Mestrado em Engenharia Informática 2015/2016

**Estágio**

Relatório Final

# Desenvolvimento de um plug-in para o browser Google Chrome

João André Bugalho Pedro
joaoap@student.dei.uc.pt

Orientador do DEI:
Fernando Barros

Orientador da WIT:
Frederico Lopes

Data: 1 de Julho de 2016

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA


**Departamento de Engenharia Informática**
Faculdade de Ciências e Tecnologia
Universidade de Coimbra
Pólo II, Pinhal de Marrocos, 3030-290 Coimbra

Tel: +351239790000 | Fax: +351239701266 | **info@dei.uc.pt**


**WIT Software, S.A.**
Centro de Empresas de Taveiro
Estrada de Condeixa, 3045-508 Taveiro, Coimbra

Tel: +351239801030 | Fax: +351239801039 | **info@wit-software.com**

Estagiário:
João André Bugalho Pedro
joaoap@student.dei.uc.pt

Orientador do DEI:
Fernando Barros
barros@dei.uc.pt

Júris:
Luís Macedo
macedo@dei.uc.pt
Joel Arrais
jpa@dei.uc.pt

Orientador da WIT:
Frederico Lopes
frederico.lopes@wit-software.com

# Resumo

Com a chegada em massa de *smartphones* e a possibilidade de acesso fácil à internet através dos mesmos, as aplicações OTT (*Over-The-Top*) têm vindo a ser cada vez mais utilizadas por um maior número de pessoas. Aplicações como o Facebook Messenger ou o Skype permitem a qualquer pessoa com ligação à internet comunicar com outra em qualquer canto do mundo, de forma gratuita.
Devido a este fenómeno, as operadoras móveis têm vindo a ter receitas cada vez mais baixas, tornando as aplicações OTT numa forte ameaça.

Tendo em vista a mitigação deste problema, a GSM Association criou o RCS (Rich Communication Service), que consiste numa especificação de um conjunto de funcionalidades e serviços feita com a concordância quer de várias operadoras móveis, quer de companhias relacionadas com suporte, desenvolvimento, promoção e normalização do GSM (Global System for Mobile Communication).

O objetivo deste projeto é apresentar ao cliente uma aplicação RCS que sirva de alternativa ao telemóvel, de fácil e rápido acesso, dentro do computador. A aplicação é iniciada com o sistema operativo e tem todas as principais funcionalidades de um telemóvel como: lista de contactos, envio de mensagens (*chat* e SMS), chamadas (vídeo e voz) e partilha de ficheiros.

# Palavras-chave

"Rich Communication Service", "Global System Mobile Association", "Chrome App", "Telecomunicações", "Aplicações Over-The-Top", "WebRTC"

# Abstract

With the massive arrival of smartphones and the possibility of easy access to internet, the OTT applications (Over-The-Top) have been increasingly used. Applications such as Facebook Messenger or Skype allow anyone with an Internet connection to communicate with another person in any other place in the world, for free.
Because of this, mobile operators revenues have been falling, making OTT applications a real threat.

In order to minimize this problem, GSM Association created RCS (Rich Communication Service). RCS is a specification of a set of features and services made on the agreement of several mobile operators and companies related with the support, development, promotion and standardization of GSM (Global System for Mobile Communication).

The goal of this project is to give the customer a RCS application that serves a fast and easy access alternative to the mobile phone. The application starts as soon as the operating system starts and has all the main features of a mobile phone such as contacts list, messaging (chat and SMS), calls (video and voice) and file sharing.

# Agradecimentos

Em primeiro lugar quero agradecer à WIT Software por esta oportunidade de estágio.

Quero agradecer bastante aos meus orientadores, Frederico Lopes e Fernando Barros, pela paciência, apoio e dedicação durante estes últimos nove meses em que decorreu o estágio. Quero também agradecer às pessoas dentro da WIT que me ajudaram, em especial ao André Silva, João Alves e Sandrina Silva que, mesmo à distância, estiveram presentes sempre que os problemas teimavam em persistir.

Agradeço aos meus amigos e companheiros de estágio pelos momentos de descontração, mesmo em alturas de grande *stress* e trabalho. Um obrigado muito especial à minha namorada pelo apoio em todos os momentos.

Por fim, quero agradecer à minha família por me terem tornado na pessoa que sou e por todas as oportunidades que deram para ser quem sou.

# Índice

# Índice de Figuras

# Índice de Tabelas

# Glossário

| Termo | Significado |
|---|---|
| **API** | Uma *Application Progamming Interface* designa-se por um conjunto de métodos de um *software* que é usado por terceiros, de forma a utilizar o software como um serviço. |
| **DoD** | *Definitio of Done* trata-se de um documento que visa estabelecer requisitos a ser findados aquando da conclusão de uma *User Story*. |
| **Emoticons** | *Emoticon* é uma representação tipográfica de expressões faciais que transmitem sentimentos. |
| **Emoji** | *Emoji* são pequenas imagens que descrevem situações reais e emoções. |
| **Framework** | O termo *Framework* entende-se por um conjunto de métodos e funcionalidades que auxiliam o desenvolvimento de *software*. |
| **GSMA** | A GSM Association uma associação formada em 1995 por operadoras de comunicações móveis e companhias relacionadas com suporte, desenvolvimento, promoção e normalização do GSM (Global System for Mobile Communication). |
| **Instant Messaging** | *Instant Messaging* trata-se de um tipo de mensagens enviadas pela Internet em tempo real. |
| **OTT** | *Over-The-Top* é um termo que se refere a aplicações que provêm conteúdos multimédia pela Internet. |
| **Product Backlog** | O *Product Backlog* é a compilação de todas as *User Stories* (funcionalidades, bugs e melhorias) de um projeto SCRUM. |
| **Product Owner** | O *Product Owner* é o responsável pela gestão do *Product Backlog* num projeto SCRUM. |
| **RCS** | RCS é uma plataforma que permite o estabelecimento de comunicações para além do SMS e das chamadas de voz, inovando e melhorando esses serviços. |
| **SCRUM** | SCRUM trata-se de uma *framework* de desenvolvimento de *software* ágil. |
| **Sprint** | Em desenvolvimento SCRUM, *sprint* é um período de tempo (normalmente entre 1 a 4 semanas) em que a equipa de desenvolvimento se compromete a realizar determinado número de tarefas do *product backlog*. |
| **Velocidade** | Em desenvolvimento SCRUM, a velocidade de uma equipa é traduzida pela soma dos pontos concluídos num *sprint*. A velocidade é depois usada para planear os novos *sprints*. |

# Acrónimos

| Acrónimo | Descrição |
|---|---|
| ACS | Auto Configuration Server |
| AJAX | Asynchronous Javascript and XML |
| API | Application Programming Interface |
| CSS | Cascading Style Sheets |
| DoD | Definition of Done |
| DOM | Document Object Model |
| GSMA | Groupe Speciale Mobile Association |
| GSM | Global System for Mobile Communication |
| HD | High Definition |
| HTML | HyperText Markup Language |
| IM | Instant Message |
| IMS | IP Multimedia Subsystem |
| IMDN | Instant Message Disposition Notification |
| IP | Internet Protocol |
| JSON | JavaScript Object Notation |
| MVC | Model-View-Controller |
| MVP | Minimum Viable Prototype |
| NAB | Network Address Book |
| ORTC | Object Real Time Communications |
| OS | Operating System |
| OTP | One Time Password |
| OTT | Over The Top |
| RCS | Rich Communication Service |
| SDK | Software Development Kit |
| SIM | Subscriber Identity Module |
| SIP | Session Initiation Protocol |
| SMS | Short Message Service |
| SPA | Single-Page Application |
| SVN | Subversion |
| UI | User Interface |
| UX | User Experience |
| VoIP | Voice over IP |
| WCAS | WIT Communications Application Server |
| WCS | WIT Communicator Suite |
| WebRTC | Web Real Time Communications |
| WMC | WIT Mobile Communicator |
| WWC | WIT Web Communicator |

# 1. Introdução

No âmbito da realização do estágio integrado no Mestrado em Engenharia informática, do Departamento de Engenharia informática da Universidade de Coimbra (área de Engenharia de software), foi realizado o presente relatório que incide sobre o trabalho desenvolvido na empresa WIT Software.

O estágio está a ser orientado pelo Professor Fernando Barros, professor no Departamento de Engenharia Informática e pelo Engenheiro Frederico Lopes na WIT Software.

A WIT Software foi fundada em 2001, como uma *spin-off* da Universidade de Coimbra. Desde então tem vindo a especializar-se em soluções para operadoras de telecomunicações, tendo como clientes alguns dos líderes de mercado desta área, como por exemplo a Vodafone, Deutsche Telekom, Unitel, Telefónica ou Orange.

Este capítulo está organizado em 4 secções. A primeira contextualiza este trabalho na área das telecomunicações. A segunda explica o que motivou o interesse da WIT Software neste trabalho. A terceira contém os objetivos estágio, e, por fim, a quarta secção faz uma breve descrição dos capítulos seguintes e dos apêndices deste documento.

## 1.1. Contexto

A troca de mensagens de texto via *Short Message Service* (SMS) foi, entre 2000 e 2010, a maior fonte de lucro para as operadoras móveis, tendo um crescimento elevado e contínuo. No entanto, com a entrada dos *smartphones* em grande escala no mercado, e o acesso fácil e rápido à internet nestes dispositivos, a área das telecomunicações tem ficado, de ano para ano, cada vez mais competitiva.

A chegada em massa dos *smartphones* ao mercado, e o serviço de internet acessível à grande maioria dos utilizadores, causou uma revolução na forma como as pessoas comunicam. A criação de aplicações *Over-The-Top* (OTT) de comunicação aumentou, assim como a sua utilização. Consequentemente tem sido enviado um elevado número de *Instant Messages* pelo serviço de internet através das várias OTT existentes, ficando as operadoras móveis com lucros mais reduzidos com o envio de SMS's.

Na Figura 1 estão representados dados estatísticos que comprovam o avanço das *Instant Messages* sobre as SMS.

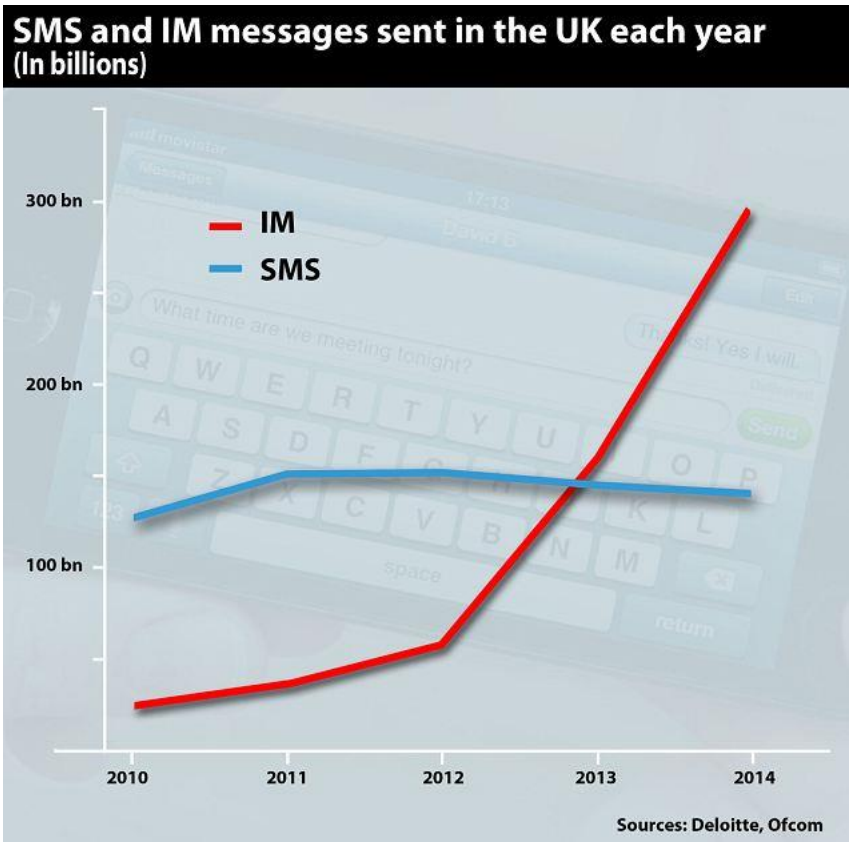



*Figura 1- Número de SMS enviadas de 2010 a 2014 no Reino Unido [1]*

Esta evolução pode ser constatada em inúmeras fontes bibliográficas, seguindo-se um exemplo de uma publicação de Novembro de 2015 feita por *Marck Zuckerberg* (Figura 2), dono e fundador da empresa Facebook, sobre alguns números das aplicações geridas pela empresa.


*Figura 2 - Número de utilizadores de várias aplicações que pertencem à empresa Facebook [2]*

Para além destas, existem muitas outras aplicações como o Skype, LINE, Viber ou Google Hangouts com elevadíssimos números de utilizadores mensais e de *Instant Messages* enviadas [3].

### 1.1.1. RCS

De forma a ajudar a combater esta tendência foi desenvolvido o *Rich Communications Service* (RCS) que foi criado pela *Groupe Speciale Mobile Association* (GSMA). Esta associação foi formada em 1995 por operadoras de comunicações móveis e companhias relacionadas com suporte, desenvolvimento, promoção e normalização do GSM (Global System for Mobile Communication).

O RCS é uma plataforma que permite o estabelecimento de comunicações para além do SMS e das chamadas de voz, oferecendo serviços como *chat*, partilha de ficheiros ou mensagens de grupo. São permitidas comunicações tanto pela rede Wi-Fi como pela rede celular das operadoras móveis.

O objetivo da criação do RCS é a evolução de funcionalidades *mainstream* como SMS e chamadas de voz para tecnologias IP (Internet Protocol) permitindo a utilização de serviços diferentes ao mesmo tempo (por exemplo, enviar uma SMS durante uma chamada de voz ou vídeo) e facilitando a conexão entre dispositivos diferentes (telemóvel, *tablet*, PC, etc…).

