INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Funcionalidades avançadas para pontos de venda móveis Caso de estudo aplicado ao produto MediaPOS

BRUNO MIGUEL DA SILVA ANDRADE
Outubro de 2016

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Funcionalidades avançadas para pontos de venda móveis

Caso de estudo aplicado ao produto *MediaPOS*

Bruno Miguel da Silva Andrade

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientador: Professor Doutor Paulo Manuel Baltarejo de Sousa

Júri:

Presidente:

[Nome do Presidente, Categoria, Escola]

Vogais:

[Nome do Vogal1, Categoria, Escola]

[Nome do Vogal2, Categoria, Escola] (até 4 vogais)

Porto, outubro 2016

Dedicatória

Dedico esta dissertação em memória do meu avô paterno (*in memoriam)*, José Gonçalves de Andrade, falecido no decorrer do processo da dissertação. Eterna saudade.

Resumo

No panorama económico nacional, o combate à fraude e evasão fiscal e aduaneira desempenham um papel importante na estabilização da economia. Vários esforços têm sido desenvolvidos ao longo dos últimos anos como forma de garantir uma justa repartição do esforço fiscal e sancionar sanções do incumprimento. Uma das medidas consiste na obrigatoriedade de utilização de sistemas de faturação certificados, assim como o programa *e-fatura*.

Os sistemas de pontos de venda desempenham na atualidade um papel importante na estratégia de combate ao incumprimento. Estes sistemas, devidamente certificados, permitem a emissão de faturas e posterior comunicação à Autoridade Tributária e Aduaneira.

O crescimento sustentado das tecnologias móveis e o aumento da acessibilidade a estes dispositivos criou várias oportunidades de negócio, conferindo não só mobilidade aos sistemas de pontos de venda, como também explorar várias funcionalidades.

O objetivo principal desta investigação foca-se no estudo e criação de uma solução para pontos de venda destinada a dispositivos móveis, recorrendo ao desenvolvimento de soluções multiplataforma. Para alcançar esse objetivo, foram realizados estudos para determinar as diversas características inovadoras a implementar. A pesquisa desenvolvida apresenta várias soluções e abordagens a nível de tecnologias de interação com dipositivos e periféricos, tirando partido das capacidades dos dispositivos móveis e conferindo novas possibilidades de automatização de processos.

Como prova de conceito e base para uma solução comercializável foi elaborado um protótipo. Foram apresentadas em detalhe as várias fases que constituíram o processo de desenvolvimento da solução, as ferramentas e técnicas abordadas, assim como a proposta de solução.

Por fim, é feita a discussão dos resultados obtidos e apresentadas as dificuldades, limitações e perspetivas de trabalho futuro.

Palavras-chave: Pontos de Venda, Aplicações Móveis, Soluções Multiplataforma, Funcionalidades Avançadas

Abstract

In the national economic panorama, the fight against fraud and tax and customs evasion play a vital part in the economy stabilization. Several efforts have been made throughout the last years as a way to guarantee ensure a fair distribution of the tax effort and penalize non-compliance penalties. Both the *e-fatura* program and the obligation of certified billing systems are some of these measures.

The point-of-sale systems (POS) play nowadays an important role in the strategy to fight non-compliances. These systems are properly certified by the Autoridade Tributária e Aduaneira and allow the issuance of invoices and subsequent reporting.

The sustained growth and ease of availability of mobile technology has opened several business opportunities, allowing to explore various features besides the ability to allow mobility in these point-of-sale systems.

The main purpose of this investigation focuses on the study, design and implementation of a point of sale solution using mobile devices, employing a cross-platform solution. To attain this goal, studies were performed to determinate various innovative features to be implemented. This study presents numerous solutions and approaches to the interaction with the device and its peripherals, taking advantage of the abilities of mobile devices and allowing new possibilities in process automation.

As proof of concept and basis to a commercial solution, a prototype was created. The various stages that formed the process of developing the solution, tools used and techniques addressed were presented with details, as the final solution proposal.

Finally, the obtained results, difficulties, limitations and perspectives for future work are discussed with some details.

Key-Words: Point of Sale, Mobile Application, Cross-Platform Solution, Advanced features

Agradecimentos

Gostaria de começar por agradecer, em primeiro lugar, ao meu orientador, Professor Doutor Paulo Manuel Baltarejo de Sousa, por todo o apoio prestado no decorrer desta dissertação, e, em particular, pelo conhecimento científico transmitido. Foram muito importantes os momentos em que me fez refletir com a colocação das perguntas certas de forma oportuna. Agradeço ainda, a sua disponibilidade e paciência demonstradas ao longo deste processo, sem esquecer o facto de me ter aceite como orientando numa fase tardia da dissertação.

Seguidamente, gostaria de agradecer ao Professor Doutor Nuno Alexandre Pinto da Silva, atual Diretor do Mestrado em Engenharia Informática do Instituto Superior de Engenharia do Porto e regente da unidade curricular "Tese / Dissertação / Estágio". O seu apoio ao longo da dissertação foi fundamental, através de palavras sábias e motivadoras, assim como a aceitação e aprovação da proposta de realização desta dissertação que teve início um pouco mais tarde.

À empresa *CimSoft* e em particular ao seu responsável, o Engenheiro José Morgado, pela oportunidade concedida e meios facultados, assim como a sua experiência pessoal no desenvolvimento de pontos de venda.

Aos meus amigos, em particular à Elisabete Alves, Ana Ferreira, Pedro Falcão, Anthony dos Santos, Inês Souto e Cristiana Valente. Sem eles esta dissertação não teria sido possível. É um processo longo e desgastante, em particular numa fase mais complicada da vida, pelo que o apoio e motivação facultados foram, sem dúvida, uma mais valia.

Por fim, e não menos importante, à minha família, em especial aos meus pais e avó, por todo o apoio e compreensão neste longo percurso académico. Foi um enorme investimento pessoal, em que mesmo nos piores momentos estiveram sempre presentes para me apoiar e motivar a prosseguir as minhas ambições. Foi um percurso académico que decorreu paralelamente com uma fase menos boa na vida pessoal e/ou familiar devido a um problema de saúde do meu Pai. O meu agradecimento especial para o meu Pai, que mesmo entre tratamentos e cirurgias não deixou de prestar o seu devido apoio!

A todos os anteriormente referidos e a todos os que possa não ter referido, que direta ou indiretamente contribuíram para esta dissertação, o meu sincero obrigado!

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

API	*Application Programming Interface*
HTML	*Hyper Text Markup Language*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*
IIS	*Internet Information Server*
IVA	Imposto sobre o Valor Acrescentado
JSON	*JavaScript Object Notation*
NFC	*Near Field Communication*
NIF	Número de Identificação Fiscal
OCR	*Optical Character Recognition*
PAN	*Personal Area Network*
PDA	*Personal Digital Assistant*
PHP	*Hypertext Preprocessor*
POS	*Point of Sale*
REST	*Representational State Transfer*
SAFT-PT	*Standard Audit File for Tax Purposes – Portuguese Version*
SGBD	Sistema Gestor de Bases de Dados
SOAP	*Simple Object Access Protocol*
XML	*EXtensible Markup Language*

1 Introdução

Este documento foi elaborado no âmbito da unidade curricular "Tese / Estágio / Dissertação", pertencente ao Mestrado em Engenharia Informática do Instituto Superior de Engenharia do Porto.

Neste primeiro capítulo é apresentado o contexto e enquadramento do problema, os principais objetivos e motivações, assim como uma descrição sumária do trabalho efetuado e a sua mais valia. Por fim, é apresentada a estrutura do documento.

1.1 Enquadramento

No panorama económico atual, o combate à fraude e evasão fiscal e aduaneira desempenham um papel importante na estabilização da economia. Vários esforços têm vindo a ser desenvolvidos nos últimos anos, como forma de garantir uma justa repartição do esforço fiscal e sancionar de uma forma mais efetiva as situações de incumprimento. Uma das medidas impostas consiste na obrigatoriedade da utilização de programas de faturação certificados, o que garante uma maior transparência no momento de pagamento de bens e serviços, nomeadamente no que diz respeito ao apuramento e liquidação de *IVA (*Imposto sobre Valor Acrescentado). O Governo de Portugal [1] publicou um relatório em 2014 que apontava Portugal no sexto lugar, da lista dos países com nível de evasão fiscal mais baixo, entre os vinte e seis países da União Europeia, tendo havido um decréscimo de evasão fiscal quando comparado com os anos anteriores.

O lançamento do programa *e-fatura*, que prevê que as empresas comuniquem mensalmente à Autoridade Tributária e Aduaneira todas as faturas emitidas, assim como a obrigatoriedade de emitir faturas em todas as operações, conduziu a um aumento de cerca de 12.3% em 2014, face ao ano anterior. Houve igualmente um incremento de 36.3% na totalidade de faturas emitidas com *NIF* (número de identificação fiscal) de pessoa singular no mesmo período de tempo [1].

A evolução dos sistemas de *POS (Point-of-Sale),* ou na tradução em português – ponto de venda, permitem, na atualidade, a emissão de faturas. Devido à obrigatoriedade da utilização de sistemas de faturação devidamente certificados e de acordo com a legislação em vigor, a adaptação dos sistemas de pontos de venda foi natural e lógica. Trata-se de uma junção vantajosa para alguns mercados, tais como a restauração, retalho, frutarias, minimercados, entre outros, visto que para além das funcionalidades comuns dos sistemas de pontos de venda, tais como a ligação a periféricos, é ainda possível gerir o negócio e emitir faturas.

Contudo, há mercados com requisitos diferentes, nomeadamente a necessidade de mobilidade. Os sistemas de pontos de venda comuns, devido às suas especificações e natureza, não o permitem. A necessidade de energia, as dimensões do *hardware*, a necessidade de ligação constante à *internet*, entre outros, são fatores limitativos. Contudo, com a evolução das tecnologias, é possível adquirir dispositivos móveis, tais como *smartphones* e *tablets*, cada vez mais capazes e a preços reduzidos. Ligações à *internet* cada vez mais rápidas, baterias com maior capacidade e consequente maior duração em uso, ligação a diversos periféricos através de vários protocolos de comunicação, fazem parte da atualidade dos dispositivos móveis.

Em 2014, um estudo [2] concluiu que mais de dois milhões de pessoas eram trabalhadores móveis em Portugal, o que reforça a necessidade de soluções de mobilidade de modo a aumentar a produtividade, melhorar a experiência e formas de relacionamento com os clientes, assim como inovar o negócio.

O trabalho proposto para esta dissertação foi desenvolvido em âmbito empresarial, no decorrer de vários meses, a tempo inteiro, nas instalações da *CimSoft* – Tecnologias de Informação, Lda*.* A *CimSoft* é uma pequena empresa que dispõe de um produto para pontos de venda, o *MediaPOS*, certificado pela Autoridade Tributária e Aduaneira, que permite faturação e gestão de pequenos negócios e ligação a diversos periféricos. Contudo, esta solução ainda não dispõe de uma vertente orientada para os dispositivos móveis. Como em qualquer negócio, quem não acompanha as tendências do mercado fica condenado à obsolescência, sendo que as tendências atuais apontam para a utilização de dispositivos móveis e computadores de reduzidas dimensões*.*

Nesse sentido, torna-se relevante e define-se como principal objetivo desta investigação, o estudo de funcionalidades avançadas para pontos de vendas móveis. Posteriormente foi ainda criado um protótipo como resultado deste estudo, utilizando algumas das funcionalidades avançadas estudadas, no produto *MediaPOS* orientado para dispositivos móveis – o *MediaPOS Mobile.*

É expectável que o valor do produto criado implique alguns sacrifícios, tais como custos de formação, assistência e aquisição. Contudo espera-se que os benefícios sejam suficientemente vantajosos, nomeadamente o reduzido investimento em material (face aos pontos de venda tradicionais), mais desempenho devido à mobilidade, maior satisfação dos clientes, assim como um negócio mais dinâmico e moderno. Espera-se com isto desenvolver uma relação a longo termo e fidelizar o máximo número de clientes, através da negociação integrativa.

1.2 Motivação

A revolução ocorrida nos últimos anos e a evolução dos sistemas operativos móveis, incluindo a massificação do uso dos mesmos, potenciou o aparecimento de um vasto leque de aplicações e periféricos. O auxílio que os *smartphones* ou *tablets* conferem no dia-a-dia aos utilizadores e às diversas áreas de negócio, permitiu aumentar a satisfação, acessibilidade, usabilidade e conforto. Deste modo, o aparecimento de várias áreas de negócio, capazes de explorar o potencial dos mesmos, foi algo natural.

Os factos anteriormente mencionados despertaram o interesse da *CimSoft*, até porque empresas concorrentes começaram a introduzir soluções semelhantes no mercado ou a adaptar-se de forma a introduzir num futuro próximo soluções inovadoras capazes de tirar partido das mais recentes tecnologias móveis. Nesse sentido, e como não dispõem de conhecimentos de sistemas operativos móveis e suas capacidades, decidiram propor um estudo de tecnologias avançadas para pontos de venda móveis. O resultado desse estudo culmina com a criação de um protótipo com vista a demonstrar algumas dessas funcionalidades relevantes, tendo em vista a posterior introdução de um produto no mercado. Contudo, e como tem vindo a ser comum na *CimSoft*, um dos grandes desafios seria perceber até que ponto é possível desenvolver uma solução utilizando um ambiente de desenvolvimento multiplataforma no desenvolvimento desta solução.

Considerando as motivações anteriormente referidas, o propósito da investigação para a realização da dissertação reside maioritariamente no desafio de aprender uma tecnologia nova e emergente, num cenário real e empresarial, sem nunca deixar de ter em conta as diversas regras e imposições da Autoridade Tributária e Aduaneira, no que diz respeito à introdução deste tipo de soluções no mercado. A inovação será atingida através da criação de um protótipo funcional, que irá servir de exemplo para o produto final, com vista a colmatar esta falha existente na empresa.

1.3 Principais Objetivos e Descrição Sumária do Trabalho Efetuado

O principal objetivo desta investigação consiste na criação de uma solução para dispositivos móveis, até ao momento inexistente, do produto *MediaPOS* criado pela empresa *CimSoft.* Um basto número de empresas que desenvolve *software* para pontos de venda e quiosques, a operar no mercado nacional, dispõe de soluções comerciais para dispositivos móveis. Nos últimos anos verificou-se uma grande evolução das tecnologias disponíveis nos dispositivos móveis, assim como o aparecimento de computadores de reduzidas dimensões (como por exemplo o *Raspberry Pi)* e custos. Ambos têm capacidade de operar como pontos de venda, fixos ou móveis, permitindo a conexão a diversos tipos de periféricos. Esta é claramente a tendência do mercado, sendo que as empresas que não acompanham as tendências acabam por perder a sua quota. A *CimSoft* ainda não possui uma solução viável e suficientemente

estável para implementar em dispositivos móveis, nem o conhecimento suficiente de sistemas operativos móveis, nem das capacidades atuais dos dispositivos móveis.

Como tal, esta investigação pretende igualmente estudar as boas práticas implementadas noutras soluções, assim como estudar e implementar funcionalidades avançadas que permitam a automatização de processos, tirando partido das capacidades dos dispositivos móveis, tais como *smartphones, tablets, smartwatches,* entre outros.

Para que seja possível construir uma solução que traga valor acrescentado aos utilizadores, foi necessário efetuar um levantamento do estado da arte das soluções de pontos de venda existentes no mercado, sendo que algumas delas dispõem de soluções para ambientes móveis. Por um lado, registaram-se as funcionalidades que se repetiam em todos os sistemas, sendo essas consideradas funcionalidades base e assumira-se como funcionalidades que os utilizadores esperam neste tipo de soluções. Por outro lado, identificaram-se as diferenças e lacunas existentes.

Retiradas as conclusões, foi possível perceber que as soluções estudadas estão limitadas pelas funcionalidades básicas das soluções previamente existentes, não aproveitando em pleno as capacidades dos dispositivos móveis. Algumas dessas possibilidades avançadas poderiam permitir melhorar a interação com pessoas e periféricos, automatizar processos, entre outros. Para a integração destas funcionalidades, foi também efetuado um levantamento do estado de arte de tecnologias relevantes, assim como o estudo de protocolos de comunicação com periféricos.

Por fim, após terminar o estudo das diversas funcionalidades, foi desenvolvida uma aplicação para os vários sistemas operativos móveis e implementadas algumas das funcionalidades avançadas previamente estudadas.

Finda a fase de desenvolvimento e implementação, foram efetuados testes ao sistema de modo a que as funcionalidades fossem testadas de acordo com os requisitos.

1.3.1 Objetivos Principais

De forma a simplificar a leitura, os objetivos principais desta dissertação são:

- Criação de uma solução para dispositivos móveis baseada no produto *MediaPOS;*
- Adicionar funcionalidades avançadas de interação com pessoas e equipamentos periféricos, tirando partido das capacidades dos dispositivos móveis da atualidade.

1.3.2 Objetivos Específicos

Como objetivos mais específicos, pretende-se:

- Estudar e utilizar uma solução de desenvolvimento multiplataforma, percebendo a sua adequação às necessidades existentes;
- Estudar os componentes dos dispositivos móveis e investigar novas possibilidades de automatização de processos tirando partido das capacidades dos mesmos;
- Estudar os protocolos utilizados na interação com dispositivos periféricos;
- Definir e implementar novas funcionalidades;

Pretende-se acima de tudo, com estes objetivos específicos, perceber o que é possível desenvolver, recorrendo a soluções multiplataforma capaz de ser executada nos três sistemas operativos mais utilizados: o *Android, Windows Phone* e *iOS.* Desta forma, para além de se reduzir o tempo gasto no desenvolvimento de três aplicações distintas, é ainda possível economizar recursos e a aprendizagem de três linguagens de programação diferentes.

1.4 Estrutura do Documento

Neste ponto é apresentada a estrutura do documento, permitindo uma visão geral do tema e de como será desenvolvido. Esta dissertação está organizada em cinco capítulos distintos de forma a tornar compreensível as diferentes fases do projeto.

- **Capítulo 1:** Este capítulo visa enquadrar o tema a ser desenvolvido, descrevendo o problema e referenciando os intervenientes e soluções concorrentes. É também feita uma descrição dos objetivos e motivações, assim como a descrição dos capítulos seguintes.

- **Capítulo 2:** Neste capítulo é aprofundado o problema, identificando os intervenientes, seguido da análise das abordagens já existentes (vulgo estado da arte). Para tal, é necessário realizar a análise de valor, explicar as restrições existentes e apresentar as devidas conclusões.

- **Capítulo 3:** No terceiro capítulo é apresentado uma lista detalhada com vários periféricos e tecnologias, abordando as possíveis vantagens da utilização de cada um deles. É também efetuado um estudo sobre os principais protocolos de comunicação com periféricos, uma análise à aplicação base (*MediaPOS)*, assim como os vários ambientes de desenvolvimento multiplataforma.

- **Capítulo 4:** Este capítulo apresenta o desenho da solução a ser desenvolvida e a sua implementação. São pormenorizadas as várias fases do projeto, recorrendo a esquemas, imagens, entre outros. É ainda feita uma apresentação dos resultados e testes.

- **Capítulo 5:** Por fim, são apresentadas as conclusões do trabalho desenvolvido, explicadas as limitações e eventuais problemas, assim como perspetivas de melhoria e trabalho futuro.

2 Estado da Arte

Este capítulo apresenta o estudo preliminar com vista a aprofundar os conhecimentos e competências para o desenvolvimento e fundamentação da ideia proposta.

Inicialmente definiu-se o problema e contexto de forma detalhada, identificando o conceito de negócio, os processos e intervenientes, assim como as restrições impostas pela *CimSoft.* Posteriormente será demonstrado o resultado da pesquisa e recolha de informação relativo às soluções e abordagens já existentes no mercado, assim como o estado da arte de tecnologias relevantes. As conclusões da pesquisa são posteriormente analisadas e apresentadas numa tabela, permitindo tirar as devidas conclusões.

Por fim, é feita uma análise de valor com vista a clarificar qual a mais valia que esta solução trás à empresa, definindo os benefícios e sacrifícios existentes.

2.1 Evolução dos Pontos de Venda

Os *Point-of-Sale*, também conhecidos pela sigla *POS* ou pela tradução Portuguesa – Ponto de Venda, não são um conceito novo. De facto, não se sabe com precisão, mas pensa-se que o primeiro sistema de pontos de venda, também conhecida como caixa registadora, apareceu no final da década de 1870. *James Ritty*, criador da primeira caixa registadora, tinha um negócio rentável, contudo problemático, visto que os seus funcionários desviavam parte do dinheiro para os seus próprios bolsos. Por esse motivo decidiu criar a primeira caixa registadora: *"Ritty's Incorruptible Cashier"* (a caixa registadora incorruptível de *Ritty*, representada na Figura 1). De forma a solucionar o problema, sempre que alguém abrisse a gaveta do dinheiro, a caixa registadora emitia um som, permitindo que as pessoas em redor da mesma tivessem conhecimento de tal [3].

Figura 1 – *Ritty's Incorruptible Cashier* [3]

A mecanização da transferência de dinheiro passou então a ser algo real e mais seguro. Rapidamente surgiram novas caixas registadoras e inovações, tais como as caixas registadoras mecânicas, o comum recibo, bem como o aparecimento do motor elétrico que permitiu transações mais rápidas e fáceis [3].

A história das caixas registadoras deu novamente um grande passo com o aparecimento dos microprocessadores. A *IBM (International Business Machines)* foi uma das primeiras fornecedoras de sistemas de pontos de venda com caixas registadoras eletrónicas, aproximadamente 100 anos depois da primeira caixa registadora. Usadas em ambiente comercial, dispunham de teclas extra (como se pode observar na Figura 2) com os principais artigos vendidos, assim como a possibilidade de imprimir relatórios das vendas [3].

Figura 2 – Caixa registadora eletrónica [3]

Posteriormente apareceram os computadores, bastante mais compactos e economicamente viáveis, o que potenciou a próxima grande evolução dos pontos de venda. Dispunham da capacidade de executar programas dotados de interface apelativa para o utilizador, com funcionalidades acrescidas, assim como a possibilidade de ligar periféricos, tais como impressoras, gavetas de moedas, leitores de códigos de barras, entre outros [3].

É precisamente nessa época, em 1991, que surge a *CimSoft* - Tecnologias de Informação *Lda*, uma microempresa a operar na área das tecnologias de informação. Começou, inicialmente, por produzir equipamentos multimédia, *software* e serviços, porém, ao longo do tempo, passou a focar-se apenas nos dois últimos. Dos seus produtos destaca-se o *MediaDoc* e o *MediaPOS,*

sendo que o primeiro pertencia inicialmente ao segundo, mas acabou por se tornar um produto independente, certificado pela Autoridade Tributária e Aduaneira de Portugal há três anos.

Com a introdução do produto *MediaPOS*, a *CimSoft* pretendia responder às necessidades existentes no mercado, aproveitando o melhor que a tecnologia tinha para oferecer, tendo-se mantido em atividade e a comercializar este produto até aos dias de hoje. Ao longo deste espaço temporal o paradigma mudou ligeiramente. O conceito *POS* deixou de ser associado unicamente ao contexto de pontos de venda, passando também a incluir o conceito de *Point-of-Service* (ponto de serviço), que compreende várias funções, tais como fazer pedidos. Ou seja, os *POS* deixaram de ser utilizados apenas para registar transações monetárias, passando a facultar serviços.

A Figura 3 exemplifica os quiosques utilizados hoje em dia em hospitais, que permitem evitar a formação de grandes filas para fazer um simples *check-in* (registo de entrada). Trata-se de um exemplo prático de um sistema *Point-of-Service.*

Figura 3 – Exemplo de um quiosque [4]

Atualmente, os sistemas mais comuns de pontos de venda são baseados em computadores com o sistema operativo *Windows* ou *Linux*. Não obstante, o aparecimento dos dispositivos móveis cada vez mais capazes, dotados de ligação rápida à *internet*, ecrãs sensíveis ao toque, capacidade de armazenamento e ligação a periféricos, entre outros, suscitou um interesse acrescido neste tipo de soluções. *Tablets* e *smartphones* estão a ditar a revolução na forma como os pontos de venda operam, visto terem preços bastante apelativos, permitindo investimentos reduzidos, bem como mobilidade, o que se apresenta como potencialmente interessante em determinadas áreas de negócio.

O produto *MediaPOS* ainda não dispõe de uma aplicação móvel estável e a empresa *CimSoft* pretende expandir-se a outros mercados e segmentos de mercado. Torna-se, deste modo, importante realizar um estudo de funcionalidades avançadas para pontos de venda móveis, que irá permitir criar um protótipo de uma aplicação móvel, baseada no produto *MediaPOS*. Este será provido de funcionalidades modernas e atuais, de modo a utilizar as capacidades dos dispositivos móveis, assim como outras tecnologias e periféricos. Existe interesse por parte dos diversos clientes numa versão que permita mobilidade e que é partilhado pela *CimSoft*, de forma a seguir as tendências de mercado e satisfazer os clientes.

2.2 Soluções de Pontos de Venda

Nesta secção serão apresentadas as principais soluções de pontos de venda existentes no mercado Português, organizadas com base na empresa que as criou, fazendo referência às suas características e funcionalidades que as distinguem.

Atendendo ao teor destas soluções, foi impossível testar grande parte delas de forma a permitir construir uma crítica devidamente fundamentada. Nesse sentido, grande parte da informação disponibilizada foi adquirida através da consulta das páginas da *Internet* dos criadores, dos revendedores e, em alguns casos através de contacto telefónico diretamente com a empresa que produziu essas soluções.

2.2.1 Grupo PIE

Fundado em 1995 com o objetivo de desenvolver soluções de gestão e controlo orientadas para o mercado do retalho, o Grupo PIE acredita que o maior custo de uma empresa é a falta de gestão [5]. Nesse sentido, tem vindo desde então a inovar e apresentar novos produtos no mercado nacional, tendo inclusive soluções líder de mercado e presença internacional.

Os seguintes pontos apresentam alguns números que traduzem o sucesso das suas soluções [5]:

- "*Utilizado diariamente em mais de 31,000 unidades de restauração e retalho;*
- *Gestão diária em mais de 100 instituições públicas;*
- *Instalado em 95% do mercado Português e Angolano;*
- *Suporte técnico 24 horas, 365 dias por ano;*
- *Versões desenvolvidas em 12 línguas;*
- *Instalado em 22 países;*
- *Gera mais de 10 milhões de tickets por dia;*
- *Um novo cliente a cada 2 horas.*"

Com base nos dados e informações anteriores pode-se concluir que se trata claramente de um grupo com bastante experiência e conhecimento dos mercados em que opera, tendo um alargado número de parcerias de longa data com empresas líder no panorama nacional. Os produtos seguintes fazem parte do leque de soluções que o grupo tem para oferecer no contexto dos pontos de venda.

2.2.1.1 WinREST FrontOffice

O *WinREST* é possivelmente a solução mais conhecida que existe no mercado nacional dos pontos de venda. Foi a primeira solução de pontos de venda que o Grupo PIE concebeu, sendo igualmente um dos primeiros *softwares* para ponto de venda de restauração da Europa com registo por ecrã tátil [5]. Trata-se de uma solução líder de mercado que consegue abranger um

alargado número de requisitos numa única solução, chegando deste modo aos mais diversos tipos de clientes e sistemas.

É uma solução bastante completa, com suporte multiplataforma, multiprocessador, multilíngue, escalável e com tecnologia de redundância de dados, assim como adaptabilidade às leis fiscais de 20 países, o que permite ao Grupo PIE a capacidade de competir a nível internacional [6]. É ainda possível utilizar periféricos, tais como impressoras, leitores de códigos de barra, gavetas de moedas, entre outros.

Das várias potencialidades da solução, destacam-se as seguintes características técnicas [6]:

- "*Sistemas operativos open-source e Windows;*
- *Capacidade de interação com outros sistemas;*
- *Multiprocessador e multiplataforma;*
- *Sistema de encriptação de dados;*
- *Sensores de temperatura;*
- *Firewall própria e redundância de dados.*"

E destacam-se igualmente as funcionalidades [7]:

- "*Controlo de movimentos de caixa;*
- *Controlo de entrega do pedido certo na mesa certa;*
- *Controlo de descontos e ofertas;*
- *Controlo de assiduidade dos funcionários;*
- *Facilidade de atribuição de responsabilidades pelos erros cometidos;*
- *Possibilidade de geração de relatórios por via remota;*
- *Geração de ficheiro SAFT-PT (Standard Audit File for Tax - Portugal, trata-se de um ficheiro normalizado em formato XML que permite exportar facilmente os registos contabilísticos, faturação, entre outros);*
- *Atualizações constantes e suporte técnico 24 horas por dia, 365 dias por ano*."

O *WinREST* tem ainda a vantagem de possuir suporte para o sistema operativo *Android*, assim como formas de pagamento modernas, tais como o *MB WAY* [8]*, MEO Wallet* [9] *e SEQR* [10], que permitem pagamentos através da *Internet* do smartphone (pagamentos móveis) [11].

2.2.1.2 PingWin

Com a experiência adquirida na construção do *WinREST*, uma solução orientada para pontos de venda em restauração, o Grupo PIE decidiu criar o *PingWin*, orientado para os pontos de venda de retalho. O *PingWin* faz parte de uma nova geração de *software,* apresentando um conceito inovador que se divide em duas partes, o core e as personalidades [12].

- O core é responsável pelo motor de negócio e é desenhado de forma a ser genérico e comum a todos os mercados. É igualmente responsável pela gestão dos periféricos,

apresentação da interface gráfica com o ecrã tátil, assim como o motor da base de dados [12].

- As personalidades representam cada interface criada para o *PingWin*, sendo desenvolvidas e adaptadas às necessidades de cada cliente e mercado, respondendo de forma eficiente às necessidades. Existem de origem personalidades para minimercado, cabeleireiro, pronto a vestir, talho, quiosque e papelaria, etc [12].

De forma a responder às necessidades de cada utilizador, esta solução foi desenvolvida em código aberto, possibilitando, deste modo, a adição de funcionalidades por parte da comunidade [13].

Segundo o Grupo PIE não existe concorrência direta ao *PingWin* como tecnologia, produto e solução, afirmando ainda que a concorrência segue na direção oposta: não existe uma solução capaz de ser desenvolvida cliente a cliente e financeiramente competitiva [14].

À semelhança das outras soluções disponíveis pelo Grupo PIE, o *PingWin* permite a utilização de diversos tipos de periféricos, tais como balanças, impressoras, leitores de código de barras, entre outros, assim como pagamentos móveis através de *SEQR, MEO Wallet* e *MB WAY* [12][11].

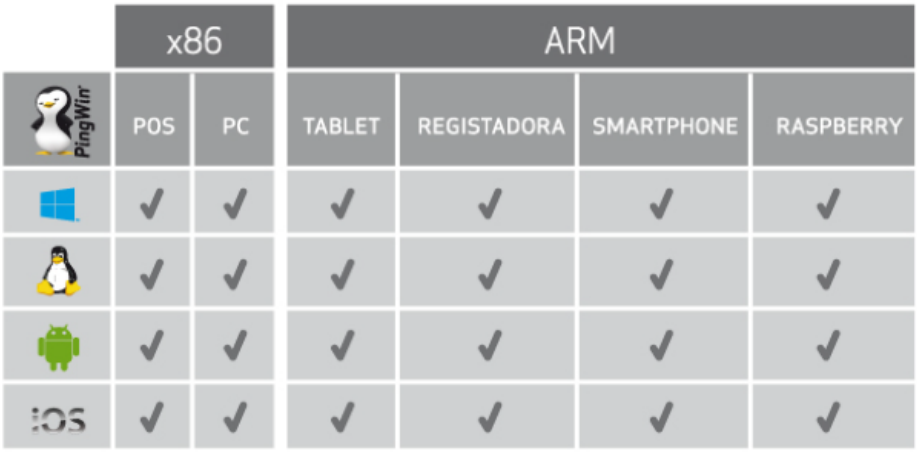

Figura 4 – Vários sistemas operativos e periféricos suportados [15]

Para além de ser uma solução disponível para vários sistemas operativos (Figura 4), destaca-se o suporte para o sistema operativo *Android* e *iOS.* Como se pode observar na Figura 5, é possível utilizar o *PingWin* num *tablet Android* como sistema de faturação, não necessitando de ligação permanente à *internet*. É ainda possível ligar, por exemplo, uma impressora *Bluetooth,* o que confere uma grande mobilidade e aplicabilidade.

Figura 5 – Tablet com o sistema *PingWin* e impressora *Bluetooth* [15]

2.2.2 Sage

Sage Portugal faz parte do Grupo *Sage*, fundado em 1981, e líder mundial no desenvolvimento de *software* de gestão para pequenas e médias empresas. Conta com 6 milhões de clientes espalhados por todo o mundo e encontra-se presente em vários países, fazendo questão de apresentar soluções utilizando a linguagem do mercado onde se encontram [16].

Em Portugal são representados por uma equipa de 170 colaboradores, cerca de 100 mil clientes e uma rede de 1500 parceiros. Têm vindo a crescer ao longo dos últimos anos, em parte através da aquisição de várias empresas com soluções para o mercado do retalho [16].

Das soluções disponíveis para pequenas e médias empresas destacam-se três devido às diferentes características que apresentam, permitindo a sua utilização em pontos de venda tradicionais e dispositivos móveis.

2.2.2.1 POS Standard

O *POS Standard* é uma solução para o mercado do retalho com vista a satisfazer as necessidades de vendas que exijam mobilidade. Onde quer que se esteja, é possível aceder aos dados de negócio, permitindo emitir faturas ou até mesmo aceder a informação de gestão [17].

Este produto esta ainda integrado com o núcleo do sistema (*back office*) de um outro produto do Grupo *Sage* chamado *Sage Retail*, permitindo assim fazer a gestão de *stocks* e das contas correntes dos fornecedores [17].

Das várias potencialidades da solução destacam-se as seguintes características técnicas [17]:

- Otimizado para terminais portáteis e processadores *ARM*, com ecrã tátil;
- Suporte para sistemas operativos *Windows, Linux* e *Android;*
- Pode ser usado em *POS* tradicionais*, PC, Tablets, PDA,* entre outros;
- Base de dados pode estar em local diferente da aplicação (servidor remoto ou *Web);*
- É possível integrar diversos dispositivos de leitura e recolha de dados tais como impressoras, leitores de código de barras e balanças, entre outros.

