IPG Politécnico da Guarda
Polytechnic of Guarda

RELATÓRIO DE PROJETO

Licenciatura em Engenharia Informática

Filipe Ferreira da Silva

setembro| 2017

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

RELATÓRIO DE ESTÁGIO

FILIPE FERREIRA DA SILVA

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO

EM ENGENHARIA INFORMÁTICA

Setembro/2017

Área curricular: Projeto de Informática

Ano letivo: 2016 / 2017

Comunicação na *cloud* através do *Twilio* com integração em *Salesforce*

Orientador: Prof. José Alberto Quitério Figueiredo

Autor do documento: Filipe Ferreira da Silva

Elementos Identificativos

Aluno

Nome: Filipe Ferreira da Silva
Número: 1011189
Curso: Engenharia Informática

Estabelecimento de Ensino

Escola Superior de Tecnologia e Gestão – Instituto Politécnico da Guarda
Morada: Av. Dr. Francisco Sá Carneiro 50, 6300-559 Guarda
Telefone: 271220120 **Fax:** 271220150

Instituição de Estágio

Dom Digital
Morada: Av. Rainha Dona Amélia, 6300-749 Guarda
Telefone: 271 224 509

Duração do Estágio

Inicio: 1 de junho de 2017
Fim: 1 de setembro de 2017

Orientador do Projeto

Nome: José Quitério

Resumo

O presente documento tem como objetivo apresentar o trabalho desenvolvido por Filipe Ferreira da Silva no âmbito do projeto em contexto de estágio do curso de Engenharia Informática do Instituto Politécnico da Guarda. Nele é descrito o projeto "Comunicação na *cloud* através do *Twilio* com integração em *Salesforce*", na área da comunicação através da *cloud*, planeado e desenvolvido na empresa Dom Digital. Serão descritos o atual Estado da Arte, a metodologia e tecnologias utilizadas, as etapas de planeamento e concepção através de linguagem de modelação, os componentes e o processo de implementação e desenvolvimento.

Trata-se de uma aplicação que foi inicialmente projetada com o intuito de integrar o *Salesforce* com ligações de voz, mas devido à sua evolução gradual, foi também possível a implementação de videochamadas e conversações de *Live-chat*.

O tema do projeto foi proposto pela Dom Digital devido a uma lacuna na sua oferta de serviços no *Salesforce*: a inexistência de meios de comunicação para sistemas telefónicos. Apesar de existirem diversas aplicações que se podiam agregar aos seus serviços, a Dom Digital resolveu criar a sua própria solução, não para concorrer com as opções já existentes, mas para evitar os seus custos e incluir mais uma ferramenta nos seus serviços, criando mais um atrativo para possíveis clientes.

Utilizando a plataforma *Twilio*, foi então criada uma aplicação do género de *Call Center*, organizada através de tarefas, que são atribuídas aos respetivos Operadores. As suas principais funções são as chamadas de voz através de sistema telefónico, videochamadas e *Live-chat* através da *Internet*, pedidos de *callback* por preenchimento de formulário, e a interface de administração que permite a adição/remoção de Operadores e respetivos departamentos. Esta aplicação foi incorporada no *Salesforce*, possibilitando uma utilização simples e rápida para o Operador, que facilmente tem acesso a toda a informação dos Contactos do *Salesforce*.

Palavras-chave: *Twilio*, *Salesforce*, *Call Center*, Sistemas telefónicos, Operador

Abstract

The purpose of this document is to describe the work developed by Filipe Ferreira da Silva within the internship project, as part of the Computer Science Engineering Course of the Polytechnic Institute of Guarda. It describes the project “Cloud communications using Twilio with Salesforce integration”, in the area of cloud communications, planned and developed in the company Dom Digital. The current state of the art, the methodology and technologies used, the planning and design stage, the components, the implementation and development process will be described.
It's an application that was initially designed to integrate Salesforce with voice calls, but due to its gradual evolution, it was also possible to implement video chat and live chat.
The project was proposed by Dom Digital due to a gap in its offer of services in Salesforce: the lack of a communication solution for telephone systems. Although there were several applications that could be added to its services, Dom Digital decided to create its own solution, not to compete with the existing options, but to avoid its costs and to include another tool in its services, creating one more attractive feature to potential customers.
Using the Twilio platform, a call center application was created, organized through tasks, which are assigned to the respective Agents. Its main functions are voice calls via telephone system, video calls and live chat over the Internet, callback requests by form filling, and the administration interface that allows the addition/removal of Agents and their departments. This application has been incorporated into Salesforce, making it easy and quick to use for the Agent, who easily has access to all Salesforce Contacts information.

Keywords: *Twilio*, *Salesforce*, *Call Center*, Telephone systems, Agent

Índice geral

Índice de Imagens

Índice de Tabelas

Lista de acrónimos

ACD: Distribuidor Automático de Chamadas, sistema de telefonia que responde e distribui chamadas recebidas para um grupo específico de terminais ou agentes dentro de uma organização.
API: Interface de Programação de Aplicações, conjunto de códigos e padrões de programação para acesso a uma aplicação de software ou plataforma web.
CEO: Chief Executive Officer, cargo mais elevado na hierarquia da gestão de uma empresa.
CRM: Gestão do Relacionamento com o Cliente, abordagem que coloca o cliente como principal foco dos processos de negócio, com o intuito de perceber e antecipar as suas necessidades, de forma a atendê-los da melhor forma.
CSS: Cascading Style Sheets, mecanismo para adicionar estilo (cores, fontes, espaçamento, etc.) a um documento web.
CTI: Integração entre Computador e Telefonia, processo pelo qual um equipamento telefónico troca informações de uma chamada com um computador, permitindo ao computador ou utilizador uma melhor gestão da chamada.
CTO: Chief Technology Officer, responsável pela arquitetura e infra-estrutura dos sistemas, pesquisas e desenvolvimento de novos produtos.
ERP: Planeamento dos Recursos da Empresa, sistema informático responsável por cuidar de todas as operações diárias de uma empresa.
HTML: Linguagem de Marcação de Hipertexto, linguagem de marcação utilizada na construção de páginas na Web.
IP: Protocolo de Internet.
ISV: Fornecedor de Software Independente, organização especializada em fazer e vender software.
IVR: Unidade de Resposta Audível, serviço para um call center que fornece respostas automáticas para os clientes, sem a intervenção de um operador.
MMS: Serviço de Mensagem Multimédia, tecnologia que permite o envio e receção de mensagens multimédia em telemóveis.
PHP: Hypertext Preprocessor, linguagem interpretada livre, usada originalmente apenas para o desenvolvimento de aplicações.
PSTN: Rede Pública de Telefonia Comutada, rede telefónica mundial comutada por circuitos destinada ao serviço telefónico.
SDK: Kit de Desenvolvimento de Software, conjunto de ferramentas de desenvolvimento e códigos pré-gravados que podem ser usados pelos desenvolvedores para criar aplicações.
SIP: Protocolo de Iniciação de Sessão, protocolo de código aberto de aplicação para iniciar sessões de comunicação entre utilizadores.
SMS: Serviço de Mensagens Curtas, tecnologia que permite o envio e receção de mensagens até 160 caracteres em telemóveis.
TI: tecnologias da informação.
UML: Linguagem de Modelagem Unificada, linguagem-padrão para a elaboração da estrutura de projetos de software.
URL: Localizador Uniforme de Recursos, endereço de rede no qual se encontra algum recurso informático.
VOIP: Voz sobre IP.

WEBRTC: Comunicação pela Web em Tempo Real, coleção de protocolos de comunicação e interfaces de programação de aplicações que permitem a comunicação em tempo real através de ligações peer-to-peer.

1. Introdução

O relatório aqui apresentado é fruto do projeto realizado pelo aluno Filipe Ferreira da Silva, no âmbito da unidade curricular de Projeto de Informática, componente pertencente ao terceiro ano do curso de Engenharia Informática da Escola Superior de Tecnologia e Gestão do Instituto Politécnico da Guarda. O estágio foi realizado de 1 de junho de 2017 a 1 de setembro de 2017 na empresa Dom Digital.

1.1 Contextualização

A existência contínua e ininterrupta do negócio de *Call Centers* revela uma necessidade contínua das empresas sobre esses serviços. É um empreendimento lucrativo tanto por parte do *Call Center* como pela empresa que o empregou.
No entanto, a possibilidade de estabelecer ligação com clientes por voz, video ou *chat* através da *Internet* não é nada de novo. Existem inúmeros *softwares* para o efeito, com os mais variados preços. Porém, visto tratar-se de *software* externo àquele usado pela empresa, origina alguma dificuldade no processo de correspondência dos dados do cliente e sua ligação à empresa, bem como ao operador, cuja organização por inúmeras janelas de diversos conteúdos torna-se num processo bastante difícil, moroso e inflexível.
Múltiplas empresas têm então optado por alternativas mais praticáveis: integração de serviço de *email*, e chamada diretamente no seu serviço de *CRM* (*Customer Relationship Management*), gerando menos labuta ao operador, mais conforto ao cliente devido às possibilidades de apoio que dispõe e, talvez a melhor vantagem, a redução de custos, visto que é uma ferramenta que já se encontra implementada no *CRM*, dispensando a aquisição de mais software.
Estas aplicações são desenvolvidas habitualmente com recurso a *APIs* (*Application Programming Interface*) criadas para estes fins e empregadas consoante as necessidades da empresa. No caso da Dom Digital, que procurava uma solução que pudesse ser incorporável com o *Salesforce*, existem algumas ofertas. Porém estas soluções, umas mais evoluídas que outras, apresentam grandes quantias a pagar, algo que a empresa queria contornar. Foi então feita uma análise económica que viabilizou a realização deste projeto.
A Dom Digital optou então por criar a sua própria solução. Foi utilizada a plataforma *Twilio* em conjunto com o *Node.js* para programar a aplicação referente aos sistemas telefónicos, videochamada e *Live-chat*, que foi posteriormente integrada no *Salesforce* onde são visíveis os Contactos com os quais é estabelecida a ligação, e é neste sentido que se enquadra o presente projeto.

1.2 Motivação

Da lista de projetos propostos pela Dom Digital para a realização do estágio, este foi a minha primeira opção, uma vez que a frequência na unidade curricular de Sistemas Distribuídos me deu a conhecer o *Node.js*, ferramenta que me suscitou grande interesse durante as aulas devido às suas possibilidades e que viria a ser utilizada em conjunto com o *Twilio* neste projeto.
A possibilidade de trabalhar com *Salesforce* também pesou na escolha, visto que foi uma plataforma com a qual já tinha tido contacto nas Jornadas de Engenharia Informática, e estando inserida na área da computação na *cloud*, reforçou a minha preferência. Combinando estes fatores, pareceu-me um projeto bastante extenso, atual e abrangente de várias tecnologias com as quais poderia solidificar conhecimentos.

1.3 Objetivos

A Dom Digital tinha como objetivo inicial para este projeto a possibilidade de fazer chamadas telefónicas através do *browser*, de forma integrada no *Salesforce*, criando uma forma fácil, económica e rápida de estabelecer contacto com os clientes, e vice-versa.
Devido à rápida progressão no desenvolvimento das ferramentas de chamada de voz, os objetivos começaram a evoluir e à medida que o projeto crescia foram surgindo as hipóteses de incluir um sistema *IVR* (*Interactive Voice Response*) dependente das escolhas do Cliente, Operadores com diferentes estados e permissões, ligação por videochamada e *Live-chat*, acabando por ser criada uma aplicação bastante mais complexa do que aquele que era o objetivo inicial.

2. Estado da Arte

O leque de soluções existentes na área do serviço de apoio ao cliente é bastante vasto, seja através de fornecedores de *software* de *call center* genérico, ou programados através de sistemas telefónicos virtuais. Neste capítulo será descrito o atual Estado da Arte nestas áreas, bem como na área dos CRMs.

2.1 Serviços de Apoio ao Cliente

Atualmente qualquer empresa de média dimensão que visa oferecer um suporte adequado dispõe de serviço de apoio ao cliente por *email*, chamada de voz ou *live chat* servindo-se de *software* externo. Existem dezenas de sistemas telefónicos projetados para uso comercial. Estes sistemas de *call center* são altamente configuráveis, podendo encaminhar as chamadas recebidas para um departamento específico, para quem não recebeu uma chamada pelo período mais longo, ou mesmo dependendo do grau de prioridade do cliente.

2.1.1 Modo de operação das empresas no seu serviço de apoio ao cliente

Nestes sistemas geralmente recorre-se a uma organização baseada em *queues*: se a primeira pessoa na fila não estiver disponível para atender a chamada, o cliente segue para a próxima pessoa na fila, com base na configuração do sistema. Se a chamada passar por toda a fila e ninguém estiver disponível para atender, o cliente pode deixar uma mensagem de voz, ou um pedido de *callback*. Desta forma, nenhum cliente fica por atender. Dependendo do nível de suporte que se quer prestar as possibilidades de *call center* são variadas.

2.1.2 Tipos de *software* de serviço de apoio ao cliente

Dependendo da maneira como as comunicações são processadas e geridas, conforme a tecnologia usada, os *call centers* são divididos pelas seguintes categorias:

- **Sistemas de *call center on-premise*** - geralmente são vendidos com uma licença de utilização mensal em que o cliente tem a responsabilidade de atualizar e manter o *call center*. Requerem instalação de *hardware* e equipamento privado e são os *call centers* mais caros;
- **Sistemas de *call center virtual*** - este modelo virtual exige que a empresa substitua o seu *call center* tradicional e centralizado por um virtual, onde se pode ligar com clientes usando o seu próprio equipamento. A diferença é que é o fornecedor quem aloja o sistema em servidores físicos e a empresa não precisa de atualizar o seu equipamento, apenas pagar as taxas mensais. O acesso a esses serviços é fornecido através de uma ligação de rede direta que pode ou não ser executada através da *Internet*;
- **Sistemas de *call center* baseados na *cloud*** - o *software* é estendido à computação na *cloud*, o que significa que o *call center* será acessível a partir de qualquer lugar, sem a necessidade de descarregar e instalar nenhum *software* específico. São os sistemas mais acessíveis.

2.1.3 Características de um *software* de serviço de apoio ao cliente

Com o crescimento dos serviços de *call center*, as expectativas quanto ao seu desempenho também aumentaram, no entanto existem algumas características indispensáveis para o seu funcionamento:

- ***ACD: Automatic Call Distributor*** - o Distribuidor Automático de Chamadas é o núcleo de todos os *call centers*, pois pode encaminhar as chamadas para o agente mais adequado, ou para o que não recebeu uma chamada pelo período mais longo;
- ***IVR: Interactive Voice Response*** - as Respostas de Voz Interativas são o recurso que cuida dos clientes antes e depois de estarem ligados aos agentes. Um cenário *IVR*, inclui uma mensagem de saudação, um menu de instruções de serviço para escolher, bem como a cobertura da fila de espera;
- ***Queues*** - cada agente ou departamento tem uma fila de espera específica onde os clientes chegam depois de serem encaminhados.

Existem também diversos atributos que permitem ao cliente distinguir por entre as várias alternativas e escolher o *software* mais adequado para a sua necessidade [1]. Algumas das qualidades que se devem ter em conta na comparação entres as diferentes alternativas são:

- **Confiabilidade de Serviço / *Uptime*** - nenhuma outra métrica é tão importante quanto a confiabilidade. Qualquer minuto em que o *call center* esteja inoperacional, significa dinheiro perdido. Ao contrário da generalidade das soluções *on-premise*, as soluções virtuais são redundantes para melhorar a confiabilidade;
- **Facilidade de uso** - mesmo as características mais complexas do *software* devem ser fáceis de aprender e utilizar;
- **Boa visão sobre o cliente** - o *software* precisa de extrair informação sobre o cliente e devolvê-la ao agente, para que o possa servir melhor;
- **Tarefas Automatizadas** - deve-se aproveitar ao máximo o tempo do agente, automatizando certas tarefas recorrentes.

Tendo em consideração a importância das características referidas anteriormente, a tabela seguinte compara os principais fornecedores de *software* de *call center*.

Tabela 1: Principais fornecedores de *software* de *call center*, adaptado de [GetVoip]

Fornecedor	inContact	Mitel	Genesys	Five9	RingCentral
Gama de custos mensais (por utilizador)	Variável	3 níveis, variando de $99,99 - $159,99	3 níveis, variando de $99,99 - $179,99	Variável	$75
Taxa de instalação	Variável	Variável	Grátis	Variável	Variável
*Setup**	9,4	8,6	7	8,4	8,2
Confiabilidade*	8,8	8,8	9	8,4	7,8
Suporte*	8,8	8,8	8	7,8	7,6
Qualidade*	9,2	8,8	9	7,8	8
*Valores obtidos através de *reviews* no site GetVoip a 15/06/2017					

Através da tabela pode-se concluir que existem diversas soluções, umas com mais qualidade que outras, qualidade essa que é proporcional ao preço.

2.2 Sistemas telefónicos virtuais

Um sistema telefónico virtual é uma tecnologia em que os fornecedores de serviços alojam sistemas telefónicos e os fornecem remotamente, podendo vender números de telefone. O funcionamento aquando da receção de uma chamada é bastante simples: o serviço na *Internet* do provedor responde às chamadas recebidas nesse número conforme o desejado, seja através de um *template* pré-definido, chamada direta, ou mesmo consultando uma aplicação para finalidades mais específicas. Podem-se então configurar atendedores automáticos para fornecer uma saudação personalizada e oferecer um menu de opções para o cliente, bem como fazer chamadas para o exterior através de um *softphone*. Tudo isto feito através de um computador, sem ser necessário um telefone físico. Todas estas particularidades são vistas como acréscimos à qualidade da ferramenta de trabalho de um operador, tornando a sua lida mais fluida e confortável. A figura seguinte demonstra, de forma simplificada, o funcionamento de um sistema telefónico virtual.

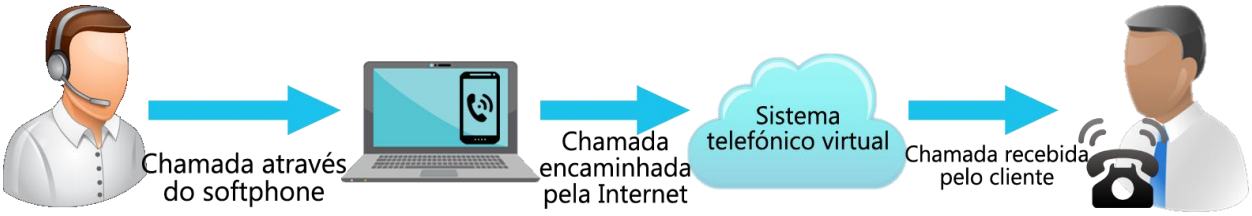

Fig. 1 : Funcionamento de um sistema telefónico virtual

Devido ao notável aumento no consumo dos serviços relacionados com chamadas, os fornecedores optaram por criar *APIs* que permitiram a integração dos seus serviços nos diferentes pontos de ação dos clientes, seja nos seus *websites*, plataformas de apoio ao cliente, *software* profissional ou nos seus serviços de *CRM*. *API* é um conjunto de procedimentos, protocolos e ferramentas que um programa envia para outro, para a construção de aplicações. São usadas para que programas individuais se possam comunicar diretamente e usar as funções uns dos outros.
Do mesmo modo que uma interface gráfica simplifica a utilização de programas ao utilizador, as *APIs* facilitam aos programadores o emprego de certas tecnologias na construção de aplicações. As *APIs* são uma das formas mais comuns pelas quais as empresas de tecnologia se complementam entre si.
Uma boa *API* facilita o desenvolvimento de um programa, fornecendo todos os blocos de construção e o programador, em seguida, monta os blocos. Uma *API* pode ser usada num sistema baseado na *web*, sistema operativo, sistema de base de dados, ou biblioteca de *software*.
O processo de funcionamento é simples: a *API* que emite chamadas remotas a um servidor ou serviço. Essas chamadas são tratadas por uma *framework* correspondente à *API*. Esta *framework* solicita recursos ao servidor da *API* sob a forma de *dependencies*, que permitem que o código funcione na metodologia para a qual foi projetado. Finalmente, os dados são fornecidos ao cliente no formato restrito determinado pela *API*.
As principais plataformas fornecedoras de *API's* de sistemas telefónicos virtuais serão abordadas nos seguintes subcapítulos.

2.2.1 *Twilio*

Para além de uma forte compreensão dos programadores e das necessidades dos clientes, a *Twilio* opera com mais de 1.000 operadoras telefónicas em mais de 150 países para oferecer os seus

serviços. O fornecedor possui altos níveis de confiabilidade avaliando rigorosamente as operadoras, de forma a oferecer serviços com o mínimo de falhas. Possui também uma consola única onde é possível obter históricos de todas as ligações, e gerar gráficos com informação útil, como duração média das chamadas, tempo de atendimento de cada operador, etc. [1]

2.2.2 *Nexmo*

Com uma grande presença global no mercado das *APIs* de comunicação, a *Nexmo* oferece números de SMS (*Short Message Service*) em mais de 35 países e números de voz em mais de 90 países. Possui encaminhamento adaptativo, o que garante que as mensagens serão entregues através da melhor rota possível com o menor tráfego. Algo exclusivo da *Nexmo* são os preços flexíveis, em que os utilizadores podem ser cobrados "por segundo" em oposição a "por minuto". [2]

2.2.3 *Plivo*

Com um alcance global mais reduzido do que as opções anteriores, o *Plivo* ainda oferece suporte a uma grande variedade de locais com mais de 50 países para números de voz e 19 países para números de mensagens SMS. Possui no mínimo duas operadoras locais por país, de forma a prevenir problemas de rede e oferecendo um serviço de 99.95% de tempo de atividade. Também possui um sistema inteligente de encaminhamento de chamadas para garantir que as ligações são feitas pelo caminho mais rápido. A gravação e o armazenamento de chamadas gratuitos são características únicas que ajudam o *Plivo* a destacar-se entre a competição [3].

Na tabela seguinte são referenciados alguns fatores importantes para a escolha de um destes serviços, como preços, facilidade de uso, *setup*, etc.

Tabela 2: Comparação Twilio, Nexmo e Plivo

Fornecedor	**Twilio**	**Nexmo**	**Plivo**
Número dedicado	$1/mês	$0.75/mês	$0.8/mês
SMS	$0.0075	$0.0064	$0.0035
Serviço *Shortcode*	$1.000/mês	$1.000/mês	$1.000/mês
SMS *Shortcode*	$0.01	$0.0065	$0.004
Chamadas *In*	$0.01/min	$0.0045/min	$0.005/min
Chamadas *Out(Fixo)*	$0.013/min	$0.0104/min	$0.00/min
Chamadas *Out(Mobile)*	$0.1150/min	$0.10/min	$0.10/min
Toll Free In	$0.0275/min	Variável	$0.210/min
Cumpre requisitos*	9,3	7,8	9,3
Facilidade de Uso*	9,1	8,2	9,1
Setup*	9	8,4	8,2
Administração*	8,4	7,3	8
Suporte/Apoio*	8,8	7,8	7,4

*Valores obtidos através de *reviews* no site G2crowd a 15/06/2017

Apesar de o Twilio ser a solução mais cara, é também a que tem maior qualidade, uma vez que cumpre os seus requisitos, possui bons recursos para *setup* e bom suporte.

2.3 *CRM*

Customer Relationship Management é uma tecnologia que permite gerir as relações com os clientes, colocando-os no foco dos processos de negócio, com o intuito de perceber e antecipar necessidades. Ajuda as equipas a atuar, tanto a nível interno como a nível externo, a reunir conhecimento social, encontrar mecânicas importantes e fornece diversos meios de comunicação. Geralmente os pagamentos são feitos com base na quantidade de utilizadores, e vão de pequenas quantias de menos de 10€ até valores superiores a 150€, dependendo do fornecedor e dos serviços adquiridos.

Uma boa plataforma *CRM* tem de: [4]

- Ajudar a reconhecer necessidades;
- Acompanhar a jornada do cliente;
- Possibilitar o foco num único ou vários clientes;
- Adquirir novos *leads*;
- Aumentar a satisfação do cliente;
- Disponibilizar permanentemente os dados;
- Melhorar a cooperação entre colaboradores;
- Criar relatórios personalizados.

2.3.1 Principais serviços de *CRM*

Das centenas de opções que existem atualmente, existem alguns serviços de *CRM* que se destacam. Na figura seguinte é apresentada a percentagem de gastos em *software CRM* por fornecedor, no ano de 2015, totalizando um mercado no valor de $26.2B. [5]

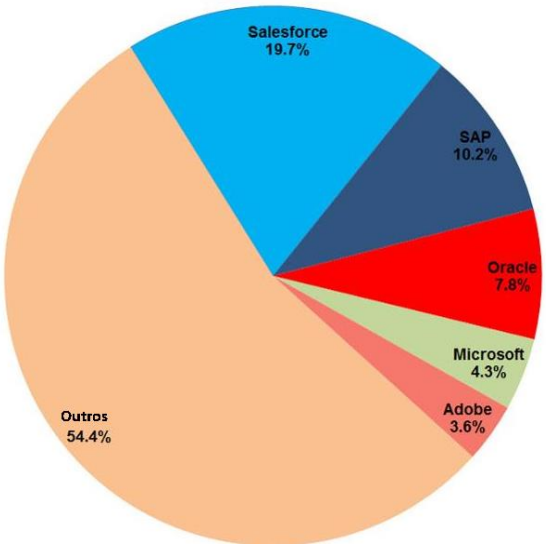

Fig. 2: Gastos em software CRM em 2015, adaptado de Forbes

Para melhor percepção da distribuição dos diferentes serviços de *CRM* pelo mercado das grandes empresas, os que mais se destacam podem ser separados por quatro categorias de modo a serem avaliados quanto ao seu posicionamento no mercado atual:

- **Produtos líderes** - produtos com alta avaliação e grande presença no mercado, como é o caso do *Salesforce* e do *Zoho*;
- **Alto desempenho** - produtos com alta avaliação, mas que não alcançam a quota de mercado e a escala dos líderes (ex.: *Nimble*, *bpm'online*, *Pipedrive*, *Prophet*, *PipelineDeals*, *HubSpot* e *ProsperWorks*);
- **Significativos** - têm recursos e uma presença significativa no mercado, mas receberam uma baixa avaliação por parte dos utilizadores (ex.: *Microsoft Dynamics 365 for Sales*, *Oracle Siebel*, *SAP* e *Oracle CRM On Demand*);
- **Nicho** - baixa presença no mercado, e avaliação insuficiente (ex.: *Sugar*, *Contactually* e *Insightly*).

