Integração de Sistemas de Voz com TV Digital
Uma Demostração de Conceito

Ana Margarida Ramos Dias

20150189

Orientadores

Prof. Doutor Eurico Ribeiro Lopes

Dissertação apresentada à Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Desenvolvimento de Software e Sistemas Interativos, realizada sob a orientação científica do Doutor Eurico Ribeiro Lopes, do Instituto Politécnico de Castelo Branco.

Junho 2021

Composição do júri

Presidente do júri

Doutor, Alexandre José Pereira Duro da Fonte

Vogais

Doutor, Mário Marques Freire

Prof. Catedrático da Universidade da Beira Interior

Doutor, João Manuel Leitão Pires Caldeira

Prof. Adjunto da Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco

Doutor, Eurico Ribeiro Lopes

Prof. Coordenador da Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco

Agradecimentos

Após mais de um ano de trabalho, a minha tese de mestrado chega ao fim. Olhando para trás, lembro-me das pessoas sem as quais este feito não seria possível, e a quem é devida uma menção.

Em primeiro lugar, gostaria de agradecer à Celfocus por ter aceitado a minha proposta, de fazer do meu projeto profissional o tema da minha tese de mestrado. Em especial à minha equipa, que sempre se mostrou disponível para me ajudar em qualquer situação.

Em segundo lugar, gostaria de expressar a minha gratidão ao meu orientador, o Prof. Eurico Lopes, pelo seu constante apoio e motivação para levar esta tese até ao fim;

- à minha família, que sempre me incentivou a seguir os meus sonhos e a não deixar nada a meio. Foi constante o apoio de todos. Não posso retribuir-lhes por tudo o que me deram, por isso reservar-lhes-ei sempre algum do meu tempo como sinal da minha gratidão.

- ao meu namorado, o Daniel, que comigo partilhou hesitações, incertezas, alguns desaires, sem nunca desistir, apoiando-me sempre. Foi admirável a sua permanente disponibilidade. Com ele, tornei-me mais forte. Agradeço-te pelos bons conselhos, por cuidares de mim e apoiares os meus sonhos, *no matter what*.

E não deixo em esquecimento aqueles amigos que, navegando incertezas, sentiam o assalto de muitas hesitações. Estou-lhes grata, já que as suas palavras de estímulo, em dias de algum desalento, me mantiveram no rumo que traçara com tanto entusiasmo.

Receio, finalmente, que possa ter esquecido alguém que, nestes dois anos, me ajudou, direta ou indiretamente, na realização deste trabalho. A todos me sinto muito obrigada - e perante todos me comprometo a mantê-lo em aberto, e nunca mais o terminar.

Resumo

O aparecimento de assistentes de voz na casa dos clientes dos serviços de televisão tem vindo a ser uma tendência crescente nos últimos anos. Um dos recursos previstos como uma extensão significativa do serviço de televisão é a capacidade de o utilizador poder interagir com o serviço usando a voz, em vez de um comando tradicional. Com o principal objetivo de fazer com que a interação dos espectadores com o serviço de televisão seja simplificada, a Vodafone junta-se à Amazon, para que a partir do seu assistente de voz, a Alexa, seja possível controlar todas as operações nas boxes de televisão nas casas dos seus clientes.

Este projeto, desenvolvido pela Celfocus, tem uma arquitetura de microsserviços que visa fomentar o desenvolvimento ágil e a escalabilidade da solução. Foram também utilizados vários serviços disponíveis na Amazon Web Services, como as funções Lambda, o RDS (Relational Database Service) e o CloudWatch. Para a ligação entre a Alexa e os microsserviços foi necessária a criação de uma aplicação que permite a utilização de recursos ativados por voz nos dispositivos inteligentes conectados (Amazon Alexa Skill).

Com o desenvolvimento deste projeto criou-se um sistema de integração que conecta a assistente de voz, Alexa, e os novos produtos de televisão desenvolvidos pela Vodafone. Desta forma é dada a oportunidade aos telespectadores de usufruírem de uma nova experiência ao verem televisão.

Palavras chave

Assistente de Voz, Alexa, Digital TV, Set-Top-Box, Arquitetura de Microsserviços.

Abstract

The appearance of voice assistants in the homes of television service customers has been a growing trend in recent years. One of the features envisioned as a significant extension of the television service is the ability for the user to interact with the service using voice, rather than a traditional command. With the main goal of making the interaction of viewers with the television service simplified, Vodafone is joining Amazon, so that from its voice assistant, Alexa, it will be possible to control all operations in the television boxes in the homes of its customers.

This project, developed by Celfocus, has a microservices architecture that aims to promote agile development and scalability of the solution. Several services available at Amazon Web Services were also used, such as Lambda functions, RDS (Relational Database Service), and CloudWatch. For the connection between Alexa and the microservices, it was necessary to create an application that allows the use of voice-activated features in the connected smart devices (Amazon Alexa Skill).

With the development of this project, an integration system was created that connects the voice assistant, Alexa, and the new television products developed by Vodafone. In this way, viewers are allowed to enjoy a new experience when watching television.

Keywords

Voice Assistant, Alexa, Digital TV, Set-Top-Box, Microservices Architecture.

Índice geral

Índice de figuras

Lista de tabelas

Lista de abreviaturas, siglas e acrónimos

API	Application Programming Interface
AWS	Amazon Web Services
BE	Backend
BSS	Business Support Systems
CI/CD	Continuous Integration and Continuous Deployment
CRUD	Create, Read, Update and Delete
DB	Database
DTV	Digital Television
EC2	Amazon Elastic Compute Cloud
GUI	Graphical User Interface
ID	Identificador
IoT	Internet of Things
MM	Message Manager
MQTT	Message Queuing Telemetry Transport
NLP	Natural Language Processing
OSS	Operations Support Systems
POC	Prove Of Concept
RCG	Real Time Communication Gateway
RDS	Relational Database Service
RGPD	Regulamento Geral sobre a Proteção de Dados
S3	Amazon Web Services Simple Storage Service
SDK	Software Development Kit
STB	Set-Top-Box
TV	Televisão
UI	User Interface
UX	User Experience
VFE	Voice Front End
VPC	Virtual Private Cloud
VTV	Vodafone Television

1. Introdução

Um dos recursos previstos como uma extensão significativa do serviço de televisão é a capacidade de o utilizador poder interagir com o serviço usando a voz, em vez de um comando de televisão tradicional. Este já é o caso em alguns cenários, utilizando o microfone embutido que está disponível para versões premium das set-top-boxes (STB) – mais conhecidas como boxes de televisão - de algumas operadoras. No entanto, o aparecimento de assistentes de voz na casa dos clientes é uma tendência crescente e que, provavelmente, vai predominar nos próximos anos.

Com o principal objetivo de fazer com que a interação dos espectadores com o serviço de televisão fosse apenas por comandos de voz, a Vodafone junta-se à Amazon, para que, a partir do seu assistente de voz, a Alexa, seja possível controlar todas as operações nas STB da casa dos seus clientes.

Este projeto, cujo cliente é o grupo Vodafone, foi atribuído à unidade de Digital TV (DTV) da Celfocus, em setembro de 2019, e até à data, continua em desenvolvimento.

1.1. Contextualização do trabalho

Fundada em 2000, a Celfocus (*Home :: celfocus*, 2020) é uma *joint venture* entre a Vodafone Portugal (*Vodafone*, 2020), uma subsidiária integral do Grupo Vodafone, e a Novabase (*Novabase*, 2020), líder portuguesa em Tecnologias de Informação, estando cotada na Euronext Lisbon (*Lisbon | euronext.com*, 2020) desde 2000 (*História*, 2020).

A Celfocus (*Home :: celfocus*, 2020) desenvolve soluções tecnológicas que servem de ajuda à transformação digital dos provedores de serviços de comunicações, oferecendo recursos baseados em componentes que abordam os problemas mais críticos das operadoras. Estes componentes podem evoluir, permitindo a flexibilidade da transformação de projetos nas áreas de Business Support Systems (BSS) e Operations Support Systems (OSS) diminuindo significativamente a complexidade dos desafios que surgem na indústria. A oferta de IoT (Internet of Things) da Celfocus ajuda as operadoras a expandir a própria oferta de IoT, fornecendo soluções para planear e agregar serviços. Desta forma as operadoras podem expor e lucrar com os principais recursos de telecomunicações como um serviço (*Who We Are :: celfocus*, no date). A Celfocus fornece também soluções de comunicação e colaboração que os provedores de serviços precisam para colocar em prática a estratégia de comunicação unificada, isto é, garantir a união total de serviços fixos e móveis. O serviço de Digital TV (Televisão) tem como foco oferecer uma experiência incomparável ao utilizador e para isso acontecer é suportado pelas tecnologias mais inovadoras (*Domains of Expertise :: celfocus*, 2020).

Nos últimos anos, a forma como os clientes têm a experiência de ver televisão mudou significativamente. Esta tornou-se mais rica e complexa, sendo distribuída por vários dispositivos.

Com o objetivo de a experiência do utilizador ser a melhor, é importante para os provedores de serviços de comunicações considerarem a tecnologia usada e, também,

qual a forma de como obter o conteúdo mais atraente. A experiência da televisão do futuro está totalmente relacionada com o que é que o utilizador pode ver, quando e onde.

A Celfocus possui a experiência e a competência para ajudar a suportar esta estratégia, fornecendo assim uma oferta abrangente e ampla, com foco em dar ao utilizador uma experiência incomparável, sendo esta suportada pelas tecnologias mais inovadoras para facilitar e gerir tanto o conteúdo como os parceiros.

Com a finalidade de levar aos consumidores os assistentes de voz, vários produtos, acessíveis, foram lançados para o mercado nos últimos anos (Hoy, 2018). Os assistentes de voz são softwares executados em altifalantes ou smartphones. O software escuta constantemente uma palavra-chave para ser "acordado". Ao ouvir essa palavra-chave, este grava a voz do utilizador e envia a mesma para um servidor especializado, que a processa e interpreta como um comando. Dependendo do comando, o servidor fornecerá ao assistente de voz informações apropriadas para serem lidas de volta para o utilizador, podendo estas reproduzir conteúdo solicitado pelo utilizador ou concluir tarefas que utilizem serviços e dispositivos conectados. O número de serviços que oferecem suporte a comandos de voz está em rápido crescimento, e os fabricantes de dispositivos de IoT estão, cada vez mais, a incorporar o controlo a partir da voz nos seus produtos (Hoy, 2018).

Lançada pela Apple como uma aplicação em 2010 e embebida no sistema iOS no ano seguinte, a Siri é, atualmente, a assistente de voz mais antiga. Depois da Apple, também a Microsoft anunciou, em 2013, a Cortana (Hoy, 2018). Lançada em 2014, pela Amazon, a Amazon Alexa, ou, como é mais conhecida, simplesmente Alexa**,** é uma aplicação controlada por voz para toda a gama de dispositivos Echo (Lopatovska *et al.*, 2019). Por fim em 2016 a assistente da Google foi anunciada, juntamente com o Google Home, altifalante onde está incorporada, e com uma aplicação para smartphones Android (Hoy, 2018).

Embora cada assistente de voz tenha recursos exclusivos, estes partilham algumas semelhanças e são capazes de realizar tarefas básicas como enviar e receber mensagens e e-mails; fazer chamadas; definir temporizadores, alarmes; adicionar eventos ao calendário; responder a questões básicas (ex.: "Que horas são?", "Como está o tempo?",...); controlar a reprodução de media de serviços conectados (ex.: Amazon, Google Play, iTunes, Netflix, Spotify,...); controlar dispositivos IoT (ex.: termostatos, luzes, alarmes, fechaduras,..) (Hoy, 2018).

Para além de todas estas tarefas, os assistentes de voz podem adicionar outros recursos chamados "*skills*" que expandem as suas funcionalidades interagindo com outros programas por meio de comandos de voz (Lopatovska *et al.*, 2019). As *skills* são desenvolvidas por terceiros, de forma semelhante ao desenvolvimento de aplicações para os smartphones. Com a opção de desenvolver *skills* para a Alexa, a Celfocus, iniciou o desenvolvimento de uma *skill* personalizada para este projeto.

Ainda com algumas dúvidas de que seria possível a junção entre os serviços Alexa e as set-top-box (STB) da Vodafone, foi pedida por parte do cliente deste projeto, a Vodafone, uma prova de conceito.

Desenvolvida em setembro de 2019, a *proof of concept* (POC) demonstrou, tanto para a equipa de desenvolvimento como para o cliente, que seria possível fazer a junção da Alexa com a televisão da Vodafone. Desta forma deu-se início ao desenvolvimento do projeto, Voice Front End (VFE), depois de fornecida pelo cliente a especificação dos seus objetivos para a realização de cada tarefa.

1.2. Objetivos

Para fazer a junção dos serviços de televisão do grupo Vodafone com a assistente de voz Alexa, e com o principal objetivo de melhorar a experiência da utilização do serviço de televisão com o apoio dos assistentes de voz, foram definidas algumas tarefas para estre projeto. Todas estas tarefas são objetivos que a equipa de desenvolvimento da Celfocus tem para conseguir desenvolver o projeto e entregar o produto final à Vodafone.

Os objetivos deste projeto são os seguintes:

- Criar, configurar e desenvolver ferramentas para conectar a Alexa com a Set-Top-Box (STB).
- Desenvolver um simulador para a STB.
- Fazer uma prova de conceito.
- Implementar os fluxos que permitem a interação com o utilizador e os comandos de voz.
- Mapear e transformar as diretrizes de voz.
- Desenvolver as páginas de login, na conta Vodafone TV do utilizador, dentro da Alexa App.

1.3. Cronograma

Este projeto, ainda em desenvolvimento, já conta com mais de um ano de tarefas cumpridas. Este relatório tem vindo a ser escrito desde o início do desenvolvimento do projeto para acompanhar o mesmo e cobrir todas as alterações e atualizações que sejam feitas. É previsto que o projeto termine no decorrer de 2021 e, por sua vez, que seja colocado no mercado em 2022. Os países pioneiros na utilização deste serviço serão a Alemanha, a Espanha e a Itália. Na Figura 1 apresenta-se o cronograma que representa a ordem de trabalhos e as tarefas deste projeto.

Figura 1 – Cronograma do projeto

1.4. Planeamento e Organização

Este relatório encontra-se dividido em cinco capítulos.

No primeiro capítulo, é elaborada a introdução, é descrito o âmbito do projeto e quais são os seus objetivos. Também nele se apresenta um cronograma com a ordem de trabalhos, e, por fim, é descrita apresenta-se a organização do relatório.

No segundo capítulo, é abordado o estado da arte, quais as tecnologias similares que estão no mercado e quais as principais concorrentes à solução desenvolvida.

No capítulo número três é descrita a metodologia utilizada no desenvolvimento do projeto e as razões de esta ter sido escolhida.

Seguidamente, no quarto capítulo é abordada a solução desenvolvida, onde são demonstrados os passos necessários para atingir os objetivos, e, quais as ferramentas e estruturas utilizadas para os atingir.

No quinto capítulo são apresentados os resultados obtidos.

No capítulo seguinte, o sexto, é feita a discussão dos resultados.

Por fim, no sétimo capítulo é apresentada a conclusão que inclui uma rápida previsão do trabalho futuro.

2. Estado da Arte

Desde cedo, grande parte dos utilizadores de computadores mostrou a sua vontade de interagir com o mesmo através da voz. Há algumas décadas, a ideia de manter uma conversa significativa com um computador parecia futurista, mas, atualmente, a tecnologia para tornar possíveis e eficientes as interfaces de voz não é mais uma miragem (Këpuska and Bohouta, 2018).

Vários produtos desenvolvidos nos últimos anos trouxeram assistentes de voz acessíveis para o uso diário, e mais funcionalidades e plataformas estão a ser acrescentadas a toda a hora. Os utilizadores podem fazer tudo, desde simples perguntas informativas até reproduzir música, ligar para o telefone de outra pessoa ou ligar e desligar as luzes de sua casa através do controle por voz (Hoy, 2018; Lopatovska *et al.*, 2019).

De uma forma mais simples, pode-se afirmar que os assistentes de voz são a realização do sonho de interagir com os computadores falando com eles (Këpuska and Bohouta, 2018). Atualmente, o mercado dos assistentes de voz é liderado por três grandes empresas, a Apple, a Amazon e a Google (Brill, Munoz and Miller, 2019; Lopatovska *et al.*, 2019). A estas correspondem, respetivamente, as seguintes assistentes: a Siri (Apple Inc., 2011), a Alexa (Amazon, 2020), e a Google Assistant (*Google Assistant, your own personal Google*, no date). Por trás de cada uma delas está software que funciona com dispositivos de altifalantes, propositadamente concebidos para o efeito, ou smartphones (Hoy, 2018; Lopatovska *et al.*, 2019). O software espera, constantemente, uma palavra-chave para acordar. Uma vez ouvida essa palavra-chave, grava a voz do utilizador e envia-a para um servidor especializado, que a processa e interpreta como um comando. Consoante o comando, o servidor fornecerá ao assistente de voz a informação apropriada para ser lida de volta ao utilizador, seja reproduzir conteúdos solicitados pelo utilizador, ou completar tarefas que interajam com serviços e dispositivos conectados. O número de serviços que suportam comandos de voz está a crescer rapidamente, e os fabricantes de dispositivos de IoT estão também a introduzir o controlo por voz nos seus produtos (Hoy, 2018).

Lançada como uma aplicação em 2010 e incluída no iOS em 2011 (Berdasco *et al.*, 2019), a Siri - assistente a Apple - é a mais antiga das principais competidoras do mercado. Pouco depois, a Amazon lançou a Alexa com o seu *home speaker* Echo em 2014 (Lopatovska *et al.*, 2019), e a assistente da Google foi anunciada em 2016 juntamente com o seu *home speaker,* e está também incorporada na aplicação Google para smartphones Android (Hoy, 2018). Cada assistente tem as suas próprias características únicas, mas as funções principais são as mesmas (Hoy, 2018; Brill, Munoz and Miller, 2019). As assistentes de voz atuais diferem das tecnologias anteriores, também ativadas por voz, na medida em que podem responder a um maior número de comandos e perguntas (Berdasco *et al.*, 2019). Isto porque estão sempre ligadas à Internet; cada interação é enviada de volta para um sistema

informático central que analisa os comandos de voz do utilizador e fornece ao assistente a resposta adequada (Hoy, 2018).

Seguidamente são apresentadas as três assistentes acima referidas - Siri, Alexa e Google Assistant – é feita uma descrição de cada uma, quais as suas funcionalidades e por fim apresenta-se uma comparação entre elas.

2.1. Siri

A Siri é uma assistente de voz integrada nos dispositivos que contêm sistemas operativos desenvolvidos pela Apple, como é o caso do iOS, iPadOS, watchOS, macOS, e tvOS (*Use Siri on all your Apple devices - Apple Support*, no date). A assistente utiliza pedidos de voz e uma interface de utilizador em linguagem natural para responder a perguntas, fazer recomendações e executar ações enviando pedidos a um conjunto de serviços da Internet. Com a sua utilização contínua, este software adapta-se aos usos, pesquisas e preferências linguísticas dos utilizadores.

O assistente de voz foi lançado como uma aplicação para iOS em 2010 (Schonfeld, 2010), e foi adquirido pela Apple meses mais tarde (Rao, 2010; Wortham, 2010). A Siri foi então integrada no iPhone 4S no seu lançamento em 2011 (Golson, 2011) . Nessa altura, a aplicação foi retirada da App Store do iOS (Kumparak, 2011). Desde então, a Siri tornou-se parte integrante dos produtos da Apple, tendo sido adaptada para outros dispositivos de hardware ao longo dos anos, incluindo novos modelos de iPhone, bem como iPad, Mac, AirPods, Apple TV (Sumra, 2015), e HomePod (Gartenberg, 2017). A Mangrove Capital Partners fez uma previsão de que a Apple fosse lançar um SiriOS na sua conferência de desenvolvimento de 2020, para fazer crescer ainda mais o ecossistema Siri (Miller, 2019), o que não se verificou. O SiriOS poderia rivalizar com algo como a plataforma Alexa Skills da Amazon, o que facilita aos programadores a implementação da funcionalidade Alexa. Atualmente a Apple oferece o SiriKit aos programadores, mas uma possibilidade era que um SiriOS pudesse funcionar através do iOS, iPadOS, e macOS com facilidade (Miller, 2019).

