INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Análise das diferentes soluções disponíveis para desenvolvimento de aplicações móveis.

HUGO FILIPE TRIGO CERDEIRA
Outubro de 2020

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Análise das diferentes soluções disponíveis para desenvolvimento de aplicações móveis.

Estudo do custo de desenvolvimento.

Hugo Filipe Trigo Cerdeira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Engenharia de Software.

Orientador: Paulo Baltarejo Sousa

Porto, Outubro 2020

Agradecimentos

Primeiramente quero agradecer ao ISEP, que foi a instituição que tornou possível a realização desta tese quer por disponibiliza a oportunidade de a realizar quer por me dar a oportunidade de adquirir conhecimento ao longo do meu percurso académico. Agradeço também a todos os docentes que me acompanharam do meu percurso académico dentro do ISEP.

Agradeço com especial atenção ao professor Paulo Baltarejo Sousa pela orientação e tempo dedicado, fatores que contribuíram positivamente para o desenvolvimento desta tese.

Estou também agradecido à empresa DIGITAL SKWARE por me permitir o uso dos seus dados internos para realizar este estudo.

Por fim agradeço à minha família e amigos pelo apoio que me deram ao longo do meu percurso académico.

Resumo

Desde o lançamento do primeiro iPhone é possível observar que ao longo dos anos a utilização de smartphones pela população mundial, tem crescido (Turner, 2020). E é esperado que a tendência continue a manifestar-se.

Com o aumento da utilização de smartphones como meio de acesso à Internet é cada vez mais evidente a disputa pela atenção do utilizador. Nessa "guerra", travada por todas as organizações cujo negócio revolve à volta da sua presença na Internet, as aplicações móveis são uma das mais eficazes maneiras de manter o utilizador interessado e envolvido com aquilo que as empresas produzem.

No entanto existem desafios no mercado relativo a aplicações móveis. Com seu crescimento tem sido notada uma escassez de recursos humanos tais como programadores capazes de trabalhar no desenvolvimento de aplicações móveis. Com isto é natural que as empresas e os seus trabalhadores procurem soluções que permitam um desenvolvimento mais eficiente. Umas das soluções que tem ganho popularidade é o uso de tecnologias que permitam o desenvolvimento de aplicações móveis nativas multiplataforma usando apenas uma base de código. Isto permite aos programadores ter uma eficiência mais elevada.

O facto é que muitas dessas tecnologias, que permitem desenvolvimento de aplicações móveis multiplataforma, são relativamente modernas. Isso faz com que ainda não existem dados que as comparem na sua eficiência de desenvolvimento. Esses dados seriam úteis para programadores ou empresas, uma vez que possibilitava a escolha da tecnologia mais eficiente.

Neste estudo é desenhada e implementada uma metodologia que permite recolha de dados relativos à eficiência de desenvolvimento de aplicações móveis. A metodologia foi desenhada e implementada para determinar a tecnologia de desenvolvimento de aplicações móveis mais eficiente. Esse objetivo não foi cumprido devido à falta de dados.

Palavras-chave: Eficiência, Multiplataforma, Nativa.

Abstract

Since the launch of the first iPhone it is possible to observe that over the years the use of smartphones by the world population has grown (Turner, 2020)). The trend is expected to continue to manifest itself.

With the increase in the use of smartphones as a means of Internet access, the dispute for the user's attention is increasingly evident. In this "war", waged by all organizations whose business revolves around their presence on the Internet, mobile applications are one of the most effective ways to keep the user interested and involved with what companies produce.

However, there are challenges in the market for mobile applications. With their growth has been noted a shortage of human resources such as programmers capable of working in the development of mobile applications. With this it is natural that companies and their workers look for solutions that allow a more efficient development. One of the solutions that has gained popularity is the use of technologies that allow the development of native multiplatform mobile applications using only one code base. This allows developers to have a higher efficiency.

The fact is that many of these technologies, which allow the development of multiplatform mobile applications, are relatively modern. This means that there is still no data to compare them in their development efficiency. This data would be useful for programmers or companies, since it would allow them to choose the most efficient technology.

In this study, a methodology is designed and implemented to collect data related to the development efficiency of mobile applications. The methodology was designed and implemented to determine the most efficient mobile application development technology. This objective was not accomplished due to lack of data.

Keywords: Efficiency, Multiplatform, Native.

Indice

Lista de Figuras

Lista de Tabelas

Lista de Excertos de Código

Acrónimos e Glossário

Lista de Acrónimos

HTML	HyperText Markup Language
JSX	JavaScript XML
JS	JavaScript
KPI	Key Performance Indicator
RN	React Native
UCP	Use Case Point
UCPs/h	Use Case Points por hora
UML	Unified Modeling Language

1 Introdução

Interagir com o seu *smartphone* tornou-se um hábito que as pessoas fazem de forma completamente natural e instintiva. Não há qualquer dúvida que o desde que a Apple lançou o primeiro iPhone há mais de dez anos os *smartphones* passaram a ser um instrumento quase essencial para o quotidiano de um indivíduo.

Muitas das interações de um utilizador com um *smartphone* passam pela utilização de uma aplicação móvel. Em 2017 foi realizado um estudo onde foi possível concluir que, num determinado momento, existem entre 60 a 90 aplicações móveis instaladas num *smartphone,* (Thompson, 2017). Com o número crescente de pessoas qua usam *smartphone*s, cerca 3.5 mil milhões em 2020 (Turner, 2020), é natural que programadores de aplicações móveis sejam das pessoas mais requisitadas na área de desenvolvimento de *software* (Green, 2019).

O trabalho realizado no contexto desta dissertação, vai incidir sobre o tempo dedicado, por programadores a criar e manter aplicações móveis. Sendo que neste primeiro capítulo de introdução está presente informação relacionada sobre alguns problemas identificados na criação de aplicações móveis. É também apresentado o objetivo deste estudo bem como os resultados esperados e a abordagem usada para os alcançar. Por fim este capítulo termina com a estrutura do documento.

1.1 Desenvolvimento de aplicações móveis

Quando uma aplicação móvel é desenvolvida é, geralmente, objetivo do criador da mesma que ela seja utilizada pelo maior número de pessoas possíveis. Porém para alcançar isso geralmente é necessário efetuar o desenvolvimento da aplicação móvel para os dois sistemas operativos, com mais utilizadores, Android e iOS. O facto de as aplicações móveis desenvolvidas para Android não serem compatíveis com o iOS, e vice-versa, faz com que seja necessário o desenvolvimento de duas aplicações, com duas bases de código diferentes, um para cada sistema.

As aplicações móveis nativamente desenvolvidas para Android são desenvolvidas usando a linguagem de programação Kotlin ou Java, e as desenvolvidas para iOS, Swift ou Objective-C. Estas linguagens de programação são diferentes, o que faz com que aproveitar código de uma aplicação móvel desenvolvida para Android para acelerar o desenvolvimento da mesma aplicação em iOS impossível.