Estas quatro categorias estão representadas no gráfico seguinte, onde são organizadas num eixo cartesiano em função da sua presença no mercado e satisfação dos clientes.

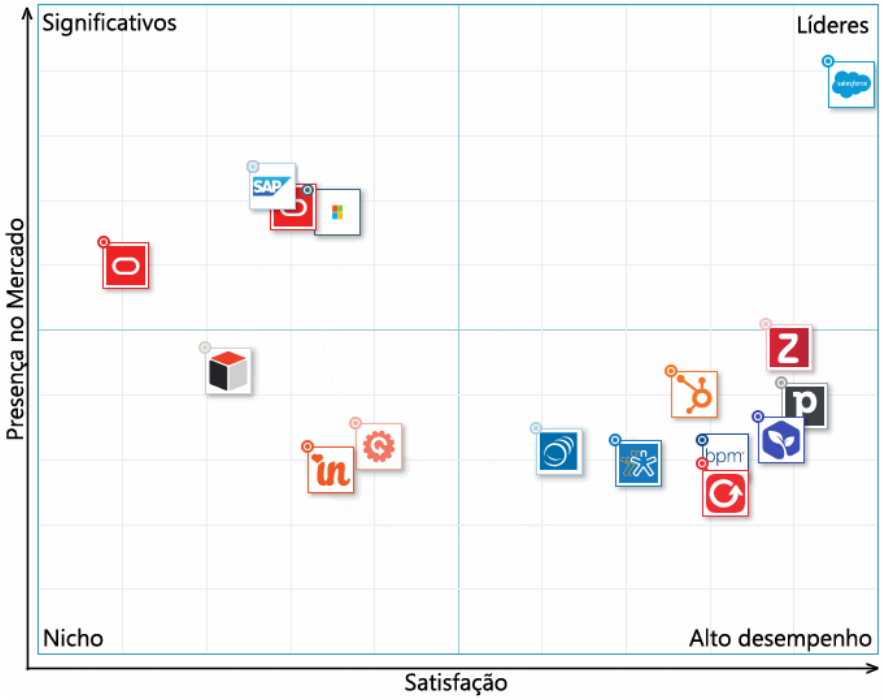

Fig. 3: Categorias dos principais serviços de CRM, adaptado de G2Crowd

2.3.2 Alternativas no *Appexchange*

As duas principais alternativas no que toca a aplicações *CTI* (*Computer Telephony Integration*) disponíveis no *Appexchange* são o *NewVoiceMedia* e o *CloudCall*. Oferecem basicamente o mesmo serviço: relatórios de chamadas, gravação de chamadas, anúncios através de chamadas, reencaminhamento de chamadas, *pop-ups*, sistema *IVR*, etc, porém a *NewVoiceMedia* conta com mais clientes devido aos seus serviços serem completamente compatíveis com dispositivos mobile. Os preços são variáveis, com base na escala do cliente [6].

3. Metodologia e tecnologias utilizadas

Neste capítulo serão descritas as metodologias aplicadas, as tarefas em conjunto com um diagrama de Gantt, e as tecnologias que foram utilizadas para a realização do projeto.

3.1 Metodologias aplicadas

Para a concepção deste projeto foi utilizada a metodologia de Desenvolvimento Ágil que tem como objetivo acelerar o desenvolvimento do *software*, através de iterações e melhorias contínuas. O processo da metodologia utilizado foi o *Scrum*, onde as tarefas são divididas em ciclos, tipicamente de 2 a 4 semanas [7]. Foram realizadas pequenas reuniões diárias com o supervisor do projeto e *CTO* (*Chief Technology Officer*) da Dom Digital, Micael Costa, com o objetivo de dar uma visão geral do estado do trabalho e apontar/corrigir pequenas falhas e possíveis adições. Também foram realizadas duas outras reuniões, mais rigorosas, com a presença do *CEO* (*Chief Executive Officer*) da Dom Digital, onde foram feitas demonstrações mais meticulosas e pormenorizadas, onde se acabava também por discutir os próximos passos e possíveis adições ao projeto.
Para acelerar o processo de desenvolvimento do *software*, qualquer intervalo de tempo que houvesse durante a elaboração do projeto, era despendido na consulta de documentos de formação em duas plataformas: *Trailhead* (*Salesforce*) ou *Radical Skills* (*Twilio*).
No final de cada semana era enviado um documento com os progressos dessa semana, de forma a se manter um historial do projeto, e identificar as etapas mais demoradas.
Sendo esta uma aplicação com alguma magnitude, foram usadas *APIs*, onde grande parte do código já estava escrito, porém sem qualquer tipo de documentação, o que causou alguma dificuldade na sua percepção e entendimento. Para ultrapassar este obstáculo, para além da ajuda de alguns *developers* da Dom Digital, foram consultados vídeo-tutoriais, ferramentas de aprendizagem como o *Codecademy*, ou mesmo código concebido em certas disciplinas da licenciatura.

3.2 Descrição das tarefas

As principais tarefas em todo o planeamento e desenvolvimento do projeto são as seguintes:

1. **Estudo das Tecnologias** - estudo e aprendizagem sobre as tecnologias *Salesforce*, *Twilio*, e linguagens de programação (44 horas);
2. **Estado da Arte** - pesquisa sobre as atuais soluções relativas/semelhantes ao projeto (72 horas);
3. **Análise de requisitos** - identificação das necessidades do projeto (40 horas);
4. **Planeamento** - planeamento das etapas e respetivos objetivos (32 horas);
5. **Consulta de documentação** - consulta de documentação e informação útil acerca de código já existente (96 horas);
6. **Codificação** - escrita do código da aplicação, e da *interface* de pesquisa no *Salesforce* (100 horas);
7. **Escrita do relatório** - composição e formatação do relatório (56 horas).

Na figura seguinte é demonstrado o mapa de Gantt, relativo às tarefas descritas anteriormente. Este diagrama é usado para ilustrar o avanço das diferentes etapas de um projeto. Os intervalos

de tempo representando o início e fim de cada fase aparecem como barras coloridas sobre o eixo horizontal do gráfico. [8]

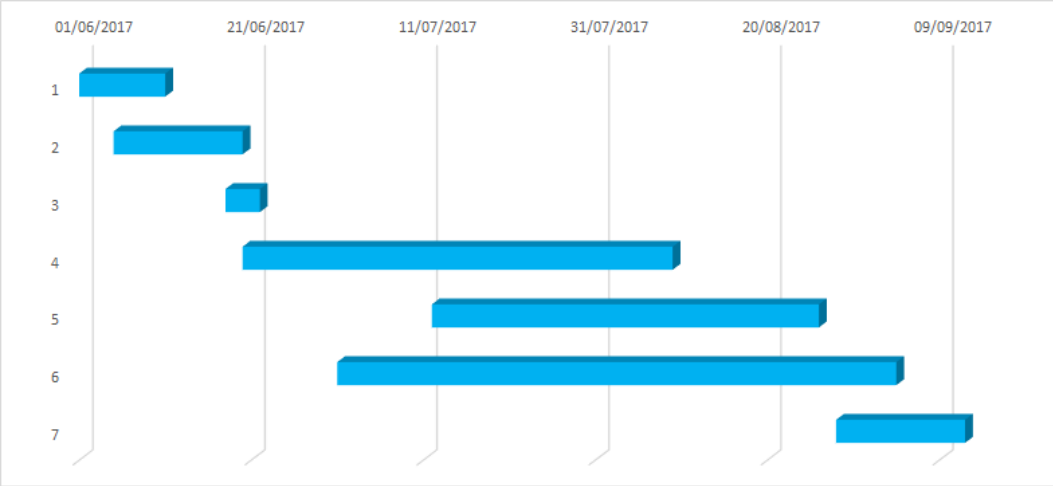

Fig. 4: Mapa de Gantt

3.3 Tecnologias e ferramentas utilizadas

Neste subcapítulo irão ser enumeradas e abordadas as tecnologias e ferramentas que foram utilizadas: as linguagens trabalhadas, as plataformas fornecedoras de serviço e o restante *software*.

3.3.1 Computação na *Cloud*

Cloud computing é um modelo (e não uma tecnologia) que está a revolucionar a forma como as pessoas e as empresas utilizam os recursos disponibilizados pelas TI (Tecnologias de Informação). É um modelo que permite o acesso ubíquo, conveniente e *on-demand*, através da rede, a um conjunto de recursos de computação partilhados (redes, servidores, armazenamento, aplicações, serviços, etc), que podem ser aprovisionados ou libertados, com um mínimo de esforço e sem interação com o fornecedor. O serviço não se encontra registado num determinado computador, mas sim uma rede, sendo que a localização desta rede é desconhecida pelo utilizador. Não existem restrições quanto ao sistema operativo. [9]

O modelo *Cloud* é composto por cinco características essenciais, três modelos de serviço e quatro modelos de implementação (modelo 5x3x4).

Características essenciais:

- ***On-demand self-service*** - o utilizador pode unilateralmente aprovisionar recursos, sem interagir com o fornecedor;
- **Acesso generalizado à rede** - acesso ao serviço em qualquer equipamento ligado à *Internet*;
- **Acesso partilhado a recursos** - optimização dos recursos, gerando menos desperdício;
- **Elasticidade rápida** - possibilidade de *upgrade* ou *downgrade* instantâneo;
- **Serviço medido (*Pay-as-you-go*)** - serviço pago à medida da sua utilização.

Modelos de serviço:

- **IaaS:** ***Infrastructure-as-a-Service*** - utilização de recursos de infraestrutura básicos de computação e armazenamento;
- **PaaS:** ***Platform-as-a-Service*** - utilização de um ambiente de desenvolvimento ou serviço de bases de dados, tipicamente para programadores;
- **SaaS:** ***Software-as-a-Service*** - disponibilização de uma aplicação, vulgarmente através de uma interface *Web* normal; serviço de *cloud* que está mais próximo do consumidor final.

Modelos de Implementação:
Dois tipos principais:
- ***Cloud*** **privada**
- ***Cloud*** **pública**

Duas variantes:
- ***Cloud*** **híbrida** - Permite que uma empresa armazene as suas aplicações que necessitam de um nível elevado de segurança na *cloud* privada e as restantes numa *cloud* pública;
- ***Cloud*** **comunidade** - Partilhada por várias empresas diferentes, mas que têm interesse de partilhar diversas informações, como a missão, os requisitos de segurança, política e considerações sobre o cumprimento.

Ao selecionar um novo sistema de Planeamento de Recursos da Empresa (*ERP*), um dos fatores mais críticos na decisão é a escolha entre *ERP* na *cloud* ou *ERP* local.
Ao contrário do serviço na *cloud*, o *software on-premise* é instalado localmente, nos próprios computadores e servidores de uma empresa.
Os sistemas *ERP* na *cloud* tornaram-se muito mais populares nos últimos anos, especialmente entre pequenas e médias empresas. Esta revela-se uma opção muito mais barata, cómoda e prática do que a tradicional. [9]

Exemplo de custos recorrentes de um servidor ativo (*on-premise* ERP): [10]
- Energia elétrica para fornecer energia ao servidor;
- Energia elétrica do ar condicionado;
- Licenças anuais de *software* e sistema operativo;
- Custos de manutenção (serviços subcontratados, etc);
- Investimento em equipamentos complementares;
- Tempo de "investigação" de problemas;
- Recursos Humanos (equipa de técnicos, muitas empresas não consideram na manutenção);
- *Upgrades* (tempo de vida útil cada vez mais reduzido).

Vantagens do serviço *Cloud*:

- Custos previsíveis ao longo do tempo;
- Custos de equipamentos minimizados;
- Investimento inicial mais barato;
- Nenhum investimento de *hardware* adicional;
- Oferecer maior estabilidade;
- Atualizações contínuas e automáticas;
- Implementação rápida;
- *Backups* constantes;
- Trabalho de equipa facilitado;
- Número crescente de plataformas desenvolvedoras de aplicações *web.*

3.3.2 *Salesforce*

Sendo a Dom Digital a primeira empresa parceira de *Salesforce* em Portugal e sendo este o seu principal foco tecnológico, o serviço de *CRM* escolhido para implementar este projeto não poderia ser outro. A *Salesforce.com, inc* é uma empresa americana fundada em 1999 por Marc Benioff (antigo executivo da *Oracle*), Parker Harris, Dave Moellenhoff e Frank Dominguez, tendo como propósito oferecer um SaaS capaz de criar aplicações empresariais inovadoras. O seu âmago é o serviço de *Customer Relationship Management*, repartido por múltiplas ferramentas de gestão de vendas, análise de dados, suporte ao cliente, automação de eventos, bem como bases de conhecimento, através de um dispositivo *mobile* ou *desktop*. Através destas tecnologias são desenvolvidos e lançados serviços e páginas *web*. A plataforma *Salesforce* é suportada por *cloud computing* e é inteiramente acessível através do *browser*, sendo dispensável infraestruturas, manutenção ou qualquer programa externo para usufruir das suas funcionalidades.
A *Salesforce* assenta-se num modelo de negócio por subscrições, distribuindo os seus serviços por diferentes módulos, consoante as finalidades pretendidas.
Encontra-se completamente traduzida em 18 línguas e no seu conjunto de produtos e serviços destacam-se:

- ***Appexchange*** - mercado online para aplicações de computação na *cloud* produzidas para a plataforma *Salesforce*;
- ***Chatter*** - plataforma de colaboração em tempo real para os clientes;
- ***Connect Offline*** - possibilidade de trabalhar *offline* (sem sincronização);
- ***CRM*** - gestão de clientes;
- ***Force.com*** - plataforma de desenvolvimento de aplicações com integração na plataforma *Salesforce*;
- ***Marketing Cloud*** - coleção de produtos para fins de *marketing*, como *emails* personalizados, SMS/notificações, outros anúncios e interação com clientes na *social media*;
- ***Sales Cloud*** - aplicação para aceder à plataforma por *desktop* ou *mobile*;
- ***Salesforce1*** - nova plataforma móvel para desenvolvedores, *ISVs* (*Independent Software Vendors*), administradores e clientes, e ligada para serviços de vendas, serviços e *marketing*;
- ***Service Cloud*** - serviço de apoio ao cliente;
- ***Setup*** - permite ao cliente configurar e personalizar as suas aplicações de *CRM*;
- ***Web Services*** - *Salesforce* oferece serviços de *APIs* que permitem a integração com outros sistemas. [11]

Os *developers* podem, mediante o negócio do cliente que está a requerer uma aplicação, desenvolver as aplicações em qualquer *cloud*. Cada *cloud* tem os melhores recursos na sua área.
Os clientes da *Salesforce* justificam a sua singularidade em três fatores principais:

- **Rapidez** - o sistema de *CRM* tradicional pode levar mais de um ano para implementar na sua plenitude;
- **Facilidade** - trata-se de um *software* bastante intuitivo e de fácil utilização, permitindo aos clientes a perda de pouco tempo na sua aprendizagem;
- **Eficácia** - dadas as justificativas anteriores, e as suas possibilidades de personalização para atender às necessidades da empresa, os clientes acham o *Salesforce* muito eficaz.

3.3.2.1 *Force.com*

Force.com é uma plataforma de desenvolvimento que permite construir e implementar aplicações da Salesforce.
O Force.com permite que os desenvolvedores criem e implementem rapidamente aplicações confiáveis, sólidas, seguras e escaláveis. Oferece ferramentas e serviços para automatizar processos de negócios, integrar aplicações externas, integração móvel entre outras. [12]

3.3.2.2 Distinção: *Salesforce* vs *Force.com*

A *Salesforce* é a empresa (*Salesforce.com*) que fornece serviços de computação na *cloud*.
Force.com é a plataforma de desenvolvedores e aplica-se à infra-estrutura e base de código que é a base para toda a solução do próprio *Salesforce*.
O termo "*Force*" foi separado de "*Salesforce*" e depois de separado foi sendo mais utilizado para reforçar a ideia de que têm significados diferentes, porque atualmente o *Salesforce* oferece muito mais do que o serviço *CRM*. No fundo são intercambiáveis do ponto de vista técnico/*developer*. [13]

3.3.2.3 *Visualforce*

O *Visualforce* é uma *framework* que permite aos desenvolvedores criar *interfaces* de utilizador sofisticadas e personalizadas que podem ser alojadas nativamente na plataforma *Force.com*. Inclui uma linguagem baseada em *tags*, muito semelhante ao *HTML* (*HyperText Markup Language*) e um conjunto de "controladores padrão" do lado do servidor que fazem operações básicas de base de dados, como *queries* e armazenamentos. Cada *tag Visualforce* corresponde a um componente da *interface*, como uma secção de uma página, uma lista relacionada ou um campo. O comportamento dos componentes do *Visualforce* pode ser controlado pela mesma lógica que é usada nas páginas padrão do *Salesforce*, ou os desenvolvedores podem associar a sua própria lógica a uma classe de controlador escrita em *Apex*. [14]

3.3.2.4 *Apex*

Apex é uma linguagem de programação orientada a objetos exclusiva do *Salesforce*, que permite aos desenvolvedores executar instruções no servidor da plataforma *Force.com* envolvendo ligações com a *API Force.com*. O seu funcionamento é semelhante ao *Java*, porém adaptado à lógica do *Salesforce* e com comandos restritos a esta plataforma, podendo ser inicializado por solicitações de serviço da *Web* e por *triggers* em objetos [15].

3.3.2.5 *Service Cloud*

O *Service Cloud* é um *CRM* de serviço de apoio ao cliente com base no *CRM* para os profissionais da *Salesforce*. Permite aos utilizadores "ouvir"/responder aos clientes e encaminhá-los para um agente apropriado a partir de uma única *interface*, acessível por *desktop, tablet* e *smartphone*. Os clientes também podem aproveitar os benefícios do *Service Cloud*, acedendo à base de conhecimento do sistema e entrando em contacto com outros membros das comunidades à medida que procuram soluções para os seus problemas. Com este tipo de configuração, o *Service Cloud* torna-se uma plataforma que aumenta a fidelidade do cliente e assegura a sua permanência.
O serviço fornecido é altamente personalizado e faz com que os clientes se sintam sempre uma prioridade devido à previsão e antecipação de acontecimentos do sistema, o que ajuda a evitar problemas antes mesmo de acontecerem.
Existe também uma *suite* de colaboração, em que os agentes partilham os seus conhecimentos e se auxiliam em situações críticas. A *suite* também pode ser usada para fins de comunicação e partilha de documentos, para que os agentes também possam discutir problemas e evitá-los em conjunto. Consequentemente o gerente pode facilmente rastrear e avaliar o desempenho dos agentes sem o *feedback* imediato dos clientes. [16]

Principais características do *Service Cloud*

- Definição de objetivos e recompensas;
- Base de Conhecimento Dinâmica;
- *Live-chat;*
- Automação e aprovação de fluxo de trabalho;
- Colaboração de agentes e equipas;
- Integração social;
- *APIs*;
- Gestão de Casos (Atribuição automática, regras e *queues*);
- *CTI - Computer Telephony Integration*;
- Gestão de ativos e acompanhamento de produtos;
- *Role Permissions;*
- Relatórios e análises personalizáveis;
- Multi-Língua;
- Consola personalizável;
- *Mobile;*

Para melhor percepção da importância deste serviço, abaixo é apresentado um gráfico que representa as origens das receitas da *Salesforce*, de onde concluímos que a subscrição do *Salesforce* (*CRM*), juntamente com este serviço de suporte constituem os principais fatores no domínio das receitas.

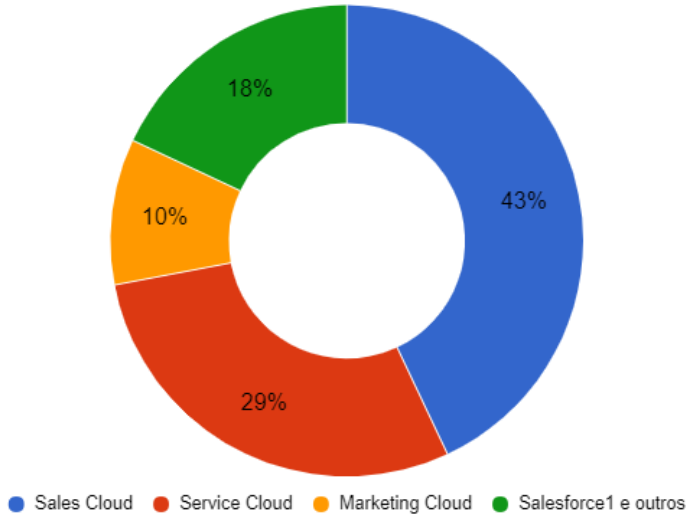

Fig. 5: Origens das receitas do Salesforce, adaptado de Salesforce.com

3.3.2.6 *AppExchange*

Uma das peças de tecnologia que diferencia a plataforma *Force.com* de outras plataformas é o *AppExchange*. O *AppExchange* é uma plataforma onde aplicações criadas no *Force.com* estão disponíveis para clientes do *Salesforce*. Os desenvolvedores podem enviar as suas aplicações para o *AppExchange* se quiserem partilhá-las com a comunidade. Inclui aplicações das mais variadas funções como gestão de folhas de pagamento, integração de telefonia, pesquisas de serviço e suporte ou *dashboards* personalizadas. Algumas das aplicações são criadas internamente pela *Salesforce*, mas a maioria é construída por parceiros e desenvolvedores individuais. [17]

Instalar uma aplicação publicada no *AppExchange* é um processo seguro e simples. Depois de fazer uma pesquisa, analisando as descrições, revisões e demonstrações das aplicações do *AppExchange*, chega-se à escolha de uma aplicação que cumpra as necessidades. A partir daí o programador passa pelas seguintes fases:

- **"*Test Drive*"** - teste de uma demonstração totalmente funcional apenas de leitura para planear implementação;
- **Instalação** - incorporação da aplicação e todos os seus componentes ao ambiente do *Salesforce*;
- **Configuração** - disponibilização de acesso à aplicação aos utilizadores (com permissão).

3.3.2.7 *Open CTI*

A *API Open CTI* permite aos programadores criar integrações entre o *Salesforce* e sistemas telefónicos simples. A *API* é baseada em *JavaScript* e funciona em conjunto com um *iFrame* oferecido pela *Salesforce* para incorporar *softphones* e controlo de chamadas de terceiros. Uma vez que a aplicação está incorporada, esta pode então invocar a funcionalidade do *CTI*. Na maioria dos casos, os clientes associam-se a parceiros da *Salesforce* ou a fornecedores próprios de telefonia para a integração do *CTI*. [18]
Esta interface de controlo permite:

- Mostrar as capacidades de integração do *CTI* da *Salesforce*, inclui funcionalidades como *pop-ups*, *click-to-dial*, registo de chamadas, relatórios e configuração, sem adaptadores externos;
- Servir como ponto de partida e amostra de código para parceiros e clientes que desejem criar integrações *CTI* para o *Salesforce*.

3.3.2.8 *Lightning* vs *Classic*

Uma das mais recentes grandes atualizações do *Salesforce* foi a reformulação da interface da sua plataforma principal. A solução renovada é o *Salesforce Lightning*, que apresenta uma interface muito mais apelativa visualmente, porém carece de algumas funcionalidades existentes no *Salesforce Classic*, bem como documentação útil para programadores.
A qualquer altura se pode navegar e trocar entre as duas versões, no entanto, as aplicações que funcionam em consola (sem necessidade de atualizar a página) apenas funcionam no *Lightning*. Sendo que a proposta deste projeto é a realização de conversações em vários meios através do browser, a única plataforma onde este serviço se torna praticável é o *Lightning*, visto que a página não pode ser atualizada. Na figura seguinte é apresentada uma comparação das duas versões.

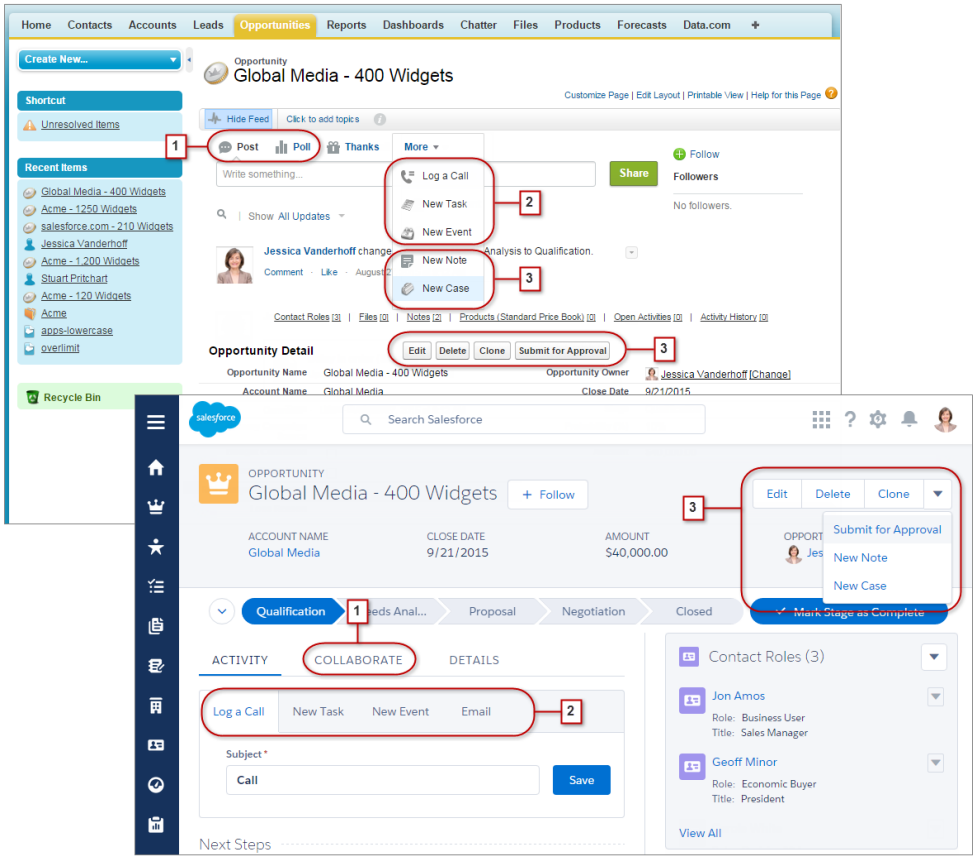

Fig. 6: Comparação Salesforce Classic (esq.) e Lightning (dir.)