**Principais funcionalidades:**

- Lista de contactos melhorada: informações sobre presença, capacidade do serviço (*capabilities*, forma de comunicação disponível).
- Sistema de mensagens melhorado: maior variedade de tipos de mensagem (SMS, chat, multimédia, grupos de mensagens), partilha / transferência de ficheiros, *emoticons*, partilha de localização.
- Chamadas melhoradas: partilha de conteúdo multimédia durante uma chamada (voz ou vídeo).

Uma das grandes conquistas do RCS foi conseguir pôr as operadoras a trabalharem em conjunto no combate às aplicações OTT, concordando com definições e especificações comuns, sendo possível ter um cliente RCS nativo e de fácil utilização. A criação do RCS contou com várias empresas de renome mundial como a Orange, Telecom Italia, Telefónica, TeliaSonea, Ericsson, Nokia Siemens Networks, Nokia, Sony Ericsson, Samsung e WIT Software.

## 1.2. Motivação

Como foi analisado anteriormente, as operadoras de telecomunicações apostam na evolução e enriquecimento dos serviços como o SMS e as chamadas de voz. Dado isto, a WIT Software viu uma boa oportunidade de desenvolver uma aplicação que se destacasse pela facilidade e rapidez de acesso por qualquer dispositivo com um *browser* instalado. O browser

escolhido foi o *Google Chrome*, que é também o browser mais utilizado em todo o mundo (71,4% de utilização, registada em Maio de 2016, seguindo-se do *Firefox* com 16,9%), e a aplicação será desenvolvida como uma *Google Chrome App* [4].

Com esta aplicação será possível o utilizador fazer comunicações imediatamente após ligar o computador, sendo apenas necessário o seu número de identificação móvel para uma primeira configuração. Depois não precisará mais de recorrer ao telemóvel para fazer qualquer tipo de comunicações. Esta aplicação está inserida dentro dos produtos da WIT Communication Suite.

## 1.3. Objetivos

Os objetivos do estagiário centram-se na aplicação e consolidação de todos os conhecimentos adquiridos ao longo do seu percurso académico. Não só competências técnicas são esperadas, mas também as competências pessoais desenvolvidas.

A empresa tem como principal objetivo a conclusão deste projeto com sucesso, isto é, as *features* apresentadas à Universidade devem ser desenvolvidas com qualidade e atempadamente. Para tal a empresa fornece todos os elementos necessários, seja equipamento ou formação. Outro objetivo da empresa é a integração dos estagiários dentro da comunidade da empresa, assim como nas equipas de desenvolvimento referentes a cada projeto. Desta forma pretende-se que haja interação com os colegas, minimizando assim o tempo de resolução de dúvidas ou problemas no decorrer do estágio.

Tecnicamente o estagiário terá de desenvolver uma aplicação Web capaz de ter uma lista de contactos sincronizada com a lista presente no seu dispositivo móvel, enviar mensagens SMS e Chat, fazer chamadas Voice over IP (VoIP) e GSM, entre outras funcionalidades indispensáveis a uma aplicação de comunicação.

Antes de começar a ser desenvolvido todo este conjunto de tarefas, é necessário planear todo o trabalho, dando prioridade às seguintes tarefas:

- Estudo das tecnologias – é necessário adquirir conhecimentos sobre as tecnologias e possíveis ferramentas a serem usadas.
- Análise de requisitos – é necessário identificar, avaliar e priorizar os requisitos.
- Definição da arquitetura – a forma como todos os requisitos vão comunicar entre si tem que ser definida antes de começar o desenvolvimento.

## 1.4. Estrutura do documento

De modo a facilitar a navegação ao leitor neste documento, esta secção foi criada e descreve brevemente as secções que se seguem:

- **2. Estado da arte** – Nesta secção está presente a pesquisa e estudo feito sobre os produtos já existentes na área de telecomunicação, concorrentes ao que vai ser

desenvolvido. Também uma análise às alternativas de tecnologias e *frameworks* está descrita.

- **3. Abordagem** – Esta secção contém explicada a metodologia utilizada assim como o planeamento elaborado durante o estágio.
- **4. Arquitetura** – Nesta secção é descrita a arquitetura seguida e todas as decisões de desenvolvimento tomadas.
- **5. Desafios na implementação** – Nesta seção são evidenciados os maiores desafios durante a implementação e como foram resolvidos.
- **6. Testes e casos de uso** – Nesta secção são apresentados os testes desenvolvidos e também os casos de uso das funcionalidades implementadas.
- **7. Conclusão** – Nesta secção é escrita uma revisão geral sobre todo o trabalho desenvolvido assim como algumas sugestões e trabalhos futuros.

Este documento é complementado com apêndices que contêm informação essencial, mais detalhada e confidencial para a empresa. Os apêndices são os seguintes:

- **Apêndice A**: Abordagem – explicação mais detalhada sobre a metodologia seguida. Está também presente o *product backlog* do projeto.
- **Apêndice B**: Arquitetura – descrição mais detalhada sobre a arquitetura dos componentes mais sensíveis em termos de confidencialidade.
- **Apêndice C**: Casos de uso – este anexo contém os casos de uso das funcionalidades implementadas.
- **Apêndice D**: Testes – este anexo contem todos os testes realizados.

# 2. Estado da Arte

Nesta secção será descrita a análise feita em relação aos competidores, e assim criada uma forma de determinar a viabilidade do protótipo final criado.

Os competidores estão divididos em duas subsecções: os competidores diretos da aplicação a ser desenvolvida (*Chrome Apps* de comunicação) e os competidores ao cliente que poderá interessar-se pela aplicação criada (neste caso as aplicações OTT, como WhatsApp ou Skype, são os competidores das operadoras de telecomunicações, que por sua vez são clientes da WIT Software). No entanto são feitas outras considerações nomeadamente o facto das tecnologias e ferramentas/*frameworks* utilizadas também terem concorrentes.

## 2.1. Tecnologia

O protocolo de comunicação utilizado é o RCS e tem como mais relevante concorrente o MQTT (MQ Telemetry Transport).

- **MQTT**

O MQTT foi criado pelo Dr Andy Stanford-Clark da IBM, e por Arlen Nipper da Arcom (agora Eurotech), em 1999 [5]. É um protocolo de troca de mensagens do tipo *publish/subscribe*, bastante simples e leve, construído para operar em redes com baixa banda de transmissão e com elevada latência. Os principais objetivos deste protocolo são a redução do consumo da largura de banda assim como a redução do consumo de bateria dos dispositivos. No entanto pretende ser um protocolo bastante fiável e que assegure a entrega das mensagens [6].

MQTT é também o protocolo usado pelo Facebook Messenger para fazer a entrega das suas mensagens, tendo sido escolhido pelas suas qualidades de poupança de bateria dos dispositivos, baixo uso da largura de banda da rede e pela entrega rápida e segura das mensagens [7].

- **ORTC**

A *Software Development Kit* (SDK) da WIT faz comunicação com a Gateway recorrendo à tecnologia *Web Real Time Communications* (WebRTC).

Recentemente começou a ser introduzida a tecnologia Object Real Time Communications (ORTC), em termos funcionais muito parecida com WebRTC. O novo *browser* da Microsoft, Edge, já permite a utilização desta *Application Programming Interface* (API) [8].

De acordo com os engenheiros que estão a frente do projeto, o WebRTC fornece uma API de demasiado alto nível, retirando assim algum controlo sobre as comunicações aos programadores [9].

## 2.2. Competidores

Nesta análise é tido em conta um padrão, que pode variar consoante as informações disponibilizadas pelas empresas que detêm estas aplicações. São indicadas informações gerais da aplicação (tipo de aplicação, número de utilizadores), um pouco da sua história, o modelo de negócio e ainda as principais funcionalidades.

### 2.2.1. Competidores da aplicação

Nesta secção são analisados competidores diretos, *Google Chrome Apps* que tenham funcionalidades e finalidades semelhantes ao protótipo final a ser desenvolvido no âmbito do estágio.

- **VideoLink2me**

**Chrome App com 16.668 utilizadores (*chrome web store*)** [10]

VideoLink2me é uma aplicação de videoconferência criada em Amsterdão em 2013 por Konstantin Goncharuk e Galyna Dunaievska [11]. Esta aplicação oferece chamadas de vídeo P2P (*peer-to-peer*) com a possibilidade de trocar mensagens de texto e de partilha de ecrã. Tem como tecnologia base o WebRTC, no entanto muda para Flash quando o WebRTC não é suportado. As chamadas suportam 5 pessoas no máximo e a aplicação já se encontra disponível em 8 línguas diferentes (Inglês, Russo, Espanhol, Francês, Italiano, Alemão, Holandês e Chines) [12].

A aplicação é grátis, no entanto disponibiliza uma versão *premium* (4.99€ mês ou 59,88€ ano) [13]:

- Sem publicidade
- Modo *High Definition* (HD) disponível
- Partilha de ficheiros ilimitada e demostração de fotos em tempo real

Principais funcionalidades:

- Troca de *Instant Messages* (IM)
- *Emojis*
- Mensagens de vídeo (vídeo para utilizadores offline)
- Chamadas de voz e vídeo por IP
- Partilhar de ficheiros
- Demonstração de fotos (os participantes da chamada podem ver imagens e fotografias juntos durante a chamada)
- Partilha de ecrã
- *Links* persistentes (sala pessoal para chamadas de vídeo com um URL único na internet)
- Informação pessoal de conta (avatar, frase de apresentação, etc.)

- **Google+ Hangouts:**

**Chrome App 4.776.712 utilizadores (*chrome web store*)** [14]

O Google Hangouts foi criado em 2013 e é o produto final da combinação de 3 serviços do google: google video, google messenger e google talk [15]. Utiliza a tecnologia WebRTC para as comunicações (tanto voz como vídeo), dispensando assim a instalação de *plug-ins* de terceiros [16] .

O Google possibilita a criação de Apps Hangouts, ou seja, aplicações que usam a API Hangouts de forma a ligar os clientes Hangouts dentro de aplicações de terceiros (por exemplo: partilha de scores num jogo entre dois clientes Hangouts). As Apps Hangouts são desenvolvidas nos modelos *standard* de HTML, *JavaScript* e *Cascade Style Sheet* (CSS). [17]
O Google Hangouts permite [18]:

- Enviar mensagens (individuais ou em grupo)
- Enviar SMS
- Chamadas (vídeo ou apenas voz)
- Partilha de fotos, localização e *stickers*

Requisitos de utilização:

- Conta google
- Instalar plug-in Hangouts no browser

### 2.2.2. Competidores dos operadores

Nesta secção são analisados competidores dos operadores. As operadoras de telecomunicações são os clientes da WIT Software e os seus grandes competidores são as aplicações OTT de comunicações.

- **Facebook Messenger**

Facebook Messenger é uma aplicação de comunicação pela internet desenvolvida pelo Facebook em 2011. Com esta aplicação os utilizadores da rede social podem comunicar com amigos sem terem que fazer login no *site* [19].

A aplicação começou por ser implementada primeiramente para dispositivos móveis (Android, iOS e BlackBerry OS). Houve versões para Windows (desktop) e um *plug-in* para o browser Firefox, no entanto em 2014 ambas foram descontinuadas. Ainda no ano de 2014 foi lançada uma App para Windows Phone 8 e uma versão nativa para iPad.

Em Agosto de 2014 a funcionalidade de envio de mensagens foi totalmente desativada da aplicação do Facebook para dispositivos móveis, sendo os utilizadores obrigados a instalar o Facebook Messenger. Esta decisão foi justificada pelo CEO do Facebook (Mark Zuckerberg) como uma decisão que levaria uma melhor experiência de utilização: "Ten billion messages

are sent per day, but in order to get to it you had to wait for the app to load and go to a separate tab" [20]. Recentemente foi colocado *online* uma versão *web* do Facebook Messenger, sem ser necessário entrar no site do Facebook [21].

Estima-se que esta aplicação tenha ultrapassado a marca dos 600 milhões de utilizadores espalhados pelo mundo [22].

A utilização do Facebook Messenger é totalmente gratuita.

Principais funcionalidades [23]:

- Envio de mensagens (IM)
- Partilha de localização
- Partilha de ficheiros (fotos, documentos, músicas, etc.)
- *Emojis*, *stickers*, *gifs*
- Capacidade de instalar aplicações auxiliares (plataforma de aplicações)
- Chamadas de voz e vídeo sobre IP
- Sincronização de contactos
- *Group Chat*
- Notificações
- Edição de informação pessoal
- Mensagem de voz
- Envio instantâneo de fotos (câmara integrada na aplicação)

- **Skype**

Skype é uma aplicação de comunicação, lançada pela primeira vez em 2003 por Dane Janus Friis e Swede Niklas Zennström. Em 2005 foi comprado pelo eBay e em Maio de 2011 foi adquirida pela Microsoft, num negócio que rondou os 8,5 biliões de dólares. Em Agosto de 2015, o Skype registava cerca de 300 milhões de utilizadores *online* por mês [24].

Aquando da entrada para a Microsoft, o Skype ditou o fim do Windows Live Messenger (também conhecido por MSN), uma antiga aplicação de comunicação da Microsoft. O MSN esteve ativo cerca de 14 anos e chegou a ter mais de 300 milhões de utilizadores ativos por mês.

Neste momento o Skype está disponível para sistemas móveis Android, iOS e Windows Mobile; plataformas *desktop* Windows, Mac OS X e Linux; e recentemente foi lançada uma plataforma web onde é possível realizar todas as ações disponíveis nas versões para mobile e desktop (Skype Web precisa da instalação de plug-ins para funcionarem as chamadas de vídeo e voz).

O Skype é uma aplicação gratuita, no entanto tem uma versão *premium* que permite realizar videoconferências, chamadas para dispositivos móveis (*SIM card target*) dando ainda acesso a *Live Customer Support* [25].

Principais funcionalidades:
- Troca de mensagens (IM)
- SMS
- Group chat
- Videoconferência (acessível com conta *premium* apenas)
- Chamada de voz e vídeo sobre IP
- Partilha de ecrã
- Envio de ficheiros (Fotos, documentos, vídeos, etc.)
- Alterar informação pessoal (Nome, foto, informações de presença, etc.)