A Apple oferece uma ampla gama de comandos de voz para interagir com a Siri, entre os quais se podem destacar os seguintes (Purewal and Cipriani, 2017):

- Telefonar e enviar mensagens;
- Enviar emails;
- Saber informações sobre variados temas, como o tempo ou o trânsito;
- Fazer conversões seja de medidas ou de moedas;
- Tirar fotografias;
- Abrir aplicações;
- Ligar ou desligar determinadas funcionalidades, como WiFi ou Bluetooth;
- Definir alarmes;
- Criar eventos no calendário;
- Fazer pesquisas na internet;
- Reproduzir música.

Outras características notáveis da Siri incluem: a capacidade de enviar mensagens áudio, ditar para o texto ser escrito no dispositivo, e traduzir entre diferentes línguas com a nova aplicação "*Translate*" que permite a tradução de conversas e pode funcionar completamente offline (Eckel, 2020).

2.2. Alexa

A Amazon Alexa, também conhecida simplesmente como Alexa, é uma assistente virtual desenvolvida pela Amazon, projetada para os dipositivos Amazon Echo. (Kelly, 2018). Com o objetivo de executar comandos dados por voz, a Alexa, conecta-se, via internet, à Amazon AWS, servidores na *cloud* pertencentes à Amazon, ou a outros dispositivos em rede.

A Alexa é ativada quando o seu software de reconhecimento de voz recebe uma palavra ou frase chave de um utilizador, por exemplo, a palavra "Alexa" pode ser usada para ativar o dispositivo, mas esta palavra pode ser personalizada pelo utilizador (Lopatovska *et al.*, 2019).

A batalha entre os *smart speakers* da Google e da Amazon tem sido constante. Em abril de 2017, a Google lançou uma nova característica para a sua assistente, a capacidade de reconhecer vozes individuais. Desta forma todas as respostas dadas são personalizadas para cada um dos utilizadores de um determinado dispositivo. Desta forma os utilizadores recebem informações sobre a sua agenda ou reproduzem as suas *playlist*s, e quando o utilizador pedir para telefonar a um contacto este não é confundido caso existam vários telefones associados ao mesmo Google Home. Meses mais tarde, em outubro do mesmo ano, a Amazon anunciou que a Alexa conseguia fazer a mesma coisa. Passou, então, a ser-lhe possível configurar o reconhecimento de voz de um utilizador dos dispositivos Echo. Os utilizadores leem 10 frases para Alexa, e a partir dos dados que recebe, esta cria um perfil de voz para o utilizador (Welch, 2017).

A Alexa oferece diferentes funcionalidades como a capacidade de interagir com voz, reproduzir música, fazer listas de tarefas, definir alarmes, reproduzir podcasts, e fornecer informações sobre o tempo, trânsito, além de outras informações em tempo real, tais como notícias. A Alexa pode também controlar vários dispositivos inteligentes tornando-se ela própria um sistema de automação de casas inteligentes. Os utilizadores são capazes de ampliar as capacidades da Alexa instalando *skills* - funcionalidades adicionais desenvolvidas por terceiros, mais comummente chamadas aplicações, como por exemplo programas meteorológicos.

2.3. Google Assistant

A Google Assistant (*Google Assistant, your own personal Google*, no date) é uma assistente virtual desenvolvida pela Google, presente em dispositivos móveis e dispositivos domésticos inteligentes, como é o caso do Google Home. A Google Assistant foi concebida com o objetivo de dar ao utilizador a possibilidade de ter uma conversa com o dispositivo, - o que representa uma evolução em relação ao

assistente anterior, o Google Now, que apenas dava informações (López, Quesada and Guerrero, 2018).

Em maio de 2016 a Google Assistant foi revelada durante a conferência Google I/O (*Google I/O 2021*, 2021), como parte do altifalante inteligente Google Home. O CEO da Google, Sundar Pichai, explicou que a assistente foi desenvolvida para ser uma experiência de conversação com dois sentidos, e também uma experiência de ecossistema que se estende através de dispositivos (Brandom, Dzieza and O'Kane, 2016).

Com a Google Assistant é possível pesquisar na Internet, agendar eventos e alarmes, ajustar configurações no dispositivo, mostrar informações da conta Google do utilizador, juntamente com muitas outras funcionalidades que estão presentes tanto na Alexa como na Siri.

A Amazon, com os dispositivos Echo e com a sua assistente virtual, Alexa, teve um grande avanço no mercado antes da apresentação do Google Home. No entanto desde o lançamento do Google Home e da Google Assistant, a empresa anuncia regularmente novas funcionalidades para tentar recuperar terreno. Um ano depois do seu lançamento, a Google começou a lançar novas e variadas funcionalidades para a sua assistente. Em janeiro de 2017, lançou as "*Actions*" para a Google Assistant. As "*Actions*" são como as "*Skills"* da Amazon Alexa, na medida em que permitem aos programadores de terceiros integrar facilmente as suas aplicações e serviços com o Google Home (Martin, 2017b). Passado um mês, em fevereiro de 2017, a Google anunciou que os utilizadores, com a Google Assitant, podiam comprar produtos no Google Home, através do serviço de compras Google Express (Martin, 2017a; Steele, 2017). Mais tarde, em abril de 2017, foi também lançada uma nova característica que revolucionou o mercado, a capacidade de reconhecer vozes individuais (Welch, 2017).

2.4. Comparação

Com a existência de diferentes assistentes, é necessário fazer uma seleção da qual iria integrar este projeto. Apesar da escolha ser feita por parte do cliente, a Vodafone, e esta estar relacionada com o negócio e não com qual a melhor opção, foi feita uma comparação entre as três assistentes apresentadas.

A Siri, da Apple, é a assistente mais antiga, que tem vindo a ser menos desenvolvida e a perder para as suas competidoras. Uma desvantagem clara são os seus problemas de compatibilidade (Kozuch, 2020), Tabela 1. Em comparação com a Alexa e a Google Assistant que estão disponíveis para dispositivos Android e Apple, a Siri só pode ser utilizada em dispositivos Apple.

A Google Assistant consegue personalizar os resultados que dá aos utilizadores com base no perfil de voz e tem o poder de pesquisa da Google por trás, o que torna as respostas muito assertivas (*Alexa vs Google vs Siri: Which Smart Assistant Is Best for You? - The Plug - HelloTech*, 2020; Kozuch, 2020). No entanto, a expressão "Ok Google" utilizada para acordar os dispositivos não é considerada muito amigável, e também

não é compatível com tantas aplicações e serviços como a Alexa. De facto, esta, não sendo tão inteligente como a rival Google Assistant tem a vantagem de ser compatível com milhares de serviços e aplicações. Outra vantagem ainda é que a sua palavra-chave, Alexa, é personalizável e mais pessoal (Kozuch, 2020). Seguidamente, na Tabela 1 é apresentada uma comparação entre as três assistentes.

Tabela 1 - Tabela comparativa entre Siri, Alexa e Google Assistant

	***Skills* ou Ações**	**Compatibilidade com Android e iOS**	**Capacidade de reconhecer voz**	**Palavra-Chave Personalizável**
SIRI	X	X	X	X
AMAZON ALEXA	✓	✓	✓	✓
GOOGLE ASSISTANT	✓	✓	✓	X

Perante os prós e contras que cada assistente apresentava, a opção caiu sobre a Alexa, assistente da Amazon, não só pela sua grande compatibilidade com diferentes serviços e aplicações, mas porque teve-se também em conta a parceria que, noutros sectores, a Vodafone mantém com a Amazon.

3. Metodologia de Desenvolvimento do Projeto

Neste capítulo irá ser abordada a metodologia utilizada para o desenvolvimento do projeto. Aderir a uma metodologia de desenvolvimento previamente definida permite a um projeto fornecer melhores estimativas e sistemas estáveis, manter o cliente informado, criar uma compreensão clara da tarefa a realizar, e identificar os imprevistos mais cedo, permitindo ter tempo para fazer ajustes. No decurso do projeto, houve mudança em relação à organização do trabalho, e também a metodologia que estava a ser utilizada foi alterada. Esta alteração teve como objetivo ajudar a equipa a melhorar o seu desenvolvimento e também a incrementar a entrega de software. Inicialmente foi utilizada a metodologia Waterfall mas, com o avanço do projeto e com a vontade de introdução de uma forma mais moderna de trabalhar, os processos foram sendo alterados aos poucos, tendo como alvo uma metodologia ágil, neste caso, o Scrum.

3.1. Waterfall

Nos últimos anos a utilização das metodologias ágeis na indústria da tecnologia da informação teve um considerável incremento, devido a fatores como alta produtividade e a qualidade dos produtos elaborados.

No início do desenvolvimento deste projeto foi tomada a decisão, por parte da gestão do projeto e da unidade de DTV, que a metodologia a ser implementada seria a metodologia *Waterfall*, contrariando a tendência do mercado.

O modelo *waterfall*, ou cascata, é um modelo de desenvolvimento de software sequencial. A origem do termo *waterfall* está, frequentemente, relacionada com um artigo publicado por Winston W. Royce, em 1970, no qual é descrita esta metodologia, sem a identificar pelo o nome que lhe é dado hoje, e é também defendida uma abordagem iterativa para o desenvolvimento de software (Royce, 1970). Este modelo é composto por seis fases distintas: a análise de requisitos, desenho do projeto, desenvolvimento, implementação, validação e manutenção, como indica a Figura 2 (Royce, 1970; Zulqadar, 2019).

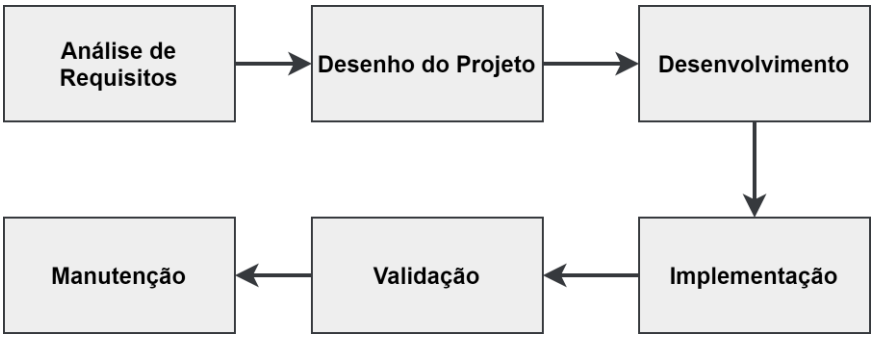

Figura 2 - Fases do Modelo *Waterfall*

W. W. Royce destaca o risco inerente a este modelo: "Eu acredito neste conceito, mas a implementação descrita acima possui riscos e convida às falhas" (Royce, 1970; Rivas and Souza, 2014). Um dos maiores riscos deste modelo é a eventual necessidade de um redesenho de todo o projeto, já que, caso surja um contratempo que não foi previamente detetado, as mudanças necessárias podem ser tão disruptivas que implicam uma alteração de todo o projeto (Rivas and Souza, 2014). Outros pontos negativos deste modelo são a falta de flexibilidade, a necessidade de terminar uma fase para ser possível iniciar a seguinte, a falta de feedback para corrigir erros e também a pouca adaptação a projetos complexos (Chandra, 2015; Dubey *et al.*, 2015).

Todavia também existem pontos positivos associados a esta metodologia. Entre os principais, destacam-se a clara definição dos requisitos antes da fase de desenvolvimento, a fácil implementação do modelo pois este é linear e os recursos necessários são, por norma, poucos e definidos de antemão (Balaji and Murugaiyan, 2012)(Chandra, 2015; Dubey *et al.*, 2015).

Durante os primeiros três meses de desenvolvimento do projeto foi utilizada esta metodologia. Antes da fase de desenvolvimento, e como as boas práticas do método *waterfall* assim sugerem, foi feita uma análise de requisitos por parte da gestão do projeto juntamente com o cliente. Seguidamente passou-se para a fase de desenho do projeto, onde a equipa de desenvolvimento em conjunto com membros da equipa de arquitetura definiram qual seria a melhor estrutura e tecnologias a utilizar. Com as duas primeiras etapas definidas e fechadas, passou-se ao desenvolvimento do projeto. É de realçar que até esta fase não tinham surgido problemas relacionados com a metodologia selecionada. Ao evoluir no desenvolvimento do software a equipa

debateu-se com alguns problemas sendo de realçar a falta de organização por parte da equipa e também da gestão em relação às tarefas que já tinham sido realizadas e às que faltava implementar. Tendo em conta este entrave, sugeri à equipa a utilização de uma ferramenta que ajudasse a acompanhar o estado de cada tarefa, o JIRA.

Desenvolvido em 2002 pela Atlassian Corporation, o JIRA é uma plataforma para ajudar as equipas a gerir o seu trabalho (Fisher, Koning and Ludwigsen, 2013). Originalmente projetado com o objetivo de seguir bugs e defeitos no software, hoje em dia, é utilizado como uma ferramenta de gestão para diferentes casos, desde gestão de requisitos e casos de teste até ao desenvolvimento ágil de software (Fisher, Koning and Ludwigsen, 2013) (Mishra and Mishra, 2013). De forma a organizar as tarefas utilizou-se um quadro Kanban, com o objetivo de fazer com que a equipa visualize o trabalho, limite o trabalho em andamento e maximize a eficiência (ou fluxo) (Rehkopf, 2019). Este quadro organiza-se por colunas. No caso deste projeto, inicialmente existiam três: "To Do", "In Progress" e "Done". Na primeira coluna estavam, em forma de cartões, as tarefas para fazer; na segunda coluna, as que estavam a ser desenvolvidas pela equipa, e por fim, na última coluna, as tarefas já terminadas. Cada cartão tem o tipo de tarefa, quanto tempo irá demorar a ser feito e o nome da pessoa a quem foi atribuído. Na Figura 3 é possível ver um exemplo de um quadro Kanban com os seus cartões.

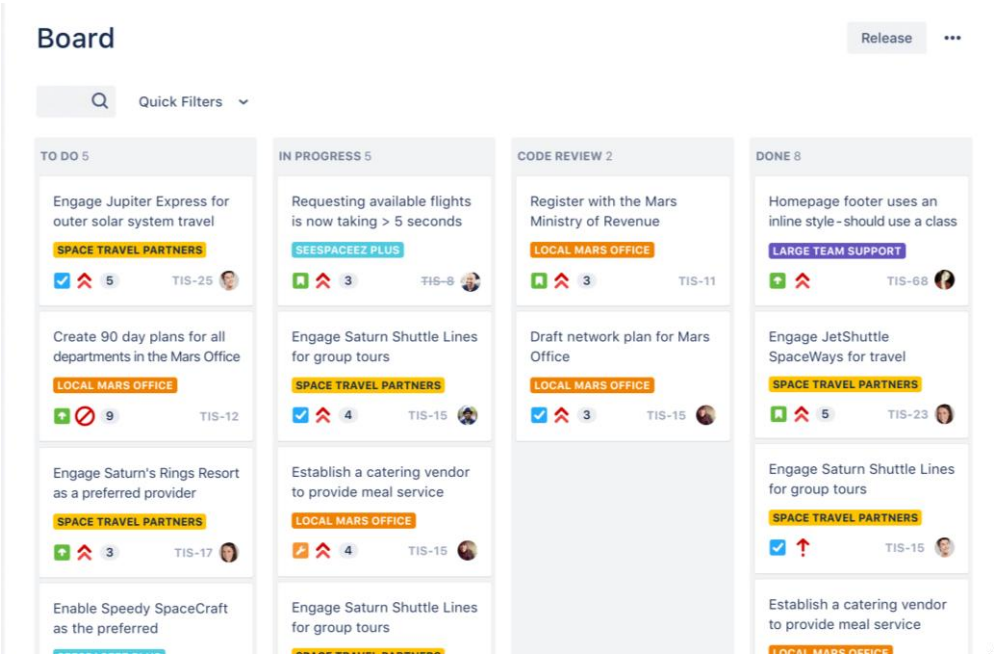

Figura 3 - Exemplo de *Kanban board* no JIRA.

Fonte:(*Jira | Issue & Project Tracking Software | Atlassian*, no date)

A introdução desta ferramenta trouxe vantagens tanto para a equipa como para a gestão. Com o JIRA e o quadro Kanban foi possível manter uma lista das atividades de

conclusão urgente e, ao mesmo tempo, ter uma estimativa de quanto tempo demora o desenvolvimento, pois cada tarefa tem uma duração associada.

No entanto, e apesar desta melhoria, surgiram problemas relacionados com integrações externas e, informado o cliente, este levantou algumas questões em relação a determinados requisitos. Tendo em conta este cenário de impasse e na impossibilidade de voltar à fase inicial do modelo *waterfall* foi necessário mudar a abordagem até aqui utilizada.

Por parte da gestão do projeto foi tomada a decisão de se iniciar uma metodologia mais ágil que fosse de encontro à possível alteração de requisitos, e é neste momento que surge a metodologia *scrum*.

3.2. Scrum

O *scrum* é uma metodologia ágil para gerir projetos de desenvolvimento através de um método iterativo e incremental (Schwaber and Sutherland, 2011). Esta metodologia introduz o conceito de *sprint* que representa uma iteração de um ciclo de desenvolvimento num período, com duração de duas semanas a um mês. Na verdade, o *scrum* é composto por um conjunto de *sprints*, e, no final de cada *sprint* há a entrega de *software* funcional. Além disso, define três funções, quatro artefactos, para aprimorar a comunicação, e cinco reuniões, para aumentar a colaboração (Schwaber and Sutherland, 2011).

Esta metodologia de desenvolvimento está dividida em três funções principais:

- **Scrum Master**: indivíduo responsável pelos processos do *scrum*, utilizando-os de forma correta de forma a maximizar os seus benefícios. Este papel é, normalmente, mal-interpretado e confundido com o papel do gestor de projeto. O *scrum* master tem como objetivo remover impedimentos e barreiras, ser o facilitador, em vez de fornecer um plano de trabalhos e monitorizar a equipa, pois, a mesma deve organizar-se sozinha (Rising and Janoff, 2000).
- **Product Owner**: pessoa responsável pela definição dos objetivos do negócio e pelo alinhamento dos objetivos relacionados com o desenvolvimento. Geralmente, esta função é atribuída a um representante do cliente ou ao gestor de projeto, devido ao conhecimento que estes têm do modelo de negócio. A principal responsabilidade do *product owner* é orientar a equipa no desenvolvimento de um projeto de sucesso.
- **Team (equipa)**: responsável por entregar o produto. Por norma, a equipa é constituída por poucos membros, até a um máximo de dez, com diferentes competências, que são auto organizados e autogeridos. A equipa desempenha as suas funções sob a orientação do *product owner* e são apoiados pelo *scrum master* (Cockburn and Highsmith, 2001).

A relação e a colaboração entre estas funções são cruciais e devem seguir os valores definidos no Manifesto Ágil (Hazzan and Dubinsky, 2014). No entanto, estes valores dificilmente são cumpridos nas empresas (Cockburn and Highsmith, 2001).

Além disso, o *scrum* também identifica quatro artefactos que são assegurados pela equipa ao longo do ciclo de desenvolvimento (Schwaber, 1997; Sutherland, Ph and Scrum, 2007):

- **Product Backlog**: lista priorizada de tudo o que é necessário para concluir o produto. O *backlog* é semelhante ao conceito de contrato de requisitos usado em metodologias de desenvolvimento tradicionais, como *waterfall*.
- **Sprint Backlog**: lista de tarefas a serem realizadas durante um sprint, ou seja, traduz parte do *product backlog* em software funcional. Este tema auxilia os membros da equipa a focarem-se em objetivos a curto prazo, a fim de maximizar a qualidade e o tempo de entrega do produto.
- **Release Burndown Charts**: gráficos que mostram o avanço do projeto ao longo do tempo, permitindo à equipa ter uma visão global do desenvolvimento do mesmo.
- **Sprint Burndown Charts**: gráficos onde a equipa pode ver o progresso do sprint ao longo do mesmo, de modo a acompanhar os desenvolvimentos.