3.3.3 *Twilio*

O *Twilio* é um serviço baseado na *cloud* que permite a comunicação entre dispositivos móveis, aplicações, serviços e sistemas de um negócio, preenchendo a lacuna na comunicação convencional. Fornece um serviço *web* de infraestrutura de telefonia (hospedado por *Amazon Web Services*) através de uma *REST API* na *cloud*, permitindo aos programadores o uso de linguagens *web* padrão para integrar chamadas de telefone, mensagens de texto e comunicações de voz *IP* (Internet Protocol) nas suas aplicações, o que possibilita experiências mais atraentes e ajuda a automatizar tarefas habituais.
No portal de desenvolvedor existem várias funcionalidades que se destacam. Desde a *dashboard*, que apresenta bastantes recursos interessantes, à funcionalidade *TaskRouter*, ou até mesmo a própria documentação das *APIs* que se encontram devidamente demonstradas.
A consola do desenvolvedor possui vários gráficos de análise (análise de voz e SMS, mostrando o uso de ambos) e acesso rápido a outras funcionalidades. O acesso aos números e a visão geral de *billing* também estão disponíveis, com informações sobre pagamentos recentes e resumos de utilização.
O desenvolvedor tem também acesso a *logs* de atividades, com durações de chamadas, erros que podem ter ocorrido e gráficos de estado, que mostra em que cenários as chamadas terminaram (ocupado, sem resposta, falhou, etc.).
Como seria de esperar de um serviço baseado na *cloud*, as comunicações são disponibilizadas através do modelo *pay-as-you-go*. Não há necessidade de investimento inicial em *hardware/software* ou contratos a longo prazo.

Podem ser criados *triggers* para, por exemplo, enviar um *email* para o programador sempre que uma chamada exceda um certo período de tempo, ou quando o número do desenvolvedor recebe uma chamada ou um texto específico, o *Twilio* é capaz de executar um *script* previamente definido, gravando a chamada.
O programador pode também aceder à sua conta através da *API*, verificar os seus números e chamadas, receber notificações de eventos (úteis para *debugging*) e gerir as autorizações das aplicações. [19]

3.3.3.1 Funcionamento

O *Twilio* virtualiza toda a infraestrutura necessária para construir comunicações modernas num ambiente global baseado na *cloud*. Através das *APIs* do *Twilio* é possível prototipar, construir e implementar soluções de *software* de telefonia, adaptando-se às mudanças das necessidades de comunicação de seus clientes.
Depois do programador se inscrever no *Twilio*, escolhe um número virtual para enviar e receber mensagens de voz ou SMS. De seguida mapeia o número virtual para um URL de "*request*" (o URL da aplicação, que o *Twilio* solicitaria do servidor de aplicações do desenvolvedor ao receber uma chamada de voz em nome do desenvolvedor/cliente). O URL criado pelo desenvolvedor da aplicação, descreve ao *Twilio* como controlar o conteúdo das chamadas telefónicas, como é demonstrado no diagrama abaixo.

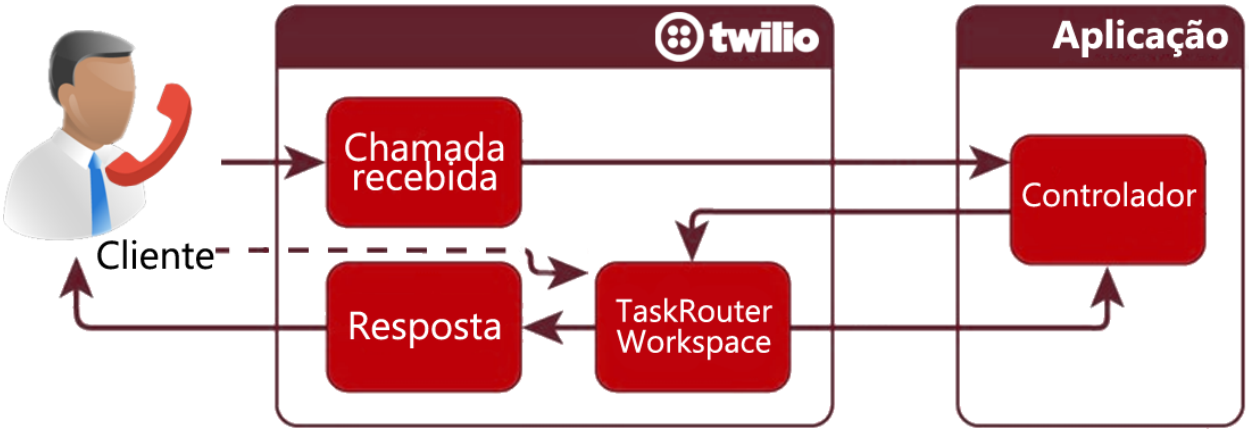

Fig. 7: Diagrama básico de funcionamento do Twilio

3.3.3.2 Áreas de ação

Twilio Voice

O desenvolvedor programa a aplicação para decidir o que fazer com cada chamada em tempo real. As aplicações de voz do *Twilio* utilizam recursos únicos, como encaminhamento de chamadas e controlo completo de chamadas usando *APIs*, baixa latência global e a capacidade de se ligar a operadoras de telefone públicas ou *VoIP*. Outros recursos de chamadas poderosos como gravação de áudio, *Text-to-Speech*, *Call Queues* e conferência global, são incorporados diretamente no *Twilio Voice*. As áreas de produtos de voz da *Twilio* incluem:

- ***PSTN Voice*** - Fazer e receber chamadas através da rede telefónica pública;
- **Cliente *Web*** - Chamadas diretamente através de navegadores *Web* com base na *WebRTC*;
- **Cliente móvel** - Chamada nativa em aplicações *Android* e *iOS*;
- ***Twilio SIP*** - Integração de dispositivos *SIP* (*Session Initiation Protocol*);

- ***TaskRouter*** - Criação de fluxos de encaminhamento de chamadas.

Twilio Messaging

Permite a criação de aplicações que enviam e recebem mensagens SMS e MMS (Media Message Service), bem como *APIs* que permitem conversas *in-app*, recursos de entrega inteligentes e informações de entrega para garantir que as mensagens são recebidas.

- ***Twilio* SMS** - Programação de envio, recepção e controlo de SMS em todo o mundo;
- ***Twilio* MMS** - Disponível apenas nos EUA e Canadá;
- **Mensagens *IP*** - Incorporar mensagens em aplicações móveis e *web*;
- **SMS *Short Codes*** - Para SMS e MMS de alto volume;
- ***Toll Free* SMS** - SMS em números *Toll Free*.

Twilio Authentication (Authy)

O *Authy* fornece uma plataforma de *authentication-as-a-service* para programadores, oferecendo *APIs* e *SDKs* (*Software Development Kit*) para incorporar Autenticação por Dois Fatores. Habilita uma série de casos de uso de autenticação, incluindo a geração de *time-based tokens*, *push notifications* e *tokens* entregues por SMS ou voz. Está disponível com três abordagens diferentes, que podem ser combinadas:

- ***Authy SoftToken*** - Substituição direta de um *token* de *hardware* (ex.: *SecureID*). Fornece uma senha com tempo limitado e que só pode ser usada uma vez, para verificar a identidade do utilizador;
- ***Authy OneCode*** - Código de acesso entregue a um telefone registado via SMS e/ou chamada telefónica;
- ***Authy OneTouch*** - aplicação de *smartphone* que envia solicitações de autenticação que podem ser verificadas através de um desafio fácil (prevenção contra *bots*).

3.3.3.3 Porquê o Twilio?

Para além da forte compreensão das necessidades dos desenvolvedores e dos clientes refletida na simplicidade no uso das *APIs*, a *Twilio* opera em mais de 150 países com mais de 1.000 operadoras telefónicas devidamente avaliadas para sustentar os seus serviços de voz e SMS. Possui também uma *API* de *feedback* única que é usada para monitorizar ativamente a qualidade do seu serviço.

3.3.4 *JavaScript*

Javascript é uma linguagem dinâmica de programação com a capacidade de inserir código numa página *web*. Sendo bastante leve, é normalmente utilizada em implementações cujo *script* do lado do cliente interaja com o utilizador e faça páginas dinâmicas. É normalmente colocado num ficheiro *HTML* e é executado diretamente da página *web* [20].

3.3.5 *HTML*

HTML é uma linguagem com a qual se definem páginas *web*. Basicamente trata-se de um conjunto de etiquetas (*tags*) que servem para definir de que forma será apresentado o texto e outros elementos da página. É constituída por códigos que delimitam conteúdos específicos, segundo uma sintaxe própria e que definem o tipo de letra, o tamanho, cor, espaçamento, e vários outros aspetos de uma página *web*.

3.3.6 *AJAX*

O *AJAX* permite criar aplicações *web* mais rápidas e interativas, devido à sua ligação com o *JavaScript*. Grande parte das aplicações *web* transportam informações para e do servidor usando solicitação síncrona. Isto significa que para o utilizador enviar e receber informações de um servidor é sempre necessário atualizar a página ou ser redirecionado para outra. Com o *AJAX*, sempre que é enviada informação para um servidor, o *JavaScript* faz uma solicitação ao servidor, interpreta os resultados e atualizar a *interface* atual, sem haver necessidade de atualizar a página [21].

3.3.7 Node.js

O *Node.js* é um *runtime environment* em código aberto para programação em *JavaScript* do lado do servidor, graças ao *engine Chrome V8*. É um sistema que usa programação baseada em eventos para criar aplicações escaláveis e programas de rede sendo especialmente útil para a construção de servidores *web*. Oferece utilidades que anteriormente apenas eram possíveis com *PHP* (*Hypertext Preprocessor*) ou *Ruby*, como aceder aos ficheiros armazenados no computador, comunicar com *HTTP requests* na máquina, aceder diretamente a bases de dados, etc.
Também permite a incorporação de módulos para expansão de funcionalidades e acelerar o desenvolvimento de aplicações através de *frameworks*.

3.3.8 *Ngrok*

O *Ngrok* é um *software* multi-plataforma que permite a geração de um *endpoint* público para um servidor local por tunelação. Essencialmente o *Ngrok* cria um URL através do qual qualquer pessoa pode aceder ao nosso *localhost*. Ao contrário de outras plataformas como o *Heroku*, a função do *Ngrok* é, indiretamente, criar uma ligação ao servidor local, não ocorrendo qualquer *deploy* da nossa aplicação para a *web*, como é demonstrado na figura abaixo.

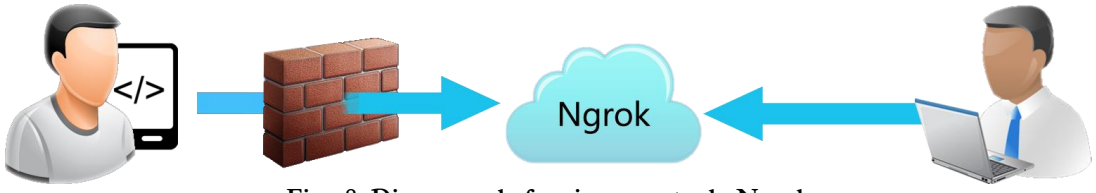

Fig. 8: Diagrama de funcionamento do Ngrok

3.3.9 *Git/GitHub*

O *Git* é uma ferramenta de controlo de versões de código, criando um histórico de modificações.
O *GitHub* é uma plataforma que permite o armazenamento desse código na *cloud*, oferecendo um sistema de controlo de versões, acessível em qualquer lado, o que o torna uma grande ferramenta para trabalhos em equipa.
Cada projeto é guardado num repositório, onde são armazenados todos os ficheiros. Por padrão o repositório cria automaticamente um ramo (*branch*) com o nome "*master*", que é considerado o ramo definitivo. Outros *branches* são usados para fazer testes e edições antes de dar *commit* para o *branch* master.

3.3.10 *Heroku*

O *Heroku* é um *PaaS* que oferece ferramentas de *deploy*, gestão e execução de aplicações desenvolvidas em *Ruby*, *Node.js*, *Java*, *Python*, *Scala*, *Go* e *PHP*. A sua integração com o *GitHub* possibilita uma fácil implementação de aplicações *online*, criando várias versões, o que acaba também por funcionar como *backup* redundante, visto o *GitHub* ser a alternativa predileta para *backups* e consultas de versões [22]. No *Heroku* existem duas possibilidades de *deploy* através do Github:

- ***Deploy* manual** - opção adotada para este projeto, através de ordem manual, faz *deploy* imediato de qualquer *branch* do repositório *GitHub* ligado à aplicação. Também pode ser usado para implementar temporariamente um *branch* que não o configurado para *deploy* automático (para fins de teste);
- ***Deploy* automático** - qualquer novo *push* para um *branch* do *GitHub*, é automaticamente implementado pelo *Heroku*. Existem medidas de precaução que podem ser ativadas, para limitar estes *deploys*, com base no contexto do *commit* que foi feito.

No capítulo seguinte serão abordados o planeamento e a concepção do projeto, descrevendo as suas funcionalidades, casos de uso e seu contexto, e diagramas referentes à instalação.

4. Planeamento e concepção

Na fase de planeamento do projeto foi importante ter em conta que usos é que os futuros utilizadores lhe vão dar, para isso foram feitas várias reuniões com o supervisor do estágio, de forma a que pudessem ser tiradas conclusões acerca do funcionamento da aplicação.

4.1 Funcionalidades e objetivos

A aplicação deverá poder interagir com três tipos de utilizadores: o Administrador que deverá ter capacidade para criar e eliminar utilizadores do tipo "Operador" e adicionar, editar ou eliminar opções *IVR*; o Operador que poderá fazer pesquisas *inbound*/*outbound*, gerar relatórios, fazer chamadas *outbound* e aceitar tarefas; e o Cliente, que poderá fazer chamadas *inbound*, fazer pedidos de *callback*, iniciar *live-chat* e iniciar videochamadas.

4.2 Desenho e modelação

De modo a estruturar e clarificar os atributos descritos anteriormente, foram usadas práticas, diagramas e símbolos gráficos de modelação descritos na linguagem *UML* (*Unified Modeling Language*), de forma a uniformizar o projeto através do emprego de componentes genéricos que contribuem para a melhor compreensão e detalhamento dos objetos e comunicação estipulada entre eles.

4.2.1 Diagrama de contexto

O diagrama seguinte demonstra o intuito e objetivo do projeto através de um esquema. No que toca ao desenvolvimento de *software*, é considerado o diagrama de fluxo de dados de maior nível, ou seja, o diagrama que representa todo o sistema.
Demonstra como todas as entidades interagem com o sistema indicando as suas possíveis ações. No diagrama de contexto não é considerada a estrutura interna do sistema, sendo o seu principal objetivo a deteção de limitações/requisitos.
Através dos requisitos enunciados anteriormente é então praticável a esquematização do diagrama da figura 9, onde se descreve a premissa da aplicação, com a indicação da direção dos movimentos de comunicação entre o sistema e os atores [23].

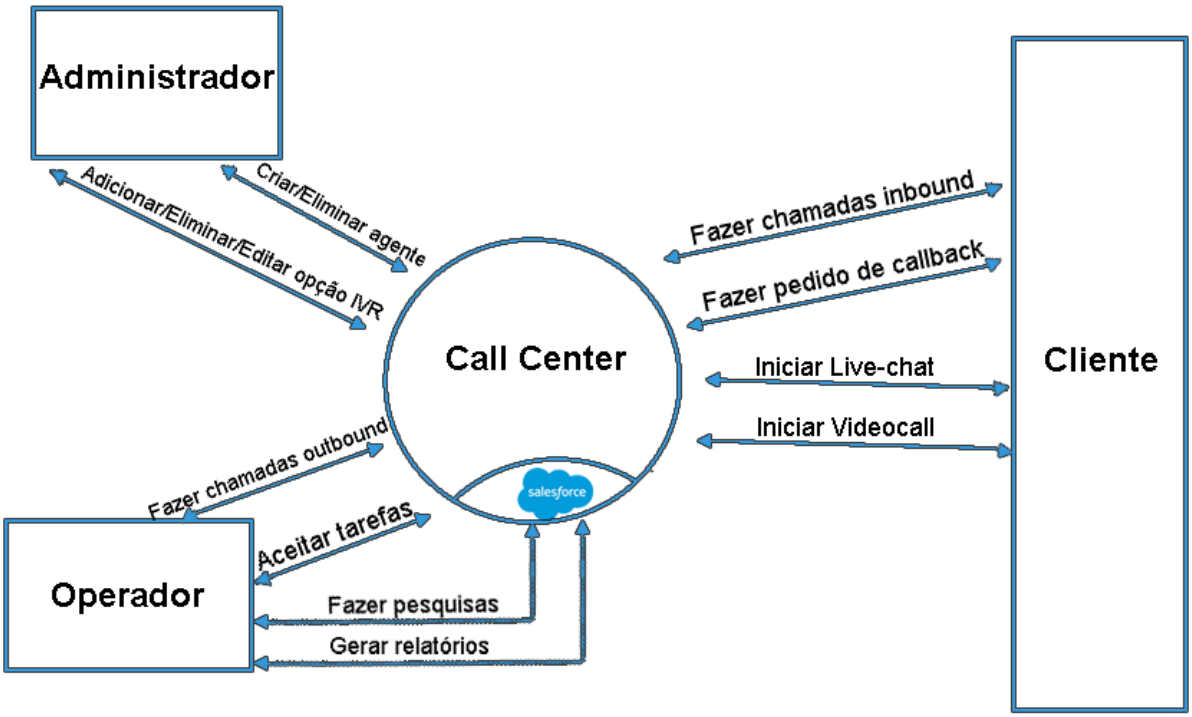

Fig. 9: Diagrama de Contexto

4.2.2 Diagrama de casos de uso

Este diagrama documenta o que o sistema faz do ponto de vista do utilizador. Noutras palavras, descreve a relação das funcionalidades do sistema com os utilizadores. Tal como no diagrama anterior, neste também não são aprofundados detalhes técnicos do sistema [24].
Os diagramas de casos de uso são compostos basicamente por quatro partes:

- **Cenário** - sequência de eventos que acontecem quando um utilizador interage com o sistema;
- **Ator** - tipo de utilizador do sistema;
- **Caso de uso** - tarefa ou uma funcionalidade realizada por um ator;
- **Comunicação** - é o que liga um ator a um caso de uso.

Também se destaca o conceito <<*include*>>, que significa que um caso de uso, para ter sua funcionalidade executada, precisa de chamar outro caso de uso.

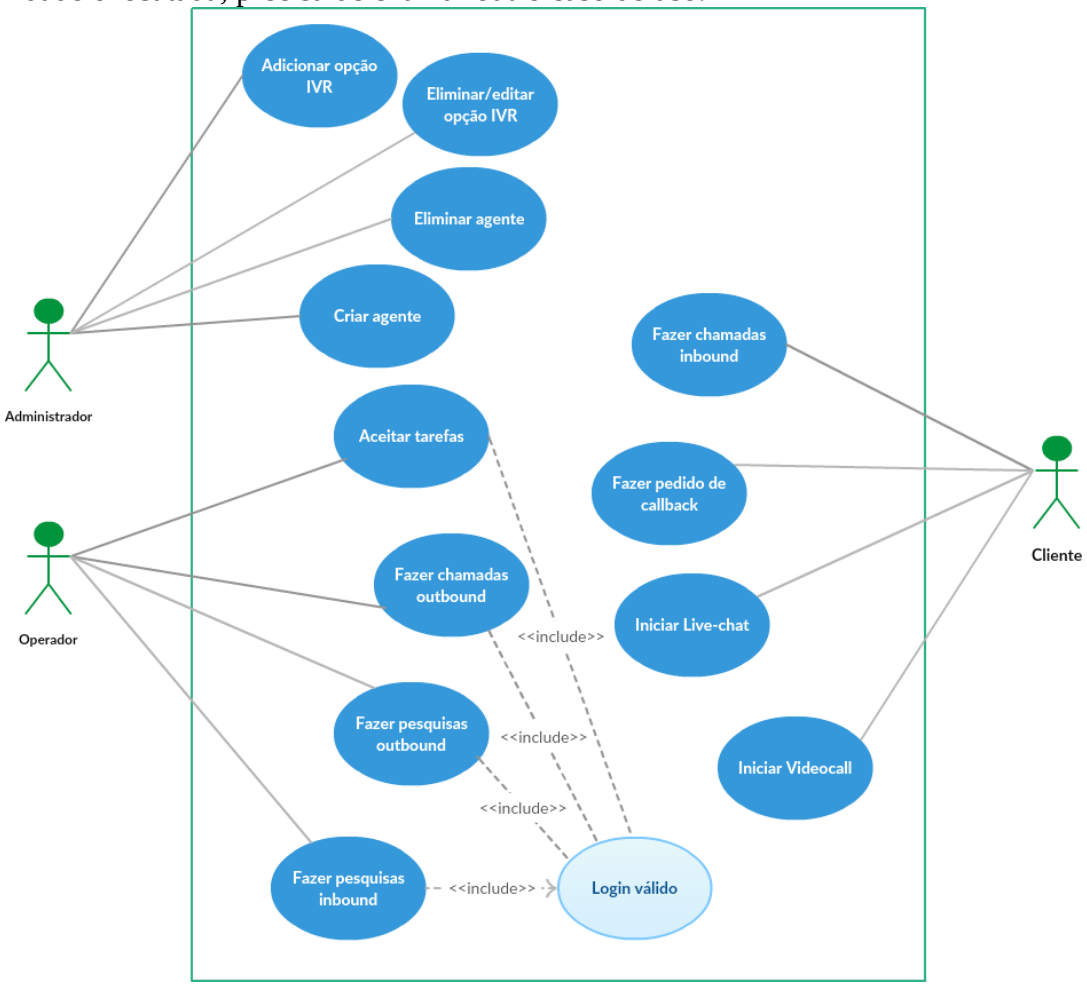

Fig. 10: Diagrama de casos de uso

4.2.3 Descrição de casos de uso e Diagramas de Sequência

A descrição de casos de uso, para além do já mencionado, permite achar caminhos, trajetos ou cenários alternativos e momentos de exceção. A descrição sequencial possibilita a descoberta de pré-condições ou requisitos anteriores à ocorrência, bem como a descoberta de sucessos que podem ocorrer após o término do caso de uso.
Os quadros a seguir, que descrevem os casos de uso referidos anteriormente, incluem referências a controladores, conteúdo que só é abordado no capítulo 5. Estão divididos em oito secções: nome, objetivo, atores envolvidos, pré-condição, fluxo principal, fluxos alternativos, pós-condição e casos de teste.

4.2.3.1 Fazer chamada *outbound*

A ideia original deste projeto teve como ponto orientador dois importantes casos de uso, sendo este um deles. A tabela seguinte descreve o caso de uso em que o Operador faz uma chamada de voz para o exterior. Pretende-se portanto dar ênfase a este momento, já que será uma das funcionalidades mais frequentes da aplicação.
Para esta operação o Operador, depois de ter o número que para o qual deseja ligar inserido no *dialer*, apenas tem de clicar no botão "*Call*".

Tabela 3: Caso de Uso – Fazer chamada *outbound*

Nome	Fazer chamada *outbound*
Objetivo	Fazer chamada de voz para um Cliente
Atores envolvidos	Operador
Pré-condição	*Login* válido
Fluxo Principal	1. O ator insere o número para o qual quer ligar no *dialer* e clica no botão "*Call*"
	2. O controlador *Phone* estabelece chamada
Fluxos alternativos	2a. Os dados do *setup* não são válidos, chamada anulada
	2b. O ator não tem ligação à base de dados, chamada anulada e mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se *setup* é válido

Dada a análise anterior torna-se possível chegar a um diagrama de sequência que resulta da descrição do caso de uso, onde figura a interface do *dialer*, o controlador do *Setup* e o controlador do *Phone*. Antes de qualquer função que usufrua dos serviços do *Twilio*, é sempre necessária fazer uma validação dos dados de configuração, visto o *Twilio* ser um serviço pago.

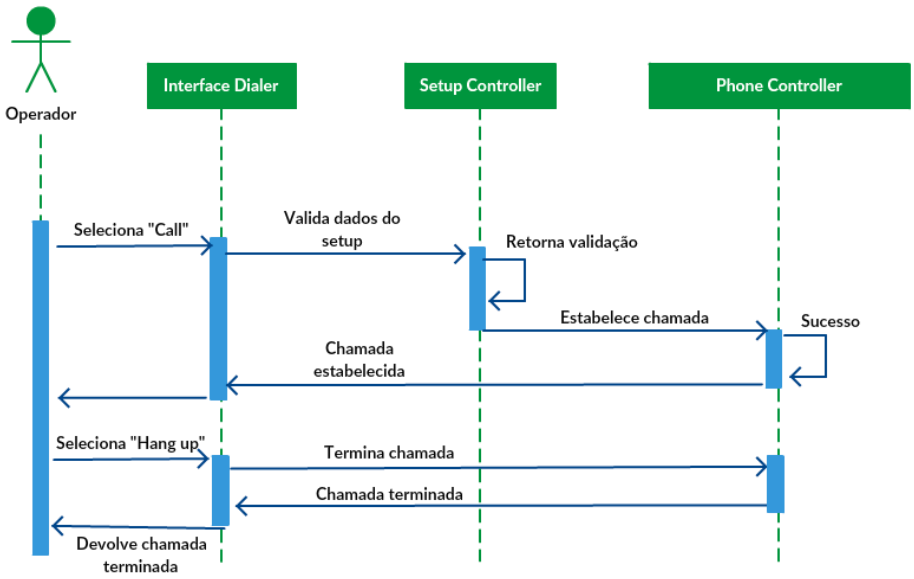

Fig. 11: Diagrama de Sequência – Fazer chamada *outbound*

4.2.3.2 Fazer pesquisa *outbound*

A tabela seguinte descreve o caso de uso em que o Operador faz uma pesquisa na base de dados do *Salesforce*, de modo a recolher o número de telefone de um certo Cliente a quem queira fazer uma chamada *outbound* representada anteriormente. Neste caso de uso opera-se tanto com a aplicação de *Call Center*, como com o próprio *Salesforce*, visto que assim que o utilizador tem acesso ao número desejado, muda de *interface*. É fundamental que seja um processo simples e com poucos passos. Chegou-se então a uma solução compreensível e de fácil execução, com base no uso do *Clipboard*. O Operador faz *search* do contacto ao qual quer fazer a chamada, e depois de ser selecionado, cola o número no *dialer* para estabelecer chamada.