Este problema faz com que um programador que queira desenvolver uma aplicação móvel, e disponibilizá-la tanto para os utilizadores de Android como para os utilizadores de iOS, tenha custos acrescidos. Uma vez que precisa de desenvolver duas aplicações móveis com bases de código completamente diferente.

1.2 Soluções para o desenvolvimento de aplicações móveis

Com o crescente número de aplicações móveis a serem desenvolvidas, o interesse sobre tecnologias para acelerar o desenvolvimento das mesmas aumenta. Com isso, a quantidade de tecnologias, para agilizar e facilitar o processo de desenvolvimento, presentes no mercado é cada vez maior (Martin, 2020).

Durante os últimos anos várias tecnologias que visam a aumentar a velocidade do desenvolvimento de aplicações móveis têm surgido. Entre elas estão presentes tecnologias como React Native, Xamarin, Flutter ou Ionic. Sendo as anteriores, tecnologias provadas e usadas no desenvolvimento de muitas das mais populares aplicações (Feoktistov, 2020).

Apesar de, de uma forma geral, uma maior oferta de soluções para um problema é vista como algo positivo, esse mesmo facto levanta por si só outro problema.

Diferentes tecnologias permitem ao programador ter eficiências diferentes, ou seja, diferentes tecnologias fazem com que o tempo necessário para desenvolver a sua aplicação móvel varie. Quando um programador opta por usar uma das tecnologias acima referidas, e tem de fazer a escolha de qual usar, o programador não consegue usar a eficiência de desenvolvimento de cada uma delas como critério, uma vez que não existem dados concretos que as compare nesse aspeto.

Em suma, as tecnologias apresentadas no início desta secção têm o propósito de acelerar o desenvolvimento de aplicações móveis, porque permitem, com uma única implementação, criar aplicações para Android e iOS. Porém não existem quaisquer dados que permitam comparar, as diferentes tecnologias, na eficiência de desenvolvimento. Entende-se como eficiência a quantidade de tempo necessário para o desenvolvimento de uma aplicação móvel numa determinada tecnologia, tecnologias com um desenvolvimento mais eficiente permitem chegar ao produto final em menos tempo.

1.3 Objetivo

Tendo em conta o facto dos problemas existentes no desenvolvimento de aplicações móveis, apresentados na Secção 1.1, serem tão evidentes é natural que tenham surgido várias tecnologias que os tentem mitigar ou eliminar.

Como apresentado na Secção 1.2 escolher uma das tecnologias, que permite usar uma base de código comum parar criar aplicações para Android e iOS, nem sempre é fácil. O facto destas tecnologias serem relativamente novas faz com que muitas vezes programadores tenham dificuldade em encontrar informação que as compare entre si. Caso o programador pretenda tomar uma decisão sobre que tecnologia utilizar baseando-se no quão bem cada uma destas tecnologias cumpre o seu propósito, que efetivamente é o desenvolvimento de aplicações móveis de forma mais rápida, é quase impossível tomar uma decisão fundamentada.

Deste estudo deve resultar uma metodologia de recolha de dados que permita recolher informação relacionada com a eficiência do desenvolvimento de aplicações móveis. Essa metodologia tem de ser capaz de recolher dados das diferentes tecnologias em estudo.

Como referido na Secção 1.2, no contexto deste estudo entende-se como eficiência no desenvolvimento, o tempo necessário para implementar uma determinada tarefa usando uma determinada tecnologia.

Com estes fatores em mente, este estudo tem o objetivo de responder à seguinte questão:

- **Qual a tecnologia de desenvolvimento de aplicações móveis que permite um desenvolvimento de aplicações para sistemas Android e iOS mais eficiente?**

1.4 Resultados esperados

Como identificado na Secção 1.3 é necessário identificar qual a tecnologia que permite um desenvolvimento mais eficiente. Para tal é necessário recolher dados que serão usados para comparar as tecnologias, sendo necessário recolher informação relacionada com a velocidade de desenvolvimento de cada uma das aplicações.

Deste estudo é esperado que resulte uma metodologia que permita recolher dados, relacionados com a eficiência do desenvolvimento de aplicações móveis, de cada uma das tecnologias, bem como o tratamento e comparação dos dados recolhidos. De modo a ser possível eleger a tecnologia mais eficiente no desenvolvimento de aplicações móveis para os sistemas, Android e iOS. Sendo assim possível responder à questão levantada na Secção 1.3.

1.5 Abordagem

Numa fase inicial do estudo será feita um levantamento das diferentes tecnologias para o desenvolvimento de aplicações móveis. Essas tecnologias serão analisadas e selecionadas as que permitem o desenvolvimento de aplicações nativas para iOS e Android com a mesma base de código ou uma grande percentagem da base de código partilhada.

Após a seleção das tecnologias viáveis será necessário elaborar a metodologia de recolha de dados. Da metodologia elaborada fará parte um conjunto de problemas a serem resolvidos usando as diferentes tecnologias. Os problemas elaborados terão em consideração as funcionalidades que são frequentes na maior parte das aplicações móveis, algo que terá de ser estudado.

De seguida um conjunto de programadores será apontado para implementar as soluções aos problemas propostos, em cada uma das ferramentas. Esses programadores, que farão parte do estudo, serão referidos como sujeitos a partir deste ponto do documento. Todos os sujeitos indicados serão selecionados com um conjunto de competências que os fará aptos para resolverem os problemas anteriormente referidos.

Aos sujeitos, após resolverem os problemas, será pedido para responderem a um questionário a partir dos quais serão retiradas conclusões. Nesse questionário será pedido que os sujeitos indiquem o tempo que usaram para resolver cada problema resolvido, bem como o código correspondente à implementação da solução de cada um dos problemas. Após validação do que o código recebido efetivamente resolve os problemas indicados, os dados desse sujeito serão tidos em conta. De seguida os dados serão separados por tecnologia e usados para comparar a eficiência no desenvolvimento de aplicações móveis de cada tecnologia.

1.6 Impacto do COVID-19 na abordagem do estudo

No decorrer deste estudo houve uma pandemia causada pelo vírus COVID-19. Esta pandemia obrigou a fazer alterações à abordagem inicialmente planeada, uma vez que a abordagem inicial previa o contacto com sujeitos para a recolha de dados necessários ao estudo. Com o estado de confinamento presente no decorrer do estudo isso tornou-se impossível.

As alterações à abordagem inicialmente planeada foram feitas no sentido de ser possível obter os dados necessários ao estudo enquanto o estado de pandemia estava presente. Estas modificações à abordagem inicialmente pensada são apresentadas no Capítulo 5.

1.7 Estrutura

Este documento é dividido em vários capítulos. A organização e o conteúdo dos mesmo é a seguintes:

Capítulo 1 – Introdução: Neste capítulo é apresentada uma descrição inicial do contexto onde este estudo se insere e é feita, de forma resumida, uma primeira apresentação do problema. São também explicados os objetivos e resultados esperados do estudo bem a estrutura deste documento.