Destacam-se ainda as seguintes funcionalidades [18]:

- Emissão rápida e intuitiva dos documentos de venda;
- Emissão de guias de transporte em conformidade com a legislação;
- Análise de vendas e número de visitas por cliente, produtos mais vendidos, entre outros;
- Gestão de *stocks* e datas de validade;
- Acesso a relatórios e gráficos estatísticos.

Esta solução está certificada pela Autoridade Tribuária e Aduaneira e é atualizada periodicamente, de forma a cumprir as alterações legais e fiscais.

2.2.2.2 GesRestII

O *Sage GesRest II* é uma solução voltada para o mercado da restauração, bares e discotecas que permite uma melhor gestão dos espaços. Com o intuito de aumentar a qualidade dos serviços e com clientes cada vez mais exigentes, o Grupo *Sage* criou este produto que permite diferenciar e personalizar o atendimento do cliente através de ferramentas de fidelização que permitem gerar ações de promoções atrativas.[19]

Tal como a maioria dos *softwares* concorrentes, o *GesRestII* pode ser utilizado em *PC, POS* tradicionais, *Tablets, PDA (Personal Digital Assistant),* ementas eletrónicas, entre outros, sendo compatível com os sistemas operativos *Windows, Linux* e *Android* [19].

As principais funcionalidades e características do *Sage GesRestII* são [19][20]:

- Gestão integrada: permitindo comunicar entre várias unidades do restaurante e toda a gestão diária acessível num só equipamento;
- Definição de diferentes taxas de IVA por área do estabelecimento (o IVA do serviço de mesa pode ser diferente do IVA do *take away*);
- Permite definir diferentes preços por cliente ou grupo de clientes, produtos, dia ou hora da semana, aplicando descontos automáticos ou ofertas promocionais (como por exemplo “2 em 1”);
- Fluidez de operações, com menus dinâmicos e opções diferenciadas consoante os produtos do menu;
- Permite ligação ao *back office* da solução *Sage Retail*;
- Emissão e análise do ficheiro *SAFT-PT;*
- Produto certificado pela Autoridade Tributária e Aduaneira e de acordo com as alterações legais e fiscais.

Conclui-se que se trata de uma solução orientada para o cliente, estando atenta ao tipo de cliente e produtos consumidos. Deste modo, é possível premiar os clientes mais fiéis através de campanhas promocionais, concessão de crédito, entre outros. Existe uma forte componente de gestão operacional, o que permite simplificar e agilizar os processos.

2.2.2.3 Retail

Supermercados, lavandarias, lojas, talhos e perfumarias fazem parte da lista de alguns dos setores de atividade suportados pelo *Sage Retail*, uma solução para a área do comércio e retalho. À semelhança do produto anterior, existe uma forte componente de gestão que possibilita tornar o negócio mais rentável e organizado. É exequível consultar a lista de produtos mais rentáveis, consultar *stocks* e evitar as quebras dos mesmos, com resultados em tempo real [21].

Tal como seria expectável neste tipo de soluções, as obrigações fiscais estão asseguradas, permitindo emitir faturas e guias de transporte, ficheiro *SAFT-PT* e relatórios de impostos e contribuições com comunicação a aplicações de contabilidade. Trata-se de uma solução certificada pela Autoridade Tributária e Aduaneira [22].

A ligação a periféricos é suportada, como por exemplo, através de ecrãs táteis, teclados programáveis, leitores de código de barras, entre outros dispositivos. Os computadores pessoais e pontos de venda tradicionais com o sistema operativo *Windows* fazem parte dos requisitos técnicos da solução, sendo que não existe suporte para outros sistemas operativos, tal como *Linux* e *Android* [21].

Das várias funcionalidades existentes nesta solução, ganham especial relevância [21][22]:

- Preenchimento automático da identificação de clientes, com base no *NIF (*Número de Identificação Fiscal) através da *Internet;*
- Ofertas e promoções com base no tipo de cliente;
- Comunicação de campanhas e outras informações através de mensagem escrita enviada para o telemóvel;
- Personalização total do ambiente de trabalho;
- Possibilidade de funcionar num único estabelecimento ou numa rede de estabelecimentos;
- Interligação com sistemas de videovigilância;
- Organização e controlo do negócio através de relatórios e gráficos, apoio ao planeamento de encomendas, alerta de *stock*, entre outros;
- Controlo sobre a caixa de modo a evitar desvios de tesouraria.

No fundo trata-se de uma solução bastante completa, permitindo servir as várias unidades de uma empresa ou diferentes áreas de negócio. Facilidade de utilização, segurança e adaptação ao tipo de negócio são algumas das suas características.

2.2.3 Zone Soft

A *Zone Soft* pode ser considerada uma empresa jovem, uma vez que opera no mercado apenas há 10 anos, focando-se no desenvolvimento de *software* para a área de restauração e bebidas, comércio em geral e mobilidade [23].

Têm como objetivo proporcionar *software* fiável, de fácil instalação e execução e com o mínimo de manutenção, de forma a que os clientes possam usufruir do investimento sem dificuldades. A empresa afirma que o seu *software* foi desenvolvido para os clientes e pelos clientes, visto que estes fornecem as sugestões, conhecimento e críticas, de modo a tornar o produto rápido, simples e intuitivo, tentando assim corresponder às necessidades dos mais diversos tipos de utilizadores [23].

A *Zone Soft* é uma empresa que também investe em soluções para área de pontos de venda móveis, tendo sido pioneira no desenvolvimento do serviço de faturação com comunicação em tempo real com a Autoridade Tributária e Aduaneira [23].

2.2.3.1 ZS REST

O produto *ZS REST* foi desenvolvido para o mercado da restauração e bebidas, podendo ser utilizado por restaurantes, pastelarias, discotecas, *take-away*, entre outros [24].

Tentando-se manter a par das novidades, a *Zone Soft* incluiu os métodos de pagamentos mais recentes em todos os seus produtos, sendo que o *ZS REST* não foi exceção. Existe integração nativa com o serviço *MB WAY* e com as carteiras virtuais *MEO Wallet* e *SEQR* [24].

A ligação a periféricos está assegurada através da ligação a balanças, *PDA*, monitores, *tablets* e *smarthphones*, entre outros.

Do leque de funcionalidades realçam-se as seguintes [24]:

- *ZS BMS*: *back office* com gestão *online*, permitindo a gestão remota dos produtos, *stocks*, relatórios, emissão do *SAFT-PT* e guias de transporte, entre outros. Esta plataforma permite ainda o acesso centralizado a todos os produtos utilizados que tenham sido criados pela *Zone Soft;*
- *ZS CLOUD*: toda a informação gerada pelos diversos pontos de venda é sincronizada em tempo real e armazenada, sem custos extra, na nuvem (*cloud*), permitindo ter a informação sempre disponível, sem que haja risco de perda. Isto é particularmente útil tendo em vista o acesso a este tipo de dados, independentemente do local onde se encontre;
- *ZS REST APP*: a aplicação permite apresentar os produtos num *tablet Android*, proporcionando um serviço mais rápido e eficiente. Esta aplicação está totalmente integrada no *software ZS REST* e permite apresentar ementas digitais, registos de pedidos (exemplo: pedido dos clientes na mesa), questionários e apresentação da conta, podendo o utilizador efetuar o pagamento no próprio dispositivo móvel (*MB WAY*, *MEO Wallet* e *SEQR).*
- Comunicação em tempo real de faturas ou emissão mensal do ficheiro *SAFT-PT*.

O *ZS REST* encontra-se disponível para o sistema operativo *Windows* e a aplicação móvel (*ZS REST APP*) está disponível para o sistema operativo *Android* [24] .

As imposições legais da Autoridade Tributária e Aduaneira são cumpridas e o *software é* certificado pela mesma [24].

2.2.3.2 ZS POS MOBILE

O *ZS POS MOBILE* é uma solução que permite mobilidade no comércio, possibilitando a realização de pagamentos e emissão de faturas nos mais diversos locais, de modo simplificado e acessível. Trata-se de uma solução particularmente útil para clientes que prestam serviços sem estarem fixos num determinado local, como por exemplo serviços ao domicílio, táxis, venda ambulante, entre outros [25].

Não existe necessidade de possuir um computador, basta utilizar um *smartphone* ou um *tablet*, o que permite reduzir os custos. O sistema operativo *Android* é o único suportado até ao momento [25].

Através da geolocalização é possível associar os dados da venda a um determinado ponto geográfico. Esses dados são automaticamente sincronizados através da *Internet*. Contudo, é possível funcionar e emitir faturas sem ligação à mesma [25].

À semelhança das outras soluções da *Zone Soft* existe ligação ao *back office*, o que permite uma gestão *online* de toda a informação, *stocks*, emissão de relatórios, entre outros, assim como como se encontra de acordo com os requisitos legais impostos pela Autoridade Tributária e Aduaneira [25].

Para além de permitir pagamentos móveis através de tecnologias como *MB WAY*, *MEO Wallet* e *SEQR*, é ainda possível imprimir faturas em qualquer lugar através de impressoras *Bluetooth.* Também permite a ligação a balanças [25].

2.2.3.3 ZS POS

Disponível nas versões *basic, lite* e *pro,* a solução *ZS POS* é bastante semelhante a todas as outras disponibilizadas pela *Zone Soft.* A grande diferença está no mercado para o qual a mesma está indicada: o comércio de retalho [26].

Com o intuito de ajudar ao crescimento dos negócios, esta solução possui igualmente um *back office* centralizado que permite a gestão *online* das mais diversas variáveis do negócio, possui ligação a periféricos e às mais recentes tecnologias de pagamento, incluindo ligações a terminais multibanco. É ainda possível utilizar identificação biométrica para validação dos operadores [27].

Tal como todas as soluções concorrentes, o *ZS POS* é um *software* certificado que permite a emissão de guias de transporte e comunicação em tempo real ou mensal das faturas e ficheiro *SAFT-PT* [27].

Esta solução encontra-se disponível para os vários sistemas operativos *Windows* [27].

2.2.4 Primavera

Criada em Portugal, em 1993, a *Primavera Business Software Solutions* foi a primeira empresa a desenvolver soluções de gestão para o sistema operativo *Windows* [28].

Cerca de 40 mil empresas, o que representa um universo de mais de 150 mil utilizadores, recorrem diariamente às suas soluções para otimizarem os processos de negócio. Essas empresas estão espalhadas por 20 países, sendo que a *Primavera* é líder de mercado em alguns deles [28].

Do seu leque de soluções para as pequenas, médias e grandes organizações, destacam-se duas que permitem a utilização de pontos de venda no mercado do retalho e da restauração [28].

2.2.4.1 Tlim

Tendo em vista a automatização de operações, a solução para pontos de venda *Tlim* tornou as operações mais rápidas. Todo o negócio é simplificado e mais eficiente, permitindo a gestão de *stocks*, o controlo de funcionários, promoções e campanhas, entre outras funcionalidades [29].

Recorrendo a um ecrã tátil e a uma *interface* com o utilizador intuitiva, é possível qualquer operador executar as funções mais complexas sem grandes dificuldades. [29]

Salientam-se como principais características do *Tlim* [29]:

- Aplicação intuitiva, elevada usabilidade e *design* atrativo;
- Orientada para o utilizador, de forma a ser intuitiva e dispensar formação;
- Baixos requisitos de *hardware*, sendo compatível com *Linux* e *Windows;*
- Elevado nível de controlo de negócio;
- Certificado pela Autoridade Tributária e Aduaneira;
- Suporta diferentes idiomas;
- Facilidade de gestão de artigos e *stock*;
- Ambiente personalizável com base nas operações diárias;
- Suporte técnico com base numa rede de 500 parceiros, em diversos países.

À semelhança de outras aplicações, o *Tlim* permite exportar o ficheiro *SAFT* segundo as regras estabelecidas em Portugal, assim como a ligação a diversos dispositivos (balança, leitores de código de barras, entre outros) [29].

2.2.4.2 Pssst! e Pssst! Experience

A Pssst! é uma solução para pontos de venda semelhante a muitas outras, difere apenas no mercado para a qual é orientada: o mercado da restauração [30].

Os pedidos feitos pelos clientes são intuitivos e de gestão rápida através desta solução, sendo ainda possível definir flexibilidade nos preços, campanhas de fidelização, entre outros [30].

A ligação a periféricos está igualmente assegurada, tornando-se um produto útil em restaurantes *take-away*, sendo possível definir horas com preços diferenciados (*happy hour*) [30].

Par além do sistema de pontos de venda, existe ainda uma aplicação chamada Pssst! Experience. Menos orientada para o fornecedor e centrada no cliente, esta solução permite uma melhor comunicação entre os dois [31].

Esta aplicação encontra-se disponível gratuitamente para os sistemas operativos *Android* e *Ios* e permite consultar rapidamente num *smartphone* ou *tablet* toda a informação sobre um restaurante, efetuar reservas, fazer pedidos, permitindo que o cliente seja informado sobre o valor final [31].

A aplicação possibilita a consulta de um sistema de menu digital, criando uma experiência diferente e moderna. Para além das vantagens para o cliente, o fornecedor (por exemplo: um restaurante) tem também beneficia de [31]:

- *"Gestão de pedidos mais rápida e eficiente;*
- *Imagem sofisticada e moderna;*
- *Comunicação eficiente com os clientes;*
- *Maior rentabilização dos recursos;*
- *Redução de custos operacionais;*
- *Rápido retorno do investimento;*
- *Facilidade de atendimento de clientes em diversas línguas. "*

Ofertas personalizadas e campanhas apelativas são deste modo simplificadas. A aplicação pode ser utilizada pelos restaurantes que utilizem a solução *Primavera Pssst!*, com a particularidade de esta se adequar especificamente a cada um deles, com base na escolha do consumidor [31].

2.3 Análise Comparativa das Soluções de Pontos de Venda

As soluções anteriormente mencionadas foram comparadas com a finalidade de determinar, não só os pontos em comum, mas também aquilo em que diferem, de forma a estabelecer um conjunto de funcionalidades e características base.

Ao invés de se focar somente numa comparação parâmetro a parâmetro de forma a perceber qual a melhor solução, esta comparação tem como objetivo apurar os requisitos dados como adquiridos pelos utilizadores de sistemas de pontos de venda em geral, assim como num contexto móvel. Pretende-se ainda descobrir novos requisitos, de modo a adicionar funcionalidades avançadas ao sistema a desenvolver.

A Tabela 1 e a Tabela 2 responde às perguntas usadas para comparar as várias soluções, resumindo também as principais características que se destacam.

Tabela 1 — Comparação das soluções de pontos de venda analisadas

	WinREST FrontOffice	**PingWin**	**POS Standard**	**GesRestII**	**Retail**
Qual o mercado alvo?	Restauração	Vários, incluindo retalho e mobilidade	Retalho e mobilidade	Restauração	Retalho
Suporta o sistema operativo *Windows?*	Sim	Sim	Sim	Sim	Sim
Suporta o sistema operativo *Linux?*	Sim	Sim	Sim	Sim	Não
Suporta o sistema operativo *Android* e/ou *iOS?*	Sim, Android	Sim, ambos	Sim, Android	Sim, Android	Não
Existe suporte para mais do que uma língua?	Sim	Sim	Não	Não	Não
É possível utilizar periféricos?	Sim	Sim	Sim	Sim	Sim
A solução é certificada e de acordo com as leis da AT?	Sim	Sim	Sim	Sim	Sim
Emite o ficheiro *SAFT-PT?*	Sim	Sim	Sim	Sim	Sim
Possui funcionalidades de gestão e estatísticas?	Sim	Sim	Sim	Sim	Sim
Existe alguma funcionalidade extra?	Sim, possui sensores de temperatura	Sim, suporta o *Raspberry*	Sim, suporta o *Raspberry*	Não	Sim, publicidade via *SMS*
Possui uma aplicação para dispositivos móveis orientada ao cliente?	Sim, Android	Sim, ambos	Não	Não	Não
Permite pagamentos através dos dispositivos móveis?	Sim, MB WAY, MEO Wallet e SEQR	Sim, MB WAY, MEO Wallet e SEQR	Não	Não	Não

Tabela 2 — Continuação da comparação das soluções de pontos de venda analisadas

	ZS REST	**ZS POS MOBILE**	**ZS POS**	**Tlim**	**Pssst!**
Qual o mercado alvo?	Restauração	Mobilidade	Retalho	Retalho	Restauração
Suporta o sistema operativo *Windows?*	Sim	Não	Sim	Sim	Sim
Suporta o sistema operativo *Linux?*	Não	Não	Não	Sim	Sim
Suporta o sistema operativo *Android* e/ou *iOS?*	Sim, Android	Sim, Android	Sim, Android	Não	Não
Existe suporte para mais do que uma língua?	Não	Não	Não	Sim	Sim
É possível utilizar periféricos?	Sim	Sim	Sim	Sim	Sim
A solução é certificada e de acordo com as leis da AT?	Sim	Sim	Sim	Sim	Sim
Emite o ficheiro *SAFT-PT?*	Sim	Sim	Sim	Sim	Sim
Possui funcionalidades de gestão e estatísticas?	Sim	Sim	Sim	Sim	Sim
Existe alguma funcionalidade extra?	Não	Sim, Geolocalização	Não	Sim, uma versão gratuita (*Tlim Express*)	Sim, uma versão gratuita (*Pssst! Express*)
Possui uma aplicação para dispositivos móveis orientada ao cliente?	Sim, Android	Não	Não	Não	Sim, Android e iOS
Permite pagamentos através dos dispositivos móveis?	Sim, MB WAY, MEO Wallet e SEQR	Sim, MB WAY, MEO Wallet e SEQR	Sim, MB WAY, MEO Wallet e SEQR	Não	Não

Os parâmetros utilizados na criação das perguntas, com o intuito de serem usados na comparação, foram determinados com base numa análise transversal de todas as soluções previamente apresentadas. Nesse sentido foi possível extrair as características que definem cada solução individualmente, assim como compreender funcionalidades em comum entre todas as soluções. Assume-se como uma funcionalidade ou característica base toda aquela que se repete, uma vez que na perspetiva do utilizador é encarada como um requisito mínimo neste tipo de sistemas.

Embora as perguntas tenham uma designação suficientemente autoexplicativa, a linguagem utilizada pode estar fora do contexto do leitor, pelo que se torna necessário elucidar o que se pretende em cada uma das perguntas:

- “Qual o mercado alvo?”
  Refere-se ao mercado ou mercados que a solução em questão pretende satisfazer;
- “Suporta o sistema operativo *Windows*?”
  Indica se a solução pode ser instalada e executada num ponto de venda com o sistema operativo *Windows* instalado, sendo que requer licenciamento do mesmo;
- “Suporta o sistema operativo *Linux*?”
  Indica se a solução pode ser instalada e executada num ponto de venda que possua o sistema operativo de código aberto (*open-source) Linux*;
- “Suporta o sistema operativo *Android e/ou iOS*?”
  Indica se o sistema pode ser instalado e executado num ponto de venda que tenha o sistema operativo *Android* ou *iOS* instalado;
- “Existe suporte para mais do que uma língua?”
  Refere se a solução apresenta alternativas para além da língua Portuguesa;
- “É possível utilizar periféricos?”
  Informa se a solução permite utilizar periféricos, tais como balanças, leitores de códigos de barras, ecrãs que indicam o preço, entre outros;
- “A solução é certificada de acordo com as leis da AT?”
  Averigua se a solução foi devidamente certificada e cumpre com todas as imposições legais impostas pela Autoridade Tributária e Aduaneira (AT);
- “Emite o ficheiro *SAFT-PT*?”
  Refere se é possível criar e analisar o ficheiro *SAFT-PT* de modo a fazer a submissão do mesmo à Autoridade Tributária e Aduaneira (AT);
- “Possui funcionalidades de gestão e estatística?”
  Indica se a solução permite consultar *stocks*, movimentos de caixa, dados do cliente, apresentação de relatórios e outros dados estatísticos, entre outras funcionalidades;
- “Existe alguma funcionalidade extra?”
  Informa se existe alguma funcionalidade fora do comum, que não seja esperada neste tipo de soluções, ou que a distinga;
- “Possui uma aplicação para dispositivos móveis orientada para o cliente?”
  Refere se, independentemente de ter uma solução para dispositivos móveis do ponto de vista de quem fornece o serviço (exemplo: o restaurante), existe uma aplicação para para sistemas operativos móveis que os clientes possam usar nos seus próprios *smartphones* ou *tablets;*
- “Permite pagamentos através dos dispositivos móveis?”
  Averigua se é possível usar alguma tecnologia moderna que permita pagamentos através dos dispositivos móveis.

2.4 Análise de Valor

A análise de valor é uma metodologia que foi inicialmente utilizada na empresa *General Electric Co* no decorrer da segunda guerra mundial e como consequência da mesma, dada a falta de trabalho qualificado, matérias-primas e partes de componentes. *Lawrence Miles*, *Jerry Leftow*

e *Harry Erlicher*, trabalhadores da *General Electric Co*, tentaram encontrar alternativas à altura, tendo concluído que muitas delas, por vezes, permitiriam reduzir custos e melhorar o produto. Acidentalmente descobriram aquilo que hoje classificámos como análise de valor [32].

Uma proposta de valor é uma visão global do conjunto de produtos e serviços disponibilizados por uma empresa que tenham valor para o cliente. Assim sendo, uma proposta de valor bem definida permite definir uma estratégia para conquistar novos clientes. Para tal, é necessário perceber o significado de valor percebido pelos clientes.

O valor, no contexto da gestão e desenvolvimento de um novo produto, segundo Toledo e Machado [33], pode ser definido como a capacidade de fornecer aos clientes um produto que eles queiram, em tempo oportuno e por um preço que eles considerem justo. Ou seja, é a criação de um valor perante uma necessidade, sendo que esse valor deve ser aceite e considerado compensatório por parte dos clientes. Segundo [34],"a criação de valor é a chave para qualquer negócio, e qualquer atividade de negócio consiste na troca de um bem ou serviço, tangível e/ou intangível, sendo o seu valor aceite e compensatório pelos consumidores ou clientes, seja dentro da empresa ou rede colaborativa, ou fora da empresa" (tradução nossa).

Clientes distintos atribuem valores diferentes quando na presença do mesmo produto ou serviço. A isto chama-se valor percebido. De facto, o valor atribuído por quem fornece um produto ou serviço tem um peso ou significado diferente quando comparado com o valor percebido pelo cliente. Geralmente, o fornecedor do produto ou serviço é menos sensível ao preço, enquanto o cliente final é mais sensível à qualidade do produto.

2.4.1 Valor, Perspetiva Longitudinal de Calor e Cenários de Negócio

O valor do produto a ser criado para os clientes atuais e futuros clientes pode ser dividido em benefícios e sacrifícios.

Começando pelos sacrifícios, é de esperar que exista um preço/custo associado de aquisição, assim como custos de formação e assistência iniciais. Trata-se de um investimento quer a nível de *software*, quer a nível de *hardware*.

Contudo, é de esperar que os benefícios sejam mais vantajosos e superem os sacrifícios. Dos possíveis benefícios, espera-se: investimento reduzido face aos pontos de venda tradicionais (para novos clientes apenas), maior desempenho (devido à mobilidade), maior satisfação dos clientes e negócio mais dinâmico e moderno.

A perspetiva longitudinal de valor tem em vista analisar as variações do valor ao longo de um período de tempo. Inicialmente verifica-se um desejo / necessidade de desenvolver um produto, a que se associa a criação de expectativas acerca do que o produto pode providenciar. Contudo, nesta fase, o valor real ainda não esta completamente definido visto que não há experiência de utilização. Seguidamente, no ato de compra, pode ser necessário realizar para que se possa obter benefícios futuramente. Se se associar um preço ao benefício desejado, o valor em si tem uma perceção diferente. Haverá ainda um período de adaptação, no qual a perceção do valor

sofrerá alterações. Na fase pós-compra os benefícios tornam-se mais evidentes porque já existe experiência de utilização associada. Nesta fase espera-se que o negócio seja mais dinâmico e moderno, haja maior desempenho com a mobilidade que a solução confere, entre outros. Por fim, se o produto não corresponder às expectativas existe a fase de venda ou escoamento, que dá lugar a um valor de reembolso do produto.

Dos vários cenários de negócio possíveis, espera-se que exista uma cooperação entre o cliente e a empresa, de modo a conseguir obter o máximo possível de benefícios para ambas as partes (negociação *win-win)*. Trata-se de uma negociação integrativa com vista a desenvolver relações de longo termo e fidelizar o maior número de clientes, algo que tem vindo a ser uma politica da *CimSoft*. É expectável definir-se um mínimo/máximo aceitável de benefícios e sacrifícios e negociar em torno disso, de forma a que ambas as partes cooperem e possam obter o máximo possível de benefícios.

2.4.2 Ideia de negócio utilizando o Business Model Canvas

O *Business Model Canvas*, em Português, quadro de modelo de negócio, é uma ferramenta de gestão estratégica que permite desenvolver modelos de negócio novos ou previamente existentes. Trata-se de uma tabela dividida em 9 blocos, cada um deles com um significado específico, que nos permite uma melhor interpretação do modelo de negócio.

Na Figura 6 esta representado o *Business Model Canvas* do modelo de negócio existente na *CimSoft*, referente ao produto *MediaPOS* e a sua futura implementação.

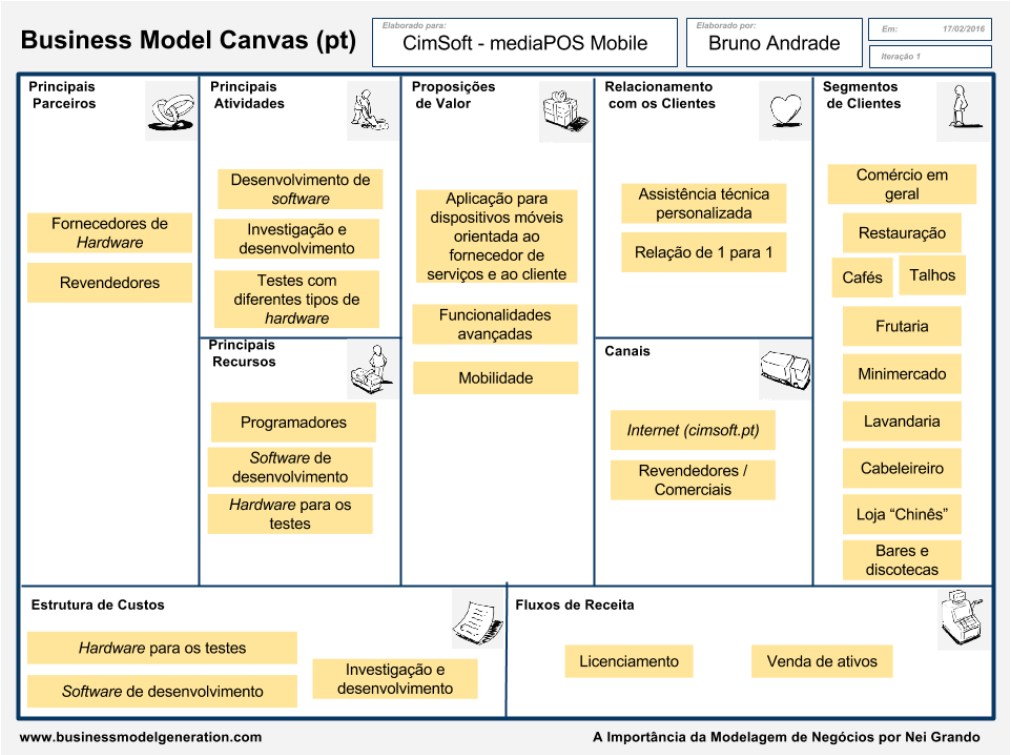

Figura 6 – *Business Model Canvas* [35]

Cada uma das nove divisões possui um significado específico. No caso do *MediaPOS*, pretende-se dotar o produto com funcionalidades avançadas que aproveitem o melhor que os ambientes móveis têm para oferecer. A empresa *CimSoft* ainda não possui uma solução para *smartphone* ou *tablets* suficientemente desenvolvida.

Nesse sentido, comecemos por analisar o primeiro quadrado, correspondente aos Segmentos de Clientes. É esperado neste campo que se identifique o público alvo, quem são os potenciais clientes e para quem se esta a criar o valor. No fundo, os clientes são o coração de qualquer empresa e é para eles que todo o trabalho é desenvolvido. Assim, a resposta a este primeiro ponto é:

- Comércio em geral;
- Restauração;
- Cafés;
- Talhos;
- Frutaria;
- Minimercados;
- Lavandarias;
- Cabeleireiros;
- Lojas do "Chinês";
- Bares e discotecas.

No fundo, trata-se do mercado para o qual o produto é desenvolvido. Seguidamente, é importante saber a proposta de valor, a mais valia que isso trás ao negócio, que problema pode ser resolvido, entre outros. Destacam-se os seguintes pontos:

- Mobilidade;
- Aplicação para dispositivos móveis orientada para o fornecedor de serviços e para o cliente;
- Funcionalidades avançadas.

O produto ainda não oferecia mobilidade, algo que é uma mais valia para alguns clientes. A possibilidade de adicionar ou verificar *stocks* junto do local onde os mesmos se encontram armazenados, as funcionalidades avançadas que a aplicação móvel pode oferecer, assim como uma aplicação que para além de servir o fornecedor de serviços (como por exemplo o dono de uma frutaria), possa também servir o cliente final é, sem dúvida, algo valorizável.

Uma vez definidos quais os nossos clientes ou segmento de mercado e o que temos para oferecer de valor, é necessário perceber quais serão as receitas que daí provêm, qual o valor que os clientes estão verdadeiramente dispostos a pagar e como cobrar aos nossos clientes. Qualquer negócio, para se tornar funcional, tem de gerar receita. O Fluxo de Receita provém de:

- Licenciamento;
- Venda de ativos.

Ou seja, espera-se obter receita através das licenças de utilização do *software*, assim como da venda de material para pontos de venda fixos e móveis.

Para obter receita, é necessário algum investimento prévio, o que implica custos. Na Estrutura de Custos é fundamental ter em conta quais os mais importantes neste modelo de negócio, quais são os recursos-chave de maior encargo monetário, quais os custos fixos e variáveis, entre outros. Destacam-se os seguintes encargos monetários:

- *Hardware* para os testes / desenvolvimento;
- *Software* de desenvolvimento;
- Investigação e desenvolvimento.

Os custos fixos correspondem ao pagamento de salários aos investigadores e programadores. Existe ainda gastos com o *software* necessário para desenvolver esta proposta de valor, assim como diverso *hardware* para a criação de protótipos e testes.

E quem apoia o negócio? Os Principais Parceiros. Estes ajudam a reduzir o risco e as incertezas, fornecem algum material, entre outros. Este negócio tem como principais parceiros:

- Fornecedores de *Hardware*;
- Revendedores.

É possível realizar parcerias com os fornecedores de *hardware* que permitem reduzir o risco, facilitar a aquisição de recursos e otimizar o produto através de uma relação próxima com os fornecedores. Por outro lado, os revendedores têm uma boa visão do mercado, estão em contacto direto com os clientes e sabem o que estes procuram, fornecendo assim algumas garantias.

Quais as principais atividades a serem desenvolvidas?

- Desenvolvimento de *software*;
- Investigação e desenvolvimento;
- Testes com diferentes tipos de *hardware.*

Espera-se que as Principais Atividades a serem desenvolvidas incluam o desenvolvimento da solução em si, a que se seguirá a fase de testes e criação de protótipos, recorrendo a diversos tipos de *hardware.* Contudo, para que isto seja possível é necessário fazer alguma investigação e planeamento prévios.

De forma a desenvolver as Principais Atividades são necessários recursos, que se dividem em: físicos, intelectuais, humanos e financeiros. Quais os recursos necessários para o desenvolvimento desta solução:

- Programadores;
- *Software* de desenvolvimento;
- *Hardware* para os testes.

Os programadores investigam e desenvolvem todo o sistema, sendo que para tal necessitam de *software* de desenvolvimento e, ainda, *hardware* para efetuarem os testes e experimentarem os protótipos.

E como os clientes são importantes para o negócio, é importante perceber qual o tipo de relacionamento que eles pretendem.

- Assistência técnica personalizada;
- Relação de 1 para 1.

Com este Relacionamento com os Clientes é esperado impulsionar as vendas, fidelizá-los e, eventualmente, conquistar novos. Nesse sentido, é necessário perceber quais os Canais preferenciais dos clientes para fazer chegar os produtos e serviços, campanhas, entre outros:

- *Internet (cimsoft.pt);*
- Revendedores / Comerciais.

Através da *Internet* é possível chegar a um maior número de clientes, permitindo inclusive a distribuição do produto (no caso de não necessitar de adquirir nenhum material físico). Os revendedores / comerciais desempenham igualmente um papel importante na estratégia de *marketing*, permitindo o fornecimento do produto e serviço, campanhas, entre outros.

2.4.3 Criação de valor

No processo de criação de valor é preciso analisar/modelar ou quantificar, pelo que se torna necessário recorrer a modelos e métodos.

A *Game Theory* (Teoria dos Jogos) é uma das técnicas adequadas à análise e criação de valor neste estudo. Através desta é possível analisar e tomar decisões lógicas num ambiente competitivo. Para tal, é importante conhecer e identificar os intervenientes, saber quais as preferências e necessidades, assim como as consequências de determinadas decisões. Nesse sentido, o fundamento base consiste em antecipar as reações dos intervenientes face às ações a serem executadas.

2.5 Sumário

Após a contextualização do panorama atual dos pontos de venda, descrevendo a problemática do objeto em estudo, conclui-se que o futuro passa pela utilização de pontos de venda móveis.

Definidos os intervenientes e as restrições, dá-se especial enfâse ao facto de se tratar de uma solução proprietária e a *CimSoft* não ter interesse em se tornar obsoleta, mas sim no melhoramento de forma a tentar acompanhar as tendências de mercado. Embora a *Cimsoft* não seja uma empresa com uma solução líder de mercado, é fundamental seguir as tendências do mercado atual e criar uma versão do produto *MediaPOS* para dispositivos móveis que, para além das funcionalidades já existentes na plataforma fixa, permita utilizar as tecnologias modernas e funcionalidades inerentes a estes dispositivos.