Tabela 4: Caso de Uso – Fazer pesquisa *inbound*

Nome	Fazer pesquisa *outbound*
Objetivo	Fazer pesquisa de Cliente para copiar o seu número
Atores envolvidos	Operador
Pré-condição	*Login* válido
Fluxo principal	1. O ator escreve o nome ou parte do número na barra de pesquisa e clica no botão de procura
	2. O sistema retorna Clientes referentes à pesquisa
	3. O ator copia o contacto e cola no *dialer*
Fluxos alternativos	2a. O ator não tem ligação à base de dados, mensagem de erro
Pós-condição	Não existe
Casos de teste	Não existe

No diagrama de sequência seguinte estão então explícitas as interfaces do *Salesforce* e o *dialer* da aplicação.

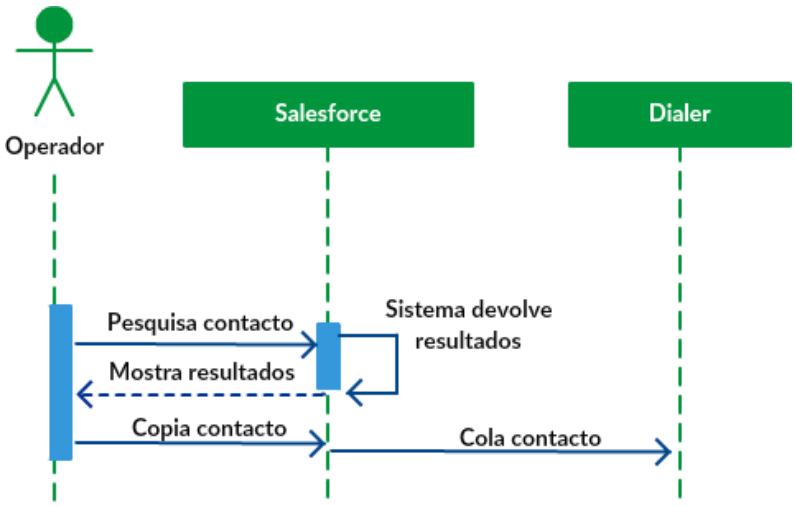

Fig. 12: Diagrama de Sequência -Fazer pesquisa *outbound*

4.2.3.3 Aceitar tarefa

As tarefas são aquilo que condiciona o estado do Operador e são desencadeadas através de um pedido de um Cliente. Sempre que um Cliente faz uma chamada *inbound*, faz um pedido de *callback*, inicia um *Live-chat*, ou inicia uma videochamada, o Operador fica reservado para uma certa tarefa, que após aceitação, muda o estado do Operador para "*Busy*" e inicia um processo de comunicação direta com o Cliente. Apesar de ser um caso de uso simples, é um dos mais significativos para o bom funcionamento do *Call Center*, visto estar conjugado com vários casos de uso do Cliente, tendo de funcionar harmoniosamente.

Tabela 5: Caso de Uso – Aceitar tarefa

Nome	Aceitar tarefa
Objetivo	Aceitar tarefa que é reservada para o ator
Atores envolvidos	Operador
Pré-condição	*Login* válido
Fluxo Principal	1. O ator acede ao *dialer* em estado "*Idle*"
	2. O sistema delega uma tarefa ao ator
	3. O ator clica "*Accept*"
Fluxos alternativos	3a. O ator não tem ligação à rede, mensagem de erro
Pós-condição	Operador fica com estado "*Busy*" e entra em ligação com Cliente
Casos de teste	Não existe

No seguinte diagrama de sequência estão representados o *Workflow Controller* e a interface do *dialer*, através da qual o Operador pode aceitar tarefas.

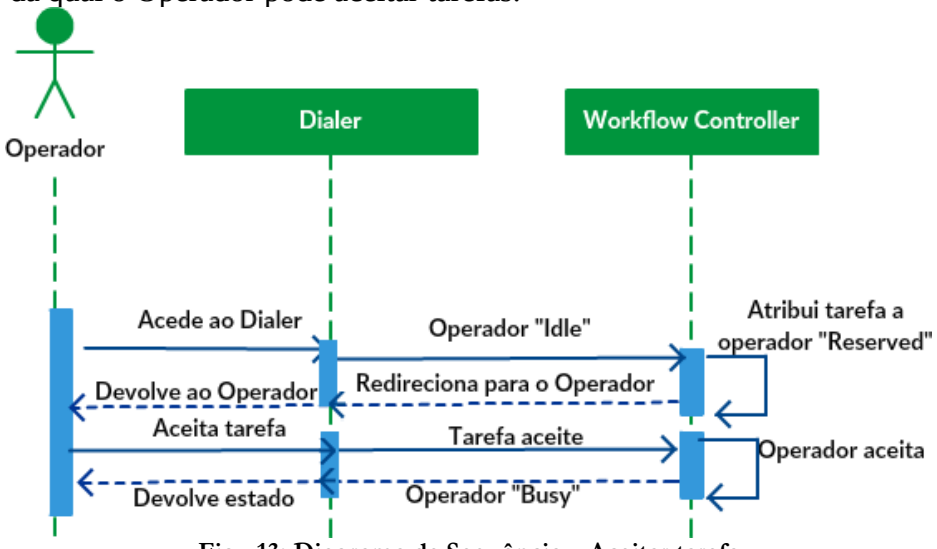

Fig. 13: Diagrama de Sequência – Aceitar tarefa

4.2.3.4 Fazer pesquisa *inbound*

A tabela seguinte descreve o caso de uso em que o Operador faz uma pesquisa na base de dados do *Salesforce*, de modo a identificar o cliente que estabeleceu contacto e abrir a sua ficha de cliente. De forma semelhante às pesquisas *outbound*, também se opera tanto com a aplicação de *Call Center*, como com o próprio *Salesforce*, em conjunção com o *Clipboard*. O Operador clica no botão de copiar dados do Cliente, de seguida cola na barra de pesquisa do *Salesforce* para verificar a identidade do Cliente. De seguida pode abrir a ficha do cliente, para obter detalhes adicionais.

Tabela 6: Caso de Uso – Fazer pesquisa *inbound*

Nome	Fazer pesquisa *inbound*
Objetivo	Fazer pesquisa de Cliente com o qual se está ligado para abrir ficha de Cliente
Atores envolvidos	Operador
Pré-condição	*Login* válido
Fluxo principal	1. O ator clica no botão de copiar contacto do *dialer*, cola na barra de procura e clica no botão de procura
	2. O sistema retorna Clientes referentes à pesquisa
	3. O ator seleciona o Cliente
Fluxos alternativos	2a. O ator não tem ligação à base de dados, mensagem de erro
Pós-condição	O sistema abre a ficha do Cliente
Casos de teste	Não existe

No diagrama de sequência seguinte estão então explícitas as interfaces da aplicação e do próprio *Salesforce*.

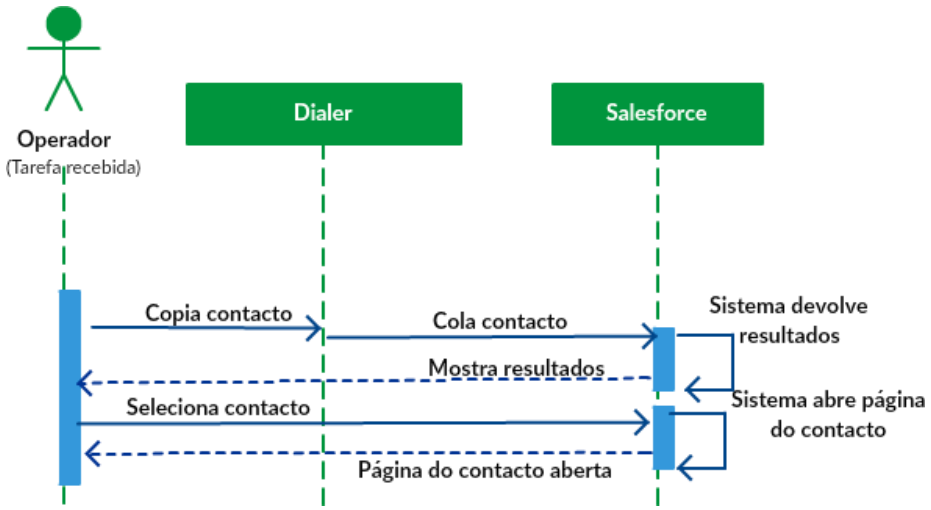

Fig. 14: Diagrama de Sequência – Fazer pesquisa *inbound*

4.2.3.5 Fazer chamada *inbound*

Este trata-se do segundo caso de uso mais importante relativamente à orientação e objetivo principal deste projeto: a receção de chamadas *inbound* de Clientes, organizadas através de um sistema *IVR*. Neste caso de uso, o Cliente faz uma chamada de voz para o número definido no *setup*, à qual é saudado com uma mensagem de áudio que o informa dos departamentos de atendimento disponíveis. Após a escolha do departamento que deseja contactar através de *input* numérico, é dada a opção de ficar em lista de espera, ou a receção de uma chamada de volta assim que houver um Operador vago. Em qualquer uma das opções é criada uma tarefa, que muda o estado do primeiro Operador vago do departamento escolhido para "*Reserved*", porém, caso a opção escolhida seja o *callback*, o caso de uso do Cliente termina, sendo entregue uma nova tarefa ao Operador, gerando o caso de uso de "Aceitar tarefa".
Se escolher a opção de entrar em lista de espera, a chamada continua, e o cliente terá de aguardar até algum Operador se encontrar inocupado para estabelecer ligação.

Tabela 7: Caso de Uso – Fazer chamada *inbound*

Nome	Fazer chamada *inbound*
Objetivo	Fazer chamada de voz para um Operador
Atores envolvidos	Cliente, Operador
Pré-condição	Não existe
Fluxo Principal	1. O Cliente faz uma chamada telefónica para o número definido no *setup* da aplicação
	2. O sistema estabelece chamada e devolve áudio do *IVR*
	3. O Cliente prime uma tecla para escolher departamento
	4. O sistema devolve aúdio do *IVR*
	5. O Cliente escolhe uma opção (Lista de espera)
	6. Sistema cria tarefa, redireciona para *Workflow* e delega tarefa a Operador "*Reserved*"
	7. Operador aceita tarefa
	8. Sistema muda estado do Operador para "*Busy*" e estabelece ligação Cliente/Operador
Fluxos alternativos	2a. Os dados do *setup* não são válidos, chamada anulada
	2b. Não existe ligação à aplicação, chamada anulada e mensagem áudio de erro
	3a. O Cliente escolhe um departamento inválido, sistema devolve áudio *IVR* de novo
	3b. O Cliente não escolhe um departamento, sistema devolve áudio *IVR* de novo
	5a. Cliente escolhe *callback*, chamada terminada e sistema cria tarefa de *callback* para primeiro Operador vago

Pós-condição	Não existe
Casos de teste	1. Verificar se *setup* é válido
	2. Verificar se opção inserida é válida

Visto ser um caso de uso que trabalha com chamadas de voz, *IVR* e estados dos Operadores, temos que considerar os controladores de *Phone*, *IVR*, e *Workflow*, para além do de Setup.
Em seguida é apresentado o diagrama de sequência para a primeira possibilidade de escolha do Cliente (lista de espera).

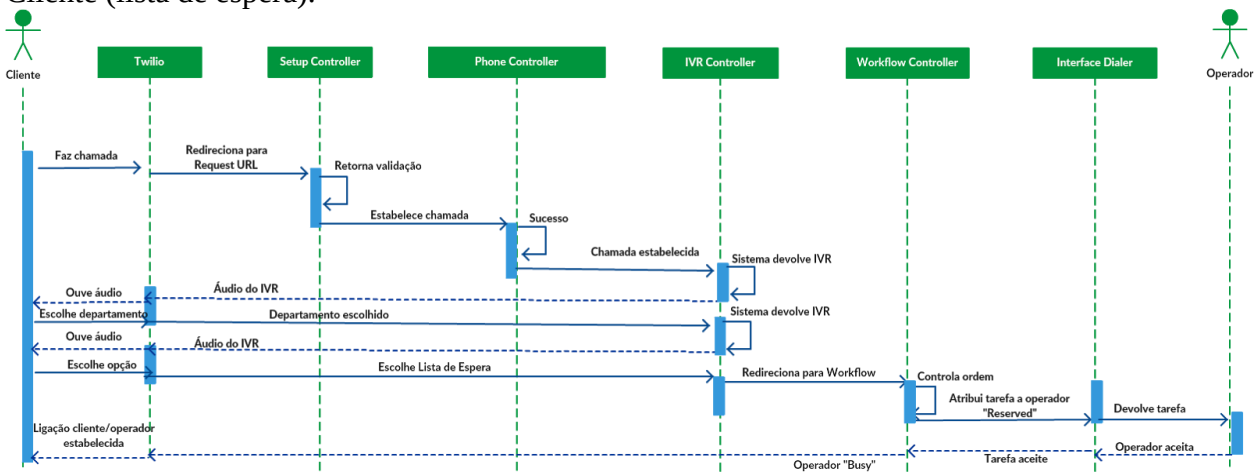

Fig. 15: Diagrama de Sequência – Fazer chamada inbound (Lista de Espera)

O diagrama de sequência seguinte apresenta a segunda possibilidade de escolha do Cliente (callback).

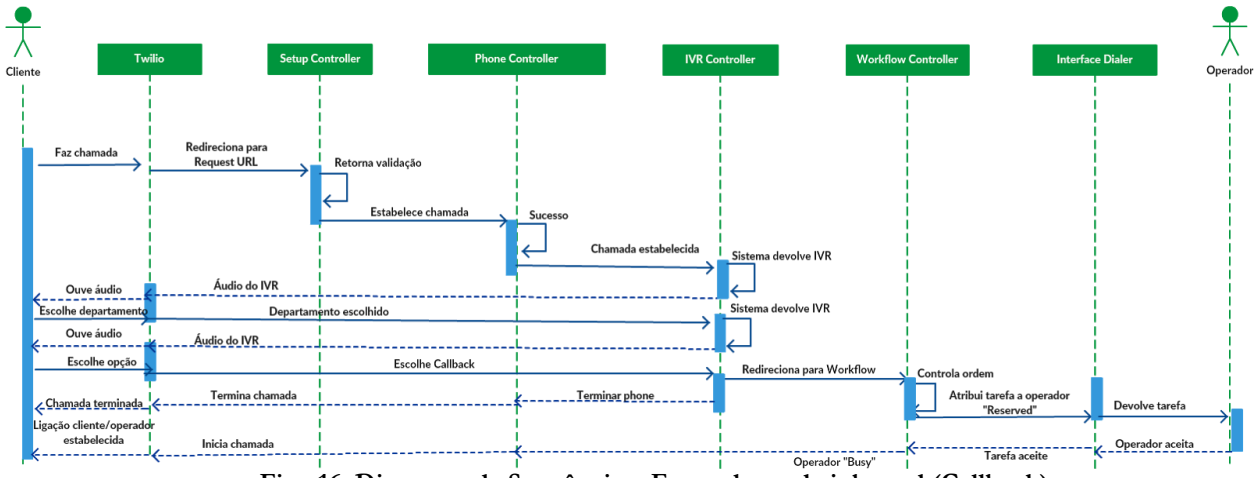

Fig. 16: Diagrama de Sequência – Fazer chamada inbound (Callback)

4.2.3.6 Fazer pedido de *callback* por formulário

A seguinte tabela retrata o caso de uso em que o Cliente pede uma chamada para o seu número assim que algum Operador se encontrar disponível. A diferença entre este *callback* e o anteriormente descrito é que este é solicitado através do *browser*, enquanto que no antecedente o *callback* é solicitado por *input* durante o processo de *IVR*.
Para fazer o pedido o Cliente tem de escolher o departamento que quer contactar, expôr a sua questão, identificar-se e fornecer o número para o qual solicita contacto. Assim que todos estes dados estiverem validados o Cliente pode submeter o seu pedido, e automaticamente é criada uma tarefa. Assim que o Operador aceita é automaticamente feita uma ligação por chamada de voz ao Cliente.

Tabela 8: Caso de Uso – Fazer pedido de *callback*

Nome	Fazer pedido de *callback*
Objetivo	Fazer pedido de chamada de volta para primeiro Operador vago
Atores envolvidos	Cliente, Operador
Pré-condição	Não existe
Fluxo Principal	1. O Cliente submete dados na *interface* de *callback*
	2. O sistema valida os dados inseridos, valida os dados do *setup*, cria tarefa, redireciona para *Workflow* e delega tarefa a Operador "*Reserved*"
	3. Operador aceita tarefa
	4. Sistema muda estado do Operador para "*Busy*" e estabelece ligação por chamada de voz Operador/Cliente
Fluxos alternativos	2a. Dados inseridos não são válidos, mensagem de erro
	2b. Os dados do *setup* não são válidos, mensagem de erro
	2c. Não existe ligação à aplicação, mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se dados do Cliente são validados corretamente
	2. Verificar se *setup* é válido

No diagrama de sequência seguinte demonstra-se que o *Phone Controller* é o último a ser chamado, isto porque só é estabelecida chamada depois da validação, de ser criada a tarefa, e desta ser atribuída e aceite.

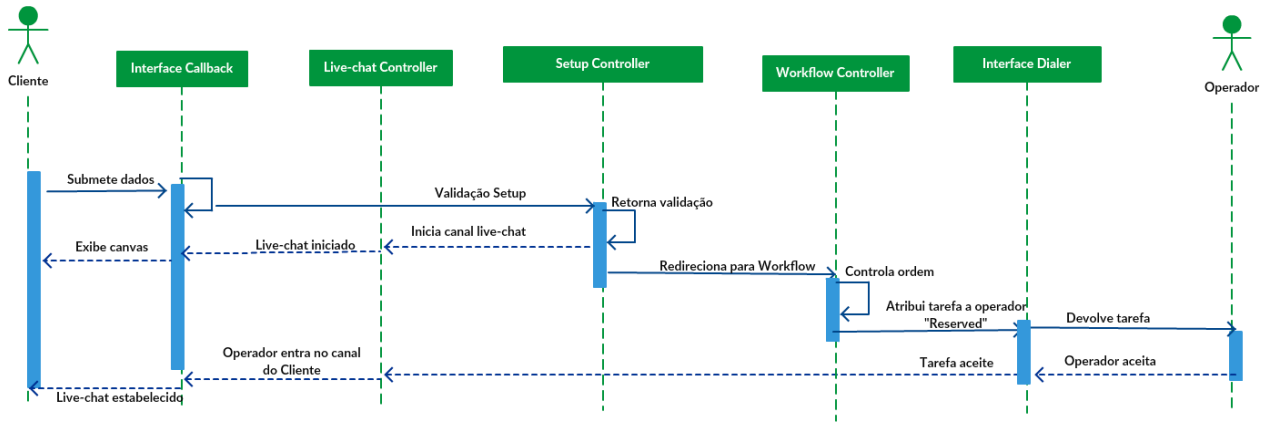

Fig. 17: Diagrama de Sequência – Pedido de *callback* por formulário

4.2.3.7 Iniciar *Live-chat*

A seguinte tabela retrata o caso de uso em que o Cliente inicia uma sessão de *Live-chat*.
Para isso o Cliente tem de identificar-se e assim que os dados estiverem validados o Cliente pode submeter o seu pedido, e automaticamente é criada uma tarefa. O Cliente é redirecionado para uma *interface* de chat e é avisado assim que um Operador que tenha aceitado a tarefa entre na conversação.

Tabela 9: Caso de Uso – Iniciar Live-chat

Nome	Iniciar *Live-chat*
Objetivo	Iniciar sessão de *live-chat* com Operador
Atores envolvidos	Cliente, Operador
Pré-condição	Não existe
Fluxo Principal	1. O Cliente submete dados na interface de *Live-chat*
	2. O sistema valida os dados inseridos, valida os dados do *setup*, cria tarefa, redireciona para *Workflow* e delega tarefa a Operador "*Reserved*"
	3. Operador aceita tarefa
	4. Sistema muda estado do Operador para "*Busy*" e estabelece ligação por *live-chat* Operador/Cliente
Fluxos alternativos	2a. Dados inseridos não são válidos, mensagem de erro
	2b. Os dados do *setup* não são válidos, mensagem de erro
	2c. Não existe ligação à aplicação, mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se dados do Cliente são validados corretamente
	2. Verificar se *setup* é válido

No diagrama de sequência seguinte demonstra-se que é criado um canal ainda antes de a tarefa ser atribuída. Desta forma o Cliente fica já inserido num canal criado, onde aguarda a chegada de um Operador.

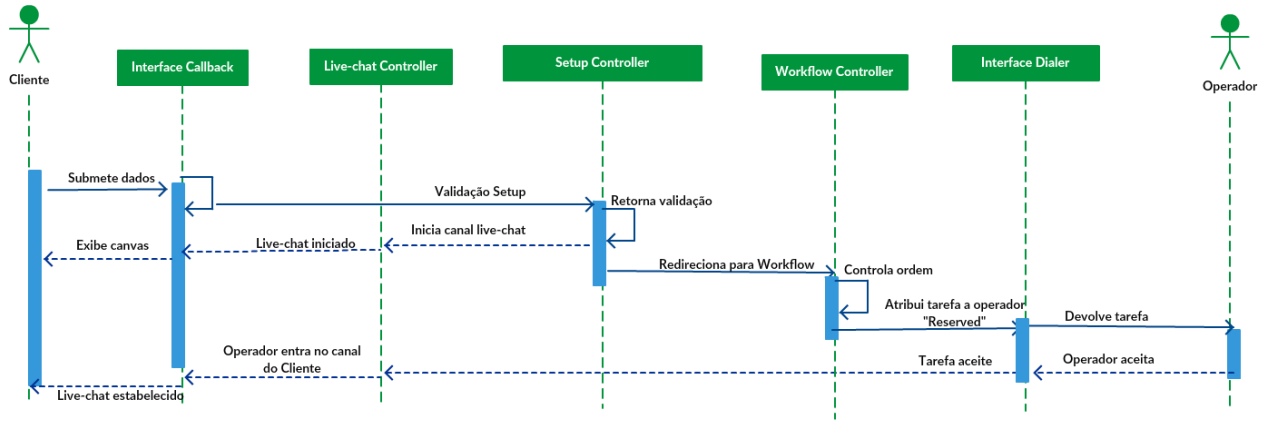

Fig. 18: Diagrama de Sequência – Iniciar *Live-chat*

4.2.3.8 Iniciar Videochamada

A seguinte tabela retrata o caso de uso em que o Cliente inicia uma videochamada.
Para isso o Cliente tem de identificar-se e assim que os dados estiverem validados o Cliente pode submeter o seu pedido, e automaticamente é criada uma tarefa. O cliente é redirecionado para uma *interface* de videochamada e é avisado assim que um Operador que tenha aceitado a tarefa entre na conversação.

Tabela 10: Caso de Uso – Iniciar Videochamada

Nome	Iniciar Videochamada
Objetivo	Iniciar sessão de videochamada com Operador
Atores envolvidos	Cliente, Operador
Pré-condição	Não existe
Fluxo Principal	1. O Cliente submete dados na interface de videochamada
	2. O sistema valida os dados inseridos, valida os dados do *setup*, cria tarefa, redireciona para *Workflow* e delega tarefa a Operador "*Reserved*"
	3. Operador aceita tarefa
	4. Sistema muda estado do Operador para "*Busy*" e estabelece ligação por videochamada Operador/Cliente
Fluxos alternativos	2a. Dados inseridos não são válidos, mensagem de erro
	2b. Os dados do *setup* não são válidos, mensagem de erro
	2c. Não existe ligação à aplicação, mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se dados do Cliente são validados corretamente
	2. Verificar se *setup* é válido

No diagrama de sequência seguinte, tal como com o *Live-chat*, demonstra-se que é criado um canal ainda antes de a tarefa ser atribuída. Desta forma o Cliente fica já inserido num canal criado, onde aguarda a chegada de um Operador.

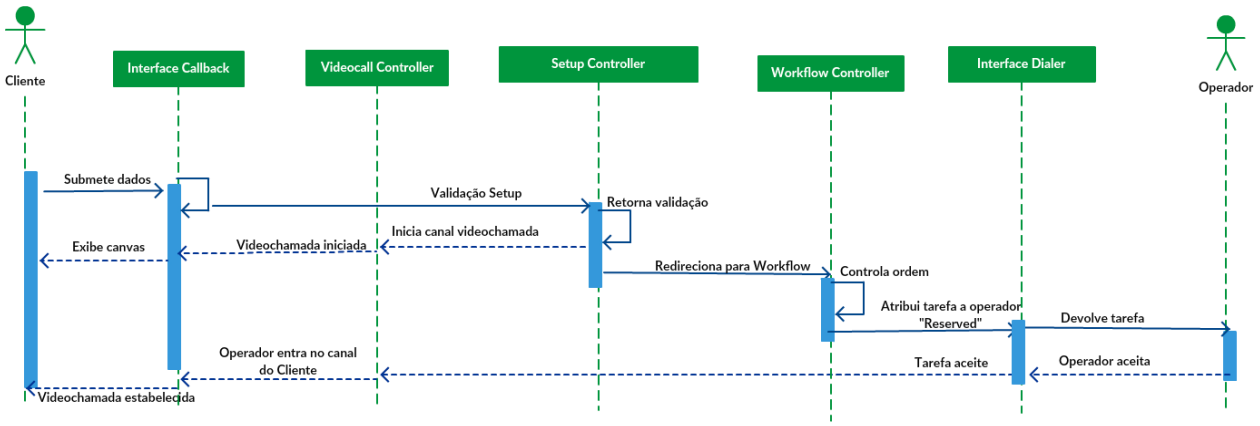

Fig. 19: Diagrama de Sequência – Iniciar videochamada

4.2.3.9 Adicionar opção *IVR*

A seguinte tabela retrata o caso de uso em que o Administrador adiciona uma nova opção para o sistema *IVR*, ou seja um novo departamento, providenciando mais uma possibilidade de escolha ao Cliente. Para isso depois de abrir a interface que permite a adição de uma nova opção *IVR*, apenas tem decidir a sua designação, e a que tecla está atribuída.

Tabela 11: Caso de Uso – Adicionar opção IVR

Nome	Adicionar opção *IVR*
Objetivo	Adicionar uma nova opção no sistema *IVR*
Atores envolvidos	Administrador
Pré-condição	Não existe
Fluxo Principal	1. O ator acede à página de administração
	2. O sistema devolve lista de *workers* e de opções *IVR*
	3. O ator acede à *Interface* "*Add IVR Option*"
	4. O sistema devolve formulário
	5. Ator preenche campos e submete
	6. Sistema valida inserção
Fluxos alternativos	1a. Não existe ligação à aplicação, mensagem de erro
	6a. Inserção inválida, mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se dados são validados corretamente

No diagrama de sequência seguinte constatamos que também é feito um *request* aos *workers*, devido ao facto de a interface de adição/remoção de Operadores e de opções *IVR* se encontrarem agregadas.