- **WhatsApp**

WhatsApp é uma aplicação multiplataforma de *messaging* criada idealmente para dispositivos móveis. WhatsApp Inc. foi formada em 2009 (Califórnia, USA) por Brian Acton e Jan Koum, ex-trabalhadores da *Yahoo!*. Em Junho de 2013 a aplicação atingiu a marca de 250 milhões de utilizadores, em Fevereiro de 2014 foi comprada pelo Facebook por aproximadamente 19 biliões de dólares e em 2015 registou a marca de 800 milhões de utilizadores [26].

Está disponível para as plataformas móveis Android, BlackBerry, iOS, Symbian, Windows Phone e Nokia. Em janeiro de 2015 o WhatsApp lançou uma versão *web* [27] possível de ser executada em *browsers* como o Google Chrome, Firefox, Opera e Safari [28].

A utilização da aplicação é gratuita no primeiro ano, sendo cobrado 0,99 dólares anualmente após esse período. WhatsApp dotou uma regra na empresa em que se recusa a utilizar publicidade na sua aplicação de forma a oferecer uma melhor experiencia ao utilizador [29].

Principais funcionalidades [30]:
- Mensagens de texto
- *Emojis*
- Envio de ficheiros (foto, vídeo, documentos, etc.)
- Mensagem de voz
- Chamadas de voz
- *Group Chat*
- Modificar informações pessoais (frase de estado, nome na aplicação)

- **LINE**


**Chrome App com 696.932 utilizadores (*chrome web store*)** [31]

Esta aplicação foi proposta à NHN (New Humam Network) por Lee Hae Jin, um dos gestores da empresa. O primeiro lançamento aconteceu no Japão em 2011 chegando aos 100 milhões de utilizadores em apenas 18

meses. Em 2013 foi considerada a maior rede social do Japão. É utilizado em mais de 230 países diferentes.

Começou por ser desenvolvido apenas para dispositivos móveis (android e iOS), no entanto neste momento já existem versões para BlackBerry, Nokia Asha, Windows Phone, Firefox OS, para dispositivos desktop (Windows e Mac OS) e aplicação para o *browser* Google Chrome (disponível nas versões para Chrome OS, Windows, Mac e Linux) [32].

A utilização da aplicação é grátis: "*Free Messaging - Wherever, Whenever*".

Principais funcionalidades:

- *Stickers*
- Partilha de ficheiros (com tamanho até 1GB)
- Troca de mensagens (IM, entre utilizadores LINE)
- *Group chat*
- LINE *memos* (espaço para tirar notas com a capacidade de anexar fotos)
- Captura de ecrã (pode ser anexado a um *memo* ou enviado para um contacto por mensagem)
- Mensagens de voz e vídeo
- Partilha de contactos e localização
- Rascunhos de mensagens são guardados enquanto não forem enviados

## Análise comparativa:

Segue-se uma tabela com a análise das aplicações supramencionadas.

| | **Messenger** | **Skype** | **WhatsApp** | **Line** | **WWC** |
|---|---|---|---|---|---|
| Necessita de criação de conta | ✓ | ✓ | | ✓ | |
| Lista de contactos remota | ✓ | ✓ | ✓ | ✓ | ✓ |
| *Chat* | ✓ | ✓ | ✓ | ✓ | ✓ |
| Notificação de estado da mensagem (em *Chat*) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Notificação *isTyping* (em *Chat*) | ✓ | ✓ | ✓ | ✓ | ✓ |
| SMS | ✓ | ✓ | | | ✓ |
| *Group Chat* | ✓ | ✓ | ✓ | ✓ | ✓ |

| | | | | | |
|---|---|---|---|---|---|
| Notificação de estado da mensagem (em Group Chat) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Notificação *isTyping* (em Group Chat) | ✓ | ✓ | ✓ | | ✓ |
| Partilha de localização | ✓ | | ✓ | ✓ | ✓ |
| Partilha de contactos | | ✓ | ✓ | ✓ | ✓ |
| Chamadas de voz IP | ✓ | ✓ | ✓ | ✓ | ✓ |
| Chamadas de vídeo IP | ✓ | ✓ | | ✓ | ✓ |
| Mensagens de voz | ✓ | ✓ | ✓ | ✓ | ✓ |
| Mensagens de vídeo | ✓ | ✓ | ✓ | ✓ | ✓ |
| *Stickers* | ✓ | ✓ | | ✓ | ✓ |
| *Emoticons* | ✓ | ✓ | ✓ | ✓ | ✓ |
| *Emojis* | ✓ | ✓ | ✓ | ✓ | ✓ |
| Estado do contacto (*online, offline*) | ✓ | ✓ | ✓ | | ✓ |
| Imagem de perfil | ✓ | ✓ | ✓ | ✓ | ✓ |
| Partilha de ficheiros | ✓ | ✓ | ✓ | ✓ | ✓ |

*Tabela 1 - Análise comparativa dos competidores*

Esta análise permitiu ao autor conseguir criar uma lista de requisitos desejáveis bastante completa.
As aplicações mais completas são o Skype e o Facebook Messenger, falhando numa funcionalidade apenas: partilha de localização e partilha de contacto, respetivamente. Pela análise feita, conclui-se que o Line tem uma grande falha (em relação a outras aplicações) pois não permite aos utilizadores saberem o estado dos seus contactos (*online* ou *offline*). O WhatsApp, embora tenha um processo de login facilitado (apenas necessário o numero do utilizador), não permite fazer qualquer tipo de comunicação para utilizadores que não tenham WhatsApp instalado (apenas dispõe de comunicações IP, não envia SMS ou faz chamadas GSM).

Visto isto percebe-se que as aplicações analisadas que dominam o mercado estão bastante evoluídas, no entanto é possível ainda melhorar alguns aspetos. Começando pelo início de sessão, a aplicação desenvolvida pelo autor não precisará de qualquer tipo de criação de conta

em nenhum serviço, apenas será necessária a utilização do número de contacto. Outra grande vantagem da aplicação é a possibilidade de fazer comunicações através da rede dos operadores móveis (mesmo que os contactos não possuam uma aplicação RCS), seja SMS ou chamadas.

## 2.3. Frameworks

Nesta secção são descritas as *frameworks* e bibliotecas web analisadas para escolha e utilização no desenvolvimento do protótipo.

- **jQuery**


jQuery é uma biblioteca leve e de rápida execução para *JavaScript*. O seu principal objetivo é tornar o desenvolvimento de código *JavaScript* mais fácil e rápido ao programador [33]. Esta biblioteca é suportada pelos mais utilizados *browsers* em todo o mundo, como o Google Chrome, Firefox, Opera, Internet Explorer, Edge ou Safari [34]. Pode ser utilizada em conjunto com outras bibliotecas ou *frameworks* como o AngularJS ou o Bootstrap. O seu primeiro lançamento aconteceu em Dezembro de 2006 por John Resig.

"*jQuery, write less, do more.*" [35] Frase presente no *site* oficial, um exemplo disso é a forma de selecionar um determinado elemento HTML pelo seu ID:

```
//JavaScript
document.getElementById('divTeste')

//jQuery
$('#divTeste')
```

Principais funcionalidades [36]:

- Seleção e manipulação de elementos do DOM
- Eventos (*click, scroll, focus*, etc.)
- Efeitos e animações
- Pedidos AJAX
- *JavaScript Object Notation* (JSON)
- Redução do código
- Manipulação de CSS facilitada
- Enorme comunidade que desenvolve projetos com recurso a esta *framework* (grande suporte online e inúmeros plug-ins desenvolvidos por terceiros que podem ser implementados em qualquer projeto)

- **AngularJS 1**

AngularJS 1 é uma *framework* de JavaScript que tem como principal função auxiliar o desenvolvimento de aplicações *single-page* (aplicações web onde toda a interação acontece sem ser necessário atualizar a página, de forma a oferecer uma utilização muito mais fluida da aplicação ao utilizador) [37]. A sua primeira *release* teve lugar em 2009 e desde então que é mantida pela Google e por uma comunidade enorme de programadores. A mais recente e estável *release* lançada ocorreu em Maio de 2016, versão 1.5.6.

Esta *framework* faz uso da arquitetura *Model-View-Controller* (MVC) de forma que seja criado código com qualidade, organização e rapidez [38].

Principais funcionalidades [39]:

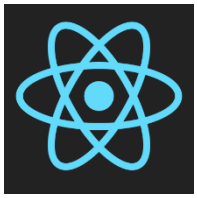
- Data Binding – é uma forma automática de atualizar a *View* (HTML visível ao utilizador) sempre que o *Model* é modificado. Desta forma pode-se considerar que a manipulação direta da DOM pelo programador é eliminado, sendo camuflada por uma funcionalidade simples de usar e eficiente.
- Controllers – controlam todo o comportamento do código que está por trás dos elementos HTML da DOM.
- Diretivas – as diretivas são funcionalidades únicas do AngularJS que permitem o programador criar novos elementos HTML (nova sintaxe) específicos para cada aplicação. Para além disso, as diretivas são completamente reutilizáveis, podendo ser utilizadas em qualquer ponto da aplicação.

AngularJS tem total suporte no desenvolvimento de aplicações e extensões para o *browser* Google Chrome [40].

- **AngularJS 2**

AngularJS 2 é também uma *framework* de JavaScript, evolução do AngularJS 1. No entanto, a sua sintaxe foi remodelada, não sendo possível uma migração de AngularJS 1 para 2 "pacífica" [41]. A primeira *release* foi lançada em Setembro de 2014, não existindo ainda uma versão estável de AngularJS 2 (a ultima *release* foi lançada em Dezembro de 2015 e está em fase beta) [42].
Toda a arquitetura e funcionamento do AngularJS foi remodelada nesta nova versão, tornando-a numa *framework* mais rápida e eficiente [43].

- **React**



React é uma *framework* web, criada por Jordan Walke, um engenheiro que trabalha para o Facebook. A sua primeira grande utilização foi em 2011, no *newsfeed* do Facebook [44].

Uma das grandes *features* do React é a DOM (*Document Object Model*) virtual associado a cada *view*. Quando *view* tem que ser atualizada (atualização nos dados), é gerada uma nova DOM virtual. Assim que completo, o React descobre quais as diferenças entre a DOM virtual antiga e a nova e aplica apenas as alterações indicadas pela comparação [45].

Tendo em conta a análise feita a estas *frameworks* optou-se por se utilizar AngularJS 1. Para além de ser uma *framework* com um nível de maturação elevadíssimo, esta é totalmente compatível com a base de desenvolvimento da aplicação, a Chrome App. Dentro da WIT, enumeras pessoas utilizam AngularJS 1, tendo assim o autor um maior suporte a curta distância. No entanto, também *online*, existe uma enorme comunidade de programadores de AngularJS 1, que respondem a dúvidas e ajudam na resolução de possíveis problemas. Tecnicamente AngularJS 1 é uma boa escolha, pela forma como o código é organizado e pela quantidade de artefactos disponíveis para fazer um desenvolvimento mais rápido e de qualidade.

# 3. Abordagem

A metodologia de desenvolvimento do estágio foi previamente escolhida pela empresa. Nesta secção será explicada a metodologia de desenvolvimento adotada para o estágio. De seguida será descrito todo o planeamento e calendarização feita no início do projeto assim como a comparação entre o planeado e o que foi feito.

## 3.1. Metodologia

Dentro da empresa as equipas têm liberdade de escolherem o tipo de metodologia que mais vai de encontro às necessidades do projeto a desenvolver. As equipas baseiam-se muito em metodologias ágeis, onde é possível ver um contínuo crescimento do *software* a desenvolver e tornar fácil a inclusão de novas funcionalidades, sempre que necessário, ao plano de trabalhos.

Para o desenvolvimento do estágio foi escolhida a metodologia ágil SCRUM. Esta metodologia permite fazer (e atualizar) a priorização das tarefas durante o decorrer do desenvolvimento, assim como introduzir novas funcionalidades.

### 3.1.1. Cargos dos intervenientes

A metodologia SCRUM define três importantes atores no desenvolver de um projeto: *scrum team*, *scrum master* e *product owner*.

- **Scrum Team**

Este ator representa a equipa de desenvolvimento que está encarregue de cumprir as metas estabelecidas na entrega do produto.

- **Scrum master**

O *scrum master* funciona como um facilitador para a *scrum team*. Este ator é responsável por orientar e ajudar a equipa de desenvolvimento. É um facilitador na medida em que desbloqueia a equipa sempre que existem problemas em entregar o produto a tempo, faz a comunicação entre o *product owner* e a equipa.

- **Product owner**

Este ator tem como principais funções priorizar e adicionar tarefas ao *product backlog*. O *product owner* está encarregue de gerir a expectativa e de mostrar o trabalho ao cliente do projeto e aos *stakeholders*.

### 3.1.2. Processo

Para dar início ao SCRUM é necessário que o *product owner* prepare e elabore o ***product backlog***. O *product backlog* pode ser definido como uma lista de requisitos que devem ser

concluídos com o desenvolver do projeto e são descritos na forma de ***user stories***. Uma *user story* trata-se de uma frase que retrata uma ação de forma percetível mesmo a quem não tenha conhecimentos técnicos sobre desenvolvimento de *software*. Depois de criadas, a *scrum team* atribui **pontos** às *user stories* consoante o esforço que estimam que seja necessário a sua conclusão.

Após a definição do *product backlog*, é definido o ***sprint*** pelo *scrum master* e *scrum team*. O *sprint* trata-se de um período de tempo (normalmente entre 1 a 4 semanas) em que a *scrum team* se compromete a realizar determinado número de tarefas do *product backlog*.

As tarefas são adicionadas a cada *sprint* aquando do seu início. São retiradas tarefas do *product backlog* e calculada a soma dos seus pontos. Esse valor denomina-se de **velocidade** e indica o número de pontos que a equipa se propõe a realizar. Após realizados alguns *sprints*, a velocidade começa a tender para um valor constante que resulta em melhores estimativas futuras na criação de novos *sprints*.

A cada dia do *sprint* existe um evento, ***daily scrum*** (*scrum* diário), que consiste numa pequena reunião de 10 a 15 minutos onde o *scrum master* fala com a equipa de forma a saber qual o ponto de situação do trabalho. O principal intuito desta reunião é saber se o trabalho está a decorrer como esperado e se existe algum bloqueio.