Capítulo 2 – Estado de arte e Análise de valor: Neste capítulo é feita uma contextualização dos problemas atuais no desenvolvimento de aplicações móveis. São também apresentadas as tecnologias atuais com mais relevância no desenvolvimento de aplicações móveis, sendo apresentados o estado de arte das mais relevantes para este estudo. A análise de valor que este estudo terá quer para o programador de aplicações móveis como para o utilizador final das mesmas é também referenciada deste capítulo.

Capítulo 3 – Abordagens existentes: Neste capítulo é apresentada a abordagem que vai ser implementada durante o decorrer do estudo. É também onde é apresentado o conceito de metodologia de recolha de dados no contexto deste estudo.

Capítulo 4 – Design da metodologia de recolha de dados: Neste capítulo é apresentado o design da metodologia a usar. É apresentada a explicação das métricas usadas para obter conclusões.

Capítulo 5 – Implementação da metodologia de recolha de dados: Neste capítulo é apresentado o processo de implementação da metodologia de recolha de dados. Bem como os problemas experienciados durante a sua implementação e as diferentes entre o design da metodologia e o resultado final.

Capítulo 6 – Apresentação dos dados recolhidos: Neste capítulo são apresentados os dados recolhidos usando a metodologia implementada. Estes são os dados que serão usados para retirar as conclusões deste estudo, relativas à eficiência de cada tecnologia.

Capítulo 7 – Conclusões: Neste capítulo são apresentadas as conclusões retiradas deste estudo e identificado se os objetivos deste estudo foram ou não atingidos.

2 Estado de arte e análise de valor

Este capítulo explica, de forma mais detalhada que o Capítulo 1, o contexto em que este estudo se insere. Entra em pormenores sobre o problema em questão, fazendo um levantamento do estado da arte das tecnologias relevantes ao estudo. É também apresentada a análise de valor deste estudo.

Ao longo deste capítulo vão ser dadas respostas às seguintes perguntas:

Quão grande é o uso de aplicações móveis da sociedade atual?

Qual a importância do desenvolvimento de aplicações móveis?

Quais as dificuldades da escolha da tecnologia a usar para o desenvolvimento de aplicações móveis?

Qual o valor deste estudo para entidades que pretendem desenvolver uma aplicação móvel?

Quais as tecnologias mais relevantes para o desenvolvimento de aplicações móveis?

2.1 Contexto

A população mundial está cada vez mais habituada ao uso de *smartphones* e, a acompanhar essa tendência, é possível verificar que o número de *downloads* de aplicações móveis tem sofrido um forte crescimento (Blair, 2020).

Com o crescimento do número de pessoas a utilizar um *smartphone*, torna-se cada vez mais evidente o importante papel que os mesmos possuem na vida de uma pessoa. Estudos publicados em 2018 pela EPRS, *European Parliamentary Research Service,* afirmam que o mercado à volta de aplicações móveis está longe de saturado. Estimativas apontam para que em 2021 o mercado esteja avaliado em 6.3 biliões de dólares (Szczepański, 2018). No mesmo estudo, é identificada como umas das principais barreiras para o crescimento do mercado de aplicações móveis, a falta de recursos humanos. O facto de existir escassez de programadores para o desenvolvimento de aplicações móveis faz com que empresas que desenvolvem este tipo de aplicações tentem aproveitar os seus recursos humanos de forma mais eficiente possível.

Com estes fatores é natural que empresas que procuram desenvolver aplicações móveis o tentem fazer da forma mais eficiente. Tendo isso em conta, durante os últimos anos têm surgido várias tecnologias que permitem um desenvolvimento mais rápido, como por exemplo o React Native. Esta afirmação é suportada no facto que, o interesse em tais tecnologias tem vindo a crescer ao longo dos últimos anos, como é possível observar na Figura 1, que mostra o interesse no termo de pesquisa “React Native”. Durante este documento os valores relativos ao interesse de termos de pesquisa no Google são apresentados numa escala de 0 a 100, valores

mais elevados correspondem a um interesse, ou popularidade, maior nesse termo. A avaliação da popularidade é feita avaliando a popularidade de um determinado termo relativamente a outros do mesmo contexto. O resultado é normalizado para um valor compreendido entre 0 e 100 (Google, 2020).

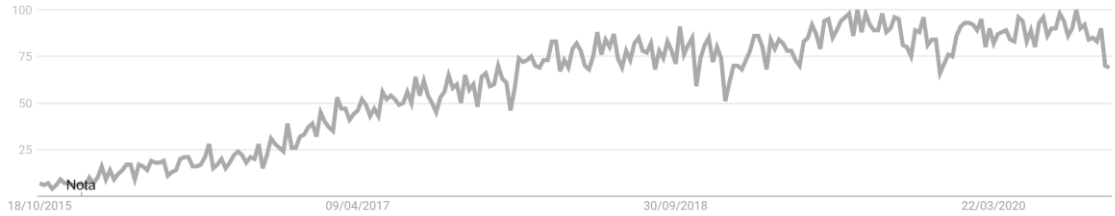

Figura 1 – Interesse no termo "React Native" no motor de pesquisa Google desde 2015 a nível mundial (GoogleTrends, 2020).

Em inquéritos feitos em 2018 na plataforma usada por programadores*, Stackoverflow,* são também evidentes o interesse e o desejo em trabalhar com tecnologias como o React Native ou o Xamarin, (Developer Survey Results, 2018).

Ambas as tecnologias, React Native e Xamarin, possibilitam o desenvolvimento de aplicações móveis usando apenas uma base de código comum e produzindo uma aplicação idêntica para os sistemas iOS e Android. Sendo apenas, por vezes, fazer algumas alterações na interface gráfica destinada a cada sistema. Esta possibilidade foi, por exemplo, uma das principais razões da aplicação "Walmart" ter sido desenvolvida em React Native (Keerti, 2016).

Em suma, o mercado para aplicações móveis nunca foi tão grande e programadores em todo o mundo têm cada vez mais interesse em usar tecnologias que possibilitem um desenvolvimento mais eficiente.

2.2 Problema

Com o surgimento de várias tecnologias com o objetivo de facilitar e acelerar o desenvolvimento de aplicações móveis, a escolha relativamente a qual tecnologia usar torna-se cada vez mais complicada.

Como referido na Secção 2.1 a eficiência no desenvolvimento de aplicações móveis é cada vez mais importante. Isso faz com que para a escolha da tecnologia a usar para o desenvolvimento de uma aplicação móvel seja importante ter em conta a eficiência de desenvolvimento que cada tecnologia permite.

O problema é que não existem dados comparativos da eficiência de desenvolvimento das tecnologias que atualmente existem. Fazendo com que seja impossível fazer uma escolha de que tecnologia usar no desenvolvimento da aplicação móvel usando a eficiência do desenvolvimento como critério.