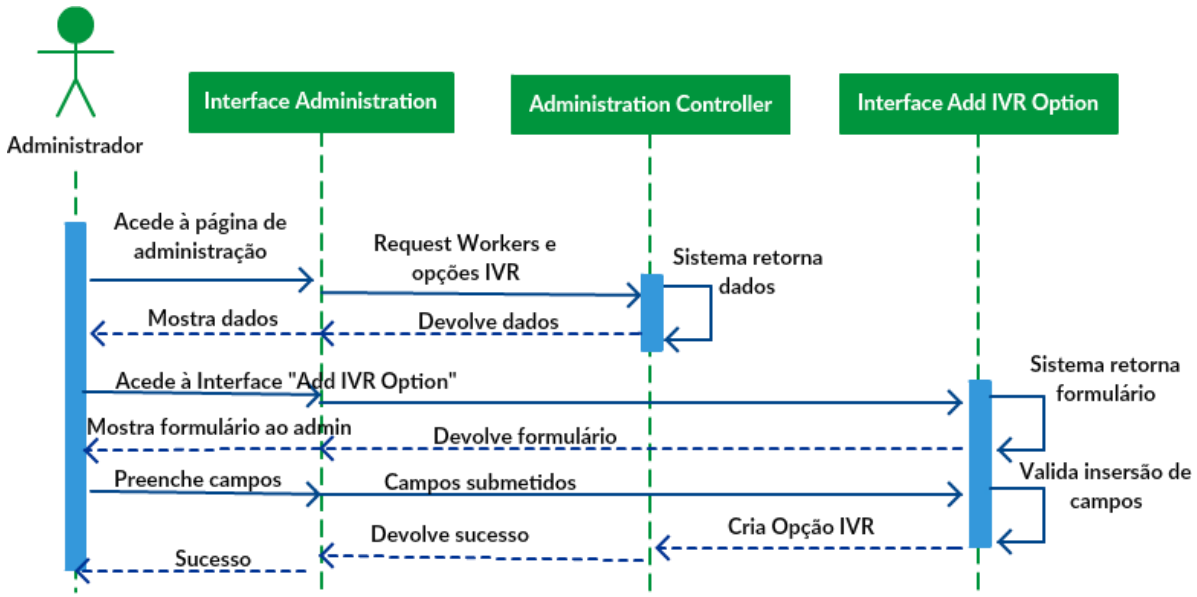

Fig. 20: Diagrama de Sequência – Adicionar opção IVR

4.2.3.10 Remover/editar opção *IVR*

A seguinte tabela retrata o caso de uso em que o Administrador remove ou edita uma opção para o sistema *IVR*. Para isso depois de abrir a interface de administração apenas tem de escolher qual é a opção a remover/editar. Estas duas operações encontram-se aqui reunidas devido às características da *interface* de administração, em que o processo para edição, ou remoção de uma opção *IVR* encontra-se todo na mesma área o que origina um processo muito semelhante do ponto de vista do Administrador.

Tabela 12: Caso de Uso – Remover/editar opção IVR

Nome	Remover/editar opção *IVR*
Objetivo	Remover/editar opção do sistema *IVR*
Atores envolvidos	Administrador
Pré-condição	Não existe
Fluxo Principal	1. O ator acede à página de administração
	2. O sistema devolve lista de *workers* e de opções *IVR*
	3. O ator remove/edita opção *IVR* e submete
	4. Sistema valida inserção
Fluxos alternativos	1a. Não existe ligação à aplicação, mensagem de erro
	4a. Inserção inválida, mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se dados são validados corretamente

Tal como no diagrama de sequência anterior, constatamos que também é feito um *request* aos *workers*, devido ao facto de a interface de adição/remoção de Operadores e de opções *IVR* se encontrarem agregadas.

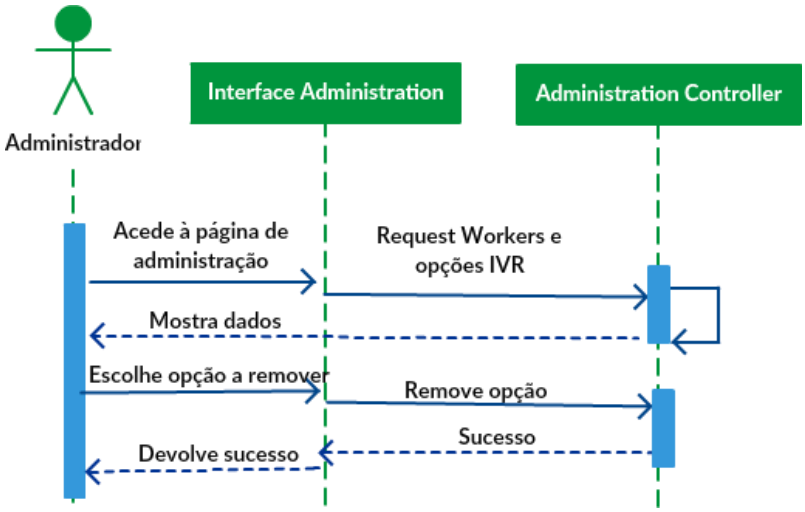

Fig. 21: Diagrama de Sequência – Remover opção IVR

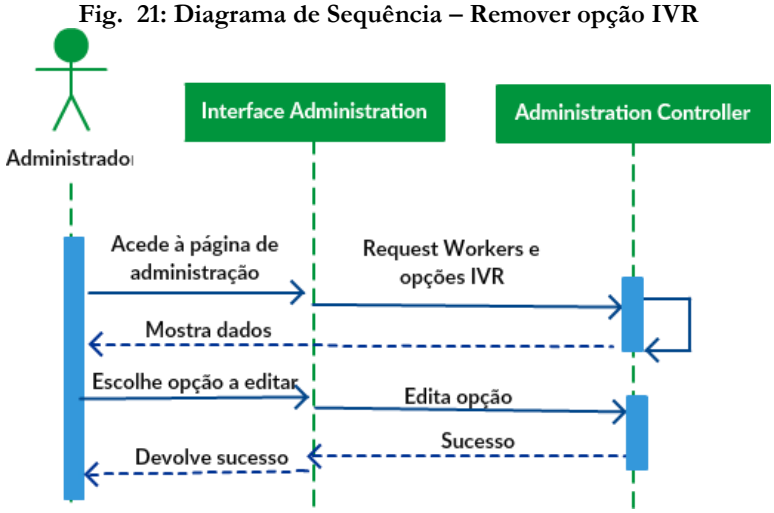

Fig. 22: Diagrama de Sequência – Editar opção IVR

4.2.3.11 Criar Operador

A seguinte tabela retrata o caso de uso em que o Administrador cria um novo agente ou Operador. Para isso depois de abrir a interface de administração que permite a criação de novos Operadores, apenas tem de decidir a sua designação, a que actividades tem acesso (*Phone, Chat, Video*) e a que departamento pertence. Assim que os dados estiverem validados o Administrador pode criar o novo Operador.

Tabela 13: Caso de Uso – Criar Operador

Nome	Criar Operador
Objetivo	Adicionar um novo Operador
Atores envolvidos	Administrador
Pré-condição	Não existe
Fluxo Principal	1. O ator acede à página de administração
	2. O sistema devolve lista de *workers* e de opções *IVR*
	3. O ator acede à Interface "*Create Agent*"
	4. O sistema devolve formulário
	5. Ator preenche campos e submete
	6. Sistema valida dados inseridos
Fluxos alternativos	1a. Não existe ligação à aplicação, mensagem de erro
	6a. Dados inseridos inválidos, mensagem de erro
Pós-condição	Sistema cria novo Operador
Casos de teste	1. Verificar se dados são validados corretamente

No diagrama de sequência seguinte constatamos o que foi anteriormente descrito, sendo feito um *request* das opções *IVR* em conjunto com os *workers*, devido ao facto de a interface de adição/remoção de Operadores e de opções *IVR* se encontrarem agregadas.

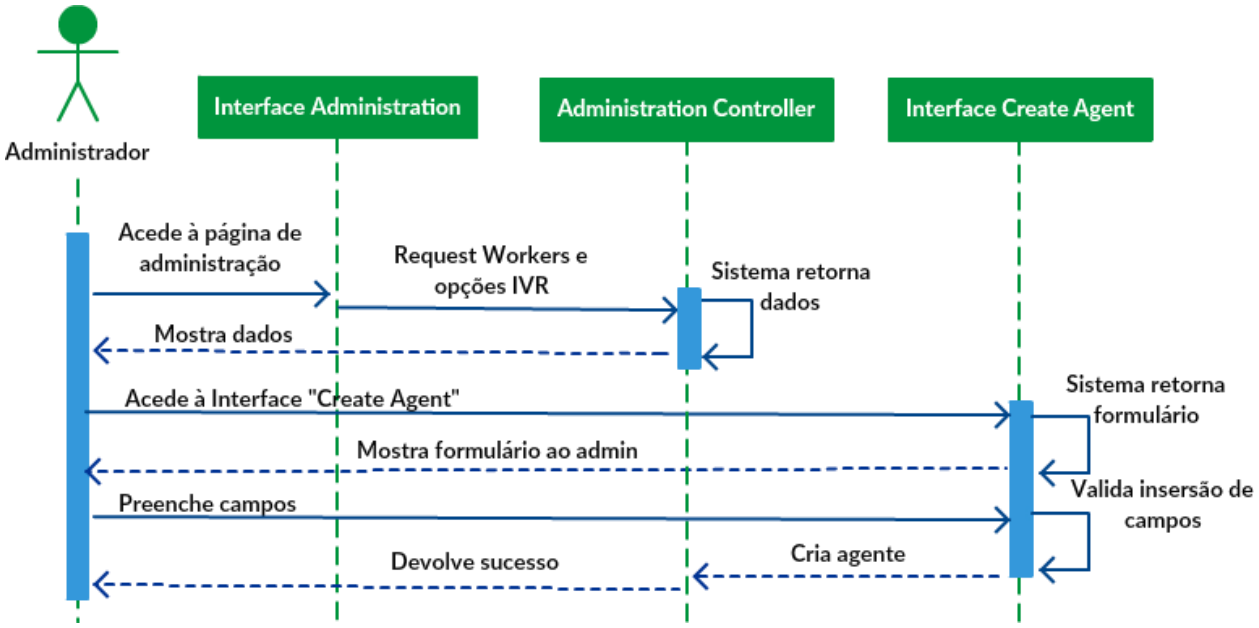

Fig. 23: Diagrama de Sequência – Criar Operador

4.2.3.12 Remover Operador

A seguinte tabela retrata o caso de uso em que o Administrador remove um agente ou Operador. Para isso depois de abrir a interface de administração onde são listados todos os Operadores, basta escolher aquele que quer remover e selecioná-lo para remoção.

Tabela 14: Caso de Uso – Remover Operador

Nome	Remover Operador
Objetivo	Remover Operador do sistema
Atores envolvidos	Administrador
Pré-condição	Não existe
Fluxo Principal	1. O ator acede à página de administração
	2. O sistema devolve lista de *workers* e de opções *IVR*
	3. O ator remove Operador
Fluxos alternativos	1a. Não existe ligação à aplicação, mensagem de erro
Pós-condição	Não existe
Casos de teste	1. Verificar se dados são validados corretamente

No diagrama de sequência seguinte constatamos o que foi anteriormente descrito, sendo feito um *request* das opções *IVR* em conjunto com os *workers*, devido ao facto de a interface de adição/remoção de Operadores e de opções *IVR* se encontrarem agregadas.

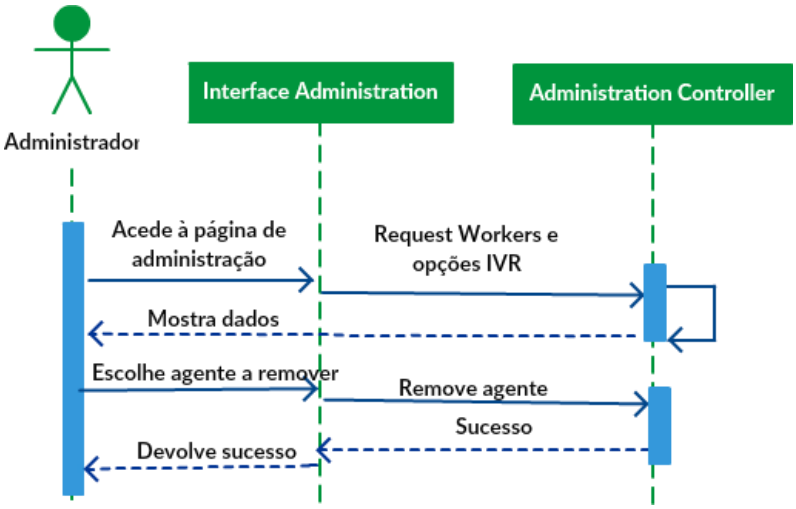

Fig. 24: Diagrama de Sequência – Remover Operador

4.3 Componentes e instalação

Neste subcapítulo são demonstrados e descritos diagramas relativos aos componentes do projeto e sua instalação.

4.3.1 Diagrama de componentes

Em *UML*, os diagramas de componentes mostram a estrutura do sistema, descrevendo os componentes do *software*. É possível utilizar diagramas de componentes para modelar sistemas de *software* num nível alto ou para mostrar componentes num nível de pacote mais baixo.
Este tipo de diagrama suporta o desenvolvimento com base em componentes no qual um sistema de *software* é dividido em componentes e interfaces que são reutilizáveis e substituíveis [25].
Os diagramas de componentes são úteis pelos seguintes motivos:

- Definir os aspetos executáveis e reutilizáveis de um sistema de *software*;
- Revelar problemas de configuração de *software* através de relacionamentos de dependência;
- Mostrar uma representação precisa de uma aplicação antes de fazer alterações ou aprimoramentos.

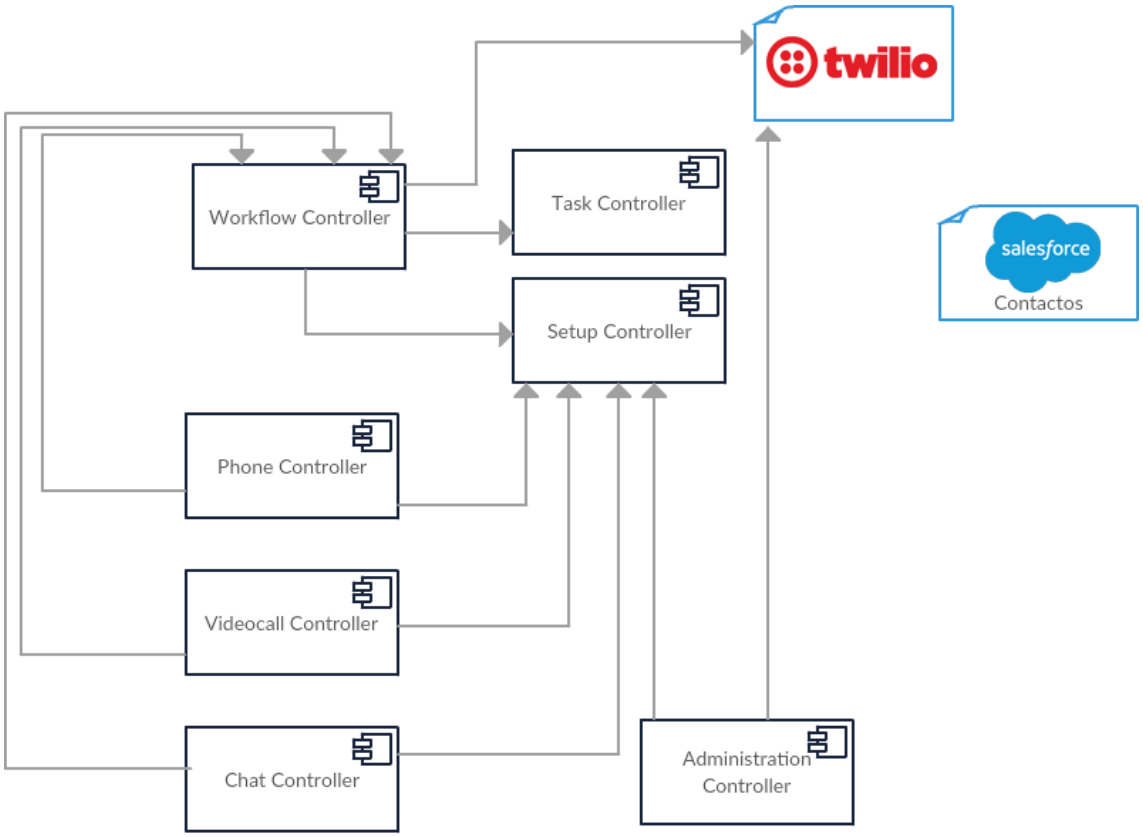

Fig. 25: Diagrama de componentes

4.3.2 Diagrama de instalação

Em *UML*, os diagramas de instalação modelam a arquitetura física de um sistema. Mostram os relacionamentos entre os componentes de *software* e *hardware* no sistema e a distribuição física do processamento.
Os diagramas de instalação, que normalmente são preparados durante a fase de desenvolvimento da implementação, mostram a organização física dos nós num sistema distribuído, onde são feitos armazenamentos e outros elementos que sejam implementados [26].

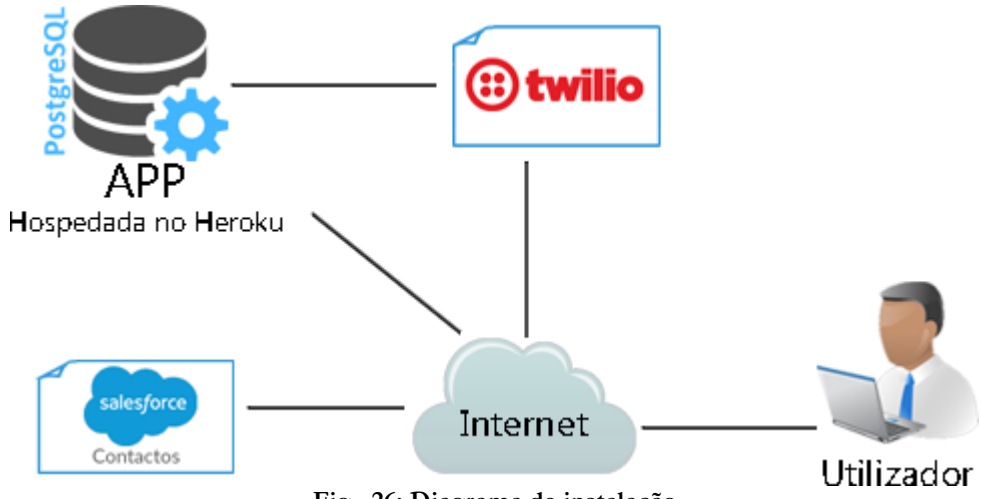

Fig. 26: Diagrama de instalação

5. Implementação e desenvolvimento

Para a criação da aplicação foi usado como alicerce o mais popular serviço do *Twilio*, o *Twilio Voice*, que oferece serviços de chamadas *inbound*/*outbound*, reprodução de áudio através de *text-to-speech* e *call queues*. O primeiro passo para a criação de uma aplicação com recurso ao *Twilio Voice* é a compra de um número, que no caso foi um número local português, que permite tanto chamadas *inbound* como *outbound*.
O *Twilio Voice* usa o protocolo *PSTN* (*Public Switched Telephone Network*) para as suas comunicações telefónicas. Isto permite a ocorrência de chamadas telefónicas processadas por código.

5.1 Controlador de *setup*

Para a utilização de qualquer serviço do *Twilio* é necessário que a conta registada esteja com saldo positivo. Este é o principal, mas não o único motivo devido ao qual é essencial fazer-se uma validação dos dados de *setup* da aplicação. Outros motivos são, por exemplo, a necessidade de verificar se o *Workspace* está ativo, se foi adicionado um novo estado de Operador, ou se o foi criado outro Chat *Service* [27]. Para que esta validação ocorra, foi criado um controlador de *setup*, que é chamado pelos restantes controladores sempre que é necessária esta validação. Este controlador, verifica inicialmente se houve algum *update* na aplicação desde a última validação de *setup*. Isto para que, caso seja o caso, possa sincronizar os *queues* e evitar falhas no *workflow* (abordados no cap.5.2), ou mesmo iniciá-los, caso seja a primeira vez que a aplicação é inicializada.
De seguida, dependendo da operação que irá suceder a respetiva validação, serão autenticados:

- ***Account SID*** - código único de identificação da conta do *Twilio*;
- ***Auth Token*** - código único de autenticação da conta do *Twilio*;
- ***Phone Number*** - número de telefone associado à aplicação;
- ***SID* do *Workspace*** - código único de identificação do *Workspace* (explicação no cap. 5.2);
- ***SID* do *Chat Service*** - código único de identificação do *Chat Service* (explicação no cap. 5.5);
- ***SID* da *Key* da *API* do *Twilio*** - código único de identificação, atribuído a desenvolvedores, para terem acesso a certas propriedades da *API* do *Twilio*;
- ***Secret* da *Key* da *API*** - código para validar o código anterior, só é visível no momento da criação da Key da *API*;
- **Estados dos Operadores** - validação dos estados possíveis dos Operadores (explicação no cap. 5.2).

Desta forma todas as *keys* e dados de *setup* precisos são validados, seguindo então para o passo que procede à validação do *setup*.

5.2 Operadores

Visto que um dos objetivos da aplicação era não apenas estabelecer chamadas simples "origem-destino", como também dar a possibilidade de através de um único número poder criar várias ligações de clientes para operadores e vice-versa, foi necessária a criação de um *Workflow* ou fluxo de trabalho. Trata-se de uma sequência de passos organizados de forma a automatizar um procedimento, que neste caso é a correta distribuição de tarefas pelos seus *queues* (ou filas). Neste

caso foi criado um *Workflow* através da ferramenta do *Twilio* mais adequada para este efeito, o *TaskRouter* [28].

5.2.1 *TaskRouter*

À primeira vista, fazer corresponder uma tarefa de um certo tipo ao Operador que lhe compete pode parecer um processo simples, mas a estruturação e criação do código para resolver este problema em larga escala pode tornar-se uma tarefa bastante demorada. Foi por isso usado o *TaskRouter*, uma ferramenta que organiza não só o *Workflow*, como o estado dos Operadores que nele atuam.
Através do *TaskRouter* é possível organizar vários canais de atendimento, divididos por canais (telefone, *chat* ou video), e relacionar cada ligação ao seu Operador respetivo através das suas competências e permissões [29].
O seu funcionamento é processado através das escolhas do Cliente. Assim que é feita uma ligação *inbound* à aplicação, o *TaskRouter* encarrega-se de criar uma tarefa que é dependente do meio de ligação do Cliente e/ou do departamento que pretende contactar. Desta forma são criadas várias "filas" de tarefas semelhantes, que são gradualmente redirecionadas para um Operador assim que se encontre vago, como é demonstrado no diagrama seguinte.

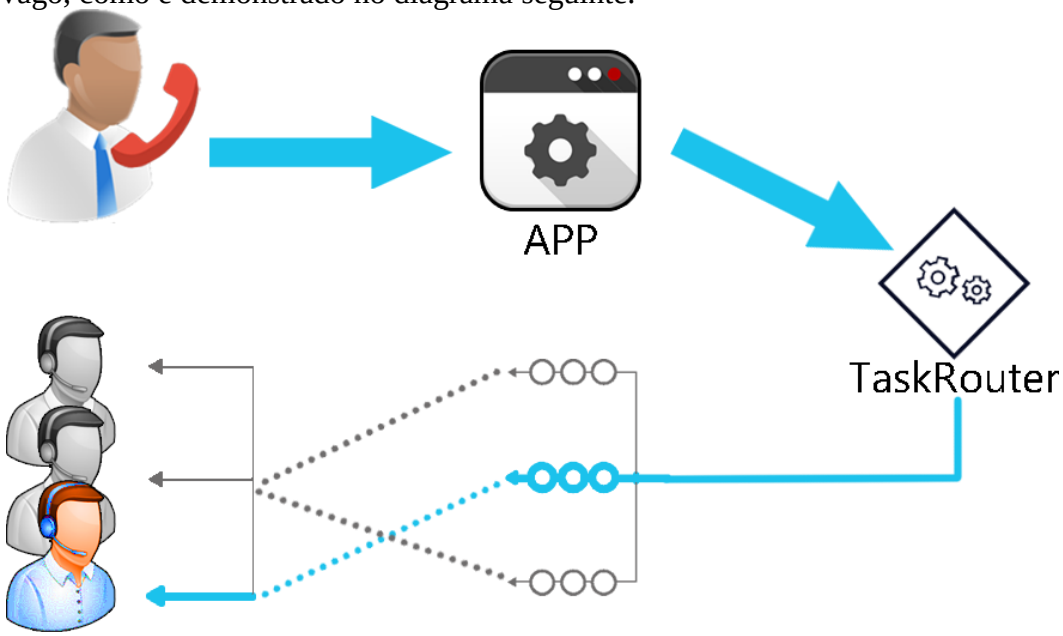

Fig. 27: Funcionamento do TaskRouter

Quando o *TaskRouter* seleciona um *Worker*, ele faz o seguinte:

1. Define o estado da atribuição da tarefa para "*reserved*";
2. Cria uma ligação da tarefa ao Operador;
3. Ao mesmo tempo que a reserva é criada, uma solicitação *POST* é feita ao *Workflow Controller* onde são incluídos os detalhes completos da tarefa, do trabalhador selecionado e da reserva.

Um Operador apenas pode atuar sobre uma tarefa de cada vez, podendo então ser representado por 4 estados possíveis, que se encontram representados na seguinte tabela:

Tabela 15: Estados do Operador

Estado	Descrição
Idle	Operador vago, à espera de tarefa.
Reserved	Operador reservado a uma certa tarefa, ainda não aceite.
Busy	Operador aceitou uma tarefa e ainda não completou.
Offline	Operador *offline*.

No diagrama seguinte é demonstrado de que maneira o estado do Operador pode variar.