Nesse sentido, foi realizado um estudo do qual é possível tirar várias conclusões relacionadas com as soluções de pontos de venda existentes no mercado. Por um lado, salienta-se a crescente importância das aplicações de pontos de venda para dispositivos móveis, sendo que apenas três das dez soluções apresentadas, não possuem esta funcionalidade. Quatro das dez soluções possuem aplicação para dispositivos móveis orientada para o cliente (por exemplo uma ementa digital), sendo que duas dessas quatro apresentam suporte para o sistema operativo *Android* e *iOS.* Foi também possível concluir que a utilização de periféricos, funcionalidades de gestão e estatística, assim como certificação e emissão do ficheiro *SAFT* são considerados básicos para os clientes, visto que todas as soluções apresentam essas funcionalidades. O sistema operativo *Windows* parece ser o mais comum, ainda que tendencialmente se verifique o crescimento da utilização do sistema operativo *Linux*, assim como *Android.* Cinco das dez soluções apresentadas permitem pagamentos através de dispositivos móveis e tecnologias recentes, das quais se destaca o *MB WAY, MEO Wallet* e *SEQR.* No que diz respeito a funcionalidades extra é difícil encontrar um padrão, embora tendo em conta as tendências de mercado, pode destacar-se a utilização do produto *Raspberry Pi*. Devido ao preço e dimensões reduzidas deste produto, utilização de sistema operativo de código aberto (*open-source)*, assim como a capacidade de ligação direta a diversos tipos de periféricos, é sem dúvida uma tendência a ter em consideração.

Embora não exista muita informação sobre as aplicações móveis das soluções existentes, sabe-se que o intuito destas é apenas disponibilizar as funcionalidades previamente existentes nas soluções fixas, garantindo como extra, a mobilidade, assim como a possibilidade de utilizar soluções físicas mais económicas (um *tablet* pode ser mais acessível do que um computador/*POS* tradicional). Não existe grande inovação, havendo sim a restrição às funcionalidades básicas. Um *smartphone* ou *tablet* recente é capaz de oferecer muito mais. Por exemplo, as capacidades internas, tais como os sensores, câmara fotográfica, *NFC*, entre outras tecnologias, podem ser uma vantagem, tornando-se importante a realização de um estudo sobre as funcionalidades avançadas para pontos de venda móveis.

Por fim, foi feita uma análise de valor que demonstra a importância e valor da solução a ser desenvolvida, assim como foi analisada a ideia de negócio e representada através do *Business Model Canvas.*

3 Análise de Propostas

Este capítulo pretende apresentar ao leitor o produto *MediaPOS*, através de uma análise das suas funcionalidades e características principais. Será ainda abordado, de forma sucinta, o funcionamento, arquitetura e tecnologias utilizadas neste produto, que conta com largos anos de desenvolvimento.

Serão apresentadas algumas abordagens que permitem o desenvolvimento de funcionalidades avançadas de interação com dispositivos periféricos e utilizadores enaltecendo, assim, as suas características técnicas e possíveis aplicações no âmbito desta dissertação e elaboração de um protótipo. O mesmo se irá aplicar aos ambientes de desenvolvimento.

3.1 Análise da Solução Base

O propósito desta dissertação tem por base estudar e aplicar funcionalidades avançadas para dispositivos móveis, aplicadas a um caso de estudo concreto: o produto *MediaPOS.*

O *MediaPOS* é uma solução para pontos de venda, comercializada no mercado português pela empresa *CimSoft* – Tecnologias de Informação Lda, (uma microempresa criada em 1991, sediada no norte de Portugal).

Tal como alguns produtos existentes no mercado, o *MediaPOS* está disponível para várias áreas de negócio tais como: cafés, frutarias, minimercados, restauração, talhos, cabeleireiros, entre outros [36].

Trata-se de um produto desenhado a pensar em ambientes com ecrãs táteis, exigindo o mínimo de interações com o utilizador. Outras características do *MediaPOS* são [36]:

- Suporte para as diversas versões de *Windows* e *Linux;*
- Gestão de mesas e funcionários;

- Suporte para diversos periféricos: leitor de código de barras, gavetas de dinheiro, balanças, impressoras e ecrãs com informação;
- Suporte multilingue;
- Impressão de faturas e relatórios em modo texto ou gráfico, utilizando impressoras térmicas;
- Ajustável a diferentes resoluções de ecrã;
- Multiposto e multiutilizador;
- Gestão de stock, vendas, artigos, entre outros;
- Produto certificado pela Autoridade Tribuária e Aduaneira (licença nº 777), estando de acordo com todas as normas exigidas, incluindo a emissão de relatórios mensais *SAFT-PT.*

Uma das características que diferencia este produto é o preço. A empresa, como não se assume líder do mercado, opta por se manter em atividade através de licenças vitalícias ou anuais com um preço muito competitivo, quando comparado com outras soluções (líderes ou não de mercado). Para além disso, é possível tirar partido de *software* livre, utilizando bases de dados *PostgreSQL*, optando por um suporte reduzido e eliminando alguns extras de menor importância. Outra estratégia utilizada pela empresa consiste na poupança de recursos relacionados com *marketing*, recorrendo a comerciais pagos por comissões.

De um ponto de vista mais técnico, o *MediaPOS* é um *software* desenvolvido em *C#*, utilizando o *Visual Studio 2005* com cerca de 17 projetos e 1 370 514 linhas de código. De modo a ser possível executar em Linux, a solução utiliza o *Mono*, uma ferramenta grátis e de código fonte aberto que permite *compilar* aplicações em *C#* para sistemas operativos diferentes do *Windows.*

Trata-se de uma aplicação cliente com uma forte componente de base de dados, visto que tudo é armazenado no mesmo computador ou noutro diferente (no caso de ser multiposto), recorrendo à base de dados relacional *PostgreSQL.* O acesso ao servidor com a base de dados é feito de forma direta, através de *queries SQL* enviadas, recorrendo a uma ferramenta de código fonte aberto baseada em *ADO.NET*, chamada *Npgsql.* Esta ferramenta é particularmente interessante, pois pode ser utilizada em conjunto com o *Mono* em sistemas operativos diferentes do *Windows.*

A comunicação com dispositivos é feita principalmente através de portas de série e *USB*, definindo o *baud rate*, paridade, entre outros.

No que diz respeito à comunicação com a Autoridade Tributaria e Aduaneira, o ficheiro *SAFT-PT* é gerado, contudo é necessário fazer a submissão manual do mesmo, dado não existir comunicação por *Web Service.*

Não existe suporte para pagamentos através de carteiras digitais, nem existe uma aplicação móvel, tal como se pode observar na Tabela 3.

Tabela 3 — Resumo do produto *MediaPOS*

	MediaPOS
Qual o mercado alvo?	Restauração, Retalho, etc
Suporta o sistema operativo *Windows?*	Sim
Suporta o sistema operativo *Linux?*	Sim
Suporta o sistema operativo *Android* e/ou *iOS?*	Não
Existe suporte para mais do que uma língua?	Sim
É possível utilizar periféricos?	Sim
A solução é certificada e de acordo com as leis da AT?	Sim
Emite o ficheiro *SAFT-PT?*	Sim
Possui funcionalidades de gestão e estatísticas?	Sim
Existe alguma funcionalidade extra?	Não
Possui uma aplicação para dispositivos móveis orientada para o cliente?	Não
Permite pagamentos através dos dispositivos móveis?	Não

3.2 Abordagens Consideradas

Nesta secção, o principal objetivo é evidenciar possíveis soluções que possam fazer parte das funcionalidades avançadas para pontos de vendas móveis e, em concreto, para o produto/protótipo *MediaPOS Mobile.* Do leque disponível de abordagens a considerar foi necessário estabelecer uma divisão lógica, optando-se pela criação de um grupo relacionado com *Hardware*, um outro para as diversas tecnologias de comunicação e ainda um grupo de tecnologias relacionadas com pagamentos móveis. Deste modo, espera-se tornar a interação com dispositivos mais enriquecedora.

3.2.1 Hardware

A secção de *Hardware* pretende realçar possíveis soluções físicas ou tecnologias que permitam uma melhor interação com pessoas e dispositivos, evidenciando possíveis cenários nos quais podem ser aplicados, caso se venha a verificar tal necessidade.

3.2.1.1 Raspberry Pi

O *Raspberry Pi* é um computador de baixo custo, desenvolvido em Inglaterra. Embora existam várias versões comercializadas, incluindo algumas mais antigas, o destaque vai para a última versão, lançada em fevereiro de 2016: o *Raspberry Pi 3 Model B* [37].

Como se pode observar na figura 7, trata-se de um computador de reduzidas dimensões, que apresenta uma considerável capacidade de processamento e interfaces com dispositivos, assim como uma alargada lista de distribuições baseadas em *Linux,* disponíveis de forma gratuita.

Figura 7 – *Raspberry Pi 3 Model B* [38]

A terceira geração do *Raspberry Pi* apresenta as características evidenciadas na Tabela 4:

Tabela 4 — Resumo do produto *Raspberry Pi Model B* [38]

	Raspberry Pi Model B
Arquitetura	ARMv8 (64/32 bits)
SoC	Broadcom BCM2837
CPU	ARM Cortex-A53 64 bit quad core 1.2 GHz
GPU	Broadcom VideoCore IV
Memória	1GB partilhados
Portas USB	4 *USB* 2.0
Entrada Vídeo	15-pinos MIPI Camera Interface
Saída Vídeo	*HDMI* (ver 1.3)
Entrada Áudio	Sim, I^2S
Saída Áudio	Sim, 3.5mm conector de áudio
Armazenamento	Entrada MicroSDHC
Placa Rede	*Ethernet* 100 Mbit/s 802.11n *Wireless* Bluetooth 4.1 Bluetooth Low Energy (BLE)
Periféricos Baixo Nível	17 GPIO
Potência Máxima	800 *mA* (4.0 *W*)
Fonte Alimentação	5V MicroUSB ou GPIO
Tamanho e Peso	85.60 mm * 56.5 mm, 45 gr

Através das portas *USB* e *GPIO* é possível ligar uma série de dispositivos, tais como: teclado, rato, disco externo, câmara, entre outros.

Os principais pontos de interesse na utilização do *Raspberry Pi* neste estudo passam pela utilização do mesmo como servidor de base de dados, servidor *Web* (de modo a armazenar potenciais *Web Services*) e ligação a periféricos (tais como impressoras, balanças, entre outros). As suas reduzidas dimensões fazem com que seja possível colocar em qualquer lugar,

necessitando apenas de uma ficha para fornecer energia. As ligações sem fios ajudam nesse sentido.

3.2.1.2 Cabo OTG

Também conhecido como cabo *USB On-The-Go, USB OTG* ou simplesmente *OTG.* Trata-se de um cabo que permite que dispositivos móveis comuniquem com outros dispositivos, para além de um computador. De uma forma sintetizada, os dispositivos móveis assumem o papel de *host* (hospedeiro) ou *master* (mestre), permitindo assim que outros dispositivos *USB* se conectem, assumindo o papel de *slave (*escravo) ou dispositivo periférico [39]. Assim, é possível criar um canal de comunicação entre ambos os dispositivos.

Tal como se pode observar na figura 8, são constituídos, geralmente, por uma saída micro *USB* macho, que se liga ao dispositivo móvel, tendo na outra extremidade uma entrada *USB* fêmea, que permite a ligação de dispositivos tais como: impressoras, discos ou armazenamento externo, teclados, ratos ~~e~~ leitores de cartões.

Figura 8 – Cabo *OTG* [40]

Estes cabos são de fácil acesso e com um valor de venda considerado módico, sendo que com alguns conhecimentos de eletrónica é possível criar um, tal como é possível verificar na figura seguinte:

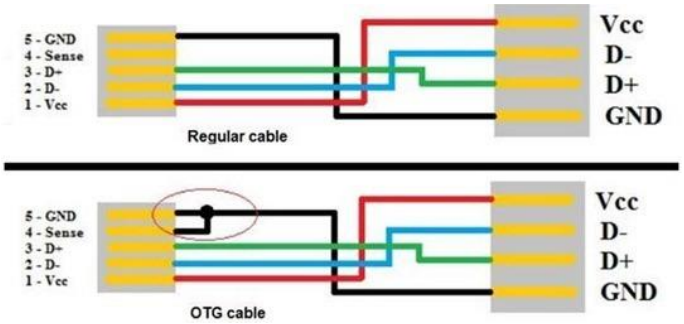

Figura 9 – Esquema de um cabo *OTG* [41]

Embora esta pareça ser uma boa solução para interagir com dispositivos, existem algumas desvantagens: funciona apenas para *Windows Mobile* e *Android*, nem todos os dispositivos móveis suportam esta funcionalidade, assim como apenas as versões mais recentes dos sistemas operativos móveis mencionados o permitem.

O *iOS*, instalado em dispositivos móveis como o *iPhone* e o *iPad*, utiliza ligações proprietárias em detrimento do padronizado *USB.* Os dispositivos mais antigos da *Apple* utilizam uma ligação larga com 30 pinos e os novos utilizam uma ligação mais fina designada de *Lightning.* Trata-se de dois tipos de ligações para os quais existem adaptadores criados pela própria *Apple* com o intuito de ligar dispositivos *USB.* Esses adaptadores chamam-se "*USB Camera Adapters"* (representados na Figura 10) e foram criados para interagirem com câmaras, leitores de cartões, entre outros, embora sejam bastante mais caros e limitados que um simples cabo *OTG*. Incompatibilidades, limitações de energia e lentidão são algumas das queixas apontadas pelos utilizadores na própria página web da Apple [42], relativamente às funções mais básicas deste adaptador.

Figura 10 – *Apple USB Camera Adapters (Lightning* e *30-Pin)* [42], [43]

O potencial interesse desta solução, no desenvolvimento desta dissertação, seria no âmbito da ligação a dispositivos. Contudo, não de trata de uma solução transversal aos três sistemas operativos móveis, havendo ainda uma variedade de limitações e problemas associados. Seria difícil e dispendioso testar e contornar os problemas, que surgem de origem na solução apresentada pela *Apple.* Uma solução com vários problemas de compatibilidade e limitações não fornece as garantias suficientes. No entanto, caso se tratasse de uma solução desenvolvida apenas para *Android* e *Windows Phone*, poderia ser uma solução viável a considerar.

3.2.1.3 Câmara Fotográfica

Atualmente os *smartphones* e *tablets* possuem câmaras fotográficas altamente eficientes, com uma elevada definição e estabilização de imagem. Câmara frontal e traseira, *flash*, estabilizadores de imagem, entre outros, são hoje uma realidade na maioria dos dispositivos.

Embora as câmaras fotográficas já existam nos telemóveis há mais de uma década, só começaram a ser verdadeiramente exploradas com o aparecimento de processadores mais potentes, sistemas operativos mais capazes, assim como placas gráficas embutidas que permitem processamento gráfico. É precisamente nesta altura que se introduz a visão computacional em ambientes móveis. A visão computacional consiste em realizar tarefas visuais em imagens ou vídeos captados. Essas tarefas podem ser a deteção de caras ou expressões, deteção de marcas e padrões, movimento de objetos no âmbito da segurança, realidade aumentada, reconhecimento de gestos, entre muitas outras possibilidades.

No âmbito desta dissertação e a pedido da *CimSoft¸* as câmaras fotográficas obtêm um papel de destaque através das possíveis aplicações, nomeadamente na captação de imagem de produtos, deteção e leitura de códigos de barras e/ou *QR Code*, identificação do utilizador através do reconhecimento de caras, reconhecimento de texto (*OCR – Optical Character Recognition),* entre outros.

3.2.1.4 Sensores

Os sensores são uma das principais características que transformaram os vulgares telemóveis em *smartphones.* Com a evolução da tecnologia tornaram-se pequenos, bastante mais rápidos e com uma precisão maior.

No fundo são pequenos dispositivos capazes de medir uma característica física, convertendo posteriormente esse sinal em um valor facilmente interpretável.

A oferta é grande, porém nem todos têm aplicabilidade num *smartphone* para o consumidor comum. Os mais comuns são [44]:

- Acelerómetro: medem as mudanças de velocidades que ocorrem em 2 ou 3 eixos, resultando o valor da aceleração;
- Giroscópio: sensor que permite medir a orientação;
- Proximidade: permitem detetar a presença de objetos próximos sem que estejas em contacto físico. Encontram-se geralmente colocados perto do auscultador de forma a que o ecrã se desligue quando uma chamada é atendida utilizando o *smartphone* perto do ouvido;
- Luminosidade: os sensores de luminosidade permitem medir o brilho da luz ambiente. Este tipo de dados permite que o *smartphone* ajuste o brilho do ecrã de forma automática;
- Magnetómetro: permite detetar campos magnéticos, tornando a criação de um compasso digital (bússola) possível;
- Barómetro: calcula a pressão atmosférica. Os dados resultantes determinam a altura do dispositivo face ao nível da água do mar, resultando num sinal *GPS* com maior precisão;
- Batimento cardíaco: existem várias maneiras de o calcular, sendo que os mais comuns em *smartphones* consistem numa luz *LED* e um sensor, capazes de calcular a frequência com que o sangue passa ou é bombeado no local da medição. Desse cálculo resulta o valor batimento cardíaco por minuto;
- Impressões digitais: permitem recolher os dados biométricos (impressões digitais). A sua segurança e eficácia depende do tipo de sensor: ótico, capacitivo ou ultrassónico. São bastante úteis para verificar a autenticidade do utilizador, em detrimento de palavras-chave ou cartões;
- *GPS:* calculam as coordenadas geográficas recorrendo à posição de leitura do utilizador face à posição de três ou mais satélites (triangulação).

Embora exista uma grande oferta, no contexto específico desta dissertação, apenas o sensor *GPS* e o de impressões digitais podem ter algum interesse, nomeadamente em localizar a posição da pessoa (no caso de uma venda no exterior, como por exemplo numa praia), assim como identificar o funcionário ou cliente através das impressões digitais.

3.2.2 Tecnologias de Comunicação

A secção das tecnologias ligadas à comunicação entre dispositivos e pessoas visa enaltecer potenciais utilizações no protótipo resultante desta dissertação. Existe uma grande diversidade de dispositivos, alguns com ligações físicas e outros com ligações sem fios, como o *Bluetooth* e entre outras opões existentes numa panóplia de tecnologias emergentes.

3.2.2.1 Wi-Fi

O *Wi-Fi* é uma rede local sem fios *(WLAN*) baseado no padrão 802.11 definido pelo *IEEE (Institute of Electrical and Electronics Engineers).* Trata-se de uma tecnologia de comunicação sem fios, que opera na banda 2.4 *GHz* e/ou 5*GHz,* e que pode alcançar uma distância de 250m no exterior, utilizando as antenas padrão. Dependendo da potencia da antena, do padrão escolhido e dos objetos ou interferência de sinais rádio, esta distância pode ser amplamente aumentada.

Devido à frequência em que opera, esta rede está sujeita a interferências de aparelhos micro-ondas (em especial os mais antigos e com pior isolamento), dispositivos *Bluetooth*, telefones sem fios, entre outros. Os canais de distribuição variam consoante a frequência utilizada e os países. No Japão estão disponíveis 14 canais, enquanto na Austrália e Europa estão disponíveis apenas 13. Já nos Estados Unidos da América são unicamente 11, utilizando a frequência dos 2.4 *GHz* [45]. Nesta frequência o sinal ocupa 5 canais, existindo alguns casos de sobreposição (figura 11). Por outro lado, a frequência dos 5*GHz* não tem sobreposição e oferece 23 canais.

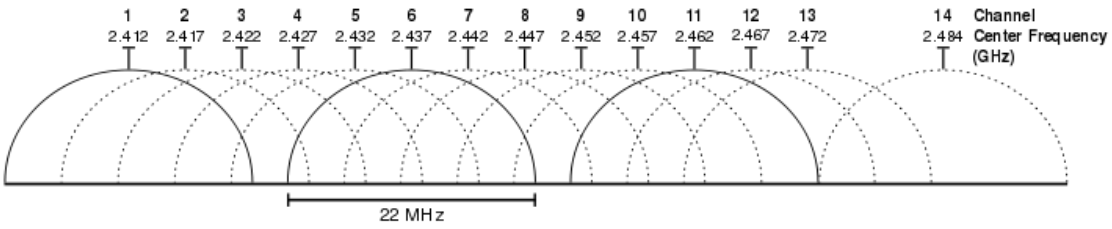

Figura 11 – Sobreposições na banda dos *2.4GHz* [46]

A velocidade da ligação depende do padrão escolhido [47]:

- *IEEE 802.11a:* Padrão para frequência 5 *GHz* com capacidade teórica de 2 *Mbps*;
- *IEEE 802.11b:* Padrão para frequência 2.4 *GHz* com capacidade teórica de 11 *Mbps*;

- *IEEE 802.11g:* Padrão para frequência 2.4 *GHz* com capacidade teórica de 54 *Mbps*;
- *IEEE 802.11n:* Padrão para frequência 2.4 *GHz* e/ou *5GHz* com capacidade teórica de 150 *Mbps* a 600 *Mbps*;
- *IEEE 802.11ac:* Padrão para frequência *5GHz* com capacidade teórica de até 780 *Mbit/s* a 160 *MHz*, podendo este valor aumentar com base no número de antenas;

Existem, contudo, mais padrões e o protocolo continua a evoluir, sendo esperado que atinja velocidades mais altas para uso comercial.

O *Wi-Fi* utiliza o protocolo *IP (Internet Protocol)*, de forma a conseguir trocar pacotes entre dispositivos. O principal objetivo deste protocolo é entregar os dados da origem num destino estipulado. Assim sendo, os protocolos *IPv4* ou *IPv6* possuem identificadores únicos (endereços de *IP)* e as mensagens podem ser enviadas para um destino (*unicast)*, todos os destinos (*broadcast)* ou para um ou mais conjuntos de destinos (*multicast).*

As redes sem fios estão naturalmente mais expostas e sofrem de problemas de segurança. Nesse sentido, foram criados diversos mecanismos que protegem as redes, permitindo encriptação e acesso controlado. Alguns dos principais mecanismos são: o *WEP, WPA* e o *WPA2*.

Esta é uma das tecnologias de interesse a ser utilizada no protótipo, visto permitir que dispositivos móveis se liguem à internet, a *Web Services*, comunicação na rede local sem fios (através de ligação direta a outros dispositivos ou servidores), assim como a possível interação com dispositivos preparados para comunicações sem fios por *Wi-Fi.* Hoje em dia é comum que dispositivos como impressoras, computadores fixos, entre outros, suportem esta tecnologia.

3.2.2.2 NFC

Near Field Communication [48], ou o acrónimo *NFC*, é a evolução da tecnologia de identificação por radiofrequência (*RFID)*. Trata-se de um conjunto de protocolos que permite que dois dispositivos eletrónicos estabeleçam uma comunicação rádio quando se tocam ou encontrem a uma curta distância. Esta tecnologia rádio de curto-alcance (até 10cm) opera na frequência 13.56 MHz e permite uma largura de banda até 424 Kbit/s [48].

Atualmente presente e suportada pelos diversos sistemas operativos móveis, a aplicabilidade desta tecnologia é imensa:

- Cartões bancários para pagamentos sem contacto direto;
- Bilhetes em geral;
- Pagamentos através de *smartphones*, recorrendo a serviços tipo *MEO Wallet;*
- Programar etiquetas (*tags)* de forma a automatizar vários processos (por exemplo enviar um e-mail automaticamente, automatizar processos em casa);
- Identificar artigos em lojas.

Estas são apenas algumas das opções, o que torna a sua aplicabilidade quase infinita.

As comunicações em *NFC* podem ser definidas em três modos [48]:

- Modo leitura/escrita: permite a um dispositivo *NFC* ler e/ou escrever em etiquetas ou outros formatos semelhantes da tecnologia *NFC;*
- Modo *Peer-2-Peer (P2P):* permite que os dois intervenientes troquem dados entre si;
- Modo emulação de cartão: trata-se de um modo seguro que permite que um dispositivo *NFC* ativo em modo cartão (*smart card),* seja lido por um outro dispositivo *NFC* dito normal. No fundo são dois dispositivos idênticos, sendo que um funciona como um cartão (por exemplo: para um pagamento).

A variedade de modos permite que a aplicabilidade do *NFC* seja frequente e fundamental em cenários do dia-a-dia, motivando cada vez mais as empresas a investir nesta tecnologia.

Uma dessas aplicações são as etiquetas. As etiquetas *NFC* são dispositivos passivos que operam sem uma fonte de energia própria. Quando em contacto com um dispositivo ativo (como um *smartphone* com suporte para *NFC)*, fornecem informações. De forma a fornecer energia às etiquetas, é utilizada indução eletromagnética, que cria uma corrente. Este processo é conseguido através de bobines do dispositivo ativo (*smarthphone*) que produzem ondas eletromagnéticas [49]. Por sua vez, estas ondas são novamente transformadas em corrente, por parte da bobine do dispositivo passivo (a etiqueta).

As etiquetas apresentam-se de várias formas e feitios. Existem vários tipos de etiquetas (que permitem uma maior ou menor capacidade de memória e velocidade de comunicação), e o seu preço varia. Todavia, trata-se de um produto acessível e com uma aplicabilidade/benefício que facilmente supera o custo. Na figura 12 seguinte podemos observar o aspeto de algumas etiquetas:

Figura 12 – Diferentes tipos de etiquetas *NFC* [50]

Embora não se trate de uma tecnologia segura, o facto de apenas permitir que exista comunicação a uma curta distância (ou idealmente em contacto direto) torna-se um beneficio face a outro tipo de comunicações com maior alcance, que se tornam mais suscetíveis a ataques.

Sendo uma tecnologia comum aos diversos sistemas operativos móveis, de baixo custo e elevado potencial, existe à partida um potencial interesse e aplicabilidade no protótipo desta

dissertação. Identificar funcionários, artigos ou locais, pagamentos sem fios, transmissão de dados a clientes, são algumas das possíveis implementações.

3.2.2.3 Bluetooth

O *Bluetooth* é uma tecnologia sem fios que permite a troca de dados a curtas distância e que tem como objetivo substituir os cabos que conectam diversos dispositivos eletrónicos. Este protocolo de comunicação gasta pouca energia e os *microchips* são de baixo custo, comunicando em frequências compreendidas entre 2400 *MHz* e os 2483.5 *MHz.* A maioria dos dispositivos disponíveis no mercado suportam o *Bluetooth* versão 4 (ou perfis da mesma), que permite velocidades de 25 *Mbit/s* e uma distância de 60.96 m. Porém, a versão 5 já se encontra disponível e permite 50 *Mbit/s* a 243.84 metros de distância.

Cada dispositivo possui um identificador único de 48 bits (*BD_ADDR*, ou seja, *Bluetooth Address* ou endereço *Bluetooth*) [51]. Parte desse endereço é atribuído pelo fabricante, sendo uma outra parte a identificação do fabricante [51]. No entanto, como o endereço não é visualmente apelativo, o mesmo é representado por um nome definido pelo utilizador.

Quando uma ligação é estabelecida, existe uma relação *Master-Slave*, sendo que o dispositivo responsável pelo início da comunicação e envio de pedidos é o *master* (mestre), enquanto que o outro dispositivo assume o papel de *slave* (escravo), limitando-se apenas a receber, reconhecer, tratar e enviar as respostas. Neste tipo de ligações o *master* pode-se ligar até 7 *slaves* formando uma *PAN (Personal area network,* rede de área pessoal) denominada de *piconet* [52]. Quando formadas em áreas sobrepostas, até um máximo de 10, denominam-se de *scatternet,* como se pode observar na Figura seguinte:

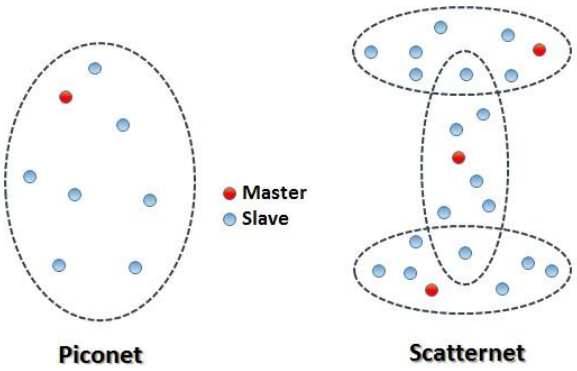

Figura 13 – Exemplo de uma *piconet* e *scatternet* [52]

Como em qualquer tecnologia de comunicação, existe uma série de regras e padrões às quais as mesmas obedecem, os protocolos. No caso do *Bluetooth* existe vários protocolos, sendo os mais importantes os protocolos núcleo ou protocolos de transportes, divididos nas seguintes camadas, como se pode verificar na figura 14 e de acordo com a fonte [53]:

- *RF* (*Radio frequency*): camada responsável por lidar com os aspetos ligados à radiofrequência;

- *Baseband*: camada que determina como os dispositivos localizam e comunicam com outros aparelhos *Bluetooth*, assumindo cada um o papel de *master* ou *slave;*
- *LMP* (*Link Manager Protocol*): responsável por lidar com parâmetros de autenticação, taxas de transferências de dados, criptografia, níveis de potência, entre outros;
- *HCI (Host Controller Interface):* esta camada disponibiliza uma interface de comunicação com o *hardware Bluetooth*, garantindo a interoperabilidade entre dispositivos diferentes;
- *L2CAP (Logical Link Control and Adaptation Protocol):* liga as camadas superiores e inferiores e garante a qualidade do serviço.

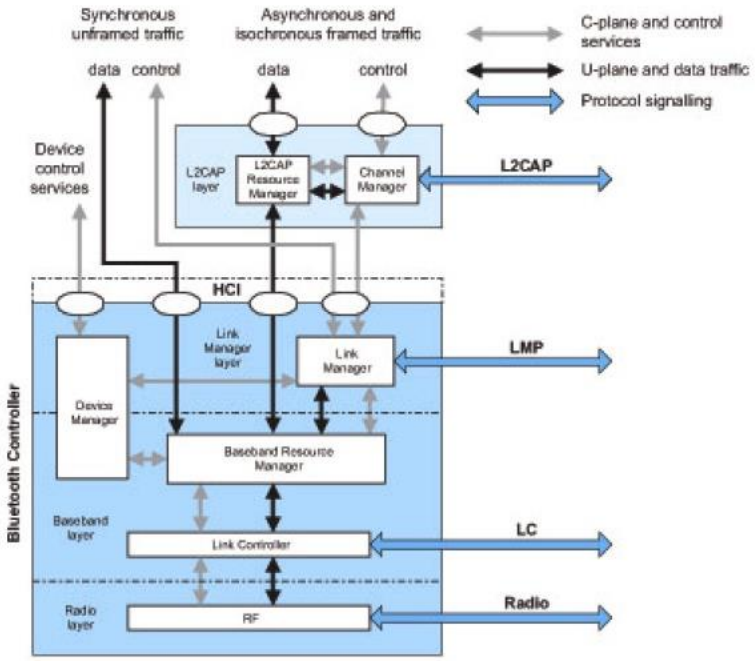

Figura 14 – Arquitetura *Bluetooth* [53]

O *Bluetooth* fornece alguma segurança através de autenticação, confidencialidade e chaves (*PIN).* Existem vários níveis de segurança, sendo que a escolha de um depende um pouco do ambiente e dos requisitos em questão.

As mais valias oferecidas pela utilização de *Bluetooth* no protótipo seriam, por exemplo, na utilização de impressoras *Bluetooth* (no caso de ser implementado um cenário totalmente móvel e sem acesso físico a energia ou impressoras fixas), e outros dispositivos periféricos que suportem esta tecnologia de comunicação. Um outro cenário possível seria na comunicação com dispositivos móveis "vestíveis" (*wearable)*, tais como pulseiras ou relógios, de forma a transmitir algumas mensagens ou chamar um funcionário.

A figura 15 demonstra uma impressora *Bluetooth* compatível com pontos de vendas móveis.

Figura 15 – Impressora *Bluetooth* [54]

3.2.2.4 Bluetooth Low Energie / Beacons

O *Bluetooth Low Energie*, também conhecido por *Bluetooth Smart*, *BLE* ou *Bluetooth LE,* é uma tecnologia de redes sem fios, que quando comparado com o tradicional *Bluetooth,* gasta bastante menos energia.

Integrado com o *Bluetooth 4.0*, o *Bluetooth Low Energie* (*e*mbora seja parecido com o *Bluetooth* tradicional e opere na frequência dos 2.4 *Ghz)* usa um sistema de modulação mais simples, utilizando um conjunto de canais diferentes. Em vez de utilizar 79 canais de 1 *MHz*, como o *Bluetooth* tradicional, o *BLE* utiliza 40 canais de 2 *MHz* [55][56].

Na prática as diferenças que isso proporciona face ao *Bluetooth* tradicional são [55]:

- Alcance superior a 100 metros;
- Taxa de 1 *Mbit/s*;
- Menor latência de ativação e comunicação;
- Não permite transmissão de áudio;
- Entre 0.01 e 0.5 W de gasto energético (face a 1 W do *Bluetooth* tradicional).

Das várias aplicações possíveis do *BLE* existe uma que se destaca devido à sua potencial aplicabilidade em pontos de venda móveis: o *Bluetooth Beacons.*

Bluetooth Beacons são pequenos dispositivos que transmitem periodicamente um tipo específico de mensagens através do *Bluetooth Low Energie.* Qualquer dispositivo com interface *Bluetooth 4.0* pode tornar-se num *Beacon* ou receber mensagens emitidas pelos mesmos.

Na prática estes dispositivos fazem *broadcast* para todos os dispositivos ao seu alcance, permitindo que os mesmos executem tarefas. Por exemplo, uma das aplicações desta tecnologia consiste no envio de um cupão de descontos quando o utilizador se encontra na presença do produto em questão. Outros exemplos são a navegação em interiores, monitorização da localização de empregados, publicidade, automação com base na localização, entre outras utilizações.

Embora o *Bluetooth Beacons* esteja bastante ligado à localização geográfica e a ações, esta tecnologia não permite saber a localização exata do utilizador, permitindo apenas saber se o utilizador esta perto ou longe de um ponto geográfico conhecido (neste caso, a localização onde o dispositivo *Beacons* foi instalado). Constata-se, ainda, que as ações despoletadas na presença de um dispositivo *Beacons* não acontecem sem que o utilizador tenha uma aplicação compatível instalada no dispositivo móvel. *Beacons*, por si só, não tem inteligência para realizar tarefas, na prática apenas enviam dados.