2.3 Análise de valor

Este estudo incide sobre as diferentes tecnologias que visam a aumentar a eficiência de desenvolvimento de aplicações móveis para diferentes sistemas operativos. Como consequência a utilização dessas tecnologias, com ferramentas de desenvolvimento, diminui o custo de desenvolver uma aplicação móvel para os dois principais sistemas operativos de *smartphones,* Android e iOS.

As conclusões retiradas neste estudo vão permitir que um programador, que tenha interesse em desenvolver uma aplicação móvel, consiga escolher que tecnologia usar no seu projeto. Não só tendo em conta a sua preferência e requisitos, mas também a eficiência de desenvolvimento em cada uma delas. Isto é algo que vai ter benefícios não só para o programador que desenvolve as aplicações móveis, que vai conseguir diminuir custos, mas também para o utilizador final.

2.3.1 Valor para a programador que cria aplicações móveis

Este estudo vai disponibilizar métricas sobre a eficiência de desenvolvimento de aplicações móveis fazendo uso das diferentes tecnologias em estudo. Estes dados vão permitir que um programador, quando está a escolher que tecnologia usar para desenvolver uma aplicação, possa usar a eficiência de desenvolvimento como critério. Isto permite ao programador escolher a tecnologia que permite um desenvolvimento mais eficiente resultando num custo de desenvolvimento mais reduzido.

2.3.2 Valor para o utilizador de aplicações móveis

Quanta mais informação existir relativamente às tecnologias que permitem desenvolvimento de aplicações móveis para Android e iOS usando a mesma base de código, mais provável essas tecnologias são de ser usadas. Isto significa que quanta mais informação sobre estas tecnologias estiver disponível menor o número de aplicações móveis que estarão presentes apenas para um sistema operativo. Para o utilizador final isto significa que, independentemente do sistema operativo que o seu *smartphone* tiver, com o aumento da utilização destas tecnologias no desenvolvimento de aplicações será mais improvável encontrar aplicações que estejam disponíveis para, por exemplo, iOS e não para Android.

Este estudo não só vai aumentar a informação, relacionada com estas tecnologias, disponível como também vai permitir diminuir o custo que o programador. Uma vez que vai permitir ao programador usar a tecnologia com mais eficiente de desenvolvimento. Com isto é possível afirmar que este estudo contribuirá para aplicações mais baratas e um maior número aplicações disponíveis para ambos os sistemas, Android e iOS.

2.4 Estado da arte das tecnologias relevantes

Como identificado na Secção 2.2, o problema atual consiste no facto de não existir informação conclusiva que compare a eficiência de desenvolvimento de aplicações móveis das diferentes tecnologias.

Para obter essa informação é necessário conhecer as tecnologias, que têm o objetivo de aceleração do processo de desenvolvimento, com mais relevância.

À parte das tecnologias identificadas na Secção 2.1, React Native e Xamarin foram identificadas as tecnologias Ionic e Flutter. Foi feita uma listagem dos repositórios no GitHub, destinados ao desenvolvimento de aplicações móveis, e é possível verificar que os repositórios do Flutter e do Ionic são os dois com mais estrelas, (Github, 2020). Na Figura 2 é possível observar um gráfico de área empilhado, que mostra as tendências na pesquisa de cada uma das tecnologias identificadas, nos últimos 5 anos. É possível observar um aumento no interesse nos termos "React Native" e "Flutter" verificando que, no gráfico de área presente na Figura 2, a área dedicada a cada um desses tecnologias vai aumentando ao longo do tempo. Pela razão oposta é possível verificar que o interesse no termo "Ionic" vai dimiundo ao longo do tempo.

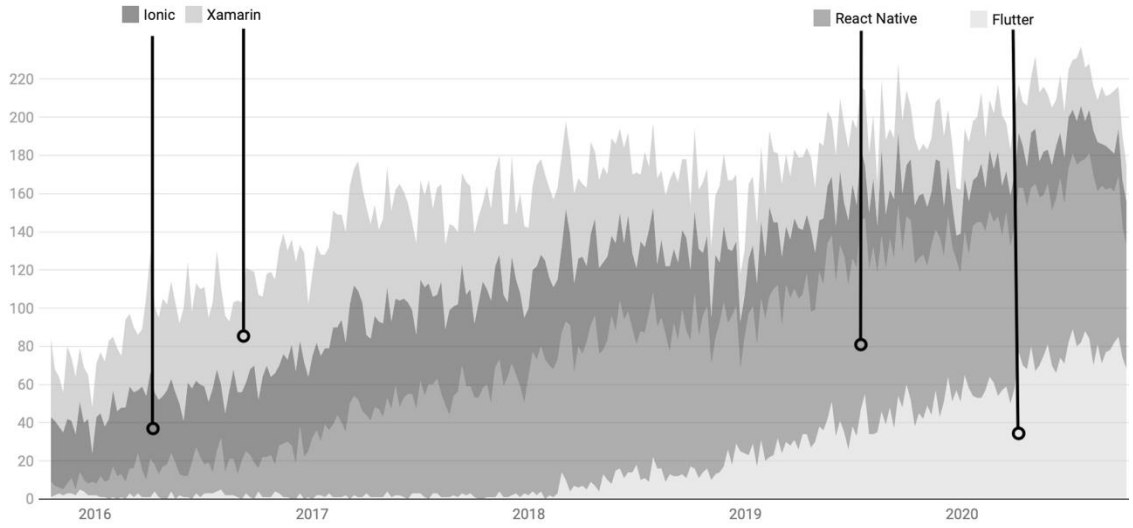

Figura 2 – Interesses nos termos de pesquisa das tecnologias identificadas ao longo dos últimos 5 anos, (GoogleTrends, 2020).

Na Tabela 1 é possível visualizar uma comparação das várias tecnologias identificadas, é possível verificar as tecnologias que produzem aplicações com elementos nativos, as que usam apenas uma base de código e as que possuem o seu código disponível ao publico.

Estas três características foram escolhidas como termo de comparação pois todas possuem uma importância para este contexto:

- **Elementos Nativos:** Numa aplicação móvel, destinada a um *smartphone* que usa um determinado sistema operativo, é esperado que estejam presentes elementos com o mesmo comportamento e aparência gráfica que os restantes elementos do sistema

operativo. Com as tecnologias que produzem elementos nativos é possível garantir que o utilizador final das aplicações apenas interage com elementos que tem o comportamento comum ao seu sistema operativo, mantendo a coerência. Tecnologias que não usam elementos nativos usam *web views*, para desenhar os seus elementos. *Web Views* são *web browsers* que ocupam o ecrã inteiro da aplicação, permitindo à aplicação apresentar elementos HTML, *HyperText Markup Language*. Os elementos apresentados tirando partido de *web views* não são elementos nativos.

