage up to 7 GB
- Public share links allow sharing without the need of an account
- Shared folders within multiple OneDrive users
- Microsoft web apps integration
- Mobile clients (Android, iOS, Windows Phone)

The project proposal from WIT Software included the integration with Dropbox, OneDrive and Google Drive. The facts presented above, essentially the users that currently use each of the studied services also support this choice. Currently, among the analyzed services, these are in fact the ones holding more users. Regarding users alone, OneDrive and its 250 Million users are the cloud storage service with more personal subscriptions. Dropbox follows not too far, at 200 Million users, while Google Drive stands at 120 Million users.

Additionally, given the predicted growth tendency of the cloud storage market, the initiative of providing integration with such services may reveal a good investment for the product.

Figure 2.1 shows the predicted evolution of the number of subscriptions in cloud storage services until 2017, based on a report from 2012 [45].

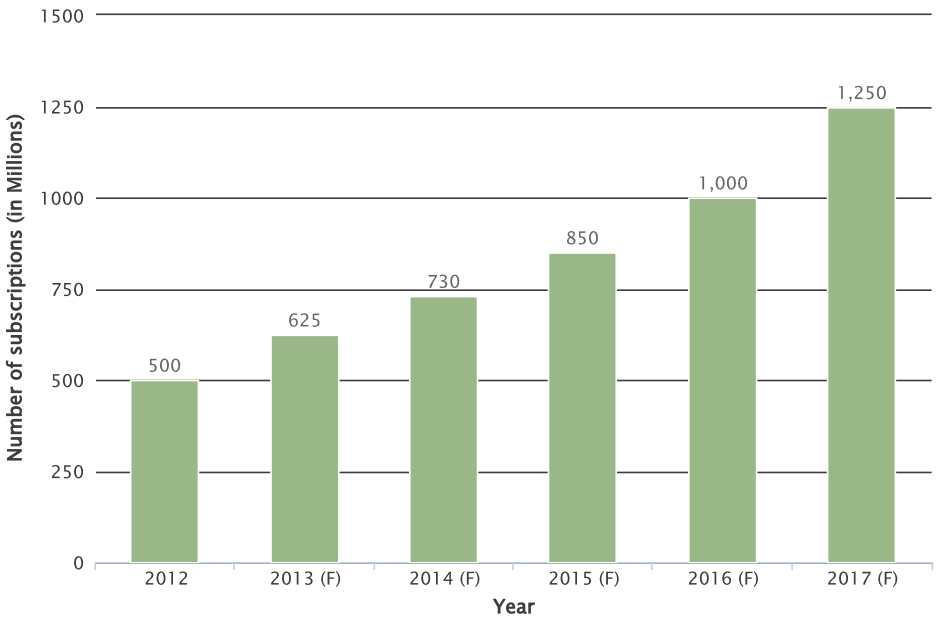

Figure 2.1: Cloud storage usage predictions 2012-2017 [7] [45].

From **Figure 2.1** it is possible to infer the predicted evolution of the personal usage for cloud storage services. Mobility is a determinant factor nowadays, and these services provide a simple solution to solve file access in different places and across several devices. Yahoo Mail is an example of that, allowing the users to attach files directly from their Dropbox accounts [46]. The facts already stated and the good adoption from the users is a good basis to justify an integration with such services inside communication apps.

2.3 Social Network Services

Nowadays, the usage of social network services cannot be ignored. In a continuous tendency to keep growing over the next years, targeting at 2.33 Billion users accessing social network services monthly in 2017 [47], the integration with these services may be considered a must. In **Figure 2.2** this prediction evolution for social networks access on a monthly basis is illustrated.

[7] F stands for Forecast.

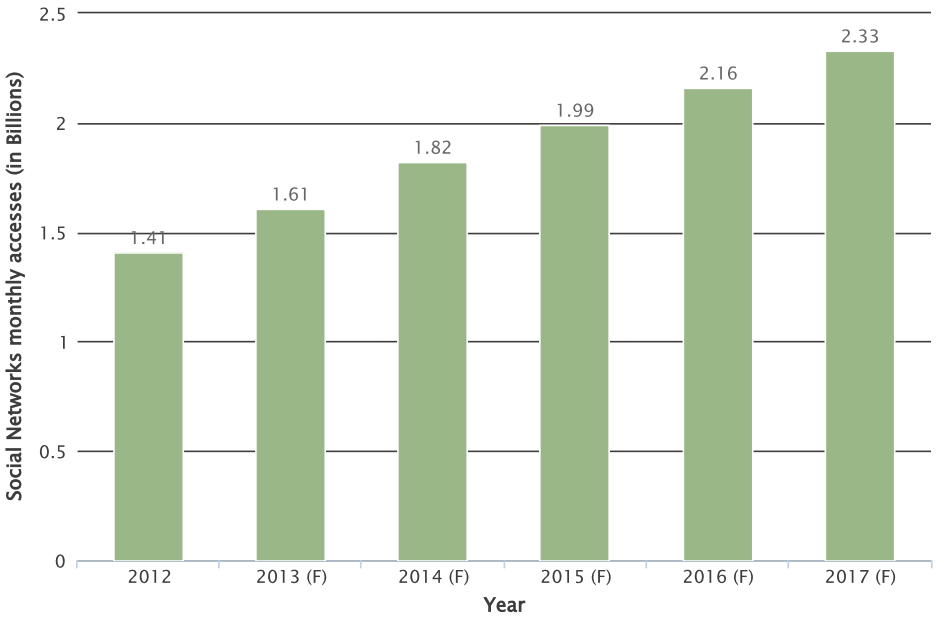

Figure 2.2: Social networks monthly access predictions 2012-2017 [8] [47].

From **Figure 2.2** it is possible to conclude that social networks came to stay. Their growth is slowing down but the predictions point out that they will still be growing at least for the next 4 – 5 years, going from 1.41 Billion monthly active users in 2012 to 2.33 Billion in 2017. Therefore, since the project proposal includes the integration of the three major social network services, over the next pages an analysis of different social network is conducted. In the end, some brief conclusions regarding the data collected are presented.

Facebook

Facebook is currently the most used social network at the global scale. FAcebook started in February, **2004**, when Mark Zuckerberg created it with the help of some friends, as a private social network for Harvard students. The platform was later opened to neighbor colleges, before expanding to high-schools. Nowadays, Facebook is open for anyone that states to be at least 13 years old [48]. It is currently used not only by single individuals but also by social personalities and companies to promote themselves and for advertisement. Facebook also led gaming to a completely new paradigm, mainly because of people's adherence to the many games that appeared inside the platform. Nowadays, Facebook is accessible not only through the web browser but also from a variety of mobile operating systems, such as Android, iOS and Windows Phone.

[8] F stands for Forecast.

User Count: 1.19 Billion users [49]

Key Features:

- Chat support inside the web page and through dedicated messaging applications for Android and iOS
- Search for people, pages, events or media content under the platform
- Groups allow people to join their friends under public or private groups where content can be shared
- Event pages allow to create events and invite users, which can answer to the request
- Available for Android, iOS and Windows Phone mobile operating systems

Google+

Google+ is the social network created by Google. It appeared in **2011** with a slightly different concept from Facebook, since the *Circles* allow to organize friends in groups (circles). Creating different circles allows the users to control who views the contents they share on the platform, and that was the key selling point of Google+ when it was launched [50]. Later, some other features where integrated, which helped the platform to grow and get the 500 Million users it has today. The platform is accessible through the web browser and as an application for Android.

User Count: 500 Million users [51]

Key Features:

- Circles allow to organize people in groups to set different access levels to shared content
- Communities allow to create groups of people to share contents within each other
- Events let users to create events which integrate within Google Calendar
- Integration with Google Hangouts lets users chat and make calls

Instagram

Instagram is a social network that was created with a different purpose than Facebook or Google+. Instagram is about taking photos and recording videos, in order to share them online with friends. The application was created in October **2010**, and it took only 28 months until it reached the 100 Million users mark, in February, 2013 [52]. The application was acquired by Facebook in April, 2012, but Facebook decided to keep both services apart from each other, which revealed to be a successful strategy, given that Instagram keeps growing its registered users. The application is available in the web platform, however there are also mobile clients for Android and iOS devices.

User Count: 130 Million users [51]

Key Features:

- Ability to select filters to apply on pictures and videos before upload
- Integration with other social networks allows to share content in other places
- Following other users allows to see the content that they make available on the platform
- Available for Android and iOS devices

Pinterest

Pinterest is another social network that provides a different approach on social interaction. The idea is to *pin* media content online, in order to make it easily available at a later time. The interesting feature is the ability to follow other people and what they are *pinning*. This social network began as a small beta website, launched in March, **2010**, and worked only with invitations in order to test the service [53]. About an year later, a mobile application for iPhone would release, in March, 2011, bringing in a lot more downloads and users than previously expected. Nowadays this social network is available on the web browser and for the Android and iOS mobile operating systems.

User Count: 70 Million users [51]

Key Features:

- Bookmark media content by *pinning* it on a board
- Ability to follow other users on order to see their boards and what they are *pinning*
- Users can browse by categories and let the application suggest pins from other users
- Available to Android and iOS devices

Twitter

Twitter began in **2006**, and it is currently one of the top social networks, being used by celebrities and companies to post contents online, available to all their followers. Originally designed as a system to work like SMS, each *tweet* has a maximum size of 140 characters [54]. Since its inception, Twitter is growing both in numbers and features, acquiring small companies seeking to provide the best user experience possible. Twitter can be accessed through the web browser and a variety of mobile devices, including Android, iOS, BlackBerry OS and Windows Phone.

User Count: 500 Million users [51]

Key Features:

- Share of small text messages with followers

- Ability to follow other users in order to see their *tweets*
- *Retweet* feature allows to share directly the content shared by another user
- *Hashtags* allow to associate a *tweet* with some category defined by the tag
- Available for Android, iOS, BlackBerry and Windows Phone operating systems

Having in mind the social network platforms analyzed above, the three most prominent are clearly Facebook, Google+ and Twitter. The registered users for these three platforms is by no means comparable to the remaining (see **Table 2.6**), which clearly justifies the choice made by WIT Software regarding the social platforms to integrate within the joyn application. Additionally, the predictions for the upcoming year for Facebook and Twitter are to at least keep up the current users, while Google+ will keep growing the users inside the platform [55].

Social Network	**Registered users (in Millions)**
Facebook	1190
Google+	500
Twitter	500
Instagram	130
Pinterest	70

Table 2.6: Comparison on the registered users of the analyzed social networks.

Regarding the cloud storage services, the conclusion is exactly the same. The services which currently present more users are those which WIT Software selected to be integrated within the joyn product, as displayed in **Table 2.7**.

Cloud Storage	**Registered users (in Millions)**
OneDrive	250
Dropbox	200
Google Drive	120
Box	15
Mega	5

Table 2.7: Comparison on the registered users of the analyzed cloud storage services.

2.4 Other relevant applications

During the State of the Art analysis, other applications were found gaining relevance in the mobile communications market, either by their fast-growing user count or the unique features they provide. Among those, three were chosen to be part of this section, as their analysis was useful in the course of the internship. A brief summary on their history and features follows next.

Snapchat

Snapchat is a communications application with focus in the media sharing segment. It was originally developed by two Stanford University students, Evan Spiegel and Robert Murphy [56]. This application is distinctive from the previously analyzed because it intends to leave no trail either on the sender or the receiver devices: after a media item – video or still photo – is sent to another user, the sender will no longer watch its contents, and the receiver can only view it for a short period of time. The sender defines for how long the content will be available on the receiver side if it is a picture; if it is a video, it vanishes after playing.

Since its inception in 2011, Snapchat has evolved into a 60 million installations product, with 30 million monthly active users by the end of 2013 [57]. Additionally, the company already refused a **$3 Billion** acquisition offer from Facebook, in the last year [58].

User Count: 30 Million **monthly active** users [57]

Key Features:

- Instant photo and video ephemeral messages
- Instant text messaging
- If the receiver tries to take a *screenshot* of the content, the sender is notified
- Video chatting feature allows for two users to stream video simultaneously to each other

Confide

Confide is an application which intends to provide the same ephemeral effect that Snapchat does, but targeting at the enterprise segment, delivering instant, confidential and ephemeral text messages. The idea for this application, says the co-founder Jon Brod, came from the necessity of discussing privately the hiring of a potential employee for his company [59]. The message on the receiver side comes in covered by panes in front of the words, which disappear when the user slides a finger over them. After the message is read, it is erased forever. This application is proliferating in the business environment,

but the founders are not willing to reveal the current user count.

User Count: N. A.