Apesar de ser uma tecnologia que em termos práticos se pareça com o *NFC*, as diferenças existentes são consideráveis, sendo a distância a mais significativa. O preço é também bastante diferente. Na figura seguinte é possível observar alguns dispositivos *Beacons.*

Figura 16 – Dispositivos *Bluetooth Beacons* [57]

Trata-se de uma tecnologia suportada pelos sistemas operativos móveis *Android, iOS* e *Windows Phone*, e disponível na maioria dos dispositivos móveis. Por estes motivos e os previamente descritos é, sem dúvida, uma tecnologia com potencial interesse de ser aplicada em pontos de vendas móveis. Mensagens publicitárias e cupões com base na localização do utilizador são algumas das principais aplicabilidades da tecnologia.

3.2.3 Pagamentos Móveis

Na secção das abordagens de pagamentos móveis pretende-se dar a conhecer algumas das soluções viáveis para utilização no mercado português, enaltecendo as principais características das mesmas.

Um estudo [58] realizado em 2015 indicou que, embora apenas 18% dos Norte Americanos inquiridos utilizem pagamentos móveis regularmente, 52% deles têm conhecimento de tais meios de pagamento e da sua disponibilidade, ainda que não os utilizem. Embora estes números não sejam extraordinários, um outro estudo [59] apontou que 1 em cada 5 utilizadores de *smartphones* realizariam pagamentos móveis no decorrer de 2016, nos Estados Unidos da América.

3.2.3.1 MB WAY

O *MB WAY* é um serviço de pagamentos através do telemóvel[8], criado pelo Grupo SIBS, que conta já com alguns serviços de elevado reconhecimento: o Multibanco e o *MB NET.* Este serviço associa cartões bancários a um número de telemóvel, permitindo efetuar pagamentos de compras ou transferências através dos mesmos. Recentemente o serviço passou também a permitir a criação de cartões virtuais (*MB NET)* através da própria aplicação do *MB WAY,* possibilitando assim efetuar pagamentos com cartões *American Express, MasterCard* ou *Visa.*

De modo a utilizar o serviço, o consumidor tem de se dirigir a uma caixa multibanco para efetuar a adesão, associando um número de telemóvel e um código de segurança. Posteriormente, o consumidor deve instalar a aplicação *MB WAY* para o sistema operativo do seu *smartphone*, estando a mesma disponível para *Windows Phone, Android* e *iOS.* Uma vez instalada a aplicação, é necessário introduzir os dados previamente criados e esperar por um código de ativação enviado por *SMS.*

A adesão a este serviço não tem custos, porém o banco pode taxar os valores normais associados a transferências ou ainda cobrar taxas extras por cada operação, de acordo com o contrato estabelecido. Nem todos os bancos aderiram ao serviço [60], sendo que no caso do banco do consumidor ter aderido, é obrigatório possuir um cartão multibanco e um número de telemóvel português.

Segundo a *SIBS* [8]*,* o *MB WAY* é simples, cómodo, rápido, grátis e ainda seguro, visto que o consumidor final não precisa de fornecer dados bancários.

Do ponto de vista do comerciante, este serviço pode ser integrado com os sistemas já existentes, sejam eles uma loja *online* ou um ponto de venda.

De forma a facilitar a integração do *MB WAY*, a *SIBS* criou uma página [61] com todos os detalhes técnicos que os programadores precisam de saber. O *MB WAY* é um *Web Service* que utiliza o protocolo *SOAP*, sendo que as transações são efetuadas por *HTTPS,* o que torna o serviço relativamente seguro. As mensagens obedecem à normal *XML,* podendo ser enviadas de forma assíncrona. Existe ainda códigos de retorno que permitem validar os resultados ou perceber mensagens de erro em questão. É possível testar a implementação do serviço através de um ambiente de teste (*sandbox)*, recorrendo a dados fictícios.

Das várias opções financeiras disponíveis destaca-se a opção de compra, devolução, anulação, assim como a consulta de operações financeiras.

De modo a poder usufruir do serviço, o comerciante tem de efetuar o registo e aceder às condições comerciais do contrato, efetuar a devida implementação anteriormente descrita e testar. Após este processo, essa mesma implementação será avaliada e certificada pela própria *SIBS*, para que possa usufruir do serviço.

O serviço *MB WAY*, no âmbito desta dissertação, poderá ser uma mais valia. Embora a sua implementação pareça extensa e algo complexa, o resultado final traria benefícios para os clientes, permitindo uma nova forma de pagamento.

3.2.3.2 MEO Wallet

O MEO Wallet é uma carteira virtual que permite efetuar pagamentos através de dispositivos móveis[9].

O consumidor tem ao seu dispor uma aplicação para *Android* ou *iOS,* que permite fazer a gestão, bem como pagamentos com o *MEO Wallet.* De forma a usufruir do serviço, o utilizador deve efetuar a adesão ao serviço, fornecendo alguns dados como o *NIF* (Número de Identificação Fiscal), número do Bilhete de Identidade ou Cartão de Cidadão, nome, morada, número de telemóvel, entre outros. Seguidamente, o consumidor deve instalar a aplicação para o sistema operativo móvel pretendido (opção facultativa, visto ser possível efetuar pagamentos por *SMS*). De forma a poder usar a aplicação para efetuar pagamentos, o consumidor pode carregar através de transferência bancária, por multibanco ou até mesmo associar um cartão bancário.

A aplicação necessita de um endereço de *e-mail*, palavra-chave e ainda de um *PIN* para validar o utilizador e, assim, o permitir efetuar pagamentos (obtendo os dados por *QR Code, NFC*, entre outros), consultar movimentos, efetuar transferências, gerir cartões, etc. Todo o serviço é gratuito para o consumidor, nomeadamente o pagamento e transferência de saldo, havendo apenas uma pequena taxa de 1% do valor no caso de carregamentos efetuados de forma imediata (por cartão bancário ou multibanco).

Por outro lado, o comerciante lucra com um serviço moderno e dinâmico. Através da conta de comerciante é possível receber pagamentos por: cartão *Visa* ou *MasterCard*, saldo *MEO Wallet* do cliente e referência multibanco. À semelhança da conta de cliente, é necessário fornecer alguns dados para a criação de conta, nomeadamente: os dados da empresa (nome, *NIF*, morada, contactos, etc), a informação pública e os dados do representante da mesma.

De forma a poder receber pagamentos é necessário instalar a aplicação *MEO Wallet,* que permite identificar a transação por *QR Code, SMS, NFC* ou até mesmo dados de cartão bancário. Para além de permitir consultar os movimentos, efetuar transferências e levantamento de fundos é ainda possível utilizar a aplicação em modo caixa, sendo apenas necessário introduzir o valor a faturar, como se pode ver na figura 17.

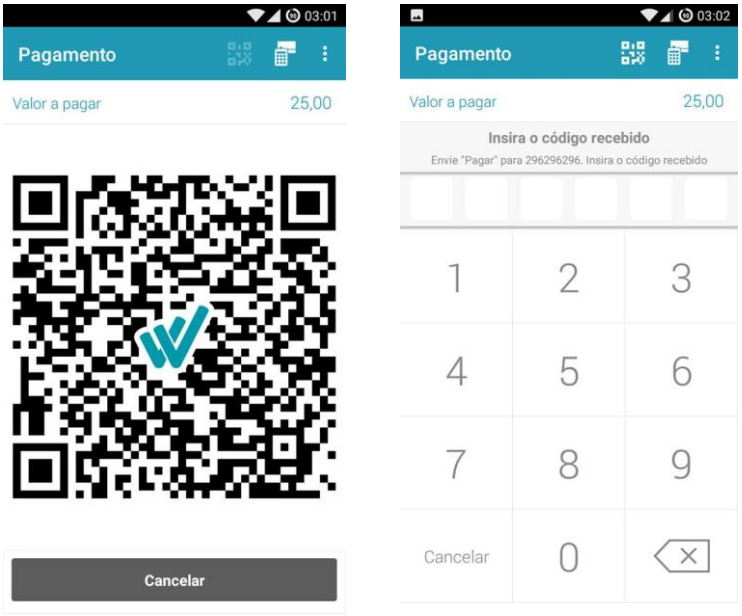

Figura 17 – Ecrã de emissão de pagamento *MEO Wallet*

Para além da aplicação, é possível ter um terminal de pagamentos automáticos (*TPA)* portátil com ligação *Bluetooth*, totalmente funcional com a aplicação do *MEO Wallet:*

Figura 18 – Terminal pagamentos automáticos *MEO Wallet* [62]

Os movimentos ou pagamentos recebidos pelo comerciante estão sujeitos a uma taxa aplicada por parte do *MEO Wallet* e pode ser consultada na sua página *web* [63].

De um ponto de vista mais técnico, o *MEO Wallet* tem por base um *Web Service REST* baseado em comunicações *HTTPS*, sendo os dados enviados e recebidos representados em *JSON* [64]. O comerciante recebe uma chave única que o permite identificar em todos os pedidos realizados pelo mesmo. À semelhança de outros serviços, é possível criar pedidos de pagamento, consultar operações, efetuar devoluções, entre outros, de forma assíncrona. Estes recursos da *API* encontram-se devidamente identificados e documentados [65], assim como a estrutura dos dados e das mensagens de erro.

De forma a evitar testar com dados e dinheiro real, o *MEO Wallet* providencia um ambiente de teste (*sandbox*) [66], onde é possível utilizar cartões fictícios fornecidos pelos mesmos, assim como existem aplicações oficiais para os dispositivos móveis com o único intuito de testar as implementações.

Esta solução tem elevado interesse para uma possível implementação no protótipo a ser desenvolvido, tendo em conta que oferece vários métodos de pagamento num único serviço, bem como um ambiente de teste simples.

3.2.3.3 SEQR

O *SEQR* (lê-se como *se.cure)* é um serviço de carteira móvel criado na Suécia, em 2001, pela empresa *Seamless*. Embora não se encontre tão difundido em Portugal quando comparado com os restantes países, o *SEQR* processa 3.1 mil milhões de transações por ano, através de 525 mil pontos de venda distribuídos por mais de 30 países [67].

À semelhança das soluções anteriormente abordadas, este permite que qualquer pessoa que disponha de um *smartphone* efetue pagamentos, transferências de dinheiro, etc. Através da aplicação móvel é possível efetuar a leitura de um *QR Code* ou utilizar a tecnologia *NFC* de forma a obter os dados de pagamento, sendo apenas necessário a introdução de um *PIN* para validar. A aplicação móvel encontra-se disponível para *Android*, *iOS* e *Windows Phone* (não em português, pelo que o ideal será esperar por uma solução adequada a Portugal).

A grande diferença do *SEQR* é que, internamente, apenas é usado o *IBAN* do consumidor de forma a efetuar débitos diretos dos valores. Os débitos são emitidos através da rede *SEPA*, sendo efetivamente debitados dois dias após ocorrer a transação.

De um ponto de vista mais técnico, a página desenvolvida [68] para programadores oferece diferentes modos de integração do serviço:

- Aplicação: permite efetuar pagamentos diretamente na aplicação criada pelo comerciante, podendo esta utilizar *Web Services* ou lançar a aplicação oficial *SEQR,* de modo a que o cliente possa efetuar o pagamento;
- *eProducts* e *Webshop*: *API* orientada para pagamentos disponibilizados em páginas *web;*
- Pontos de venda: funcionalidades para pontos de venda físicos e individualmente identificados, permitindo pagamentos *SEQR;*
- Serviços: pagamentos realizados através de pedidos *HTTP* a *Web Services* (*SOAP);*
- *SEQR instante checkout / invoice service:* permite realizar pagamentos através de pedidos e respostas *HTTP* a um *Web Service* (*SOAP* e *REST*). A grande diferença desta *API* consiste no facto de ser gerado um *QR Code* de pagamento, de forma instantânea, sem a necessidade de intervenientes (comerciante). O consumidor apenas necessita ler o código (*QR Code)*, introduzir o *PIN* e esta feito o pagamento.

Existe ainda uma aplicação criada para ambientes de testes evitando, assim, movimentação de dinheiro real.

Trata-se de uma solução igualmente interessante com vista a implementar no protótipo final, tendo em consideração a pequena desvantagem de apenas recorrer a débitos diretos, que podem demorar entre 2 a 5 dias a serem processado.

3.3 Ambiente de Desenvolvimento

Os dispositivos móveis e a sua crescente evolução desempenham, cada vez mais, um papel fundamental na sociedade. Segundo a *Google* [69]:

- 68% dos utilizadores de *smartphones* usam-no passados 15 minutos após acordar;
- 30% dos utilizadores admitem ficar ansiosos quando não têm o *smartphone* com eles;
- 87% dos utilizadores têm o *smartphone* por perto durante o dia e a noite;
- Consultam-nos em média 150 vezes por dia e despendem 177 minutos a utilizar *smartphones;*
- 82% dos inquiridos admite consultar os *smartphones* para verificar informações sobre uma compra que estão prestes a fazer, enquanto se encontram fisicamente na loja;
- 69% dos utilizadores tendem a comprar algo em empresas cujas aplicações móveis ou páginas *web* sejam práticos e facilitem o acesso à informação que procuram.

A facilidade que os mesmos conferem, em especial na organização e definição das tarefas, permite um aumento de produtividade e qualidade nas mais diversas tarefas desempenhadas.

A par das grandes evoluções a nível dos mais diversos sistemas operativos móveis, está uma grande evolução das características físicas dos *smartphones*, *tablets*, etc. Cada vez mais estes dispositivos são dotados de tecnologia mais recente e capaz, tais como: uma diversidade de sensores, periféricos e tecnologias de comunicação, baterias com elevada autonomia, ecrãs de diversas dimensões com uma resolução interna ao nível de um computador. Atualmente este tipo de tecnologias e sistemas operativos estão ainda disponíveis em outros meios, como os televisores, relógios, automóveis, entre outros.

Nesse sentido, os sistemas operativos móveis vêem-se obrigados a uma constante adaptação e evolução, de modo a tirar partido destes dispositivos. Segundo um estudo [70] elaborado pela *IDC (International Data Corporation)*, concluiu-se que no segundo trimestre de 2016, o sistema operativo móvel *Android* ocupava a primeira posição do mercado com uma quota de utilização de 87.6%. Em segundo lugar, com uma percentagem de utilização bastante inferior, encontra-se o *iOS* com 11.7%, seguido pelo *Windows Phone* com 0.4% do mercado. Os restantes 0.3% correspondem a todos os outros sistemas operativos móveis.

Tendo em conta os três principais sistemas operativos móveis previamente descritos, assim como o crescimento dos mercados dos dispositivos móveis, torna-se necessário ser capaz de

desenvolver soluções que consigam abranger os três sistemas, em simultâneo. Para além disso, tendo em consideração o aparecimento de ambientes de desenvolvimento multiplataforma torna-se fulcral perceber qual a melhor opção.

Manter a mesma aplicação para três sistemas operativos diferentes, de forma individualizada, pode tornar-se uma tarefa desgastante e dispendiosa, visto ser necessário conhecer bem os sistemas em questão (que por sua vez utilizam linguagens de programação bastante distintas), assim como o próprio *software* de desenvolvimento ou os requisitos para tal. Através deste tipo de soluções multiplataforma, em vez de serem necessárias três equipas de desenvolvimento, passa a ser necessária apenas uma única equipa de programadores, capaz de dominar uma linguagem.

Tal como se pode observar na figura 19, as possibilidades vão desde simples páginas *web* adaptadas às proporções dos dispositivos móveis, até aplicações nativas. Aplicações totalmente nativas criadas individualmente para cada um dos três sistemas operativos móveis não serão abordadas, visto que um dos requisitos deste estudo é tentar perceber a viabilidade de criar uma única solução multiplataforma. Questões como desempenho e interface com o utilizador, face ao tempo gasto e aos custos, podem igualmente ser observados na figura seguinte.

Um outro ponto a destacar é o facto de a *CimSoft* usar a linguagem de programação *C#* nas suas aplicações, contendo grande parte das classes e métodos nessa mesma linguagem. Reutilizar código seria uma mais valia.

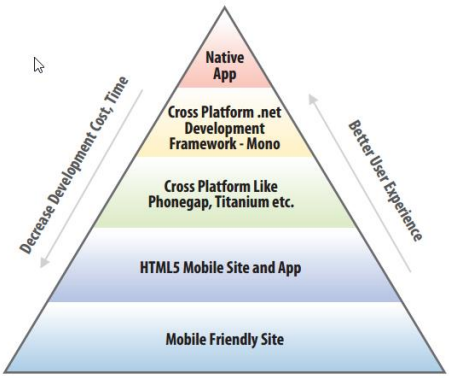

Figura 19 – Prós e contras da utilização de aplicações nativas, híbridas ou *web* [71]

3.3.1 Adobe PhoneGap / Apache Cordova

O *PhoneGap* é um projeto de código fonte aberto (*open-source)* lançado em dezembro de 2006 com vista a potenciar o desenvolvimento de aplicações móveis para multiplataformas. Tratava-se de uma solução interessante e altamente valorizada pelas organizações, visto que permitia diminuir os custos e tempo de desenvolvimento.

Em 2011, imediatamente antes da aquisição da *Nitobi*, a empresa que criou e geriu o projeto *PhoneGap,* doou-o à *Apache Software Foundation*, tendo em vista a longevidade e gestão do mesmo. Pouco tempo depois, a *Nitobi* concluiu o processo de venda, passando a pertencer à

Adobe. É precisamente a partir desse momento que o *PhoneGap* muda de nome, passando a existir em duas versões semelhantes, mas com nomes diferentes, tal como se pode observar na figura 20:

- *Apache Cordova*: em tudo semelhante ao projeto inicialmente conhecido como *PhoneGap,* contudo agora controlado pela *Apache Software Foundation* [72].
- *Adobe PhoneGap:* uma distribuição do *Apache Cordova* com algumas adições e funcionalidades extra adicionadas pela *Adobe* [72]*;*

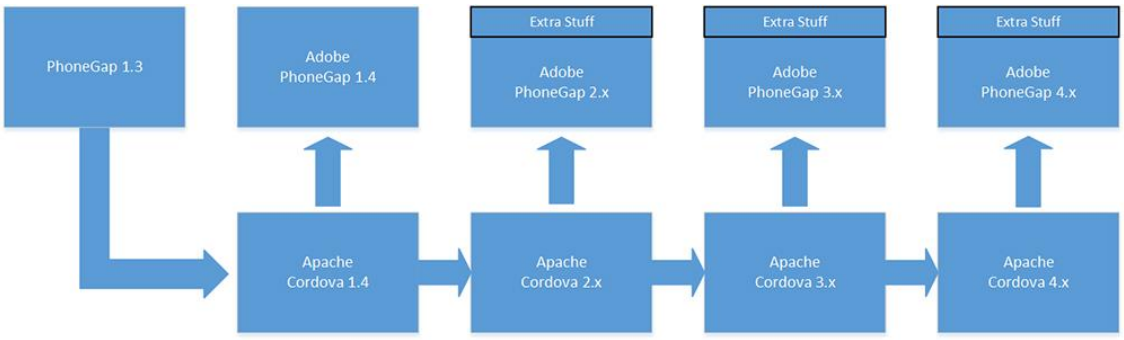

Figura 20 – PhoneGap e a origem do *Apache Cordova* e *Adobe PhoneGap* [72]

O *Apache Cordova* permite desenvolver aplicações para dispositivos móveis utilizando *CSS3, HTML5* e *JavaScript* [73].

A interface é implementada utilizando as tecnologias *web* previamente descritas, o que se traduz numa página *web* capaz de utilizar a largura e altura do ecrã dos dispositivos móveis. Esta página é interpretada pela *WebView*, a classe responsável pela interpretação do código *HTML, CSS e JavaScript,* de forma nativa em cada um dos dispositivos móveis suportados (tais como *iOS, Android, BlackBerry, Nokia Symbian OS, Windows Phone, Firefox OS, etc).* Esta classe tem diferentes nomes dependendo do sistema operativo em questão: *android.webkit.WebView* no caso do *Android*, *UIWebView* no caso do *iOS*, etc [73].

Através do uso de *foreign function interface* (ou *FFI*, um mecanismo que permite que um programa seja escrito numa linguagem de programação e usar serviços ou rotinas escritas numa outra linguagem de programação), o *Apache Cordova* introduz o código *HTML5* numa WebView nativa, sendo que estas *WebViews* têm acesso a funções nativas como a câmara fotográfica, acelerómetro, geolocalização, armazenamento, entre outras funções.

O resultado disso são aplicações híbridas, visto que não são totalmente nativas nem totalmente baseadas na *Web.* No entanto, são instaladas nos dispositivos móveis como se de uma aplicação nativa se tratasse: é criado um ficheiro *IPA* no caso do *iOS*, um ficheiro *APK* no caso do *Android* e um ficheiro *XAP* no caso do *Windows Phone.*

Através de uma *API* (*Application Programming Interface*) é possível aceder a funcionalidades nativas dos diversos sistemas operativos móveis, utilizando *JavaScript*: a lógica da aplicação é escrita em *JavaScript* e a *API* encarrega-se da comunicação com o sistema operativo (*WebView)* [73].

Na Figura seguinte é possível ver a estrutura da arquitetura alto nível das aplicações desenvolvidas:

Figura 21 – Arquitetura de alto-nível das aplicações [73]

Essas aplicações funcionam como um cliente que comunica com a aplicação alojada num servidor, sendo esse servidor responsável por comunicar com a base de dados. A aplicação alojada no servidor é geralmente escrita utilizando a linguagem de programação *PHP* ou *.NET* e executada em servidores *Web*, tais como o *Apache* ou *IIS.*

Esta solução torna-se particularmente interessante, visto que a comunicação com a base de dados não é feita diretamente através da aplicação móvel. Por outro lado, a comunicação entre a aplicação móvel e a aplicação servidor é feita através de pedidos *HTTP*, utilizando serviços como *REST-ful XML, JSON* ou *SOAP.* A página ou interface do lado da aplicação móvel vai-se atualizando conforme recebe os dados da aplicação servidor, sendo que esta página esta sempre carregada em memória [73].

Uma das melhorias introduzidas pela *Adobe* no caso do *Adobe PhoneGap* é a criação de um serviço ou servidor que permite *compilar* o código de forma nativa para os sistemas operativos escolhidos pelo utilizador, evitando as dificuldades ou complicações em configurar tal processo. Os programadores criam um ficheiro onde incluem os recursos utilizados, assim como o código (*HTML5, CSS* e/ou *JavaScript)* que é posteriormente enviado para o servidor. Caso os programadores optem por não utilizar a versão criada pela *Adobe*, o processo é ligeiramente mais complicado e trabalhoso, dado que no caso do *iOS* irão precisar de um computador com o sistema operativo *Mac OS* instalado ou um computador com o sistema operativo *Windows* instalado, no caso de quererem criar uma aplicação *compilada* nativamente para *Windows Phone* [72].

3.3.2 Appcelerator Titanium

O *Appcelerator Titanium* é um projeto de código fonte aberto (*open source)* que contém uma plataforma para programadores *web* que permite criar soluções multiplataforma para dispositivos móveis usando a linguagem *JavaScript* [74].

Essas soluções podem ser nativas, híbridas ou puramente baseadas na *web*, suportando os sistemas operativos *iOS, Android, Windows, BlackBerry* e ainda versões puramente *web* em *HTML5* [74].

Os principais componentes do *Appcelerator Titanium* são o *Titanium SDK,* que fornece as diferentes *API's*, e o *Alloy*, que simplifica a criação da *interface* com o utilizador.

À semelhança de outras soluções multiplataforma, o *Appcelerator Titanium* permite o acesso a uma panóplia de *API's* possibilitando a utilização de diversas funcionalidades do sistema operativo móvel em questão e do dispositivo, tais como: a localização por *GPS*, sensores, câmara fotográfica e a própria interface com o utilizador [75].

Escritas em *JavaScript*, as aplicações são traduzidas para código nativo de cada um dos sistemas operativos móveis escolhidos. Parte desse código não é compilado, mas sim interpretado, invocando funções nativas para executar a função pretendida. Assim, no decorrer da execução da aplicação, são criadas pontes que servem de ligação entre as funções nativas da *Titanium API* e o código *JavaScript.*

De forma resumida, uma aplicação criada em *Appcelerator Titanium* possui três componentes [76]:

- Código fonte em *JavaScript;*
- Implementação da *Titanium API* para o sistema operativo móvel em questão;
- Um interpretador de *JavaScript* (*Mozzila Rhino* no caso do *Android* e *BlackBerry* e *JavaScriptCore* no caso do *iOS).*

Um outro componente é o *Alloy*, uma ferramenta de código fonte aberto que permite criar aplicações de acordo com o padrão *MVC,* o que permite separar a camada de *interface*, camada lógica e a camada de dados. De forma a personalizar a *interface* com o utilizador, o *Alloy* usa *XML* e *CSS* [76].

Uma das desvantagens desta solução para criação de soluções multiplataforma, comum a muitas ferramentas multiplataforma, consiste no facto de o desempenho não ser tão bom quando uma aplicação semelhante de forma totalmente nativa [76].

3.3.3 Xamarin

O *Xamarin* é uma plataforma resultante do esforço de vários anos de desenvolvimento. Trata-se da junção de vários projetos/livrarias resultantes do *Mono.* O *Mono* é um projeto grátis e de código fonte aberto, baseado num *compilador* de *C#* e um *CLR* (*Common Language Runtime*, um componente de máquina virtual encarregue da execução da aplicação), que permite executar aplicações escritas em *C#* em sistemas operativos alternativos ao *Windows,* como o Linux e o *Mac OS.*

As várias livrarias resultantes do *Mono* formam aquilo que chamamos de *Xamarin Platform* (plataforma *Xamarin):*

- *Xamarin.MAC,* resultante da evolução do projeto *MonoMac;*
- *Xamarin.iOS*, proveniente do *MonoTouch;*
- *Xamarin.Android*, uma evolução do *Mono* para *Android (MonoDroid).*

Embora existam versões individuais do *Xamarin* para *Android* e o *iOS,* o princípio base é diferente do esperado para esta dissertação. Nesse tipo de soluções, é necessário criar e programar versões diferentes para cada um dos sistemas operativos, podendo partilhar algum código entre as versões. Trata-se de uma solução indicada para:

- Aplicações que requeiram interações nativas;
- Aplicações que utilizem demasiadas *API's* dependentes da plataforma;
- Aplicações que necessitem de uma interface com o utilizador polida, sendo o aspeto da aplicação mais importante do que a partilha de código.

Na Figura 22 é possível verificar o princípio base deste paradigma:

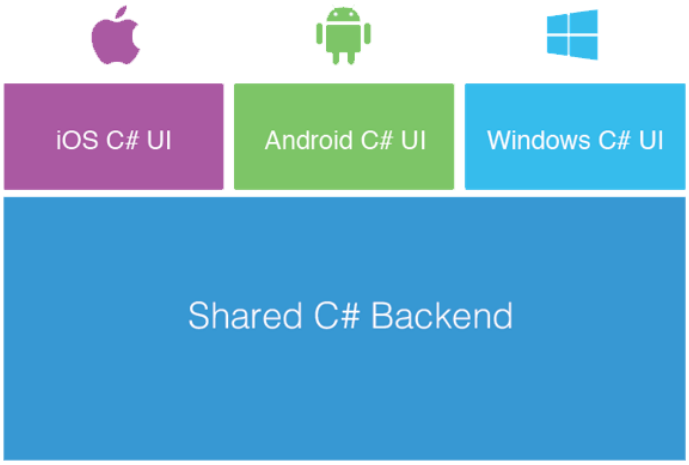

Figura 22 – Princípio base da plataforma *Xamarin* [77]

Por outro lado, é possível obter aplicações igualmente nativas, desenvolvidas em *C#*, partilhando a mesma interface com o utilizador entre os três sistemas operativos, poupando bastante tempo no desenvolvimento individualizado da mesma. O *Xamarin.Forms* é utilizado fundamentalmente em:

- Aplicações que não necessitem funcionalidades dependentes de cada uma das plataformas;
- Protótipos ou provas de conceito;
- Aplicações em que a partilha de código seja mais importante do que interfaces com o utilizador personalizadas;

A Figura 23 representa o princípio base das soluções desenvolvidas em *Xamarin.Forms:*

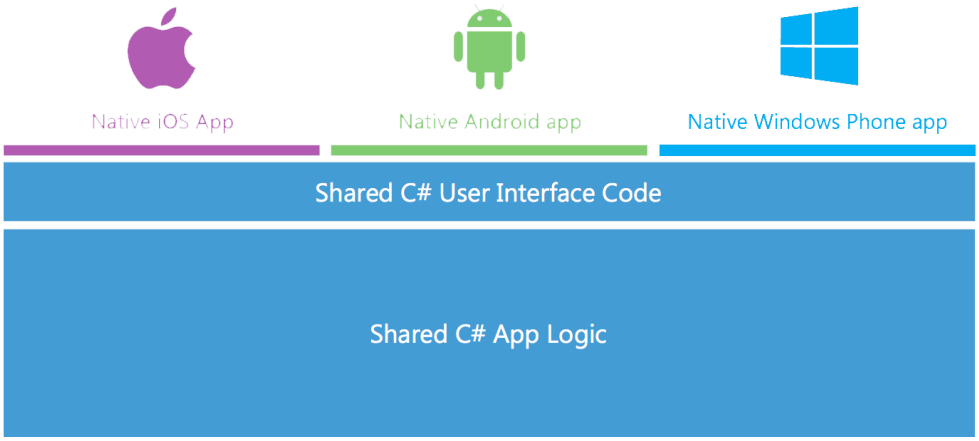

Figura 23 – Princípio base da plataforma *Xamarin.Forms* [78]

Uma única interface com o utilizador, utilizando a premissa do mínimo denominador comum. Botões, caixas de texto, barras, páginas e layouts são comuns e funcionam da mesma maneira entre todos os sistemas operativos escolhidos. Embora isto seja uma vantagem, é igualmente uma limitação no caso de se pretender desenvolver algo mais complexo ou diferenciado. Uma outra desvantagem é a interface com o utilizador ter de ser totalmente configurada através de código *XAML* ou *C#,* não existindo nenhuma ferramenta gráfica para tal.

A tecnologia disponibilizada pelo *Xamarin* é promissora e encontra-se em constante melhoria através das diversas atualizações. Encontrar informação e ajuda através dos fóruns do *Xamarin* ou *StackOverflow* não é de todo um problema, assim como a disponibilização da documentação das *API's* e guias. A recente compra do *Xamarin* por parte da *Microsoft* vem comprovar o seu valor e longevidade futura [79]. Um outro fator positivo é o facto de atualmente o *Xamarin* vir incluído de forma gratuita no *Visual Studio 2015,* sendo este um ambiente de desenvolvimento bastante conhecido dos programadores.

Segundo uma apresentação decorrida na conferência anual (*Evolve 16)* realizada pelo *Xamarin*, que decorreu entre 24 e 28 de abril de 2016, quando se compara uma aplicação híbrida e uma aplicação multiplataforma nativa *Xamarin,* conclui-se que as aplicações nativas são: [80]

- 25% mais rápida a iniciar;
- 62% mais rápidas a abrir um conjunto de dados (*dataset);*
- 50% menos memória utilizada;
- 76% menos *CPU* utilizado;
- 22x mais rápida a completar tarefas que requeiram capacidade de processamento.

Nessa mesma apresentação, e tal como se pode comprovar na figura 24, consideram o *Xamarin* e o *React* como soluções de terceira geração no que diz respeito ao desenvolvimento de aplicação multiplataforma nativas. Ao contrário das outras soluções de primeira e segunda geração, esta é mais completa e não falha onde outras falham. Realçam o facto de serem as duas soluções que atualmente permitem uma experiência de utilização e interface nativa, alto

desempenho, multiplataforma, código unificado, acesso fácil às *API's* dos sistemas operativos móveis em questão, assim como a interação com o *hardware*.

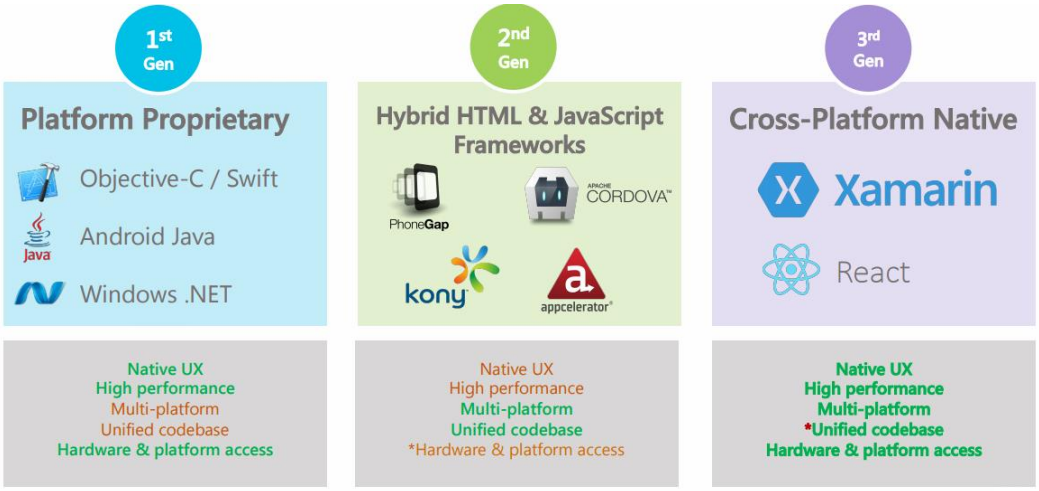

Figura 24 – Soluções nativas *versus* Multiplataforma [80]

3.3.3.1 Restrições e desenvolvimento

Embora o *Xamarin* suporte diversos sistemas operativos móveis, através da instalação de uma extensão para o *Visual Studio* ou através do *Xamarin Studio*, a instalação de uma série de extras não é dispensada.

Para desenvolver aplicações para o sistema operativo móvel *Android,* é necessário que o *SDK* e as ferramentas do mesmo estejam instaladas, consoante a versão do *Android* / *API level* desejada. O mesmo se aplica à instalação do Java (*JDK).* Estas ferramentas fornecem o acesso ao compilador, *emulador*, que permitem a implementação e testes das aplicações. No fundo é semelhante ao desenvolvimento tradicional de uma aplicação *Android* e todos os requisitos que isso implica.

Já o *iOS* é um caso totalmente diferente. Para começar, é fundamental possuir um computador da *Apple* com o sistema operativo *Mac OS* instalado e devidamente registado na *App Store*. Seguidamente é necessário instalar o *Xcode* através da *App Store* e fazer o download das *API's e emuladores* necessários, de forma a poder compilar e simular um *iPhone* ou *iPad.* É igualmente imprescindível instalar o *Mono* e o *Xamarin* no *Mac OS.* Existe ainda a necessidade de efetuar algumas configurações no computador com *Mac OS* de forma a permitir conexões *SSH* e *login* remoto [81].