A interação das três funções utilizando os quatro artefactos acima apresentados leva a vários tipos de reuniões que fazem parte desta metodologia estas são (Schwaber, 1997; Sutherland, Ph and Scrum, 2007):

- **Release Planning**: reunião de *kick-off* para definição dos itens iniciais do *product backlog* bem como para a realização de uma análise ao cronograma e orçamento do projeto de desenvolvimento. Geralmente, é uma reunião breve, consumindo apenas cerca de vinte por cento do tempo de uma reunião de planeamento inicial tradicional.
- **Sprint Planning**: a equipa de desenvolvimento e o cliente discutem os temas necessários e definem metas para o próximo sprint. Nesta reunião, o sprint *backlog* é preenchido e é definido um plano para o ciclo de desenvolvimento.
- **Grooming**: reunião onde a equipa juntamente com o *product owner* fazem uma revisão às tarefas que se encontram no *backlog*, com o objetivo de verificar se este tem as tarefas apropriadas, se estão bem priorizadas e se não têm dependências.
- **Daily Scrum**: reunião breve e descontraída de quinze minutos para a equipa de desenvolvimento identificar questões que surjam durante o decorrer do *sprint*, falar sobre o seu desempenho nas tarefas que estão a completar e identificar possíveis melhorias dentro do uso da metodologia.
- **Sprint Review**: demonstração do software desenvolvido no *sprint* ao cliente e outras partes interessadas. Esta reunião é particularmente

relevante, pois recolhe feedbacks importantes dos utilizadores finais e de gerentes que financiam a solução desenvolvida.

- **Sprint Retrospective**: a equipa realiza uma autoavaliação em relação ao último *sprint*. Problemas e métricas de desempenho devem ser detalhados, analisados e usados para obter informações sobre o processo real e sugestões para melhorias futuras.

Na Figura 4, o *scrum* é mostrado como uma metodologia de desenvolvimento iterativa e incremental. A fase de planeamento e arquitetura do sistema ocorre na reunião de *release planning*, enquanto os sprints são compostos por *sprint planning*, *daily scrum*, sprint *review* e *sprint retrospective*. Na fase de fecho o produto final é entregue ao cliente.

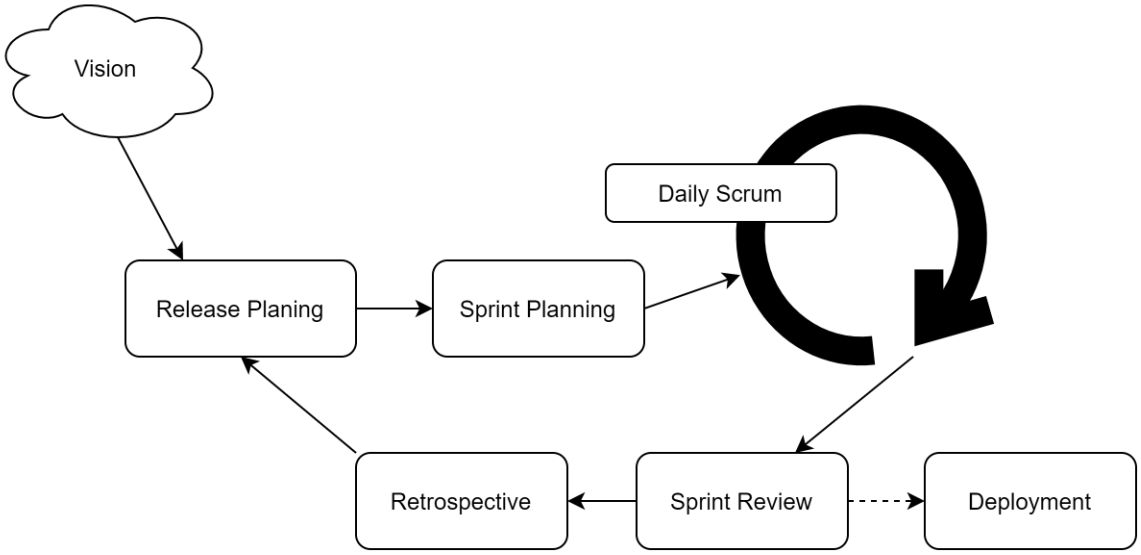

Figura 4 - Fases da metodologia *Scrum*

Embora o *scrum* defina um conjunto de conceitos simples, mas importantes, e que permitem o desenvolvimento ágil de software, a implementação do modelo teórico não garante o sucesso dos projetos por si só, uma vez que é necessária uma mudança de cultura, bem como a adoção de valores ágeis. A adoção de conceitos *scrum* é difícil para equipas que estão separadas ou com clientes distantes, uma vez que a proximidade na colaboração entre as equipas de desenvolvimento e os clientes é enfatizada. Assim, são necessárias soluções complementares para melhorar esta desvantagem.

Devemos, no entanto, realçar que a implementação desta metodologia trouxe vantagens à equipa e ao desenvolvimento do projeto, sendo estas:

- Facilidade de estimar a duração de uma tarefa,
- O trabalho é realizado de forma mais lógica,
- Alterações que surjam são introduzidas rapidamente,
- Maior controlo sobre o cronograma do projeto,
- Aumento da qualidade do que é entregue,

- Maior satisfação por parte do cliente otimizando o tempo e capacidade de resposta às solicitações feitas pela equipa.

Todas estas vantagens foram aparecendo devido, também, a uma grande dedicação por parte da equipa, pois o sentido de cooperação aumentou.

Neste momento, a equipa segue as boas práticas desta metodologia, com elementos seus a desempenhar as funções de *scrum master*, *product owner*, e toda a equipa de desenvolvimento comprometida com os temas, e realizando as reuniões mencionadas. A utilização do JIRA, tal como no modelo *waterfall*, continua implementada, o que é uma mais valia para acompanhar as tarefas tanto durante o *sprint* como no *backlog*.

Durante a fase de transição entre metodologias foram surgindo algumas dúvidas em relação aos procedimentos e boas práticas, por exemplo, no caso da *daily scrum*, esta reunião várias vezes tinha uma duração muito extensa, bem maior do que as boas práticas indicam. Atualmente este ponto foi melhorado; por vezes, a equipa, consegue reuniões com duração inferior a quinze minutos.

3.3. Comparação - Waterfall vs Scrum

A metodologia *waterfall* é uma metodologia tradicional de desenvolvimento de software, na qual todas as fases do processo são realizadas de forma sequencial. Cada fase começa apenas quando a anterior terminou. É possível voltar à fase anterior, mas não é possível voltar no processo, para acomodar por exemplo, uma mudança substancial de requisitos. Esta metodologia requer a definição de um conjunto estável de requisitos, que devem ser apresentados apenas durante a fase de definição de requisitos do sistema, porque futuros feedbacks para as fases anteriores não são facilmente introduzidos. Em resposta a este tipo de metodologias, rígidas e difíceis de seguir, foram introduzidas as Metodologias Ágeis, assim denominadas no Manifesto Ágil (Beedle *et al.*, 2001). Entre estas está o *scrum*, a que atrás foi referida, ferramenta baseada em desenvolvimento de software incremental, onde a entrega regular de software é enfatizada. O *scrum* é uma *framework* ágil e simples, que se adapta a diferentes contextos do desenvolvimento de software. Desta metodologia fazem parte três funções (Product Owner, Scrum Master, Team), seis cerimónias (Release Planning, Sprint Planning, Grooming, Daily Scrum, Sprint Review e Sprint Retrospective) e quatro artefactos (Product Backlog, Sprint Backlog, Release Burndown Charts e Sprint Burndown Charts).

A adoção desta metodologia num projeto implica o uso de iteração com um prazo definido e a divisão do trabalho numa lista de entregas menores, ordenadas por prioridade definida pelo *Product Owner*. Mudanças nos requisitos não são aceites durante a fase de iteração, mas são aceites fora desta fase. Os projetos *scrum* são organizados com a ajuda de reuniões diárias, Daily Scrums, e Sprints, ou iterações, que são projetados para manter o projeto a fluir rapidamente. Geralmente, no final de cada iteração, a equipa entrega o código que realizou. Deste sistema faz ainda parte

uma *Sprint Retrospective*, reunião de retrospetiva, realizada também para procurar maneiras de melhorar o processo para a próxima iteração (Cocco *et al.*, 2011).

A Tabela 2 ilustra as principais diferenças entre estas duas metodologias.

Tabela 2 - Comparação entre a metodologia Waterfall e o *Scrum*

	WATERFALL	**SCRUM**
Objetivo	Projeto	Produto
Mudanças	Não espera mudanças	Espera e aceita mudanças
Documentação	Mais documentação	Menos documentação
Probabilidade de Sucesso	Baixa	Alta
Desenvolvimento	Fase a fase e sequencial	Iterativo e incremental

No caso deste projeto, apesar de as melhorias com a implementação da metodologia *scrum* serem bastantes, verificou-se uma lacuna no que diz respeito à documentação. Sendo este projeto desenvolvido em ambiente laboral, e tendo em conta que a equipa contou com a adição de mais membros, foi necessária documentação que relatasse quais as tarefas já concluídas, em que fluxos se incluíam, quais os componentes já feitos. Desta forma e verificando-se esta necessidade foi decidido por parte da gestão, em conjunto com a equipa, que a cada entrega corresponderia um documento do qual fazem parte o diagrama de relacionamento das entidades da base de dados, o diagrama de componentes no qual são apresentados todos os fluxos, diagramas de sequência para cada um dos fluxos, quais as diretivas de voz que estavam implementadas, entre outros dados como quais as portas em que se encontram expostos os serviços e as definições das APIs que são chamadas por serviços externos.

Em suma o *scrum* foi uma boa mudança na dinâmica da equipa. Apesar dos seus desafios iniciais e do período de adaptação, a organização da equipa melhorou bastante, o que fez com que a produtividade aumentasse. Com a implementação das formalidades desta metodologia foi mais fácil para a gestão do projeto entender quais as maiores dificuldades da equipa e os entraves mais impactantes. O *scrum* serviu também para melhorar a comunicação entre os membros da equipa tendo em conta que na *daily scrum todos* partilhavam a situação atual e eram discutidas soluções para possíveis problemas. O único ponto menos positivo, que já foi referido anteriormente, está relacionado com a falta de documentação desta metodologia mais ágil, que se ultrapassou com a criação de um relatório no final de cada *sprint,* onde constam as funcionalidades aí desenvolvidas.

4. Desenvolvimento da Solução

Neste capítulo será abordada a arquitetura do projeto desenvolvido. Focaremos os componentes desenvolvidos, o modo como se interligam e quais as ferramentas utilizadas para o desenvolvimento. É de salientar que a escolha das ferramentas e da linguagem escolhida para o desenvolvimento dos componentes foi feita conjuntamente entre a equipa de gestão do projeto e o cliente.

4.1. Arquitetura da Solução

Nesta secção vão, primeiramente, ser apresentados os componentes desenvolvidos até à data. Seguidamente serão detalhados apenas alguns dos mais relevantes. É de notar que certos dados sensíveis não podem ser divulgados dada a confidencialidade do projeto, tendo em conta que o mesmo ainda não entrou na fase de produção.

4.1.1. Componentes

Os microsserviços tornaram-se a arquitetura de software preferida para o desenvolvimento de aplicações em diferentes negócios (Knoche, 2016; Nunes, Santos and Rito Silva, 2019). Como referido em (Nunes, Santos and Rito Silva, 2019), inicialmente originados na Netflix e na Amazon, os microsserviços resultam da necessidade da divisão, tanto das equipas de desenvolvimento de software como dos componentes, para, respetivamente, fomentar o desenvolvimento ágil e a escalabilidade horizontal.

Uma arquitetura de microsserviços é composta por um conjunto de serviços originados da decomposição de uma aplicação monolítica, sendo que esses serviços, idealmente, encapsulam diferentes funcionalidades do sistema (Salah *et al.*, 2017; Nunes, Santos and Rito Silva, 2019). De acordo (Nunes, Santos and Rito Silva, 2019), a utilização desta arquitetura tem associadas as seguintes vantagens:

- Aumento da qualidade e rapidez no desenvolvimento de aplicações grandes e complexas.
- Facilidade de implementação com recurso a novas tecnologias, já que cada serviço tem sua própria fronteira - e diferentes serviços podem ser escritos em diferentes linguagens de programação e organizados em função das suas capacidades de negócio.
- Permite escalabilidade independente e elástica, pois cada serviço pode ser replicado de forma autónoma, dependendo da carga de trabalho que tem associada.

Para este projeto a arquitetura selecionada foi, a acima indicada, uma arquitetura de microsserviços. Sendo possível afirmar que a primeira e a terceira vantagem, previamente descritas, foram visíveis. Em relação à segunda, esta não se verificou tão notavelmente pois todos os componentes foram desenvolvidos na mesma linguagem de programação, Java.

Na Figura **5**, dentro do quadrado azul (VPC (Virtual Private Cloud) VFE), é possível verificar-se que este projeto está dividido em vários serviços, cada um com uma funcionalidade diferente. Dentro destes destacam-se:

- **Voice Front End (VFE) Auth:** atua como um servidor de autorização OAuth2, serve também como um intermediário entre o utilizador final e os serviços de autenticação e de descoberta de dispositivos da Kaltura. É o serviço responsável pela comunicação com a API da Kaltura e com o Query Location Service.
- **VFE DB:** microsserviço que implementa operações CRUD (Create, Read, Update, Delete) na base de dados Amazon RDS (Relational Database Service).
- **VFE Default Directives:** responsável pela tradução das Alexa *directives* em ações STB e vice-versa. Recebe *directives* da função Lambda e envia-as para o VFE Device Comms. Em suma, recebe uma Alexa *directive* feita pelo utilizador e responde com um evento de resposta Alexa.
- **VFE Alexa Search:** pedidos de voz relacionados com a procura de séries ou filmes, ou seja, cuja interface é do tipo "RemoteVideoPlayer", são encaminhados para este serviço. Aqui as diretivas vão ser mapeadas para uma query ksql, esta query será enviada à Kaltura e a resposta é enviada ao VFE Alexa Search. Que mapeará os resultados numa ação STB. Este serviço comunica com os serviços VFE Auth, VFE DB e VFE Device Comms.
- **VFE Device Comms:** microsserviço que encaminha as ações STB vindas do VFE Default Directives para o RCG MM e consome mensagens de feedback e de status da STB. As mensagens de feedback, estão relacionadas com a solicitação de ações recebidas e, são enviadas na resposta das ações. As mensagens de status são encaminhadas para o VFE Report. O VFE Device Comms é responsável pela comunicação com o RCG MM e com os serviços de IoT da AWS.
- **VFE Alexa Events:** este serviço recebe Skill Events. Os eventos recebidos podem ser de três tipos diferentes, Account Linked, Skill Enabled ou Skill Disabled. O VFE Alexa Events comunica diretamente com a função lambda e com a base de dados.

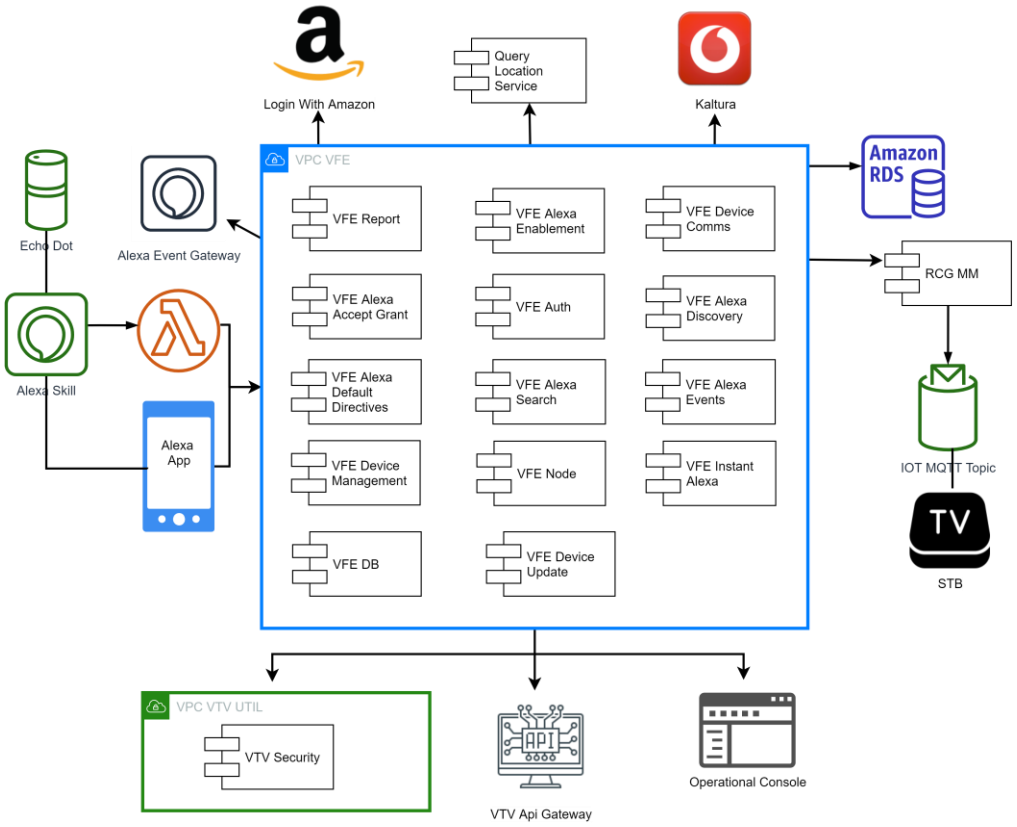

Figura 5 - Diagrama de componentes

4.1.1.1. VFE Auth

O serviço VFE Auth é responsável pela comunicação com a Kaltura e com o Query Location Service, exclusivamente. Tem também interações com o serviço da base de dados. Este serviço é chamado quando o utilizador, após ter feito login na Alexa App e selecionar a *skill,* habilita a mesma. Estas interações são exemplificadas na Figura 6.

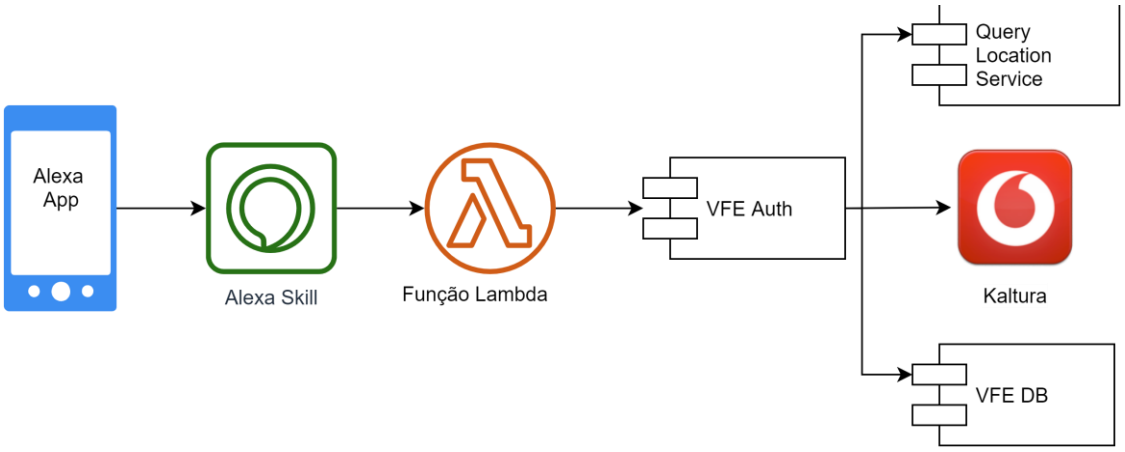

Figura 6 - Fluxo do componente VFE Auth

Após o utilizador ativar a *skill*, é necessário que este se autentique na sua conta Vodafone TV para ser possível identificar quais as STBs associadas a esta mesma conta. Este *login* é feito na API da Kaltura, onde estão guardadas as informações da conta Vodafone TV, inclusive quais as *boxes* subscritas pelo utilizador. Este processo

de descoberta dos dispositivos serve para o utilizador poder selecionar qual o seu dispositivo de *smart home* que quer controlar com o dispositivo Alexa, e é também utilizado no processo de *discovery*. A interface "Alexa.Discovery" descreve mensagens utilizadas para identificar os *endpoints* associados à conta Vodafone TV. Os pedidos de *discovery* chegam por parte da Alexa à função lambda e são reencaminhados para o microsserviço VFE Alexa Discovey. A resposta à diretiva de *discovery* serve para que o utilizador possa descobrir os dispositivos e as *capabilities* associadas à *skill.* As *capabilities* são as interfaces Alexa que a *skill* desenvolvida suporta, por exemplo a *skill* pode suportar operações relacionadas com o volume caso implemente a interface "Alexa.Speaker". As interfaces podem ser comparadas a um conjunto de operações que podem ser realizadas caso estejam presentes na resposta ao pedido de *discovery*.