Key Features:

- Ephemeral text messaging
- End-to-end encryption of messages
- If receiver tries to *screenshot* the message, the sender is notified

Glympse

Glympse is currently a reference in social location sharing. It allows for multiple users to share their location in real-time, based on the mobile devices location sensors. The application is very straightforward to use, and does not require any authentication to share the current location with others: it creates a session that expires after a preset amount of time, and the people with access to the session can watch it inside the application or even in the web browser, through a temporary link for that session.
The application allows for multiple persons to share their location in the same session, and it already counted over 10 million users, as of August, 2013.

User Count: 10 Million **monthly active** users [60]

Key Features:

- No need to sign in
- The application is not required to watch someone's location, web browser is enough
- Glympse sessions automatically expire when the time is up
- Share the same session with multiple users allows to watch everyone's location on map

The three applications above analyzed were part of the State of the Art study, and later led to the implementation of two Proof of Concepts developed by the trainee to introduce new features to the joyn application developed by WIT Software, as described ahead on this document.

3 Approach

This chapter introduces the approach which took place in order to put in practice the planned work for the internship. **First section** explains the methodology used along the project, which is based in Scrum. The **second section** introduces the planning for both semesters and their real outcome, analyzing the deviations from the high-level plan.

3.1 Methodology

As previously said, Scrum is the basis of the development plan for the project. In Scrum, the different project phases occur in many iterations along the project, in order to incrementally add demonstrable features. This is one of the reasons why many companies are adopting this methodology. Scrum framework focuses on the priority of the requirements, which defines the implementation order. It relies on two important concepts regarding the implementation: roles and artifacts. **Appendix A** presents complementary information on the methodology, which is briefly resumed in the following pages.

3.1.1 Scrum Roles

Scrum comprises three main roles: the Product Owner, the Scrum Master and the Team Members. The association of these roles to the respective person in the project will follow next. Additionally, the differences from a traditional Scrum environment are introduced [61].

Product Owner

The Product Owner organizes the work that takes place in the project. In pure Scrum, the Product Owner creates and maintains the Product Backlog, helped by the Scrum Master. He defines the priority of the features that must be implemented, causing important features to be implemented first.

In the current project, the Product Owner is the Engineer Frederico Lopes, since he was the responsible for delegating the creation of the Product Backlog to the trainee, and he prioritized the User Stories in the Product Backlog.

Scrum Master

The Scrum Master provides the communication between the Product Owner and the Team Members. He also solves the team's Impediments and is responsible for helping the Team Members to take the maximum advantage of their skills to fulfill their Tasks on each Sprint.

In the project context, the Scrum Master role is played by the trainee, as he was the responsible for solving the Impediments along the way, and to communicate with the Product Owner.

Team Member

The Team Members are responsible for executing the User Stories in the backlog. In the scope of this project, the team is composed of three elements: the trainee hereby reporting his internship and two other trainees, also working under the supervision of Frederico. This allowed the trainees to communicate every morning, to talk about the current progress of their work, as with would be in an usual Scrum environment. However, most of the tasks are executed independently: each trainee has his own Product Backlog, with different User Stories to fulfill on each Sprint. **Figure 3.1** illustrates this small team of trainees in the global scope of the company's RCS team.

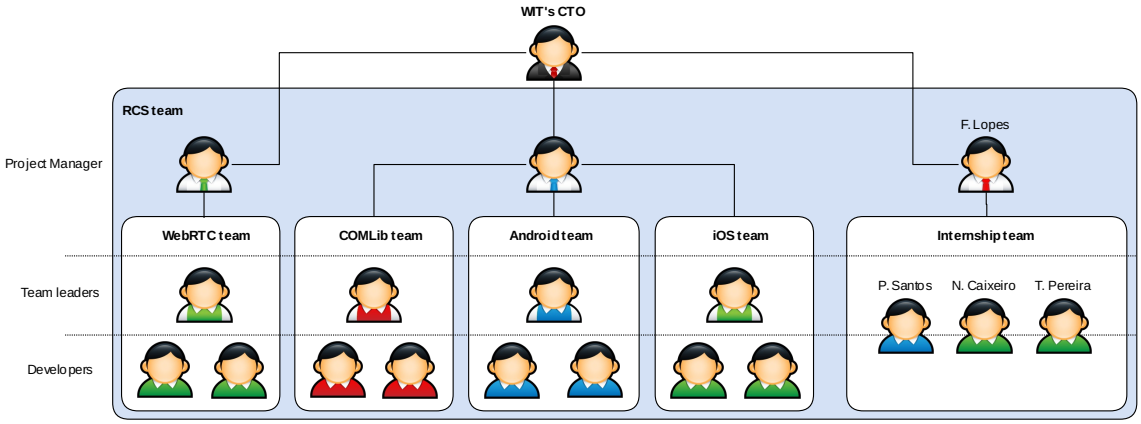

Figure 3.1: RCS team hierarchy at Wit Software.

3.1.2 Scrum Artifacts

Scrum methodology defines a set of documents that provide guidance along the project. The Product Backlog defines all the project requirements in the form of User Stories. The next artifact is the Definition of Done (DoD), which states the compromise from the team to the implementation of requirements. The Sprint Backlogs provide the User Stories that the team committed to fulfill on each Sprint [62].

Product Backlog

The Product Backlog was used to specify the project requirements in the form of User Stories. The full artifact is present in the **Appendix A.5**, more precisely in **Table A.1**. By creating this artifact, the trainee was able to provide suggestions for the project scope, which were appreciated by the Product Owner and included in the backlog. Two examples of features suggested by the trainee were the backup and restore of messages and the implementation of an active message synchronization service, both supported by the user's Dropbox account. The User Stories that compose the Product Backlog are based on the functional requirements collected, which are available in the **Chapter 4** of this report.

Definition of Done

The DoD is a compromise of the Team Members to the validation of their own work. For this project, a DoD document was created before the first implementation activity – the Dropbox functionality prototype. In **Figure A.2**, we can see the three levels defined for the DoD document: story, sprint and release. These levels and the tasks regarding each of them are fully explained in the **Appendix A.2**.

Sprint Backlogs

The Sprint Backlogs are artifacts limited to the Sprint lifetime. They consist on a backlog of the User Stories that the trainee compromises to fulfill along the Sprint. The Sprint Backlogs are created before the start of each Sprint. In **Section A.3** the Sprints completed during the whole internship and respective Sprint Backlogs are presented in detail.

The selected duration for the project Sprints was 2 weeks. Between a Sprint's end and the beginning of the next a meeting occurs, where the trainee and the Product Owner review the outcome of the last Sprint and prepare the next, defining which stories will be held by the trainee.

3.2 Planning

This chapter introduces the planning for both semesters, based on the project requirements and the available time to meet the milestones defined by the Department of Informatics Engineering. The planning hereby presented for both semesters is a general guidance, since the methodology adopted for the project (strongly influenced by Scrum) states that the order in which the features are implemented depends on their priority, which varied during the internship. The Gantt charts presented next were then a guideline for the project, but it must be noted that the priority of the tasks suffered changes, influencing the implementation order.

Both initial plans – for the first and second semesters – are presented next, and compared to the real outcome. A brief analysis on the deviations of each semester is also provided.

3.2.1 First Semester

The initial plan supposed that the internship would start in the beginning of September. The first task to perform was reading the joyn and RCS related documentation, in order to be prepared for the State of the Art study. Next came the creation of the Product Backlog, to define all the User Stories for the project. The prioritizing of some features by the Product Owner changed the initial plan, as one can notice by comparing **Figures 3.2** and **3.3**.

ID	Task Name	Start	Finish	Duration
1	**State of the Art study**	**02-Sep-13**	**14-Oct-13**	**31d**
2	Read joyn/RCS documentation	02-Sep-13	16-Sep-13	11d
3	Indirect competitors analysis	16-Sep-13	01-Oct-13	12d
4	Direct competitors analysis	01-Oct-13	14-Oct-13	10d
5	Create Product Backlog	07-Oct-13	14-Oct-13	6d
6	**Prototyping & mockups**	**14-Oct-13**	**11-Nov-13**	**21d**
7	Dropbox integration prototype/POC	14-Oct-13	28-Oct-13	11d
8	Facebook integration prototype/POC	14-Oct-13	28-Oct-13	11d
9	joyn app study & selection of the integration points	28-Oct-13	11-Nov-13	11d
10	Mockups for the selected integration points	28-Oct-13	11-Nov-13	11d
11	**Solution architecture**	**11-Nov-13**	**10-Dec-13**	**22d**
12	Study current solution architecture	11-Nov-13	25-Nov-13	11d
13	Propose the required changes to integrate with social networks & cloud storage	26-Nov-13	10-Dec-13	11d
14	Write the chapter of the internship report regarding the project requirements & solution architecture	26-Nov-13	10-Dec-13	11d
15	**Development**	**10-Dec-13**	**28-Jan-14**	**36d**
16	Social networks APIs study	10-Dec-13	17-Dec-13	6d
17	Implement Facebook authentication	10-Dec-13	17-Dec-13	6d
18	Implement Google authentication	17-Dec-13	24-Dec-13	6d
19	Contact Enrichment	24-Dec-13	14-Jan-14	16d
20	Preparing the solution for demonstration	14-Jan-14	28-Jan-14	11d
21	**Report writing**	**22-Oct-13**	**07-Jan-14**	**56d**
22	Write state of the Art chapter	22-Oct-13	19-Nov-13	21d
23	Write report introduction	10-Dec-13	17-Dec-13	6d
24	Write architecture & preliminary results chapters	17-Dec-13	31-Dec-13	11d
25	Write future work chapter	31-Dec-13	07-Jan-14	6d

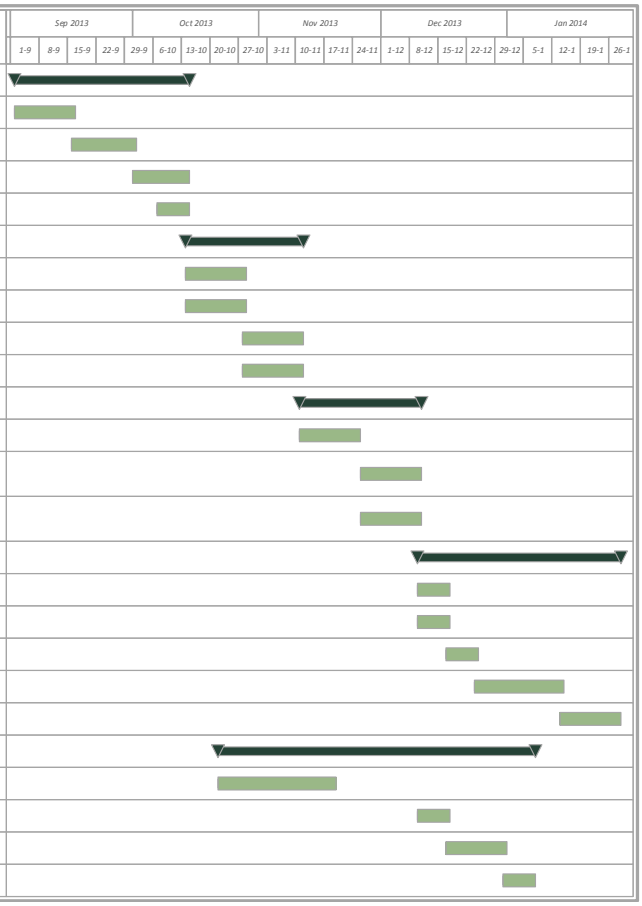

Figure 3.2: Initial high-level plan for the first semester of the project.

From **Figure 3.2** one can conclude that some implementation on the company's joyn product was initially planned for the first semester, mainly referring to authentication on social networks and the development of the Contact Enrichment feature in the project. In fact, the Product Owner decided that the cloud storage integration should take an higher priority, which influenced the outcome of the first semester, as proven by **Figure 3.3**. The report writing task suffered from a slight delay, mainly due to the later kick-off of the project.

ID	Task Name	Start	Finish	Duration
1	**State of the Art study**	**15-Oct-13**	**12-Nov-13**	**21d**
2	Read joyn/RCS documentation	15-Oct-13	28-Oct-13	10d
3	Indirect competitors analysis	29-Oct-13	12-Nov-13	11d
4	Direct competitors analysis	29-Oct-13	05-Nov-13	6d
5	Presentation with WIT competitors to be part of the company's documentation	05-Nov-13	12-Nov-13	6d
6	Create Product Backlog	12-Nov-13	19-Nov-13	6d
7	**Use cases & mockups**	**12-Nov-13**	**10-Dec-13**	**21d**
8	joyn app study & selection of the integration points	12-Nov-13	26-Nov-13	11d
9	Mockups for the selected integration points	26-Nov-13	10-Dec-13	11d
10	Ephemeral Messaging PoC	26-Nov-13	24-Dec-13	21d
11	**Prototyping**	**26-Nov-13**	**03-Dec-13**	**6d**
12	Study Dropbox APIs	26-Nov-13	29-Nov-13	4d
13	Dropbox integration prototype/POC	26-Nov-13	03-Dec-13	6d
14	**Solution architecture**	**03-Dec-13**	**24-Dec-13**	**16d**
15	Study current solution architecture	03-Dec-13	10-Dec-13	6d
16	Propose the required changes to integrate with cloud storage	10-Dec-13	17-Dec-13	6d
17	Write the chapter of the internship report regarding the project requirements & solution architecture	10-Dec-13	24-Dec-13	11d
18	**Report writing**	**22-Oct-13**	**14-Jan-14**	**61d**
19	Write state of the Art chapter	22-Oct-13	19-Nov-13	21d
20	Write report introduction	10-Dec-13	17-Dec-13	6d
21	Write architecture & preliminary results chapters	24-Dec-13	07-Jan-14	11d
22	Write future work chapter	07-Jan-14	14-Jan-14	6d

Figure 3.3: Diagram reflecting the real outcome of the first semester, based on the prioritizing of the features.