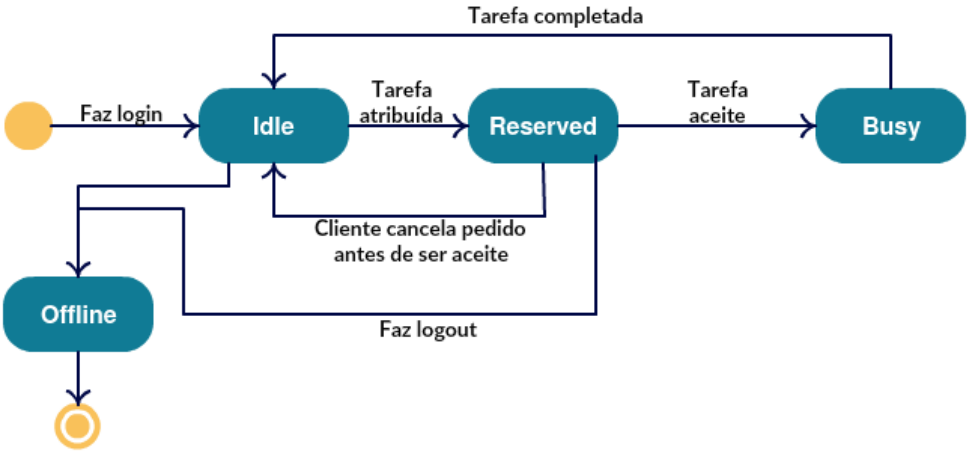

Fig. 28: Diagrama de Estados – Estados do Operador

Primariamente, através da consola do *Twilio* foi criado um *Workspace*, nome dado a cada ligação entre aplicação e o *TaskRouter*. Nele foram definidos parâmetros como o tempo limite de reserva de tarefas, o estado de Operador padrão ou a prioridade de ordem (*First In, First Out*). Foi então criado um controlador para ligar a aplicação ao *TaskRouter* denominado de *Workflow Controller*. O trabalho deste controlador consiste em estabelecer ligação com o *Workspace* para enviar os dados de cada nova ligação efetuada por um Cliente, para que desta forma o *TaskRouter* possa organizar os *queues*. Estes mesmos dados são controlados por um "sub-*controlador*" chamado *Task Controller*. Neste controlador é especificado se a tarefa criada advém de um pedido de *callback*, de uma conversação de *Live-chat*, de uma videochamada ou de uma chamada de voz *inbound* e também o departamento que se pretende contactar (se for o caso). Estes controladores estão constantemente a trocar informação com o *TaskRouter*, devido não só à progressão normal do *Workflow*, mas também devido à possibilidade de uma reserva poder dar *timeout* (devido a uma ligação fraca de uma das

partes), uma reserva ter sido cancelada por parte do Cliente, ou qualquer problema que não seja expectável, para impedir que um Operador fique "retido" a uma certa tarefa ou Cliente.
Na interface do *TaskRouter* no *Twilio*, é possível ver em tempo real o número de Operadores com *login* ativo e o seu respetivo estado, as tarefas em curso, a média de tempo de aceitação, para além de outras ferramentas úteis como a criação de gráficos (figura 29).

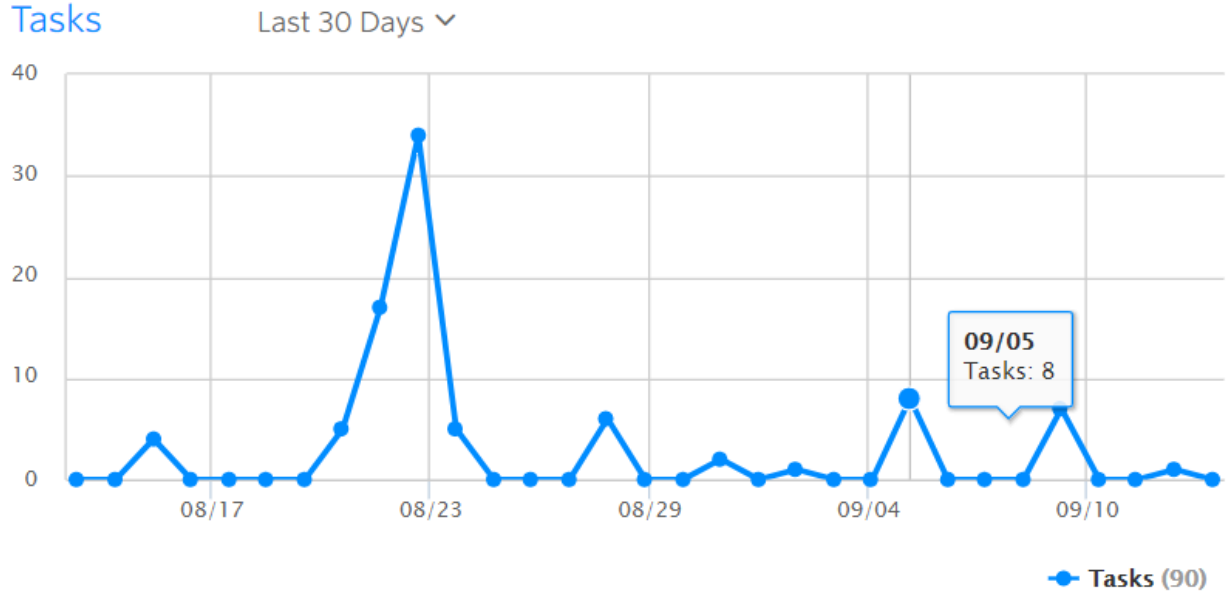

Fig. 29: Gráfico Gerado na consola do TaskRouter

5.2.2 *Login*

Para o login dos Operadores foi criado um controlador que após a confirmação de que o nome introduzido na interface de *login* está devidamente registado na base de dados (processo abordado no cap.5.6), faz uma ligação ao *Workflow Controller* de forma a alterar o estado do Operador no *Workspace* de "*Offline*" para "*Idle*". Desta forma a partir do momento que o Operador faz login torna-se acessível ao Cliente. Caso já lhe tivesse sido atribuída uma tarefa antes do seu último *logout* mas tenha ficado com a tarefa por aceitar, é-lhe devolvida essa mesma tarefa.

5.3 Chamadas de voz

Usando a *API REST* do *Twilio*, para que seja possível os operadores fazerem chamadas a partir do *browser*, teve de ser criado um *capability token*. Os *capability tokens* servem para controlar exatamente as permissões de cada utilizador. Neste caso o servidor fornecerá todos os utilizadores com *tokens* que lhes permitem fazer chamadas *outbound*.
Foi utilizada a biblioteca do *Twilio Node Helper* para gerar e configurar um *capability token* com o *SID* da conta do *Twilio* e respetivo *token* de autenticação (AUTH_TOKEN).

5.3.1 *Dialer*

O *softphone* do *Twilio* é representado por Twilio.Device e o seu estado pode assumir 3 valores: *offline*, *ready* ou *busy*. Foi configurado um botão de CALL/HANGUP com o comando Twilio.Device.connect para iniciar uma chamada. O botão fará uma ligação ao *Phone Controller*

para lidar com a chamada. Para desligar a chamada é usado o Twilio.Device.disconnectAll(), que termina a chamada atual.
Também foi criada uma caixa de *input*, para que pudesse ser inserido um número no formato E.164 (formato utilizado pelo *Twilio*) para chamada *outbound*, e 3 botões com as seguintes funções:

- **Botão PT** - insere o indicativo de Portugal na caixa de input;
- **Botão "Copiar"** - copia automaticamente para o *Clipboard* o contacto que estabeleceu ligação *inbound*, para que possa ser pesquisado nos Contactos do *Salesforce* (processo abordado no cap.5.8);
- **Botão "*Clear*"** - limpa a caixa de *input*.

A aparência do dialer é demonstrada na figura seguinte.

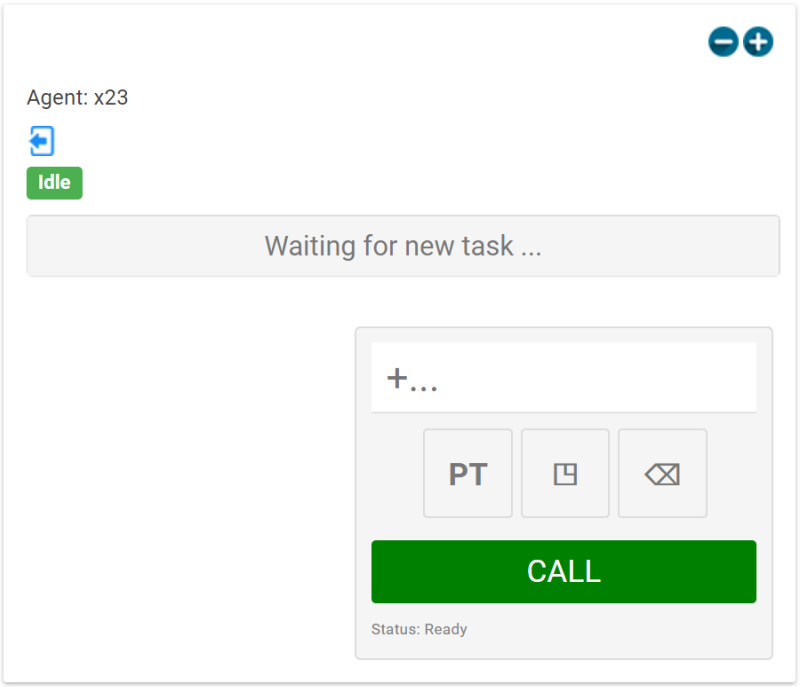

Fig. 30: Interface dialer

Sempre que é solicitada uma chamada de voz, seja *outbound* ou *inbound*, é o *Phone Controller* o responsável em controlar a chamada. Tem de inicializar o telefone do *browser* sempre que uma chamada é estabelecida, e dar *feedback* se a chamada está a decorrer sem problemas ou se foi terminada da forma esperada, bem como apresentar mensagens de erro caso se tenha perdido a ligação. É também o *Phone Controller* que recolhe o número inserido na caixa de *input* e através da *API REST* o passa para o *Twilio*, onde também se pode posteriormente consultar um breve relatório acerca da chamada. Para saber se o Cliente desligou a chamada foi usada uma instrução do *Twilio* chamada *StatusCallback*. Um *request* do *StatusCallback* é uma solicitação que acontece em tempo real quando a chamada termina, mas é também muitas vezes desencadeado por outros motivos, como um sinal ocupado ou sem resposta.

5.3.2 *IVR*

De forma a responder a uma ligação telefónica de alguém que faça uma chamada *inbound* para o nosso número, foi configurado um *webhook* (HTTP GET). Trata-se de um *link* que é acedido pelo

Twilio de forma a receber instruções sobre o passo seguinte. Neste caso o *link* remete para o sistema *IVR*. A partir do momento que o cliente faz a chamada *inbound*, e é saudado por este sistema, já estão a ser enviadas informações ao *TaskRouter*, de maneira a que possa organizar adequadamente os *queues*.
Através da instrução <Gather> é reproduzida uma mensagem de boas-vindas com as instruções para o utilizador escolher o departamento que pretende contactar. Sempre que é executado um <Gather> é esperada a inserção de algum dígito. Para este caso o <Gather> foi configurado com os seguintes atributos:

- ***action*** - inclui um URL relativo para que assim que o Cliente insira um dígito possa ser feito um GET request incluindo os parâmetros descritos abaixo;
- ***method*** - podendo assumir o valor GET ou POST, neste caso assumiu 'GET' para especificar o tipo da solicitação feita no "action";
- ***timeout*** - define o limite em segundos (4) que o *Twilio* esperará para que o Cliente pressione um dígito antes de seguir em frente e fazer uma solicitação ao URL 'action';
- ***numDigits*** - define o número de dígitos (1) que podem ser inseridos pelo Cliente e envia os dados para o URL 'action', assim que estes sejam inseridos.

Uma vez conhecido o *input* do utilizador, através da instrução <analyzeKeypadInput>, é criado um parâmetro <Digits> com o número escolhido pelo Cliente, que define o departamento que quer contactar, e o mesmo é enviado para o *Workflow Controller*. Dependendo do *input* do cliente, existem duas rotas. Caso insira um valor inválido, ou não insira nenhum *input* por um período de 4 segundos, é reproduzida uma mensagem de erro e através do comando <Redirect> é redirecionado de volta para a mensagem de boas-vindas. Se o *input* for válido é reproduzida outra mensagem que questiona o cliente quanto à preferência por um "*callback*" assim que haja um operador disponível, ou se prefere ficar em linha, sendo encaminhado para um agente do departamento escolhido, assim que esteja livre. Se todos os operadores estiverem ocupados, o cliente fica em espera até que seja atendido. É então criada uma tarefa com base nos dígitos pressionados.
Nesta componente surgiram problemas relativamente à língua do áudio do sistema *IVR*. Apesar do *Twilio* ter suporte para a língua portuguesa na sua ferramenta de *text-to-speech*, algumas condições e normas já presentes em parte do código que foi reproduzido a partir da documentação do *Twilio* impediram a alteração da língua inglesa. Para contornar esta situação, e permitir que a aplicação tanto possa ter um sistema *IVR* em inglês como em português, utilizou-se uma abordagem distinta daquela utilizada no *IVR* em inglês. Enquanto que no *IVR* em inglês o texto reproduzido estava escrito no código e era convertido na hora da chamada para áudio através do comando <Say>, no caso do *IVR* em português a solução encontrada foi a reprodução de ficheiros MP3, através do comando <Play>, com o texto desejado já gravado. Apesar de não ser uma solução tão *user-friendly* como a versão do *IVR* em inglês, visto que se posteriormente à implementação da aplicação, caso sejam adicionados novos departamentos, têm de haver novas gravações de ficheiros MP3 e adições no código, esta resolução cumpre o seu propósito e é completamente funcional. Para a gravação do áudio em português foi usada a ferramenta *text-to-speech* do site *iSpeech.org* e o armazenamaneto MP3 dos ficheiros foi feita através do *Salesforce*, em contas próprias da Dom Digital. Abaixo é apresentado um diagrama de atividade que demonstra o seguimento de uma chamada *inbound*.

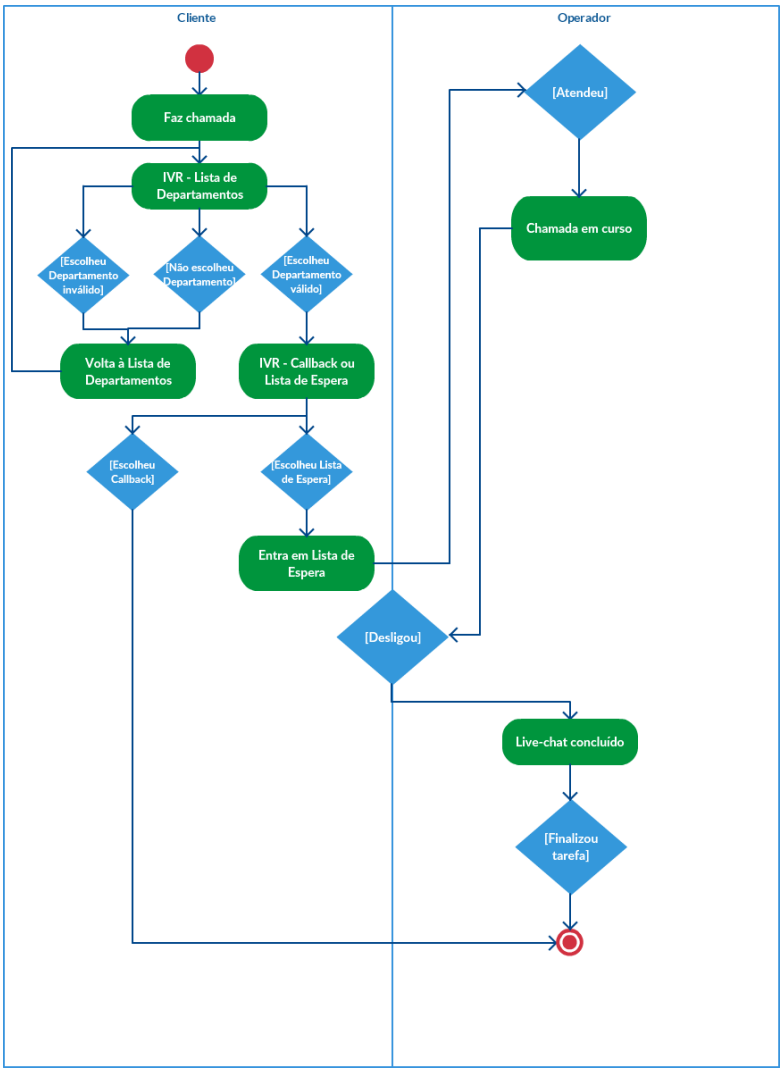

Fig. 31: Diagrama de Atividade – Chamada *inbound*

5.3.3 Pedido de *Callback* por formulário

Ao contrário do pedido de *callback* referido no capítulo anterior, em que o pedido é feito através do sistema *IVR*, neste caso o Cliente faz o pedido através do preenchimento de um formulário *online*, onde são introduzidos o departamento que pretende contactar, a sua questão/problema, o seu nome e o número para o qual deseja ser contactado. Foi usado *AJAX* para enviar o formulário de forma assíncrona, uma vez que *requests* síncronos bloqueiam a execução do código o que pode causar travagens e falta de resposta para o Cliente. No caso dos *requests* assíncronos, existe uma mensagem de retorno assim que os dados são recebidos. Desta forma o *browser* pode continuar a funcionar normalmente enquanto o pedido do Cliente é efetuado. Esta é uma implementação comum do método jQuery.post() do *jQuery*. Uma vez introduzidos todos os campos e corretamente validados, são redirecionados para o *TaskRouter* que se encarrega de criar a devida tarefa. Assim que o Operador aceitar esta tarefa ser-lhe-ão apresentados todos os dados fornecidos pelo cliente. O processo de *callback* pode ser facilmente entendido através do seguinte diagrama de atividade.

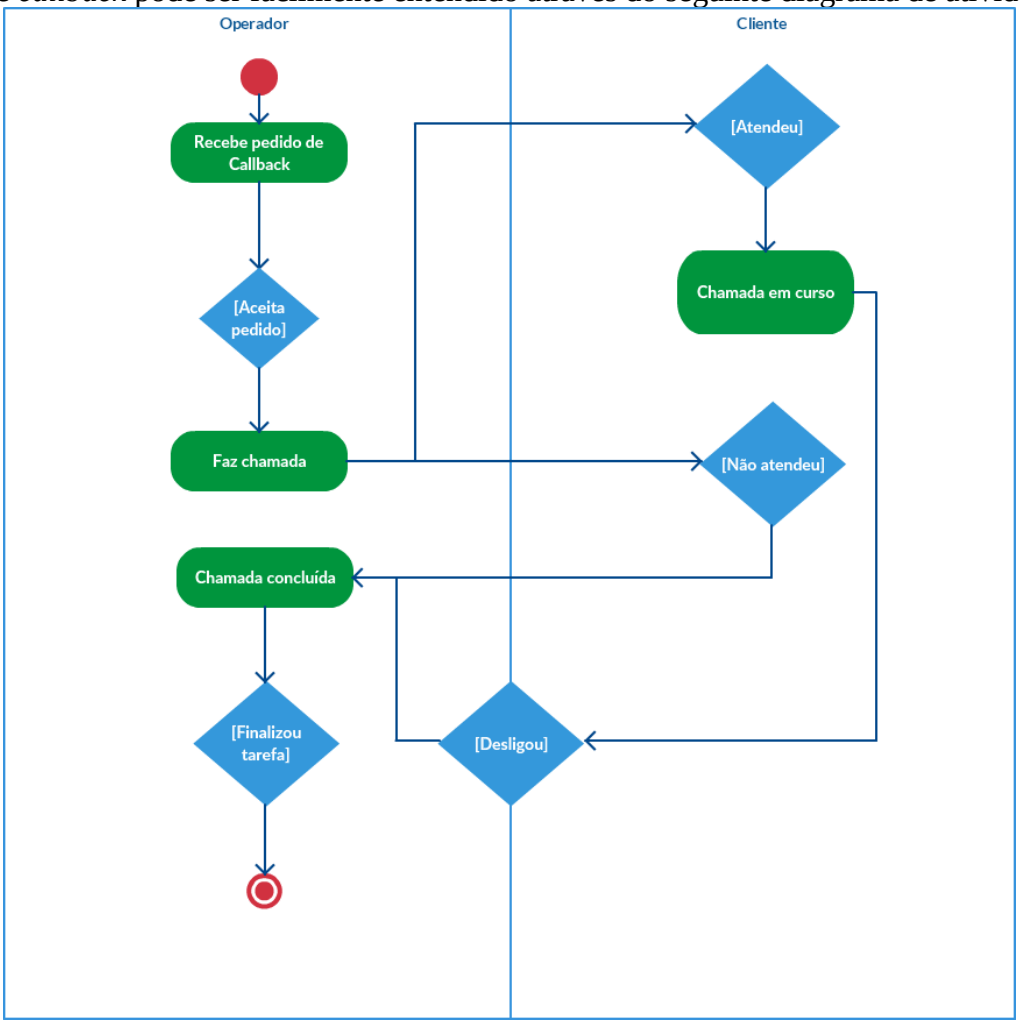

Fig. 32: Diagrama de Atividade – Pedido de Callback

Na imagem seguinte é demonstrada a *interface* do Operador no momento em que lhe é atribuída uma tarefa de *callback* por formulário.

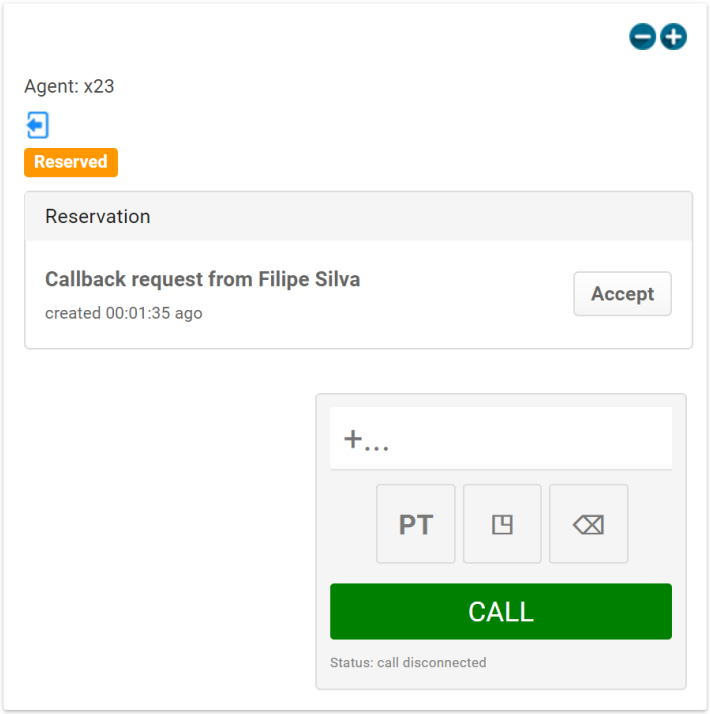

Fig. 33: Interface com pedido de *callback* por aceitar

5.4 Videochamadas

Para a realização das videochamadas foi utilizada a *Twilio Video Rooms API* de forma a poder estabelecer conversações em tempo real de video e voz. Apesar do cliente do *Twilio Voice* necessitar de uma ligação de meio de comunicação aos servidores de *Twilio* (para ligar um operador de telecomunicações), o *Twilio Video* possui um *design* diferente. Os clientes de video ligam-se diretamente uns aos outros por *peer-to-peer* visto que cada um dos pontos funciona tanto como cliente quanto como servidor. Os *SDK* do *Twilio* utilizam a família de padrões *WebRTC*. O *WebRTC* permite a transmissão segura de conteúdo de video e áudio diretamente de um dispositivo para outro através de *browsers*. O *Twilio Video* usa *WebRTC* para capturar e transmitir o video e áudio, e usa a infraestrutura do *Twilio* para lidar com a negociação e *setup* de chamadas automaticamente, como é demonstrado na figura seguinte.

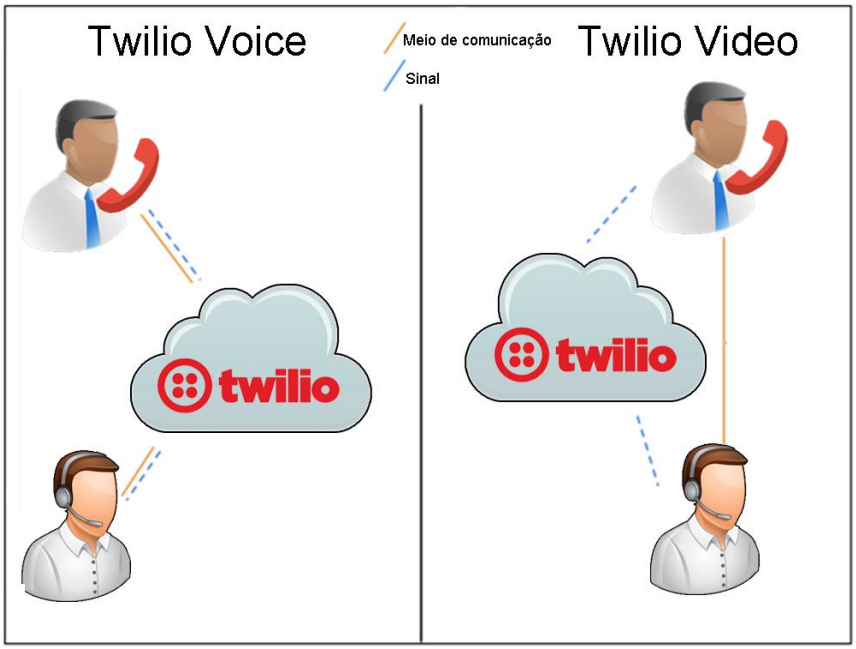

Fig. 34: Twilio Voice vs Twilio Video

Foi criado um controlador chamado *Videocall Controller* onde foi definida a configuração das conversações. Assim que um Cliente solicita o início de uma videochamada é automaticamente criado um canal de conversação onde o Cliente é inserido. Nesse momento é também iniciada a respetiva tarefa, que é enviada para o *Workflow Controller* para que o cliente possa ser atendido. Assim que o Operador aceite a tarefa, o mesmo é redirecionado para o canal criado previamente pelo Cliente. A partir desse momento a ligação está estabelecida. Foram também criados dois botões na *interface* do cliente. Um para que possa ocultar a sua câmara se por alguma razão o achar necessário, e outro para terminar a conversação. Para além disso foram também definidos no *Videocall Controller* várias possíveis mensagens de erro que podem ser causadas por uma ligação fraca que informam se algum dos intervenientes perdeu a ligação, e avisos de inacessibilidade à *webcam* ou ao microfone devido a restrições do *browser*. O progresso de uma videochamada está descrito no seguinte diagrama.