Por fim, existe a ***sprint meeting*** (reunião de *sprint*) que ocorre no fim de cada *sprint*. Nesta reunião são mostrados os avanços feitos no projeto em geral e também as tarefas realizadas individualmente no *sprint* em questão. Estão presentes todos os intervenientes do SCRUM (*scrum master*, *scrum team* e *product owner*) assim como os clientes do projeto. Ainda nesta reunião é preparado o novo *sprint* que começa no dia seguinte.

Na Figura 3 está representado um diagrama simplificado do processo SCRUM.

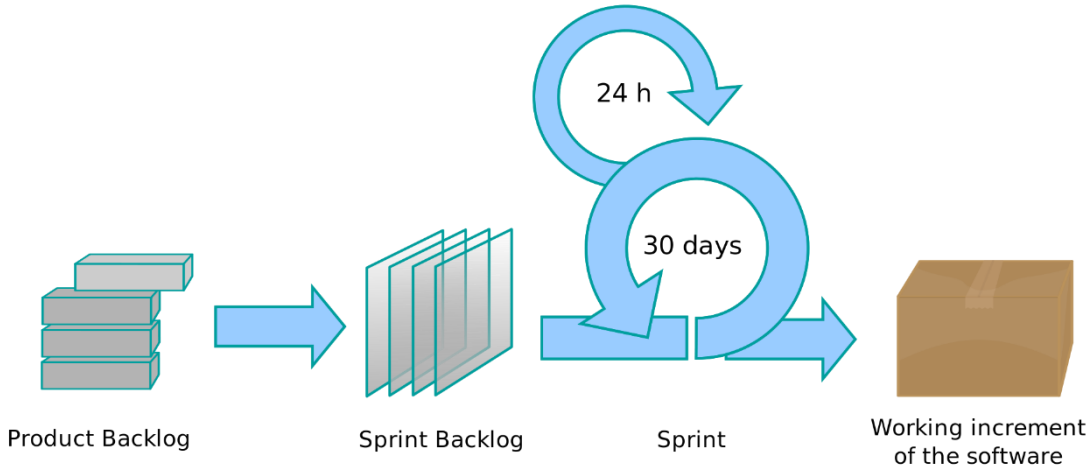



*Figura 3 - Diagrama simplificado do processo da metodologia SCRUM [46]*

### 3.1.3. "Definition of Done"

*Definition of Done* (DoD) – definição de feito – em SCRUM entende-se por uma lista de instruções que devem ser concluídas até o *software* estar realmente terminado. Este artefacto traz transparência e consistência ao trabalho feito por uma equipa. O DoD é definido no início do projeto.

Nesta secção está presente o documento criado pelo autor em conjunto com os restantes estagiários da WIT Software no início do projeto. O DoD criado abrange dois níveis de compromisso: o primeiro está ao nível das *user stories*, onde cada *user story* apenas é marcada como feita após essas condições estarem concluídas; o segundo está ao nível do *sprint*.

#### 3.1.3.1. User Stories

A este nível foram criadas sete instruções que devem estar concluídas para que se possa dar uma *user story* como fechada. As instruções são:

- Implementação
  - Implementação do código completa.
  - Código totalmente integrado no protótipo.
- Testes
  - Teste de aceitação.
  - Testes unitários (apenas quando se revelar necessário).
  - Testes de plataforma (a funcionalidade implementada deve estar funcional em todas as plataformas propostas).
- *Deployment*
  - *Commit* no *Subversion* (SVN) com o número de identificação da *user story* do Redmine.
- Documentação
  - Documentação de código (*JavaDoc* caso se revele necessário ou criação de documento de especificação técnica).

#### 3.1.3.2. Sprint

A este nível foram criadas seis instruções que devem estar concluídas para que se possa considerar um *sprint* como terminado com sucesso. As instruções são:

- Atualizar ***SlideDeck*** (*SlideDeck* trata-se de uma apresentação atualizada com a ultima versão do protótipo com um pequeno resumo das funcionalidades concluídas e alguns casos de uso, o seu objetivo é ter sempre esta apresentação preparada para ser mostrada)

- Especificação técnica criada e atualizada (criação de diagramas, descrição de algum ponto fulcral de determinada funcionalidade, etc.)

- Pacote de instalação preparado

- Aplicação pronta para ser demonstrada
- Versão da aplicação atualizada
- Versão final guardada na respetiva *branch* do SVN (*commit* associado com a tarefa do Redmine), e *tagada*.

## 3.2. Planeamento/Plano de trabalhos

Para este estágio foi adotada uma metodologia ágil e como tal, não foi definido um planeamento rígido para todo o projeto logo de início. O planeamento vai sendo feito de *sprint* em *sprint*.

Começou-se por definir os cargos de cada interveniente no SCRUM:

- ***Scrum Team*** – João Pedro
- ***Scrum Master*** – João Pedro
- ***Product Owner*** – Frederico Lopes

O *product owner* é o orientador do estágio. Como se trata de um estágio, o único membro da equipa é o autor. No entanto é também o autor quem faz a comunicação com o orientador (*product owner*), por isso faz sentido o autor acumular as duas funções. A análise de requisitos foi feita pelo autor, tendo sido depois validada e priorizada pelo *product owner* (analise de requisitos está descrita no **apêndice A**).

Existe ainda outro interveniente, o **tutor**, que é definido no início de cada estágio pela empresa. O tutor é o Engenheiro André Silva da WIT Software e dá auxílio ao estagiário nos aspetos mais técnicos do estágio.

Neste projeto cada *sprint* dura duas semanas e a *sprint meeting* é feita no último dia. Em cada *sprint meeting* estão presentes os estagiários (*scrum master* e *scrum team*), os orientadores (*product owner*) e, sempre que possível, o tutor. Cada estagiário apresenta o que fez durante o *sprint*, executa o demo das funcionalidades implementadas e partilha as dificuldades encontradas durante o *sprint* com os restantes. Os orientadores dão feedback, tanto do resultado geral do *sprint*, como de cada funcionalidade (*user story*) implementada.

Apenas o primeiro mês não foi categorizado como *sprint*, pois durante esse tempo foi feito o estudo das tecnologias, análise de competidores e a análise de requisitos.

Foram concluídos um total de dezassete *sprints* que se enumeram, de seguida, de forma sucinta. Para análise de informação completa e comparação do que foi planeado com o que foi terminado, consultar **Apêndice A**.

- **Sprint #0** – de 05/10/2015 a 27/10/2015

Neste período de tempo todas as tecnologias necessárias ao projeto foram estudadas assim como a análise dos competidos. Elaborou-se o *product backlog* e criaram-se as *user stories*.

- **Sprint #1** – de 27/10/2015 a 10/11/2015

Este foi o primeiro *sprint* oficial. Foram dados os primeiros passos no desenvolvimento, tendo sido criada uma *Chrome App* de teste. Foi iniciado o desenvolvimento dos ecrãs de login. Detetaram-se algumas incompatibilidades entre a API de comunicação a ser usada com a *Chrome App*. Esses problemas foram reportados e registados. Devido a estas incompatibilidades não foi possível implementar totalmente as funcionalidades de login.

- **Sprint #2** – de 10/11/2015 a 24/11/2015

Neste *sprint* foram corrigidos alguns pormenores no design dos ecrãs de login. Para além disso foi possível terminar a implementação dos métodos do login, visto que algumas das incompatibilidades foram resolvidas. No final do *sprint* já era possível visualizar a lista de contactos.

- **Sprint #3** – de 25/11/2015 a 09/12/2015

Durante o *sprint* anterior houve mudanças na *User Interface* (UI) do login, tendo sido agendado para este terceiro sprint uma remodelação na UI de login. Foram desenvolvidas as funcionalidades de procura e remoção de contactos.

- **Sprint #4** – de 09/12/2015 a 22/12/2015

Neste quarto *sprint* foi terminada a gestão de contactos, tendo sido desenvolvidas as funcionalidades de adição e edição de contactos.
Foi iniciado o desenvolvimento das funcionalidades referentes às mensagens (envio, receção, notificações de estado das mensagens).

- **Sprint #5** – de 22/12/2015 a 05/01/2016

Neste *sprint* foram corrigidos alguns problemas encontrados na adição e edição de contactos. Foram implementadas as funcionalidades de envio de SMS e *emoticons*/*emojis*.
Foram detetados problemas com listas de contactos longas que foram devidamente registados e reportados.

- **Sprint #6** – de 05/01/2016 a 19/01/2016

Parte deste *sprint* foi dedicado exclusivamente à escrita do presente documento. Foi criada uma *user story* para correção de *bugs* encontrados.

No início do semestre foi também definido o **MVP** (*Minimum Viable Prototype*). MVP consiste na definição de determinadas tarefas estarem prontas num determinado prazo de tempo. Foi definido que a aplicação teria de trocar mensagens (IM e SMS) no fim do *Sprint* #7 (que vai de dia 19/01/2016 a 02/02/2016). Isso incluía estar todo o trabalho anterior feito, ou seja, todas as *user stories* que aparecem antes desta no *product backlog* têm que estar terminadas. Esta é uma das razões do *product backlog* estar priorizado. O MVP vai ser cumprido e no fim do

*sprint* será decidido se a aplicação está num nível de maturação suficiente para ser apresentada numa das feiras de telecomunicações mais importantes da Mundo, Mobile World Congress (MWC).

- **Sprint #7** – de 19/01/2016 a 02/02/2016

Neste *sprint* foi iniciado um processo de teste e identificação de problemas pelo autor. Os próximos *sprint* foram também dedicados a correção e identificação de problemas. Para além disso, foi implementado a funcionalidade de histórico de mensagens.

- **Sprint #8** – de 02/02/2016 a 12/02/2016

Este *sprint* foi dedicado à resolução de problemas encontrados, como problemas de apresentação de informação (elementos fora do sitio ou com tamanhos errados) ou fluxos inconsistentes na aplicação. Houve ainda lugar para iniciar a implementação da transferência de ficheiros com a implementação da funcionalidade de envio de *stickers*.

- **Sprint #9** – de 12/02/2016 a 01/03/2016

Este *sprint* foi exclusivamente dedicado à resolução de problemas. Foram feitas algumas mudanças nos ecrãs de adição e edição de contactos em relação à inserção do código do país, visto que não existia nenhum campo para esse efeito.

- **Sprint #10** – de 01/03/2016 a 15/03/2016

A partir deste *sprint* voltou-se à implementação de novas funcionalidades, após os últimos três terem sido dedicados quase exclusivamente à correção de problemas que iam sendo encontrados. Durante os *sprints* seguintes continuaram-se a corrigir todos os problemas encontrados, mas numa escala menor.

Neste *sprint* foi implementada a funcionalidade de envio de ficheiros guardados localmente no computador. Também a receção e visualização dos ficheiros, na aplicação, foi implementada.

- **Sprint #11** – de 15/03/2016 a 29/03/2016

Neste *sprint* foi implementada a funcionalidade de gravação de um clipe de áudio instantâneo. Além disso foi desenvolvido um método que reconhece *links* dentro do conteúdo das mensagens de texto e os destaca.

- **Sprint #12** – de 29/03/2016 a 12/04/2016

Neste *sprint* foi implementada a partilha de localização (com recurso ao Google Maps API) e a partilha de contactos (v-cards). Foi revista o *parse* dos números de contacto.

- **Sprint #13** – de 12/04/2016 a 26/04/2016

Neste *sprint* foram implementadas as funcionalidades de envio de fotografia e gravação de vídeo instantâneo. O vídeo, por sua vez, teve que ser alterado para a geração de um GIF animado, de forma a ser compatível com as aplicações RCS da WIT. Foi implementada a funcionalidade de remoção de conversas com um determinado contacto.

- **Sprint #14** – de 26/04/2016 a 10/05/2016

Neste *sprint* foi implementada a funcionalidade para carregar mais mensagens do histórico apenas com o *scroll* para o topo do elemento que contem as mensagens (sem ser necessário clicar explicitamente no botão de *Load More*). Deu-se início à implementação das notificações desktop, neste caso, para as mensagens. Foi também implementado um ecrã de confirmação, aquando da remoção de conversas com um determinado contacto.

- **Sprint #15** – de 10/05/2016 a 24/05/2016

Neste *sprint* foi desenvolvido o menu de configurações. Implementaram-se os ecrãs de gestão das informações de presença do utilizador; configurações sobre mensagens, chamadas e gerais. Foi implementada a funcionalidade de expansão dos *links* encontrados no texto das mensagens.

- **Sprint #16** – de 24/05/2016 a 07/06/2016

Neste sprint foram implementadas as funcionalidades de gestão de chamadas (receber chamadas, iniciar chamadas, gerir eventos da chamada em curso). Foi criado o ecrã de configurações das chamadas e implementadas as suas notificações desktop.

- **Sprint #17** – de 06/06/2016 a 04/07/2016

Ao contrário dos outros *sprints*, este começou um dia antes (por motivos de compatibilidade de horário) e teve a duração de, aproximadamente, um mês. Neste *sprint* foram fechados todos os problemas encontrados até à data e implementadas todas as funcionalidades de gestão de chamadas de vídeo. Também foram desenvolvidas as funcionalidades de colocar uma chamada em espera e desativar o microfone do utilizador. Por fim foram implementados os ecrãs do histórico de chamadas e do *dialer* alfanumérico.

A terceira semana deste *sprint* foi dedicada à escrita e revisão deste documento.

## 3.3. Análise de riscos

Para além de planeamento, é importante antecipar e prever potenciais riscos ao projeto. Dado isto, foi feita uma análise de riscos. Cada risco tem uma descrição, gravidade, probabilidade de acontecer e plano de mitigação. A gravidade e a probabilidade dos riscos são classificadas na seguinte escala: **baixa**, **média**, **alta**, **muito alta**.

- **Utilização de novas tecnologias**

O autor tem que utilizar ferramentas e tecnologias com as quais não está familiarizado (como Chrome Apps e AngularJS). Isto pode levar a que a resolução de eventuais problemas seja muito mais demorada.

Gravidade: Muito alta
Probabilidade: Muito alta
Plano de mitigação: resolução de tutoriais numa fase inicial do projeto para o autor se familiarizar com as tecnologias. O contacto com pessoas da empresa que já tenham trabalhado com estas tecnologias e ferramentas é essencial.