By looking at **Figure 3.3** the first difference to the initial plan that stands out is the start date of the project, which caused the time to perform the first semester tasks to be shortened. The prioritizing of the cloud storage integration together with the lack of time led to drop the Facebook prototype and to focus on the Dropbox folder explorer prototype. Apart from that, slight delays were caused because of the delayed start, but the remaining plan could be accomplished, not compromising the report delivery neither the architecture design for the first feature to implement – the Dropbox integration.

3.2.2 Second Semester

For the second semester, the initial high-level plan started with the Dropbox integration. The feature to follow was the implementation of the Facebook share, and the enrichment of the user's native contacts with information from that social platform, as one can see in **Figure 3.4**.

The features planned to be implemented by the middle of this semester were the integration of the remaining cloud storage platforms, and also sharing content on Twitter and Google+. The development stage was supposed to finalize with the implementation of a contact timeline, providing a place inside the contact details for the user to see the last Facebook updates of that contact. Later, a final testing phase and the report writing were planned.

ID	Task Name	Start	Finish	Duration
1	**Development**	**11-Feb-14**	**17-Jun-14**	**91d**
2	Implement Dropbox integration	11-Feb-14	04-Mar-14	16d
3	Implement Facebook share	04-Mar-14	18-Mar-14	11d
4	Implement Contact Enrichment	18-Mar-14	08-Apr-14	16d
5	Implement Google Drive integration	08-Apr-14	22-Apr-14	11d
6	Implement SkyDrive integration	22-Apr-14	06-May-14	11d
7	Implement Twitter share	06-May-14	20-May-14	11d
8	Implement Google+ share	20-May-14	03-Jun-14	11d
9	Implement aggregated Timeline (Facebook, Twitter, Google)	03-Jun-14	17-Jun-14	11d
10	Final Testing	03-Jun-14	01-Jul-14	21d
11	Report writing	13-May-14	01-Jul-14	36d

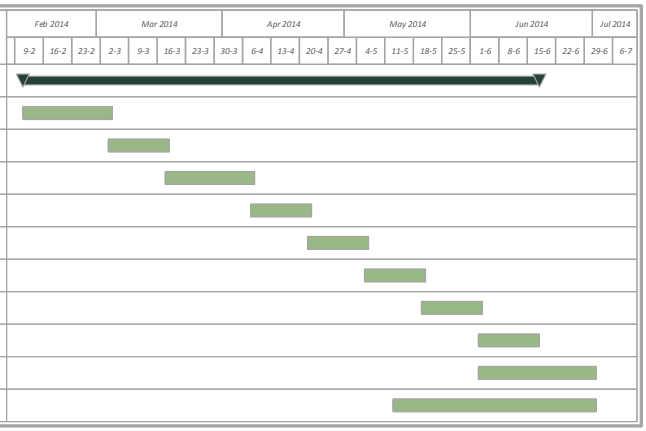

Figure 3.4: Diagram containing the high-level plan for the second semester.

The planning for the second semester – created before the interim report delivery – contained several development tasks, which were ordered by their current priority at that time. Given the fast pace of the telecommunications market, in the space of weeks a lot of things can change. Indeed, the priority and consequent order of the planned tasks changed slightly when one looks at the final outcome, visible in **Figure 3.5**.

ID	Task Name	Start	Finish	Duration
1	**Development**	**05-Feb-14**	**06-Jun-14**	**88d**
2	Implement real-time location	05-Feb-14	04-Mar-14	20d
3	Implement Dropbox share and save	03-Mar-14	25-Mar-14	17d
4	Improvements on real-time location	25-Mar-14	15-Apr-14	16d
5	Dropbox datastore prototype	02-Apr-14	18-Apr-14	13d
6	Testing real-time location	10-Apr-14	18-Apr-14	7d
7	Implement Dropbox backup and restore conversations	16-Apr-14	03-Jun-14	35d
8	Improvements on Dropbox share and save	30-Apr-14	05-May-14	4d
9	Testing Dropbox share and save	06-May-14	09-May-14	4d
10	Testing Dropbox backup and restore conversations	02-Jun-14	06-Jun-14	5d
11	Final Testing	03-Jun-14	19-Jun-14	13d
12	Report writing	13-May-14	01-Jul-14	36d
13	**Future Development**	**23-Jul-14**	**27-Aug-14**	**26d**
14	Implement contact timelines	23-Jul-14	05-Aug-14	10d
15	Implement contact enrichment	06-Aug-14	27-Aug-14	16d

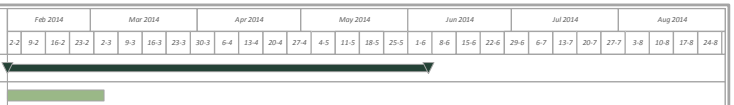

Figure 3.5: Diagram reflecting the real outcome of the second semester, and the tasks planned for the future.

The first thing to notice was the introduction of a new feature, the real-time location share. In fact, this feature had great impact on the whole second semester: it started to be a Proof of Concept (PoC) to be demonstrated in the Mobile World Conference forum, by the end of February, but the implementation and testing of the feature continued until the second half of April. In the meantime, the Dropbox share and save functionalities were developed. The priority established at that point was to finalize the integration with Dropbox, and only then advance to the social networks. Another prototype followed, so that the trainee could get to know better another Dropbox Application Programming Interface (API) to implement the backup and restore features. The implementation of those features followed, with some testing of previous features performed at the same time.

The time window occupied by the real-time location feature forced the integration of social networks in the project to be delayed, as the trainee still had to perform a further validation on all the implemented features, and also had in hands the writing of the report. Two features are currently scheduled to take place after the internship's final presentation: the contact timelines and contact enrichment. These will allow the application to add some value from social networks like initially planned, enhancing the user experience even further.

4 Project Requirements

This chapter presents the requirements identified for this project, covered under two different scopes: functional and non-functional. These requirements were later converted in User Stories to create the Product Backlog of the project, containing all the small tasks to execute along the internship. **First section** introduces the functional requirements identified for this project, while **the latter** explains the non-functional requirements.

4.1 Functional Requirements

The functional requirements specify the *functional* behavior of the features to implement. This list of requirements includes intermediate processes, such as creating prototypes to interact with the APIs which will be used to integrate certain features in the final solution.

Following, the functional requirements for the integration of both cloud storage and social network services in WIT Software's joyn product are presented.

Cloud Storage – Authentication (Dropbox, OneDrive and Google Drive)

- **Req #1 – Link account:** The user must be able to insert his credentials to give permissions to joyn, so that it can integrate with the user's cloud storage account.
- **Req #2 – *Unlink* account:** The application must provide an entry point where the user can *unlink* his account. The application should no longer be able to access the user's data.

Cloud Storage – Save and share files to/from cloud storage services (Dropbox, OneDrive and Google Drive)

- **Req #3 – Save received files from inside the chat window:** The user must be able to save received files in a conversation to a linked cloud storage account.
- **Req #4 – Share files from a cloud storage account:** The application must provide entry points for the user to select and send a file from a linked cloud storage account.

Cloud Storage – Backup and Restore conversations (Dropbox only)

- **Req #5 – Backup conversations manually:** The user must be able to manually save his conversations in the application and save them to his Dropbox linked account.
- **Req #6 – Backup conversations automatically:** The application must provide an option to periodically back up the user conversations to his Dropbox account.
- **Req #7 – Restore conversations:** The user must be able to restore his previously backed conversations from his linked Dropbox account, recovering the state of those conversations.

Cloud Storage – Dropbox navigation prototype (Dropbox only)

- **Req #8 – Link account:** The prototype must provide an option for the user to link his account with the prototype, in order to access his Dropbox folders.
- **Req #9 – *Unlink* account:** The prototype must provide an option to *unlink* a previously linked account.
- **Navigate the user folders:** The prototype must provide the option for the user to browse his folders, displaying folders and files in an ordered list.
- **Req #10 – Upload a file:** The prototype must provide the option to upload a file to a previously selected folder in the user's Dropbox.
- **Req #11 – Download a file:** The prototype must provide an option to download a file from the user's Dropbox, previously selected by him.

Cloud Storage – Dropbox datastore API prototype (Dropbox only)

- **Req #12 – Link account:** The prototype must provide an option for the user to link his account with the prototype, through the Dropbox Sync API, so that the prototype can use his account datastores.
- **Req #13 – *Unlink* account:** The prototype must provide an option to *unlink* a previously linked account.
- **Req #14 – Backup objects:** The prototype must provide an option to backup objects from a list to the datastores inside the user's Dropbox.
- **Req #15 – Restore objects:** The prototype must provide an option to restore the local list of objects to the previously backed up state, recurring to the remote datastores over the user's Dropbox.

Social Networks – Authentication (Facebook, Twitter and Google+)

- **Req #16 – Login:** The user must be able to connect his social network account, in order to grant permissions to the application to access his data from the social network.
- **Req #17 – Logout:** The application must provide an entry point where the user can disconnect his account from joyn. The application should no longer be able to access the user's social network account content.

Social Networks – Contact Enrichment (Facebook only)

- **Req #18 – Update contacts profile with Facebook information:** The application must be able to connect to Facebook through their APIs and get information on the user friends. An association must be established between Facebook friends and contacts from the native contact list, and the latter must be complemented with information from the social network profiles.
- **Req #19 – Configurable fields for updating contacts profile:** The user must have an entry point in the application where he can configure which information the application can/cannot override during the Contact Enrichment process.
- **Req #20 – Update a single contact profile:** The application must provide an entry point where the user can start the Contact Enrichment process for a single contact of his native

address book.

- **Req #21 – Update all contact profiles:** The application must provide an entry point where the user can start the Contact Enrichment process for all the contacts in his native address book.
- **Req #22 – Birthday reminder inside the chat window:** The application must warn the user inside the chat window with a specific user that it is his birthday.
- **Req #23 – Birthday reminder before a call:** The application must warn the user inside the call screen when calling a specific user that it is his birthday.
- **Req #24 – Birthday notification in the notifications bar:** The user must be warned when it is the birthday of one or more of his friends, through a notification in the notifications bar.
- **Req #25 – Enable/disable birthday notifications:** The application must provide an entry point where the user can select whether the notifications should be enabled or disabled.

Social Networks – Contact Timelines (Facebook only)

- **Req #26 – Display the basic information from a friend's profile:** The joyn application must include in the contact details the basic information about that contact, retrieved from a social network.
- **Req #27 – Display latest posts and photos:** The application must include the latest posts and photos by a contact in the contact details page, retrieved from a social network.

Ephemeral Messaging

- **Req #28 – Set the duration of the ephemeral messages:** The user must be able to define, in the application settings, the duration of his ephemeral messages on the receiver's end.
- **Req #29 – Enable and disable the ephemeral messaging mode:** The application must provide an entry point inside the one-to-one chat window for the user to enable or disable the ephemeral mode.
- **Req #30 – Send a media file in ephemeral mode:** With the ephemeral mode enabled, the user must be able to send a media file as an ephemeral message, which disappears from his chat window after reaching the other party.
- **Req #31 – Receive and view ephemeral messages:** The receiver must receive an ephemeral file and be able to watch it. After viewing the content of the ephemeral message, both the message and the files must be deleted from the receiver's phone.

Real-Time Location Share

- **Req #32 – Start a real-time location session in walking mode:** The user must be able to start a real-time session from inside the one-to-one chat window in walking mode, with a duration selected from the available options.
- **Req #33 – Start a real-time location session in driving mode:** The user must be able to start a real-time session from inside the one-to-one chat window in driving mode, with a duration selected from the available options.
- **Req #34 – View the current position in real-time:** The application must be able to present the sender's position on map in real-time, in both sender and receiver sides.

- **Req #35 – Recover from internet losses:** The application must be able to recover from internet connection losses and draw the path which was traveled during the offline time.

General – Network Operations

- **Req #36 – Check for Internet connection to update the UI/UX accordingly:** The application must listen for Internet connectivity changes, disabling the features that require an Internet connection, in case it is not available.