Por fim, os projetos criados para *Windows 8/8.1/10 Mobile* não necessitam configurações adicionais, visto que o *Visual Studio* contem quase tudo o que é necessário, sendo apenas preciso efetuar o *download* do *emulador*, de forma a poder testar a aplicação (no caso de não possuir um dispositivo físico). É ainda aconselhado que o computador de desenvolvimento tenha o *Windows 8* ou idealmente o *Windows 10* instalado.

A Figura seguinte representa as diversas possibilidades de ambientes de desenvolvimento, consoante o sistema operativo para desenvolvimento escolhido. Destaque especial para o *Visual Studio* em ambiente *Windows,* utilizando *Xamarin.Forms.*

	MACOS	WINDOWS	
Development Environment	XAMARIN STUDIO	VISUAL STUDIO	XAMARIN STUDIO
Xamarin.iOS	Yes	Yes (with Mac computer)	No
Xamarin.Android	Yes	Yes	Yes
Xamarin.Forms	iOS & Android only	Android, Windows, Windows Phone (iOS with Mac computer)	Android only
Xamarin.Mac	Yes	Open project & compile only ^	No

Figura 25 – Ambientes de desenvolvimento e os sistemas operativos [82]

3.3.3.2 Shared Projects

No *Xamarin.Forms* o código pode ser partilhado de duas maneiras diferentes. Uma delas é os *"Shared Projects".*

Shared Projects, ou em português "Projetos Partilhados", é possivelmente o mais simples de utilizar. Visualmente é fácil distinguir este tipo de projeto visto que o código é dividido da seguinte forma:

- Um projeto que contém o código comum a todas as plataformas;
- Um projeto por cada sistema operativo escolhido, comummente um destinado ao *Android*, outro ao *iOS* e outro destinado a *Windows Mobile.* Cada um destes projetos contêm código ou implementações específicas de cada um dos projetos.

Dentro do projeto partilhado é ainda possível definir através da condição *"#if*, partes de código que correspondem um sistema operativo específico, sendo esse bocado de código apenas interpretado para esse sistema operativo. Por exemplo, se utilizar *"#if __ANDROID__",* escrever algum código e finalizar com o *"#endif"*, todo o código compreendido entre ambas as *tags* será apenas interpretado pelo *Android.*

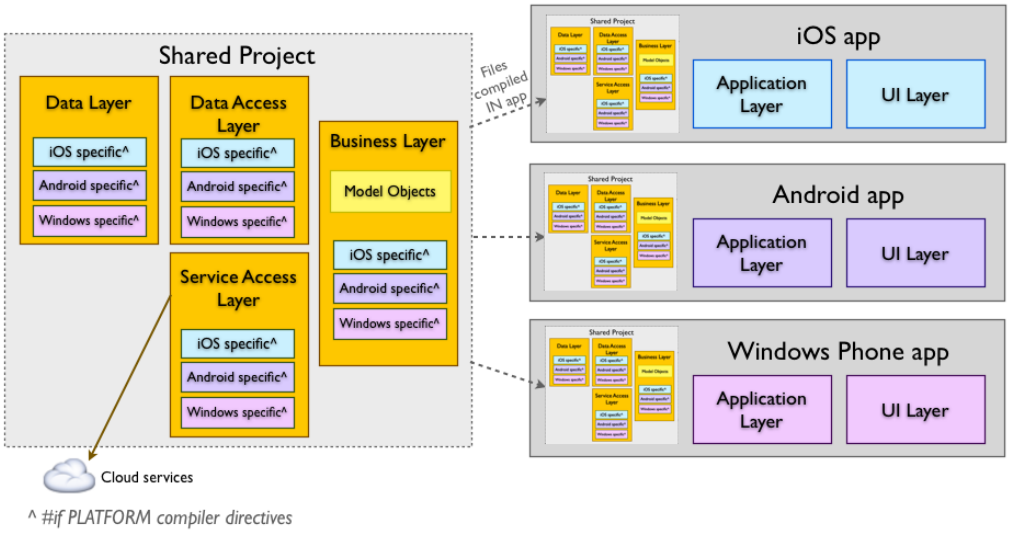

Figura 26 – Funcionamento dos *Shared Projects* [83]

Como podemos observar na figura 26, um *Shared Project* não é compilado individualmente, mas em conjunto com cada um dos projetos específicos de cada plataforma.

Uma das desvantagens apontadas para este tipo de projeto de código partilhado consiste no facto de não produzir um *DLL* para o programa principal, mas sim para cada uma das bibliotecas utilizadas / referenciadas (neste caso, os projetos individuais dos sistemas operativos móveis em questão).

3.3.3.3 Portable Class Libraries

Neste método de partilha de código, e ao contrário do que foi dito no anterior, verifica-se a criação *DLL's* destinados a plataformas específicas. Na prática isso previne que um programa desenvolvido para o *Android* seja reutilizado numa outra plataforma.

Os projetos *Portable Class Libraries* permitem especificar uma ou mais plataformas para as quais aquele programa pode ser executado. Essas especificações são guardadas num identificador de perfil que descreve quais as plataformas que essa libraria suporta [84].

Enquanto o *Xamarin.iOS* e o *Xamarin.Android* suportam todos os perfis, o mesmo não acontece com o os projetos para *Windows Phone*, visto que existe algumas limitações por parte das livrarias escolhidas. A Tabela 5 representa essas mesmas limitações.

Feature	.NET Framework	Windows Store Apps	Silverlight	Windows Phone	Xamarin
Core	Y	Y	Y	Y	Y
LINQ	Y	Y	Y	Y	Y
IQueryable	Y	Y	Y	7.5 +	Y
Serialization	Y	Y	Y	Y	Y
Data Annotations	4.0.3 +	Y	Y		Y

Tabela 5 — Bibliotecas disponíveis para os diferentes sistemas operativos móveis [84]

O funcionamento do método anteriormente descrito pode ser observado na seguinte Figura 27:

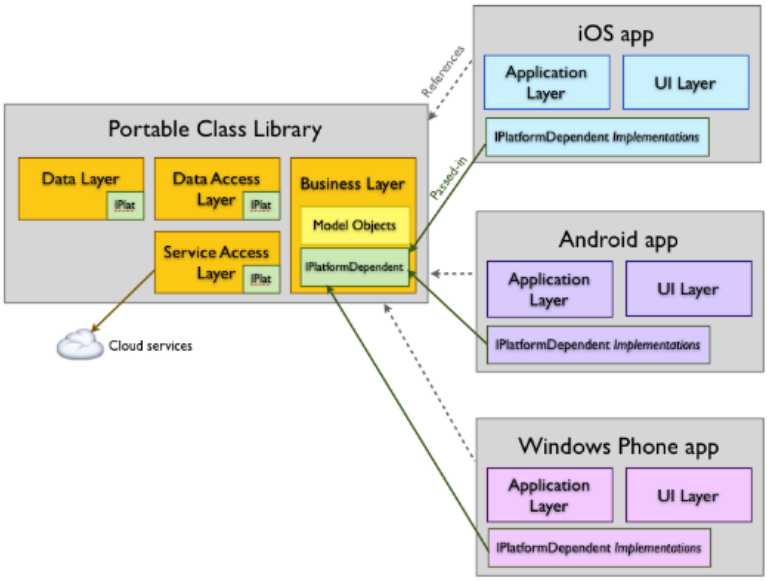

Figura 27 – Arquitetura dos projetos *Portable Class Libraries* [84]

Os principais benefícios dos projetos *Portable Class Libraries* passam pela partilha de programas desenvolvidos, permitindo que outros programas referenciem o programa inicial. Como desvantagem destaca-se a incompatibilidade de livrarias entre as mais diversas plataformas, como explicado anteriormente. O método *Dependency Injection*, como demonstrado na figura anterior, permite contornar alguns desses problemas.

3.3.3.4 Interface com o utilizador

O *Xamarin.Forms* disponibiliza uma série de elementos gráficos que tem por base o mínimo denominador comum entre os diversos sistemas operativos móveis selecionados. Por exemplo, o elemento gráfico "*Entry*", que corresponde a uma caixa de introdução de texto, é na prática uma *UITextView* no caso do *iOS,* uma *EditText* no caso do *Android* e uma *TextBox* no caso do sistema operativo móvel *Windows Phone.*

São disponibilizados vários *Layouts, Pages* e *Controls* [85] que de um modo geral conseguem cobrir a maioria das necessidades dos utilizadores. Embora isto seja uma mais valia, por vezes a customização a que os ambientes nativos acostumaram os programadores não se encontra disponível, sendo as opções mais limitadas. Porém, através da instalação dos mais diversos pacotes de *Nuget* [86] e *Components* [87] disponíveis, é possível estender as funcionalidades, assim como obter novos controlos e *API's* para as mais diversas necessidades dos programadores.

O *Xamarin.Forms* permite a criação de interfaces com o utilizador através de código nativo em *C#,* assim como em *XAML.* O *XAML* proporciona o desenho através de métodos como *drag-and-drop,* o que por si só é algo apelativo, contudo, essa funcionalidade não se encontra disponível em projetos *Xamarin.Forms Shared Project.* Por um lado, verifica-se que a tecnologia *XAML* tem alguns problemas de desempenho, tendo em conta que o código não é compilado. Trata-se, assim, de uma desvantagem clara para aplicações que utilizem um considerável volume de dados, em especial no caso de utilizarem *ListViews.* Por outro lado, a interface com o utilizador escrita em *C#*, permite um maior desempenho e controlo, assim como é mais acessível para iniciantes.

3.4 Restrições Existentes

Tal como referido anteriormente, esta dissertação foi realizada em ambiente empresarial (estágio). Embora a *CimSoft* pretenda um estudo de funcionalidades avançadas para pontos de venda móveis, culminando com um protótipo para exemplificação, existem algumas restrições impostas pela mesma.

Nos dois produtos que comercializam, a politica da empresa é tentar partilhar código e recursos. Assim sendo, têm vindo a utilizar *C#* como linguagem de programação e não pretendem mudar. De forma a conseguir executar as soluções no sistema operativo *Linux*, para além do *Windows*, utilizam o *Mono* [88] e sempre que possível *API's (Application Programming Interface)* de código aberto e gratuitas. O *Mono* é um projeto grátis de código fonte aberto que permite *compilar* código em *C#* que pode ser executado em diversos sistemas operativos como o *Windows, Linux* e *Mac OS.*

Assim sendo, a utilização da plataforma *Xamarin* [89] é uma imposição, com vista a verificar o quão viável, no estado atual, a plataforma se encontra. O *Xamarin* permite partilhar grande parte do código, se não mesmo a totalidade, sendo posteriormente *compilado* nativamente para os mais diversos sistemas operativos móveis, nomeadamente o *Android, iOS* e *Windows Phone*. Esta prática pode resultar numa grande poupança de tempo e recursos, visto que o mesmo código ou solução é viável nos diversos sistemas operativos.

Uma outra restrição imposta pela empresa é que a mesma solução se comportasse da mesma forma nos três sistemas operativos móveis. A ideia seria contornar as dificuldades e limitações existentes em cada um deles numa solução comum e funcional em todos. Só assim faria sentido suportar os três sistemas operativos móveis. Caso contrário, existiria sempre algum com alguma

vantagem extra ou funcionalidade, pelo que haveria um desequilíbrio na oferta. A empresa aplica esta regra nas suas duas soluções comerciais.

Uma outra restrição, não imposta diretamente pela empresa, era o acesso a parte do material. No decorrer deste estudo veremos que uma das funcionalidades interessantes seria a utilização e comunicação com algum material *Bluetooth*. Contudo, a empresa não dispunha do mesmo nem demonstrou interesse em efetuar qualquer investimento nesse sentido. Assim sendo, tornou-se complicado implementar e comprovar algumas funcionalidades interessantes.

3.5 Sumário

Ao longo deste capítulo foi possível perceber a base na qual esta dissertação assenta, tendo sido enaltecida a base tecnológica da aplicação em estudo: o *MediaPOS,* bem como a inexistência de uma aplicação que possa interagir ou funcionar como um ponto de venda independente.

De modo a desenvolver um protótipo da aplicação foram consideradas algumas abordagens tecnológicas, nomeadamente a nível de *Hardware*, tecnologias de comunicação com periféricos e alguns métodos de pagamentos móveis.

A dissertação e estudo foram elaborados com base num estágio proposto pela *CimSoft* e aceite pelo Instituto Superior de Engenharia do Porto. Como tal, este projeto foi desenvolvido num ambiente empresarial, em que grande parte das funcionalidades e abordagens foram decididas pelo representante da *CimSoft,* tendo em consideração os objetivos e interesses da empresa. Nesse sentido, torna-se um pouco irrelevante tirar conclusões das diversas tecnologias apresentadas e estudadas neste capítulo, em detrimento de outras. Ao invés, torna-se fundamental identificar e analisar as várias abordagens, que possam constituir funcionalidades avançadas de interação com pessoas e equipamentos. Foram ainda evidenciados alguns dos possíveis cenários ou aplicabilidades dessas mesmas abordagens.

O mesmo se aplica aos ambientes de desenvolvimento. Um dos objetivos propostas consiste na utilização da plataforma *Xamarin* no desenvolvimento de uma solução multiplataforma. Foram apresentadas algumas plataformas alternativas, porém é evidente a superioridade da plataforma *Xamarin*, não só pelo facto de permitir aplicações verdadeiramente nativas e com maior desempenho, mas também pelo facto de permitir um grande leque de *API's*, que permitem tirar partido das mais diversas funcionalidades disponíveis nativamente nos dispositivos móveis. Um dos pontos a favor desta decisão consiste na aquisição do *Xamarin* por parte da *Microsoft*, sendo esperado um grande crescimento e aposta no decorrer dos próximos anos, assim como um maior suporte.

4 Proposta e Desenvolvimento de Solução

Este capítulo visa descrever todos os aspetos referentes à implementação do protótipo proposto nesta dissertação. Numa primeira fase é identificada a proposta de solução e os seus componentes constituintes. São igualmente identificados os requisitos funcionais e não funcionais, assim como a arquitetura da solução proposta. Numa segunda fase é abordada a implementação da solução, na qual é evidenciado o aspeto gráfico, as funcionalidades e a metodologia utilizada.

4.1 Proposta de Solução

A proposta de solução identificada nesta secção visa dar uma resposta às necessidades e problemas previamente descritos no âmbito desta dissertação. A lacuna existente na *CimSoft* é clara e potenciou uma oportunidade de negócio: a criação de um sistema de pontos de venda para dispositivos móveis. Muito embora as empresas concorrentes já disponham de soluções semelhantes à proposta, essas soluções são proprietárias e em nada resolvem o problema existente na *CimSoft*, nem servem como resposta aos objetivos propostos.

O único membro da equipa que também é o responsável pela empresa, não detém conhecimentos na área de desenvolvimento para dispositivos móveis, nem das tecnologias emergentes. No entanto, possui uma larga experiência no desenvolvimento de *software* para pontos de venda e tem conhecimento das necessidades existentes neste mercado, assim como das exigências dos clientes. Nesse sentido, utilizando os conhecimentos adquiridos ao longo da realização deste estudo, encontram-se reunidas as condições ideais para projetar um sistema protótipo que servirá como base, após as devidas validações legais, para um sistema pronto a ser utilizado pelos clientes.

A *CimSoft* não dispõe de todo o material necessário para o desenvolvimento de algumas das propostas de solução analisadas, tais como impressoras, balanças ou outros dispositivos, que comunicam através do protocolo *Wi-Fi* ou *Bluetooth.*

Tendo em conta a politica interna da empresa, os interesses e expectativas para o protótipo, respeitando o público-alvo e reaproveitando a lógica de serviço e infraestruturas já existentes, a solução abarcará os seguintes componentes:

- Uma aplicação móvel (*Windows Phone, Android* e *iOS)*, desenvolvidas em *Xamarin,* que permitam*:*
    - Auxiliar o negócio através da consulta de faturas e estatísticas;
    - Utilização da câmara fotográfica na automatização de processos;
    - Pagamentos móveis através do *MEO Wallet;*
    - Executar o ciclo de venda comum de um ponto de venda (escolher, adicionar ao carrinho e pagar).
    - Imprimir uma éspecie de fatura / recibo.
- Um *Raspberry Pi* ou computador que permitirá:
    - Alojar a base de dados;
    - Alojar o *Web Service*;
    - Comunicar com periféricos.

Tal como referido anteriormente, as escolhas da abordagem a implementar no protótipo, assim como as funcionalidades avançadas, ficaram ao encargo da *CimSoft.*

As próximas secções procuram esclarecer os requisitos do sistema a desenvolver bem como a especificação da proposta. Numa primeira fase são demarcados os intervenientes e os casos de uso, enaltecendo os requisitos funcionais e não funcionais. Numa fase posterior é apresentada a arquitetura da solução, assim como as suas formas de interação e comunicação.

4.1.1 Requisitos Funcionais e Não Funcionais

4.1.1.1 Atores do sistema

No levantamento de requisitos é fundamental identificar corretamente quais os intervenientes do sistema:

- Funcionário: indivíduo pertencente à empresa que utiliza a solução desenvolvida, capaz de interagir com o sistema para consultar faturas, emitir pagamentos, “adicionar ao carrinho”, entre outros;
- Cliente: indivíduo que poderá ocasionalmente interagir com o sistema de modo a efetuar um pagamento;

4.1.1.2 Casos de uso

Nesta secção é apresentado um diagrama de casos de uso. A Figura 28 ilustra as interações do Funcionário e do Cliente com a solução.

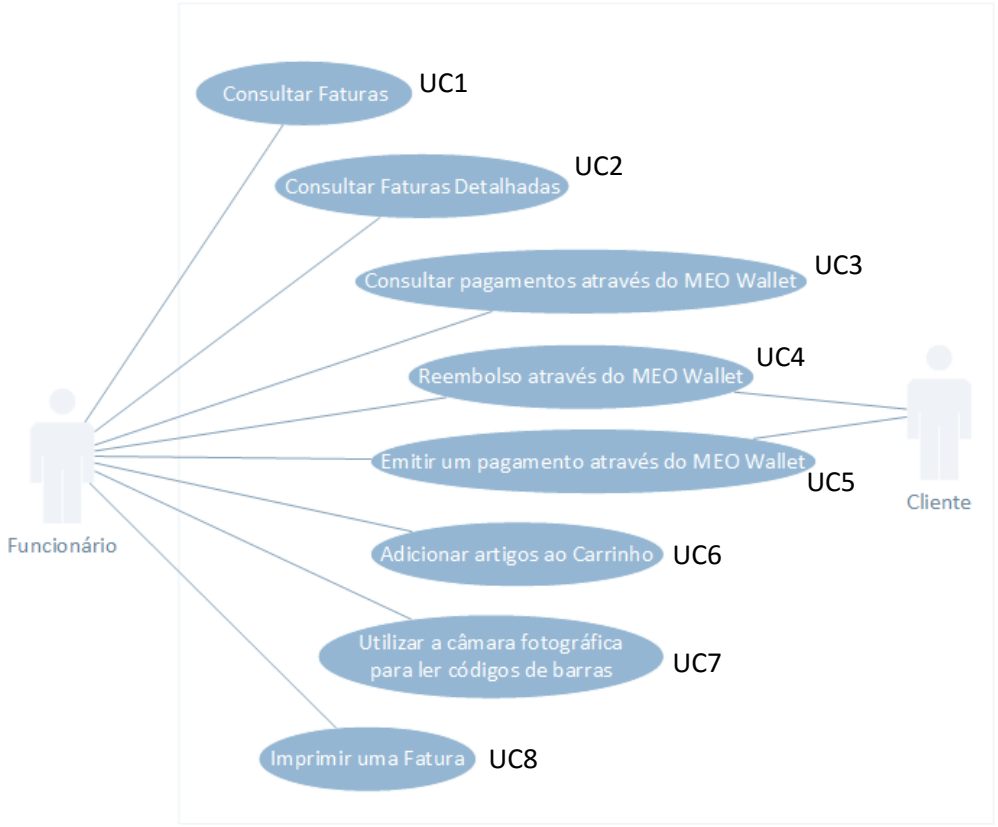

Figura 28 – Diagrama de casos de uso da solução

Os casos de uso são utilizados de forma a apresentar as funcionalidades principais do sistema do ponto de vista dos atores do sistema. Cada caso tem uma identificação única que permitirá associar a requisitos funcionais no decorrer das próximas secções.

4.1.1.3 Requisitos funcionais

Os requisitos funcionais representam as funções que o sistema terá de desempenhar e disponibilizar aos utilizadores. As funções podem ser interpretadas como os processos que recebem uma entrada, tratam-na e apresentam uma saída. A Tabela 6 apresenta os requisitos funcionais e o seu plano de testes.

Tabela 6 — Requisitos Funcionais do Sistema

ID	Descrição	Plano de Teste	Casos de Uso
RF1	O sistema deverá permitir que o Funcionário aceda a uma listagem de faturas de acordo com um conjunto de parâmetro pré-estabelecidos (data, *ID*, etc). O sistema retorna uma lista de faturas filtrada pelos parâmetros.	Introduzir parâmetros correta e incorretamente em tentativas seguidas e verificar os resultados obtidos em ambas.	UC1 e UC2
RF2	O sistema deverá permitir que o Funcionário consulte uma fatura específica em detalhe.	Introduzir a identificação de uma fatura existente e de uma inexistente, verificando os resultados obtidos.	UC2
RF3	O Funcionário deve ser capaz de aceder a uma listagem de pagamentos previamente efetuados utilizando a *API* do *MEO Wallet,* com base em parâmetros tais como a data atual, uma data específica, etc.	Aceder ao menu de pagamentos e verificar os resultados obtidos com base nos parâmetros introduzidos de forma correta e incorreta.	UC3
RF4	O Funcionário deverá poder emitir um pedido de reembolso de um pagamento previamente efetuado, podendo o valor reembolsado ser total ou parcial. Por seu lado, o Cliente recebe a confirmação desse reembolso.	Após efetuar um pagamento bem-sucedido, verificar o sucesso ou insucesso do pedido de reembolso, com base em valores totais ou parciais ,corretos e errados.	UC4 e UC5
RF5	O Funcionário deverá ter a possibilidade de requerer um pedido de pagamento utilizando o serviço do *MEO Wallet,* fornecendo ao Cliente o acesso a um *QR Code* ou possibilidade de efetuar o pagamento utilizando o seu dispositivo móvel.	Após adicionar ao carrinho alguns artigos e aceder ao menu de pagamento, seleciona-se a opção de pagamento por *MEO Wallet*. Deve-se verificar o sucesso ou insucesso do pagamento utilizando tal serviço, assim como a mensagem de *feedback* correto no ecrã.	UC5 e UC6
RF6	Os artigos devem poder ser adicionados ao Carrinho através da interação do Funcionário com o sistema.	Selecionar alguns artigos através do ecrã inicial, verificando o aparecimento correto dos mesmos no Carrinho, assim como o preço e quantidade.	UC6
RF7	O Funcionário deve ter a possibilidade de utilizar a câmara fotográfica para efetuar a leitura de códigos de barras, por exemplo na identificação de um artigo ou cliente.	Através do menu inicial acede-se à funcionalidade de leitura de códigos de barras de um artigo. Verificar através de experiências sucessivas o sucesso e insucesso da leitura de artigos válidos e inválidos.	UC6 e UC7

ID	Descrição	Plano de Teste	Casos de Uso
RF8	O Funcionário deve poder indicar ao sistema a intenção de imprimir uma fatura, comunicando com o dispositivo periférico: impressora.	Através do ecrã de pagamentos, tentar de forma sucessiva imprimir diversas faturas e verificar os resultados obtidos	UC8

4.1.1.4 Requisitos não funcionais

Os requisitos não funcionais são os requisitos que estão relacionados com a utilização da aplicação no que diz respeito à usabilidade, segurança, *interface*, desempenho, entre outros. Trata-se de características do produto que não representam funcionalidades, mas aumentam a qualidade da solução proposta.

Tabela 7 — Requisitos Não-Funcionais do Sistema

ID	Descrição	Plano de Teste
RNF1	O sistema deverá estar acessível em qualquer lugar, através da ligação à *Internet.*	Executar a aplicação móvel utilizando a rede local onde o sistema se encontra e uma outra rede distinta. Verificar se a conexão é estabelecida em ambos os casos.
RNF2	A *interface* com o utilizador deve ser uniforme e adaptável a qualquer tipo de ecrã.	Utilizando vários dispositivos móveis deve-se verificar o comportamento da *interface* gráfica e adaptabilidade às diferentes resoluções, assim como simplicidade da mesma. Verificar os resultados obtidos.
RNF3	As interações com o utilizador devem ser breves e objetivas.	Testar os diversos menus, verificando se o objetivo dos mesmos é claro e permite interações breves com as mais diversas funcionalidades.
RNF4	A aplicação deverá ser o mais rápida possível a devolver os dados de consulta, permitindo uma maior fluidez de utilização.	Consultar diversas faturas e pagamentos, anotando se a experiência de utilização foi satisfatoriamente rápida ou não.
RNF5	Na primeira utilização da aplicação móvel, as mensagens devem ser intuitivas e induzir o utilizador aos menus de configuração do sistema, permitindo testar os dados de ligação introduzidos, entre outros.	Executar a aplicação num dispositivo pela primeira vez, verificando se os procedimentos foram intuitivos de modo a configurar o sistema para estabelecer ligação com o *Web Service.*

4.1.2 Arquitetura da Solução

Com vista a obter uma solução o mais robusta possível e dando resposta aos requisitos funcionais e não funcionais previamente descritos, é necessário projetar uma arquitetura para o sistema.

O contexto na qual esta solução é criada tem por base a elaboração desta dissertação, com vista à criação de um protótipo como prova de conceito e estudo de funcionalidades e viabilidade de um sistema multiplataforma. Nesse sentido, a solução proposta inclui duas partes essenciais: as aplicações móveis e o *Raspberry Pi* (ou um computador).

As aplicações móveis são uma solução multiplataforma criada para interagir com o servidor (*Raspberry Pi* ou um computador) e possibilitar ao utilizador (neste caso os funcionários) a interação com o sistema em si. O *Raspberry Pi* ou um computador são responsáveis pelo armazenamento e comunicação com a base de dados, assim como interação com periféricos.

4.1.2.1 Arquitetura do *hardware*

Esta subsecção destina-se ao estudo da ligação entre os diferentes componentes físicos pertencentes ao *Raspberry Pi*, computador e dispositivos móveis.

A versão utilizada do *Raspberry Pi* integra, para além de uma ligação *ethernet*, uma placa de rede de acesso ao *Wi-Fi,* importante para o sucesso das comunicações entre os vários componentes do sistema à *Internet*. Os dispositivos móveis, por seu lado, possuem igualmente ligação a redes sem fios, nomeadamente acesso ao *Wi-Fi* ou redes móveis tais como o 3G ou 4G.

Como se pode observar na Figura 29, os dispositivos móveis ligam-se à *Internet* ou rede local de modo a poderem interagir com o *Raspberry Pi* ou um computador. Estes são responsáveis pelos seguintes constituintes da solução:

- Base de dados;
- Acesso à base de dados;
- *Web Service;*
- Acesso a dispositivos e comunicação com os mesmos.

Através da *Internet* é ainda possível aceder a *Web Services* de pagamentos móveis.

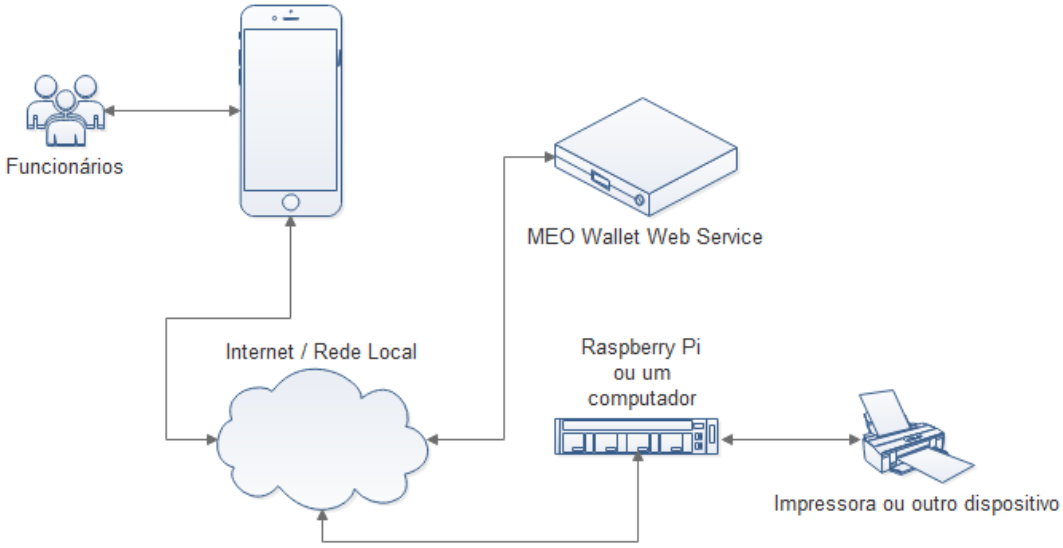

Figura 29 – Esquema da arquitetura do *hardware* da solução

4.1.2.2 Arquitetura da solução

O *software* desenvolvido apresenta várias partes, disponibilizadas quer para os dispositivos móveis, quer para o *Raspberry Pi* / computador. Nesta subsecção serão estudados esses componentes e as comunicações.

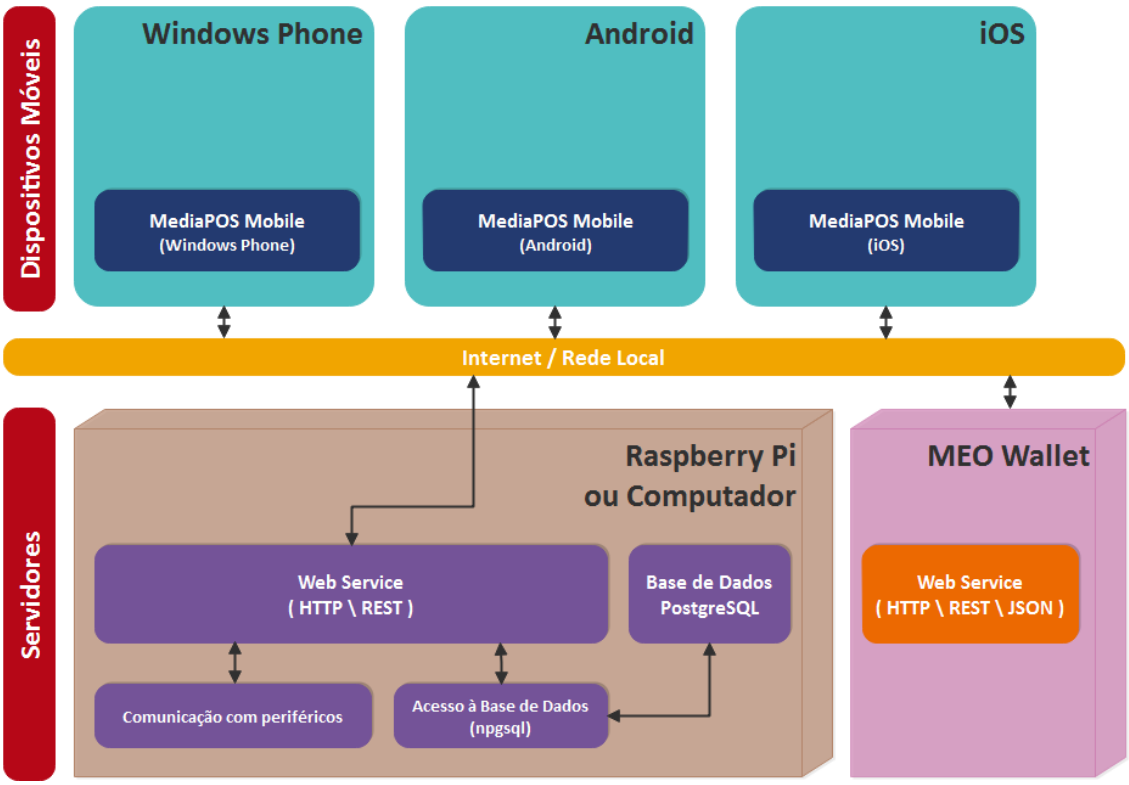

Figura 30 – Esquema da arquitetura do *software* da solução

A Figura 30 representa o esquema utilizado no desenvolvimento da solução. Trata-se de uma arquitetura Cliente-Servidor tradicional.

Como é possível observar, o *Web Service* é o principal componente presente no *Raspberry Pi* ou computador. As comunicações com periféricos (por exemplo uma impressora ou uma balança), assim como o acesso à base de dados são garantidas pelo mesmo:

- Comunicação com periféricos: parte constituinte do *Web Service* que gere a comunicação com periféricos, execução de tarefas / comandos, etc;
- Acesso à base de dados: responsável pela comunicação com a base de dados, permitindo assim autenticar, executar *queries,* entre outros.

O *Raspberry Pi* / computador possuem uma base de dados *PostgreSQL* instalada, sendo que a comunicação com a mesma é sempre realizada através do componente responsável (acesso à base de dados) e nunca diretamente. Isto permite uma maior segurança, impossibilitando a exposição da base de dados a terceiros.

Através da *Internet* é ainda possível comunicar com serviços de pagamentos móveis, como por exemplo o *MEO Wallet,* demonstrado na Figura 30.

4.2 Tecnologias Utilizadas

A secção que se segue visa documentar as diversas tecnologias utilizadas no desenvolvimento da solução proposta, fazendo uma divisão entre as tecnologias utilizadas na aplicação móvel e as utilizadas no *Raspberry Pi* ou computador.

4.2.1 Aplicação Móvel

No decorrer desta dissertação, um dos grandes desafios propostos para estudo consiste no desenvolvimento de uma aplicação móvel para as plataformas *Android*, *Windows Phone* e *iOS,* através de uma solução multiplataforma. Como tal, a aplicação móvel assume um papel de destaque e extrema relevância, visto ser o principal contacto que o utilizador tem com o sistema.

A interação entre os diversos componentes da solução, nomeadamente a vertente de alojamento de serviços (*Raspberry Pi* ou computador) e o utilizador, é feita através das aplicações móveis desenvolvidas.