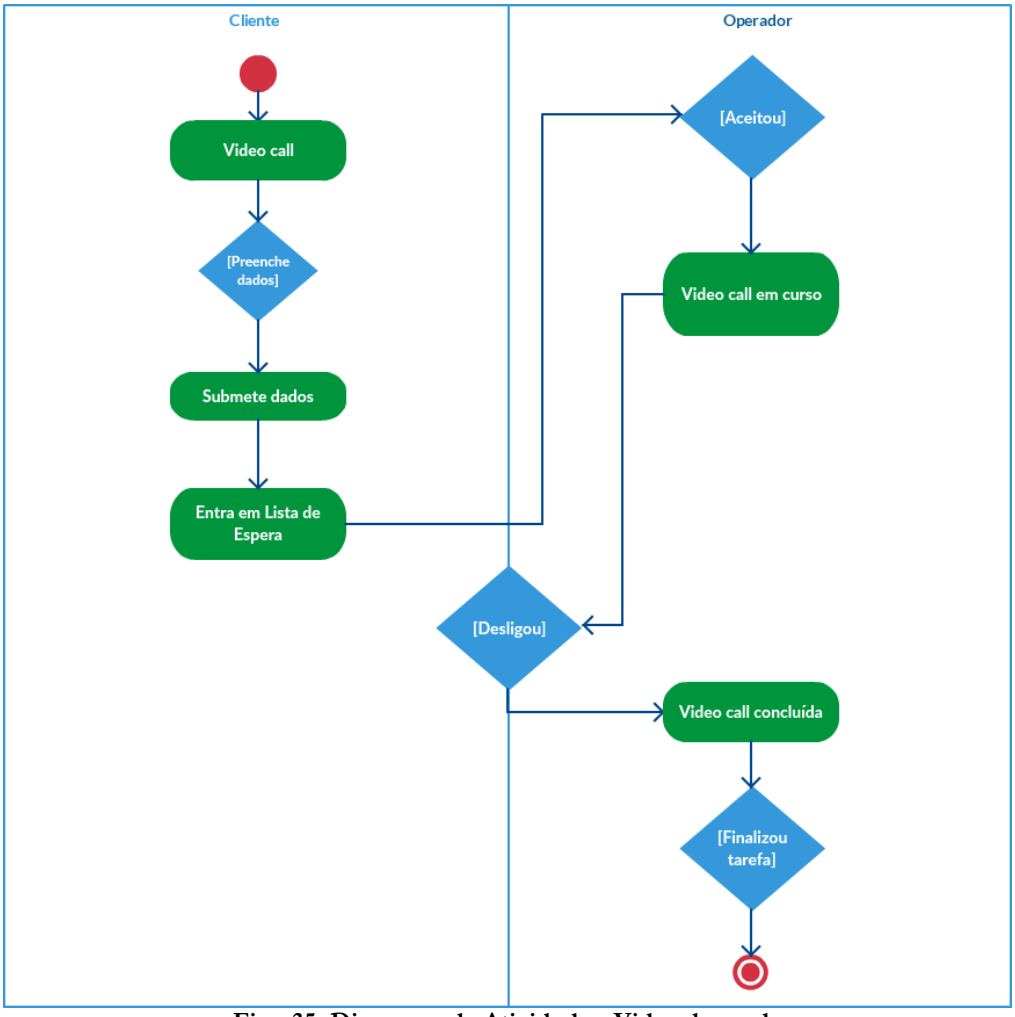

Fig. 35: Diagrama de Atividade - Videochamada

Na figura seguinte está representada a *interface* do Cliente durante uma videochamada, totalmente funcional em sistemas *mobile*.

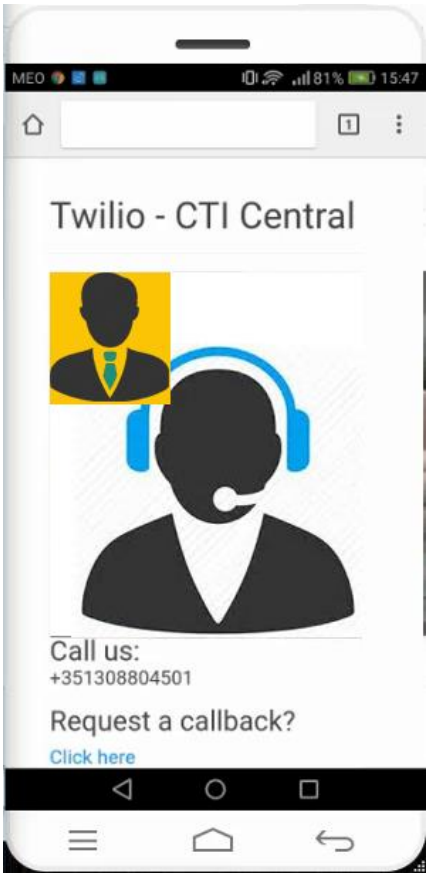

Fig. 36: Interface do Cliente durante videochamada (*mobile*)

5.5 *Live-chat*

A funcionalidade que inicialmente parecia a de mais fácil implementação revelou-se a mais difícil. Isto porque a funcionalidade de *Live-chat*, ao contrário das chamadas de voz e das videochamadas, em que uma vez estabelecido contacto não existe mais nenhum processo até se terminar a ligação, contém sempre comunicações a ser feitas desde o canal de L*ive-chat* para a aplicação, e vice-versa. Para a resolução deste serviço foi utilizada a *Twilio Programmable Chat API* para este serviço. Foi também criado um controlador chamado *Chat Controller* onde foi definida a configuração das conversações. Assim que um Cliente solicita o início de uma conversação de *Live-chat* é automaticamente inserido num canal de conversação, que caso ainda não exista é automaticamente criado e é-lhe também exibida a *interface* de *chat*. Tal como nas videochamadas, é também iniciada a respetiva tarefa, que é enviada para o *Workflow Controller* para que o cliente possa ser atendido. Assim que o Operador, para o qual a tarefa seja remetida a aceite, o mesmo é redirecionado para o canal criado previamente pelo Cliente. A partir desse momento a ligação está estabelecida e temos de fazer *listen* aos eventos que são ativados. Todos os passos e eventos para estabelecer uma conversação de *Live-chat* e as interações Operador/Cliente estão descritos na seguinte tabela.

Tabela 16: Passos e eventos do Live-chat

Ativação	Efeito	Condição	Resultado
Cliente submete dados	Validação de Setup	Setup válido?	Sim - Serviço inicializado
			Não - Erro ao carregar setup
Serviço inicializado `connectMessagingClient`	Channel inicializado	Ligação ao cliente?	Sim - Tarefa criada / Estado = 'Reserved'
			Não - Erro ao iniciar channel
Tarefa criada		Agente aceita / Sucesso?	Sim - Ligação estabelecida / Estado = 'Busy'
			Não - Servidor retorna erro
Ligação estabelecida `InitializeChat`	Exibir canvas	Sucesso?	Sim - 'An agent will be available shortly'
			Não - Servidor retorna erro
Membro entrou `memberJoined`	identidade + " has joined the channel'" '		
Membro saiu `memberLeft`	identidade + " has left the channel'" '		
Membro iniciou escrita `typingStarted`	identidade + " started typing'" '		
Membro terminou escrita `typingEnded`	identidade + " stopped typing'" '		
Membro enviou mensagem `messageAdded`	Mensagem enviada		
	Retorna momento 'HH:mm'		
Ninguém envia mensagem durante 30min `tokenExpired`	Sessão expirada		

Na figura seguinte está representada a *interface* do Cliente durante uma conversação de *Live-chat*, completamente funcional em sistemas *mobile*.

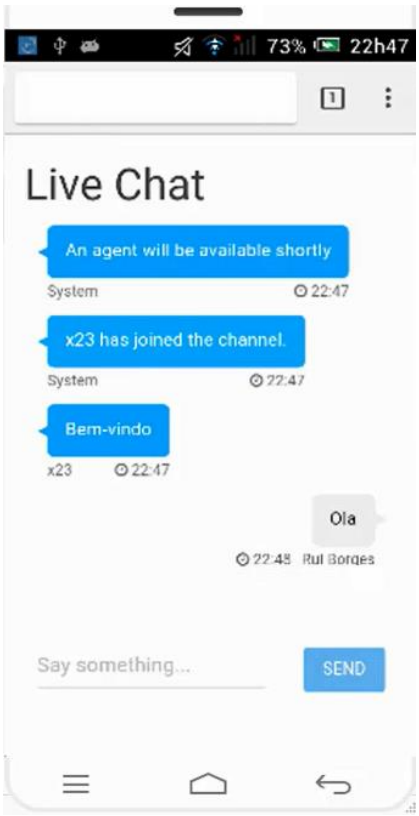

Fig. 37: Interface do Cliente durante Live-chat (*mobile*)

5.6 Administração

Os Administradores têm a possibilidade de gerir as várias opções do sistema *IVR*, bem como remover Operadores existentes e adicionar novos, especificando o departamento a que pertencem e os meios de comunicação a que têm acesso (*Phone*, *Chat*, *Video*). Apesar da adição e remoção de Operadores poder ser realizada totalmente através da *interface* do *TaskRouter* no *Twilio*, a mesma não conta com nenhuma ferramenta para lidar com as opções *IVR*. Como tal, foi criada essa ferramenta na própria aplicação, dentro de uma interface dedicada inteiramente para a administração. A exibição da lista dos Operadores e opções *IVR* existentes, bem como esta interface é controlada pelo *Administration Controller*, onde também foi adicionada a possibilidade de adição e remoção de Operadores, para dispensar o acesso à consola do *Twilio*. Este controlador está diretamente ligado ao *TaskRouter*, onde são guardados todos os dados relativos a Operadores e opções *IVR* (classificadas na consola do *Twilio* como *Custom Task Channels*) bem como ao *Setup Controller*. Para a criação de um Operador são usados 3 parâmetros:

- ***contact_uri*** - nome do Operador;
- ***channels*** - meios de comunicação a que tem acesso;
- ***team*** - departamento a que pertence.

Para criar um Operador diretamente na interface do *TaskRouter* a sintaxe do código é a seguinte:

```
{"channels":["MEIOS_DE_COMUNICAÇÃO"],"contact_uri":"client:NOME","team":"DEPARTAMENTO"}
```

Exemplo de código para a criação de um Operador com acesso a atendimento por telefone, *chat* e video, com o nome "x23" e pertencente ao departamento de vendas:

```
{"channels":["phone","chat","video"],"contact_uri":"client:x23","team":"sales"}
```

A cada Operador também está associado o parâmetro "activitySid" relativo ao estado atual, que por padrão fica definido como "offline".
Para a criação de uma opção *IVR* só são necessários 2 parâmetros:

- ***option*** - nome da opção/departamento;
- ***ivr.options*** - número da opção.

Sempre que é adicionado um novo Operador ou opção *IVR* é feita uma nova ligação ao *Setup Controller* para validar todos os dados necessários, visto que o *TaskRouter* é uma ferramenta da consola do *Twilio* e para usufruir dos seus serviços é necessário que a conta tenha saldo positivo.
Nas figuras abaixo são demonstradas a interface de adição/remoção de Operadores (figura 38) e a interface de adição/remoção de opções *IVR* (figura 39).

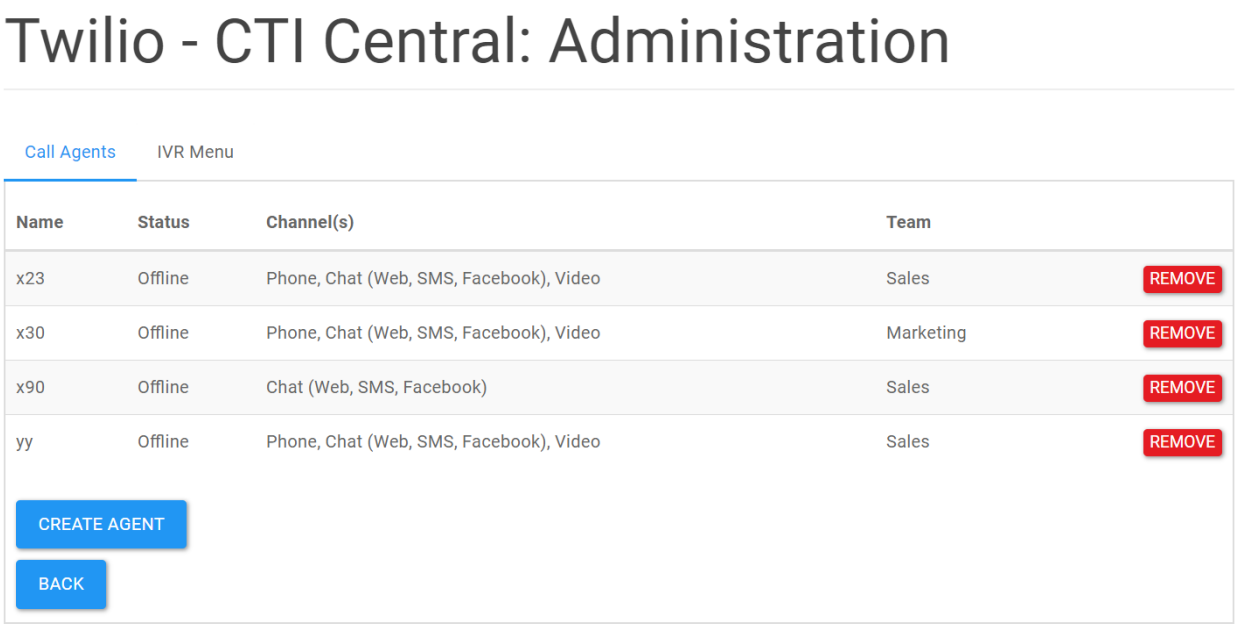

Fig. 38: Interface de adição/remoção de Operadores

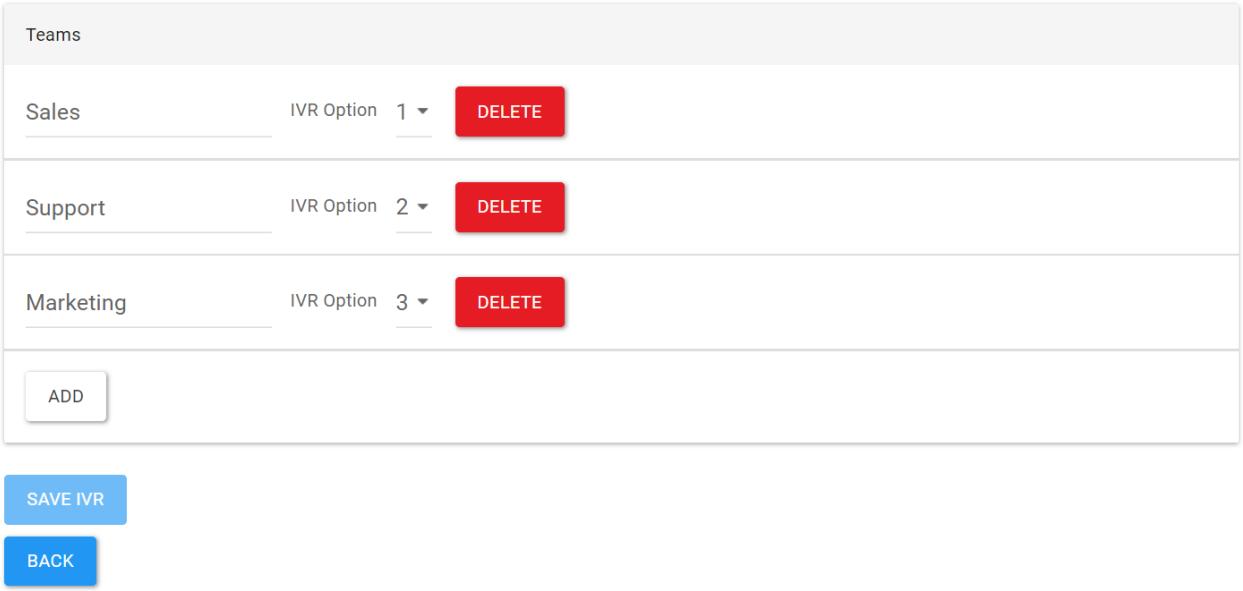

Fig. 39: Interface de adição/remoção de opções IVR

Uma vez que toda a interface estava bastante simples em termos visuais, foi sugerido que se fizesse uma melhoria na *homepage*, visto que é a primeira página que é vista pela administração. Para isso foi descarregado um *template* gratuito de uma página e foi personalizado para corresponder e combinar com o serviço que se oferece.
Foram criados seis ícones identificativos utilizando o *Adobe Photoshop*, demonstrados na imagem abaixo.

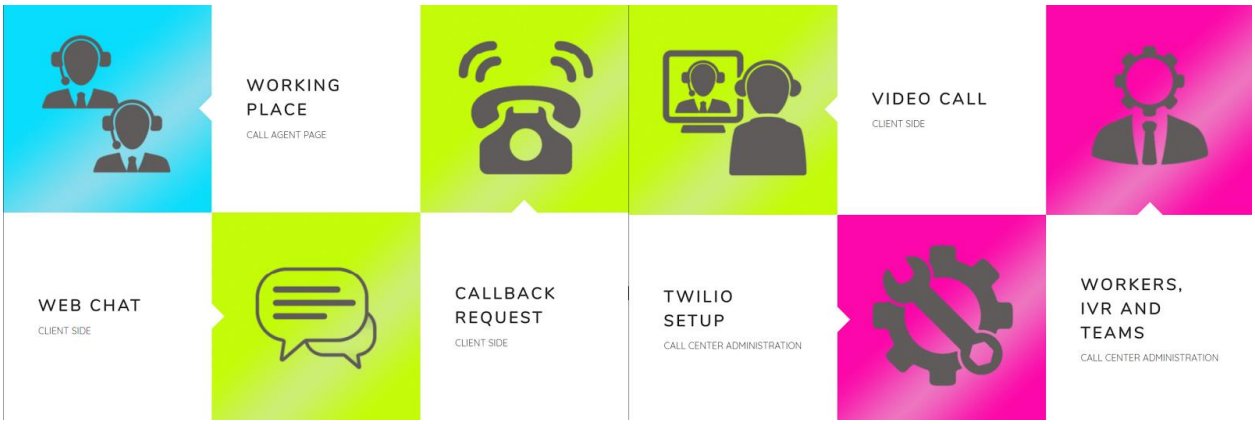

Fig. 40: Homepage da administração

5.7 Configuração *GitHub* e *Heroku*

Foi utilizado o *GitHub* como ferramenta de controlo de versões e armazenamento de repositórios.
Inicialmente foi criado um repositório vazio no *GitHub* e através do *Git Bash* foi inserido o seguinte comando de modo a ter acesso local ao repositório no computador, copiando o repositório para um subdiretório do diretório atual:

```
git clone https://github.com/NOME_UTILIZADOR/NOME_REPOSITORIO.git
```

Sempre que foram feitas alterações no código foram usados os seguintes comandos, para fazer "*commit*" das modificações:

1. `git add .` - específica ficheiros a que desejamos fazer "*commit*". Com o "." no final do comando significa que todas as alterações do repositório vão fazer "*commit*";
2. `git commit -m "MENSAGEM"` - faz "*commit*" dos ficheiros selecionados anteriormente, e é especificada uma mensagem relativa às alterações realizadas, para melhor organização;
3. `git push origin master` - faz "*push*" dos ficheiros aos quais foi feito o "*commit*" para o repositório online do *GitHub*.

Também foram regularmente utilizados os seguintes comandos auxiliares:

- `git status` - exibe o estado do diretório de trabalho, permitindo ver que ficheiros foram alterados e que mudanças foram alvo de "*commit*;
- `git log` - exibe "*commits*" confirmados, exibindo uma lista do histórico do repositório, com possibilidade de filtrar e procurar mudanças específicas. Ao contrário do comando *git status*, o *git log* só funciona no histórico que sofreu "*commit*".

Para fazer *deploy* do projeto foi usado o *Heroku*. Foi criada uma app dentro do *Heroku* à qual foi associado como método de *deploy* o *GitHub*. Para isso foi preciso apenas autenticar o *login* do *GitHub* e especificar que repositório se pretende associar à *app*. Foi escolhida a opção de *deploy* manual ao invés de *deploy* automático, como se demonstra na figura seguinte. Desta forma se houver algum "*commit*" ao qual seja feito um "*push*" erroneamente, a app do *Heroku* não sofre modificações.

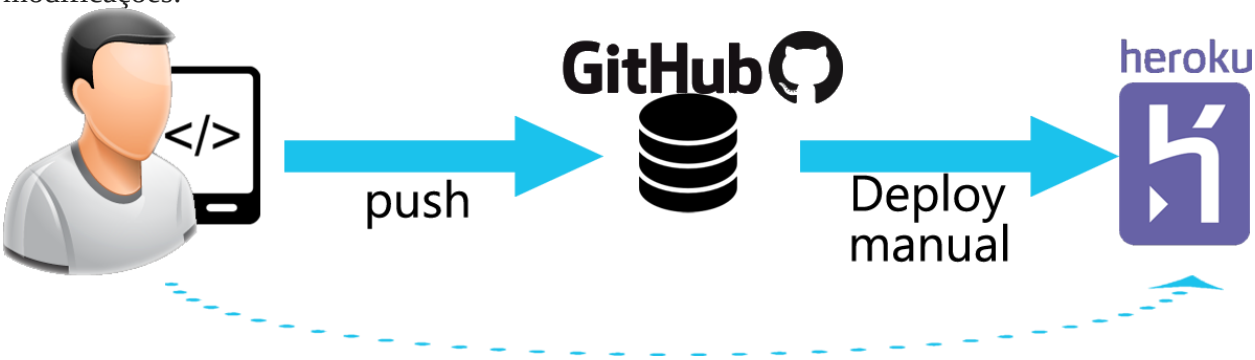

Fig. 41: Diagrama de processo de *deploy*

A aplicação também ficou configurada de forma a que, possam ser criados vários URLs distintos, referentes a várias aplicações do *Twilio*, sem fazer qualquer alteração no código para as diferenciar. Para isso, no primeiro *deploy* apenas têm de ser inseridos os dados seguintes, relativos ao serviço do *Twilio*:

- ***Account SID*** - código único de identificação da conta do *Twilio*;
- ***Auth Token*** - código único de autenticação da conta do *Twilio*;
- ***SID* do Workspace** - código único de identificação do *Workspace*;
- ***SID* do *Chat Service*** - código único de identificação do *Chat Service*;
- ***SID* da *Key* da API do *Twilio*** - código único de identificação, atribuído a desenvolvedores, para terem acesso a certas propriedades da *API* do *Twilio*;
- ***Secret* da *Key* da *API*** - código para validar o código anterior, só é visível no momento da criação da *Key* da *API*.

Todos estes dados ficam guardados automaticamente no *Heroku*, através duma base de dados gerida em *PostgreSQL*.

5.8 Integração no *Salesforce*

Desde a génese deste projeto que, sendo o *Salesforce* um dos principais serviços e ofertas da Dom Digital, foi procurado um meio de complementar a aplicação do *Twilio* com este serviço. Mas sendo esta a última etapa do projeto, o curto espaço de tempo que restou não permitiu a implementação daquela a que se poderia chamar de "solução ideal". Chegou-se, no entanto, a uma solução que, com apenas duas semanas de tempo despendido, se revelou totalmente funcional e de fácil aprendizagem/utilização.

De seguida serão descritos os diferentes passos para a integração da aplicação do *Twilio* no *Salesforce*. Começou por ser criada uma nova *App* dentro do *Salesforce*. Para isso, através do *Setup* foi aberto o *App Manager* e escolhida a opção "*New Lightning App*". Foi dado o nome à *app* "*Twilio - CTI Central*", escolhido o estilo de navegação "*Console*" e foram adicionados os separadores que um Operador tenha mais tendência em consultar: Contactos, Casos, Contas, Tarefas, Relatórios, Oportunidades e Contratos.

5.8.1 Aba de Operador

Através do *Lightning App Builder* foi criado um novo separador, onde estará visível a aplicação do *Twilio* e uma ferramenta de pesquisa dos Contactos do *Salesforce*. O conteúdo desse separador é apenas uma *Visualforce Page*. Dentro dessa *Visualforce Page* foram inseridos diversos elementos. O primeiro a ser adicionado foi uma barra de pesquisa, cujo *input* tanto pode ser um número de telefone como o nome de um Contacto. De seguida foi adicionado um botão de *search,* um botão para abrir a página do Contacto, e uma tabela, onde poderão ser visualizados os resultados da procura. Para isso foi criado um controlador dentro do *Salesforce*, totalmente programado em *Apex*. Sempre que o botão de *search* é pressionado, irá ser executada uma *query* a todos os Contactos, que retornará e exibirá na tabela os que contiverem o *input* da barra de pesquisa num dos seguintes atributos:

- Name;
- Phone;
- MobilePhone;
- HomePhone;
- OtherPhone;
- AssistantPhone.

De seguida foi colocada na *Visualforce Page* uma *iFrame* com o *link* para a página de Operador da aplicação do *Twilio*, como demonstra a figura abaixo.

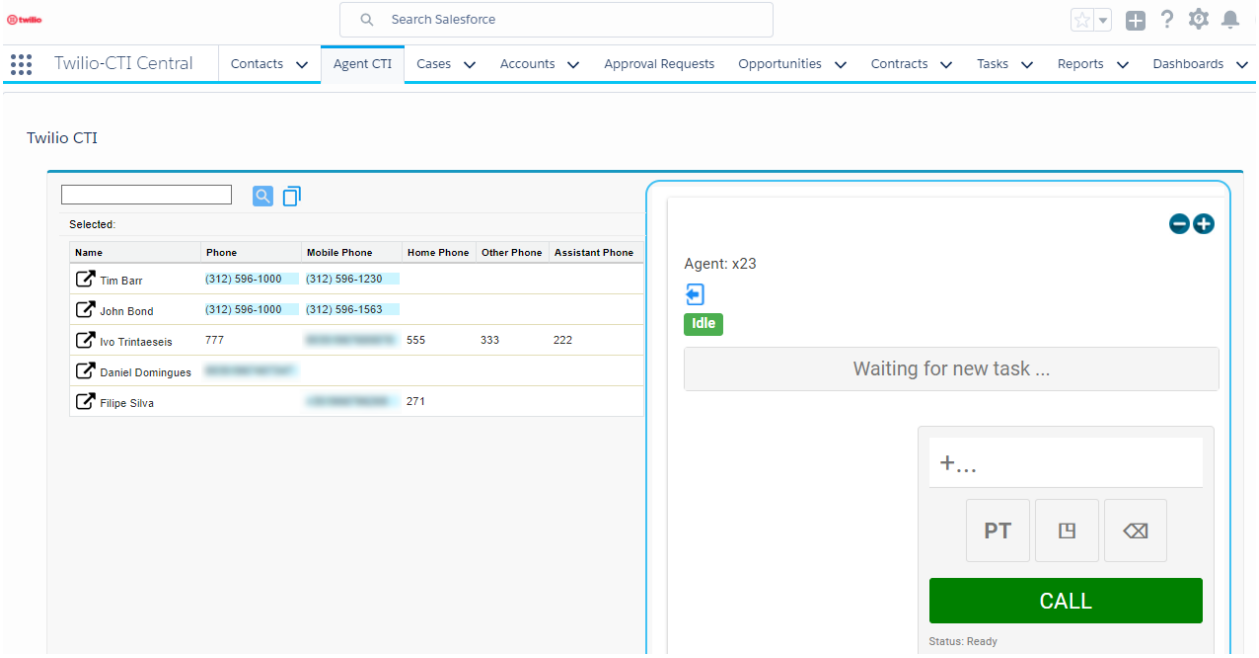

Fig. 42: Interface de Operador no *Salesforce*

Foi também adicionado um botão de Copiar para que, depois do Operador selecionar o número para o qual deseja estabelecer uma chamada, possa copiá-lo automaticamente para o seu *Clipboard*, para colá-lo no *dialer* da aplicação do *Twilio*.