Given the methodology adopted for this project – explained in the previous chapter and respective appendix document – a prioritization of the tasks to execute along the internship took place recurring times. The top priority was attributed to the integration with cloud services, namely Dropbox. By the end of December, the Ephemeral Messaging feature was introduced in the internship's scope, thus leading to its requirements to be fulfill until the end of January. The Real-Time Location requirements became top priority as of February, in order for WIT Software to take a demonstrable prototype for the Mobile World Congress, which took place in Barcelona by the end of that month.

With the introduction of these two additional features in the internship, some requirements were dropped. First, the cloud storage-related features focused in the integration with Dropbox, leaving the remaining services behind. The integration with social networks also was shortened, giving up the integration with Twitter and Google+. Nevertheless, the extended development of the real-time location feature caused a delay in the integration with Facebook, which is now planned as a future work as stated in **Section 7.2**.

4.2 Non-Functional Requirements

The non-functional requirements for the application include necessary checks to perform in order to provide the features specified in this project proposal, with the quality that WIT Software maintains on their current joyn solution. The non-functional requirements inherent to the integration of all the proposed features are presented along the following pages.

Android programming

- **Req #37 – Support for Android 2.3 and above:** The current joyn solution from WIT Software supports Android version 2.3 and above. Therefore, the implemented solution shall keep the support for all the system versions already supported.
- **Req #38 – Implementation according both Android and WIT Software patterns:** The implementation of all the classes and modules in the application must follow, whenever possible, both the guidelines suggested by the Android Developers website[1] and the programming patterns adopted by the company's Android development team.

Application performance

- **Req #39 – Requests to network services:** Network requests are a blocking operation. Therefore, it is imperative that no request is made from the UI thread, to avoid freezing the application UI.
- **Req #40 – Detect connected network type:** The application must perform differently, whenever possible, if over Wi-Fi or cellular networks, in order to adjust requests that are executed periodically.

Application UI/UX

- **Req #41 – Accordance of the new features with the existing application:** The UI and UX of the new features must follow the pattern noticeable across the already existing modules, providing a simple and intuitive usage of the application.
- **Req #42 – Consistency between UI/UX and available resources:** When a resource required to perform an action is not available (e.g., no Internet connection), the options must be adjusted to hide those which won't be able to perform their purpose.
- **Req #43 – Access to cloud storage services through the native applications:** If the user has applications from the cloud storage services installed on his Android device (e.g., the Dropbox client for Android), the access to the user cloud accounts must be made through their specific `Intents`.

Preventive error handling

- **Req #44 – Error handling:** The modules to be introduced in the application must treat errors in a preventive way, avoiding the application to crash if some required resource is is corrupted or not available. An error occurred in the new modules must not compromise the remaining application stability, since it runs important communication services that must always be available.

Regarding the non-functional requirements presented, all of them were fulfilled when of the implementation of each feature in the product. Starting by the support for the Android platform 2.3 and above, all the features were tested and succeeded against a variety of device brands and versions of the Operating System (OS). All the long-running tasks were taken into account not to harm the application's UX, being executed asynchronously and apart from the application's UI thread. A preventive approach was taken while introducing the new features in the product, to maintain its stability even if something related to the new features fails.

[1] Android Developers website: `https://developer.android.com/guide/index.html`

5 | Solution Architecture

This chapter presents the architecture designed for the project. It refers to all the features planned and implemented along the internship, and also provides the use cases of each of those functionalities. The first section shows the global architecture of the final solution, including all the components developed by the trainee. The following sections offer functional details on each of the main features, together with a textual explanation on each component.

First section provides an overview on the application's architecture before and after the conclusion of the features, at two different application levels. The following sections present more detail on each of the implemented functionalities: **Section two** explains the architecture for the Dropbox save and share features; the **third section** introduces the backup and restore of conversations to Dropbox; **section four** explains the architecture behind the ephemeral messaging functionality and the **last section** details the architecture for the real-time location share.

Important note: The joyn application's structure is a private property of WIT Software, whose interest is to keep it as private as possible. Therefore, full diagrams containing the architecture of each feature are omitted in this section of the report, but are present in the **Appendix C**. This appendix is very important, since it shows all the interactions for each feature at the class-level, describing the functionality of each class. To fully understand the whole work hereby presented, the reading of **Appendix C** is strongly recommended.

5.1 Architecture Overview

The joyn application is part of the telecommunications market, and provides rich contents to the end users by connecting to an IMS network. It provides a lot of services which are based on network infrastructures, and at the same time it connects to the usual circuit-switched network to provide "backwards compatibility" with the traditional calls and messages. **Figure 5.1** illustrates three different device types connected to an IMS network.

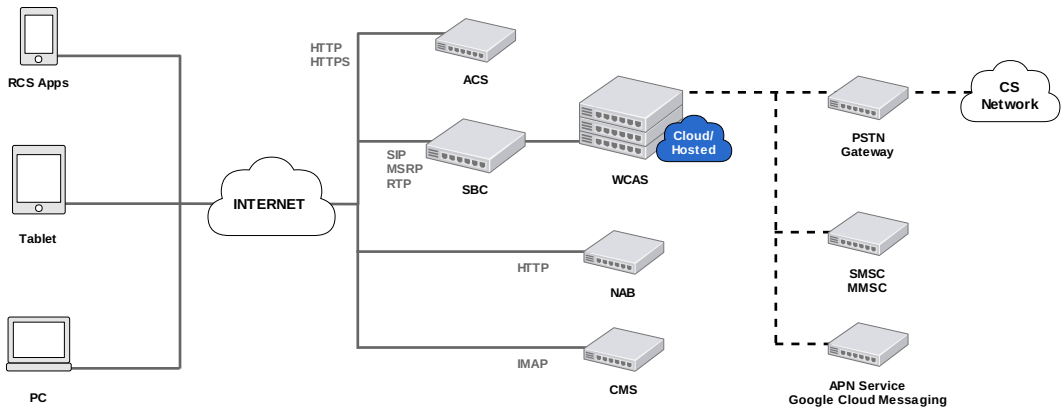

Figure 5.1: Functional architecture of WIT's RCS solution connected to an IMS network.

5.1.1 Communications Library Layer

One of the key things in from the previous figure is the variety of network infrastructures and different communication protocols used. Therefore, the key for success in this field is to have proper tools to communicate, and this is reflected in the structure of WIT Software's RCS solution. The RCS applications in the three presented platforms (smartphones, tablets and PC) use a communications library – designated COMLib – whose main purpose is to provide all the network communications the application needs: registering in the network, messaging, voice and video calls, content share, etc. Additionally, a lot of libraries are required to process media or to provide security for the communications, and this is also present in this library. **Figure 5.2** shows the original architecture of COMLib, previous to any changes implemented by the trainee.

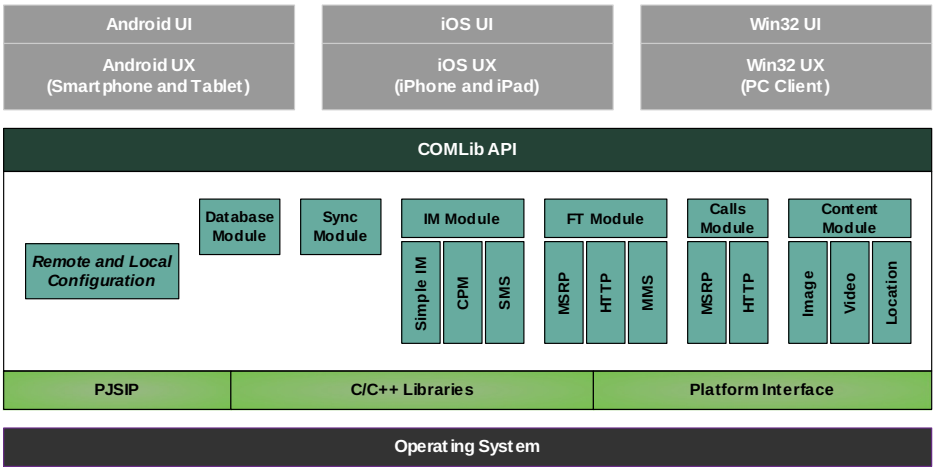

Figure 5.2: High-level diagram of the application's original architecture.

As seen in the figure, on top of the communications library stands an API layer, responsible for providing access to the library modules. Specifically, in the Android platform this layer marks the transition from the C++ language of the communications library to the Java language. The access to the native code is obtained by using the Java Native Interface (JNI) framework.

Regarding the features implemented, only one of them required development in the communications library – the backup and restore of conversations. The need to perform this task inside COMLib is all about the usage of an existing sync module to implement an active synchronization of the messages, to keep the backup ready to be saved in Dropbox all the time. Additionally, the conversations are handled in COMLib's Database module, which also required some modifications. **Figure 5.3** is an high-level illustration of the final library architecture.

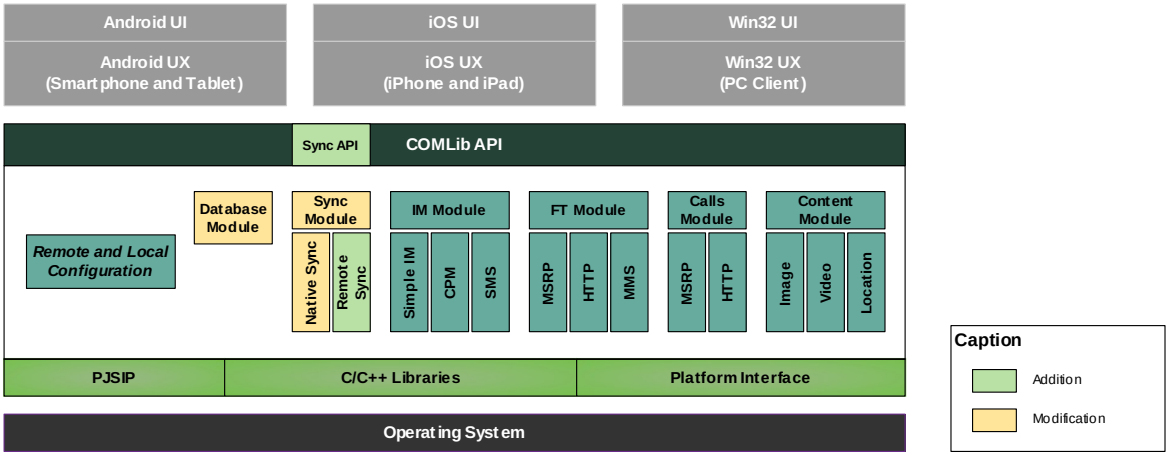

Figure 5.3: High-level diagram of the application's final architecture.

As previously said, the Database module suffered some modifications, as did the Sync module: a new remote sync sub-module was created, to perform the synchronization of the user's conversations to Dropbox whenever a backup takes place. Some code re-factoring was also performed in the native sync module, and a new API was created in the COMLib API layer, to provide access from the UI layer to the sync module.

Some of the modifications in the communications library were introduced together with another trainee, Nuno Caixeiro. Namely, the re-factoring of the native sync module, the newly created remote sync module and the database interactions. Still, the Sync API layer includes some work only related to the Android project – the JNI layer of the module, which allows the Android application to access the sync module in native (C++) code. This later developments were performed alone by the trainee reporting his work.

5.1.2 UI Layer

Regarding the UI layer, the best way to show how it works is a package diagram of the application. In **Figure 5.4** that diagram is illustrated with the more relevant packages for the project, presenting the modifications and additions performed. Packages out of the scope of the internship were omitted for obvious privacy reasons.

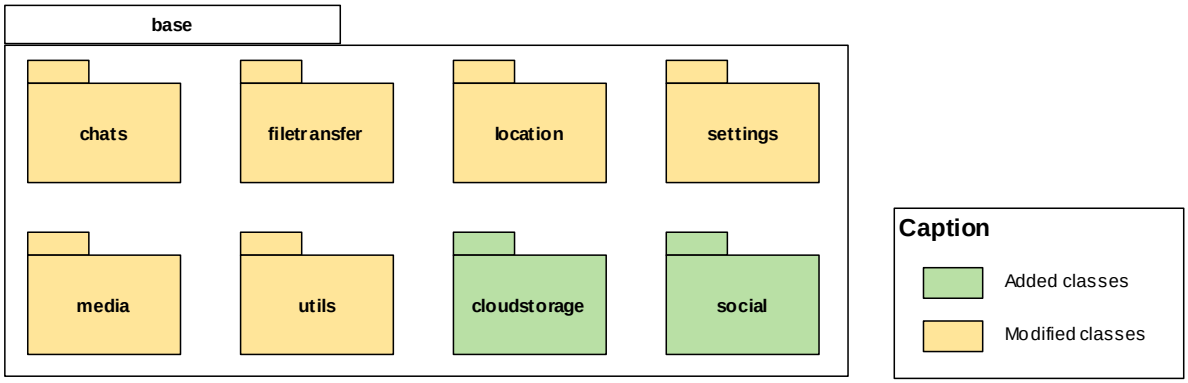

Figure 5.4: Package diagram of the final solution.