- **Forte competição de mercado**

Tendo em conta o capítulo anterior, o mercado onde este projeto se integra é bastante competitivo e as aplicações existentes já se encontram bem consolidadas na sociedade. Pode ser complicado introduzir um novo produto no mercado.

Gravidade: Alta
Probabilidade: Alta
Plano de mitigação: é necessário estar sempre atualizado sobre as novas funcionalidades implementadas pelos competidores. Desenvolver *software* de qualidade e com características diferenciadoras e de valor. Sessões de *brainstorming* para criação de ideias inovadoras e diferenciadoras dos competidores.

- **Chrome App incompatível com API**

Para o desenvolvimento deste projeto será usada uma Web API criada pela WIT Software. Esta API é suportada na maior parte dos *browsers*, no entanto nunca foi testada numa Chrome App. Podem existir incompatibilidades e o estágio ficar condicionado.

Gravidade: Muito alta
Probabilidade: Alta
Plano de mitigação: estudar alternativas e criar *user stories* dedicadas ao processo de compatibilização entre a Chrome App e a API a usar.

# 4. Arquitetura

Neste capítulo é definida toda a arquitetura referente ao projeto com o objetivo de obter uma implementação com maior qualidade mas também para facilitar o desenvolvimento de trabalhos futuros (pelo próprio ou por outras pessoas).

Este capítulo é constituído por explicações arquiteturais dos seguintes tópicos:

- Arquitetura geral
- WebRTC Gateway SDK
- WWC Chrome App

Por fim, são explicados os módulos implementados do protótipo, com os respetivos casos de uso. Esses módulos são:

- Auto-Start
- Autenticação
- Contactos
- Mensagens
- Chamadas
- Notificações
- Configurações

## 4.1. Arquitetura geral

De uma forma geral, a arquitetura pode ser definida em dois grandes grupos: a aplicação desenvolvida pelo autor (*front-end*) e a Gateway que dá acesso a todas as funcionalidades de comunicação.

A comunicação entre a aplicação e a Gateway é feita com recurso a uma API criada para o desenvolvimento de aplicações web que acedam aos servidores da WIT por terceiros.

A Figura 4 representa a arquitetura da aplicação desenvolvida usando a WebRTC Gateway da WIT.

A Gateway usa uma API de transporte de JSON sobre Websockets para comunicar com os clientes, gerindo a configuração inicial e estabelecendo assim o canal de *signaling*. Será por este canal que todos os pedidos feitos pela API da Gateway virão. A ***Media Gateway*** é responsável por configurar todos os fluxos de chamadas no *browser*. O módulo ***Network Address Book*** permite o acesso a lista de contactos dos clientes a partir de qualquer *browser* ou dispositivo. A **SDK em JavaScript** camufla os detalhes WebRTC e as diferenças entre *browsers*. São utilizados canais *WebSockets* para garantir a comunicação da SDK com a Gateway. A **User Interface** diz respeito à aplicação desenvolvida pelo autor, seguindo as normas de aplicações especificadas pelo RCS.

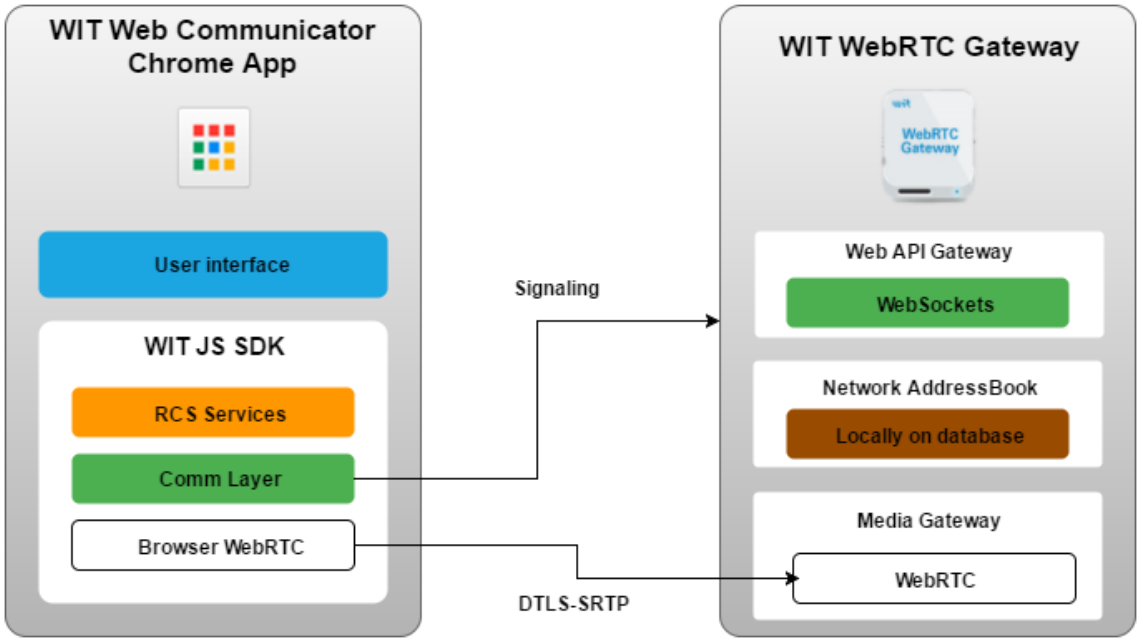



*Figura 4 - Diagrama da arquitetura externa da aplicação*

## 4.2. WebRTC Gateway SDK

*WebRTC Gateway* é um produto que faz parte da arquitetura interna da WIT Software. Abaixo segue a imagem que ilustra essa arquitetura (Figura 5).

Este produto permite comunicações em tempo real entre qualquer *browser* no mercado, sem necessidade de instalar *plugins*. A Gateway usufrui das vantagens da tecnologia WebRTC e fornece a conversão necessária para protocolos Telecom. Também permite comunicações em tempo real de *browsers* para *smartphones*, *tablets*, PC *softphones*, telefones *Session Initiation Protocol* (SIP) e telefones GSM / PSTN. Na Figura 5 está representado um esquema da arquitetura da WIT Communications Suite.

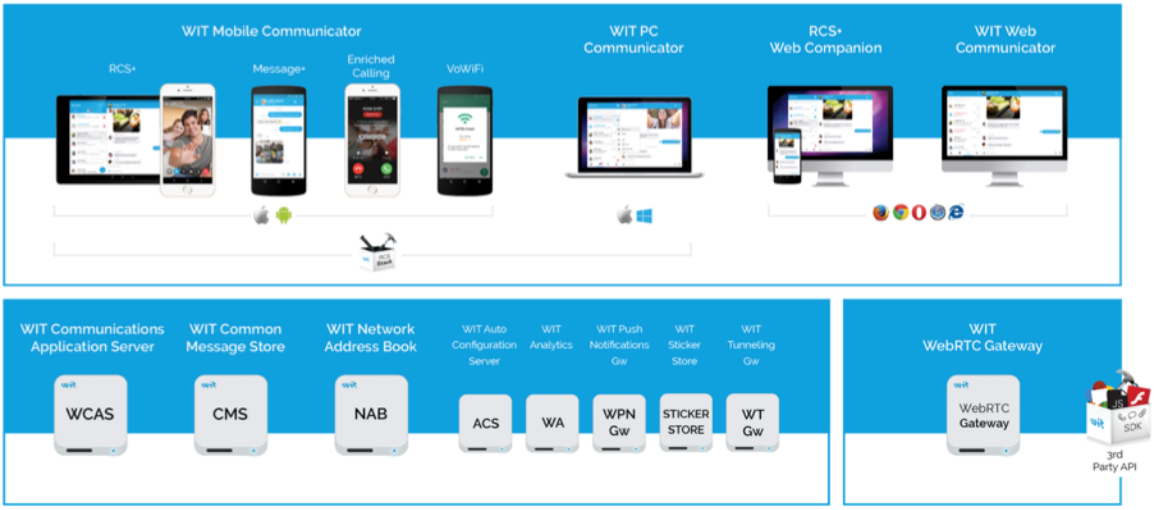



*Figura 5 - Diagrama de produtos da WIT*

Explicando a Figura 5, a WebRTC GW liga-se ao WCAS (WIT *Communications Application Server*) que funciona como uma rede IP Multimedia Subsystem (IMS), conseguindo assim fazer ligação a outros dispositivos. É pelo WCAS que a WebRTC GW consegue ter acesso ao NAB (*Network Address Book*) ao ACS (*Auto Configuration Server*) e ao Message Storage (CMS).

A aplicação desenvolvida integra-se desta forma (Figura 6) nos produtos da WIT Software.


*Figura 6 - Integração da Chrome App na arquitetura da WIT Software*

A plataforma é construída com uma arquitetura hibrida, suportando a mais recente especificação WebRTC, assim como a tecnologia Adobe Flash de forma a ser compatível com os *browsers* que não suportam WebRTC [47].

Este capítulo tem como principal objetivo explicar e apresentar a parte *back-end* do projeto, tendo em vista os módulos e funcionalidades da Gateway SDK da WIT. A Gateway de comunicação utilizada foi totalmente desenvolvida pela WIT e tem 3 principais camadas: camada de Serviço (*Public Javascript API*), camada da Lógica de Negócio (*SDK*) e a camada de Comunicação (*Comm Layer*).

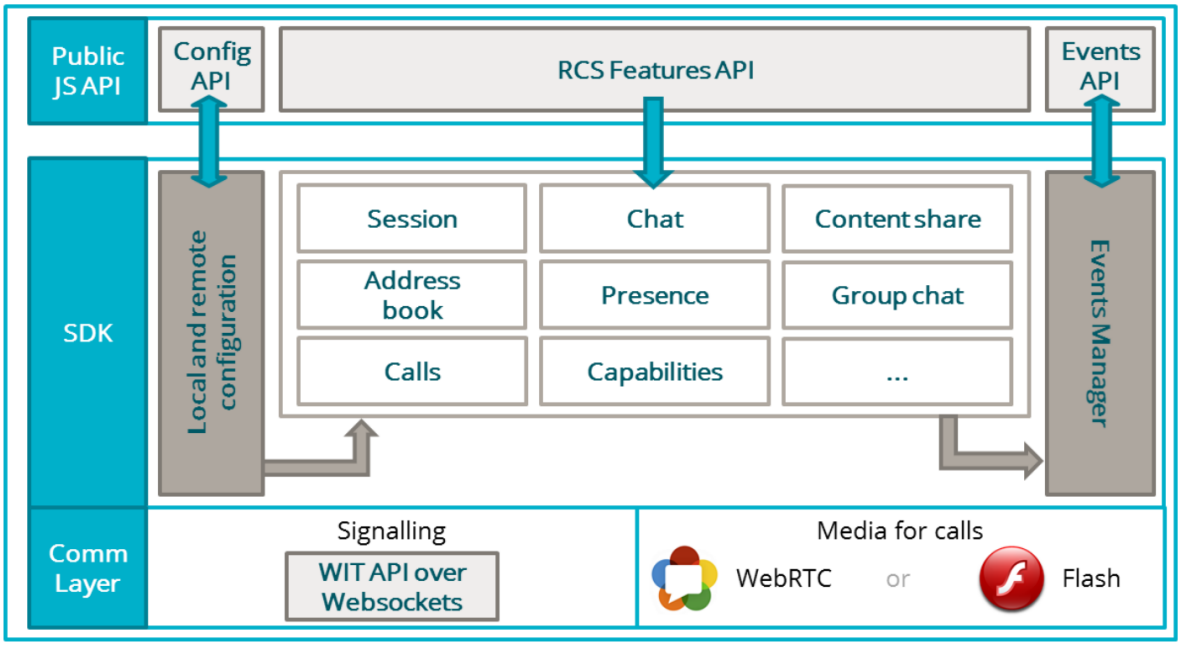



*Figura 7 - WIT WebRTC SDK*

Por motivos de confidencialidade, a informação mais detalhada sobre a Figura 7 está no **Apêndice B – Arquitetura**. Neste documento é feita uma abordagem de alto nível aos tópicos.

A camada de comunicação é o componente responsável pela comunicação da SDK com o servidor.

A camada de negócio é onde se encontra toda a logica da SDK (não acedida diretamente pelas aplicações Web, mas sim através da API). É uma camada que consiste em vários *Managers* (de forma a facilitar a perceção do que são *Managers*, estes podem ser vistos como classes, Figura 7), cada um com funções diferentes, estes são:

- **Conf** - Este *manager* trata da gestão da configuração do acesso do utilizador aos serviços RCS.
- **Session** - O *manager* da sessão permite a aplicação iniciar conexão com o servidor e gerir toda a sessão.
- **Event** - Este *manager* define os eventos que devem notificar a aplicação.
- **Capabilities** - O *manager* das *capabilities* permite obter as *capabities* de cada contacto, e também fazer *polling* aos números da lista de contactos, de forma a receber atualizações do estado dos contactos regularmente.
- **Features** - Este *manager* está referente aos serviços que aplicação que está a executar consegue utilizar, ou seja, permite obter as *capabilities* da própria aplicação.
- **AddressBook** - Este *manager* visa a gestão da lista de contactos de um utilizador, permitindo obter a lista, adicionar, editar e remover contactos e carregar fotos dos contacto.
- **Chat** - Este *manager* permite a utilização de métodos relacionados com o envio de mensagens por IP e gestão de notificações de estado.
- **Messaging** - Este *manager* permite o envio de mensagens SMS a partir de uma aplicação Web.

- **File** - O *manager* de ficheiros permite gerir os fluxos de transferências de ficheiros, seja o seu recebimento ou envio.
- **Call** - Este *manager* implementa os métodos usados para iniciar e gerir chamadas (de voz e vídeo).
- **Presence** - O *manager Presence* gere a informação de presença de um contacto que está publica para todos os utilizadores de aplicações RCS.

A camada de serviço está diretamente disponível ao programador. Trata-se de uma API em *JavaScript* que dá acesso a todos os métodos e canais de comunicação disponíveis nos servidores da WIT. O programador precisa de dados de autenticação confirmados pela entidade que fornece o serviço. Assim consegue configurar a API (com os dados de autenticação) pode aceder às funcionalidades (envio de mensagens, chamadas ou envio de ficheiros etc.) e aos eventos da SDK (chegada de nova mensagem ou de *capabilities*).