Com o objetivo de melhorar a experiência do utilizador e de a tornar mais personalizável é feito um pedido ao serviço Query Location. Este indica, com base no endereço IP do dispositivo com que o utilizador faz o emparelhamento da conta Alexa com a conta Vodafone TV, qual a operadora a que pertence. Desta forma as páginas web apresentadas ao utilizador estão na língua que a operadora escolheu por defeito.

4.1.1.2. VFE DB - Base de Dados

Um dos componentes com mais interações nesta solução é o componente da base de dados. Neste são armazenados dados do utilizador. Quando o utilizador faz o emparelhamento da sua conta Vodafone TV com a *skill*, são armazenados no VFE DB os seus dados, os *tokens* de autorização, o identificador único da conta Alexa do utilizador e o *household* ao qual pertence. Na base de dados são também guardadas informações sobre o *household* e os dispositivos, ou seja, STBs, que cada um contém. Neste componente é também feito o armazenamento de configurações das operadoras, além de outras entidades menos impactantes.

Na Figura 7 é possível observar, de uma forma muito geral, as principais relações entre algumas das entidades mais relevantes da base de dados.

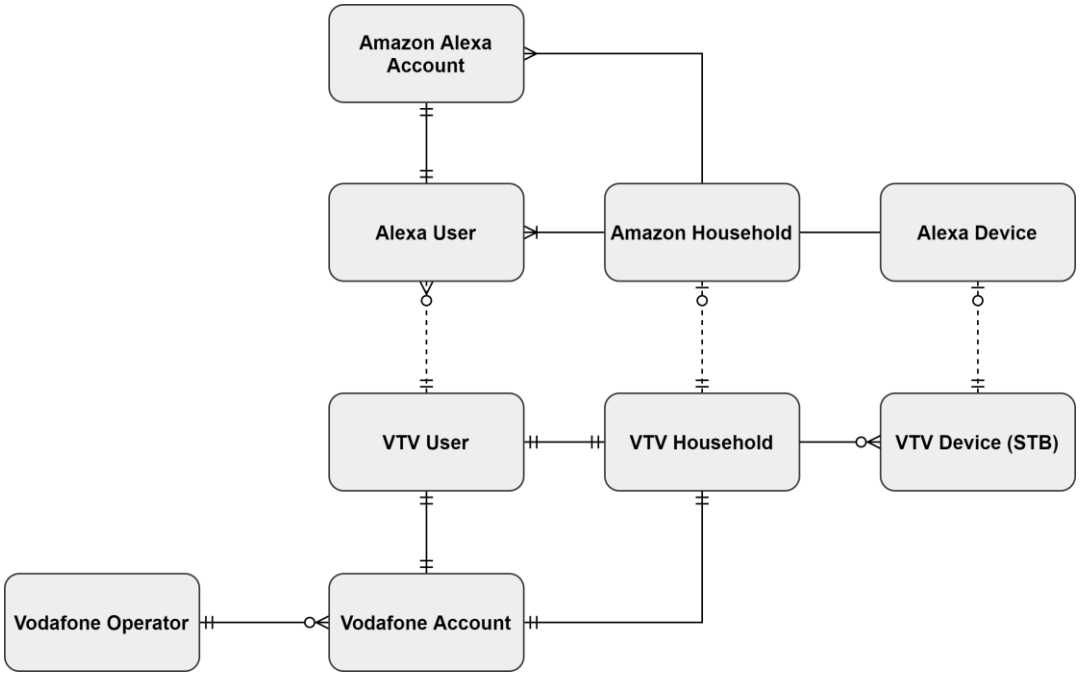

Figura 7 - Diagrama de Entidade-Relação

Em suma, um utilizador que tenha uma conta Amazon Alexa, faz o emparelhamento desta com a conta Vodafone TV (VTV), através da *skill* instalada na Alexa App. Neste momento são guardadas informações sobre o utilizador Alexa (Alexa User) e o utilizador Vodafone (VTV User). Em relação a este é também guardado o seu *household* (VTV Household) e as STBs (VTV Device) que tem associadas à sua conta VTV. No final deste fluxo de emparelhamento o utilizador pode selecionar o dispositivo Alexa com que quer controlar uma das STBs associados à sua conta.

Algumas das entidades apresentadas na Figura 7 não estão presentes na base de dados; são apenas representativas para um melhor entendimento de um dos principais fluxos do projeto, que é o emparelhamento entre a conta Amazon Alexa e a conta VTV.

4.1.1.3. VFE Default Directives

O VFE Default Directives é o serviço responsável pelo mapeamento das diretivas de voz dadas pelo utilizador ao seu dispositivo Alexa. Como a Figura 8 indica este fluxo inicia-se no dispositivo Alexa, por exemplo num Echo Dot, e segue para a *skill*. Esta é ativada com base no pedido que o utilizador faz. Como consequência da ativação da *skill*, a função lambda é chamada. De acordo com o pedido que recebe, esta encaminha o mesmo para o respetivo serviço do VFE.

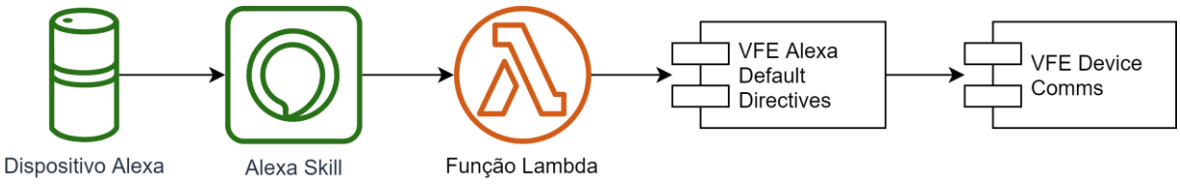

Figura 8 - Fluxo do componente VFE Default Directives

Os pedidos feitos pelo utilizador são analisados pelo sistema de Natural Language Processing (NLP) da Alexa; desta forma é descoberta a intenção do utilizador. Seguidamente a Alexa coloca esta intenção numa mensagem de formato JSON, - formato em que as diretivas são recebidas pela função lambda e, por sua vez, pelo VFE. Todas as diretivas têm associados um *namespace* e um *name* - e são estes campos que indicam a que área a diretiva pertence e qual o objetivo da mesma.

Após chegarem à função lambda e serem reencaminhadas para o VFE Default Diretives, as diretivas, são transformadas em ações STB, o que acontece porque a Alexa e as STBs necessitam de mensagens com formatos distintos. Para tal é preciso um mapeamento entre o *namespace* da diretiva e a operação enviada para a *box*. Este mapeamento foi fornecido juntamente com os outros requisitos necessários para desenvolver esta tarefa.

Em seguida a ação é enviada para o serviço VFE Device Comms depois de ser feito o mapeamento da diretiva da Alexa, cujo campo "*namespace*" é mapeado para o campo "*operation*" da ação que é enviada para a STB.

Após a STB processar este pedido, a resposta ao mesmo é enviada de volta para o VFE Device Comms, que por sua vez, a envia para o serviço VFE Default Diretives. Esta resposta tanto pode ser positiva como negativa, caso tenha sido detetado algum erro durante a execução deste pedido. A construção deste objeto JSON de resposta, por parte da STB, é da responsabilidade da equipa encarregue dos desenvolvimentos das *set-top-box*, que não é a mesma que desenvolveu o projeto aqui documentado.

Por fim, e como último passo deste fluxo, é feito um novo mapeamento. Este trata a resposta que chega através do serviço VFE Device Comms e faz a tradução para uma resposta Alexa.

4.1.1.4. VFE Alexa Search

O VFE Alexa Search é um serviço dedicado às diretivas do tipo RemoteVideoPlayer. As diretrizes deste tipo estão relacionadas com a pesquisa e reprodução de conteúdo tal como séries e filmes (Amazon, no date g). Apesar destas diretivas serem também dadas pelo utilizador, existe a necessidade extra de verificar na API da Kaltura se existe alguma resposta para conteúdo pedido. Como foi referido anteriormente o VFE Auth é o serviço que comunica com a Kaltura, portanto o pedido feito a partir do VFE Alexa Search passa pelo VFE Auth e chega depois à Kaltura. Este serviço comunica também com a base de dados e com o componente que envia as ações para a *box,* como é possível observar na Figura 9.

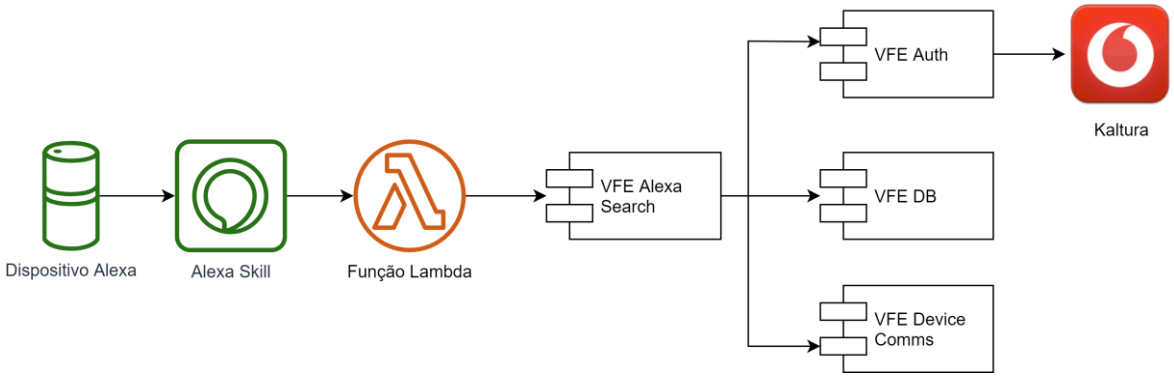

Figura 9 - Fluxo do componente VFE Alexa Search

Quando o VFE Alexa Search recebe a diretiva, proveniente da função lambda, irá utilizar as entidades dos campos "type" e "value" para efetuar um pedido à Kaltura e encontrar os dados correspondentes à consulta. O cliente forneceu uma tabela que contém as informações necessárias para fazer o mapeamento entre as diretivas, vindas da função lambda, e o pedido que será enviado à Kaltura.

Os resultados da pesquisa têm de ser novamente mapeados para ser enviados para a STB. Estes serão também filtrados por conteúdo para adultos. Para isso, cada operadora deverá configurar no VFE os valores que a mesma tenha classificado como "conteúdo adulto". Poderá ser apenas um valor, por exemplo "21" anos, ou vários valores, como "21" e "22" anos.

Em suma, o VFE Alexa Search, recebe a diretiva que provem da função lambda; mapeia a mesma para um pedido que será enviado à Kaltura, através do VFE Auth; recebe a reposta do pedido e traduz o mesmo para uma ação STB, que é enviada para a box através do VFE Device Comms.

4.1.1.5. VFE Device Comms

O VFE Device Comms é reponsável por receber as ações, previamente traduzidas pelo VFE Default Directives e encaminhá-las para o RCG (Real Time Communication Gateway) MM (Message Manager). Por sua vez, o RCG MM comunica com o AWS IoT MQTT que envia as ações para as STBs. Este fluxo é possível observar-se na Figura 10.

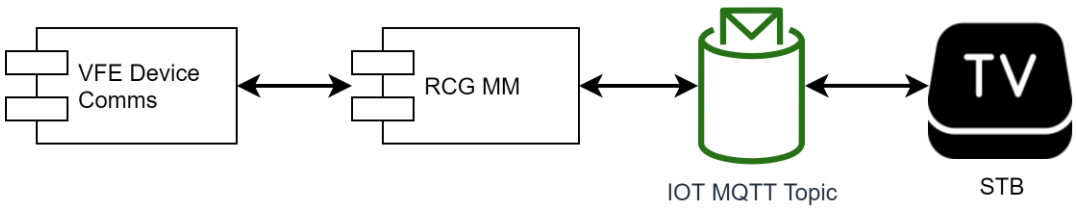

Figura 10 - Fluxo do componente VFE Device Comms

A fim de receber ações invocadas pelo dispositivo Alexa, cada STB deve-se conectar ao RCG. Para tal é necessária a autenticação de um *token* gerado aquando da autenticação do utilizador na conta Vodafone, como foi indicado na secção 4.1.1.1 VFE Auth. A comunicação entre o VFE Device Comms e a STB é feita através do AWS IoT. Este serviço, AWS IoT, permite a comunicação segura entre dispositivos ligados à

internet, neste caso as STBs e a *cloud* AWS, onde está alojado o serviço RCG MM. Neste caso a comunicação é feita por MQTT.

O protocolo MQTT (Message Queuing Telemetry Transport) (*MQTT - The Standard for IoT Messaging*, no date) é um protocolo de mensagens baseado na arquitetura *publish/subscribe* (Rodrigues Martins and Luís Zem, 2015) e precisa de um *broker,* que é responsável por encaminhar as mensagens entre os remetentes e os destinatários (Rotta, Charão and Dantas, 2017). Cada cliente que publica uma mensagem para o *broker* inclui um tópico na mensagem. O tópico é a informação de encaminhamento para o broker. Cada cliente que deseja receber mensagens subscreve um determinado tópico e o *broker* entrega todas as mensagens com o tópico correspondente ao cliente. Portanto, os clientes não precisam de se conhecer, apenas comunicam sobre o tópico. Esta arquitetura permite soluções altamente escaláveis sem dependências entre os produtores e consumidores de dados. No cenário desenvolvido neste projeto, podemos identifica como *publisher* o serviço do RCG MM e como *subscriber* as STBs.

Depois de passarem pelo IoT MQTT as ações são enviadas para as set-top-box, onde são analisadas consoante a sua estrutura. Por fim, e depois de validada a ação, é dada uma resposta que percorre o caminho inverso até chegar ao VFE Device Comms. Como referido anteriormente, na secção 4.1.1.3 VFE Default Directives, esta resposta é então enviada para o VFE Default Diretives e aí formatada para um JSON equivalente a uma resposta Alexa.

4.1.1.6. VFE Alexa Events

Um *skill event* é gerado sempre que um utilizador habilita ou desabilita a *skill,* conecta ou desconecta uma conta de terceiros com a *skill,* concede ou altera permissões à *skill* (*Skill Events in Alexa Skills | Alexa Skills Kit*, no date). Os eventos chegam à função lambda e são enviados para o VFE Alexa Events, onde consoante o seu tipo são efetuadas diferentes operações no componente da base de dados. Na Figura 11 pode observar-se o fluxo dos *skill events*.

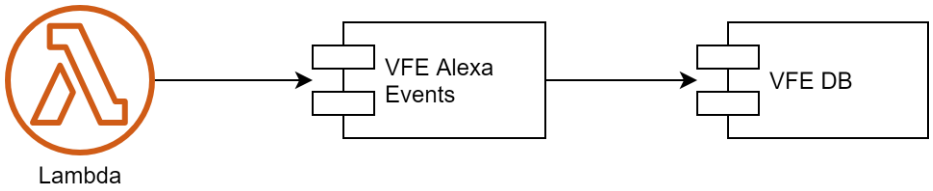

Figura 11 – Fluxo do componente VFE Skill Events

Neste serviço, VFE AlexaEvents, foram ativados três tipos de eventos:

- **Account Linked Event:** Este evento indica que um cliente ligou uma conta numa aplicação de terceiros com a aplicação Alexa. Neste caso a aplicação de terceiros é a Vodafone TV. Neste evento é enviado o "alexaUserId", identificador único da conta Alexa do utilizador. O "alexaUserId" é guardado na base de dados do VFE, para mais tarde servir de comparação com o mesmo campo que está presente no evento de *skill disablement.*

- **Skill Enabled Event:** Evento enviado quando um utilizador ativa a skill. Quando recebido pelo VFE este evento apenas serve de informação e é escrito no ficheiro de *logs*.
- **Skill Disabled Event:** Este evento é enviado quando um cliente desabilita a *skill*. Ao receber este evento o VFE Alexa Events efetua um pedido à base de dados, com o objetivo de apagar todos os dados relacionados com o utilizador. De forma a saber qual o utilizador a eliminar, é enviado no *skill event* o "alexaUserId". Desta forma, e como no Account Linked Event o mesmo identificador é guardado no VFE DB, é procurado o utilizador Alexa que na base de dados tem o mesmo "alexaUserId" que está presente no evento de *skill disablement.*

4.2. Ferramentas Utilizadas

Nesta secção enunciam-se as ferramentas utilizadas para o desenvolvimento do projeto.

4.2.1. Amazon Web Services

Subsidiária da Amazon, a **Amazon Web Services** (AWS) fornece plataformas de computação em nuvem *on-demand* e APIs para indivíduos e empresas, com base no pagamento consoante o uso. Estes serviços web de computação em nuvem oferecem uma variedade de infraestrutura técnica básica abstrata e, também, as ferramentas que servem de base para a computação distribuída.

Um destes serviços é o **Amazon Elastic Compute Cloud** (EC2), que permite ao utilizador ter à disposição um cluster virtual de computadores, sempre disponível, por meio da Internet. As instâncias EC2 têm diferentes tamanhos consoante a necessidade do utilizador, no caso deste projeto é utilizada uma instância do tipo M5 onde estão alojados todos os microserviços.

A **função Lambda** é mais um serviço da AWS relacionado com a computação, tal como o EC2. A tecnologia sem servidor da AWS permite executar funções na nuvem. É uma grande economia de custos, pois apenas se paga quando a função é executada. A função lambda utilizada neste projeto é ativada quando a *skill* recebe um comando de voz, O objetivo desta função é encaminhar as diretivas, consoante a sua interface para o serviço VFE correto.

Outro dos serviços da AWS, utilizado neste projeto é o **Relational Database Service** (RDS), que permite ter uma base de dados relacional, neste caso PostgreSQL na *cloud*. Estas bases de dados são totalmente geridas pela AWS, como a instalação de antivírus e *patches*.

O **CloudWatch** coleta dados de monitoramento e de operações na forma de *logs*, de métricas e de eventos, e visualiza-os usando painéis automatizados de forma a que o utilizador tenha uma visão unificada dos recursos, aplicações e serviços da AWS executados na AWS e em servidores locais. Este serviço é utilizado tanto para os logs dos microsserviços da instância de EC2 como para os da função lambda. Para este

projeto, esta ferramenta tem sido uma grande ajuda na altura de fazer *debug* a algum erro encontrado, por exemplo, durante uma diretiva de voz.

4.2.2. Amazon Alexa Skills

A Amazon Alexa é um assistente digital, mas também se pode considerar uma plataforma como os sistemas iOS ou Android. Tal como estes sistemas mais completos, a Alexa tem aplicações que podem expandir sua utilidade. A Amazon chama estas aplicações *skills*.