- **Uma base de código:** Umas das grandes vantagens da utilização de tecnologias de desenvolvimento de aplicações móveis já enumeradas é que permite utilizar apenas uma base de código para produzir aplicações móveis para ambos os sistemas, Android e iOS. Esta característica é importante pois é um dos principais motivos que contribui para estas tecnologias serem eficientes no desenvolvimento de aplicações móveis.

- ***Open-source*:** Uma das preocupações quando uma tecnologia é usada por um programador é se a mesma vai continuar a ser mantida durante um período significativo de tempo. Tecnologias *open-source* permitem ao programador observar com transparência o progresso de *updates* e atividade do projeto em si. Sendo assim mais fácil avaliar se o projeto responsável por manter e atualizar a tecnologia continua a ser ativamente trabalhado.

É relevante que, para esta comparação, a característica de ter uma base de código comum não tem em conta pequenas modificações que possam ter de ser feitas na interface gráfica.

Tabela 1 – Comparação das tecnologias identificadas.

Tecnologia	Elementos nativos	Uma base de código	Open-source
Xamarin	Sim	Sim	Sim
Flutter	Sim	Sim	Sim
React Native	Sim	Sim	Sim
Ionic	Não	Sim	Sim

Apesar da sua inicial popularidade no GitHub, o Ionic tem vindo a perder relevância e popularidade, como é possível observar na Figura 2. Esse fator mais o facto de não produzir aplicações com elementos nativos, mas sim fazendo uso de *web views,* faz com que a mesma não se enquadre, comparativamente às outras tecnologias, neste estudo.

2.4.1 React Native

Lançado em 2015 pelo Facebook o React Native surgiu quando Jordan Walke, criador da biblioteca de JavaScript ReactJs, descobriu uma maneira de criar elementos nativos em iOs através de uma *thread* de JavaScript (Occhino, 2015). Utiliza elementos do ReactJs sendo ambos *open-source*. É possível verificar a estrutura base de uma aplicação de React Native na Figura 3.

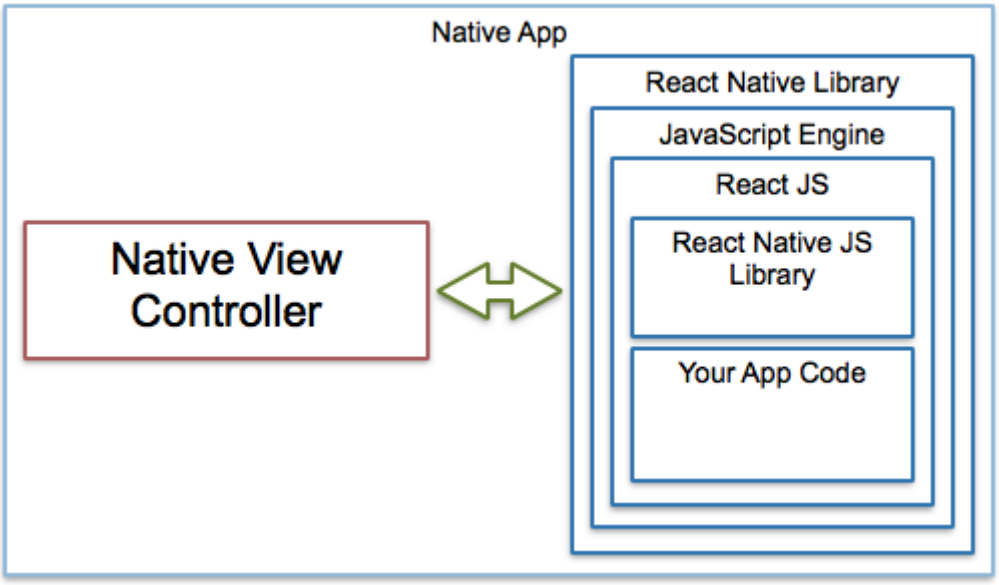

Figura 3 – Estrutura de uma aplicação React Native (Bershadskiy, 2015).

O código escrito com o objetivo de ser usado numa aplicação móvel desenvolvida com React Native é escrito usando a linguagem JavaScript, JS. É por esse motivo que existe, conforme exemplificado na Figura 3, um JavaScript *engine*. Esse JavaScript *engine*, executado numa *thread* dedicada, é responsável por executar todo o código JS que constitui a aplicação móvel desenvolvida. O código da aplicação móvel é constituído pelo código desenvolvido pelo programador bem como pelo código das bibliotecas de JS, React e React Native.

Num processo de execução diferente é executada a *thread* principal, comum a todas as aplicações móveis. Essa *thread* é responsável por mostrar ao utilizador os elementos da interface e processar os *inputs* realizados pelo mesmo.

A comunicação das duas *threads* é feita de forma assíncrona sendo assistida pela biblioteca core do React Native, identificada como "React Native Library" na Figura 3.

Uma aplicação desenvolvida em React Native tem uma estrutura baseada em componentes, que independentemente da sua implementação possuem algumas características em comum. No excerto de Código 1 está presente um exemplo de um componente, "Blink".

Todos os componentes são desenhados como elementos nativos do sistema operativo do dispositivo móvel. Os componentes geralmente possuem um método "*render*" (linha 18) que normalmente retorna uma estrutura de JSX, JavaScript XML. Todos os componentes possuem uma propriedade "*state*" (linha 16) que espelha o estado atual do componente. Sempre que a propriedade "*state*" é alterada força uma nova chamada do método "*render*", este comportamento faz com que a interface gráfica do React Native seja considerada reativa.

```
1 import React, { Component } from 'react';
2 import { Text, View } from 'react-native';
3
4 class Blink extends Component {
5
6       componentDidMount(){
7
8             setInterval(() => (
9                   this.setState(previousState => (
10                    { isShowingText: !previousState.isShowingText }
11                  ))
12                ), 1000);
13      }
14
15       //state object
16       state = { isShowingText: true };
17
18       render() {
19            if (!this.state.isShowingText) {
20                  return null;
21            }
22
23          return (
24            <Text>blinking!!</Text>
25          );
26        }
27 }
```

Código 1 - Exemplo de um componente React Native.

No excerto de Código 1, é possível observar a classe de um componente, neste caso o componente "Blink". É possível observar o método "*componentDidMount*" (linha 6, este método é comum a todos os componentes de React Native, e é chamado após o componente ser inicializado e antes de o mesmo ser apresentado ao utilizador.

Neste componente em específico no método "*componentDidMont*" (linha 6) é possível observar a criação de um timer que de segundo em segundo atualiza o "*state"* do componente. Em React Native o "*state*" de um componente guarda a informação, gerida pelo próprio componente, que tem influência no output do método "*render"* (linha 18). Sempre que o "*state*" do componente é atualizado o componente é novamente desenhado.