Six sub-packages of the application were modified. They mainly refer to the instant messaging and multimedia sharing features, and the application settings. An utility package also required modifications: it suffered from several additions by the trainee during the implementation of all the features.

Regarding the added packages, a new package for all the classes related to the cloud storage integration was created in the product. The social networks integration – which is planned as a future work – will cause the addition of another package in the application that will hold all the logic behind the contact timelines and contact enrichment features. Please refer to the **Appendix C.1** to see the package diagram with an higher level of detail, and the explanation on the additions and modifications of each package.

5.2 Dropbox Share and Save

The Dropbox share functionality is composed of three main tasks in the sender side. First, the user navigates his Dropbox account folders and selects the desired file. When the file is selected, if a thumbnail for that file is available, it is downloaded to the user's phone. The third step consists of sending the information of the original file in the user's Dropbox – a public link for the file – and the downloaded thumbnail. This is what happens when a File Transfer over HTTP takes place, and the implementation of this feature was based on it. **Figure 5.5**

illustrates the process.

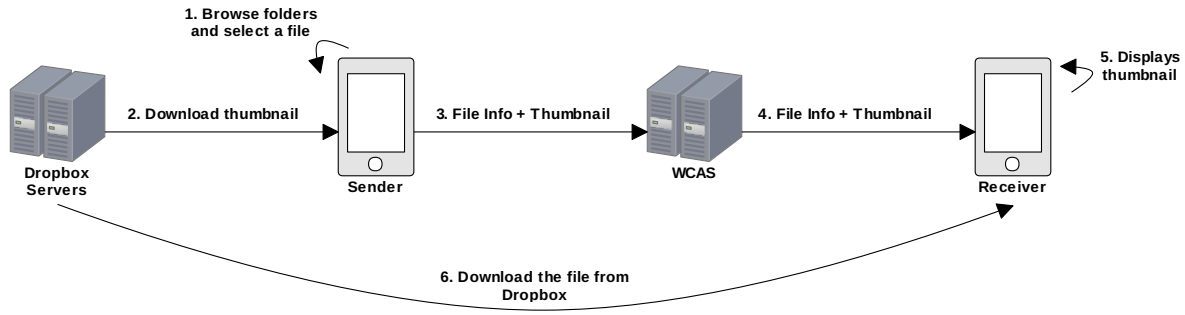

Figure 5.5: Functional overview of the Dropbox share feature.

The sender receives the information of the File Transfer along with the thumbnail. The information is parsed, and the File Transfer is interpreted as a File Transfer over HTTP, meaning that the actual file is stored somewhere in a server, and that it must be downloaded by using the provided URL. Still, if a thumbnail comes with the File Transfer information, it is displayed in the chat window while the file is downloading from – in our specific case – the sender's Dropbox.

On the other hand, the save functionality is very different from the share. When a file is received, the user has a new option to save it to his Dropbox account right from inside the chat window. When the user selects that option, he is able to select the folder where he wants to place the file. When a folder is selected, the file upload takes place. After the operation is finished the user is notified, as detailed in **Figure 5.6**.

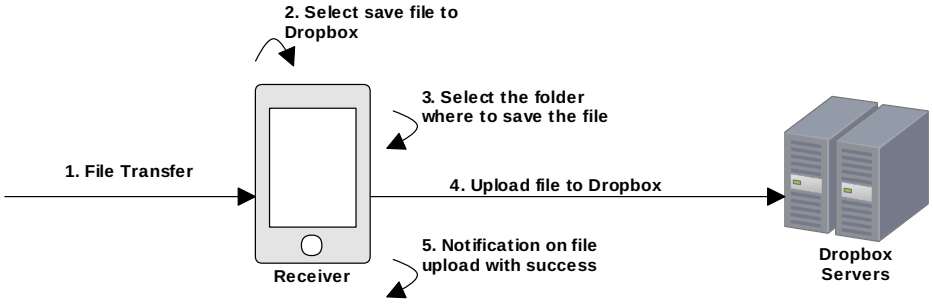

Figure 5.6: Functional overview of the Dropbox save feature.

For a class diagram for the Dropbox share and save features and implementation details, please refer to the **Appendix C.2**.

5.3 Backup and Restore Conversations

Regarding the backup and restore of conversations for the user's Dropbox, the authentication and linking of the joyn application is always the first step, as pointed in **Figure 5.7**.

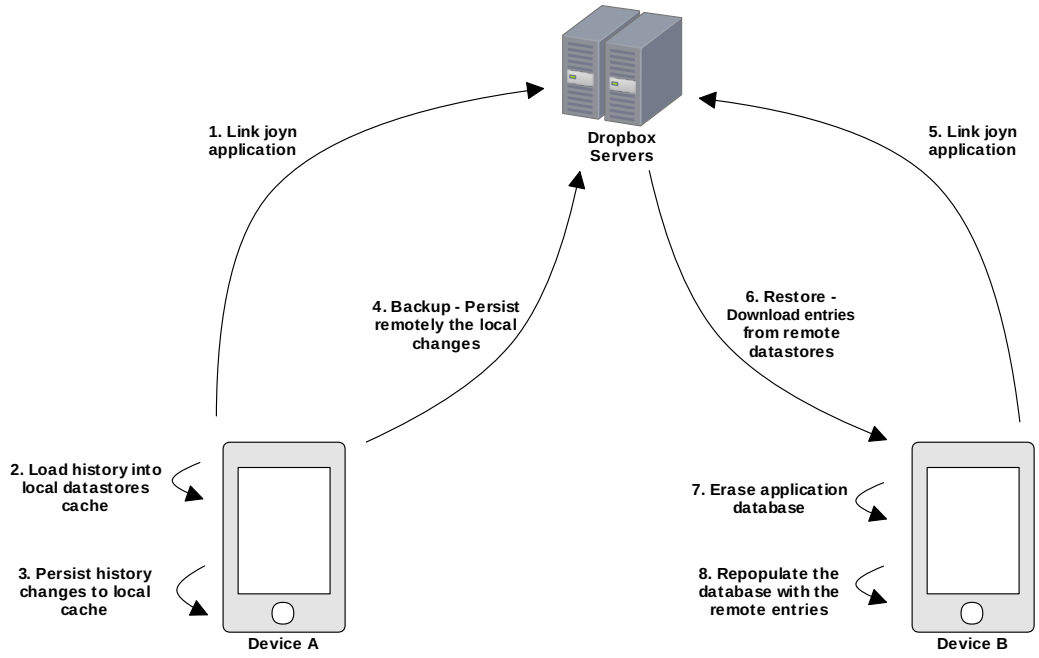

Figure 5.7: Functional overview of the Dropbox backup and restore features.

The figure comprises a scenario where both backup and restore operations take place. In the Device A, after linking the joyn application, the history of current conversations is loaded and persisted into the local cache of the datastores [63]. As history changes take place (messages added, updated or deleted), the local cache of the datastores is updated. When the user of the device A selects the backup option, the local changes are persisted in the remote datastores over the Dropbox servers, recurring to the Dropbox Datastore API.

In device B, the user authenticates with the same Dropbox account, and gives permissions to the joyn application. Later, he selects the restore option, which will fetch the remote entries contained in the datastores to the user's device. The next step is to erase the application's database tables referring to the messaging entries. After that, the entries retrieved from Dropbox are inserted in the database, restoring the state backed up from the device A.

For a class diagram of this feature and implementation details on both layers of the application – UI and COMLib – please refer to **Appendix C.3**.

5.4 Ephemeral Messaging

The ephemeral messaging feature relies on the File Transfer module, which makes use of MSRP or HTTP protocols to transfer files from one user to the other. When the ephemeral mode is enabled, the user can select a media file (image, video or audio) which will be sent in ephemeral mode. The file passes by the application server as usual, and is delivered on the other end, as illustrated in **Figure 5.8**.

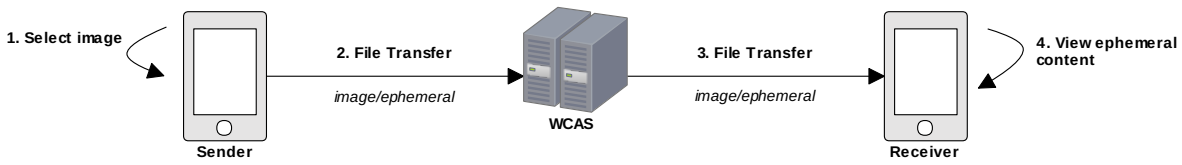

Figure 5.8: Functional overview of the ephemeral messaging feature.

In order to flag the file as an ephemeral message, a specific content type is set on the file. From the figure, the image's MIME Type would be something like `image/png`, but the content type associated with the file sent is `image/ephemeral`. This way, when the file is received on the other end, the application will be able to distinguish it from a normal file transfer, and present it as an ephemeral message to the receiver.

The **Appendix C.4** contains a class diagram related to the ephemeral messaging feature, and explains in detail how the feature works.

5.5 Real-Time Location Share

The real-time location share feature relies on two different modules of the application, the File Transfer module and the Chat module. **Figure 5.9** overviews the flow of a real-time location session inside the joyn application.

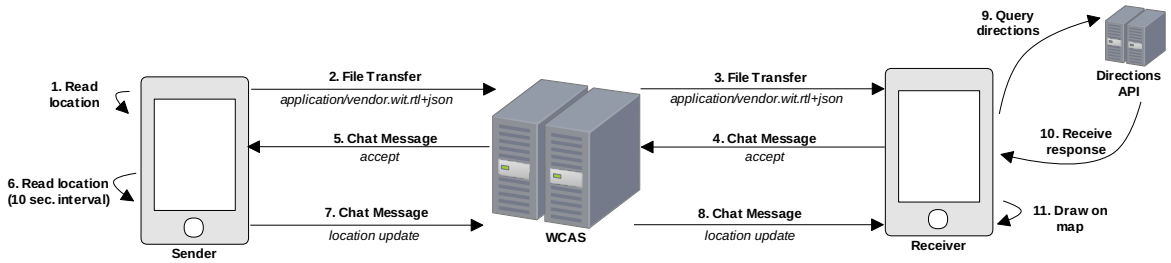

Figure 5.9: Functional overview of the real-time location share feature.

When the sender intends to make an invitation for a real-time location session, the application starts by reading his current location, which is sent together with additional information in a JSON file via File Transfer. The file contains a specific MIME Type which, as in the ephemeral messaging feature, is an identifier of the specific feature. In this case, the `application/vendor.wit.rtl+json` flags this file as a real-time location session. In the above flow the receiver accepts the invite. To do so, the application in the receiver sends an *accept* inside a Chat Message, which is sent in the current MSRP session established between both parties. The Chat Message also contains our specific content type, so that it can be parsed on the other end. When the message arrives to the sender of the invitation, the session starts: his location is read, and from that moment on, it is read periodically, sending *location update* messages to the receiver even when the application goes to background.

In the particular example above, the real-time session is in driving mode. This additionally implies the receiver to query the Google Directions API for a route to the new location of the user, in order to draw the sender's track on the map streets.

Whenever one of the parties in the diagram wants to finish the session, they just send a *finish* message with the specific MIME Type. Otherwise, the session automatically expires after the defined duration has elapsed.

For further details on the implementation of this feature and its complete class diagram, please see the **Appendix C.5**.

6 Implementation Analysis and Validation

This chapter will present two important stages of the development process: the implementation and validation of the solution. Regarding the implementation, in the **first section** some of the challenges and problems solved are presented. **Section two** presents all the validation steps performed regarding each feature, in order to evaluate the final solution of the product, integrating all the implemented features.

6.1 Challenges and Problem Solving

During the implementation stage some problems were addressed. Most of them were predicted far ahead, and the architecture for the features they relate to was designed with those problems in mind. Nevertheless, some other problems were completely unpredictable at a first glance, thus requiring minor changes during the implementation of the features, with some of them being reformulated to work in a proper way.

This section of the document will briefly explain the implementation of each feature, and focus on the main challenges they brought to the project, along with the alternative solutions found.

6.1.1 Dropbox Share and Save

The Dropbox share and save features look pretty straightforward on the final version of the application. The usage of the Dropbox `Dialogs` to handle the folder navigation is much more familiar for the user than a folder browser inside the application. Still, a big issue was addressed in order to allow one the features to work properly.

In order to display these `Dialogs` in front of the joyn application – as illustrated in **Figure 6.1** – the joyn application sends an `Intent` directly at the Dropbox application. The problem is, with an usual `Intent` it would not be possible to get the public link of the file, only to download its contents.