4.2.1.1 *Xamarin*

O ambiente de desenvolvimento multiplataforma *Xamarin* é, sem dúvida, aquele que mais se adequa ao desenvolvimento da solução proposta. Para além de todas as vantagens previamente referidas no capítulo 3, destaca-se ainda o facto da linguagem de programação ser

o *C#* (linguagem utilizada pela *CimSoft* nos seus produtos), assim como uma experiência de utilização e desempenho quase nativo.

As aplicações móveis foram desenvolvidas utilizando a plataforma *Xamarin Forms*, tendo sido criado um projeto do tipo *Shared Projects*. Foi tomada esta opção porque assim é possível utilizar um código específico por cada sistema operativo móvel escolhido e porque torna os projetos com mais perspetivas de futuro e escalabilidade.

O *Xamarin.Forms* disponibiliza vários elementos gráficos básicos, representados de forma nativa em cada um dos sistemas operativos móveis, recorrendo ao principio do mínimo denominador comum. Ou seja, os elementos disponibilizados existem nos três sistemas operativos móveis, o que logo à partida reduz a possibilidade de utilização de alguns elementos mais específicos de cada um dos sistemas operativos móveis.

A Figura 31 resume os vários elementos gráficos disponíveis pela *API* do *Xamarin.Forms:*

Figura 31 – Elementos gráficos da *API Xamarin.Forms* [90]

Os elementos gráficos disponíveis estão divididos em três grupos:

- *Pages*: Representam os vários tipos de páginas que podem ser utilizados, definindo o tipo de navegação entre as páginas, entre outros;
- *Layouts*: Elementos que atuam dentro das *Pages.* Permitem definir a localização dos vários elementos e *Controls* representados;
- *Controls*: Este grupo disponibiliza todos os controlos disponíveis para interação com o utilizador.

A *API* do *Xamarin.Forms* tem algumas limitações. Diversas funcionalidades disponibilizadas são bastante limitadas, permitindo apenas a interação básica esperada. Por exemplo, se o programador quiser interagir com o utilizador através uma caixa de texto tipo *MessageBox* ou *Toast*, apenas é possível mostrar o texto pretendido e um botão de confirmação e cancelamento. É possível detetar se o utilizador decidiu confirmar ou cancelar, contudo não é possível inserir uma lista com escolhas, entre outras opções mais elaboradas.

Assim, de forma a contornar as limitações existentes, incluindo algumas funcionalidades que mesmo nativamente apenas estão disponíveis para dois dos três sistemas operativos móveis suportados, utilizaram-se extensões *NuGet* (que disponibilizam uma série de *API's),* compatíveis com *Xamarin.Forms.* Muitas dessas extensões têm uma vasta lista de dependências que acabam por influenciar outras extensões previamente instaladas, bem como a sua escolha e versão. Isto exigiu um grande esforço extra e experimentação, de forma a conseguir atingir o equilíbrio pretendido.

A Tabela 8 representa as principais e mais relevantes bibliotecas / *API* / extensões utilizadas no desenvolvimento do protótipo:

Tabela 8 — Biblioteca / *API* / extensões utilizadas na aplicação móvel *MediaPOS Mobile*

Nome e Versão	Android	iOS	WinPhone	Descrição
Acr.Support 2.1.0	Sim	Sim	Não	Funções e utilidades que permitem gerir serviços de *backend* do *Android e iOS*
Acr.UserDialogs 5.2.2	Sim	Sim	Sim	Permite utilizar caixas de diálogo com escolha múltipla, data, informações, progresso, entre outros
Acr.XamForms.UserDialogs 5.0.0	Sim	Sim	Sim	Informações dos dispositivos, caixas de diálogo e definições multiplataforma
AndHUD 1.2.0	Sim	Não	Não	Caixas de diálogo, progresso, entre outros, para *Android*
BTProgressHUD 1.2.0.3	Não	Sim	Não	Barras de progresso

Nome e Versão	Android	iOS	WinPhone	Descrição
Coding4Fun.Toolkit.Controls 2.1.8	Não	Não	Sim	Diversos elementos e controlos para aplicações *Windows Phone*
Newtonsoft.JSON 9.0.1	Sim	Sim	Sim	Ferramenta de alto desempenho para ler, fazer *serialize* e *deserialize* de texto *JSON*
RestSharp 105.2.3	Sim	Sim	Sim	Cliente *HTTP* que permite consumir *Web Services*, nos três sistemas operativos móveis
WPtoolkit 4.2013.8.16	Não	Não	Sim	Facilita a programação para *Windows Phone* disponibilizando vários controlos, métodos, animações, entre outros
WriteableBitmapEx 1.5.0	Não	Não	Sim	Manipulação de *Bitmap* para *Windows Phone*
Xam.Plugin.Connectivity 2.2.13	Sim	Sim	Sim	Informações do estado da rede, ligação à *Internet*, entre outros
Xam.Plugins.Forms.ImageCircle 1.4.1	Sim	Sim	Sim	Controlo personalizado que permite imagens redondas
Xam.Plugins.Settings 2.1.0	Sim	Sim	Sim	Definições multiplataforma
Xamarin.Forms 2.1.0.6529	Sim	Sim	Sim	*API's* para os três sistemas operativos móveis, permitindo partilhar código
ZXing.Net.Mobile 2.1.47	Sim	Sim	Sim	Livraria de código fonte aberto para leitura de códigos de barra
ZXing.Net.Mobile.Forms 2.1.47	Sim	Sim	Sim	Livraria de código fonte aberto para leitura de códigos de barra

Para além das bibliotecas / *API's* / extensões da Tabela 8, foram ainda utilizadas outras, das quais as previamente referidas dependem.

Para desenvolver o protótipo foi também utilizado:

- Computador *Windows* 10 *Education* 64 *bits*:
  - *Visual Studio Enterprise* 2015 *Update* 3;
  - *Xamarin* 4.1.1.3;
  - *Xamarin.Forms* 2.1.0.6529;
  - *Emulador* de *Android, iOS* e *Windows Phone*;

- Portátil *Mac OS* X 10.11 *El Capitan* 64 *bits*:
    - *Xcode* 7.3.1;
    - *Xamarin Studio* 6.0.1;
    - *Mono* 4.4.1.0;
    - *Emulador iOS* 9.2 e 8.4;
- *Smartphone Android 6.0.1;*
- *Smartphone iOS 10.0.2;*
- *Smartphone Windows Phone 10.1607.*

De modo a poder desenvolver para o *iPhone* foi necessário utilizar um portátil com o sistema operativo *Mac OS X*, devidamente configurado (*login* partilhado e possibilidade de realizar conexões *SSH).* Embora o projeto tenha sido desenvolvido em ambiente *Windows,* era necessário efetuar o *debug* no portátil *Mac OS X* que se ligava por *SSH* ao *Windows.*

4.2.1.2 Câmara Fotográfica

Uma das funcionalidades avançadas a ser implementada no caso de uso UC7 consiste na utilização da câmara fotográfica na leitura de códigos de barras. As câmaras fotográficas são, hoje em dia, um dos componentes base presentes em todos os *smartphones* e *tablets*, permitindo a utilização e acesso às mais diversas funcionalidades. Se no passado serviam apenas para capturar momentos através da aquisição de fotografias em formato digital ou vídeos, hoje em dia o paradigma de utilização é bastante mais alargado. Podem ser utilizadas para monitorização, identificação fácil da íris e realidade aumentada.

A sugestão de utilização da câmara fotográfica para este protótipo consiste na leitura de códigos de barra e/ou *QR Codes.* Trata-se de uma mais valia que faz sentido explorar, tendo a conta a importância dos mesmos nos pontos de venda. Os pontos de venda tradicionais utilizam "pistolas" de leitura de códigos de barras dedicadas apenas a essa funcionalidade. Através da câmara fotográfica dos dispositivos móveis é possível, sem recorrer a dispositivos externos ou gastos monetários, efetuar essa mesma leitura e reconhecimento.

Um dos cenários possíveis é a leitura do código de barras dos diversos produtos existentes nos clientes, procurando-os seguidamente na base de dados de forma a obter informações extra como o nome e preço do produto. Um outro cenário, igualmente implementado, consiste na leitura de um cartão criado através do Portal das Finanças / *e-fatura*, que contêm um código de barras e o *NIF* (Número de Identificação Fiscal) do cliente. Trata-se de uma função que permite o rápido acesso ao cliente, automatizando desta forma o processo, sem que o mesmo tenha que verbalizar o *NIF.* O cartão em questão tem o seguinte aspeto:

Figura 32 – Cartão *e-fatura*

O código de barras, ou em inglês *Barcode*, consiste numa representação gráfica de dados numéricos ou alfanuméricos de acordo com a fonte [91]. Existem dois tipos de código de barras, nomeadamente: os lineares ou de uma dimensão *(1D)*, que consistem em barras paralelas com espaçamento e largura variável e os códigos de barra bidimensionais *(2D)* constituídos por pontos, hexágonos, retângulos e outras figuras geométricas. Na Figura seguinte é possível identificar alguns exemplos de códigos de barras:

Figura 33 – Representação de um código de barras *1D* (à esquerda) e dois *2D* (*QR Code* e *Facebook*)

Os códigos de barra *2D* estão a ser cada vez mais utilizados, frequentemente associados a *Marketing* devido ao seu potencial. Por exemplo, é frequente ver-se códigos de barra *QR Code*, que após leitura permitem ao utilizador aceder a uma página *Web*, entre outras funcionalidades.

Contudo, nos sistemas de pontos de venda e comércio, em geral, os mais utilizados são os códigos de barra *1D*, existentes com vários padrões e formatos. Esses formatos variam de acordo com a área geográfica e comercial em que são utilizados, permitindo uma maior ou menor representação numérica. Para tal, é utilizada uma quantidade diversa de barras paralelas, com diferente largura e espaçamento. As características, de cada um dos padrões diferencia-os entre si, tendo em conta as regras da simbologia. A simbologia trata o modo como os dados são codificados.

No comércio Português, o padrão de código de barras *1D* mais comum é o *EAN-13 (European Article Number)*, sendo este padrão bastante utilizado a nível mundial, à exceção do Canadá e

Estados Unidos da América. Trata-se de uma representação que codifica treze números que contêm quatro componentes, como se pode observar na Figura 34:

- Origem do código de barras: primeiros 2 ou 3 dígitos;
- Empresa fabricante: seguintes 5, 6 ou 7 dígitos;
- Produto produzido: seguintes 3, 4 ou 5 dígitos
- Dígito de verificação: último dígito.

Figura 34 – Código de barras *1D EAN-13* [92]

Doze desses números referem-se ao produto em si, sendo que o número restante corresponde ao dígito de verificação (*check digit*, semelhante ao bit de paridade utilizado na informática).

A título de curiosidade, o código de Portugal é o 560, pelo que todos os produtos encontrados/fabricados em Portugal devem começar por 560.

O dígito de verificação é relativamente fácil de calcular. Vejamos o seguinte exemplo:

- Código de barras de uma garrafa de água da Luso: 560116306051

Tabela 9 — Cálculo dígito de verificação

Posição	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Valor 12 dígitos	5	6	0	1	1	6	3	0	6	0	5	1
EAN-13	1	3	1	3	1	3	1	3	1	3	1	3
Multiplicação dígito * EAN	5*1 =5	6*3 =18	0*1 = 0	1*3 = 3	1*1 = 1	6*3 =18	3*1 = 3	0*3 = 0	6*1 = 6	0*3 = 0	5*1 = 5	3*1 = 3

No padrão *EAN-13*, o número mais a direita começa por 3, seguido de 1 e assim sucessivamente, como se pode observar na Tabela 9. O valor é calculado através das diversas multiplicações e somas.

Neste caso, o resultado da soma é 62 (5+18+0+3+1+18+3+0+6+0+5+3). O dígito de verificação é obtido através da subtração do múltiplo de 10 mais próximo. Ou seja, 70-62 = 8, ou alternativamente: 62 % 10 = 2, sendo que 2 é o resto da divisão por 10. Posteriormente esse valor é subtraído a 10 e calculado o resto de 10 novamente: ((10-2) % 10) = 8

De seguida, existe uma codificação (ou descodificação no caso da leitura de um código de barras), com regras bastante específicas dependendo do padrão utilizado, aplicando codificação binária.

Pode-se, assim, concluir que cada padrão tem as suas regras específicas, sendo que o *EAN-13* não é o único utilizado em Portugal. A criação de um algoritmo de raíz capaz de ler a maioria dos códigos existentes exigirá um grande esforço.

Nesse sentido, após alguma pesquisa, encontrou-se uma solução compatível com o *Xamarin.Forms* e os três sistemas operativos móveis em questão. Essa solução (*ZXing.Net.Mobile*) consiste numa implementação para ambientes móveis, escrita em *C#,* baseada numa biblioteca de código fonte aberto (*ZXing Zebra Crossing)*, que permite o processamento de imagem de códigos de barra *1D* e *2D*. Embora a biblioteca de origem suporte um leque mais alargado de formatos de código de barras, a sua implementação em *C#* encontra-se ligeiramente mais limitada, porém continua a ser uma mais valia para o projeto, sendo capaz de dar uma resposta às necessidades na realidade Portuguesa. Os seguintes formatos são suportados pela mesma: *Aztec , Code 128, Code 39, Code 93, EAN13, EAN8, PDF417, QR Code, UPC-E.* [93]

4.2.1.3 Comunicação com o *Web Service*

A comunicação com *Web Services* é uma das principais funcionalidades existentes nas aplicações móveis disponibilizadas. A solução criada para os *smartphone*s não inclui uma base de dados local alojada em cada um dos dispositivos, pelo que a comunicação para consultar e enviar dados é conseguida através da comunicação com o *Web Service* disponibilizado. Não seria prudente comunicar diretamente com uma base de dados numa rede local ou através da *Internet*, tendo em conta que o envio dos dados de autenticação seria feito de forma pouco segura.

Para além da comunicação com a base de dados, a aplicação *MediaPOS Mobile* também comunica com outros *Web Services* disponibilizados por terceiros, nomeadamente o acesso à plataforma *MEO Wallet.*

A comunicação com periféricos é feita através do *Web Service.* Tal como referido previamente, este esta encarregue de fazer a gestão dos periféricos e comunicação, através das *API's* disponíveis em *C#* e *Mono.*

Nesse sentido, e de forma a suportar os mais diversos tipos de *Web Services*, formatos de texto, formas de comunicação e autenticação, foi utilizado uma biblioteca de código fonte aberto, escrita em *C#*, chamada *RestSharp.*

O *RestSharp* suporta o *Xamarin.Forms* e os seus diversos sistemas operativos, nomeadamente o *Android, iOS* e *Windows Phone*, assim como métodos assíncronos, bem como fornece ao programador uma série de métodos que são uma mais valia.

4.2.1.4 MEO Wallet

O pagamento através de dispositivos móveis é uma das principais funcionalidades avançadas que a *CimSoft* pretende implementar nos clientes. Trata-se de um *Web Service REST* que gere a complexidade de efetuar pagamentos através de diversos métodos. Ou seja, ao contrário de outras soluções, o *MEO Wallet* permite vários métodos de pagamento, como por exemplo através de referência multibanco, cartão *Visa* ou *MasterCard*, ou saldo *MEO Wallet* do cliente.

A implementação deste *Web Service* requer o registo da empresa na página *Web* do *MEO Wallet*, fornecendo os dados necessários para validar e confirmar a autenticidade do utilizador. Apenas após este registo é possível consultar a chave da *API (MERCHANT-API-KEY)* através do *backoffice*.

De forma a facilitar a implementação do serviço, foi utilizado o ambiente de testes providenciado (*sandbox).* Trata-se de um servidor que existe paralelamente ao servidor de produção, contudo é de, pelo que não lida com dinheiro real. São inclusivamente fornecidos dois cartões de crédito de teste. Disponibilizam ainda aplicações cliente que operam no ambiente de testes.

Todas as operações entre a aplicação móvel *MediaPOS Mobile* e o *Web Service MEO Wallet* incluem um cabeçalho de autorização/ autenticação, no qual é incluída a chave da *API.* Desse modo, todas as movimentações ficam associadas a uma conta de comerciante, permitindo posterior consulta através da página de *backoffice.*

Através da página para programadores, é possível consultar todas as particularidades e instruções de implementação e utilização da *API.* Existem dois tipos de recursos que é possível manipular: os *checkouts* e as operações.

Através da criação de um *checkout* é possível efetuar um pedido de pagamento, fornecendo (não obrigatoriamente) alguns dados como o montante, uma lista de artigos, entre outros. Embora a *API* refira a existência de dois métodos associados a esta operação, sendo que um permite obter o estado do pagamento e o outro permite o cancelamento do pedido, não foi possível implementar nenhum deles. Provavelmente tratam-se de métodos antigos que já não existem e a página *Web* poderá estar desatualizada.

As operações permitem obter dados mais detalhados de transações prévias, sendo possível listar todas as decorridas num determinado espaço de tempo ou uma operação em específico. Através das operações é possível efetuar um pedido de reembolso, quando aplicável, de movimentos previamente realizados.

4.2.2 Raspberry Pi ou Computador

Das diversas abordagens possíveis, a escolhida passa pela utilização de um dispositivo *Raspberry Pi* ou, em alternativa, os computadores normais que os clientes da *CimSoft* já possuem.

Os vários cenários disponíveis para utilização da aplicação são:

- Uma solução de baixo custo, em que o *Raspberry Pi* em conjunto com um dispositivo móvel (*smartphone* ou *tablet*) formam a totalidade do sistema. O *Raspberry Pi* seria o responsável pelo alojamento da base de dados, disponibilização dos serviços, assim como a interação com dispositivos (impressoras, balanças entre outros), tendo em conta as limitações do *Xamarin*, dispositivos móveis e soluções que abranjam a totalidade dos sistemas operativos móveis. Neste cenário os dispositivos móveis assumem o papel principal de interação com o utilizador, sendo assim o único sistema que permite recolher os dados e faturar;
- Em alternativa, podem-se utilizar os computadores já existentes nos clientes como meio de aceder a dispositivos tais como impressoras, de forma a emitir um recibo ou fatura. Nesta situação os atuais clientes mantêm a estrutura já existente, nomeadamente o alojamento da base de dados e utilização do *MediaPOS* para clientes que tenham computadores. A diferença seria o papel assumido pelos dispositivos móveis e a utilização da aplicação *MediaPOS Mobile:* estes assumiriam um papel secundário, que confere a mobilidade inexistente na solução que já possuíam. Por exemplo, poderiam ser utilizados para recolher dados diretamente numa mesa, no caso de o negócio ser um restaurante, por exemplo.

A solução desenvolvida suporta ambos os cenários, podendo ser disponibilizados de forma local ou utilizando a *Internet*.

Nesse sentido, e tendo em conta que alguns clientes têm interesse em consultar os dados do negócio (tal como o volume de vendas, estatísticas, entre outros) de forma remota, foi utilizada uma panóplia de tecnologias seguidamente descritas.

4.2.2.1 NO-IP

Embora a grande maioria dos clientes atuais da *CimSoft*, disponham de um serviço de *Internet*, não possuem um endereço de *IP* estático. Se por um lado a disponibilização do *Web Service* na rede local não é um problema, visto que é possível atribuir um endereço de *IP* estático, por outro lado quando se trata da disponibilização à distância através da *Internet* existe alguma dificuldade. Os endereços de *IP* dinâmicos são uma das dificuldades de utilização de *Web Services* pela *Internet.*

Nesse sentido, a solução proposta e utilizada passa pelo uso do *NO-IP.* O principio básico de funcionamento consiste na associação de um *IP* dinâmico a um *hostname* estático. Ou seja, o serviço *NO-IP* verifica, com a frequência que o utilizar desejar, as alterações de *IP* e regista. Posteriormente, esse novo *IP* dinâmico é associado a um nome de domínio (por exemplo: www.brunoandrade.noip.me), que nunca sofre alterações.

Este serviço é uma aplicação cliente e esta disponível para os sistemas operativos *Windows, Mac, Linux*¸ assim como *Raspberry Pi,* sendo que a configuração é bastante simples e não requer interações posteriores.

As principais vantagens desta abordagem são a possibilidade de aceder a um computador ou *Raspberry Pi* em qualquer altura ou local, independentemente de existir um *IP* estático ou não, bem como se torna mais simples relembrar um nome de domínio, em vez de um endereço de *IP.*

4.2.2.2 Mono

Como tem vindo a ser politica da empresa, o *C#* é a linguagem de programação de eleição. Esta linguagem era até há bem pouco tempo restrita a sistemas operativos *Windows*, sendo que recentemente a *Microsoft* tornou o seu *compilador* num projeto de código fonte aberto. Embora isto venha a possibilitar o aparecimento e melhoria de outros *compiladores* de *C#*, a *CimSoft* conta já com alguns anos de experiência na utilização do *Mono.*

O *Mono* é um projeto de código fonte aberto criado pela atual empresa *Xamarin*, recentemente adquirida pela *Microsoft,* que disponibiliza um conjunto de ferramentas, nomeadamente um *compilador* alternativo para *C#* e o *CLR (Common Language Runtime).*

Na prática isto traduz-se na possibilidade da execução de programas desenvolvidos em *C#,* em sistemas operativos concorrentes como por exemplo o *Linux*.

A aplicabilidade do *Mono* versão 4*,* nesta solução em particular, consiste na sua utilização em clientes que utilizem o *Linux* como sistema operativo do computador ou utilizem o *Raspbian* (sistema operativo para o *Raspberry Pi,* baseado em Linux) no *Raspberry Pi*. Deste modo, o *Web Service* e outras aplicações desenvolvidas em *C#* podem ser executados. Caso os clientes utilizem o sistema operativo *Windows*, o *Mono* não é necessário.

4.2.2.3 XSP ou IIS

O *XSP* (versão 4.0.5) é um servidor *Web* escrito em *C#*, bastante leve e básico, ideal para provas de conceito ou pequenos projetos, visto que é possível observar informações de *debug* na janela de execução. Este servidor funciona sobre compiladores em tempo real do *Mono* e *.NET Framework,* permitindo, assim, que o sistema operativo *Linux* ou distribuições baseadas no mesmo hospedem/executem aplicações *Web* escritas em *ASP.NET*

IIS (Internet Information Services), à semelhança do anterior, é igualmente um servidor *Web* que permite interpretar páginas *Web*, hospedar páginas *ASP.NET*, entre outros. Trata-se de um servidor criado pela *Microsoft*, disponível de origem nos seus sistemas operativos (após ativar o mesmo).

A utilização dos mesmos depende do sistema operativo utilizado, contudo ambos têm o mesmo fundamento nesta solução: disponibilizar o *Web Service.*

Em alternativa, num cenário futuro, poder-se-ia utilizar o servidor *Web Apache* em conjunto com o extra *mod_mono*, tendo em conta as funcionalidades extra e vantagens que este apresenta, embora seja mais complexo.

4.2.2.4 Web Service

O *Web Service* é a principal ferramenta presente na solução no *Raspberry Pi* ou computador, estando responsável pela comunicação e integração do sistema. No fundo, desempenha um papel de servidor, estando à espera de pedidos feitos pelas aplicações móveis, processando-os posteriormente e fornecendo uma resposta, caso necessário.

Através da utilização das soluções descritas nas últimas subsecções, nomeadamente o serviço *NO-IP*, *Mono* e um dos servidores *Web*, é possível expor o *Web Service* numa rede local ou *Internet.* Alojado num *Raspberry Pi*/computador, a aplicação cliente (neste caso o *MediaPOS Mobile)* pode ser acedida através do endereço de *IP* e respetiva porta do serviço, ou alternativamente ao nome do domínio e porta.

O tipo de *Web Service* utilizado nesta solução foi o *WCF (Windows Communication Foundation) RESTful.*

Ao contrário dos cenários mais comuns, em vez da utilização do protocolo *SOAP (Simple Object Access Protocol)* será utilizado a arquitetura REST (Representational State Transfer). Algumas das vantagens na utilização de *REST* face ao *SOAP* são a disponibilização de formatos de texto como *JSON*, *XML*, assim como a escalabilidade, eficiência e interoperabilidade.

A comunicação dos dados é feita sobre o protocolo *HTTP (Hypertext Transfer Protocol)* utilizando os diversos métodos disponíveis:

- GET: lê a representação de um recurso específico;
- PUT: atualiza um recurso específico com base nos dados fornecidos;
- DELETE: apaga um recurso específico;
- POST: cria um recurso específico com base nos dados fornecidos;

4.2.2.5 PostgreSQL

O *SGBD* (Sistema Gestor de Base de Dados) utilizado nesta solução chama-se *PostgreSQL.* Trata-se de um projeto de código fonte aberto, que disponibiliza bases de dados relacionais.

A escolha da base de dados *PostgreSQL,* versão 9.4*,* teve por base o facto de a *CimSoft* utilizar *PostgreSQL* em todos os seus produtos comerciais, dispondo de um vasto conhecimento e ferramentas.

A comunicação com a base de dados é feita através do *Web Service* criado para o efeito.

4.3 Implementação

Na secção da implementação são documentados alguns aspetos e exemplos, com base nas funcionalidades e requisitos. Utilizando as tecnologias previamente explicadas, é ainda apresentado o resultado desta implementação, através de diversas Figuras dos menus e ecrãs, assim como o código utilizado.

4.3.1 Aplicação Móvel *MediaPOS Mobile*

Nesta subsecção sobre a implementação da aplicação móvel, é apresentado o resultado final da aplicação assim como as diversas funcionalidades criadas. Seguidamente é exemplificado a metodologia utilizada para tal, através de código exemplificativo da interface gráfica, utilização da câmara fotográfica e a comunicação com o *Web Service* e os pagamentos móveis.

4.3.1.1 Funcionalidades e Aspeto Gráfico

A aplicação móvel *MediaPOS Mobile* apresenta-se como uma aplicação simples e intuitiva. No fundo trata-se de uma prova de conceito, que embora não disponha de um alargado leque de funcionalidades, possui as condições necessárias para que novas funcionalidades sejam criadas.

A interface gráfica criada é comum aos três sistemas operativos móveis e, recorrendo aos elementos nativos de cada um, adapta-se aos diferentes tamanhos de ecrã e assim diversos dispositivos. Mesmo perante as limitações existentes na partilha de código entre os vários sistemas operativos móveis, tentou-se seguir as indicações de desenho e usabilidade de aplicações móveis.

A rapidez de resposta depende do volume de dados a consultar, assim como o tamanho da base de dados em si. Existem clientes com bases de dados com cem produtos, bem como existem outros com mil produtos. Tal fator tem um peso no desempenho de determinadas funcionalidades. O *Xamarin.Forms,* embora apresente um desempenho bastante aceitável, continua a não ser equiparável ao desenvolvimento nativo em cada um dos sistemas operativos em questão. Isso vê-se através do tempo de abertura da própria aplicação e da otimização de determinadas bibliotecas.

A navegação entre os vários ecrãs é feita através de *NavigationPages*, apresentando uma seta no canto superior esquerdo que permite retroceder a qualquer altura. Existe sempre um titulo

na barra superior que permite ao utilizador perceber em que menu se encontra, assim como um menu de contexto que permite o acesso a opções especificas.

A aplicação apresenta as seguintes funcionalidades e ecrãs.

- Ecrã principal: Representado pela Figura 35. Disponibiliza uma lista com as diversas famílias de produtos, assim como os produtos e preço. Para os escolher é apenas necessário pressionar sobre o artigo desejado. Na barra de título existe ainda o acesso ao Carrinho de Compras (ilustrado por um carrinho) e um menu de contexto.

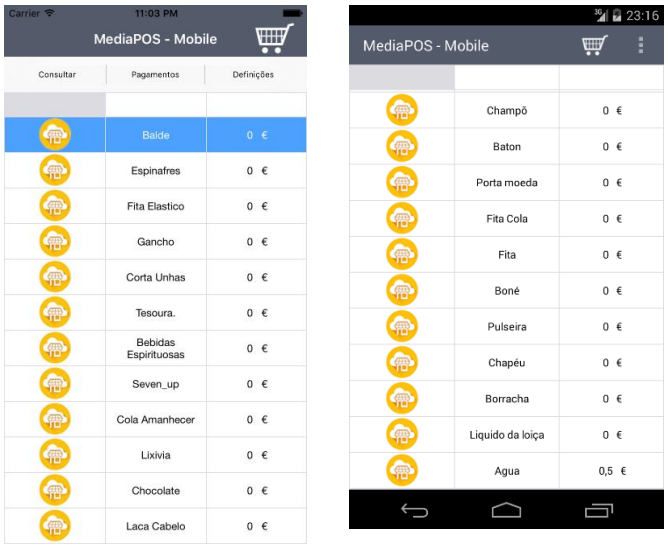

Figura 35 – Ecrã principal

- Carrinho de compras: Permite a consulta dos artigos adicionados à lista de compras, o preço total e o acesso ao menu de Pagamento, tal como se pode ver na Figura 36:

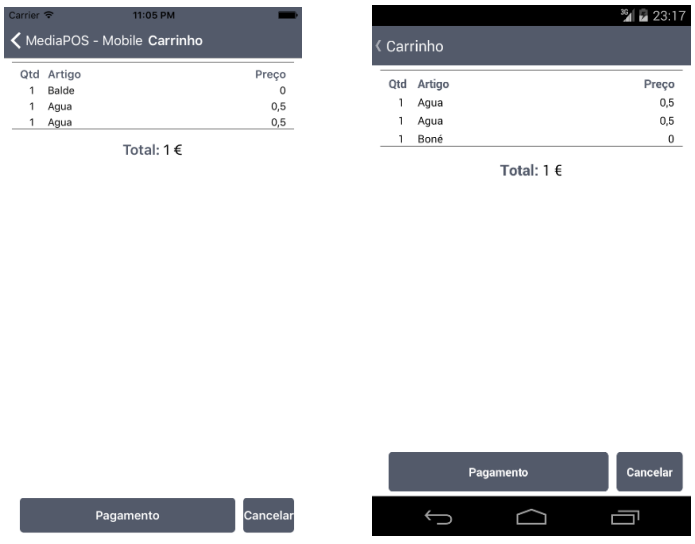

Figura 36 – Ecrã com o Carrinho de Compras

Dentro do carrinho de compras existe ainda o Menu de Pagamento que por sua vez alberga os Pagamentos por MEO Wallet:

- Menu de Pagamento: Permite adicionar o nome do cliente à fatura, ler o *NIF* (Número de identificação fiscal) através das funcionalidades da câmara fotográfica, declarar que foi pago em dinheiro ou alternativamente emitir um pedido de pagamento por *MEO Wallet.* Neste menu é ainda possível efetuar um pedido ao *Web Service* de modo a imprimir uma fatura, como se poder observar na Figura 37:

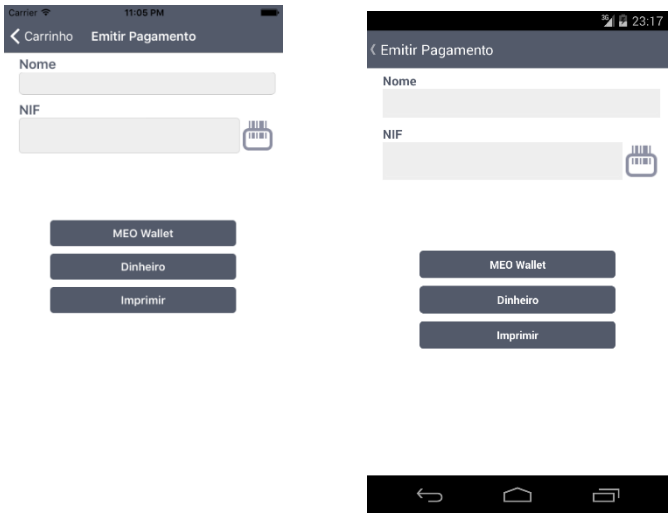

Figura 37 – Menu de Pagamento

- Pagamento por *MEO Wallet:* embora a *API* disponha de uma panóplia de funcionalidades, as que foram implementadas na totalidade, para este protótipo foram simplificadas. Ou seja, de modo a reduzir o número de interações com o utilizador não é apresentada a hipótese de colocar uma morada, nome, contacto de e-mail, entre outros, associado ao pedido de pagamento. A Figura 38 exemplifica o pagamento:

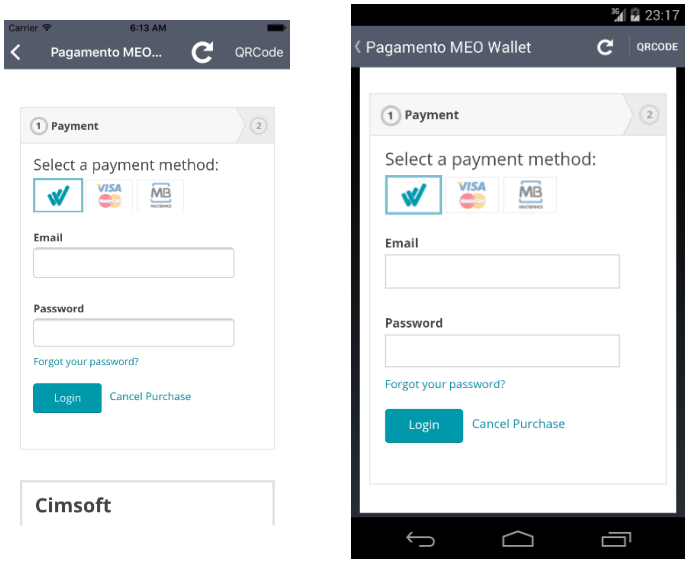

Figura 38 – Pagamento por *MEO Wallet*

No canto superior direito do ecrã principal é possível aceder ao Menu de Contexto. Este permite aceder às definições de configuração da aplicação, consultar faturas, assim como algumas operações relacionadas com pagamentos por *MEO Wallet.*

- Consultar: Neste menu (Figura 39) é possível consultar as diversas faturas previamente emitidas, recorrendo a vários critérios de seleção, tais como faturas diárias (do dia atual), faturas de uma data especifica ou compreendidas entre duas datas. É ainda possível consultar faturas com base na sua identificação única, bem como todas as faturas presentes na base de dados (embora seja um processo lento). Estas faturas são apresentadas através de uma tabela que apresenta diversos campos relevantes, tais como o *ID*, data de emissão, cliente, valor, entre outros. Se o utilizador pressionar prolongadamente a tabela são apresentadas diversas funcionalidades, como por exemplo ordenar por data, agrupar por funcionário, agrupar por dia, entre outras opções, assim como é possível consultar todos os detalhes de uma fatura em específico.