Por último é possível verificar a existência de um método "*render*" (linha 18), método responsável por desenhar o componente. Neste exemplo o componente desenha um elemento "*Text*" com a *label* "blinking!!" consoante a variável "*isShowingText*" do "*state*" do componente está ou não a *true (linha 19).*

2.4.2 Flutter

O Flutter é uma nova tecnologia, apresentada pela Google como o futuro do desenvolvimento de aplicações móveis. A primeira versão estável foi lançada em dezembro de 2018. Aplicações desenvolvidas com o Flutter são escritas utilizando a linguagem de programação Dart, também criada pela Google.

À semelhança do React Native uma aplicação em Flutter é composta por vários componentes, neste caso esses componentes chamam-se *widgets*. O Flutter possui uma biblioteca de *widgets* simples, e é possível criar *widgets* mais complexos usando composição dos disponibilizados pela Google. Por sua vez esses *widgets* originaram elementos nativos do sistema operativo onde a aplicação está a correr. É possível observar esta estrutura do Flutter na Figura 4.

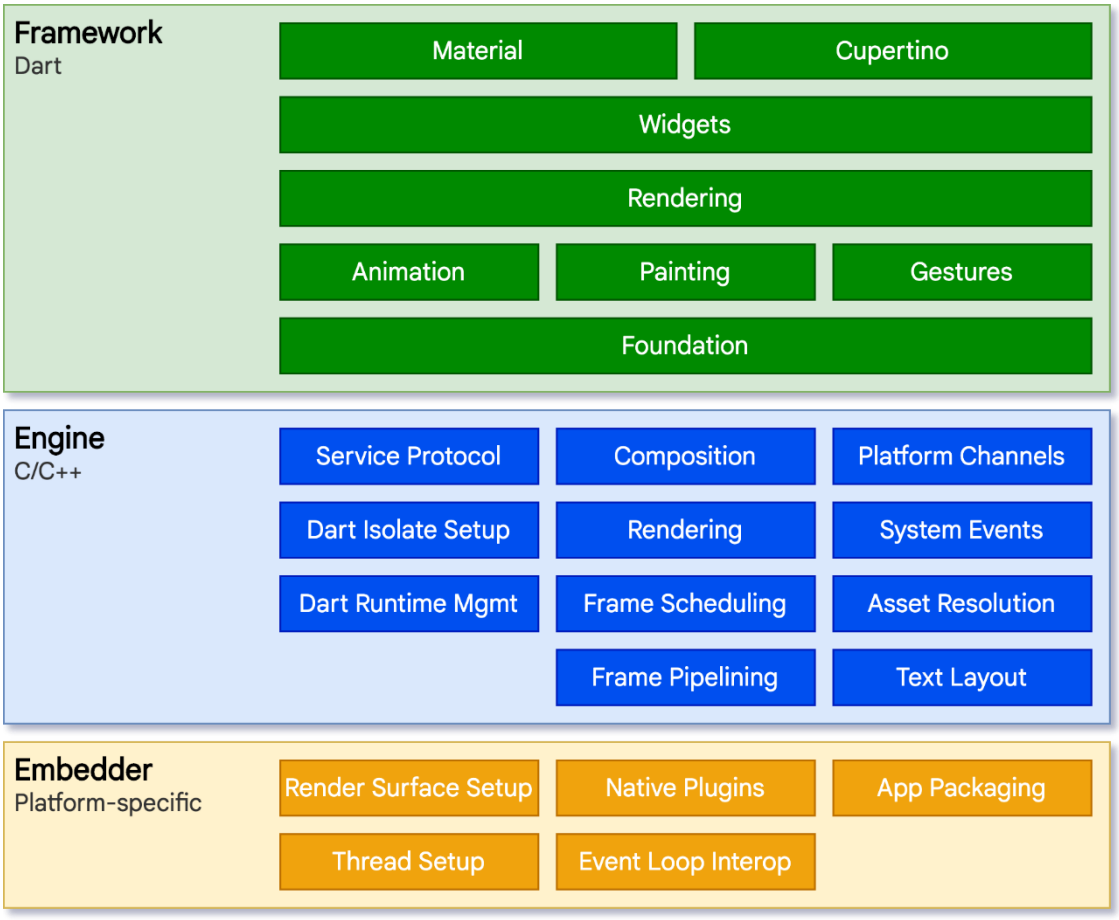

Figura 4 – Arquitetura do Flutter de alto nível. (Flutter architectural overview, 2020)

A Figura 4 apresenta a arquitetura da tecnologia Flutter, composta por várias camadas. À semelhança do React Native, existe uma camada, o *embedder,* responsável por permitir e coordenar o acesso do *engine* do Flutter aos serviços e recursos do sistema. Para cada sistema operativo existe um *embedder* específico desenvolvido numa linguagem de programação usada nesse sistema operativo, Java e C++ para Android e, Objective-C e Objective-C++ para iOS.

O *engine* do Flutter, desenvolvido maioritariamente C ++, é responsável por executar e compilar código Dart. Ao usar a *framework* Dart do Flutter o programador ganha acesso a um conjunto de *widgets* que pode usar para criar o *layout* da sua aplicação móvel.

Inspirado no React Native, o Flutter possui também uma interface gráfica reativa. Significando que a *framework* atualiza os elementos da interface consoante mudanças no estado da aplicação (Flutter architectural overview, 2020).

```
1       import 'package:flutter/material.dart';
2
3       void main() => runApp(MyApp());
4
5       class MyApp extends StatelessWidget {
6         @override
7         Widget build(BuildContext context) {
8           return MaterialApp(
9             title: 'Welcome to Flutter',
10                  home: Scaffold(
11                    appBar: AppBar(
12                      title: Text('Welcome to Flutter'),
12                    ),
13                    body: Center(
14                      child: Text('Hello World'),
15                    ),
16                  ),
17             );
18        }
19      }
```

Código 2 - Exemplo de uma aplicação Flutter.

No excerto de Código 2 é notável a arquitetura do Flutter, e a anteriormente referida composição por *widgets*. É possível verificar a existência de um método "*build*" (linha 7) dentro da classe "*MyApp*" (linha 5) que retorna um objeto da classe "*Widget*". Este *widget* é o elemento que vai ser desenhado.

No início desta subsecção foi referido que é possível a criação de *widgets* por composição de outros *widgets*. No excerto de Código 2 é possível observar essa propriedade, o método "*build*" (linha 7) está a retornar um *widget* do tipo "*MaterialApp*" que recebe como parâmetros outros *widgets* (linha 10) usados no seu processo de desenho.

2.4.3 Xamarin

O Xamarin surgiu em 2013, quando uma empresa com o mesmo nome, Xamarin, lançou um conjunto de produtos para o mercado. Esses produtos tornaram possível a criação de aplicações móveis para Android, Windows e iOS, usando o Visual Studio (AltexSoft, 2019). Em 2016 a mesma acabaria por ser adquirida pela Microsoft, que acabou por tornar a plataforma *open-source.*

Para o desenvolvimento de aplicações móveis utilizando o Xamarin a linguagem de programação C# é usada, tirando partido da *framework* .NET. Como referido na Secção 2.4 o Xamarin também permite a criação de aplicações nativas usando o mesmo código para ambos os sistemas operativos. Na Figura 5 é possível observar um diagrama que demonstra como o Xamarin está organizado.