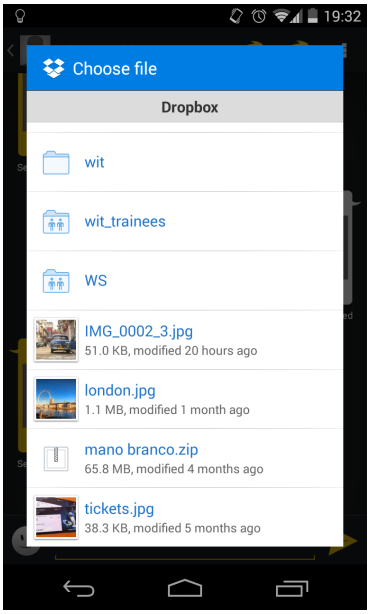

Figure 6.1: Sharing an item from Dropbox inside joyn.

Recently, Dropbox released a new SDK called Chooser [64]. The purpose of this SDK is to allow for easier integration with Dropbox, providing the exact same `Dialog` that Dropbox displays when responding to an `Intent` such as the one we need to trigger. This SDK was analyzed and inside the classes provided by Dropbox lies the secret to get the public link of a file and its associated thumbnails. It consists of a proprietary Dropbox `Intent`, instead of the default `Intents` defined by the Android platform.

By using this specific `Intent`, the result of choosing a file will allow to get the public link for the selected file and also a public link for three different thumbnail sizes. It provides all the necessary information to simulate a File Transfer over HTTP in the joyn application, by sending the file public link and the previously downloaded thumbnail. On the receiver side, the difference from an usual HTTP File Transfer is imperceptible.

Regarding the save feature, the above-mentioned problem does not apply, as one wants to pass to the Dropbox the file contents, through an `Intent`, so that it can upload the file inside the folder selected by the user.

6.1.2 Dropbox Backup and Restore Conversations

Regarding the Dropbox backup and restore, its minimalist UI hides all the horsepower and logic beneath it. To start with, the authentication process is launched from the Java layer of the communications library, since the Dropbox sync manager responsible for all the logic related to the backup and restore features is required to be there. Therefore, it is also responsible for starting the authentication flow, in order to keep track of the currently connected account.

When the user authenticates and links his Dropbox account to joyn, this manager loads all the current history of conversations to the local cache of the Dropbox datastores (for more information on the Dropbox datastores, refer to **Appendix C.3**). From that moment on, it listens for history changes in order to persist them to the datastores cache. When the user selects the "Backup now" option under the settings, all the information is already persisted in the local cache, and it is flushed to the remote datastores by calling a `sync()` method on each of the datastores. This operation persists the currently cached data and now the information stored in the remote datastores could be restored on any device which links that Dropbox account to the joyn application.

This was the approach chosen to implement the backup and restore mechanisms: an active synchronization, keeping track of all the changes and persisting them to the user's Dropbox only when he wants to backup his data. A less orthodox way to do it would be to read all the information inside the database when the user wants to perform a backup, zip all the information to a single file and send it to his Dropbox account. It would certainly be simpler, but a big opportunity would be left behind: by implementing an active synchronization mechanism relying on a remote service, it can easily be changed to synchronize conversations between multiple devices. In practice, if we had three Android devices, we could link our Dropbox account inside joyn in all of them and if we run out of power in the first, we would have the conversations synced on the remaining.

To make this possible, only a few changes are required in the current architecture. First, the history changes had to be synchronized to the remote datastores on the fly, which they currently don't. Additionally, the synchronization manager had to be listening for remote changes on all the datastores, to detect the changes and perform the history change on the device. As simple as that.

Another thing to notice is the continuous sync of the File Transfers, recurring to the Drobpox Sync API [65]. This means that when an history change involves a new file transfer, that file is immediately synchronized to the user Dropbox account. It was not strictly necessary, as it could be done only when the user selects the backup option. This approach has a downside, since the files are occupying the user's Dropbox quota before he wants to perform a backup. On the other hand, it addresses two issues: first, when the user selects the backup option on the device, the backup will occur a lot faster because the only thing that needs to be synchronized are the datastore records and then, in the future it will allow for the easy integration of the above-mentioned remote synchronization mechanism, once the files are already uploaded instantaneously.

Last but not least, another issue was addressed related to the backup and restore of conversations. The initially proposed *mockups* contained logic for selecting independent conversations to synchronize. This could not work under the active synchronization mechanism, because it would require to maintain separate backups per conversation, and when of the backup opera-

tion, to synchronize only the selected ones. The effort would be inglorious as few or no times the users are willing to perform a partial backup of their conversations. Additionally, from the analyzed applications in the State of the Art study, some provided options to backup the user conversations, but none of them did it partially. This option was then removed when the implementation stage began, and the mockups were redrawn to simplify the settings screen.

6.1.3 Ephemeral Messaging

The ephemeral messaging feature also relies on the File Transfer module of the joyn application. The main part of the feature involved the creation of new chat entries for representing an ephemeral message as a generic balloon, hiding the content of the shared file, and a new `Activity` and respective `Fragment` to display the shared media files.

Regarding the playback of media files, some things must be noted. In the first place, as this feature is also implemented in the iOS platform by another trainee, cross-platform compatibility must be kept. Some still pictures coming from an iPhone, when applied directly on a `ImageView` were displayed with extra rotation. The solution was to make use of an utility method which reads the rotation value of the picture from its EXIF information, and rotates the image accordingly. When the media files are audio or video files, the duration of the ephemeral message is read from the file properties. This allows for the `Fragment` and respective `Activity` to be finished immediately after the playback is finished, without any gap introduced by rounding the duration.

6.1.4 Real-Time Location Share

During the implementation of the real-time location feature four main problems were addressed. The first of them related to the need of two different modes, as it happens with many directions services. By allowing the user to select the current mode of his session, he has the power to decide whether the route he wants to share is road-based or not.

The approach to differ both working modes is the following. For the walking mode, successive received readings are connected by using a Bezier curve. The main idea behind this choice is that usually a human walk or run is not a straight line, it is more likely to be a rounded path. Using Bezier curves to calculate the intermediate points and connecting them provides a more realistic path. As for the driving mode, the Google Directions API [66] was used: this API by Google provides a way to ask for driving directions from a position to another. This is the way the driving mode connects two successive points.

However, further problems derive from the usage of the Directions API. This API has a daily limit of 2500 requests for personal usage. An higher limit applies for the enterprise environment, but for the current purpose, it was not an option. Additionally, to call this API an HTTP

request is performed, consuming mobile data from the user's plan, as this feature will be used the most on the outside, where Wi-Fi is not available. In order to spare on the requests, when a route between two points is obtained, it is saved in the file representing the real-time session, so that the next time the map is opened it can be loaded from the local file.

The next problem addressed is related to the readings from the location sensors. When the GPS sensor is not available for some reason, the network-based locations are requested. These locations are far less accurate than those coming from the GPS sensor, since the device relies on the triangulation of cell towers to calculate the user position. The accuracy [67] obtained from the network provider sometimes reaches a value of above 100 meters, which is far from good for our purposes. The problem goes even further with an arguably clear implementation on cached locations by Google: sometimes, the network provider returns cached locations from previous readings, but the *timestamp* returned does not relate to the time the location was collected, but to the time the location was requested. That said, the implemented location source for the system creates its own location cache, and when a location is retrieved it first checks if that location is fresh new; otherwise, the cached location is ignored, since it could be completely wrong.

The last problem is related to the accuracy of the readings. Even a good reading from a GPS sensor on a mobile device has an error radius of 4 to 6 meters. In an highway scenario, this is sufficient to place the reading on the opposite way, which would be a problem when requesting a driving route: the Google Directions API would face that point as placed in the other way of the street, and it would find the next cross on the street to turn around, which most of the times is not the correct path of the sender. To keep the application on track, the speed associated with the location points is also used. The average speed between two readings is calculated by averaging the speed on the start and end points. By multiplying that average speed for the time elapsed between readings, one can have a rough estimate of the distance the user traveled. Still, a small factor is added to the obtained value, to establish the max size a route obtained from the Directions API can have. This way, when the Directions API returns a route too long, it is ignored – as the last position obtained – and when the next location update arrives, the same process takes place. This means than when a more accurate position arrives, it is read from the correct side of the road, and the Directions API will return a more accurate route, which will pass our verification and keep the algorithm going.

6.2 Evaluation

The evaluation of the final solution is an important step on each software project. Regarding the features implemented, different sets of testing took place, and are presented in the next pages. A document with all the information related to these tests is present in **Appendix D**. The tests are presented by the same order they are contained in the appendices, to provide an easier reading. For brevity, a summary of all the tests is given for each of the features.

6.2.1 Functional Tests

Functional testing is a very important step of the validation process. It allows to find erratic behaviors in the application, or even potential crashes due to situations which were not predicted when of the implementation stage.

Regarding the main features implemented in the internship, a total of 85 functional tests were designed and run, to check the behavior of the application respecting to the implemented features. The feature most covered by these functional tests is the real-time location sharing (about 47% of the tests), since it also comprehends more interaction with the user, where an output is always expected.

Some of the features didn't pass all the tests at their first run. These tests allowed to find bugs in the implemented code and to correct them, in order to re-run the full set of tests again. **Table 6.1** illustrates the output of the first and last runs of tests for all the presented features.

	First run		**Final run**	
Feature	Failed	Passed	Failed	Passed
Dropbox Share and Save	0	14	0	14
Dropbox Backup and Restore	1	13	0	14
Ephemeral Messaging	1	14	0	15
Real-Time Location	3	39	0	42

Table 6.1: Functional test results per feature, on the first and last runs of the test sets.

From the table it is easy to identify the more problematic features in the tests. The real-time location share had some issues related to the map behavior, namely with the mode which locks on the last received position from the sender. In the ephemeral messaging there was a *bug* related to the balloon of the incoming ephemeral message, and regarding the backup and restore features, it was necessary to fix a problem after a succeeded authentication, which was causing a problem in the UI.

For a detailed description on the functional tests run for each feature, please refer to the **Appendix D.1**.

6.2.2 Non-Functional Tests

Next, a set of non-functional tests were performed to analyze the behavior of the application regarding different requirements. The first tests are related to network interaction, which all the features implemented rely on. The next set of tests evaluates the usability of the overall work implemented. The third set of non-functional tests evaluates the battery consumption

of the application regarding the features introduced in the internship, and compares the final solution with the base joyn application. Finally, all the features were tested in different devices and versions of the Android OS.

6.2.2.1 Networking

All the features implemented require an active internet connection to work properly. Therefore, a small set of network-related test cases was created for each feature, in order to check whether they work properly together with connectivity problems or with the total loss of connection. In the specific case of the ephemeral messaging feature, it was only a confirmation that an user is unavailable to share in ephemeral mode, because the File Transfer of the file is in charge of the communications library, which by itself already handles connection losses.

In the specific case of the real-time location feature, some more tests were run, once it requires an internet connection to fetch driving routes, when the current session mode is *driving*.

Table 6.2 presents the results of the network-related tests performed on each of the features.

	First run		**Final run**	
Feature	Failed	Passed	Failed	Passed
Dropbox Share and Save	0	3	0	3
Dropbox Backup and Restore	0	3	0	3
Ephemeral Messaging	0	1	0	1
Real-Time Location	1	4	0	5

Table 6.2: Network test results per feature, on the first and last runs of testing.

As seen in the results table, the only feature which had a problem in the first run of the network-related tests was the real-time location share. The issue was related to the test case number **N.1.5** of the real-time feature, in the **Appendix D.2**. When the connection was restored, there was a minor problem with the algorithm to continue fetching directions from the place where it stopped before the connection loss. That issue was addressed, and in the final run of tests the application passed all the test cases with success.

6.2.2.2 Usability

Later, a set of usability tests were performed in order to analyze the UX of the features implemented in the application. A total of 28 volunteers from WIT Software were used as the test subjects.

Some important data regarding the test subjects must be introduced first. A large portion of the persons selected – 19 from the 28 participants, about 67% – had previous contacted with the joyn application. Additionally, not all of them were Android everyday users. The distribution of the participants by their everyday mobile platform is as follows:

- 15 Android users
- 9 iOS users
- 3 Windows Phone users
- 1 BlackBerry user

In order to perform the validation of the features usability, a different test was performed to each of them. Each test consists of a task that the users must fulfill in the minimum possible number of interactions with the application. All the users start from the contact list screen in all tests, so that all perform the test under the same conditions. Each of the tasks is described next, along with their results.

Dropbox Share and Save
As these two features are very similar in terms of user experience, one of them was chosen representing the pair. The task assigned to the users was to share a file from the Dropbox with a specific contact. The easier way is to select the contact from the list, use the share button and from the `Dialog` which appears next, select the "Share from cloud" option, and follow obvious steps to send a file. Other alternatives would do, but they would require additional steps, so this was our comparison term, with a total of 5 required steps.

The test resulted in more than a half (53%) of the users fulfilling the task in the minimum steps. The average number of steps spent by the test subjects was 6.36, which is a very good result.