## 4.3. WIT Web Communicator (WWC) Chrome App

Este capítulo tem como objetivo explicar, do ponto de vista arquitetural, a aplicação que foi desenvolvida pelo autor. Foi utilizada a *framework AngularJS* 1 para o auxílio do desenvolvimento da *Chrome App* o que levou o estagiário a implementar um modelo de arquitetura MVC (*Model-View-Controller*). Dentro de uma aplicação desenvolvida em *Angular* o MVC é representado por, respetivamente: *Model*, representado pelos serviços, que mantêm os modelos de dados; *View*, pelo HTML e diretivas, que tratam da apresentação de toda a informação ao utilizador; *Controller*, representado pelos controladores que fazem a ponte entre o modelo de dados e a *View*.

Dentro do *AngularJS* existem vários artefactos que ajudam tanto na organização do código como no desenvolvimento rápido e estruturado da aplicação como serviços, controladores e diretivas. Para além disso, existem vários conceitos muito importantes desta *framework* que serão brevemente abordados, de modo a facilitar a compreensão da arquitetura ao leitor. Estes conceitos são: *data-binding*, *scope* e injeção de dependências.

Por motivos de confidencialidade, a informação mais detalhada sobre a aplicação desenvolvida está no **Apêndice B – Arquitetura**. Neste documento é feita uma abordagem de alto nível aos conceitos.

De seguida será explicada a base da aplicação, uma Chrome App.

### 4.3.1. Chrome App

A base de desenvolvimento do projeto é uma Chrome App, aplicação que corre dentro do *browser* Google Chrome.
As Chrome Apps são desenvolvidas em HTML5, CSS e JavaScript e a sua utilização aproxima-se bastante de uma aplicação nativa do sistema operativo. Dado que o browser Google Chrome dispõe de versões para vários sistemas operativos, torna-se possível afirmar que as Chrome Apps são aplicações multiplataforma.

#### 4.3.1.1. App Container

As Chrome Apps são aplicações diferentes das outras aplicações web pois o *App Container* (o modelo *App Container* descreve a aparência visual e os comportamentos das Chrome Apps) abstrai o utilizador de elementos de controlo habituais num browser, como a barra do URL ou a seta de retroceder. Assim, as Chrome Apps, parecem-se com aplicações nativas do sistema operativo, e como não existe o campo de URL, o utilizador não consegue interferir diretamente com o fluxo normal da aplicação. Apesar de carregarem o mesmo tipo de dados (HTML, CSS ou JavaScript) as Chrome Apps diferenciam-se em muito das usuais Web Apps na medida em que não abrem sequer num separador do browser, mas sim numa janela à parte (com se de uma aplicação nativa se tratasse, até porque o código da aplicação está armazenado localmente) [48].

#### 4.3.1.2. Ciclo de vida

O ciclo de vida de uma Chrome App é suportado por vários módulos importantes, esses são o módulo *app.runtime* e o *event page*.

O *app.runtime* gere a instalação da aplicação, controla o *event page* e pode fechar a aplicação a qualquer momento. O *event page* espera pelos eventos lançados pelo *app.runtime* e gere o que inicia e como inicia. O *app.runtime* carrega o *event page* do sistema do utilizador e o evento *onLaunch()* é chamado. O evento *onLaunch()* indica que página deve ser aberta e quais as suas dimensões.

Quando o *event page* não tem JavaScript a executar, *callbacks* pendentes ou janelas abertas, o *app.runtime* fecha a aplicação. Antes de fechar, é chamado o evento *onSuspend()* que dá a possibilidade de fazer uma limpeza nas tarefas antes de fechar. Na Figura 8 está representado um diagrama do clico de vida de uma Chrome App.

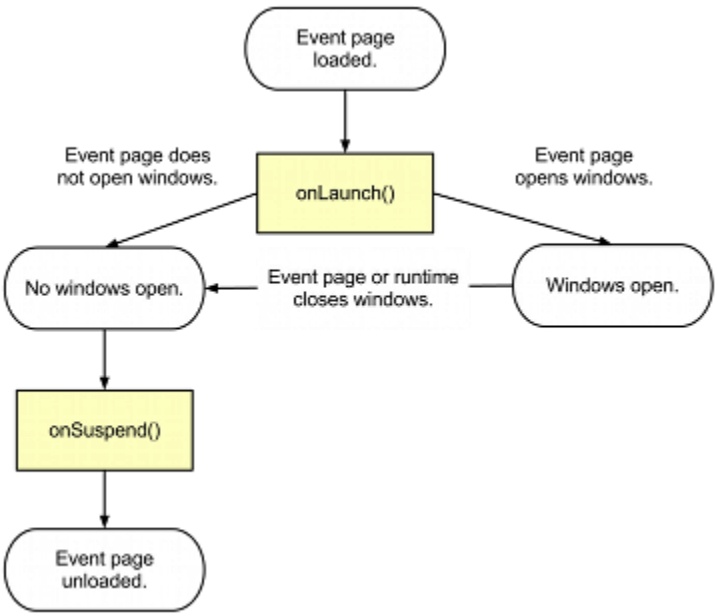



*Figura 8 - Ciclo da vida de uma Chrome App [49]*

#### 4.3.1.3. Gerar uma aplicação

Do ponto de vista do programador, a estrutura de uma Chrome App pode ser definida da seguinte forma:

- **manifest.json** – Este ficheiro específica como carregar a aplicação, indicando um ficheiro JavaScript (*background.js*) para ser executado. É também neste ficheiro que são especificadas as permissões necessárias ao funcionamento da aplicação.
- **background.js** – Este ficheiro é usado para criar o *event page* responsável pelo ciclo de vida da Chrome App. Ex.: *onLaunch( create( 'index.html' ) )*
- **index.html** – É a primeira *view* da aplicação a ser carregada.

De forma a gerar uma aplicação executável em qualquer computador com o Google Chrome é necessário ter, pelo menos, estes três ficheiros supra mencionados numa pasta (pasta do projeto). Após colocar o *browser* em modo de programador, é possível observar duas novas opções no menu das extensões, onde uma delas é "Comprimir extensão". Depois carregar nessa opção indica-se o local da pasta do projeto e é feito o *upload*. Um aspeto importante é o seguinte: após a primeira geração da aplicação, é recomendável que seja guardada a chave primária gerada para que as próximas vezes seja gerada uma aplicação sempre com o mesmo ID, de forma a facilitar o *updates* (ou caso se queira colocar a aplicação disponível na Google Web Store).

### 4.3.2. Angular

Este projeto está a ser desenvolvido recorrendo à utilização da *web framework AngularJS* 1. Trata-se de uma *framework* JavaScript que tem em vista a produção de SPA (*single-page applications*), isto é, aplicações onde apenas existe uma página, tornando a sua navegação mais fluida e sem atualizações ou ligações a outras páginas [50].

O *AngularJS* 1 foi contruído visando a organização e a simplicidade no desenvolvimento, seguindo o modelo arquitetural MVC (*Model-View-Controller*). Contem inúmeras ferramentas e artefactos que ajudam no desenvolvimento do código que são de seguida abordados.

#### 4.3.2.1. Data-binding

O *Angular* utiliza a técnica de *data-binding* para conseguir assegurar que o valor definido no controlador está atualizado na *View* (HTML) e vice-versa. Isto é feito por algoritmos de *dirty ckecking*, estes guardam o valor e sempre que é propagado um evento de alteração nos dados é feita uma comparação entre os novos valores e os antigos.

#### 4.3.2.2. Scope

O *scope* trata-se de um objeto *JavaScript* especial que faz a ponte entre o *View* (HTML) e o *controller* (*JavaScript*). Cada *scope* tem associado métodos e variáveis (*controller*) que podem ser acedidas pela *view*. É possível ainda declarar os *controllers* com um *wrapper controllerAs*, permitindo uma maior facilidade em manipular o acesso aos dados entre diferentes *scopes* na *view*.

#### 4.3.2.3. Injeção de dependências

A injeção de dependências é um *software design pattern* utilizado pelo *Angular* que permite criar referências dinâmicas para determinados componentes, não sendo necessária a sua declaração explícita (*hardcoded*). Isto ajuda a tornar os componentes altamente reutilizáveis e de fácil gestão.

Os serviços são um excelente exemplo de componente que faz pleno uso da injeção de dependências, pois são injetados seja em controladores, serviços ou mesmo filtros e diretivas.

#### 4.3.2.4. Diretivas

As diretivas em *Angular* podem ser definidas como extensões do tradicional HTML. Desta forma é possível criar elementos da DOM novos com as propriedades que se pretenda. Dentro da arquitetura MVC, a *View* corresponde à camada de apresentação dos dados, o HTML. As diretivas fazem parte desta camada, sendo elas também responsáveis pela representação da *View* na aplicação.

#### 4.3.2.5. Controladores

Os controladores são objetos *JavaScript* que controlam o fluxo dos dados na aplicação. Os controladores têm associadas variáveis, métodos e um *scope*, que se refere à zona da aplicação que está a ser manipulada. É pelo meio de controladores que é feita a ponte entre os modelos de dados e a *View*, tratando-se os controladores como o C (*Controller*) dentro da arquitetura MVC.

#### 4.3.2.6. Serviços

Os serviços foram um dos principais artefactos utilizados na criação da aplicação e representam o MODEL dentro da arquitetura MVC, contendo todos os métodos que acedem a dados e conteúdos visualizados na aplicação. Cada serviço trata-se de um *singleton* que pode ser acedido por toda a aplicação (uma vez injetado no controlador, serviço, filtro ou diretiva) e foi criado com o objetivo de facilitar a utilização dos métodos das SDK nos controladores, ou simplesmente para conter métodos gerais e utilizados em vários pontos do código.

### 4.4. Módulos e funcionalidades

Os módulos tratam-se de um conjunto de funcionalidades, tendo sido desenvolvidos os seguintes:

- Autenticação e *auto-start* – o utilizador deve ser capaz de fazer *login* na aplicação e, caso pretenda, persistir a sua sessão de forma a não precisar de introduzir novamente os dados de *login*. Para além disso a aplicação deve iniciar automaticamente com o início do sistema operativo.

- Gestão de contactos – o utilizador pode aceder à sua lista de contactos (sincronizada com o dispositivo móvel) e também gerir os contactos (adicionar, editar ou eliminar). De referir que os contactos estão guardados remotamente, podendo ser acedidos de qualquer parte do mundo.

- Troca de mensagens – o utilizador deve ser capaz de enviar e receber mensagens de texto (incluindo *emoticons* e *emojis*). Tanto a troca de SMS como *Instant Messages* estão incluídas.

- Notificações – o utilizador deve ser notificado sempre que ocorrem eventos (chegada de novas mensagens ou chamadas).

- Troca de ficheiros – o utilizador deve ser capaz de enviar e receber ficheiros de outros utilizadores.

- Chamadas – o utilizador deve ter a possibilidade de fazer e receber chamadas de voz e vídeo.

Por motivos de confidencialidade, toda a informação mais detalhada deste capítulo está no **Apêndice B – Arquitetura**.

# 5. Desafios na implementação

Durante o estágio o autor foi-se deparando com alguns problemas e entraves que acabaram por ser resolvidos. Neste capítulo são apresentados alguns dos desafios mais relevantes que surgiram durante o desenvolvimento da aplicação.

## 5.1. Incompatibilidades da WebRTC Gateway com a Chrome App

Este problema foi encontrado logo no início do desenvolvimento, quando se tentaram fazer os primeiros protótipos da aplicação. A SDK da WebRTC Gateway da WIT foi construída para funcionar perfeitamente em qualquer *browser*. No entanto as Chrome Apps desabilitaram o uso de alguns métodos *JavaScript* e propriedades dos *browsers*, alguns por questões de segurança, outros por questões de performance e *User eXperience* (UX) [51].

Foram detetadas duas incompatibilidades: uma com o acesso ao *local storage* (propriedade presente na maioria dos *browsers*) e outra com o método *unload* do jQuery.

Foram criadas *user stories* para a resolução de cada um dos problemas. O método *unload* está associado a um *shim* que é usado em *browsers* que não possuem o método *saveAs()*. Como o Google Chrome já tem esta funcionalidade implementada, este problema tornou-se irrelevante.

A propriedade *local storage* trata-se de uma área onde os *browsers* permitem que *websites* guardem informação de forma persistente e sem data de expiração (como faz a *session storage*). Essa propriedade é utilizada em inúmeros métodos da SDK, sendo obrigatório estar funcional.

As Chrome Apps removeram o acesso à propriedade *local storage*, uma vez que se tratavam de métodos síncronos e possivelmente poderiam interferir com a interação e experiencia de navegação do utilizador.

Como forma de substituição, os programadores da Google desenvolveram uma nova área de armazenamento do *browser* dedicada às Chrome Apps e Chrome Extensions, a *chrome.storage*.

De forma a conseguir prosseguir no desenvolvimento, o autor iniciou a *user story* de compatibilização do *local storage*.

Começou-se por tentar substituir diretamente o *local storage* pelo *chrome.storage*, no entanto as chamadas da nova API de armazenamento para as Chrome Apps são assíncronas e o problema permaneceu. Após algum estudo e pesquisa, foi possível chegar a uma solução onde o problema da assincronização é mitigado.

## 5.2. Problemas de performance com listas de contactos longas

Aquando do desenvolvimento da gestão de contactos, para efeitos de testes, foi criada uma lista com mais de 700 contactos. O seu objetivo é verificar o comportamento da aplicação

quando submetida a elevada carga, tanto no carregamento dos contactos, como no funcionamento após os contactos estarem disponíveis na lista.

Notou-se logo de início uma demora na resposta do servidor com a lista de contactos. Esse problema foi eliminado removendo o atributo *photo* dos contactos (aquando do seu carregamento inicial). Sendo possível, mais tarde, ir buscar a foto de qualquer contacto, consoante a necessidade.