As *skills* são aplicações que permitem a utilização de recursos ativados por voz nos dispositivos inteligentes conectados, bem como em serviços online. Os utilizadores interagem com a Alexa ao dizer uma *wake-word*, palavra chave para acordar o dispositivo, seguida de uma frase de invocação, que consiste numa *utterance* (declaração) e o nome de invocação da *skill* (Melton and Fenwick, 2019). Como é possível observar no exemplo em (Melton and Fenwick, 2019), a *utterance* "*Alexa, ask Daily Horoscopes for the horoscope for Gemini*", divide-se em três partes. "Alexa", a *wake-word; "Daily Horoscopes"* o nome pelo qual é invocada a *skill*; "*the horoscope for Gemini*" é a *utterance.*

Cada *skill* tem um modelo de interação que determina as solicitações que esta pode receber e as palavras que os utilizadores devem dizer para invocar esses pedidos. O Alexa Skills Kit oferece modelos já desenvolvidos nos quais as possíveis solicitações e declarações são predefinidas para quem vai desenvolver a *skill*, mas também podem ser definidas ao utilizar um modelo personalizado. O Alexa Skills Kit é um SDK (*software development kit*) que possibilita o desenvolvimento de *skills.* Existem cinco tipos de *skills* diferentes, cada um destes tipos responde a diferentes requisitos. Os tipos de *skills* definidos pela Amazon são:

- **Custom Skill**: uma *skill* personalizada pode lidar com qualquer tipo de solicitações, desde que seja criado código que possa responder às mesmas – e, também, fornecer dados apropriados no modelo de interação para permitir que os utilizadores invoquem a solicitação. Este é o tipo de *skill* mais flexível, mas também o mais complexo, uma vez que é necessários que quem desenvolva este recurso construa o modelo de interação (*Build Custom Alexa Skills - Alexa Skills Kit Official Site*, no date).
- **Smart Home Skill**: para desenvolver uma *skill* para controlar dispositivos domésticos inteligentes, como câmaras, luzes, fechaduras, termostatos e TVs inteligentes, deve usar-se o modelo pré-construído de *smart home*. Este modelo oferece menos controlo sobre a experiência do utilizador, mas simplifica o desenvolvimento, pois não é preciso criar a interface de voz. As *skills* deste tipo são mais fáceis de invocar pelos utilizadores finais, uma vez que eles não precisam de se lembrar de nenhum nome de invocação e podem fazer pedidos simples como " *Alexa, turn on the living room lights*" (*Build Custom Alexa Skills - Alexa Skills Kit Official Site*, no date).

- **Flash Briefing Skill**: este tipo de *skill* dá a oportunidade ao utilizador de receber um briefing rápido sobre um tema. Por exemplo "*Alexa, tell me the news*", neste caso a Alexa indica ao utilizador as últimas notícias de forma sucinta e resumida (*Build Custom Alexa Skills - Alexa Skills Kit Official Site*, no date).
- **Video Skill**: uma *skill* de vídeo possibilita que o utilizador receba conteúdos como séries ou filmes, e também a possibilidade de mudar de canal, invocado o nome ou o número do mesmo. A Alexa fica apta a responder a pedidos como "*Alexa, play Titanic*" ou "*Alexa, change to channel 4*" (*Build Custom Alexa Skills - Alexa Skills Kit Official Site*, no date).
- **Music Skill**: com esta *skill* é possível fornecer conteúdo de áudio, como músicas, listas de reprodução ou estações de rádio para os utilizadores Alexa. Este tipo de *skills* lida com *utterances* que pedem e controlam o conteúdo de áudio, e transforma essas *utterances* em solicitações que são enviadas para a *skill* (*Build Custom Alexa Skills - Alexa Skills Kit Official Site*, no date).

Neste projeto foi utilizada um *skill* de vídeo, pois tendo em conta os requisitos e as necessidades do cliente, é o tipo que melhor se adapta. Com este tipo de *skill* é também possível realizar pedidos de *smart home*, desta forma os utilizadores podem também controlar a sua televisão juntamente com a *box*. Esta *skill* ainda não se encontra na Alexa Skills Store pois o projeto não está terminado.

As *skills* desenvolvidas por terceiros ficam sujeitas a aprovação por parte da Amazon. Após a aprovação, são publicadas na Alexa Skills Store (Amazon, no date i). Em setembro de 2019 estavam disponíveis mais de 100,000 *skills* na Alexa Skills Store (Tankovska, 2020). Existem *skills* de diversas categorias, como negócios e finanças, notícias, previsão do tempo, produtividade, entre outras. Todas as *skills* são gratuitas, embora algumas exijam um serviço de assinatura para o utilizador poder usufruir de todas as suas funcionalidades (Alhadlaq *et al.*, no date).

As *skills* são feitas na Amazon Developer Console (*Amazon Developer Services*, no date), esta consola de desenvolvimento é gráfica, o que facilita os utilizadores da mesma.

4.2.3. Amazon Alexa Interfaces

Na documentação da Amazon Alexa encontra-se uma lista das interfaces (*List of Alexa Interfaces and Supported Languages | Alexa Skills Kit*, no date) que são possíveis de implementar nas *skills.* Em cada uma das interfaces é possível verificar um exemplo do pedido inicial e da resposta que a Alexa espera a esse pedido, o que significa que para todas as interfaces implementadas neste projeto a consulta dessa documentação foi crucial para o esclarecimento de dúvidas e, até, para a ajuda na resolução de problemas relacionados com os mapeamentos. Dessa lista e implementadas na *skill* desenvolvida fazem parte as seguintes interfaces:

- **Alexa.PowerController:** interface que permite aos utilizadores ligar e desligar os dispositivos *smart home* (Amazon, no date d).
- **Alexa.Speaker:** com esta interface os utilizadores podem controlar o volume dos seus dispositivos de entretenimento que disponham de altifalantes (Amazon, no date h).
- **Alexa.PlaybackController:** interface que descreve mensagens que são usadas para reproduzir, parar e navegar na reprodução de conteúdo de áudio ou vídeo (Amazon, no date e).
- **Alexa.SeekController:** esta interface fornece diretivas para navegar para uma posição específica num item de media (Amazon, no date b).
- **Alexa.RecordController:** permite gravar conteúdo e encerrar a operação de gravação atual (Amazon, no date f).
- **Alexa.Launcher:** expõe diretrizes para o lançamento: aplicações, como por exemplo o Amazon Video, e de atalhos da interface gráfica, como a Home Page ou as Configurações (Amazon, no date b).
- **Alexa.ChannelController:** interface que dá aos utilizadores a possibilidade de mudar de canal, quer indiquem o nome ou o número do mesmo (Amazon, no date a).
- **Alexa.KeypadController:** interface que possibilita aos utilizadores o controlo por voz da navegação no ecrã - como é o caso de ações como o *scroll*, a seleção de elementos ou até mostrar mais detalhes de elementos que estão selecionados (Amazon, no date c).

A interface "Alexa.RemoteVideoPlayer" (Amazon, no date g), que envia mensagens que são usadas para pesquisar e reproduzir conteúdo de vídeo, também se encontra implementada. No entanto, para esta foi criado um serviço dedicado pois há a necessidade de realizar um mapeamento extra, para a obtenção de resultados da programação, e pedidos à API da Kaltura.

4.2.4. Composição dos microsserviços - Java, Maven e Spring Boot

Como foi referido anteriormente, este projeto é composto por vários microsserviços Java.

Java é uma linguagem de programação baseada em classes e orientada a objetos, projetada para ter o mínimo de dependências de implementação (Cowell, 1997). Java é uma ótima opção para o desenvolvimento de *microsserviços*. Entre várias razões, a que mais se destaca é a sua sintaxe de anotações, que é fácil de ler. As anotações Java tornam a escrita de *microsserviços* muito mais fácil, especialmente quando ativada por uma estrutura como Spring Boot. É dado muito valor à legibilidade, especialmente quando se trabalha em sistemas complexos. Existem vários fatores que apoiam o desenvolvimento e implementação de *microsserviços* em Java. Esta linguagem de programação oferece uma interface de utilizador, bem como conectividade com recursos de *backend*, tudo dentro dos limites de uma aplicação única, mesmo quando feita de forma independente.

Como referido anteriormente, os microsserviços constituem uma arquitetura que permite aos programadores desenvolver e implementar serviços de forma independente. Cada serviço em execução tem seu próprio processo, e isso faz com que a solução seja suficientemente leve para suportar aplicações de negócios.

Para desenvolver a arquitetura de microsserviços, existem diversas *fameworks,* entre as quais se destaca o Spring Boot, pois necessita de poucas configurações e cria aplicações *stand-alone.*

O Spring Boot é uma extensão da *framework* Spring, onde foram eliminadas as configurações necessárias para configurar uma aplicação Spring. A framework Spring fornece um modelo elaborado de programação e configuração. Tem como objetivo simplificar o desenvolvimento e ajudar quem desenvolve a ser mais produtivo. O Spring concentra-se em várias áreas da aplicação em desenvolvimento e fornece uma ampla gama de recursos. Um dos principais recursos deste *framework* é a injeção de dependências, que ajuda a tornar o desenvolvimento mais simples, pois permite a construção de aplicações pouco acopladas, ou seja, aplicações cujos componentes podem estar divididos. Enquanto o Spring se foca em fornecer flexibilidade, o Spring Boot tem como objetivo reduzir a verbosidade do código e oferecer uma forma mais fácil para o desenvolvimento de aplicações. O Spring Boot diminui o tempo de desenvolvimento de uma aplicação pois contém anotações e código já feito pela *framework*. Ajuda, também, a criar uma aplicação independente, que corre localmente num dispositivo sem a necessidade de um sistema operativo, com poucas ou nenhumas configurações.

Como ferramenta de *build* foi escolhido o Apache Maven (Miller, Vandome and McBrewster, 2010a; The Apache Software Foundation, 2016), ou apenas Maven. Embora nos dias de hoje o Gradle, ferramenta de *build* desenvolvida em Groovy, esteja a roubar algum terreno ao Maven, esta ainda é a ferramenta de construção mais popular para projetos Java (Stalin, 2019).

O modelo de configuração simplificado e baseado em XML do Maven permite que os programadores descrevam ou entendam rapidamente os contornos de qualquer projeto baseado em Java, o que facilita o início de novos projetos (Casey *et al.*, 2007). O Maven também oferece suporte ao desenvolvimento orientado a testes, manutenção de projetos a longo prazo e a sua configuração declarativa e ampla variedade de plug-ins o tornam uma opção popular para CI/CD (*Continuous Integration and Continuous Deployment*) (Casey *et al.*, 2007; Miller, Vandome and McBrewster, 2010b). Do Maven fazem parte três partes essenciais:

- **POM (Project Object Model**): A raiz de um projeto Maven é o arquivo pom.xml, que descreve o projeto e suas dependências, e mostra como fazer o *build* do mesmo (Miller, Vandome and McBrewster, 2010b).
- **Pasta Maven**: A pasta Maven implementa o que é conhecido como convenção em vez das configurações, uma solução elegante perante a complicação das configurações. Em vez de exigir que os programadores

definam o layout e configurem manualmente os componentes para cada projeto, o Maven estabelece uma estrutura de projeto *default* e oferece um formato de ficherio padrão para descrever como funciona. O desenvolvedor apenas conecta os requisitos e o Maven chama as dependências e configura o projeto (Miller, Vandome and McBrewster, 2010b).

- **Repositórios centralizados**: O Maven usa repositórios centralizados para descobrir e publicar pacotes de projetos como dependências. Quando é feita uma referência a uma dependência no projeto, o Maven descobre qual é no repositório centralizado, faz o download para um repositório local e instala no projeto. Na maioria das vezes, este processo é invisível para quem desenvolve (Miller, Vandome and McBrewster, 2010b).

Alguns dos benefícios da utilização desta ferramenta são:

- **Melhor gestão de dependências:** Com o Maven, não há preocupação com as dependências transitivas. Se o projeto depende da biblioteca *A*, apenas se adiciona uma dependência direta de *A* e fica do lado de *A* preocupar-se com a sua própria dependência. O Maven identifica as dependências que não são utilizadas e remove-as, também pode criar relatórios de todas as dependências utilizadas num projeto e até mostrar de forma hierárquica todas as dependências, incluindo as transitivas (*Maven - Benefits of using Maven*, no date).
- **Melhor colaboração:** Os repositórios Maven permitem que o Javadoc de um artefacto seja publicado junto com o JAR do mesmo. Ambientes de desenvolvimento integrados, como o Eclipse, permitem que os programadores façam o download automaticamente do Javadoc e entendam como usar o artefacto (*Maven - Benefits of using Maven*, no date).
- **Estrutura do projeto mais consistente**: Todos os projetos Maven possuem uma estrutura comum, o que torna mais fácil entender cada projeto (*Maven - Benefits of using Maven*, no date).

Em suma, neste capítulo foi abordada a construção da solução e quais as ferramentas utilizadas nesta tarefa. Este projeto está alojado em diferentes serviços da AWS, de onde se destacam o EC2, o RDS e a função Lambda. No caso da *skill*, esta é uma *skill* de vídeo na qual é possível aplicar diretrizes de *smart home*. A *skill* foi criada na Amazon Developer Console, plataforma da Amazon destinada a estes desenvolvimentos.

Desta solução fazem parte vários componentes, desenvolvidos em Java e com uma arquitetura de microsserviços. Para o desenvolvimento de cada serviço foi utilizado o Maven como ferramenta de *build* e o Spring Boot como *framework* cujo objetivo é ajuda os programadores a diminuir o tempo de desenvolvimento. Os microsserviços aqui destacados têm diversas funcionalidades:

- O VFE Auth serve de servidor de autorização, é também o intermediário entre o utilizador final e os serviços da Kaltura, e comunica com o Query Location Service.
- O VFE DB, como o nome indica, representa a base de dados, onde são guardadas informações do utilizador e diversas configurações.
- Quanto ao VFE Default Directives, é responsável pelo mapeamento de diretrizes Alexa em ações STB e vice-versa.
- O VFE Device Commns é o serviço que tem a cargo o envio e recebimento de ações, de e para a STB.

5. Resultados

Neste quinto capítulo serão apresentados os resultados da implementação da solução desenvolvida, nomeadamente o fluxo de login e o da proteção de dados, diretrizes de voz e de pesquisa de conteúdo, e os e *skill events*. Posteriormente são documentados alguns dos fluxos implementados que utilizam os componentes VFE Auth, VFE DB, VFE Default Directives, VFE Alexa Search, VFE Device Comms e VFE Alexa Events.

5.1. Sequência de Login

Após iniciar a *skill* na sua Alexa App, quando o utilizador interage pela primeira vez com esta, necessita de fazer o login na sua conta Vodafone TV, de modo a emparelhar o seu dispositivo Alexa e a sua conta Alexa com a conta e dispositivo Vodafone.

A sequência de login é composta por dois fluxos, o *Accept Grant* e o *Discovery*. O fluxo de *Accept Grant* serve para obter as credenciais que identificam e autenticam um utilizador perante a Alexa (*Alexa.Authorization Interface | Alexa Skills Kit*, no date). Este fluxo é totalmente transparente para o utilizador e utiliza um componente próprio, o VFE *Accept Grant*.

Antes de ser efetuado o fluxo de Discovery, o VFE faz a autenticação do utilizador através da Kaltura, fluxo realizado no componente VFE Auth. É mostrada ao utilizador uma página onde este poderá escolher se pretende introduzir o seu nome de utilizador e palavra-chave ou um código. Seguidamente a Kaltura autenticará o utilizador, devolvendo o identificador do *household* e o ID do utilizador. São também apresentadas ao utilizador as condições relativas à proteção de dados, este fluxo é detalhado seguidamente na secção 5.2.

No fluxo de Discovery ocorre a ligação entre os dispositivos. Inicialmente, é apresentada ao utilizador a lista de STBs registadas no *household*, a partir da qual este selecionará um. Para serem mais fáceis de reconhecer, as STBs têm um nome dado por defeito, chamado *friendly name* que devem ser configuráveis por operadora e devem ter um número associado, por exemplo TV Box 1. Da mesma forma, será apresentada uma lista com os dispositivos Alexa registados no *household,* da qual o utilizador irá selecionar um. Esta ligação será armazenada no *backend* da Alexa, onde

é definido o identificador da STB selecionada como o identificador do dispositivo VTV. Este fluxo tem um componente dedicado, o VFE Discovery.

Com o objetivo de mostrar as páginas do fluxo de login na língua do utilizador, é realizada uma chamada ao serviço Query Location Service, que serve para saber a que operadora o utilizador pertence com base no seu IP. Neste pedido é enviado o endereço IP do utilizador e o mesmo é mapeado para a operadora correspondente, ou seja, caso o IP seja de Portugal a operadora é a Vodafone PT. Cada operadora tem uma língua selecionada por defeito, na qual quer apresentar as informações aos seus utilizadores. Desta forma quando o utilizador ativa a *skill* é feito este pedido para a página inicial ser apresentada na língua selecionada por defeito pela operadora, que em princípio corresponde à língua do país do utilizar.

Na Figura 12 é possível observar o fluxo de login, tal como o utilizador o vê. O utilizador clica no botão "*Enable To Use*" que serve para habilitar a *skill*. Depois de redirecionado para o segundo ecrã apresentado, pode escolher se quer entrar na sua conta VTV com credenciais ou com um código. No exemplo aqui apresentado, o botão "Login" é carregado, o que significa que o utilizador necessita de introduzir o seu nome de utilizador e a sua palavra chave. Caso estas estejam corretas, é apresentada a página do RGPD (Regulamento Geral de Proteção de Dados) onde, após aceitar as condições aqui descritas, é redirecionado para uma página que indica que a *skill* foi corretamente ligada. Ao fechar esta página é iniciado o fluxo de Discovery, onde são depois mostrados os dispositivos presentes no *household* e seguidamente os dispositivos Alexa. Ao selecionar um de cada ecrã fica então com o dispositivo VTV e o dispositivo Alexa emparelhados e está pronto a iniciar as diretivas de voz.

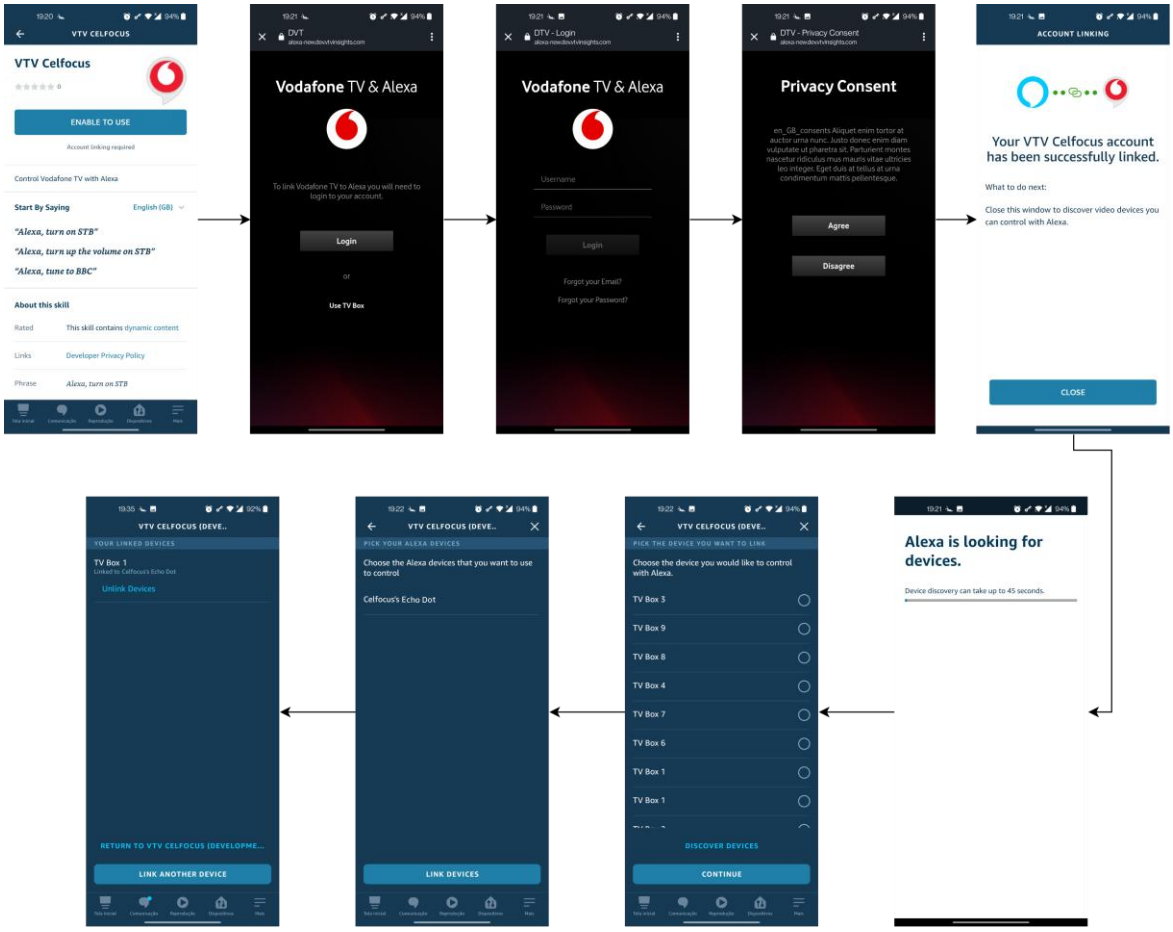

Figura 12 - Fluxo de login

5.2. Fluxo RGPD

O fluxo de RGPD (Regulamento Geral de Proteção de Dados) (Assembleia da República, 2019), é apresentado ao utilizador quando este entra pela primeira vez na conta VTV, no contexto da *skill*. Na verdade, pode ser apresentado um de dois fluxos, o *Privacy Notice*, ou Aviso de Privacidade, e o *Privacy Consent*, Autorização de Privacidade. A diferença entre estes fluxos, é que o primeiro é um aviso e o segundo uma autorização dada pelo utilizador, tal como o nome indica.