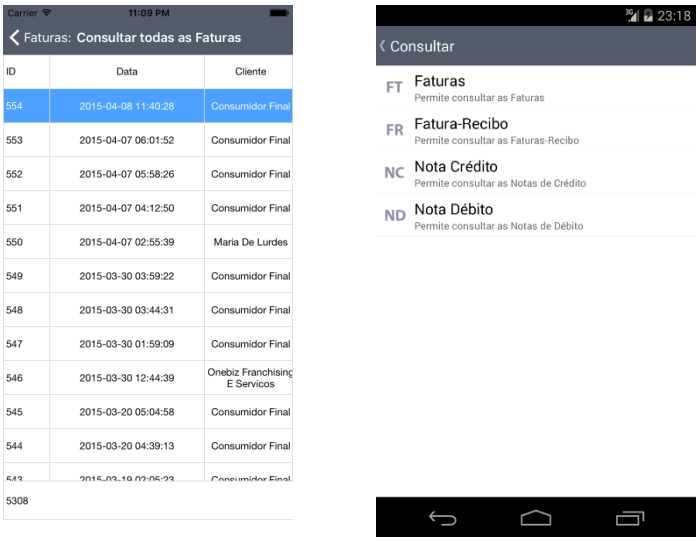

Figura 39 – Menu de Consultar Faturas

- Pagamentos: Permite consultar os pagamentos efetuados por *MEO Wallet* com base em vários critérios, como por exemplo o dia atual, pagamentos de um dia específico ou determinado espaço de tempo. Esses dados são apresentados numa tabela que, tal como o menu Consultar, permite agrupar com base nos mais diversos critérios, consultar detalhes. É igualmente possível efetuar um pedido de reembolso de um pagamento previamente efetuado. A Figura 40 exemplifica as várias interações:

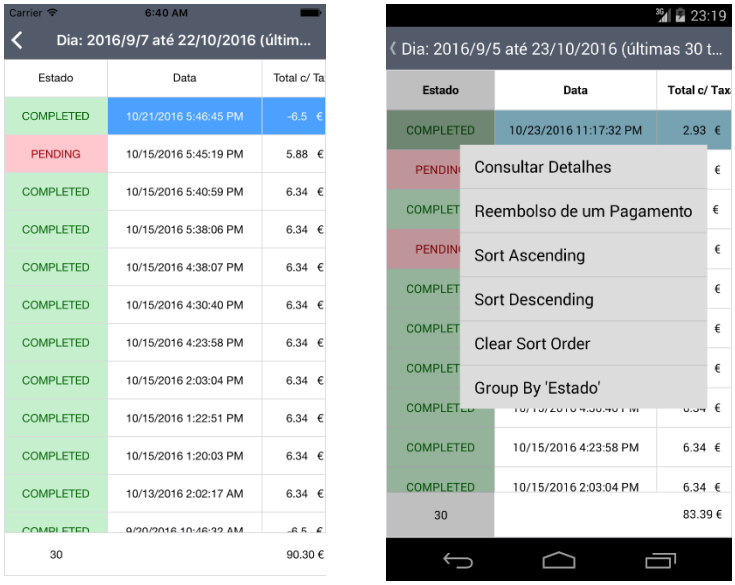

Figura 40 – Consultar Pagamentos

- Definições: Permite configurar os dados de conectividade e acesso ao pagamento por *MEO Wallet:*
    - Conectividade: A Figura 41 demonstra a introdução dos dados de acesso ao *Web Service* / Ligação direta à Base de Dados (embora esta opção esteja presente, no futuro será desativada por questões de segurança). Esses dados incluem o endereço do *Web Service*, a porta do serviço, entre outros. É ainda disponibilizado um menu de contexto no canto superior direito que permite testar a ligação ao servidor ou verificar a disponibilidade do *Wi-Fi* ou dados móveis, assim como limpar os dados de acesso ao servidor.

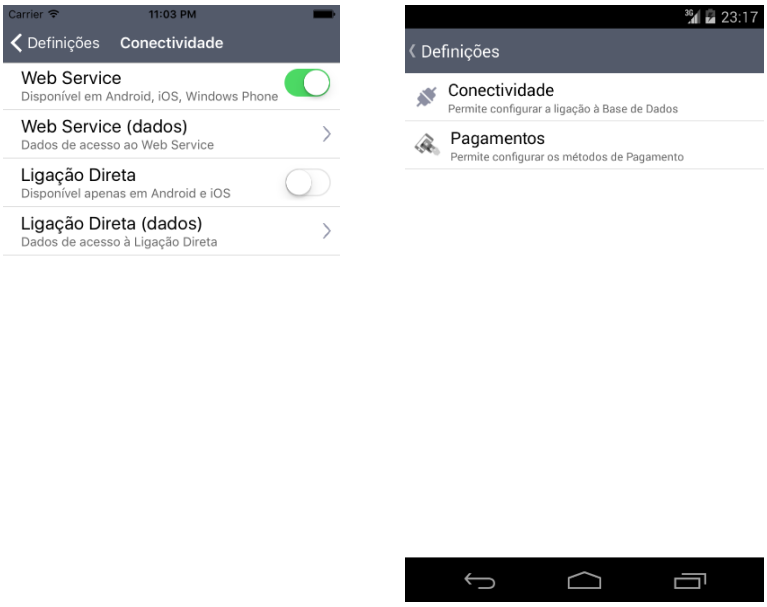

Figura 41 – Definições de Conectividade

- Pagamentos por MEO Wallet: permite introduzir a chave privada de acesso à *API* (que identifica o utilizador e todos os movimentos associados) e ainda escolher o ambiente de testes (*sandbox,* utilizado durante a fase de desenvolvimento) ou o ambiente real (de produção, usado para movimentar dinheiro real). A Figura 42 demonstram o menu das definições do *MEO Wallet:*

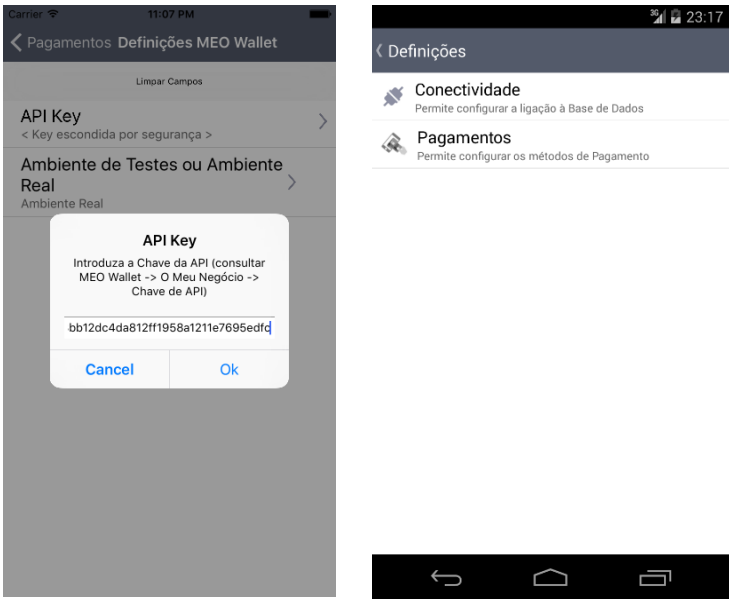

Figura 42 – Definições de Pagamentos *MEO Wallet*

4.3.1.2 Xamarin e as Aplicações Móveis

Ao utilizar *Xamarin Forms Shared Project* perde-se o acesso ao desenho gráfico das aplicações através de *XAML*, havendo, contudo, outras vantagens previamente descritas. O ambiente gráfico das aplicações desenvolvidas foi totalmente programado através de código. Trata-se de um processo altamente desgastante e consumidor de código. A título de exemplo, uma das páginas com detalhes de compras utiliza mais de mil linhas de código para uma simples página com alguns campos.

De forma a exemplificar o processo, na Figura 43 está representado um excerto do código utilizado para uma simples *ListView* com um título, uma descrição e um *Switch:*

```
var titulo_webService = new Label()
{
    Style = AppStyle.LabelListViewCellTitulo,
    Text = "Web Service",
};

var descricao_webService = new Label()
{
    Style = AppStyle.LabelListViewCellDescricao,
    Text = "Disponível em Android, iOS, Windows Phone",
};

switcher_webservice = new Switch
{
    HorizontalOptions = LayoutOptions.EndAndExpand,
    VerticalOptions = LayoutOptions.Center,
    IsToggled = true // Ligado por defeito
};
switcher_webservice.Toggled += switcher_webservice_Toggled;
```

Figura 43 – Excerto de código com duas *Label* e um *Switch*

Após a criação de duas *Label* e um *Switch* foi necessário indicar que ambas as *Label* se encontravam alinhas na vertical dentro de um *StackLayout*. Posteriormente, adicionou-se um *Switch* alinhado horizontalmente à direita, exatamente após a representação das *Labels.* Para tal, foi criado um novo *StackLayout* como se pode observar na Figura 44:

```
var titulo_descricao_webService = new StackLayout
{
    Padding = new Thickness(10, 5, 0, 0),
    Spacing = 0,
    HorizontalOptions = LayoutOptions.FillAndExpand,
    Orientation = StackOrientation.Vertical,
    Children =
    {
        titulo_webService,
        descricao_webService
    }
};

var layout_webservice = new StackLayout
{
    Spacing = 0,
    Padding = new Thickness(10, 0, 10, 0),
    Orientation = StackOrientation.Horizontal,
    HorizontalOptions = LayoutOptions.FillAndExpand,
    Children =
    {
        titulo_descricao_webService,
        switcher_webservice
    }
};
```

Figura 44 – Excerto de código que representa os dois *StackLayout*

Por fim, de forma ao utilizador da aplicação *MediaPOS Mobile* visualizar os elementos previamente descritos, é necessário adicionar ao conteúdo da página. Como se pode observar na Figura 45, é criado um último *StackLayout* que inclui o layout_webservice, entre outros.

```
            var layout = new StackLayout
            {
                BackgroundColor = Color.White,
                Children =
                {
                    layout_webservice,
                    new BoxView()
                    {
                        HeightRequest = 1,
                        BackgroundColor = Color.FromHex("#ddd"),
                    },
                    layout_webservice_dados,
                    new BoxView()
                    {
                        HeightRequest = 1,
                        BackgroundColor = Color.FromHex("#ddd"),
                    },
#if __MOBILE__
                    layout_ligDireta,
                    new BoxView()
                    {
                        HeightRequest = 1,
                        BackgroundColor = Color.FromHex("#ddd"),
                    },
                    layout_ligDireta_dados,
                    new BoxView()
                    {
                        HeightRequest = 1,
                        BackgroundColor = Color.FromHex("#ddd"),
                    },
#endif
                }
            };
            Content = layout;
```

Figura 45 – Excerto de código que representa o *StackLayout* final

Na Figura 45 é possível verificar a existência de uma das funcionalidades disponibilizadas pelo *Xamarin.Forms*: código condicional [94], permitindo que seja executado/compilado apenas nas plataformas desejadas.

- *#if __MOBILE__: Compilado* apenas em *Android* e *iOS*;
- *#if __IOS__: Compilado* apenas em *iOS*;
- *#if __ANDROID__: Compilado* apenas em *Android*;
- *#if __TVOS__: Compilado* apenas para televisores;
- *#if __WATCHOS__: Compilado* apenas para relógios;

Isto torna-se particularmente útil, tendo em conta que as *API's* disponíveis para *Windows Phone* são bastante mais limitadas, sendo, por vezes, necessário criar métodos ou partes do código, alternativas apenas para o *Windows Phone*.

A representação gráfica das linhas de código previamente descritas pode ser visualizada na Figura 46 (caixa verde):

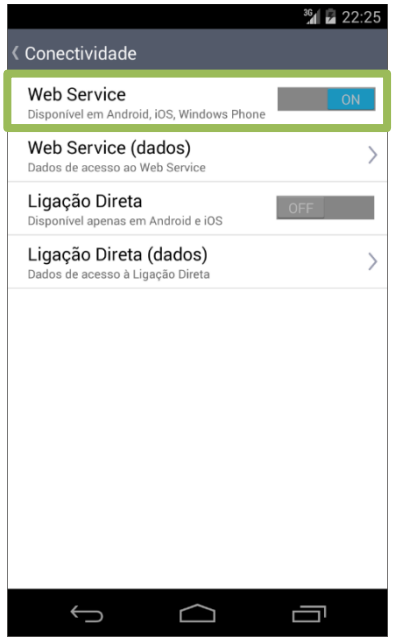

Figura 46 – *Menu* das Definições de Conectividade

4.3.1.3 Câmara Fotográfica

Através da câmara fotográfica é capturada uma imagem, que depois é processada pela *API*, fazendo a extração dos mais diversos tipos de dados, tais como o valor do código de barras e o formato, por exemplo. A aplicabilidade principal no protótipo consiste apenas na leitura do código em si, sendo que para tal são apenas necessárias algumas linhas de código, de modo a iniciar a captura da imagem de forma assíncrona, tal como se pode observar na Figura 47:

```
var tapGestureRecognizer = new TapGestureRecognizer();
tapGestureRecognizer.Tapped += async (s, e) => {

    var scanner = new ZXing.Mobile.MobileBarcodeScanner();
    scanner.TopText = "Aponte para o Código de Barras do cartão que contém o NIF";
    scanner.CameraUnsupportedMessage = "Cam não suportada";

    var result = await scanner.Scan();

    if (result != null)
        entry_nif.Text = result.ToString();
};
image.GestureRecognizers.Add(tapGestureRecognizer);
```

Figura 47 – Enxerto de código de leitura de código de barras

4.3.1.4 Comunicação com o *Web Services*

Na Figura 48 é possível observar a utilização do cliente *RestSharp* na comunicação com o *Web Service* do *MEO Wallet:*

```
public async Task<Operation> GetOperation(Guid id)
{
    RestClient client = new RestClient(api_link);
    client.AddDefaultHeader("Content-Type", "application/json");
    RestRequest request;

    request = new RestRequest("/api/v2/operations/" + id, Method.GET);
    request.AddHeader("Authorization", "WalletPT " + api_key);

    var response = await client.FixedExecuteTaskAsync(request).ConfigureAwait(false);

    if (response.StatusCode != HttpStatusCode.OK)
    {
        var a = JsonConvert.DeserializeObject<Error>(response.Content);
        throw new MeoWalletAPIException(a);
    }

    var resultado = JsonConvert.DeserializeObject<Operation>(response.Content);
    return resultado;
}
```

Figura 48 – Enxerto de código de comunicação com o *Web Service* do *MEO Wallet*

Como é possível observar, é instanciado um *RestClient* que comunica com o endereço do *Web Service* do *MEO Wallet,* assim como um pedido *RestRequest*, no qual é feito a autenticação do utilizador através do envio da *API Key.* Uma vez efetuado o pedido de forma assíncrona é obtida uma resposta que, neste caso em específico, tem um formato conhecido (*Operation)*, sendo posteriormente feito um *deserialize* da mesma.

A comunicação com os periféricos por parte do *Web Service* assemelha-se aos exemplos anteriores. Trata-se de um pedido *Web* que despoleta uma série de ações no *Raspberry Pi* ou computador.

4.3.1.5 Pagamentos por *MEO Wallet*

Através da Figura 49 é possível ver um excerto de código que representa um pedido de reembolso. Para tal, é necessário fornecer o *ID* da transação específica que se pretende reembolsar, de modo a efetuar um pedido para o recurso *"/api/v2/operations/id/refund"*, sendo o valor do *id* substituído pelo fornecido.

```
public async Task<Operation> RefundOperation(Guid id, RefundRequest refundRequest = null)
{
    request = new RestRequest("/api/v2/operations/" + id + "/refund", Method.POST);
    request.AddHeader("Authorization", "WalletPT " + api_key);

    request.AddJsonBody(refundRequest);

    var response = await client.FixedExecuteTaskAsync(request).ConfigureAwait(false);

    if (response.StatusCode != HttpStatusCode.OK)
    {
        var a = JsonConvert.DeserializeObject<Error>(response.Content);
        throw new MeoWalletAPIException(a);
    }

    var resultado = JsonConvert.DeserializeObject<Operation>(response.Content);
    return resultado;
}
```

Figura 49 – Enxerto de código do pedido de reembolso com o *Web Service MEO Wallet*

Sempre que é efetuada uma comunicação, é necessário enviar a chave da *API*, como se pode observar na segunda linha de código do método (*Authorization*). Através da classe *RefundRequest* é possível indicar se se pretende um reembolso total ou parcial, entre outros campos não obrigatórios.

Cada pedido origina sempre uma resposta por parte do *Web Service MEO Wallet*, sendo que por vezes essas respostas podem demorar um pouco mais. De maneira a tornar a aplicação fluída e evitar bloqueios, foram utilizados métodos assíncronos.

A resposta recebida, assim como todas as comunicações com o *Web Service,* utilizam o formato de texto *JSON,* sendo desse modo necessário efetuar um *deserialize* da resposta. Caso a resposta não seja a desejada, é possível efetuar um controlo de erros com base no código dos mesmos e mensagem/ motivo.

Quando se cria um pedido para um pagamento, é recebida uma confirmação por parte do *Web Service* que permite visualizar uma página de pagamento. Através dessa página é possível consultar um *QR Code* que permite que os clientes finais efetuem o pagamento através dos seus dispositivos móveis, utilizando a aplicação oficial *MEO Wallet,* tal como se pode observar na Figura 50:

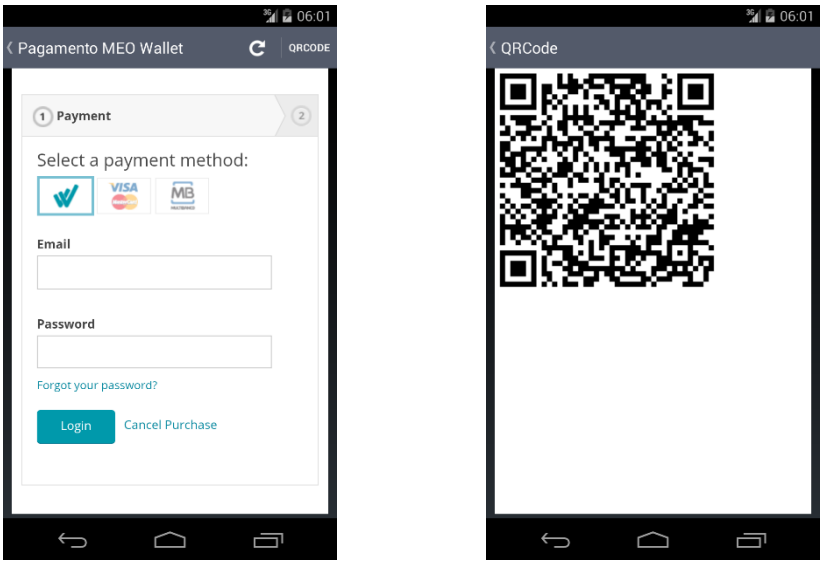

Figura 50 – Ecrã de pagamento através do *MEO Wallet*

4.3.2 Raspberry Pi ou Computador

A implementação das funcionalidades do *Raspberry Pi* ou Computador foi constituída pela criação do *Web Service*, utilizando a tecnologia previamente descrita, assim como a criação e comunicação com a base de dados.

4.3.2.1 Web Service

A adição de funcionalidades nesta camada acompanhou os desenvolvimentos efetuados nas aplicações móveis, permitindo obter desde cedo versões demonstráveis. Nesse sentido, foram sendo criados os contratos para as mais diversas funcionalidades, estipulando o método de acesso, o formato da resposta, a localização do recurso, tal como se pode observar na Figura seguinte:

```
[OperationContract]
[WebInvoke(Method = "GET",
    ResponseFormat = WebMessageFormat.Json,
    BodyStyle = WebMessageBodyStyle.Wrapped,
    UriTemplate = "familiasExistentes/")]
1 reference
string FamiliasExistentes();
```

Figura 51 – Enxerto de código do *Web Service*

Neste caso definiu-se um método chamada *FamiliasExistentes*(), que retorna uma *string.* Essa resposta utiliza o formato *JSON* (*JavaScript Object Notation)*, sendo que este formato é o ideal para comunicar com aplicações móveis, visto que utiliza menos largura de banda [95]. O método utilizado é o *GET,* tendo em conta que apenas se pretende consultar dados.

A implementação do método *FamiliasExistentes()* consulta os dados existentes na base de dados *PostgreSQL,* através da camada de comunicação com a mesma. Essa camada é responsável pela configuração e autenticação na base de dados, nomeadamente o endereço de *IP,* porta, nome de utilizador e palavra-chave. Se os dados de acesso estiverem corretos, é criada uma ligação, executado a *query* pretendida e retornado o resultado.

```
public string FamiliasExistentes()
{
    WCF.REST.Data_Tier.AccessDB ligacao = new WCF.REST.Data_Tier.AccessDB();
    string sqlQuery = "SELECT * FROM FAMILIA_POS order by fam_pos_x,id_fam_pos";

    DataSet ds = ligacao.execute_dataset(sqlQuery);
    DataTable dt = ds.Tables[0];
    var myEnumerable = dt.AsEnumerable();

    List<TabelaFamilias> myClassList =
       (from item in myEnumerable
        select new TabelaFamilias
        {
            id_fam_pos = item.Field<int>("id_fam_pos").ToString(),
            descricao = item.Field<string>("descricao"),
            activa = item.Field<bool>("activa").ToString(),
            fam_pos_x = item.Field<int?>("fam_pos_x").ToString(),
            fam_pos_y = item.Field<int?>("fam_pos_y").ToString(),
            id_fam_pos_up = item.Field<decimal?>("id_fam_pos_up").ToString(),
            //oid = item.Field<Oid>("oid").ToString(),
        }).ToList();

    string jsonString = JsonConvert.SerializeObject(myClassList);
    return jsonString;
}
```

Figura 52 – Enxerto de código do método *FamiliasExistentes()*

Como se pode ver na Figura 52, a *query* de consulta das famílias de produtos existentes é executada e devolvida sobre a forma de um *DataSet*, sendo utilizada a primeira tabela do mesmo (*DataTable).* Como se trata de uma estrutura de dados conhecida é criada uma Lista da classe TabelaFamilias, representada na Figura 53:

```
public class TabelaFamilias
{
    1 reference
    public string id_fam_pos { get; set; }
    1 reference
    public string descricao { get; set; }
    1 reference
    public string activa { get; set; }
    1 reference
    public string fam_pos_x { get; set; }
    1 reference
    public string fam_pos_y { get; set; }
    1 reference
    public string id_fam_pos_up { get; set; }
    0 references
    public string oid { get; set; }
}
```

Figura 53 – Enxerto de código da classe TabelaFamilias

Seguidamente é utilizada a biblioteca *Newtonsoft JSON.NET,* de forma a disponibilizar os dados no formato *JSON,* através do método *JsonConvert.SerializeObject().*

O resultado disponível de modo a ser consultado pelas aplicações móveis esta apresentado na Figura 54:


```
{\"id_fam_pos\":\"56\",\"descricao\":\"Laticinios.\",\"activa\":\"True\",\"fam_pos_x\":\"\",\"fam_pos_y\":\"\",\"id_fam_pos_up\":\"\",\"oid\":null},
{\"id_fam_pos\":\"57\",\"descricao\":\"Congelados.\",\"activa\":\"True\",\"fam_pos_x\":\"\",\"fam_pos_y\":\"\",\"id_fam_pos_up\":\"\",\"oid\":null},
{\"id_fam_pos\":\"58\",\"descricao\":\"Detergentes.\",\"activa\":\"True\",\"fam_pos_x\":\"\",\"fam_pos_y\":\"\",\"id_fam_pos_up\":\"\",\"oid\":null},
```


Figura 54 – Resultado *JSON* da *query* para listar as famílias existentes

4.3.2.2 Comunicação com periféricos

Através das diversas librarias disponíveis pelo *C#* e *Mono* é possível comunicar com diversos periféricos. Esta funcionalidade não foi explorada em pleno. A ideia seria verificar a possibilidade de comunicar com periféricos, através das funcionalidades mais básicas. A Figura 55 exemplifica o pedido de teste da impressora, através do *Web Service.* Para tal, é utilizado a impressora predefinida e feito um pedido de impressão:

```
public bool testeImpressao()
{
    string s = "string to print";

    PrintDocument p = new PrintDocument();
    p.PrintPage += delegate (object sender1, PrintPageEventArgs e1)
    {
        e1.Graphics.DrawString(s, new Font("Times New Roman", 12), new SolidBrush(Color.Black),
            new RectangleF(0, 0, p.DefaultPageSettings.PrintableArea.Width, p.DefaultPageSettings.PrintableArea.Height));

    };
    try
    {
        p.Print();
    }
    catch (Exception ex)
    {
        throw new Exception("Exception Occured While Printing", ex);
    }

    return true;
}
```

Figura 55 – Exemplo de impressão utilizando o *Web Service*

De igual forma, é possível comunicar com outros tipos de dispositivos. As balanças são um dos pontos de interesse. Estas ligam-se ao *Raspberry Pi* ou computador, e comunicam através da porta *USB.* Por vezes, é necessário utilizar um conversor de porta de série para *USB.* Normalmente, as balanças utilizadas nos pontos de venda necessitam que se envie um caracter específico de forma a efetuar um pedido de medição do peso. Posteriormente é enviado uma

resposta através das portas de série. Essa resposta pode posteriormente ser enviada para os dipositivos móveis utilizando o *Web Service.* A Figura 56 exemplifica a leitura de um peso numa balança.

```
string cmd_Send_data = "W";
serialPort1.WriteLine(cmd_Send_data);


bal_rx = serialPort1.ReadExisting();
```

Figura 56 – Exemplo de leitura de um peso na balança

4.3.2.3 PostgreSQL

De forma a garantir a interoperabilidade com outras soluções disponíveis pela *CimSoft* foi utilizado o mesmo esquema e *script* de criação de tabelas. Dado que este esquema é o mesmo utilizado na versão comercial do *MediaPOS* e contém informações de cariz sensível, não serão descritos todos os campos e tabelas. Utilizando o exemplo da subsecção 4.3.2.1, são criadas tabelas da seguinte forma (Figura 57):

```
CREATE TABLE public.familia_pos (
    id_fam_pos int4 NOT NULL DEFAULT nextval('familia_pos_id_fam_pos_seq'::regclass),
    descricao varchar(64) NOT NULL,
    activa bool NULL DEFAULT true,
    fam_pos_x int4 NULL,
    fam_pos_y int4 NULL,
    id_fam_pos_up numeric(11) NULL,
    "oid" oid NULL,
    CONSTRAINT familia_pos_key UNIQUE (descricao)
)
WITH (
    OIDS=FALSE
);

CREATE INDEX ind1 ON public.familia_pos (id_fam_pos);
```

Figura 57 – Excerto de código da criação da tabela *familia_pos*

Assim, é possível observar na Figura 58 algumas tabelas e as relações entre si:

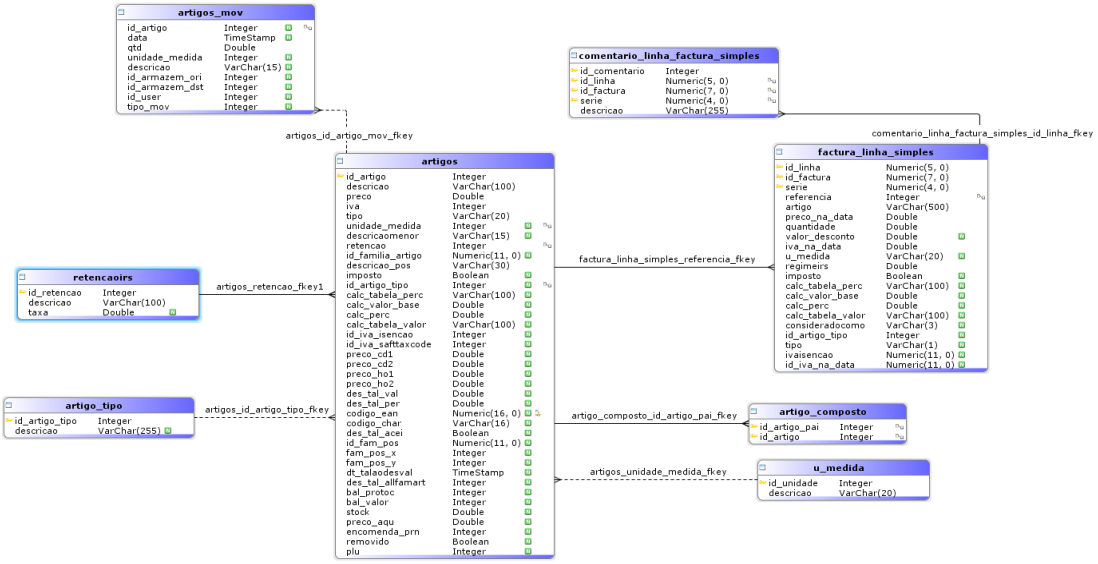

Figura 58 – Exemplo de algumas tabelas e das relações entre si

Como referido anteriormente, o acesso à base de dados não é feito diretamente por parte das aplicações móveis. Para tal, existe uma camada de acesso à base de dados, dentro do *Web Service*, responsável pela criação da ligação e consulta de dados, entre outros.

A biblioteca *Npgsql* é um componente utilizado no acesso e leitura de dados, permitindo que programas desenvolvidos em *C#* disponham de uma série de funcionalidades no acesso. Uma das vantagens desta biblioteca é a compatibilidade com o *Mono.*

4.4 Resultados

Nesta secção são apresentados os testes realizados, tendo em conta o plano de testes previamente definido. Pretende-se com isto demonstrar a implementação descrita na secção 4.3 e 4.2, quando submetida ao plano de testes apresentado na secção 4.1.

Devido ao teor da solução proposta e às imposições legais por parte da Autoridade Tribuária e Aduaneira, não é possível realizar testes de aceitação em clientes. Desde 1 de Janeiro de 2011 que é obrigatório, conforme a Portaria nº 363/2010 de 23/06 DR nº 120 – Serie I, utilizar apenas *software* de faturação devidamente certificado, obedecendo para tal a uma série de requisitos.

Nesse sentido foram realizados testes funcionais baseados na especificação, de forma a que as funcionalidades sejam testadas de acordo com os requisitos. Pretende-se com isto efetuar um

teste ao sistema já integrado, de modo a verificar os seus requisitos (funcionais e não funcionais), num ambiente de produção. A experiência será empírica, com base puramente na vivência da utilização e experimentação.

4.4.1 Condições de Teste

De modo a executar o plano de testes em questão, foi utilizado um sistema que simula um utilizador final, com dados reais de clientes (da *CimSoft)*. Com isto pretende-se verificar se o produto satisfaz os requisitos.

O material utilizado nos testes foi:

- *Smartphones:*
  - *Android 6.0.1:* Samsung Galaxy S3;
  - *iOS 10.0.2: iPhone 6s;*
  - *Windows Phone 10.1607: Microsoft Lumia 535*.
- Computador:
  - Impressora *USB Brother;*
  - Servidor *Web IIS* versão 1511;
  - *NO-IP* versão 4.1.1 (*brunoandrade.noip.me);*
  - *PostgreSQL* versão 9.4 com uma cópia de uma base de dados de um cliente.

4.4.2 Testes executados

Os testes executados estão descritos em três fases: primeiramente a arquitetura de modo a perceber os elementos englobados, seguidamente é descrito o procedimento e, por fim, os resultados obtidos, comparando com os resultados esperados. É ainda feita a referência caso o teste obtenha resultados diferentes em algum dos três sistemas operativos móveis.

Os testes foram os seguintes:

1. Introduzir parâmetros correta e incorretamente em tentativas seguidas e verificar os resultados obtidos em ambas.
   - Arquitetura: Este teste requer a utilização de três partes fundamentais, nomeadamente a aplicação móvel, o *Web Service* e a base de dados. A aplicação recolhe os parâmetros selecionados pelo utilizador e invoca o método específico no *Web Service.* Este, por sua vez, comunica com a base de dados de modo a fornecer a lista de faturas.
   - Procedimento: Abrir a aplicação *MediaPOS Mobile*, aceder ao menu de contexto no canto superior direito e escolher a opção Consultar. Seguidamente Escolher Faturas e um dos parâmetros de listagem (diárias, por data, etc). Repetir o teste para cada um dos parâmetros disponíveis.

- Resultado obtido: A aplicação móvel mostrou uma lista de faturas numa tabela para todos os parâmetros de listagem disponíveis. Como seria de esperar, a listagem de todas as faturas demorou um pouco mais visto ser mais exaustivo. O resultado obtido foi coincidente com o resultado esperado, comportando-se da mesma forma nos três sistemas operativos móveis.

2. Introduzir a identificação de uma fatura existente e de uma inexistente, verificando os resultados obtidos.
    - Arquitetura: Foram utilizados os três componentes da solução, nomeadamente o *Web Service,* aplicação móvel e a base de dados.
    - Procedimento: Através do menu inicial, aceder ao menu de contexto no canto superior direito, selecionar Consultar, seguidamente Faturas e, por fim, escolher o parâmetro por *ID*. Repetir o mesmo teste para todos os parâmetros do menu Consultar Faturas, utilizando a opção "Consultar Detalhes", disponível através da tabela de resultados apresentada.
    - Resultado obtido: A aplicação móvel alterou o conteúdo do ecrã de acordo com os detalhes da fatura escolhida. Verificou-se o mesmo comportamento quando se utilizou a opção "Consultar Detalhes", disponível nas tabelas. O resultado obtido coincidiu com o resultado esperado, nas três plataformas móveis.

3. Aceder ao menu de pagamentos e verificar os resultados obtidos com base nos parâmetros introduzidos de forma correta e incorreta.
    - Arquitetura: Neste teste foi utilizada a aplicação móvel *MediaPOS Mobile* e a comunicação com o *Web Service* disponível através do *MEO Wallet.*
    - Procedimento: Após iniciar a aplicação, aceder ao menu de contexto no canto superior direito. Deve-se seguidamente escolher as opções Pagamentos, *MEO Wallet*. Repetir o teste para todos os parâmetros disponíveis (à exceção do último que corresponde ao reembolso).
    - Resultado obtido: Não é possível que o utilizador introduza ou escolha parâmetros errados, assim como é feita uma verificação e apresentadas mensagens intuitivas no caso de não existir nenhuma operação a visualizar. A consulta de detalhes através das tabelas funcionou como esperado. O resultado obtido nas três plataformas foi coincidente com o resultado esperado.