Outro problema com listas de contactos longas é a criação, na DOM HTML, da lista de contactos, que atrasa a aplicação. Esse problema foi resolvido durante o segundo semestre com a implementação de uma lista infinita. Esta ferramenta calcula o tamanho do elemento HTML da lista de contactos e descobre quantos contactos ficam visíveis ao utilizador. A seguir carrega apenas o número de contactos visíveis ao utilizador para a DOM (Ex.: em vez de carregar 700 contactos, carrega apenas 30). Desta forma a lista de contactos fica visível em muito menos tempo e a navegação na aplicação torna-se muito mais fluida, visto estarem muito menos elementos *renderizados* na aplicação.

## 5.3. Problemas ao importar o Google Maps

As Chrome Apps foram desenvolvidas pelos programadores da Google no sentido de se assemelharem com aplicações nativas do sistema operativo, pensando sempre em maximizar a performance e segurança. Visto isto, alguns serviços/funcionalidades foram desativadas (como pôde ser visto no capitulo que aborda o problema de acesso ao *localstorage*). Nas Chrome Apps não é permitido aceder diretamente a bibliotecas externas.

Até um determinado momento esta limitação não causou nenhum contratempo, visto que era possível fazer *download* de todas as bibliotecas e usa-las como ficheiros locais (exemplo disso são: *jQuery, AngularJS, underscoreJS*, etc.). A Google Maps API (biblioteca usada no desenvolvimento da funcionalidade de partilha de localização) é impossível de ter alojada localmente, pois cada *frame* do mapa visualizada corresponde a uma imagem gerada, de forma automática, pelos serviços Google Maps.

Posto isto, foi necessário recorrer à utilização de *iframes* encapsuladas em *sandboxes* de forma a ser possível importar dinamicamente a biblioteca. As *sandboxes* criam uma página (dentro da aplicação) com um ponto de origem diferente da aplicação e têm acesso a todas as funcionalidades e serviços de uma página Web tradicional.

Depois disso surgiu outro pormenor, como as *sandboxes* são criadas de forma independente da origem da aplicação, não têm acesso aos dados da aplicação. A única forma de comunicação possível passa pela criação de uma canal especial de comunicação (*postMessage()*) entre a parte principal da aplicação e a *sandbox*.

Assim foi possível incorporar cada *iframe* (que representa um mapa) na aplicação e interagir com ela através de *postMessages*.

## 5.4. Partilha de video com aplicações móveis

De início estava definido que seria implementada a partilha de gravações de vídeo instantâneas, assim como é feito com a partilha de clipes de áudio. Após uma pesquisa e estudo sobre as formas de fazer gravação de vídeo recorrendo às tecnologias base da aplicação desenvolvida (HTML e Javascript), obteve-se que o formato final do vídeo seria do tipo WebM, um formato de vídeo de alta qualidade desenvolvido a pensar na partilha web [52].

Depois de implementada a funcionalidade com gravação de vídeo, observou-se que as aplicações móveis RCS da WIT não reconheciam o formato. Relativamente à aplicação Android, poderia ser possível a reprodução do vídeo no formato WebM, com a implementação de métodos de suporte a esse formato [53].

No entanto, na aplicação iOS, não seria possível a sua visualização, uma vez que o iOS não tem compatibilidade com este formato [54].

Visto isto, tornou-se inviável ter uma aplicação a partilhar conteúdo que outras (dentro do mesmo sector da mesma empresa) não consigam reproduzir e optou-se por fazer a gravação de uma imagem GIF animada. Para além da compatibilidade, a criação de GIFs traz outras vantagens, na medida em que ocupam menos espaço, seja no disco do dispositivo, como na largura de banda da rede, aquando do seu envio.

# 6. Testes e casos de uso

Neste capítulo são analisados os testes feitos e documentados da aplicação e também apresentados os casos de uso da aplicação final.

## 6.1. Testes

Os testes foram feitos ao longo do desenvolvimento. Como especificado no *Definition of done*, elaborado no início do ano letivo, a cada *sprint* eram feitos testes de aceitação pelo autor. Esses testes serviam para verificar se as funcionalidades implementadas estavam corretas, tanto nos casos de sucesso como nos de insucesso. Para além de serem testadas as funcionalidades desenvolvidas nesse *sprint*, eram também testadas as funcionalidades implementadas em *sprints* anteriores, de forma a prevenir que fossem estragadas funcionalidades antigas com o desenvolvimento das novas. Sempre que se detetaram problemas, eram criadas novas *user stories* com a finalidade de os resolver.

Os testes (**Apêndice D**) foram documentados seguindo as *guidelines* de testes de *software* da WIT e validados pela equipa de testes. Neste último *sprint* foi executada a verificação destes testes em duas fases.

Numa primeira fase os resultados foram os seguintes (Tabela 2):

| **Autenticação** | |
|---|---|
| **WWCCA-AUTH-0100 -** Login (OTP errada) | ✓ |
| **WWCCA-AUTH-0200 -** Login | ✓ |
| **WWCCA-AUTH-0300 -** Auto-login | ✓ |
| **Contactos** | |
| **WWCCA-CONT-0100 -** Carregar lista de contactos | ✓ |
| **WWCCA-CONT-0200 -** Adicionar contacto | ✓ |
| **WWCCA-CONT-0300 -** Adicionar contacto (campos vazios) | ✓ |
| **WWCCA-CONT-0400 -** Editar contacto | ✓ |
| **WWCCA-CONT-0500 -** Remover contacto | ✓ |
| **WWCCA-CONT-0600 -** Pesquisar contacto | ✓ |
| **Mensagens** | |
| **WWCCA-MSG-0100 -** Enviar mensagem | ✓ |
| **WWCCA-MSG-0200 -** Enviar mensagem SMS | ✓ |
| **WWCCA-MSG-0300 -** Enviar ficheiro | ✓ |
| **WWCCA-MSG-0400 -** Criar e enviar clipe áudio | ✓ |
| **WWCCA-MSG-0500 -** Criar e enviar GIF animado | ✓ |
| **WWCCA-MSG-0600 -** Partilhar contacto | ✓ |
| **WWCCA-MSG-0700 -** Partilhar contacto (campos desseleccionados) | ✓ |
| **WWCCA-MSG-0800 -** Partilhar localização | ✓ |

| | |
|---|---|
| **WWCCA-MSG-0900 -** Visualizar imagem | ✓ |
| **WWCCA-MSG-1000 -** Visualizar vídeo | ✓ |
| **Chamadas** | |
| **WWCCA-CALL-0100 -** Iniciar chamada | × |
| **WWCCA-CALL-0200 -** Atender chamada (voz) | ✓ |
| **WWCCA-CALL-0300 -** Atender chamada (vídeo) | ✓ |
| **WWCCA-CALL-0400 -** Rejeitar chamada | ✓ |
| **WWCCA-CALL-0500 -** Rejeitar chamada com mensagem | ✓ |
| **WWCCA-CALL-0600 -** Terminar chamada | ✓ |
| **WWCCA-CALL-0700 -** Colocar chamada em espera | ✓ |
| **WWCCA-CALL-0800 -** Desligar micro | ✓ |
| **Configurações** | |
| **WWCCA-SETT-0100 -** Mudar foto presença | × |
| **WWCCA-SETT-0200 -** Mudar nome de presença | × |
| **WWCCA-SETT-0300 -** Mudar configuração do Auto-Start | ✓ |

*Tabela 2 - Resultados da primeira fase de testes*

Nesta primeira fase ocorreram alguns problemas relacionados com as chamadas e com a presença do contacto.

Relativamente à funcionalidade de iniciar uma chamada, estava a falhar quando, depois de adicionar um novo contacto, não estavam a ser pedidas corretamente as *capabilities* de videochamada, ficando o botão de iniciar um videochamada (no ecrã de informações de um contacto, no menu de contactos) sempre bloqueado.

Em relação às informações de presença, o problema provem do servidor que guarda as informações de presença dos utilizadores. Os métodos que respondem aos pedidos de informação da WebRTC Gateway SDK foram alterados, pelo que o problema foi reportado à equipa responsável.

Após a correção dos problemas foi iniciada a segunda fase de execução dos testes, resultando na Tabela 3.

| **Autenticação** | |
|---|---|
| **WWCCA-AUTH-0100 -** Login (OTP errada) | ✓ |
| **WWCCA-AUTH-0200 -** Login | ✓ |
| **WWCCA-AUTH-0300 -** Auto-login | ✓ |
| **Contactos** | |
| **WWCCA-CONT-0100 -** Carregar lista de contactos | ✓ |
| **WWCCA-CONT-0200 -** Adicionar contacto | ✓ |
| **WWCCA-CONT-0300 -** Adicionar contacto (campos vazios) | ✓ |
| **WWCCA-CONT-0400 -** Editar contacto | ✓ |
| **WWCCA-CONT-0500 -** Remover contacto | ✓ |
| **WWCCA-CONT-0600 -** Pesquisar contacto | ✓ |
| **Mensagens** | |
| **WWCCA-MSG-0100 -** Enviar mensagem | ✓ |
| **WWCCA-MSG-0200 -** Enviar mensagem SMS | ✓ |
| **WWCCA-MSG-0300 -** Enviar ficheiro | ✓ |
| **WWCCA-MSG-0400 -** Criar e enviar clipe áudio | ✓ |
| **WWCCA-MSG-0500 -** Criar e enviar GIF animado | ✓ |
| **WWCCA-MSG-0600 -** Partilhar contacto | ✓ |
| **WWCCA-MSG-0700 -** Partilhar contacto (campos desseleccionados) | ✓ |
| **WWCCA-MSG-0800 -** Partilhar localização | ✓ |
| **WWCCA-MSG-0900 -** Visualizar imagem | ✓ |
| **WWCCA-MSG-1000 -** Visualizar vídeo | ✓ |
| **Chamadas** | |
| **WWCCA-CALL-0100 -** Iniciar chamada | ✓ |
| **WWCCA-CALL-0200 -** Atender chamada (voz) | ✓ |
| **WWCCA-CALL-0300 -** Atender chamada (vídeo) | ✓ |
| **WWCCA-CALL-0400 -** Rejeitar chamada | ✓ |
| **WWCCA-CALL-0500 -** Rejeitar chamada com mensagem | ✓ |
| **WWCCA-CALL-0600 -** Terminar chamada | ✓ |
| **WWCCA-CALL-0700 -** Colocar chamada em espera | ✓ |
| **WWCCA-CALL-0800 -** Desligar micro | ✓ |
| **Configurações** | |
| **WWCCA-SETT-0100 -** Mudar foto presença | ✓ |
| **WWCCA-SETT-0200 -** Mudar nome de presença | ✓ |
| **WWCCA-SETT-0300 -** Mudar configuração do Auto-Start | ✓ |

*Tabela 3 - Resultado da segunda fase dos testes de aceitação*

Pela análise da Tabela 3 pode ser visto que os testes têm aceitação máxima, tendo sido todos executados com sucesso após as correções necessárias. De seguida é apresentada a Tabela 4 como exemplo da documentação feita para os testes de aceitação.

<table>
<tr><th colspan="4">Caso de teste WWCCA-AUTH-0300: Autenticação – Auto-Login</th></tr>
<tr><td colspan="2"><u>Autor</u>:</td><td colspan="2">japedro</td></tr>
<tr><td colspan="4"><u>Resumo</u>:<br>O propósito deste teste é verificar se um utilizador consegue fazer Login, de forma automática, na aplicação, com sucesso.</td></tr>
<tr><td colspan="4"><u>Pré-condições:</u><br>Nenhuma</td></tr>
<tr><td><u>#</u></td><td colspan="2"><u>Ação:</u></td><td><u>Resultado esperado:</u></td></tr>
<tr><td>1</td><td colspan="2">Abrir aplicação</td><td>É apresentado um slider de loading com mensagens e imagens sugestivas acerca da aplicação. Passados uns instantes (tempo de login, que difere consoante a velocidade da ligação do utilizador) aparece</td></tr>
<tr><td colspan="2"><u>Tipo de execução:</u></td><td colspan="2">Manual</td></tr>
<tr><td colspan="2"><u>Importância:</u></td><td colspan="2">Média</td></tr>
<tr><td colspan="2"><u>Requisitos:</u></td><td colspan="2">Já ter executado login pelo menos uma vez na aplicação (teste WWCCA-AUTH-0100).</td></tr>
<tr><td colspan="2"><u>Palavra-chave:</u></td><td colspan="2">Nenhuma</td></tr>
</table>

*Tabela 4 - Teste exemplo WWCA-AUTH-0300*

As restantes tabelas de testes encontram-se documentadas no **Apêndice D**.

## 6.2. Casos de uso

Por motivos de confidencialidade, esta secção está feita de forma integral no **Apêndice C**.

# 7. Conclusão

Este documento reflete todo o trabalho durante os últimos nove meses em que o autor esteve em estágio. Como em qualquer projeto surgiram algumas dificuldades, no entanto com a ajuda do orientador e do tutor estas foram resolvidas da melhor forma.

O projeto desenvolvido foi criado de raiz e é totalmente inovador, na medida em que não existe no mercado nenhuma Chrome App RCS. Por se estar a trabalhar com tecnologias recentes e nunca antes implementadas em conjunto, foram encontrados e superados grandes desafios, contribuindo para o crescimento, não só profissional, mas também pessoal do autor. Foi necessária confiança e persistência para ultrapassar alguns problemas mais complicados, como a questão da performance das listas de contacto longas ou da partilha de localização.

Dada a conjugação de serviços e tecnologias, este trabalho foi apresentado à Google durante a maior feira de telecomunicações do Planeta, Mobile World Congress (MWC), realizada no mês de Fevereiro de 2016, em Barcelona.

Ao chegar ao fim deste projeto, faz-se uma análise sobre tudo o que se viveu e aprendeu e são de salientar os inúmeros aspetos, aprendidos durante o percurso académico, que foram implementados na prática, em situações reais. Desde as reuniões de trabalho periódicas, onde tudo tinha que estar preparado em condições e a horas como a responsabilidade sobre das decisões tomadas.

Apesar de este projeto estar a ser desenvolvido apenas pelo autor, as metodologias utilizadas foram as mesmas que a empresa emprega internamente, em equipas de desenvolvimento, ficando o autor acostumado ao ritmo de trabalho em ambiente real. Os conhecimentos do autor aumentaram bastante em relação a tecnologias de desenvolvimento Web, tendo consolidado imenso os conhecimentos em programação com *JavaScript* e aprendido como desenvolver uma aplicação com *AngularJS*.