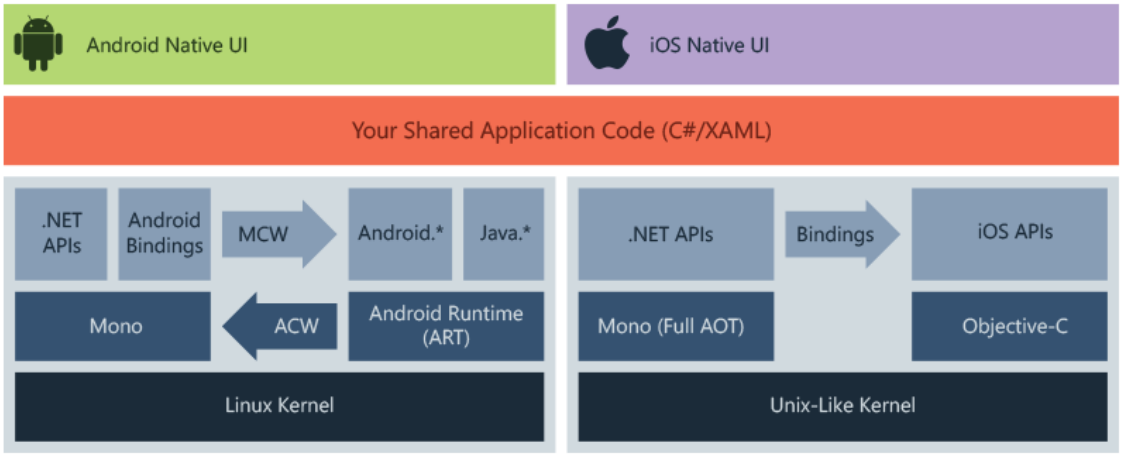

Figura 5 –Organização dos diferentes elementos que constituem o Xamarin. (Microsoft, 2020)

O Xamarin permite a criação de uma camada de código aplicacional, desenvolvido em C#, que é partilhado pelos dois sistemas, Android e iOS. Apesar de o Xamarin permitir a criação de uma interface nativa para cada sistema, é possível usar a *framework* Xamarin.Forms para utilizar a mesma base de código para gerar duas interfaces gráficas nativas para cada sistema.

Conforme apresentado na Figura 5, o Xamarin possui várias .NET *APIs*, como por exemplo Xamarin.Essentials, que cedem acesso aos serviços e recursos dos sistemas, Android e iOS, diretamente a partir do código C#. O código C# desenvolvido pelo programado é depois compilado e executado sob a forma de *assemblies* nativos para cada um dos sistemas.

O Xamarin usa XAML, *Extensible Application Markup Language,* para declarar a interface gráfica e os seus respetivos componentes. E C# para controlar o *lifecycle* dos seus componentes. No excerto de Código 3 e 4 é possível verificar ambas as situações.

```
1       <Button Text="Click Me" Clicked="Handle_Clicked" />
```

Código 3 - Exemplo de um botão declarado em xaml. (Microsoft, 2020)

```
1       int count = 0;
2       void Handle_Clicked(object sender, System.EventArgs e)
3       {
4               count++;
5               ((Button)sender).Text = $"You clicked {count} times.";
6       }
```

Código 4 - Exemplo da função *handler* do clique no botão. (Microsoft, 2020)

No excerto de Código 3 é possível verificar a declaração de um botão, usando xaml, no qual está presente a *label*, identifica no código como "*Text*", "*Click Me*" (linha 1). É também atribuído o método "*Handle_Clicked*" (linha 1) quando o botão passa ao estado *clicked*, significando que esse mesmo método será executado quando o botão é pressionado.

O método "*Handle_Clicked*" está presente no excerto de Código 4 (linha 2), onde é possível observar a incrementação de uma variável no tipo inteiro aquando da execução do mesmo. A mesma variável é usada na construção de uma String pela qual o texto no botão é substituído (linha 5). No fim resulta um botão que sempre que é pressionado tem a sua *label* atualizada com o número de vezes que o mesmo foi premido.

2.5 Sumário

Com os estudos efetuados neste capítulo é agora possível responder às perguntas levantadas.

Quão grande é o uso de aplicações móveis da sociedade atual?

Como referido na secção 2.1 o número de *downloads* de aplicações móveis tem sofrido um forte crescimento, com este crescimento está previso para 2021 que o mercado à volta das aplicações móveis chegue a valer de 6.3 biliões de dólares. Estes factos ilustram o atual uso em massa de *smartphones* e aplicações móveis pela população mundial.

Qual a importância do desenvolvimento de aplicações móveis?

No estudo apresentado na Secção 2.1 é referenciado como barreira ao crescimento do mercado de aplicações móveis a falta de recursos humanos. Com isto é natural que empresas tentem tirar o máximo partido possível dos seus programadores, tentando que os mesmo tenham o desenvolvimento mais eficiente possível. Dando assim muita relevância ao processo em si de desenvolvimento das aplicações móveis.

Quais as dificuldades da escolha da tecnologia a usar para o desenvolvimento de aplicações móveis?

Quando um programador tem de escolher que tecnologia utilizar para o desenvolvimento de aplicações móveis a escolha torna-se difícil quando a eficiência de desenvolvimento é critério para a mesma. Pois, como referido na Secção 2.2, não existe informação disponível de modo a ser possível comparar tecnologias.

Qual o valor deste estudo para programadores que pretendem desenvolver uma aplicação móvel?

Este estudo possibilita aos programadores escolherem tecnologias para o desenvolvimento de aplicações móveis, tendo como critério a eficiência de desenvolvimento cada uma das tecnologias permite, conforme referido na Subsecção 2.3.1.

Quais as tecnologias mais relevantes para o desenvolvimento de aplicações móveis de forma eficiente?

Do estudo feito às tecnologias existentes, Secção 2.4, as que mostram mais potencial para um desenvolvimento de aplicações móveis de forma mais eficiente foram o React Native, Xamarin e Flutter.

3 Abordagens existentes

Neste capítulo será explorada detalhadamente uma das abordagens existentes mais relevantes para estudos de comparação de *performance*.

Em estudos dedicados à *performance* é norma a utilização de metodologias que façam uso de métricas. Para tal uma das abordagens comuns é a identificação de um *key performance indicador* e a utilização desse indicador como meio de comparação entre os vários alvos de teste.

Ao longo deste capítulo vão ser dadas respostas às seguintes perguntas:

O que são *Key performance Indicators*?

Qual as possíveis metodologias adequadas para a recolha dos dados necessários para este estudo?

Das metodologias de recolha de dados identificadas, qual melhor se adequa a este estudo?