Dropbox Backup and Restore
To represent the backup and restore features, one of the features was chosen. Our task consisted of login with a Dropbox account which contained a previous backup of conversations, and perform the restore of those conversations to the device. The complete task consisted of 6 distinct steps, once again starting from the contact list inside the joyn application.

This task revealed to be more complex to the majority of the subjects, with only 9 out of the total 28 finishing the task in the least number of steps possible, which is roughly a third of them. Still, the average number of steps required to perform the task was 7.92 steps.

Ephemeral Messaging
This feature is arguably the one requiring more attention from the UX, as the test results proved. The user had to perform two smaller tasks: first, to change the ephemeral message timer under the settings, and later to send an image in the ephemeral mode to a specific contact. The full task comprised 11 steps.

In fact, the results could have been better. From the 28 subjects in the study, only 4 of them concluded the task in the minimum number of steps. The worst scenarios were two users taking 22 and 21 steps to perform the task, which is the double of those expected. The average number of steps required in this test was 15.07, versus the minimum 11.

From the results it was concluded that the UX of this feature can be further improved. From the suggestions of the test subjects, placing the timer option somewhere during the process of sending an ephemeral message would make the whole process more intuitive.

Real-Time Location Share
In order to test the usability of the real-time location feature, the task chosen was to select a specific contact and share the current location in real-time during 25 minutes, in driving mode.

The results of this test were also satisfactory, with about 40% of the users completing the task in the minimum number of steps required. The average number of steps required was 7.93, which is a fair value comparing to the minimum 6 steps in order to perform the task.

In the **Appendix D.3** all the tests are explained in detail, along with charts respecting the individual steps required by each subject in all tasks.

6.2.2.3 Battery Consumption

Battery consumption is a very important component of each mobile application, since the battery is the device's only power source. Having this in mind, programming for mobile platforms must always preserve the device battery, and battery tests must be written and run to measure the impact of the application in the devices. In **Appendix D.4** the full list of battery tests and further details and charts related to the tests are presented, while the following pages resume the test results.

In order to measure the battery impact of the added features in WIT Software's joyn product, a set of battery tests was conducted. A mobile device from the tests team at WIT was used – a Samsung Galaxy S III – from where all the information was erased first, by performing a factory reset. A power monitor device was used to measure the consumed energy during specific amounts of time, while the features were used in the device.

The first test analyzed the battery impact of the application while in background during four hours. To do so, the performance of the application containing all the implemented features was compared to the base joyn application (with no additional features) and also to the system default with no applications installed.

The results of the first test revealed an aggregated increase of 3% over the base joyn applica-

tion, which are not very significant since it is impossible to assure the same conditions across all tests: network and Wi-Fi signal strengths may dictate a relatively higher or lower consumption. In idle mode the features do not add much impact on the application, since no background activities are executed.

Dropbox Share and Save

The next test relates to the Dropbox share feature. By comparing a File Transfer from Dropbox with a File Transfer from the device's gallery, it was concluded that sending a file from Dropbox adds an extra 2% of energy consumed. Again, this is not a solid value, as the difference is too small to make any conclusions.

Regarding the saves to Dropbox, there was no other feature to compare with. The measured consumption – 6731.29 mAh – compares to a File Transfer to another user. In fact, as analyzed in the **Appendix D.4**, the value is pretty close to the consumption measured on File Transfers.

Dropbox Backup and Restore

The backup and restore features were analyzed by comparing a set of interactions inside a chat on an application with and other without the feature active. A total of 10 Chat Messages, 2 Locations and 1 File Transfer were exchanged with other user in both cases. Additionally, in the end a backup was performed in that version of the joyn application.

The results obtained show an increase of about 26% in the total consumed energy. This value is justifiable by the continuous synchronization of messages to the Dropbox datastores cache, and in the end by the upload of all the information (including a file) to the Dropbox. Additionally, to interact with the Dropbox Datastore API, the Dropbox SDK has services running on their own, which also increase the battery consumption.

Ephemeral Messaging

In order to check the consumption of the ephemeral messaging feature, it was compared with the simple File Transfers inside the application. The files were sent to the own device in both tests, and in the ephemeral message case the media file was also displayed in the end.

An increase of 9% was registered over the normal File Transfers. This is mainly due to the *ziping* and *unziping* of the multimedia files, and also the deletion of the file in the end.

Real-Time Location Share

The last feature was tested in two different scenarios. In the first, the device shared its location with other in walking mode, while in the second it received a session in driving mode. This allows to compare the high GPS usage (in the first scenario) with the high usage of network access (in the second scenario).

The result of this test demonstrated that in the current implementation, the second scenario wastes about 13% more battery. This is normal, since in the driving mode a single location update involves at least an HTTP request to Google Directions API. On the other hand, the sender only reads the location from GPS at a 10 second rate, which revealed to be a fair amount to keep the battery usage low.

6.2.2.4 Android Devices Coverage

Given the well-known fragmentation in the Android system [68], the features implemented were tested in various devices, running different versions of the OS, in order to find potential problems.

In fact, by running the application in devices other than the usual device used for testing – an LG Nexus 4 – a problem was found in the backup and restore feature, due to the handling of the file system paths in devices which do not have an external memory card (such as the Nexus 4). By running the tests and checking the output the problem was found and fixed. Another set of testing followed, as **Table 6.3** shows:

	First run		**Final run**	
Device	Failed	Passed	Failed	Passed
Galaxy S	1	3	0	4
Galaxy S III	1	3	0	4
Galaxy S III Mini	1	3	0	4
Galaxy Young	1	3	0	4
Nexus S	0	4	0	4
Nexus 4	0	4	0	4
Nexus 5	0	4	0	4
Xperia L	1	3	0	4

Table 6.3: First and last runs of the four major features in different devices.

It is possible to verify the problem from the backup and restore feature in a representative amount of the devices – those with an external card. Still, the problem was fixed and the outcome of the final round of tests was 100% successful.

For more information on the devices used for this test, please refer to the **Appendix D.5**.

7 | Conclusion

This chapter intends to overview the work done during this internship, and to provide information on the future work after the final presentation. The chapter is divided in three sections. The **first section** resumes the work done, and provides suggestions for improvements on the features developed. The **second section** projects the future work, addressing the features that are still in the scope of the project. **Last section** presents some final thoughts of the trainee about the whole internship.

7.1 Overview

In this section a general overview of the work done will be presented. Specifically, now that the work is finalized, a small analysis on each of the implemented features will take place, along with the suggestion of some improvements, when applicable.

7.1.1 Dropbox Share and Save

Initially the feature was thought to be implemented inside the existing joyn file browser. A Dropbox navigation prototype was created in the 1st semester, and the early version of the implementation contained a new Dropbox entry in the application's file browser. However, the user experience was very different, and another approach took place, by integrating with the native Dropbox application. Some feedback was collected from co-workers and the usability was proven to be better when people were confronted with the Dropbox `Dialogs`, which they already knew from the Dropbox application.

The final solution for navigation and folder listing was then to make use of the Dropbox application `Dialogs`, which is pretty much the base philosophy of the Android applications, the communication among each other. It proved to provide a better user experience, so it was the final solution for the share and save features. This was a case where the simple really proved to be better.

7.1.2 Dropbox Backup and Restore

Regarding the backup and restore of conversations, these are two features which involved development across all the application layers, which was a big challenge. There are still some improvements that can be introduced in the future.

The implementation covers a working mode which depends on the current state of the phone. It will only perform the automatic backup if the device is connected to a power source, allowing to spare the device battery. Nevertheless, it currently does not check the current internet connection of the device. A good improvement would be to create another mode where the device would backup the conversations only when over Wi-Fi, and this could even be configurable by the user in the settings.

Further thoughts on the improvements are related to the database insertions when restoring conversations. The current restore mechanism is composed by first dropping all the records from the database tables, and later insert each of the records in the respective table. This mechanism can be improved, since as of version 3.7.11 of SQLite it supports multiple insertions at a time [69]. As the entries are retrieved from Dropbox based on their type, it is a viable improvement which can lead to a greater performance when handling hundreds of entries.

7.1.3 Ephemeral Messaging

The concept behind the ephemeral messaging feature is straightforward at the first sight, but to provide the true functionality it is supposed to offer, some extra cautions must be handled. The current mechanism of exchanging ephemeral messages consists on exchanging a zipped file containing two files inside: one is the media file sent and the other is a simple file containing the duration of the ephemeral content.

This mentioned approach contains a pitfall. In older devices, with less available memory for the application, the zip and unzip processes are very heavy. The user experience is degraded because it takes longer to zip and send the file, and to unzip it before watching its contents.

7.1.4 Real-Time Location Share

Regarding the real-time location feature there is not much left to talk about. A series of improvements have been made, and the current algorithm provides pretty accurate results.

Still, if a route cannot be obtained for a long period of time – which is barely happening now – the map stands still with no feedback for the user. This could mean that the receiver is not getting the sender's position updates because he went offline, or because a reasonable route could not be fetched from the Directions API. The first of these could cause a message to

appear notifying the user that the connection with the other party was lost temporarily.

Additionally, the current position of the sender is represented on map by an arrow. The arrow's bearing is retrieved by calculating the angle between the last and the previous position of the sender, which not always provides the better result possible. The bearing is available for reading together with the current location on both platforms currently supporting the feature – Android and iOS – so it could also be exchanged when a position update occurs, to set the arrow bearing properly.

7.2 Future Work

As part of an evolving application such as WIT Software's RCS solution, this project does not end here. Two features are projected to be implemented on the company's product in the future – a contact enrichment module and the contact timelines. These were in the initial scope of the project, but given the introduction of the real-time location and ephemeral messaging features, there was no time to conclude them in time to be presented as part of the internship. Nevertheless, it must be noted that their implementation is already scheduled, as described in the **planning section** of this report. The architecture planned for those features was created in the beginning of the second semester, and is also illustrated in the appendices **C.6** and **C.7**.

Some improvements mentioned on the previous section may take place, depending on their priority for the company. The use cases of the developed features proved to be good, however they may require some modifications in the UI/UX, as it is the case of the ephemeral messaging feature.

7.3 Final Remarks

To conclude this report, some personal thoughts about the internship are presented. It was almost a year full of new challenges and lessons learned every day. I was given the freedom to gather the pieces together when of the Product Backlog, and some of the features included on it – such as the backup and restore of conversations to Dropbox – were suggestions of my own. It was even more rewarding to implement those.

The initial plan for the internship contained numerous suggestions for integration with social network and cloud storage services, some of them are not completed as of now. Still, the balance could not be better: by implementing two completely different features, as the ephemeral messaging and the real-time location share, I was given the opportunity to learn on distinct areas of the Android programming. Particularly, the real-time location feature was very interesting to create and keep improving, by introducing new algorithms to enhance the performance and precision of the system as a whole. It was a great experience, given that it

allowed me to work on very distinct areas of the Android programming, from collecting location data from sensors to performing requests to the Google Directions API [66] and analyze the responses to understand if they were good enough.

In the end it was a great experience, full of lessons about programming patterns, code structure and also a deeper insight on the Android programming world. I even was introduced to a completely new programming language – C++ – which I had to learn while *reading* the existing code, to understand the basic mechanisms and architecture of the communications library. The implementation of features in such a delicate part of the application was a great responsibility, but also a remarkable personal achievement for me.

Bibliography

[1] K. Whitfield. *Global mobile statistics 2012 Part C: Mobile marketing, advertising and messaging*. mobiThinking. Oct. 21, 2013. URL: http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/c.

[2] *Portio: OTT messaging apps are not cannibalizing SMS in most markets*. FierceWireless. Oct. 22, 2013. URL: http://www.fiercewireless.com/story/portio-ott-messaging-apps-are-not-cannibalizing-sms-most-markets/2013-10-22.

[3] P. Clark-Dickson. *Press release: OTT messaging traffic will be twice the volume of P2P SMS traffic by end-2013*. Informa Telecoms & Media. Apr. 29, 2013. URL: http://blogs.informatandm.com/12861/news-release-ott-messaging-traffic-will-be-twice-the-volume-of-p2p-sms-traffic-by-end-2013/.

[4] *Rich Communications*. GSMA. URL: http://www.gsma.com/futurecommunications/rcs/ (visited on 10/22/2013).

[5] *Specifications | Future Communications*. GSMA. URL: http://www.gsma.com/futurecommunications/faq/specifications/ (visited on 10/22/2013).

[6] *joyn Blackbird Product Definition Document*. Version Version 2.0. GSMA. URL: http://www.gsma.com/futurecommunications/wp-content/uploads/2013/10/Blackbird-Product-Descrption-Document-v2.0.pdf (visited on 09/26/2013).

[7] *Mobile Messaging Futures 2013-2017*. Version 7th Edition. Portio Research. July 4, 2013. URL: http://www.portioresearch.com/media/4532/Mobile%20Messaging%20Futures%202013-2017%20SAMPLE%20PAGES.pdf.