5.8.2 Relatórios de pesquisa

Para que pudesse existir a possibilidade de serem consultados diretamente no *Salesforce* os Clientes com os quais é estabelecido contacto mais regularmente, foi criado um novo atributo para cada Contacto do *Salesforce* chamado "SearchCounter". Este atributo atua como contador das vezes que o Contacto é pesquisado. Sempre que o seu nome, ou número de telefone (fixo, de casa, móvel, de assistente, ou outro) seja pesquisado, irá incrementar este contador. Assim, usando este atributo como variável de base para análise, é possível criar relatórios com dados estatísticos acerca dos Clientes, bem como a geração de gráficos, através da ferramenta do *Salesforce* "*Report Creator*", como é demonstrado na figura seguinte.

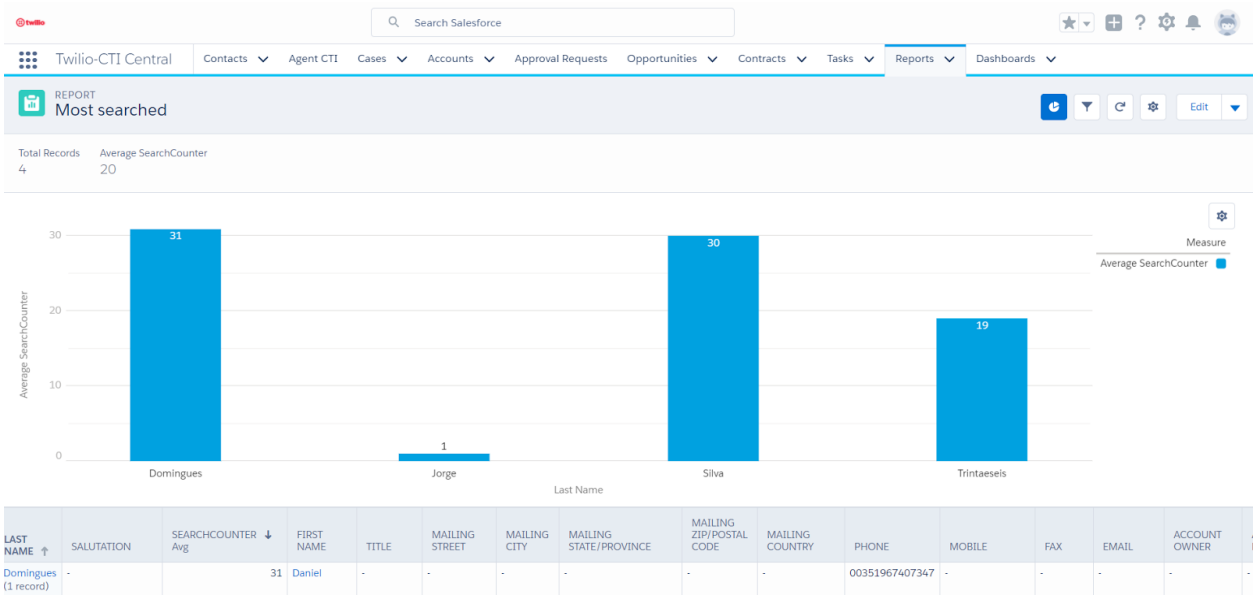

Fig. 43: Relatório gerado pelo *Salesforce*

6. Conclusão

Neste projeto foram exploradas profundamente algumas das funcionalidades do *Twilio*, *software* com o qual nunca tinha interagido anteriormente. Os objetivos a ele adjacentes foram cumpridos. Foi criada uma aplicação através da qual se podem estabelecer contactos telefónicos através do *browser*, foi criado um sistema *IVR* dependente das escolhas do Cliente e plenamente ligado aos departamentos dos Operadores, Operadores esses que possuem diferentes permissões e estados. Foi criada a possibilidade de o Cliente iniciar uma conversação através de *Live-chat* ou videochamada, bem como a possibilidade de receber uma chamada telefónica diretamente no seu número assim que haja um Operador disponível. Tudo isto com suporte a dispositivos *mobile*.

No *Salesforce*, apesar do resultado não ser tão aprimorado e capaz como a aplicação do *Twilio*, conseguiu-se desenvolver uma solução totalmente funcional, mesmo com o curto espaço de tempo que nela pôde ser despendido, dando ao Operador a possibilidade de fazer pesquisas por todos os Contactos de forma simples, bem como consultar relatórios com base nessas pesquisas. Não é, no entanto, a solução ideal. Seria preferível uma ligação direta do *Salesforce* ao *Heroku* de modo a obter diretamente os dados do Cliente, sem recurso ao uso do *Clipboard*, visto que seria menos um passo desnecessário que o Operador teria de tomar.

Outro aperfeiçoamento que poderia ter sido implementado seria a possibilidade de fazer uma partilha do ambiente de trabalho por parte do Operador ou do Cliente, uma vez que se trata de uma ferramenta que algumas aplicações de serviço de apoio ao cliente também já incluem.

A componente visual da aplicação também poderia ser mais elaborada do ponto de vista estético, apesar da sua simplicidade lhe conferir uma utilização bastante fácil, o que era uma prioridade.

A metodologia de trabalho foi favorável ao desenvolvimento do projeto, e graças às entregas dos documentos referentes ao progresso semanal conseguiu-se manter uma boa organização relativamente às tarefas propostas pelo supervisor.

Na última semana de estágio foi feita uma apresentação final para alguns membros da equipa da Dom Digital, demonstrando o funcionamento e todas as funcionalidades do projeto, com o objetivo de fazer uma avaliação deste percurso e conhecer opiniões gerais acerca do resultado.

Por parte da Dom Digital a avaliação foi bastante positiva, acreditando que este projeto se trata de uma mais valia na sua oferta.

Para finalizar, considero que este projeto em contexto de estágio foi um ponto marcante na minha aprendizagem, não só devido à consolidação de conhecimentos já adquiridos, mas também devido à descoberta de novas tecnologias e soluções, que de outra forma não teria tido oportunidade conhecer e explorar.

Bibliografia

[1]GetApp, visto a 15 de junho de 2017, disponível em:
https://www.getapp.com/it-communications-software/a/ringcentral/

[2]GetVoIP, visto a 16 de junho de 2017, disponível em:
https://getvoip.com/blog/2017/01/05/twilio-vs-nexmo/

[3]GetVoIP, visto a 16 de junho de 2017, disponível em:
https://getvoip.com/blog/2016/06/01/twilio-alternatives/

[4]SelectHub, visto a 16 de junho de 2017, disponível em:
https://selecthub.com/customer-relationship-management/crm-requirements-checklist-and-downloadable-template/

[5]Forbes, CRM Market Share in 2015, visto a 1 de agosto de 2017, disponível em:
https://www.forbes.com/sites/louiscolumbus/2016/05/28/2015-gartner-crm-market-share-analysis-shows-salesforce-in-the-lead-growing-faster-than-market/#31edb15a1051

[6]Appexchange, NewVoiceMedia/CloudCall, visto a 25 de junho de 2017, disponível em:
https://appexchange.salesforce.com/listingDetail?listingId=a0N3000000B41HaEAJ
https://appexchange.salesforce.com/listingDetail?listingId=a0N300000025yAGEAY

[7]BRQ, Metodologias ágeis, visto a 1 de agosto de 2017, disponível em:
http://www.brq.com/metodologias-ageis/

[8]Wikipedia, Diagrama de Gantt, visto a 10 de setembro de 2017, disponível em:
https://pt.wikipedia.org/wiki/Diagrama_de_Gantt

[9]Pedro Pinto, Conceito Cloud, visto a 1 de setembro de 2017.

[10]Tech Target, Cloud ERPs, visto a 1 de agosto de 2017, disponível em:
http://searchcloudapplications.techtarget.com/definition/cloud-ERP

[11]Salesforce, visto a 1 de agosto de 2017, disponível em:
https://www.salesforce.com/

[12]Tech Target, Force.com, visto a 1 de agosto de 2017, disponível em:
http://searchsalesforce.techtarget.com/definition/Forcecom

[13]Salesforce, visto a 1 de agosto de 2017, disponível em:
https://success.salesforce.com/answers?id=90630000000gtomAAA

[14]Salesforce Developers, visto a 1 de agosto de 2017, disponível em:
https://developer.salesforce.com/docs/atlas.en-us.pages.meta/pages/pages_intro_what_is_it.htm

[15]Salesforce Developer Guide, What is Apex?, visto a 10 de setembro de 2017, disponível em: https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_intro_what_is_apex.htm

[16]Salesforce, Service Cloud, visto a 10 de setembro de 2017, disponível em: https://www.salesforce.com/products/service-cloud/overview/

[17]Salesforce, Appexchange, visto a 1 de agosto de 2017, disponível em: https://www.salesforce.com/solutions/appexchange/faq/

[18]Salesforce Developers, Open CTI, visto a 1 de agosto de 2017, disponível em: https://developer.salesforce.com/docs/atlas.en-us.api_cti.meta/api_cti/sforce_api_cti_intro.htm

[19]Twilio, What is Twilio, visto a 1 de agosto de 2017, disponível em: https://www.twilio.com/learn/twilio-101/what-is-twilio

[20]Tutorials Point, Javascript Overview, visto a 11 de setembro de 2017, disponível em: https://www.tutorialspoint.com/javascript/javascript_overview.htm

[21] Tutorials Point, What is AJAX, visto a 11 de setembro de 2017, disponível em: https://www.tutorialspoint.com/ajax/what_is_ajax.htm

[22]Heroku, About Heroku, visto a 11 de setembro de 2017, disponível em: https://www.heroku.com/about

[23]Escritório de Projetos, Diagrama de Contexto, visto a 1 de setembro de 2017, disponível em: https://escritoriodeprojetos.com.br/diagrama-de-contexto

[24] DevMedia, O que é UML?, visto a 1 de setembro de 2017, disponível em: http://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408

[25]IBM, UML Support, visto a 5 de setembro de 2017, disponível em: https://www.ibm.com/support/knowledgecenter/pt-br/SS4JE2_7.5.5/com.ibm.xtools.modeler.doc/topics/ccompd.html

[26] IBM, UML Support, visto a 5 de setembro de 2017, disponível em: https://www.ibm.com/support/knowledgecenter/pt-br/SS4JE2_7.5.5/com.ibm.xtools.modeler.doc/topics/cdepd.html

[27]Twilio Skills, Intro to Twilio Voice, visto a 16 de julho de 2017, disponível em: https://twilio.radicalskills.com/projects/intro-twilio-voice/1.html

[28]Twilio Tutorials, Assign a worker, visto a 18 de julho de 2017, disponível em: https://www.twilio.com/docs/tutorials/dynamic-call-center-node-express#assign-a-worker

[29]Twilio, Task Router, visto a 18 de julho de 2017, disponível em: https://www.twilio.com/taskrouter

Anexo A – Licenças

Twilio Terms of Service, atualizado a 24 de maio de 2016, disponível em:
https://www.twilio.com/legal/tos

Salesforce.com Terms of Use , licença de developer Partner – Dom Digital
http://www.salesforce.com/assets/pdf/misc/salesforce.com_terms_of_use.pdf

Gumroad Terms, licença gratuita para uso de templates CSS
https://gumroad.com/terms

Licença MIT, licença gratuita para uso de Código
https://opensource.org/licenses/MIT

Anexo B – Análise económica

De modo a verificar se este projeto se pode tornar numa oferta viável para a Dom Digital, é necessário fazer uma análise económica, tendo em conta os custos associados à manutenção da aplicação na *cloud* e os custos dos serviços telefónicos. Desta forma, a Dom Digital pode chegar a conclusões acerca da viabilidade deste projeto, bem como os possíveis preços que devem ser cobrados aos potenciais clientes.

Durante a elaboração do projeto foi usado como ferramenta de armazenamento de código na *cloud*, o *GitHub*. No *GitHub*, o pacote empresarial mais simples, e que seria suficiente para este propósito, tem um valor de **$21/mês**. Porém, este não irá ser tomado em conta, uma vez que apenas é necessário utilizar um repositório para fazer o primeiro *deploy* para o *Heroku* (supondo que a aplicação não necessitará de sofrer alterações ao longo do tempo). Para além disso a Dom Digital, já trabalha com outra ferramenta de armazenamento de código, o *BitBucket*.

De forma a correr a aplicação na *cloud*, o plano do *Heroku* que mais se adequa a esta aplicação é o "Standard 2x". Ao contrário do plano gratuito, em que as aplicações ficam suspensas ao fim de 30 minutos de inactividade, este permanece sempre ativo e conta com um servidor com 1GB de RAM, o que é suficiente para manter um *Call Center* desta escala. Este plano tem um custo de **$50/mês**.

Uma vez que os números oferecidos pelo *Twilio* em Portugal são invulgares (308xxxxxx), e podem levar a que o cliente não se sinta seguro em fazer a chamada, a Dom Digital irá optar por fazer reencaminhamentos de chamadas. Assim, ao ligar para um número comum (271xxxxxx), o cliente é automaticamente redirecionado para o número do *Twilio*. O serviço de reencaminhamento de chamadas é gratuito, porém o preço das chamadas irá recair inteiramente sobre o contacto de destino. Portanto o preço de todas as chamadas (*inbound* e *outbound*) têm de ser levados em conta como se fossem outbound.

Para calcular os custos associados aos serviços telefónicos temos de levar em conta dois fatores: o tempo médio de espera, e o tempo médio de conversação. Ambos são significativos, uma vez que, apesar de não estar em conversação direta com o operador, assim que o cliente entra no sistema *IVR*, já está a concretizar uma chamada, e no caso do *Twilio* o preço é equivalente a estar em conversação com um operador. O tempo médio de espera inclui a quantidade de tempo que os clientes aguardam na fila de espera e enquanto o operador não atende, mas não o inclui tempo que leva para que os clientes possam navegar pelo *IVR*. A métrica global para o tempo médio de espera num *Call Center* é de 28 segundos [1]. O *IVR* da aplicação desenvolvida demora em média 25 segundos.
Pode portanto ser arredondado para 1 minuto o tempo que leva a partir do instante em que o cliente entra no sistema *IVR*, até que é atendido por um operador. Irão ser aplicados os custos para números móveis, uma vez que são bastante mais utilizados atualmente do que os números fixos. Um minuto de chamada mobile (*outbound*) é **$0.1150**.

A métrica global para a duração de uma chamada num *Call Center* é de quatro minutos por chamada [1]. Este intervalo refere-se ao instante que vai desde o momento em que o operador atende a chamada do cliente, até que esta é desligada. 4 x $0.1150 = **$0.4600**

Chegamos então a um preço médio por chamada de $0.1150 + $0.4600 = **$0.575**

Sendo esta uma oferta que tem como propósito a integração no *Salesforce*, em que o operador tem outras tarefas para além do atendimento de chamadas, não está projetada para receber um tráfego de chamadas de grande escala. Posto isto, foi estimada (por alto) uma quantidade de 30 chamadas diárias. 30 x $0.575 = **$17.25/dia**. = **$517.5/mês**.

Somando **$1**, valor cobrado mensalmente pelo *Twilio* por cada número comprado, **$50** para alojar a aplicação no Heroku, e **$517.5** para os custos das chamadas telefónicas, temos um custo final médio de **$568.5/mês**.

Concluindo este valor torna-se mais simples a idealização de um valor a cobrar pela Dom Digital a um cliente deste serviço. No entanto é apenas um valor médio, que irá oscilar bastante em função da escala e sector do cliente.

De referir que os serviços de videochamada e de *Live-chat* do *Twilio*, não têm nenhum custo a eles inerente, bastando apenas ter um saldo de conta positivo, e ter um número telefónico ativo.

[1]Talkdesk, Call Center Performance Benchmarking, visto a 15 de setembro de 2017, disponível em:
https://www.talkdesk.com/blog/call-center-performance-benchmarking

Anexo C - Glossário

ACD: Distribuidor Automático de Chamadas, sistema de telefonia que responde e distribui chamadas recebidas para um grupo específico de terminais ou agentes dentro de uma organização.
API: Interface de Programação de Aplicações, conjunto de códigos e padrões de programação para acesso a uma aplicação de software ou plataforma web.
APP: abreviatura de aplicação**.**
BACKUP: cópia de segurança. O objetivo da ação é a prevenção de uma ocasional perda de ficheiros/informação, seja por ações despropositadas do utilizador ou por mau funcionamento dos sistemas.
BILLING: sistema de cobrança.
BOT: programa concebido para simular ações humanas repetidas vezes de maneira padrão.
BROWSER: navegador de Internet, software que permite a visualização dos conteúdos apresentados numa página web.
BUSY: estado ocupado**.**
CALL: chamada.
CALL CENTER: central de atendimento que tem como objetivo fazer a interface entre o cliente e a empresa.
CALLBACK: chamada de volta do Operador para o Cliente.
CEO: Chief Executive Officer, cargo mais elevado na hierarquia da gestão de uma empresa.
CHAT: conversação.
CLICK-TO-DIAL: forma de comunicação na web em que uma chamada é iniciada através de um click num objeto.
CLIPBOARD: área de memória em que o utilizador pode armazenar temporariamente informação que pretende transferir entre programas ou pontos diferentes do mesmo programa, recorrendo aos mecanismos de copiar e colar; área de transferência.
CLOUD COMPUTING: computação em nuvem; refere-se à utilização da memória e da capacidade de armazenamento e cálculo de computadores e servidores compartilhados e interligados por meio da Internet. O armazenamento de dados é feito em serviços que poderão ser acessados de qualquer lugar do mundo, a qualquer hora, não havendo necessidade de instalação de programas ou de armazenar dados.
CRM: Gestão do Relacionamento com o Cliente, abordagem que coloca o cliente como principal foco dos processos de negócio, com o intuito de perceber e antecipar as suas necessidades, de forma a atendê-los da melhor forma.
CSS: Cascading Style Sheets, mecanismo para adicionar estilo (cores, fontes, espaçamento, etc.) a um documento web.
CTI: Integração entre Computador e Telefonia, processo pelo qual um equipamento telefónico troca informações de uma chamada com um computador, permitindo ao computador ou utilizador uma melhor gestão da chamada.
CTO: Chief Technology Officer, responsável pela arquitetura e infra-estrutura dos sistemas, pesquisas e desenvolvimento de novos produtos.
DASHBOARD: painel de instrumentos, apresenta resumos de dados importantes, permitindo ao utilizador analisá-los pormenorizadamente.

DEBUGGING: depuração, processo de encontrar e reduzir defeitos numa aplicação de software ou mesmo em hardware.
DEPENDENCIES: conjunto de recursos necessário para o funcionamento de software dependente desses recursos.
DEPLOY: processo de implementação.
DESKTOP: computador, PC.
DEVELOPER: programador, desenvolvedor, alguém que escreve, desenvolve ou faz manutenção de software.
DIALER: marcador telefónico.
DOWNGRADE: redução de qualidade.
DOWNTIME: tempo em que um sistema não está operacional.
ENDPOINT: URL por onde um serviço pode ser acessado.
ENGINE: pacote de funcionalidades que são disponibilizadas para facilitar o desenvolvimento de software e impedir que sua criação tenha que ser feita de raíz.
ERP: Planeamento dos Recursos da Empresa, sistema informático responsável por cuidar de todas as operações diárias de uma empresa.
FIRST IN, FIRST OUT: estrutura de dados do tipo fila, em que o primeiro elemento a entrar, é o primeiro a sair.
FRAMEWORK: conjunto de bibliotecas que une códigos de vários projetos para conseguir executar uma operação maior, oferecendo uma funcionalidade genérica.
HARDWARE: componentes eletrónicos físicos de um sistema.
HOMEPAGE: página inicial.
HTML: Linguagem de Marcação de Hipertexto, linguagem de marcação utilizada na construção de páginas na Web.
IDLE: estado vago.
IFRAME: código HTML que faz com que uma determinada página seja aberta dentro de outra.
IN-APP: dentro da aplicação.
INBOUND: refere-se a contactos vindos do exterior.
INPUT: entrada de informação.
INTERFACE: meio pelo qual o utilizador se comunica com o sistema para realizar tarefas.
IP: Protocolo de Internet.
ISV: Fornecedor de Software Independente, organização especializada em fazer e vender software.
IVR: Unidade de Resposta Audível, serviço para um call center que fornece respostas automáticas para os clientes, sem a intervenção de um operador.
LEAD: potencial consumidor que demonstrou interesse em consumir um certo produto ou serviço.
LINK: endereço de um documento ou página web.
LIVE-CHAT: conversação em tempo real.
LOCALHOST: localização do sistema que está a ser usado.
LOGIN: processo para acessar um sistema informático restrito.
LOGOUT: processo de encerrar uma sessão num sistema informático.
LOGS: registo de eventos relevantes.
MARKETING: conjunto de técnicas e métodos destinados ao desenvolvimento de vendas.
MMS: Serviço de Mensagem Multimédia, tecnologia que permite o envio e receção de mensagens multimédia em telemóveis.
MOBILE: sistemas mobile, smartphones ou tablets.
OFFLINE: sem ligação.

ON-PREMISE: software que é instalado em hardware presente no local.
ONLINE: com ligação à Internet.
OUTBOUND: refere-se a contactos feitos para o exterior.
PAY-AS-YOU-GO: serviço pago à medida da sua utilização.
PEER-TO-PEER: formato de rede de computadores em que a principal característica é a descentralização das funções convencionais de rede, onde o computador de cada utilizador acaba por realizar funções de servidor e de cliente ao mesmo tempo.
PHP: Hypertext Preprocessor, linguagem interpretada livre, usada originalmente apenas para o desenvolvimento de aplicações.
POP-UP: janela que se abre automaticamente no browser, utilizada para abrir alguma informação extra ou como meio de publicidade.
PSTN: Rede Pública de Telefonia Comutada, rede telefónica mundial comutada por circuitos destinada ao serviço telefónico.
PUSH NOTIFICATIONS: serviço que permite enviar mensagens ou breves notificações de um servidor para múltiplos terminais.
QUERY: consulta de dados de uma base de dados.
QUEUE: fila.
REQUEST: pedido de informação.
RESERVED: estado reservado.
ROLE PERMISSIONS: permissões dependentes de função.
RUNTIME ENVIRONMENT: ambiente para executar um programa, não incluindo as ferramentas para o alterar.
SALESFORCE: plataforma de CRM nº 1 do mundo.
SCRIPT: texto que contém as informações necessárias para o funcionamento de software.
SDK: Kit de Desenvolvimento de Software, conjunto de ferramentas de desenvolvimento e códigos pré-gravados que podem ser usados pelos desenvolvedores para criar aplicações.
SEARCH: pesquisa.
SELF-SERVICE: serviço próprio.
SETUP: configuração.
SIP: Protocolo de Iniciação de Sessão, protocolo de código aberto de aplicação para iniciar sessões de comunicação entre utilizadores.
SISTEMA OPERATIVO: conjunto de programas informáticos que permite fazer a eficaz gestão dos recursos de um computador.
SMS: Serviço de Mensagens Curtas, tecnologia que permite o envio e receção de mensagens até 160 caracteres em telemóveis.
SOCIAL MEDIA: redes sociais.
SOFTPHONE: aplicação que permite efetuar ligações telefónicas a partir de um computador ou sistema mobile.
SOFTWARE: conjunto dos programas e dos meios não materiais que possibilitam o funcionamento do computador, na execução das diversas tarefas.
TAG/ETIQUETA: palavras que servem como uma etiqueta e ajudam na hora de organizar informações, agrupando aquelas que receberam a mesma marcação, facilitando encontrar outras relacionadas.
TASKROUTER: sistema de encaminhamento de chamadas desenvolvido pela Twilio.
TEMPLATE: modelo a ser seguido, com uma estrutura predefinida que facilita o desenvolvimento e criação do conteúdo a partir de algo construído a priori.

TEXT-TO-SPEECH: técnica de sintetização da fala humana, que converte texto em áudio com linguagem normal.
TI: tecnologias da informação.
TIME-BASED: baseado num prazo.
TOKEN: senhas temporárias.
TOLL FREE: sistema em que a cobrança é feita a partir das chamadas que chegam em vez de incorrer em cobranças para o assinante do telefone de origem.
TRIGGER: recurso de programação executado sempre que o evento associado ocorrer.
TUNELAÇÃO: transmissão de dados destinados a serem utilizados somente numa rede privada, geralmente corporativa, através de uma rede pública.
TWILIO: plataforma de comunicações na cloud que permite aos desenvolvedores a realização de chamadas telefónicas através das suas APIs.
UML: Linguagem de Modelagem Unificada, linguagem-padrão para a elaboração da estrutura de projetos de software.
UPGRADE: aumento de qualidade.
UPTIME: tempo em que um sistema está operacional.
URL: Localizador Uniforme de Recursos, endereço de rede no qual se encontra algum recurso informático.
USER-FRIENDLY: fácil de usar.
VOIP: Voz sobre IP.
WEB: sistema de interligação de documentos e recursos através da internet.
WEBRTC: Comunicação pela Web em Tempo Real, coleção de protocolos de comunicação e interfaces de programação de aplicações que permitem a comunicação em tempo real através de ligações peer-to-peer.
WORKERS: nome dado aos Operadores no TaskRouter.
WORKFLOW: fluxo de trabalho.
WORKSPACE: ligação de cada aplicação ao TaskRouter.