A seleção de qual dos fluxos se deve apresentar ao utilizador está a cargo das operadoras. Estas definem qual deles mostrar e também indicam em que língua é apresentado. A mensagem mostrada ao utilizador é guardada dentro de um ficheiro num *bucket* S3. Um *bucket* S3 é um recurso público de armazenamento na *cloud* disponível na AWS S3 (Amazon Web Services Simple Storage Service) (Rouse and Carty, 2018). Este *bucket* é criado pela operadora, portanto o seu nome e o caminho para aceder ao mesmo é também indicado nas configurações dadas pela operadora.

Na Figura 13, é apresentada a página do fluxo de Aviso de Privacidade, no qual o utilizador apenas necessita de selecionar o botão “Continue”, ou seja, continua com o fluxo de login depois de ser notificado. No caso da Autorização de Privacidade, o utilizador pode ou não concordar com os termos que lhe são apresentados. Caso

aceite os mesmo então o fluxo continua e é redirecionado para a página que indica o sucesso do emparelhamento das contas e dos dispositivos. Caso o utilizador não concorde com os termos que lhe são apresentados, depara-se-lhe, então, o acesso a uma página que lhe indica que vai perder recomendações e características específicas que lhe tornariam o serviço uma melhor experiência. Nesta página surgem duas opções, a de voltar atrás e optar por concordar com a autorização ou a de continuar, que redireciona o utilizador para a página inicial. Os dois ecrãs estão representados na Figura 14.

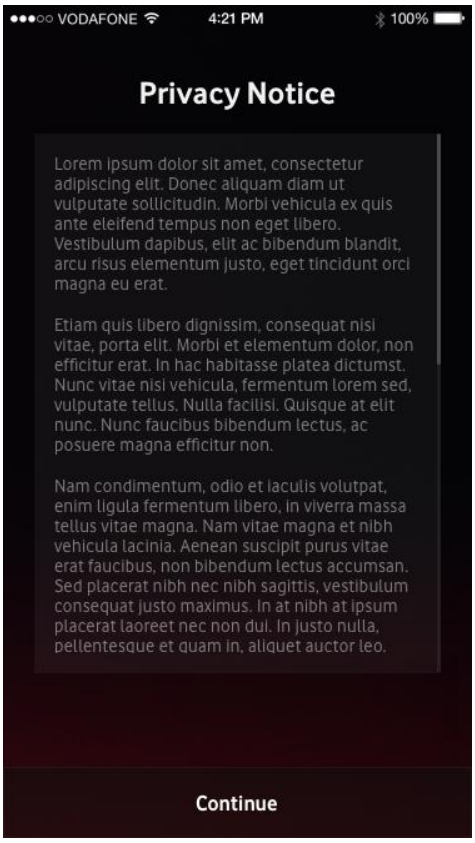

Figura 13 - Aviso de Privacidade - Fonte: própria

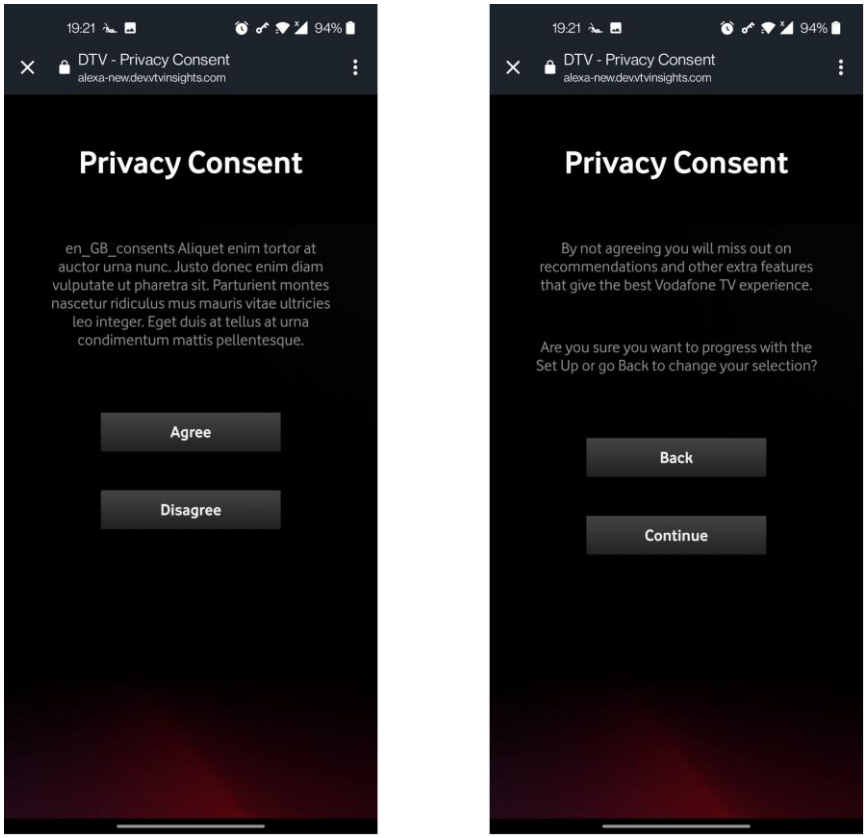

Figura 14 - Autorização de Privacidade - Fonte: própria

5.3. Diretrizes de voz

As diretrizes de voz são dadas pelo utilizador ao seu dispositivo Alexa. Estas chegam num formato JSON à função lambda e são encaminhadas neste mesmo formato para um de dois serviços. No caso de o comando de voz pertencer à interface *"RemoteVideoPlayer"*, diretrizes relacionadas com a pesquisa de conteúdo, o serviço que recebe a mesma é outro, o VFE Alexa Search. Para todas as outras interfaces implementadas, e referidas na secção 4.2.3, o serviço chamado é o VFE Defaut Directives. O diagrama da Figura 15 representa a interação que é iniciada pelo utilizador ao dar o comando de voz ao dispositivo Alexa, até o mesmo chegar à STB e ser refletido na televisão, passando pela função lambda e pelos serviços desenvolvidos neste projeto para esta interação.

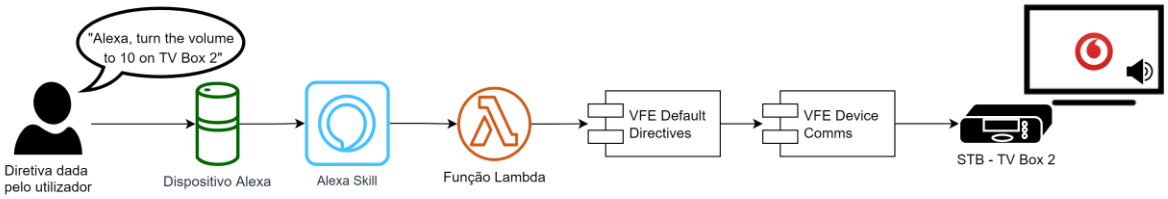

Figura 15 - Fluxo de uma diretriz de voz dada pelo utilizador

Todas as diretivas JSON que chegam ao VFE têm um formato similar. As principais diferenças encontram-se nos campos "*namespace*" e "*name*", que mudam consoante o pedido do utilizador; o "*endpointId*" que é o identificador único da STB para a qual os

comandos estão a ser enviados e o "*payload*" que contém diferente informação de acordo com o tipo da diretiva. Quanto aos campos "*messageId*" e "*correlationId*" estes são gerados automaticamente pelo Alexa Backend. O campo "*token*" contém o Bearer Token, *token* utilizado com o serviço de autorização OAuth 2.0. Este é gerado durante o fluxo de *login*, quando o utilizador emparelha a sua conta Alexa com a conta Vodafone TV. É enviado em todas as diretivas dadas pelo utilizador, e depois de descodificado, contém diversas informações que são uteis aos componentes do VFE para autenticar pedidos.

Posteriormente à chegada das diretrizes ao VFE, estas são mapeadas para ações STB, isto porque as set-top-boxes não estão preparadas para receber o JSON criado pelo Alexa Backend. Este mapeamento é efetuado com base nas especificações entregues pelo cliente. Quanto ao mapeamento da diretiva em ação, são de destacar dois campos. O campo "*operation*" que indica qual o tipo de diretiva. O campo "*parameters*" contem um par chave-valor, em que a chave é a operação que o utilizador pretende fazer e o valor correspondente.

Após ser criada, esta ação passa para o serviço VFE Device Comms, responsável pelo envio das ações para as STBs. Neste serviço é estabelecida a conceção ao RCG MM e por sua vez ao AWS MQTT, pois só através deste serviço, que utiliza o protocolo MQTT, é possível publicar mensagens para mais tarde a STB fazer a subscrição das mesmas. Depois de subscritas as ações são validadas por parte da STB e é dada uma resposta para o VFE Device Comms. Resposta esta que é encaminha para o serviço de onde partiu a ação, VFE Default Directives ou VFE Alexa Search. A ação é novamente mapeada para um JSON que corresponde a uma resposta Alexa e é enviada para a assistente de voz. No esquema da Figura 16 é possível observar-se os campos JSON relevantes para o mapeamento da diretiva em ação e da transformação contrária, de ação para diretiva. Os campos do JSON aqui apresentados tem como base o comando de voz "*Alexa, turn the volume to 10 on TV Box 2*".

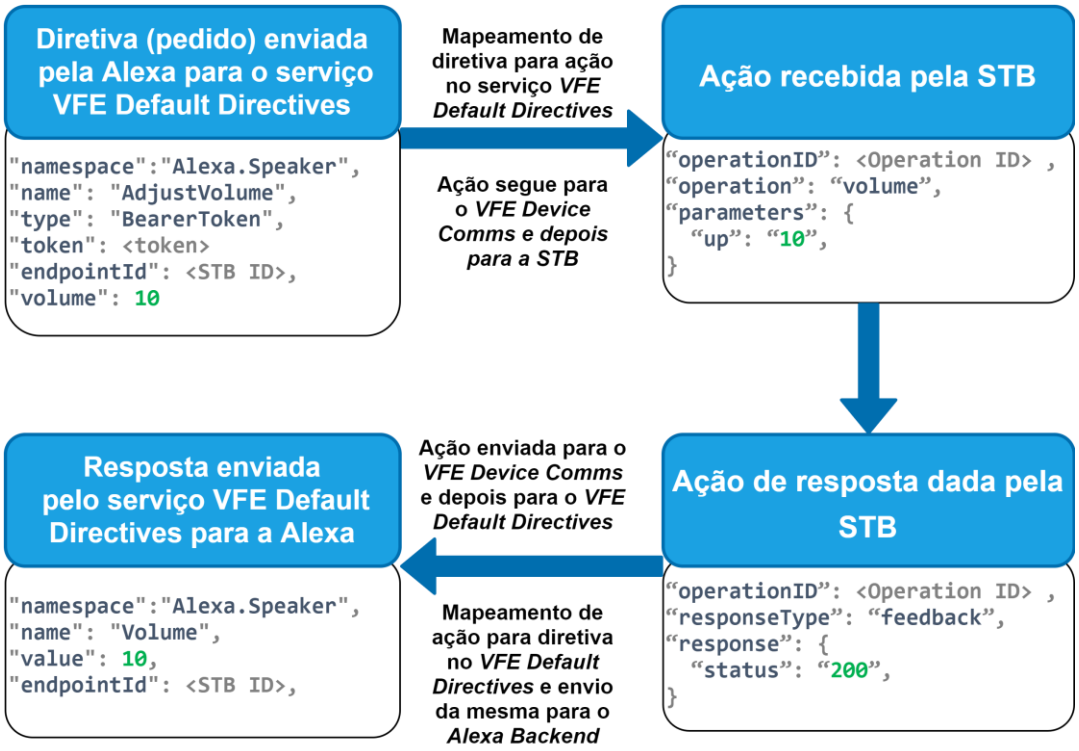

Figura 16 - Mapeamento das diretivas em ações

5.4. Diretrizes de pesquisa de conteúdo - VFE Alexa Search

Estas diretivas são similares às apresentadas na secção 5.3-Diretrizes de voz, sendo as diretivas do tipo "*RemoteVideoPlayer*" redirecionadas para um serviço dedicado. Com a necessidade de efetuar pedidos extra, tanto à Kaltura, como à base de dados, surgiu a necessidade de criar um serviço dedicado, o VFE Alexa Search. Responsável por receber as diretivas que contêm pedidos relacionados com a procura e a reprodução de conteúdo de vídeo.

As diretivas de procura de conteúdo dividem-se em duas categorias, "*SearchAndPlay*" e "*SearchAndDisplayResults*". A primeira, o utilizador faz a pesquisa pelo conteúdo que quer ver e seguidamente o mesmo é reproduzido, este cenário reproduz-se quando a diretiva é, por exemplo, "*Alexa, play Game of Thrones*". No caso de o utilizador fazer um pedido como "*Alexa, find movies directed by Tarantino*", a diretiva que é recebida chama-se "*SearchAndDisplayResults*" pois primeiro procuram-se os resultados e depois mostram-se as opções ao utilizador. Ambas as diretivas têm a mesma estrutura JSON, a diferença entre elas reconhece-se através do valor do campo "*name*". O diagrama da Figura 15 representa a interação que é iniciada pelo utilizador ao fazer um pedido ao dispositivo Alexa, até o mesmo chegar à STB e ser refletido na televisão, passando pela função lambda e pelos serviços necessários para as diretivas do tipo "*RemoteVideoPlayer*".

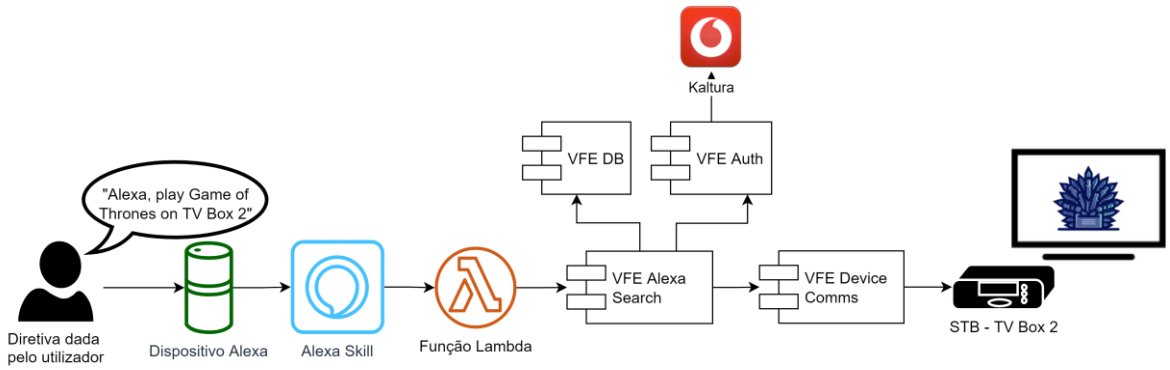

Figura 17 - Fluxo da uma diretriz de voz do tipo "*RemoteVideoPlayer*"

Os resultados aqui apresentados têm como exemplo o pedido "*Alexa, play Game of Thrones*". Esta diretiva é enviada pela função lambda e recebida pelo VFE Alexa Search. O passo seguinte, respeita o mapeamento desta diretiva num pedido à Kaltura, para tal é utilizada uma tabela que contém os mapeamentos a efetuar entre os campos "*value*" e "*type*" do objeto "*payload*", exemplificados na Figura 18. Esta tabela foi fornecida pelo cliente e está espelhada no serviço VFE DB, mas devido à confidencialidade do projeto não será apresentada.

Posteriormente serão retornados resultados por parte da Kaltura. Estes são introduzidos no pedido de ação que será enviado para a STB, como é possível observar no campo "*parameters*" da Figura 18. Neste exemplo apenas se encontra um objeto para facilitar a interpretação por parte do leitor, por norma o resultado é um array.

Por fim, tal como no fluxo das diretrizes de voz, é retornada uma resposta da box para o VFE Device Comms que será encaminhada para o VFE Alexa Search. Neste serviço, é feito o mapeamento do resultado da operação para que seja apresentada a resposta da Alexa ao utilizador.