[8] *Key Facts – Facebook's latest news, announcements and media resources*. Facebook. URL: http://newsroom.fb.com/Key-Facts (visited on 10/22/2013).

[9] T. Geron. *Facebook Messenger Takes On SMS, With No Account Needed*. Forbes. Dec. 4, 2012. URL: http://www.forbes.com/sites/tomiogeron/2012/12/04/facebook-messenger-takes-on-sms-with-no-account-needed/.

[10] D. Ruddock. *Facebook Messenger Getting Major Overhaul: SMS Dead, No Longer Need To Be Friends To Send Messages*. Android Police. Oct. 29, 2013. URL: http://www.androidpolice.com/2013/10/29/facebook-messenger-getting-major-overhaul-sms-dead-no-longer-need-to-be-friends-to-send-messages/.

[11] A. Barr. *Google's social network sees 58% jump in users*. USA Today. Oct. 29, 2013. URL: http://www.usatoday.com/story/tech/2013/10/29/google-plus/3296017/.

[12] J. Fingas. *Google launches new Google+ Hangouts platform and mobile apps with focus on conversations*. Engadget. May 15, 2013. URL: http://www.engadget.com/2013/05/15/google-hangouts-redesign/.

[13] J. Smith. *Apple's popular iMessage and FaceTime services are suffering downtime [Update: back] – Pocket-lint*. Pocket-lint ltd. Apr. 9, 2013. URL: http://www.pocket-lint.com/news/120411-apple-imessage-down-april.

[14] T. Haselton. *Tim Cook: 300 Billion iMessages Served, 70 Million Photo Streams Shared*. TechnoBuffalo. Oct. 23, 2012. URL: http://www.technobuffalo.com/2012/10/23/tim-cook-300-billion-imessages-served-70-million-photo-streams-shared/.

[15] K. Mohan. *Free Mobile Messaging App LINE Rakes In 10 Million Users In India In 3 Months From Launch*. IBT Media Inc. Oct. 9, 2013. URL: http://www.ibtimes.com/free-mobile-messaging-app-line-rakes-10-million-users-india-3-months-launch-1418486.

[16] R. Bushey. *The History of LINE, Japan's Most Popular Texting App*. Business Insider, Inc. May 15, 2013. URL: http://www.businessinsider.com/history-of-line-japan-app-2014-1.

[17] M. Swider. *Microsoft highlights 299M Skype users, 1.5B Halo games played*. TechRadar. June 27, 2013. URL: http://www.techradar.com/news/software/operating-systems/xbox-live-upgrade-includes-300-000-servers-600-times-more-than-its-debut-1161749.

[18] R. Wauters. *Confirmed: eBay Sells Skype In Deal Valuing It At $2.75 Billion*. TechCrunch. Sept. 1, 2009. URL: http://techcrunch.com/2009/09/01/confirmed-ebay-sells-skype/.

[19] J. Crook. *Viber Releases Update To BlackBerry App To Catch Up With iOS And Android*. TechCrunch. Oct. 23, 2013. URL: http://techcrunch.com/2013/10/23/viber-releases-update-to-blackberry-app-to-catch-up-with-ios-and-android/.

[20] A. Souppouris. *WhatsApp now has 350 million monthly active users*. The Verge. Oct. 22, 2013. URL: http://www.theverge.com/2013/10/22/4865328/whatsapp-350-million-monthly-active-users.

[21] N. Olivarez-Giles. *WhatsApp adds voice messaging as it hits 300 million monthly active users*. The Verge. Aug. 6, 2013. URL: http://www.theverge.com/2013/8/6/4595496/whatsapp-300-million-active-users-voice-messaging-update.

[22] W. Inc. *Why we don't sell ads*. TechnoBuffalo. June 18, 2012. URL: http://blog.whatsapp.com/index.php/2012/06/why-we-dont-sell-ads/.

[23] *Amir Sarhangi | CrunchBase Profile*. Crunch Base. Oct. 21, 2013. URL: http://www.crunchbase.com/person/amir-sarhangi.

[24] *Accreditation | RCS*. GSMA. Oct. 27, 2013. URL: http://www.gsma.com/rcs/interoperability-testing/accreditation.

[25] *Nable Communications*. Nable Communications. URL: http://www.nablecomm.com/eng/about/overview.php (visited on 10/23/2013).

[26] A. Downey. *Nable Communications, Inc. Receives 2012 Communications Solutions Product of the Year Award*. WebRTC World. Sept. 16, 2013. URL: http://www.webrtcworld.com/news/2013/09/16/7413080.htm.

[27] *Neusoft Silta RCS client receives full accreditation from the GSMA*. Neusoft Corporation. Sept. 19, 2012. URL: http://www.neusoft.com/news/html/20120919/2448154020.html.

[28] *Silta RCS clients*. Neusoft Corporation. URL: http://silta.neusoft.com/index.php?page=silta-clients (visited on 10/21/2012).

[29] *Rich Communication Alido Di Giovanni*. Informa Telecoms & Media. Oct. 23, 2013. URL: http://rich-communication.com/speaker/alido-di-giovanni/.

[30] *Summit Tech launches first RCS 5.0 joyn Android app in North America*. Summit Tech. Nov. 1, 2012. URL: http://www.richcommunicationsuite.com/pr/SummitTech_RCS_MetroPCS_Nov1_2012.pdf.

[31] J. Fingas. *Android tops 81 percent of smartphone market share in Q3*. Engadget. Oct. 31, 2013. URL: http://www.engadget.com/2013/10/31/strategy-analytics-q3-2013-phone-share/.

[32] *Box | CrunchBase Profile*. Crunch Base. URL: http://www.crunchbase.com/company/box (visited on 10/25/2013).

[33] *Mobile Apps for Access to Content from Anywhere*. Box. URL: https://www.box.com/personal/mobile-access/ (visited on 10/25/2013).

[34] L. Rao. *Box Partner Network To Expand Cloud And Collaboration Service*. TechCrunch. Feb. 5, 2013. URL: http://techcrunch.com/2013/02/05/15-million-users-strong-box-debuts-new-partner-network-to-expand-cloud-and-collaboration-service/.

[35] *Dropbox | CrunchBase Profile*. Crunch Base. URL: http://www.crunchbase.com/company/dropbox (visited on 10/26/2013).

[36] *Dropbox – Mobile*. Dropbox. URL: https://www.dropbox.com/mobile (visited on 10/26/2013).

[37] C. Warren. *How Dropbox Scaled From 2,000 to 200 Million Users*. Mashable. July 30, 2013. URL: http://mashable.com/2013/07/30/dropbox-scaling/.

[38] *Official Blog: Introducing Google Drive... yes, really*. Google. URL: http://googleblog.blogspot.in/2012/04/introducing-google-drive-yes-really.html (visited on 10/26/2013).

[39] L. Gannes. *With 120M Users, Google Drive Gets Tighter Integration With Gmail*. All Things Digital. Nov. 12, 2013. URL: http://allthingsd.com/20131112/with-120m-users-google-drive-gets-tighter-integration-with-gmail/.

[40] *Elaborate claims about new Dotcom site*. 3 News. URL: http://www.3news.co.nz/Elaborate-claims-about-new-Dotcom-site/tabid/412/articleID/275057/Default.aspx (visited on 10/26/2013).

[41] *MEGA*. Mega. May 26, 2013. URL: https://mega.co.nz/#privacycompany.

[42] Ernesto. *Mega Exits Beta With 5 Million Users and Big Plans*. TorrentFreak. Nov. 7, 2013. URL: http://torrentfreak.com/mega-exits-beta-with-5-million-users-and-big-plans-131107/.

[43] M. Joire. *Microsoft gives students 3GB additional Skydrive storage for one year*. Engadget. May 26, 2013. URL: http://www.engadget.com/2013/05/26/microsoft-gives-students-3gb-additional-skydrive-storage-for-one/.

[44] *Microsoft SkyDrive | CrunchBase Profile*. Crunch Base. URL: http://www.crunchbase.com/product/windows-live-skydrive (visited on 10/27/2013).

[45] *500 Million Personal Cloud Storage Subscriptions in 2012*. StorageNewsletter.com. Sept. 7, 2012. URL: http://www.storagenewsletter.com/rubriques/market-reportsresearch/ihs-cloud-storage-services/.

[46] L. Horn. *You Can Access Dropbox From Inside Yahoo Mail Now*. Gizmodo. Feb. 4, 2013. URL: http://gizmodo.com/5993321/you-can-access-dropbox-from-inside-yahoo-mail-now.

[47] *India Leads Worldwide Social Networking Growth*. eMarketer. Nov. 19, 2013. URL: http://www.emarketer.com/Article/India-Leads-Worldwide-Social-Networking-Growth/1010396.

[48] *Facebook | CrunchBase Profile*. Crunch Base. URL: http://www.crunchbase.com/company/facebook (visited on 10/08/2013).

[49] *Key Facts - Facebook's latest news, announcements and media resources*. Facebook. URL: http://newsroom.fb.com/Key-Facts (visited on 10/08/2013).

[50] M. Kaste. *Facebook's Newest Challenger: Google Plus*. NPR. June 29, 2011. URL: http://www.npr.org/2011/06/29/137507567/facebooks-newest-challenger-google-plus.

[51] *Latest Social Media facts, figures and statistics 2013*. Digital Insights. Sept. 4, 2013. URL: http://blog.digitalinsights.in/social-media-facts-and-statistics-2013/0560387.html.

[52] *Instagram | CrunchBase Profile*. Crunch Base. URL: http://www.crunchbase.com/company/instagram (visited on 10/29/2013).

[53] H. Hinseth. *The History of Pinterest*. Jan. 17, 2013. URL: http://thehistoryofpinterest.blogspot.pt/.

[54] A. MacArthur. *The Real History of Twitter, In Brief*. About.com. URL: http://twitter.about.com/od/Twitter-Basics/a/The-Real-History-Of-Twitter-In-Brief.htm (visited on 10/29/2013).

[55] J. DeMers. *The Top 7 Social Media Marketing Trends That Will Dominate 2014 - Forbes*. Forbes. URL: http://www.forbes.com/sites/jaysondemers/2013/09/24/the-top-7-social-media-marketing-trends-that-will-dominate-2014/2/ (visited on 09/24/2013).

[56] N. Bilton. *Disruptions: Indiscreet Photos, Glimpsed Then Gone*. NYTimes. May 6, 2012. URL: http://bits.blogs.nytimes.com/2012/05/06/disruptions-indiscreet-photos-glimpsed-then-gone/.

[57] A. Shontell. *Snapchat active users exceed 30 million*. Business Insider. Dec. 9, 2013. URL: http://www.businessinsider.com/snapchat-active-users-exceed-30-million-2013-12.

[58] E. M. Rusli. *Snapchat Spurned 3 Billion Acquisition Offer from Facebook*. The Wall Street Journal. Nov. 13, 2013. URL: http://blogs.wsj.com/digits/2013/11/13/snapchat-spurned-3-billion-acquisition-offer-from-facebook/.

[59] J. J. Roberts. *Confide: an app for execs who want sensitive messages to vanish Snapchat-style*. Gigaom. Jan. 8, 2014. URL: http://gigaom.com/2014/01/08/confide-an-app-for-execs-who-want-sensitive-messages-to-vanish-snapchat-style/.

[60] *Press Release: Glympse*. Glympse. URL: http://glympse.com/news/press/15 (visited on 01/08/2014).

[61] *Learn About Sprint Planning, Daily Standup, and Daily Scrum*. Scrum Alliance. URL: http://www.scrumalliance.org/why-scrum/core-scrum-values-roles (visited on 12/16/2013).

[62] *Learn about sprint artifacts and sprint activities*. Scrum Alliance. URL: http://www.scrumalliance.org/why-scrum/core-scrum-artifacts-activities (visited on 12/16/2013).

[63] *Android Datastore API documentation*. Dropbox. URL: https://www.dropbox.com/developers/datastore/docs/android (visited on 02/24/2014).

[64] *Android Chooser API*. Dropbox. URL: https://www.dropbox.com/developers/dropins/chooser/android (visited on 02/24/2014).

[65] *Sync API for Android*. Dropbox. URL: https://www.dropbox.com/developers/sync/docs/android (visited on 02/24/2014).

[66] *Google Directions API*. Google. URL: https://developers.google.com/maps/documentation/directions/ (visited on 04/29/2014).

[67] *Location*. Android Developers. URL: http://developer.android.com/reference/android/location/Location.html#getAccuracy() (visited on 03/12/2014).

[68] *Android Fragmentation Report July 2013*. OpenSignal. URL: http://opensignal.com/reports/fragmentation-2013/ (visited on 05/12/2014).

[69] *SQLite Query Language: INSERT*. SQLite. URL: http://www.sqlite.org/lang_insert.html (visited on 05/23/2014).