4. Após efetuar um pagamento bem-sucedido, verificar o sucesso ou insucesso do pedido de reembolso, com base em valores totais ou parciais corretos e errados.
    - Arquitetura: É utilizada a aplicação móvel *MediaPOS Mobile* e a comunicação com o *Web Service* disponível pelo *MEO Wallet.*
    - Procedimento: Através do menu de contexto disponível no ecrã principal da aplicação, deve-se aceder a Pagamentos, *MEO Wallet* e, por fim, à opção "Reembolso de um Pagamento". Após introduzir um *ID* da transação, é

necessário escolher o valor total ou parcial. Foram introduzidos dados corretos e errados.

- Resultado obtido: No caso de escolher o reembolso total, é apresentada uma mensagem de sucesso ou insucesso. Caso o reembolso seja parcial e os valores introduzidos sejam, por exemplo, inferiores ou superiores ao pagamento inicial, é apresentada uma mensagem de erro e explicação do mesmo. A aplicação comportou-se como esperado nos três sistemas operativos móveis, sendo que o resultado obtido estava de acordo com o que era esperado.

5. Após adicionar ao carrinho alguns artigos e aceder ao menu de pagamento, seleciona-se a opção de pagamento por MEO Wallet. Deve-se verificar o sucesso ou insucesso do pagamento utilizando tal serviço, assim como a mensagem de *feedback* correta no ecrã.
    - Arquitetura: Aplicação móvel *MediaPOS Mobile* e *Web Service MEO Wallet* foram os elementos utilizados neste teste.
    - Procedimento: Através do menu inicial, escolher alguns artigos e verificar se estão corretos no Carrinho. Seguidamente deve-se indicar a intenção de efetuar um Pagamento e escolher a opção *MEO Wallet.*
    - Resultado obtido: Tal como esperado, o conteúdo do ecrã apresenta a opção de pagamento por referência multibanco, visa ou conta *Wallet.* No menu superior existe ainda a possibilidade de consultar o *QR Code* de modo a simplificar o acesso ao cliente, assim como é possível verificar a atualização do estado do pagamento. As mensagens de *feedback* são fornecidas pelo *Web Service MEO Wallet* e estavam de acordo com o esperado. O resultado foi comum às três plataformas móveis.

6. Selecionar alguns artigos através do ecrã inicial, verificando o aparecimento correto dos mesmos no Carrinho, assim como o preço e quantidade.
    - Arquitetura: Neste teste é utilizada a aplicação móvel, que por sua vez acede ao *Web Service* de modo a consultar a lista de artigos e famílias. Essa lista é fornecida pela base de dados, sendo a comunicação feita por intermédio do *Web Service.*
    - Procedimento: Após iniciar a aplicação é automaticamente apresentada uma lista de artigos e famílias. Deve-se escolher aleatoriamente em testes sucessivos alguns dos artigos e consultar posteriormente a sua informação no menu Carrinho.
    - Resultado obtido: Os artigos estão presentes de forma correta a nível de identificação e preço. Contudo não são aglomerados por quantidades, sendo que cada um ocupa uma linha / entrada no Carrinho. Este resultado coincide com o comportamento expectável da aplicação / sistema. Não foram notadas quaisquer diferenças entre os três sistemas operativos móveis.

7. Através do menu inicial acede-se à funcionalidade de leitura de códigos de barras de um artigo. Verificar através de experiências sucessivas o sucesso e insucesso da leitura de artigos válidos e inválidos.
    - Arquitetura: Aplicação móvel, *Web Service* e a base de dados.
    - Procedimento: Através do menu inicial, por exemplo, deve-se aceder à funcionalidade que utiliza a câmara fotográfica na identificação de códigos de barras. O teste foi repetido várias vezes com condições de luminosidade diferentes, assim como com artigos válidos e inválidos (verificados através do *Web Service /* base de dados*).*
    - Resultado obtido: O dispositivo móvel em questão teve um grande peso neste teste. Os de gama superior, dotados de câmaras fotográficas mais capazes, conseguiram resultados de focagem e leitura mais rápidos, mesmo com condições de luminosidade desfavoráveis. Quanto mais escuro o ambiente em redor, mais difícil e demorada foi a leitura do código de barras. A verificação dos artigos por parte do *Web Service* decorreu como o esperado: caso exista é identificado e adicionado ao carrinho, caso não exista não acontece nada em específico, sendo necessária a adição manual através do ecrã inicial. Pode-se dizer que o teste foi bem-sucedido tendo em conta que em condições normais, a taxa de sucesso foi aceitável, nos três sistemas operativos móveis. O objetivo de conseguir efetuar leituras do código de barras utilizando a câmara fotográfica, foi bem conseguido.

8. Através do ecrã de pagamentos, tentar de forma sucessiva imprimir diversas faturas e verificar os resultados obtidos.
    - Arquitetura: Neste teste é utilizada a aplicação móvel, o *Web Service*, a base de dados e ainda uma impressora ligada ao computador.
    - Procedimento: Através do menu inicial deve-se iniciar um ciclo normal de venda de um produto, nomeadamente a escolha do artigo, a compra e a faturação mesmo. Uma vez pago é possível imprimir a fatura (guardada na base de dados), através do ecrã de pagamento final. O teste deve ser executado diversas vezes
    - Resultado obtido: O resultado obtido coincidiu com o resultado esperado. A fatura foi impressa a pedido do funcionário, através do *Android, iOS e Windows Phone*.

9. Executar a aplicação móvel utilizando a rede local onde o sistema se encontra e uma outra rede distinta. Verificar se a conexão é estabelecida em ambos os casos.
    - Arquitetura: Aplicação móvel, *Web Service* e base de dados.
    - Procedimento: Deve-se iniciar a aplicação quando conectado à mesma rede onde o computador (ou *Raspberry Pi)* se encontram e efetuar a consulta de uma fatura. De seguida deve-se replicar o procedimento utilizando uma rede

diferente. Neste segundo caso foram utilizados os dados móveis do *smartphone.*

- Resultado obtido: O resultado foi positivo em ambos os casos, sendo possível consultar uma fatura e consequente comunicação com o *Web Service*, independentemente da localização geográfica o utilizador.

10. Utilizando vários dispositivos móveis deve-se verificar o comportamento da interface gráfica e adaptabilidade às diferentes resoluções, assim como simplicidade da mesma. Verificar os resultados obtidos.
    - Arquitetura: Vários dispositivos móveis com a aplicação *MediaPOS Mobile* instalada.
    - Procedimento: Utilizando os três dispositivos móveis acessíveis para este teste, foram consultados os vários menus, assim como se testou rodar o ecrã dos mesmos (modo horizontal).
    - Resultado obtido: A aplicação móvel partilha o código da interface com o utilizador, pelo que o resultado obtido foi semelhante entre os três sistemas operativos móveis e *smartphone*s com diferentes tamanhos de ecrã. A aplicação móvel utiliza elementos nativos de cada um dos sistemas operativos, pelo que há determinadas funcionalidades que apresentam um aspeto ligeiramente diferente, contudo o comportamento é semelhante. O resultado obtido coincidiu com o esperado. Em alguns elementos o *iOS/iPhone* demonstrou alguns problemas, sendo que o centeúdo não era demonstrado como esperado.

11. Testar os diversos menus, verificando se o objetivo dos mesmos é claro e permite interações breves com as mais diversas funcionalidades.
    - Arquitetura: Aplicação móvel
    - Procedimento: Navegação e utilização de funcionalidades da aplicação, aleatoriamente, nos diferentes dispositivos móveis.
    - Resultado obtido: As interações com o utilizador são breves e objetivas, contudo este teste depende do tipo de utilizador em questão. Assume-se como um resultado neutro.

12. Consultar diversas faturas e pagamentos, anotando se a experiência de utilização foi satisfatoriamente rápida ou não.
    - Arquitetura: Neste teste é utilizado a aplicação móvel, *Web Service* e a base de dados.
    - Procedimento: A aplicação móvel é executada nos diversos sistemas operativos móveis e *smartphone*s, acedendo a faturas e pagamentos através dos menus específicos.

- Resultado obtido: A experiência de utilização dependeu do tipo de *smartphone.* Os de gama mais alta conseguiram providenciar respostas mais rápidas, através da apresentação das tabelas, assim como a aplicação é iniciada mais rapidamente. Quando se comparam as diversas aplicações nativas, as aplicações criadas utilizando *Xamarin* são ligeiramente mais lentas. Contudo assume-se com um teste bem-sucedido face ao esperado, tendo em conta que a experiência de utilização foi suficientemente satisfatória.

13. Executar a aplicação num dispositivo pela primeira vez, verificando se os procedimentos foram intuitivos, de modo a configurar o sistema para estabelecer ligação com o Web Service.
    - Arquitetura: Foram utilizados os três componentes da solução, nomeadamente o *Web Service,* a aplicação móvel e a base de dados.
    - Procedimento: Iniciar a aplicação móvel pela primeira vez e de seguida tentar aceder a algumas funcionalidades de forma aleatória.
    - Resultado obtido: O teste foi bem-sucedido comparando com o que seria esperado. São apresentadas mensagens de erro quando existem dados em falta, evitando assim o bloqueio da aplicação. O utilizador é ainda convidado a ser automaticamente conduzido para o menu das definições em falta.

4.4.3 Conclusões

Através dos resultados obtidos nos testes, é possível verificar que a implementação descrita na secção 4.3 e 4.2, cumpre os requisitos funcionais e não funcionais apresentados na secção 4.1.

Embora o plano de testes tenha sido executado com base puramente na experiência empírica, foi possível concluir que os testes funcionais ao sistema apresentam uma oportunidade de evolução. Na grande maioria dos testes o resultado apresentado corresponde ao resultado que se esperava obter, contudo este tipo de soluções lidam com dados sensíveis e críticos, pelo que é necessário, como trabalho futuro, efetuar um maior nível de verificações e controlo de erros.

Os resultados obtidos comprovam que o sistema faz aquilo que lhe era pedido, porém seria interessante efetuar um maior número de testes no futuro, nomeadamente testes de aceitação.

5 Conclusões

Neste último capítulo são apresentados os resultados finais do estudo desenvolvido. O capítulo é constituído por uma avaliação dos objetivos propostos inicialmente, assim como das dificuldades, limitações e prospeção de trabalho futuro. Para finalizar são feitas algumas considerações finais.

5.1 Objetivos Alcançados

No decorrer desta dissertação pretendia-se criar uma solução para dispositivos móveis baseada no produto *MediaPOS*, e dotar a mesma de funcionalidades avançadas de interação com pessoas e equipamentos periféricos.

Inicialmente foi efetuado um levantamento do estado da arte das diversas soluções para pontos de venda existentes no mercado Português. Das soluções de pontos de venda tradicionais e móveis, foi feito um estudo comparativo o que permitiu determinar os pontos comuns entre elas, assim como o que as diferencia. Através desse estudo foi possível perceber as tendências do mercado no que a tecnologias e funcionalidades diz respeito.

Um dos propósitos desta dissertação era efetuar paralelamente um estudo das diversas tecnologias com aplicabilidade em pontos de venda para dispositivos móveis. Pretendia-se com isto tirar algumas conclusões sobre as diversas formas de interagir com alguns dispositivos periféricos, componentes que permitissem a automatização de processos, assim como tirar partido das tecnologias existentes nos dispositivos móveis. De igual modo, constituía objetivo específico o estudo e utilização de soluções de desenvolvimento multiplataforma, assim como a sua adequação às necessidades existentes.

Numa fase posterior e, tal como previsto, foi desenvolvido um protótipo para demonstrar algumas funcionalidades avançadas, utilizando uma única solução comum a todos os sistemas operativos móveis. A complexidade desta etapa era alta, particularmente devido à aprendizagem de novas tecnologias, em especial a utilização do *Xamarin.* A curva de

aprendizagem da plataforma limitou imenso o progresso e a obtenção de resultados mais promissores. Contudo o protótipo criado s é a prova desse esforço.

Por fim, foram realizados alguns testes de forma a perceber se a solução correspondia aos requisitos e necessidades existentes.

Fazendo uma retrospeção do trabalho realizado, é possível verificar que os objetivos principais e específicos, previamente estabelecidos, foram cumpridos.

5.2 Dificuldades, Limitações e Trabalho Futuro

Embora os objetivos definidos inicialmente tenham sido atingidos com sucesso, trata-se de um sistema de elevada complexidade. De forma a tornar-se uma aplicação certificada pela Autoridade Tributária e Aduaneira, há ainda um longo caminho a percorrer. As regras de certificação são exigentes, nomeadamente no controlo de acesso à aplicação, cópias de segurança, criação do ficheiro *SAFT-PT*, entre muitas outras.

Grande parte das decisões tomadas no decorrer deste estudo foram condicionadas pelo facto de se tentar criar uma solução única e uniforme, que satisfaça todos os sistemas operativos móveis de igual forma. Foram descartadas algumas tecnologias que poderiam conferir excelentes funcionalidades avançadas.

Por exemplo, devido à falta de acesso a impressoras *Bluetooth* ou *Wi-Fi* perdeu-se uma excelente oportunidade de resolver a comunicação com dispositivos. Por outro lado, a utilização dos cabos *OTG* seria uma excelente solução de baixo custo para comunicação com periféricos tais como balanças ou impressoras. Infelizmente, esta solução não era comum a todos os sistemas operativos nem se comportava da mesma forma.

A taxa de incidência de dispositivos móveis *Windows Phone* é de 0.4% [70], um valor demasiado baixo para limitar opções ou um esforço acrescido de forma a suportar o mesmo. Isso é notório através nas Tabelas 1 e 2, sendo que nenhuma das soluções para pontos de venda analisadas tinham suporte para *Windows Phone.*

Uma outra limitação encontrada no decorrer do estudo foi a utilização de ambientes de desenvolvimento multiplataforma. Se por um lado utilizar *C#* e criar uma única solução comum a vários sistemas operativos móveis é algo apetecível, por outro lado, o *Xamarin* ainda é uma tecnologia pouco madura. O desempenho das aplicações é notoriamente inferior quando comparado com soluções nativas. Existe uma forte necessidade de recorrer a componentes extras para conseguir realizar algumas funcionalidades básicas, de forma a contornar as mais diversas limitações.

A título de exemplo, quando se cria um projeto novo utilizando *Xamarin.Forms Portable* e as versões de *software* descritas neste documento, automaticamente é criado com erros de compilação. É necessário efetuar inicialmente algumas modificações de forma a contornar essa situação.

A curva de aprendizagem é longa, dado que o *Xamarin* tem imensas particularidades que requerem um grande trabalho de investigação, leitura e experimentação. Possivelmente num futuro próximo, tendo em conta a recente aquisição do *Xamarin* por parte da *Microsoft*, existirá uma grande evolução e melhoria. Contudo, no momento de desenvolvimento deste protótipo, acredita-se que fosse mais simples e eficaz aprender a desenvolver para cada um dos sistemas operativos móveis em separado. Ou, como alternativa, utilizar o *Xamarin.Forms* mas não criar ou suportar o *Windows Phone*. Muitas das dificuldades encontradas durante o desenvolvimento resultaram do facto de existir muito pouco suporte de bibliotecas, entre outros, criando a necessidade de encontrar alternativas ou código específico apenas para esse sistema operativo móvel.

Por definição, um protótipo é uma versão preliminar do programa, com vista a ser testada e aperfeiçoada, sendo que não é esperado uma solução completa. Nesse sentido, e como trabalho futuro, seria interessante explorar outro tipo de funcionalidades avançadas, nomeadamente *Bluetooth Beacons* ,e tornar a aplicação independente através da criação de uma base de dados interna. De forma a não estar limitado ao *Raspberry Pi* ou computador, faria sentido utilizar impressoras *Bluetooth* ou *Wi-Fi.*

Como forma de validação, seria ainda interessante efetuar testes exaustivos, assim como aplicar em clientes selecionados como prova de aceitação.

5.3 Considerações Finais

Em primeiro lugar, é relevante referir o quão gratificante e produtiva foi a realização deste trabalho de investigação, não só pela evolução académica que potenciou, como pela experiência profissional ganha. O desenvolvimento da dissertação em ambiente empresarial permitiu o contato diário com os verdadeiros problemas dos utilizadores dos pontos de venda, assim como necessidade de resolver problemas de carácter urgente.

Com grande satisfação considera-se que o estudo desenvolvido contribuiu para a evolução contínua da *CimSoft*, nomeadamente no desenvolvimento de pontos de venda para dispositivos móveis. Embora ainda haja um grande percurso a percorrer até que o produto *MediaPOS Mobile* esteja maduro o suficiente, foram dados os passos necessários para que o mesmo se venha a tornar uma realidade.

Por fim, gostaria de salientar que a base tecnológica e científica criada irá permitir, num futuro próximo, a aplicação do *MediaPOS Mobile* em contexto real, uma vez que a empresa pretende dar continuidade ao projeto.

Referências

[1] Gabinete do Secretário de Estado dos Assuntos Fiscais, «Combate à Fraude e Evasão Fiscais e Aduaneiras», pp. 13–14, 2014.

[2] Empresas Hoje, «Mobile Enterprise Applications in Portugal», 2015. [Em linha]. Disponível em: http://empresashoje.pt/high-tech/mobile-enterprise-applications-in-portugal/. [Acedido: 17-Fev-2016].

[3] Revel Systems, «What is Point of Sale Systems». [Em linha]. Disponível em: https://revelsystems.com/pos-systems-history. [Acedido: 10-Fev-2016].

[4] «PAYATSHOP». [Em linha]. Disponível em: http://www.payatshop.com/solutions.html. [Acedido: 21-Fev-2016].

[5] GrupoPIE Portugal, «Quem Somos». [Em linha]. Disponível em: https://www.grupopie.com/empresa.html. [Acedido: 10-Fev-2016].

[6] GrupoPIE Portugal, «WinREST - Descrição». [Em linha]. Disponível em: https://www.grupopie.com/produtos.html. [Acedido: 10-Fev-2016].

[7] GrupoPIE Portugal, «WinREST FO». [Em linha]. Disponível em: https://www.grupopie.com/frontoffice.html. [Acedido: 10-Fev-2016].

[8] «MB WAY». [Em linha]. Disponível em: https://www.mbway.pt/#vantagens. [Acedido: 04-Out-2016].

[9] «MEO Wallet». [Em linha]. Disponível em: https://wallet.pt/. [Acedido: 21-Fev-2016].

[10] «Início | Seqr Portugal». [Em linha]. Disponível em: https://www.seqr.com/pt/. [Acedido: 21-Fev-2016].

[11] GrupoPIE Portugal, «MB WAY | A NOVA FORMA DE RECEBER PAGAMENTOS PELO TELEMÓVEL».

[12] GrupoPIE Portugal, «PingWin - Conceito e Tecnologia». [Em linha]. Disponível em: https://www.grupopie.com/conceito_pingwin.html. [Acedido: 10-Fev-2016].

[13] GrupoPIE Portugal, «PingWin - Descrição». [Em linha]. Disponível em: https://www.grupopie.com/pingwin.html. [Acedido: 10-Fev-2016].

[14] GrupoPIE Portugal, «PingWin - Vantagens». [Em linha]. Disponível em: https://www.grupopie.com/vantagens_pingwin.html. [Acedido: 10-Fev-2016].

[15] GrupoPIE Portugal, «A oferta tecnológica mais evoluída do mercado», pp. 1–11, 2013.

[16] Sage, «Sobre nós». [Em linha]. Disponível em: http://www.lojasage.com/gca/index.php?id=382. [Acedido: 10-Fev-2016].

[17] Sage, «Sage POS Standard». [Em linha]. Disponível em: http://www.sage.pt/solucoes/faturacao/pequenas-e-medias-empresas/sage-pos-standard. [Acedido: 10-Fev-2016].

[18] Sage, «Detalhe do Sage POS Standard». [Em linha]. Disponível em: http://www.sage.pt/solucoes/faturacao/pequenas-e-medias-empresas/sage-pos-standard/detalhes. [Acedido: 10-Fev-2016].

[19] Sage, «Sage GesRestII». [Em linha]. Disponível em: http://www.sage.pt/solucoes/faturacao/pequenas-e-medias-empresas/sage-gesrestii. [Acedido: 10-Fev-2016].

[20] Sage, «Detalhes do Sage GesRestII». [Em linha]. Disponível em: http://www.sage.pt/solucoes/faturacao/pequenas-e-medias-empresas/sage-gesrestii/detalhes. [Acedido: 10-Fev-2016].

[21] Sage, «Sage Retail». [Em linha]. Disponível em: http://www.sage.pt/solucoes/faturacao/pequenas-e-medias-empresas/sage-retail. [Acedido: 10-Fev-2016].

[22] Sage, «Detalhe do Sage Retail». [Em linha]. Disponível em: http://www.sage.pt/solucoes/faturacao/pequenas-e-medias-empresas/sage-retail/detalhes. [Acedido: 10-Fev-2016].

[23] Zone Soft, «Sobre a Zone Soft». [Em linha]. Disponível em: https://www.zonesoft.pt/conteudo.php?id=1. [Acedido: 10-Fev-2016].

[24] Zone Soft, «ZS REST». [Em linha]. Disponível em: https://www.zonesoft.pt/conteudo.php?id=4. [Acedido: 10-Fev-2016].

[25] Zone Soft, «ZS POS MOBILE». [Em linha]. Disponível em: https://www.zonesoft.pt/conteudo.php?id=6. [Acedido: 10-Fev-2016].

[26] Zone Soft, «Comércio a Retalho».

[27] Zone Soft, «ZS POS». [Em linha]. Disponível em: https://www.zonesoft.pt/conteudo.php?id=5. [Acedido: 10-Fev-2016].

[28] Primavera, «Missão». [Em linha]. Disponível em: http://pt.primaverabss.com/pt/primavera/#gca9. [Acedido: 10-Fev-2016].

[29] Primavera, «Tlim!» [Em linha]. Disponível em: http://pt.primaverabss.com/pt/catalogo/solucoes-setoriais/retalho/tlim/tlim/. [Acedido: 10-Fev-2016].

[30] Primavera, «Pssst!» [Em linha]. Disponível em: http://pt.primaverabss.com/pt/catalogo/solucoes-setoriais/restauracao/pssst/pssst/. [Acedido: 10-Fev-2016].

[31] Primavera, «Pssst! Experience». [Em linha]. Disponível em: http://pt.primaverabss.com/pt/catalogo/aceleradores-de-negocio/pssst-experience/pssst-experience/. [Acedido: 10-Fev-2016].

[32] K. Crow, «VALUE ANALYSIS AND FUNCTION ANALYSIS SYSTEM TECHNIQUE». [Em linha]. Disponível em: http://www.npd-solutions.com/va.html. [Acedido: 17-Fev-2016].

[33] N. A. S. Waldemar Karwowski,Marcelo M. Soares, *Human Factors and Ergonomics in*

Consumer Product Design: Uses and Applications. CRC Press, 2011.

[34] S. NICOLA, E. P. FERREIRA, e J. J. P. FERREIRA, *a Novel Framework for Modeling Value for the Customer, an Essay on Negotiation*, vol. 11, n. 3. 2012.

[35] «Strategyzer | Business Model Generation». [Em linha]. Disponível em: http://www.businessmodelgeneration.com/.

[36] «CimSoft ® - MediaPOS». [Em linha]. Disponível em: http://cimsoft.pt/pt/mediapos.html. [Acedido: 10-Out-2016].

[37] «Raspberry Pi 3 Model B - Raspberry Pi». [Em linha]. Disponível em: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. [Acedido: 10-Out-2016].

[38] «Raspberry Pi». [Em linha]. Disponível em: https://en.wikipedia.org/wiki/Raspberry_Pi. [Acedido: 04-Out-2016].

[39] «USB On-The-Go». [Em linha]. Disponível em: https://en.wikipedia.org/wiki/USB_On-The-Go.

[40] «What Is USB OTG? 5 Cool Ways to Use It on Android». [Em linha]. Disponível em: http://www.makeuseof.com/tag/what-is-usb-otg-5-cool-ways-use-android/. [Acedido: 04-Out-2016].

[41] «How to: Make your own USB OTG cable for an Android smartphone – Tech2». [Em linha]. Disponível em: http://tech.firstpost.com/news-analysis/how-to-make-your-own-usb-otg-cable-for-an-android-smartphone-29503.html. [Acedido: 04-Out-2016].

[42] «Customer Reviews: Apple iPad Camera Connection Kit … - Apple (UK)». [Em linha]. Disponível em: http://www.apple.com/uk/shop/reviews/MC531ZM/A/apple-ipad-camera-connection-kit. [Acedido: 04-Out-2016].

[43] «Apple's Lightning to USB Camera Adapter Review - MacReview.com». [Em linha]. Disponível em: http://macreview.com/2013/12/apples-lightning-usb-camera-adapter-review/.

[44] «Did you know how many different kinds of sensors go inside a smartphone?» [Em linha]. Disponível em: http://www.phonearena.com/news/Did-you-know-how-many-different-kinds-of-sensors-go-inside-a-smartphone_id57885. [Acedido: 04-Out-2016].

[45] «IEEE 802.11». [Em linha]. Disponível em: https://en.wikipedia.org/wiki/IEEE_802.11. [Acedido: 10-Out-2016].

[46] «List of WLAN channels». [Em linha]. Disponível em: https://en.wikipedia.org/wiki/List_of_WLAN_channels.

[47] «Different WiFi Protocols and Data Rates». [Em linha]. Disponível em: http://www.intel.com/content/www/us/en/support/network-and-i-o/wireless-networking/000005725.html.

[48] «O que é NFC (Near Field Communication)?» [Em linha]. Disponível em: http://www.infowester.com/nfc.php.

[49] «Example of an NFC application Standard-compliant NFC signals defined in detail at the press of a button».

[50] «What is NFC and how do I program NFC tags? - Pocketables». [Em linha]. Disponível em: http://www.pocketables.com/2013/02/what-is-nfc-and-how-do-i-program-nfc-tags.html. [Acedido: 04-Out-2016].

[51] «Bluetooth Basics». [Em linha]. Disponível em: https://learn.sparkfun.com/tutorials/bluetooth-basics/how-bluetooth-works. [Acedido: 10-Out-2016].

[52] «Tecnologia Bluetooth: o que é e como funciona?» [Em linha]. Disponível em: http://www.infowester.com/bluetooth.php. [Acedido: 04-Out-2016].

[53] «Bluetooth V2.1 | Seminar Report and PPT for CSE Students». [Em linha]. Disponível em: http://www.seminarsonly.com/computer science/Bluetooth-V2-1.php. [Acedido: 04-Out-2016].

[54] «Epson Unveils Mobile POS Printer With Bluetooth Support For IOS | AppleMagazine». [Em linha]. Disponível em: https://applemagazine.com/epson-unveils-mobile-pos-printer-with-bluetooth-support-for-ios/4413. [Acedido: 04-Out-2016].

[55] «The pros and cons of Bluetooth Low Energy». [Em linha]. Disponível em: http://www.electronicsweekly.com/news/design/communications/pros-cons-bluetooth-low-energy-2014-10/. [Acedido: 10-Out-2016].

[56] «Extensive Guide to Bluetooth Low Energy (BLE) Beacons». [Em linha]. Disponível em: https://kontakt.io/blog/extensive-guide-to-bluetooth-beacons/. [Acedido: 10-Out-2016].

[57] «Business use of Bluetooth Beacons | Apteriks blog». [Em linha]. Disponível em: https://blog.apteriks.com/2015/03/17/business-use-of-bluetooth-beacons/. [Acedido: 04-Out-2016].

[58] «How mobile payments will grow in 2016». [Em linha]. Disponível em: http://fortune.com/2015/10/29/mobile-payments-grow-2016/. [Acedido: 04-Out-2015].

[59] «Mobile Payments Will Triple in the US in 2016 - eMarketer». [Em linha]. Disponível em: http://www.emarketer.com/Article/Mobile-Payments-Will-Triple-US-2016/1013147. [Acedido: 04-Out-2016].

[60] «Bancos Aderentes - MB WAY». [Em linha]. Disponível em: https://www.mbway.pt/bancos-aderentes/. [Acedido: 04-Out-2016].

[61] «Implementação - MB WAY». [Em linha]. Disponível em: https://www.mbway.pt/developers/implementacao/. [Acedido: 04-Out-2016].

[62] «MEO Wallet». [Em linha]. Disponível em: https://wallet.pt/merchants/how.

[63] «MEO Wallet - Precário Comerciante». [Em linha]. Disponível em: https://wallet.pt/merchants/rates. [Acedido: 04-Out-2016].

[64] «Pro Checkout». [Em linha]. Disponível em: https://developers.wallet.pt/en/procheckout/. [Acedido: 10-Out-2016].

[65] «MEO Wallet API - Pro Checkout». [Em linha]. Disponível em: https://developers.wallet.pt/en/procheckout/. [Acedido: 04-Out-2016].

[66] «Sandbox Environment». [Em linha]. Disponível em: https://developers.wallet.pt/en/basics/sandbox.html. [Acedido: 10-Out-2016].

[67] «Acerca de | Seqr Portugal». [Em linha]. Disponível em: https://www.seqr.com/pt/acerca-de/. [Acedido: 11-Out-2016].

[68] «SEQR Developer Site». [Em linha]. Disponível em: http://seamlessdistribution.github.io/. [Acedido: 04-Out-2016].

[69] «Micro-Moments: Your Guide to Winning the Shift to Mobile».

[70] «IDC: Smartphone OS Market Share 2016, 2015». [Em linha]. Disponível em: http://www.idc.com/prodserv/smartphone-os-market-share.jsp. [Acedido: 01-Out-2016].

[71] AllianceTek, «Cross Platform Development with Mono on the .Net Framework», 2016.

[72] J. M. Wargo, «Which to Use: Cordova or PhoneGap? | | InformIT». [Em linha]. Disponível em: http://www.informit.com/articles/article.aspx?p=2478076. [Acedido: 29-Set-2016].

[73] A. Trice, «PhoneGap Explained Visually». [Em linha]. Disponível em: http://phonegap.com/blog/2012/05/02/phonegap-explained-visually/. [Acedido: 29-Set-2016].

[74] «Appcelerator Open Source». [Em linha]. Disponível em: http://www.appcelerator.org/. [Acedido: 29-Set-2016].

[75] «GitHub - Appcelerator Titanium Mobile». [Em linha]. Disponível em: https://github.com/appcelerator/titanium_mobile. [Acedido: 29-Set-2016].

[76] N. Jansma, «Appcelerator Titanium Intro (2014)». [Em linha]. Disponível em: http://www.slideshare.net/nicjansma/appcelerator-titanium-intro-2014. [Acedido: 29-Set-2016].

[77] «Build Cross-Platform Android and iOS UIs with Xamarin Forms». [Em linha]. Disponível em: https://www.sitepoint.com/build-cross-platform-android-ios-uis-xamarin-forms/. [Acedido: 01-Out-2016].

[78] «Mobile Application Development to Build Apps in C# - Xamarin». [Em linha]. Disponível em: https://www.xamarin.com/platform. [Acedido: 01-Out-2016].

[79] «Microsoft to acquire Xamarin and empower more developers to build apps on any device - The Official Microsoft Blog». [Em linha]. Disponível em: https://blogs.microsoft.com/blog/2016/02/24/microsoft-to-acquire-xamarin-and-empower-more-developers-to-build-apps-on-any-device/. [Acedido: 01-Out-2016].

[80] Xamarin, «Xamarin vs. Hybrid HTML: Making the Right Mobile Platform Choice for the Enterprise», n. Evolve 2016, p. 7, 2016.

[81] «Connecting to the Mac - Xamarin». [Em linha]. Disponível em: https://developer.xamarin.com/guides/ios/getting_started/installation/windows/connecting-to-mac/. [Acedido: 01-Out-2016].

[82] «System Requirements - Xamarin». [Em linha]. Disponível em: https://developer.xamarin.com/guides/cross-platform/getting_started/requirements/. [Acedido: 01-Out-2016].

[83] «Shared Projects - Xamarin». [Em linha]. Disponível em: https://developer.xamarin.com/guides/cross-platform/application_fundamentals/shared_projects/. [Acedido: 01-Out-2016].

[84] «Introduction to Portable Class Libraries - Xamarin». [Em linha]. Disponível em: https://developer.xamarin.com/guides/cross-platform/application_fundamentals/pcl/introduction_to_portable_class_libraries/. [Acedido: 01-Out-2016].

[85] «Build a Native Android UI & iOS UI with Xamarin.Forms - Xamarin». [Em linha]. Disponível em: https://www.xamarin.com/forms. [Acedido: 01-Out-2016].

[86] «NuGet Gallery | Home». [Em linha]. Disponível em: https://www.nuget.org/. [Acedido: 01-Out-2016].

[87] «Components / Xamarin». [Em linha]. Disponível em: https://components.xamarin.com/. [Acedido: 01-Out-2016].

[88] «Home | Mono». [Em linha]. Disponível em: http://www.mono-project.com/. [Acedido: 21-Fev-2016].

[89] «Mobile Application Development to Build Apps in C# - Xamarin». [Em linha]. Disponível em: https://www.xamarin.com/. [Acedido: 21-Fev-2016].

[90] «Build a Native Android UI & iOS UI with Xamarin.Forms - Xamarin». [Em linha]. Disponível em: https://www.xamarin.com/forms. [Acedido: 08-Out-2016].

[91] «Código de barras». [Em linha]. Disponível em: https://pt.wikipedia.org/wiki/C%C3%B3digo_de_barras. [Acedido: 10-Out-2016].

[92] «Creating EAN-13 Barcodes with C# - CodeProject». [Em linha]. Disponível em: http://www.codeproject.com/Articles/10162/Creating-EAN-Barcodes-with-C. [Acedido: 10-Out-2016].

[93] Redth, «ZXing.Net.Mobile». [Em linha]. Disponível em: https://github.com/Redth/ZXing.Net.Mobile. [Acedido: 10-Out-2016].

[94] «Part 4 - Dealing with Multiple Platforms - Xamarin». [Em linha]. Disponível em: https://developer.xamarin.com/guides/cross-platform/application_fundamentals/building_cross_platform_applications/part_4_-_platform_divergence_abstraction_divergent_implementation/. [Acedido: 10-Out-2016].

[95] «Cross-Platform Performance - Xamarin». [Em linha]. Disponível em: https://developer.xamarin.com/guides/cross-platform/deployment,_testing,_and_metrics/memory_perf_best_practices/. [Acedido: 07-Out-2016].