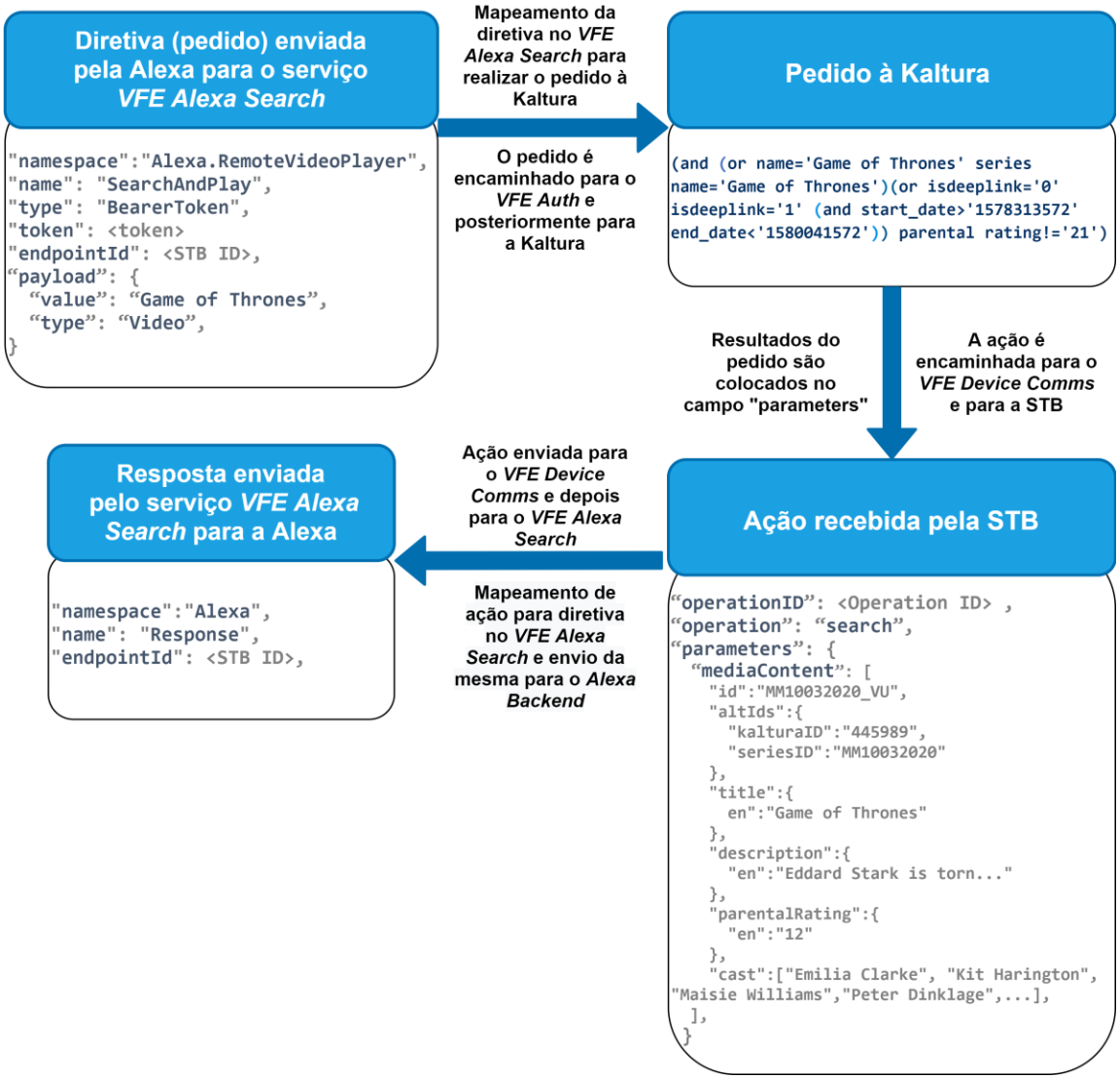

Figura 18 - Mapeamento das diretivas de pesquisa de conteúdo

5.5. Skill Events

Com o objetivo de receber a informação de que a *skill* foi ativada ou desativada por parte de um utilizador, os Skill Events foram habilitados para a *skill* desenvolvida.

O primeiro evento a ser recebido pelo VFE é o Account Linked Event. Este evento está relacionado com a ligação entre a conta Vodafone TV e a Alexa App. Portanto, logo que o utilizador insere os seus dados corretamente na página de login, é enviado este evento. Daqui é retirado o "userId" e guardado na base de dados como "alexaUserId", pois é o identificador único da conta Alexa do utilizador. Em seguida é mostrada uma mensagem JSON de exemplo de um evento do tipo "AlexaSkillEvent.SkillAccountLinked":

```
{
  "version": "string",
  "context": {
    "System": {
      "application": {
        "applicationId": "string"
      },
      "user": {
        "userId": "string",                          #Identificador único da conta Alexa
        "accessToken": "string",                        #Token para autenticar o pedido
        "permissions": {
          "consentToken": "string"
        }
      },
      "apiEndpoint": "https://api.amazonalexa.com"
    }
  },
  "request": {
    "type": "AlexaSkillEvent.SkillAccountLinked",      #Tipo do Evento
    "requestId": "string",
    "body": {
      "accessToken": "string"
    },
    "timeStamp": "string",
    "eventCreationTime": "string",
    "eventPublishingTime": "string"
  }
}
```

Logo a seguir ao Account Linked é enviado um evento do tipo Skill Enabled. Este evento é despoletado quando depois do login, o utilizador aceita as condições de privacidade e segue para o fluxo de Discovery. No final deste fluxo e de o emparelhamento entre uma STB e um dispositivo Alexa ter sido completado, é enviado para o VFE um evento de Skill Enabled, que significa que a *skill* foi ativada com sucesso. Este evento não tem uma utilização específica no VFE; serve apenas de registo para ser possível saber que a *skill* foi ativada com sucesso.

Por fim, o Skill Disabled Event. Este evento é enviado quando o utilizador desativa a *skill*. Os objetivos deste evento são: informar o VFE que o utilizador desativou a *skill* e remover as informações relacionadas com o mesmo, que foram previamente armazenadas na base de dados. Deste tipo de evento resulta uma mensagem JSON de onde é retirado o "userId" para ser comparado com o mesmo identificador que foi guardado quando se recebeu o Account Linked Event. Em seguida é apresentada essa mensagem.

```
{
  "version": "string",
  "context": {
    "System": {
      "application": {
        "applicationId": "string"
      },
      "user": {
        "userId": "string",                          #Identificador único da conta Alexa
      },
      "apiEndpoint": "https://api.amazonalexa.com"
    }
  },
  "request": {
    "type": "AlexaSkillEvent.SkillDisabled",        #Tipo do Evento
    "requestId": "string",
    "timeStamp": "string",
    "eventCreationTime": "string",
    "eventPublishingTime": "string",
    "body": {
      "userInformationPersistenceStatus": "PERSISTED"
    }
  }
}
```

6. Discussão dos Resultados

Neste sexto capítulo são analisados os resultados apresentados no capítulo 5. Temas como a usabilidade, ou a qualidade do software da solução criada são aqui discutidos. É feita uma revisão aos testes efetuados não só pela equipa de qualidade como também pela equipa de desenvolvimento. Seguidamente é descrita a forma como será efetuada a integração da solução. Por fim é abordada a possibilidade da reutilização da solução desenvolvida.

6.1. Usabilidade

Como foi possível observar no capítulo 5, referente aos resultados, esta solução tem uma reduzida componente de interação gráfica com o utilizador. Os ecrãs desenvolvidos para a sequência de login (secção 5.1) e para o fluxo RGPD (secção 5.2) foram criados pela equipa de UI/UX (User Interface and User Experience) da Celfocus e mais tarde aprovados pelo cliente deste projeto. Esta equipa tem como foco a criação e o desenvolvimento da camada de *front-end* para aplicações web e mobile. Os requisitos do cliente obrigavam a que estas páginas seguissem o *template* previamente definido e utilizado pela Vodafone. Quanto às páginas do fluxo de login, estas tinham de ser as mais simples e diretas possíveis, de modo a que o utilizador conseguisse executar esta tarefa facilmente, sem ter dúvidas. Para o fluxo RGPD seguiu-se a mesma estratégia, utilizando-se também o *template* da Vodafone. Como os ecrãs com mais interação com o utilizador são muito similares aos da Vodafone e porque os utilizadores desta solução são previamente utilizadores da Vodafone, o cliente decidiu que não seria necessária a execução de testes de usabilidade.

6.2. Testes e Qualidade do Software

Como as boas práticas indicam, todo o software desenvolvido tem de ser testado. A solução desenvolvida no decorrer deste projeto não é exceção. Desde o início dos desenvolvimentos foram sempre efetuados testes unitários. Este tipo de testes tem como objetivo testar partes individuais do código, como por exemplo um determinado método. Os testes unitários têm como principal vantagem a deteção prévia de quaisquer problemas sempre possíveis, o que facilita alterações futuras. De forma a melhorar a integração entre diferentes serviços foram também desenvolvidos testes integrados, que têm como objetivo encontrar falhas quando se juntam serviços que foram desenvolvidos separadamente, antes de que todo o sistema possa funcionar em pleno. Além de serem testadas diferentes funcionalidades com os testes integrados também é possível verificar qual a performance e a fiabilidade da solução. Tanto os testes unitários como os testes integrados foram desenvolvidos pela equipa responsável pela programação da solução.

Dentro da equipa que compõe este projeto está uma pessoa dedicada à realização de testes de qualidade, testes da solução como um todo. O objetivo deste membro da equipa é encontrar erros, cenários não contemplados, requisitos que não estejam corretamente implementados, tudo isto no software desenvolvido.

Por norma, no final de cada sprint é entregue uma nova versão do software o que significa que são precisos testes não só para as novas funcionalidades daquela versão, como também para verificar se nenhuma outra funcionalidade foi afetada por estas novas alterações.

Para as novas funcionalidades é necessário desenhar o caso de teste, pensar quais os cenários que é preciso testar e realizar os testes. Sempre que é lançada uma nova versão da solução, onde se modificam ou adicionam funcionalidades, são realizados testes de regressão que servem para verificar se erros previamente corrigidos não voltaram a aparecer com as novas alterações e também se as funcionalidades que já estavam implementadas continuam a funcionar sem erros.

No caso deste projeto, como estes testes são apenas realizados no final do sprint, a equipa de desenvolvimento está sempre um passo à frente da equipa de qualidade - situação, que, por vezes, se torna prejudicial, pois, quando detetada uma falha que precisa de ser imediatamente corrigida ou algum problema que bloqueie toda a solução, esta tem de ser colocada dentro do sprint, que está a decorrer, o que não é uma boa prática.

6.3. Robustez

Uma arquitetura de microsserviços é composta por um conjunto de serviços onde todos eles comunicam entre si, o que significa que a ocorrência de erros pode fazer com que toda a aplicação seja posta em causa. Desta forma, e com o objetivo de minimizar os erros, sempre que possível, o erro é apanhado e é lançada uma exceção com o erro juntamente com uma justificação que indique de forma clara e explícita o

que aconteceu. Para os erros do *back-end*, foi deixado ao critério dos desenvolvedores os erros que deviam ser lançados juntamente com a justificação para os mesmos. No caso do *front-end* foi dada pelo cliente uma tabela que indica qual o erro que deve ser retornado ao utilizador nos diferentes cenários de erro. Como exemplo, na Tabela 3 são apresentados alguns desses erros.

Tabela 3 - Exemplos de erros apresentados no *Front-End*

Fluxo	Descrição	Mensagem em Inglês para Apresentar ao Utilizador
LOGIN COM CREDENCIAIS	Username ou password errados	*Sorry, there was an error with your username or password. Please check your details and try again.*
LOGIN COM CREDENCIAIS	Utilizador não está ativo	*Sorry, this account is currently locked. Please contact our call center.*
LOGIN COM PIN	Pin expirado	*Sorry, we couldn't sign you in. Please try again later.*
GDPR	Erro a guardar os dados do GDPR (versão ou data)	*Unable to save Profile. Please try again.*
QUERY LOCATION	Atividade proibida neste país. Por favor tente de novo quando estiver no país onde a conta está registada	*Unable to retrieve your region. Please try again later.*

Com base no fluxo onde o erro ocorre, a mensagem de erro tem uma descrição e de acordo com essa descrição é apresentada ao utilizador a mensagem da terceira coluna. Analise-se em mais detalhe a primeira linha da tabela. No caso de o utilizador inserir um *username* ou uma password errada durante o fluxo de login, é despoletado um erro no *back-end* cuja razão indica o que está descrito na coluna da descrição: *username* ou *password* errados. Seguidamente esse erro é analisado e é enviado para o *front-end* onde depois é feito o mapeamento para a mensagem que vai ser apresentada ao utilizador, como se pode observar na Figura 19.

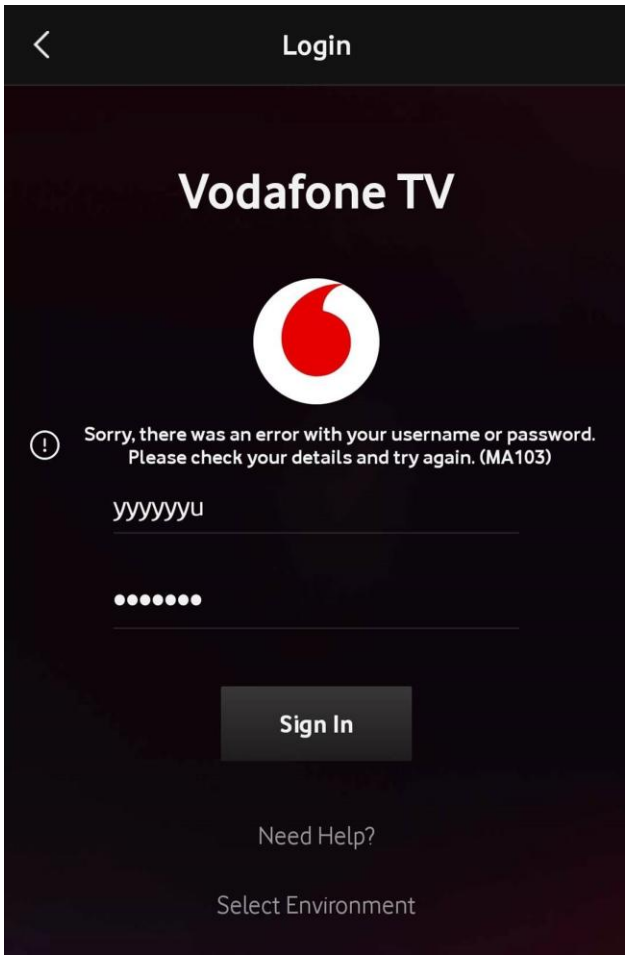

Figura 19 - Erro no Login - Username ou Password errados

6.4. Integração Prática da Solução

Durante todo o desenvolvimento deste projeto foi utilizado um simulador para simular a box. Até ao finalizar deste documento o software para a box, desenvolvido por uma empresa externa à Vodafone e à Celfocus, ainda está em fase de desenvolvimento. O simulador, desenvolvido pela Celfocus em Node.js (*Node.js*, no date), foi utilizado para imitar a futura STB, que será utlizada pelos clientes desta solução. O simulador teve como base um já existente fornecido pelo cliente.

Na solução final, a que será colocada no mercado, os utilizadores terão duas opções para fazerem a conexão da Alexa com a box. A primeira, onde o utilizador conecta o seu dispositivo Alexa à sua STB, Figura 20, será provavelmente a mais utilizada, devido à elevada quantidade de utilizadores que, nos dias de hoje, possuem já um dispositivo Alexa.

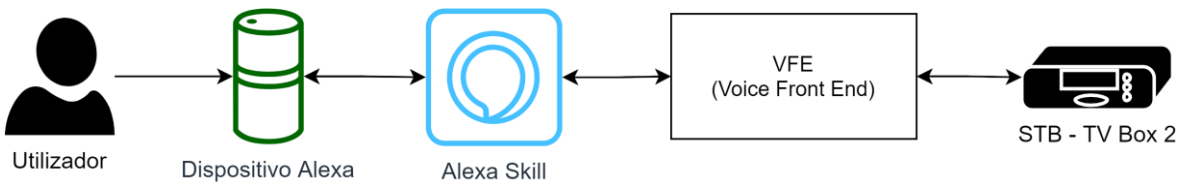

Figura 20 - Integração prática da solução - cenário 1

A alternativa ao cenário anterior é dedicada aos utilizadores que não têm um dispositivo Alexa. Na alternativa apresentada na Figura 21, as novas boxes terão a aplicação da Alexa previamente instalada, e os utilizadores podem aceder à mesma e ativar a aplicação de modo a poderem dar os comandos de voz, também a partir do comando da televisão, que inclui um microfone. Atualmente, a box VBoxPro (*Tv Box - Vodafone Portugal*, no date) que é fornecida com o plano premium da Vodafone, já tem um comando que permite que o utilizador indique quais são as ações que quer fazer através da voz. Com a adição da Alexa App dentro da box, os utilizadores vão ter acesso a mais funcionalidades da Alexa, como perguntarem como está o tempo ou, até, pedir à Alexa para contar uma piada.

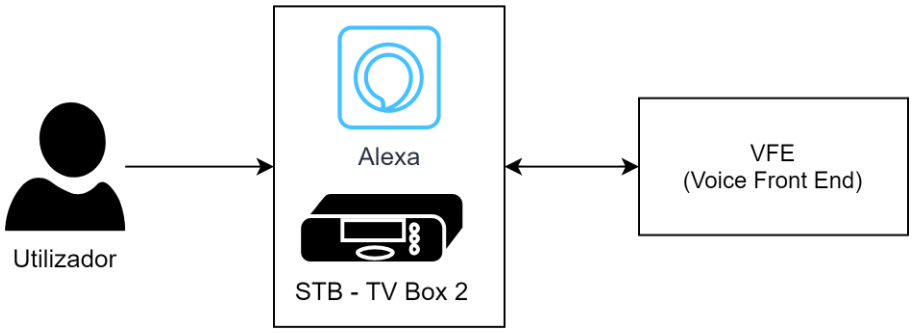

Figura 21 - Integração prática da solução - cenário 2

6.5. Reusabilidade da Solução

A solução criada no desenvolver deste projeto teve como base a Alexa, o que envolve as suas APIs e os seus formatos de pedidos e respostas com base nos pedidos feitos pelos utilizadores. Como é possível observar na Figura 22, o produto final deste projeto, o VFE, foi feito com o objetivo de servir de "tradutor" de pedidos, entre a Alexa e as Boxes. A Alexa recebe o pedido que o utilizador faz, por voz, ao dispositivo Alexa, formata-o num pedido JSON e envia-o para o VFE. No VFE esse pedido é traduzido numa mensagem JSON adequada à STB, para que esta possa processar o pedido. Depois é dada uma resposta por parte da STB a esse mesmo pedido, resposta essa que tem de ser reencaminha para a Alexa. Então o mesmo fluxo ocorre, mas desta vez no sentido inverso.

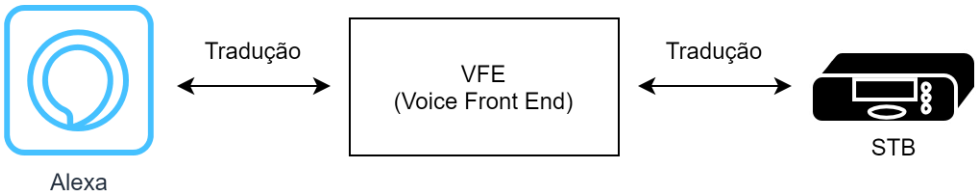

Figura 22 - Diagrama simplificado da solução desenvolvida

Desta forma e como consequência do desenho da solução, a reutilização do Voice Front End para outros assistentes de voz torna-se impossível. O VFE foi pensado com base na Alexa, nos pedidos que esta necessita de fazer às diferentes APIs com que comunica e também com base no seu formato de pedidos e respostas. Diferentes assistentes de voz têm necessidades de comunicação com APIs diferentes, como

diferem também os formatos dos pedidos e das respostas. No caso de existir a necessidade de desenvolver o mesmo projeto para outro assistente de voz pode-se reaproveitar alguma da lógica e até alguns dos serviços desenvolvidos neste projeto, mas todos os serviços que comunicam diretamente com o assistente de voz teriam de ser refeitos. Só assim estariam aptos para receber pedidos e enviar respostas com um formato adequado ao assistente de voz em questão.

7. Conclusão

Assistentes como a Google Assistant, a Amazon Alexa e a Siri tornaram-se enormes nos últimos anos. A presença destas assistentes não está só nos respetivos *smart speakers*, mas também em *smartphones* e *tablets*, cada um oferecendo diferentes vantagens e com as suas próprias características.

O serviço de televisão tem vindo a ser melhorado ao longo dos anos. Uma funcionalidade que vem revolucionar a experiência do utilizador ao ver televisão é a capacidade de o mesmo poder interagir com o serviço usando a voz, para substituir o comando tradicional. Para tal, a Vodafone junta-se com a Amazon, para que com a Alexa seja feita a implementação desta nova funcionalidade.

Este projeto teve como objetivo principal desenvolver um sistema que fizesse a integração da Alexa com o software das *boxes* de televisão da Vodafone, de modo a que o utilizador ditasse os comandos por voz para a Alexa e esta os encaminhe para a STB.

Para a integração da Alexa com a televisão da Vodafone, foi necessário desenvolver vários elementos, como a Alexa Skill, a função Lambda e os serviços que conectam a Alexa à set-top-box. No caso deste projeto foi desenvolvido um simulador para a STB pois o produto final ainda não está disponível. Foi feita uma prova de conceito ao cliente para demostrar que era possível realizar este projeto com sucesso. A implementação de todas as interfaces para a realização dos comandos de voz foi terminada. As páginas de login, na conta Vodafone TV do utilizador, dentro da Alexa App, foram desenvolvidas.