Também aspetos de organização e planeamento do trabalho foram aprendidos e consolidados. O autor aprendeu como se trabalha dentro de uma equipa de forma ágil, quais os processos da metodologia e, principalmente, aprendeu a tirar partido das vantagens desta metodologia de trabalho (de forma a melhorar o seu desempenho e a qualidade do trabalho produzido).

O desenvolvimento deste documento durante o ano letivo permitiu ainda treinar e melhorar a capacidade de escrita, argumentação, pesquisa e organização do estagiário.

Relativamente aos objetivos propostos, pode-se afirmar que o trabalho desenvolvido é positivo. É possível executar uma aplicação Web, estável, que permite gerir a lista de contactos presente no dispositivo móvel do utilizador, permite enviar mensagens SMS e *chat* e também realizar chamadas VoIP e GSM (sobre a rede celular dos operadores de telecomunicações).

No início do ano letivo foi construído o *backlog* com as funcionalidades indispensáveis a uma aplicação de comunicação. Todas as funcionalidades foram implementadas, à exceção do *Group Chat*. Essa funcionalidade foi, desde início posta como prioridade mínima e durante o segundo semestre optou-se por se dedicar mais tempo ao teste e estabilização da aplicação. Assim prescindiu-se de uma funcionalidade, no entanto certificou-se que todas as outras tinham qualidade e estavam implementadas de forma correta e sem problemas.

## Trabalho futuro

Durante o desenvolvimento foram aparecendo ideias de novas funcionalidades e estas foram sendo registadas.

Posto isto, é possível enumerar algumas funcionalidades que podem ser tomadas em conta como trabalho futuro, estas são:

- Lista de *emoticons*/*emojis* recentes. Consiste em criar uma lista onde apareçam os últimos *emoticons*/*emojis* utilizados;
- Integração com redes sociais (*Facebook* e *Google+*) para possibilitar a publicação de conteúdo na conta do utilizador.

# 8. Bibliografia


[1] "Daily Mail - SMS and IM," 214. [Online]. Available: http://www.dailymail.co.uk/sciencetech/article-2538488/SMS-takes-seat-IM-number-texts-sent-Britain-falls-time.html. [Acedido em 21 Janeiro 2016].

[2] M. Zuckerberg, "Facebook," 04 Novembro 2015. [Online]. Available: https://www.facebook.com/photo.php?fbid=10102457977071041&set=pb.4.-2207520000.1452450619.&type=3&theater. [Acedido em 04 Novembro 2015].

[3] "Android Pit," [Online]. Available: http://www.androidpit.com.br/melhor-aplicativo-de-mensagem. [Acedido em 21 Janeiro 2016].

[4] "W3 Schools - Browser Stats," [Online]. Available: http://www.w3schools.com/browsers/browsers_stats.asp. [Acedido em 22 Julho 2016].

[5] "MQTT," [Online]. Available: http://mqtt.org/faq. [Acedido em Outubro 2015].

[6] Z. Avraham, "Quora," 14 Março 2014. [Online]. Available: https://www.quora.com/Why-didnt-Facebook-Chat-Message-use-XMPP-Jabber-like-other-IMs. [Acedido em Outubro 2015].

[7] "MQTT," 12 Agosto 2011. [Online]. Available: http://mqtt.org/2011/08/mqtt-used-by-facebook-messenger. [Acedido em Outubro 2015].

[8] "Developer Microsoft Edge - Object RTC API," [Online]. Available: https://developer.microsoft.com/en-us/microsoft-edge/platform/documentation/dev-guide/realtime-communication/object-rtc-api/. [Acedido em 28 Junho 2016].

[9] "Programmable Web - What Developers Should Know About ORTC Versus WebRTC," 12 Outubro 2015. [Online]. Available: http://www.programmableweb.com/news/what-developers-should-know-about-ortc-versus-webrtc/analysis/2015/10/12. [Acedido em 28 Junho 2016].

[10] "Chrome Web Store," 22 Julho 2015. [Online]. Available: https://chrome.google.com/webstore/detail/videolink2me/hbhickbdbnacieekojgdhpimcomkgami. [Acedido em Outubro 2015].

[11] "Crunch Base," [Online]. Available: https://www.crunchbase.com/organization/videolink2-me#/entity. [Acedido em 12 Janeiro 2016].

[12] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Videolink2.me. [Acedido em Outubro 2015].

[13] "Videolink2me," [Online]. Available: http://blog.videolink2.me/. [Acedido em Outubro 2015].

[14] "Chrome Web Store," [Online]. Available: https://chrome.google.com/webstore/detail/google-hangouts/nckgahadagoaajjgafhacjanaoiihapd?hl=pt-PT. [Acedido em Outubro 2015].

[15] "Wikipedia," [Online]. Available: https://pt.wikipedia.org/wiki/Google_Hangouts. [Acedido em Outubro 2015].

[16] "Quora," 7 Setembro 2014. [Online]. Available: https://www.quora.com/How-complex-is-Google-Hangouts-under-the-hood. [Acedido em Outubro 2015].

[17] "Google," [Online]. Available: https://developers.google.com/+/hangouts/writing. [Acedido em Outubro 2015].

[18] "Google," [Online]. Available: https://support.google.com/hangouts/answer/2944865?hl=pt&ref_topic=2944848&vid=1-635796601730424729-505051703. [Acedido em Outubro 2015].

[19] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Facebook_Messenger. [Acedido em Outubro 2015].

[20] "Business insider," Novembro 2014. [Online]. Available: http://www.businessinsider.com/why-is-facebook-messenger-a-separate-app-2014-11. [Acedido em Outubro 2015].

[21] "The Verge," [Online]. Available: http://www.theverge.com/2015/4/8/8371349/facebook-messenger-web-browsers-now-available. [Acedido em Outubro 2015].

[22] "Expanded Ramblings," [Online]. Available: http://expandedramblings.com/index.php/facebook-messenger-statistics/. [Acedido em Outubro 2015].

[23] "Tech Crunch," 25 Março 2015. [Online]. Available: http://techcrunch.com/2015/03/25/facebook-launches-messenger-platform-with-content-tools-and-chat-with-businesses/#.ooq183:Timt. [Acedido em Outubro 2015].

[24] "Statista," [Online]. Available: http://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/. [Acedido em Outubro 2015].

[25] "About.com," [Online]. Available: http://im.about.com/od/im-clients/a/compare-skype-premium-and-free-accounts.htm.

[26] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/WhatsApp#Competition_and_shares. [Acedido em Outubro 2015].

[27] "Whats App," [Online]. Available: http://www.whatsapp.com/faq/en/web/28080003.

[28] "Wikipedia," [Online]. Available: https://pt.wikipedia.org/wiki/WhatsApp#cite_note-multipla-1. [Acedido em Outubro 2015].

[29] "Whats App," [Online]. Available: http://blog.whatsapp.com/245/Why-we-dont-sell-ads?.

[30] "Whats App," [Online]. Available: http://www.whatsappear.com.br/funcoes-whatsapp/.

[31] "Google Web Store," [Online]. Available: https://chrome.google.com/webstore/detail/line/menkifleemblimdogmoihpfopnplikde. [Acedido em Outubro 2015].

[32] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Line_(application). [Acedido em Outubro 2015].

[33] "Wikipedia," [Online]. Available: https://pt.wikipedia.org/wiki/JQuery. [Acedido em 12 Janeiro 2016].

[34] "jQuery," [Online]. Available: http://jquery.com/browser-support/. [Acedido em 12 Janeiro 2016].

[35] "jQuery," [Online]. Available: https://jquery.com/. [Acedido em 12 Janeiro 2016].

[36] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/JQuery. [Acedido em 12 Janeiro 2016].

[37] "Wikipedia - AngularJS," [Online]. Available: https://en.wikipedia.org/wiki/AngularJS. [Acedido em 11 Janeiro 2016].

[38] "Docs AngularJS," [Online]. Available: https://docs.angularjs.org/guide. [Acedido em 11 Janeiro 2016].

[39] "AngularJS," [Online]. Available: https://angularjs.org/. [Acedido em 11 Janeiro 2016].

[40] "Developer Chrome," [Online]. Available: https://developer.chrome.com/apps/angular_framework. [Acedido em 11 Janeiro 2016].

[41] "Blog Mgechev," 6 Abril 2015. [Online]. Available: http://blog.mgechev.com/2015/04/06/angular2-first-impressions/. [Acedido em 11 Janeiro 2016].

[42] "AngularJS Blogspot," 12 2015. [Online]. Available: http://angularjs.blogspot.pt/2015/12/angular-2-beta.html. [Acedido em 11 Janeiro 2016].

[43] "Angular," [Online]. Available: https://angular.io/. [Acedido em 11 Janeiro 2016].

[44] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/React_(JavaScript_library). [Acedido em 11 Janeiro 2016].

[45] "Code Mentor," [Online]. Available: https://www.codementor.io/reactjs/tutorial/react-js-flux-architecture-tutorial. [Acedido em 17 Janeiro 2016].

[46] "Wikimedia," [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/thumb/5/58/Scrum_process.svg/2000px-Scrum_process.svg.png. [Acedido em 13 Janeiro 2016].

[47] "Wit Software - WebRTC," [Online]. Available: https://www.wit-software.com/products/webrtc/.

[48] "Developer Chrome," [Online]. Available: https://developer.chrome.com/apps/app_architecture. [Acedido em 15 Janeiro 2016].

[49] "Developer Chrome," [Online]. Available: https://developer.chrome.com/apps/app_lifecycle. [Acedido em 15 Janeiro 2016].

[50] "Wikipedia - Single page application," [Online]. Available: https://en.wikipedia.org/wiki/Single-page_application. [Acedido em 18 Janeiro 2016].

[51] "Developer Chrome - App Deprecated," [Online]. Available: https://developer.chrome.com/apps/app_deprecated. [Acedido em 20 Janeiro 2016].

[52] "WebM Project," [Online]. Available: https://www.webmproject.org/. [Acedido em 20 Julho 2016].

[53] "Developer Android Media-Formats," [Online]. Available: https://developer.android.com/guide/appendix/media-formats.html. [Acedido em 10 Março 2016].

[54] "Developer Apple Media Layer," [Online]. Available: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/MediaLayer/MediaLayer.html. [Acedido em 10 Março 2016].

[55] "Google," [Online]. Available: https://support.google.com/plus/answer/1216376?hl=en. [Acedido em Outubro 2015].

[56] "W3schools," [Online]. Available: http://www.w3schools.com/browsers/browsers_stats.asp. [Acedido em 14 Janeiro 2016].

[57] "Silence It," [Online]. Available: http://www.silenceit.ca/wordpress/wp-content/uploads/2009/04/trunk-branch-tag.png. [Acedido em 15 Janeiro 2016].

[58] "SlideShare," [Online]. Available: http://pt.slideshare.net/mobizen/rcs-overview?related=1. [Acedido em Outubro 2015].

[59] "SlideShare," [Online]. Available: http://pt.slideshare.net/Reports_Corner/market-opportunity-rich-communications-suite-rcs-reports-corner?related=1. [Acedido em Outubro 2015].

[60] "GSMA," [Online]. Available: http://www.gsma.com/network2020/wp-content/uploads/2012/11/IDC-report.RCS-market-prospects.December2012.pdf. [Acedido em Outubro 2015].

[61] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Rich_Communication_Services. [Acedido em Outubro 2015].

[62] "GSMA," [Online]. Available: http://www.gsma.com/network2020/rcs/. [Acedido em Outubro 2015].

[63] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/GSM_Association. [Acedido em Outubro 2015].

[64] "Interop Technologies," [Online]. Available: http://www.interoptechnologies.com/rcs/. [Acedido em Outubro 2015].

[65] "GSMA," [Online]. Available: http://www.gsma.com/network2020/faq/what-is-rcs/. [Acedido em Outubro 2015].

[66] "GSMA," [Online]. Available: http://www.gsma.com/network2020/wp-content/uploads/2015/08/GSMA-Urgent-Clarification-on-RCS5-3-Implementation-v1.0.pdf. [Acedido em Outubro 2015].

[67] "GSMA," [Online]. Available: http://www.gsma.com/network2020/wp-content/uploads/2013/08/RCC_13_v2.0.pdf. [Acedido em Outubro 2015].

[68] "Ars Technica," [Online]. Available: http://arstechnica.com/information-technology/2013/05/hands-on-with-hangouts-googles-new-text-and-video-chat-architecture/. [Acedido em Outubro 2015].

[69] "C-Net," [Online]. Available: http://www.cnet.com/news/in-chrome-googles-hangouts-plugin-goes-extinct/. [Acedido em Outubro 2015].

[70] "Wikipedia," [Online]. Available: https://pt.wikipedia.org/wiki/Scrum. [Acedido em 14 Janeiro 2016].

[71] "Wikipedia," [Online]. Available: https://pt.wikipedia.org/wiki/Desenvolvimento_%C3%A1gil_de_software. [Acedido em 14 Janeiro 2016].

[72] "Diario da Rede," 25 Novembro 2013. [Online]. Available: http://www.diariodarede.com.br/2013/11/25/scrum-em-10-minutos/. [Acedido em 14 Janeiro 2016].

[73] "Wit-Software," [Online]. Available: https://www.wit-software.com/products/rcs-suite/. [Acedido em 13 Janeiro 2016].

[74] "Wit-Software," [Online]. Available: https://www.wit-software.com/products/webrtc/. [Acedido em 13 Janeiro 2016].

[75] "Tutorials Point - AngularJS MVC," [Online]. Available: http://www.tutorialspoint.com/angularjs/angularjs_mvc_architecture.htm. [Acedido em 18 Janeiro 2016].

[76] "Tutorials Point - Angular Dependency Injection," [Online]. Available: http://www.tutorialspoint.com/angularjs/angularjs_dependency_injection.htm. [Acedido em 18 Janeiro 2016].

[77] "Tutorial Points - AngularJS scope," [Online]. Available: http://www.tutorialspoint.com/angularjs/angularjs_scopes.htm.

[78] "Wit-Software RCS-Suite," [Online]. Available: https://www.wit-software.com/products/rcs-suite/. [Acedido em 20 Janeiro 2016].