Em suma, este projeto veio possibilitar os clientes da Vodafone a terem uma nova experiência ao ver televisão. A partir de agora, os utilizadores deste serviço têm a opção de falar com a sua televisão e desta forma tornar a sua vida mais inteligente, sem a necessidade de estar constantemente à procura do comando da televisão.

7.1. Trabalho Futuro

Para a conclusão deste projeto, falta fazer algumas melhorias em determinados componentes e corrigir falhas que surjam, detetadas pela equipa de testes.

A principal melhoria, ainda pendente, é a integração do Spectrum. O Spectrum, é um projeto, desenvolvido internamente pela Celfocus, que tem como objetivo priorizar os canais que estejam disponíveis em HD. Caso o utilizador peça para mudar de canal, indicando o nome do mesmo, o VFE faz um pedido ao Spectrum para saber se o canal tem uma versão HD. Caso essa versão exista, o pedido é enviado para a STB com o canal HD.

Outro fator importante para a conclusão deste projeto é o software que fará parte das boxes de televisão comercializadas pela Vodafone. Como foi referido anteriormente, foi necessário o desenvolvimento de um simulador para imitar o software das STB pois este ainda não está concluído. Neste momento, este software é o elemento que vai ditar o sucesso do termino deste projeto, pois o sucesso da sua integração com o projeto desenvolvido é fundamental.

Também a título de trabalho futuro, um projeto similar será desenvolvido em conjunto com a Google, para a integração do seu *smart speaker*, Google Home. De forma a tornar, a experiência de ver televisão mais inovadora e dinâmica para o cliente, mas, também, para expandir o leque de assistentes de voz disponíveis com este serviço.

8. Referências

Alexa.Authorization Interface | Alexa Skills Kit (no date). Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-authorization.html (Accessed: 10 March 2021).

Alexa vs Google vs Siri: Which Smart Assistant Is Best for You? - The Plug - HelloTech (2020) *Hello Tech* . Available at: https://www.hellotech.com/blog/alexa-vs-google-vs-siri (Accessed: 10 June 2021).

Alhadlaq, A. *et al.* (no date) 'Privacy in the Amazon Alexa Skills Ecosystem', in *Proceedings on Privacy Enhancing Technologies Symposium*. Available at: https://www.petsymposium.org/2017/papers/hotpets/amazon-alexa-skills-ecosystem-privacy.pdf.

Amazon (2020) *Amazon Alexa Official Site: What is Alexa?* Available at: https://developer.amazon.com/en-US/alexa/ (Accessed: 21 December 2020).

Amazon (no date a) *Alexa.ChannelController Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-channelcontroller.html (Accessed: 12 November 2020).

Amazon (no date b) *Alexa.Launcher Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-launcher.html (Accessed: 12 November 2020).

Amazon (no date c) *Alexa.Launcher Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-keypadcontroller.html (Accessed: 12 November 2020).

Amazon (no date d) *Alexa.ModeController Interface | Alexa Skills Kit*, *Amazon - developer documentation*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-powercontroller.html (Accessed: 12 November 2020).

Amazon (no date e) *Alexa.PlaybackController Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-playbackcontroller.html (Accessed: 12 November 2020).

Amazon (no date f) *Alexa.RecordController Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-recordcontroller.html (Accessed: 12 November 2020).

Amazon (no date g) *Alexa.RemoteVideoPlayer Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-remotevideoplayer.html (Accessed: 12 November 2020).

Amazon (no date h) *Alexa.Speaker Interface | Alexa Skills Kit*. Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/alexa-speaker.html (Accessed: 12 November 2020).

Amazon (no date i) *Alexa Skills | Amazon.com*. Available at: https://www.amazon.com/alexa-skills/b?ie=UTF8&node=13727921011 (Accessed: 13 November 2020).

Amazon Developer Services (no date). Available at: https://developer.amazon.com/ (Accessed: 18 November 2020).

Apple Inc. (2011) 'Siri - Apple', p. https://www.apple.com/siri/. Available at: https://www.apple.com/siri/ (Accessed: 21 December 2020).

Assembleia da República (2019) 'Lei 58/2019, 2019-08-08 - DRE', pp. 3–40. Available at: https://dre.pt/pesquisa/-/search/123815982/details/maximized (Accessed: 10 March 2021).

Balaji, S. and Murugaiyan, D. M. S. (2012) 'WATEERFALLVs V-MODEL Vs AGILE: A COMPARATIVE STUDY ON SDLC', *International Journal of Information Technology and Business Management*, 2(Software Development), p. 30. Available at: www.jitbm.com (Accessed: 3 November 2020).

Beedle, M. *et al.* (2001) 'Manifesto for Agile Software Development'. Available at: http://agilemanifesto.org/ (Accessed: 14 June 2021).

Berdasco, A. *et al.* (2019) 'User Experience Comparison of Intelligent Personal Assistants: Alexa, Google Assistant, Siri and Cortana', in *13th International Conference on Ubiquitous Computing and Ambient Intelligence UCAmI*. Toledo,Spain. doi: 10.3390.

Brandom, R., Dzieza, J. and O'Kane, S. (2016) *The 10 biggest announcements from Google I/O 2016 | The Verge*. Available at: https://www.theverge.com/2016/5/18/11701030/google-io-2016-keynote-highlights-announcements-recap (Accessed: 9 June 2021).

Brill, T. M., Munoz, L. and Miller, R. J. (2019) 'Siri, Alexa, and other digital assistants: a study of customer satisfaction with artificial intelligence applications', *Journal of Marketing Management*, 35(15–16), pp. 1401–1436. doi: 10.1080/0267257X.2019.1687571.

Build Custom Alexa Skills - Alexa Skills Kit Official Site (no date). Available at: https://developer.amazon.com/en-US/alexa/alexa-skills-kit/get-deeper/custom-skills (Accessed: 13 November 2020).

Casey, J. *et al.* (2007) *Better Builds with Maven. The How-to Guide for Maven 2.0*. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Better+Builds+with+Maven.+The+How-to+Guide+for+Maven+2.0#0%5Cnhttp://trolese.ch/stuff/DLFE-52.pdf (Accessed: 18 November 2020).

Chandra, V. (2015) 'Comparison between Various Software Development Methodologies', *International Journal of Computer Applications*, 131(9), pp. 7–10. doi: 10.5120/ijca2015907294.

Cocco, L. *et al.* (2011) 'Simulating kanban and scrum vs. waterfall with system dynamics', in *Lecture Notes in Business Information Processing*. Springer Verlag, pp. 117–131. doi: 10.1007/978-3-642-20677-1_9.

Cockburn, A. and Highsmith, J. (2001) 'Agile software development: The people factor', *Computer*, 34(11), pp. 131–133. doi: 10.1109/2.963450.

Cowell, J. (1997) 'The Java Language', in *Essential Java Fast*, pp. 24–42. doi: 10.1007/978-1-4471-0629-6_4.

Domains of Expertise :: celfocus (no date). Available at: https://www.celfocus.com/home/domains-expertise (Accessed: 3 November 2020).

Dubey, M. A. *et al.* (2015) 'COMPARATIVE STUDY: WATERFALL V/S AGILE MODEL', *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY*. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.695.9278 (Accessed: 3 November 2020).

Eckel, E. (2020) *Apple's Siri: A cheat sheet , TechRepublic*. Available at: https://www.techrepublic.com/article/apples-siri-the-smart-persons-guide/ (Accessed: 22 December 2020).

Fisher, J., Koning, D. and Ludwigsen, A. P. (2013) 'Utilizing Atlassian Jira for Large-Scale Software Development Management *', *Project Management and Collaboration*, pp. 505–508.

Gartenberg, C. (2017) *Apple announces HomePod speaker to take on Sonos , The Verge*. Available at: https://www.theverge.com/2017/6/5/15732144/apple-homepod-speaker-announced-siri-price-release-date-wwdc-2017 (Accessed: 22 December 2020).

Golson, J. (2011) *Siri Voice Recognition Arrives On the iPhone 4S* , *MacRumors*. Available at: https://www.macrumors.com/2011/10/04/siri-voice-recognition-arrives-on-the-iphone-4s/ (Accessed: 22 December 2020).

Google Assistant, your own personal Google (no date). Available at: https://assistant.google.com/ (Accessed: 21 December 2020).

Google I/O 2021 (no date). Available at: https://events.google.com/io/?lng=en (Accessed: 9 June 2021).

Hazzan, O. and Dubinsky, Y. (2014) 'The Agile Manifesto', in *SpringerBriefs in Computer Science*, pp. 9–14. doi: 10.1007/978-3-319-10157-6_3.

História (no date). Available at: https://www.novabase.pt/pt/dp/historia (Accessed: 21 December 2020).

Home :: celfocus (2020). Available at: https://www.celfocus.com/home/ (Accessed: 21 December 2020).

Hoy, M. B. (2018) 'Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants', *Medical Reference Services Quarterly*, 37(1), pp. 81–88. doi: 10.1080/02763869.2018.1404391.

Jira | Issue & Project Tracking Software | Atlassian (no date). Available at: https://www.atlassian.com/software/jira (Accessed: 5 November 2020).

Kelly, S. M. (2018) *Growing up with Alexa: A child's relationship with Amazon's voice assistant* , *CNN*. Available at: https://edition.cnn.com/2018/10/16/tech/alexa-child-development/index.html (Accessed: 22 December 2020).

Këpuska, V. and Bohouta, G. (2018) 'Next-Generation of Virtual Personal Assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)', in *IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. doi: doi:10.1109.

Knoche, H. (2016) 'Sustaining runtime performance while incrementally modernizing transactional monolithic software towards microservices', in *ICPE 2016 - Proceedings of the 7th ACM/SPEC International Conference on Performance Engineering*. doi: 10.1145/2851553.2892039.

Kozuch, K. (2020) *Alexa vs. Google Assistant vs. Siri: Which smart assistant is best?*, *Tom's Guide Website*. Available at: https://www.tomsguide.com/us/alexa-vs-siri-vs-google,review-4772.html (Accessed: 10 June 2021).

Kumparak, G. (2011) *The Original Siri App Gets Pulled From The App Store, Servers To Be Killed* , *TechCrunch*. Available at: https://techcrunch.com/2011/10/04/the-original-siri-app-gets-pulled-from-the-app-store-servers-killed/ (Accessed: 22 December 2020).

Lisbon | euronext.com (no date). Available at: https://www.euronext.com/en/markets/lisbon (Accessed: 21 December 2020).

List of Alexa Interfaces and Supported Languages | Alexa Skills Kit (no date). Available at: https://developer.amazon.com/en-US/docs/alexa/device-apis/list-of-interfaces.html (Accessed: 12 November 2020).

Lopatovska, I. *et al.* (2019) 'Talk to me: Exploring user interactions with the Amazon Alexa', *Journal of Librarianship and Information Science*, 51(4), pp. 984–997. doi: 10.1177/0961000618759414.

López, G., Quesada, L. and Guerrero, L. A. (2018) 'Alexa vs. Siri vs. Cortana vs. Google Assistant: A Comparison of Speech-Based Natural User Interfaces', in *Advances in Intelligent Systems and Computing*. Springer Verlag, pp. 241–250. doi: 10.1007/978-3-319-60366-7_23.

Martin, T. (2017a) *Google Home now lets you shop for everyday items with your voice*. Available at: https://www.cnet.com/home/smart-home/google-home-now-lets-you-shop-for-everyday-items-with-your-voice/ (Accessed: 9 June 2021).

Martin, T. (2017b) *How to use third-party Actions on Google Home*, *cnet*. Available at: https://www.cnet.com/home/smart-home/how-to-use-third-party-actions-on-google-home/ (Accessed: 10 June 2021).

Maven - Benefits of using Maven (no date). Available at: http://people.apache.org/~jvanzyl/maven-3.1.1/benefits-of-using-maven.html (Accessed: 18 November 2020).

Melton, M. and Fenwick, J. (2019) 'Alexa Skill Voice Interface for the Moodle Learning Management System', *J. Comput. Sci. Coll.*, 35(4), pp. 26–35. Available at: http://www.ccsc.org/publications/journals/SE2019.pdf#page=26 (Accessed: 14 November 2020).

Miller, C. (2019) *'Voice Tech Report' predicts Apple will launch a 'SiriOS' in 2020*, *9to5Mac*. Available at: https://9to5mac.com/2019/07/20/sirios-launch-2020-report/ (Accessed: 22 December 2020).

Miller, F. P., Vandome, A. F. and McBrewster, J. (2010a) *Apache Maven*. Alpha Press. Available at: https://dl.acm.org/doi/book/10.5555/1942838 (Accessed: 12 November 2020).

Miller, F. P., Vandome, A. F. and McBrewster, J. (2010b) *Apache Maven*. Available at: https://dl.acm.org/doi/book/10.5555/1942838 (Accessed: 12 November 2020).

Mishra, A. and Mishra, D. (2013) 'Software Project Management Tools: A Brief Comparative View', *ACM SIGSOFT Software Engineering Notes*, 38(3), pp. 1–4. doi: 10.1145/2464526.2464537.

MQTT - The Standard for IoT Messaging (no date). Available at: https://mqtt.org/ (Accessed: 15 November 2020).

Node.js (no date). Available at: https://nodejs.org/en/ (Accessed: 2 May 2021).

Novabase (no date). Available at: https://www.novabase.pt/pt (Accessed: 21 December 2020).

Nunes, L., Santos, N. and Rito Silva, A. (2019) 'From a monolith to a microservices architecture: An approach based on transactional contexts', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag, pp. 37–52. doi: 10.1007/978-3-030-29983-5_3.

Purewal, S. J. and Cipriani, J. (2017) *The complete list of Siri commands* , *CNET*. Available at: https://www.cnet.com/how-to/the-complete-list-of-siri-commands/ (Accessed: 22 December 2020).

Rao, L. (2010) *Confirmed: Apple Buys Virtual Personal Assistant Startup Siri* , *TechCrunch*. Available at: https://techcrunch.com/2010/04/28/apple-buys-virtual-personal-assistant-startup-siri/ (Accessed: 22 December 2020).

Rehkopf, M. (2019) *What is a Kanban Board? | Atlassian*, *Atlassian Agile Coach*. Available at: https://www.atlassian.com/agile/kanban/boards (Accessed: 3 November 2020).

Rivas, M. A. and Souza, E. G. de (2014) 'Modelo Tradicional (Waterfall) de Desenvolvimento de Projetos e o Modelo Ágil (Agile) em Fábricas de Software', *Revista de Sistemas e Computação*, 4(1), pp. 3–11.

Rodrigues Martins, I. and Luís Zem, J. (2015) *ESTUDO DOS PROTOCOLOS DE COMUNICAÇÃO MQTT E COAP PARA APLICAÇÕES MACHINE-TO-MACHINE E INTERNET DAS COISAS 1 , 2*, *Revista Tecnológica da Fatec Americana*. Available at: https://fatecbr.websiteseguro.com/revista/index.php/RTecFatecAM/article/view/41 (Accessed: 15 November 2020).

Rotta, G., Charão, A. and Dantas, M. (2017) 'Um Estudo sobre Protocolos de Comunicação para

Ambientes de Internet das Coisas', in *Anais da Escola Regional de Alto Desempenho da Região Sul (ERAD-RS)*. Ijuí: SBC. Available at: https://sol.sbc.org.br/index.php/eradrs/article/view/2984 (Accessed: 15 November 2020).

Rouse, M. and Carty, D. (2018) *Amazon S3 bucket*. Available at: https://searchaws.techtarget.com/definition/AWS-bucket#:~:text=An Amazon S3 bucket is,data and its descriptive metadata. (Accessed: 21 November 2020).

Royce, D. W. W. (1970) 'Managing the Development of large Software Systems', *Ieee Wescon*, (August), pp. 1–9.

Salah, T. *et al.* (2017) 'The evolution of distributed systems towards microservices architecture', in *2016 11th International Conference for Internet Technology and Secured Transactions, ICITST 2016*. Institute of Electrical and Electronics Engineers Inc., pp. 318–325. doi: 10.1109/ICITST.2016.7856721.

Schonfeld, E. (2010) *Siri's IPhone App Puts A Personal Assistant In Your Pocket*, *TechCrunch*. Available at: https://techcrunch.com/2010/02/04/siri-iphone-personal-assistant/?guccounter=1&guce_referrer=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnLw&guce_referrer_sig=AQAAAB1tTT5QdzE8Udt6myLOSpF3Exgbcn2rwvDjazgDaD_T5Fm72zDWw9C4pOi2ggjZ5IG3b0kl1eF9Eo9uz7YkxVD2Te9Yxy2DVUxmmBzaVLsJ2FzhzBIVtjEBOP2p6_CgGjOwgAfxTp96qB520kjfEcbU33uCVw0BhsFrku7kjXvu (Accessed: 22 December 2020).

Schwaber, K. (1997) 'SCRUM Development Process', in *Business Object Design and Implementation*. Springer London, pp. 117–134. doi: 10.1007/978-1-4471-0947-1_11.

Schwaber, K. and Sutherland, J. (2011) 'The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game', *Scrum. org, October*.

Skill Events in Alexa Skills | Alexa Skills Kit (no date). Available at: https://developer.amazon.com/en-US/docs/alexa/smapi/skill-events-in-alexa-skills.html (Accessed: 28 November 2020).

Stalin, lenin (2019) *Benefits of Maven for java developers*. Available at: https://medium.com/@leninstalinesec/benefits-of-maven-for-java-developers-8083f9d33665 (Accessed: 17 November 2020).

Steele, B. (2017) *Google Assistant now helps with your shopping on Google Home (updated) | Engadget*. Available at: https://www.engadget.com/2017-02-16-google-home-shopping-with-google-assistant.html?guce_referrer=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnLw&guce_referrer_sig=AQAAAM1ezXYbEPoabRScB6JcoUnZJJlWYAf8W81ZI-LRxo3Y3G4ezqU8Lanjo7hYNWOhlXFlYKIjMGC83WRLZzanZkEsT7QBPbuyIB-Oe-mAfUOyNjAjv3Ab9ErtVBNSXOP2BAjMI-VOXTstAeA4yiPOCxuG3QS02runuAl4WDhO5S4F (Accessed: 9 June 2021).

Sumra, H. (2015) *Apple Announces New Apple TV With Siri, App Store, New User Interface and Remote* , *MacRumors*. Available at: https://www.macrumors.com/2015/09/09/apple-announces-fourth-gen-apple-tv/ (Accessed: 22 December 2020).

Sutherland, J., Ph, D. and Scrum, C. (2007) *The Scrum Papers : Nuts , Bolts , and Origins of an Agile Process*. Paris.

The Apache Software Foundation (2016) 'Maven – Welcome to Apache Maven', *maven.apache.org*. Available at: https://maven.apache.org/ (Accessed: 12 November 2020).

Tv Box - Vodafone Portugal (no date). Available at: https://www.vodafone.pt/pacotes/televisao/tv-box.html (Accessed: 2 May 2021).

Use Siri on all your Apple devices - Apple Support (no date). Available at: https://support.apple.com/en-us/HT204389 (Accessed: 22 December 2020).

Vodafone (no date). Available at: https://www.vodafone.com/ (Accessed: 21 December 2020).

Welch, C. (2017) *Amazon's Alexa can now recognize different voices and give personalized responses - The Verge*, *The Verge*. Available at: https://www.theverge.com/circuitbreaker/2017/10/11/16460120/amazon-echo-multi-user-voice-new-feature (Accessed: 31 May 2021).

Who We Are :: celfocus (no date). Available at: https://www.celfocus.com/home/who-we-are (Accessed: 3 November 2020).

Wortham, J. (2010) *Apple Buys Siri, a Voice-Command App*, *The New York Times*. Available at: https://www.nytimes.com/2010/04/29/technology/29apple.html (Accessed: 22 December 2020).

Zulqadar, A. (2019) *SDLC Waterfall Model: The 6 phases you need to know about*, *Rezaid*. Available at: https://rezaid.co.uk/sdlc-waterfall-model/ (Accessed: 3 November 2020).