3.1 Key performance indicators

Key performance indicators, ou KPIs, são indicadores quantitativos usados para concluir sobre a performance de uma entidade. Os mesmos são maioritariamente usados em empresas, e permitem aos seus gestores fazer uma ligação entre indicadores e a *performance* da organização.

Para que a ligação direta entre um KPI e a *performance* de uma organização faça sentido é necessário que esses indicadores sejam selecionados tendo em conta os objetivos da mesma. Com isto os KPIs variam de forma abundante. Sendo que empresas de áreas distintas podem ter KPIs idênticos, se tiverem objetivos semelhantes, e empresas que atuam na mesma área ter indicadores divergentes, se os seus objetivos assim o ditarem (Parmenter, 2015).

Alguns exemplos de KPIs para uma empresa portuguesa que vende livros *online*, que tem como objetivo aumentar a sua presença no mercado espanhol, poderiam ser:

- Número de utilizadores registados com morada espanhola;
- Número de emails capturados para *newsletter*;
- Número total de vendas individuais para clientes com morada espanhola;

Após a identificação de KPIs é importante identificar objetivos, quantitativos, para cada um. Com estes indicadores, seria possível tirar conclusões relacionadas com o objetivo identificado pela organização.

É de notar que, neste exemplo, os KPIs não fazem nenhuma referência a lucros. Logo se a empresa tivesse prejuízos a vender para Espanha, mas os KPIs estivessem em linha com os objetivos o cenário seria de sucesso.

3.1.1 KPIs em estudos

Da mesma maneira que é possível identificar KPIs ligados aos objetivos de uma organização, é também possível identificar KPIs relacionados com um objetivo de estudo.

Utilizando como referência este estudo, que tem como objetivo tirar identificar a tecnologia que permite um desenvolvimento de aplicações móveis mais eficiente. É possível atribuir a cada tecnologia o objetivo de ser a mais eficiente, ou que quando usa permite ao programador desenvolver aplicações em menor tempo. Neste caso um KPI válido seria um valor quantitativo que correspondia à quantidade de trabalho feito, por hora.

Como o estudo se baseia numa comparação entre diversas tecnologias o objetivo atribuído ao KPI não seria um valor alvo, mas sim que o mesmo fosse superior ao KPI das outras tecnologias.

3.2 Metodologia de recolha de dados

Como apresentado na Secção 1.3, é esperado que deste estudo resulte uma metodologia de recolha de dados relativos à eficiência de desenvolvimento de aplicações móveis para cada tecnologia. Para tal, seguindo a abordagem descrita na Secção 1.5, será necessário desenvolver um conjunto de problemas a resolver por sujeitos, que por sua vez irão fornecer dados a ser usados na comparação da eficiência de desenvolvimento das diferentes tecnologias.

Porém existe também a possibilidade de contactar empresas na área de desenvolvimento de aplicações móveis, que usem as tecnologias em estudo, e tentar obter dados relativos à eficiência do desenvolvimento. Apesar desta abordagem não ser a inicialmente planeada na Secção 1.5, a implementação da mesma poderá trazer vantagens, nomeadamente um maior número de dados.

Sendo assim neste momento existem duas possibilidades quanto à metodologia de recolha de dados:

- **Recolha de dados de sujeitos -** Definição de um conjunto de problemas a resolver por um grupo de sujeitos usando as várias tecnologias em estudo. Após a implementação das soluções dos problemas propostos os sujeitos responderiam a um inquérito que permitia recolher informação sobre a sua experiência de desenvolvimento e, depois de analisada essa informação, sobre a sua eficiência.
- **Recolha de dados de empresas -** Contactar com empresas da área de desenvolvimento de aplicações móveis e que utilizem as tecnologias com o intuito de as mesmas disponibilizarem dados sobre a experiência de desenvolvimento nas diversas

tecnologias. Seria possível pedir a essas empresas lista de tarefas executadas usando as tecnologias em estudo e os respetivos tempos necessários para as completar.

3.2.1 Analytical Hierarchy Process

Para fazer uma escolha fundamenta à cerca de que metodologia seguir foi feita uma análise hierárquica, *analytical hierarchy* process, ou AHP (Saaty, 2008). A análise feita nesta secção segue o documento lançado em 2008 por Thomas L. Saaty referente à análise AHP (Saaty, 2008).

Seguindo a análise AHP é inicialmente necessário fazer um levantamento dos critérios que serão usados para fazer a escolha entre as duas metodologias de recolha de dados.

- **Fiabilidade de dados -** A fiabilidade dos dados é essencial. Se os dados utilizados para tirar as conclusões deste estudo não forem verdadeiros ou contiverem inconsistências é impossível tirar conclusões com alguma validade científica.
  A recolha de dados de sujeitos permite instruir os sujeitos antes da recolha dos dados de modo a minimizar os erros na recolha dos necessários. O facto de haver garantia que os dados são recolhidos sujeito a sujeito também é benéfico, uma vez que permite a identificação de *outliers* mais facilmente.
  A recolha de dados de empresas é necessário acreditar nos métodos de recolha de dados que as empresas implementam, bem como na veracidade dos mesmos.
- **Dados disponíveis –** Os tipos de dados disponíveis têm uma importância bastante elevada, do mesmo nível que a fiabilidade dos dados, uma vez que sem os dados necessários para avaliação da eficiência de cada tecnologia é impossível dar resposta aos objetivos deste estudo.
  Na recolha de dados de sujeitos existe o controlo total sobre os dados recolhidas, uma vez que os sujeitos podem ser instruídos nos dados que devem recolher aquando da implementação da solução dos problemas propostos.
  Na recolha de dados de empresas é impossível prever que tipo de dados que uma empresa tem disponíveis sobre o trabalho realizados dos seus colaboradores.
- **Quantidade de dados -** A quantidade de dados possibilita retirar conclusões estatisticamente mais fundamentadas e estatisticamente relevantes. Não tem tanta importância como os critérios “Fiabilidade de dados” e “Dados disponíveis” uma vez que quantidades menores de dados podem na mesma resultar em conclusões válidas, ao contrário dos outros dois critérios.
  A recolha de dados de empresas tem uma grande vantagem neste critério, uma vez que empresas possuem mais dados que aqueles que é viável retirar de um grupo de sujeitos.
- **Diversificação de dados -** Sendo este estudo um estudo comparativo, é importante ter uma quantidade de dados semelhante para as tecnologias em estudo. Isto seria algo fácil de atingir com a recolha de dados de sujeitos. Este critério tem ligeiramente mais importância que a quantidade de dados uma vez que é essencial para que seja possível recolher dados de todas as tecnologias.

A recolha de dados de empresas tem desvantagem uma vez que existem tecnologias que são menos usadas o que poderia fazer com que fosse mais difícil obter dados sobre o desenvolvimento das mesmas dentro de empresas.

3.2.1.1 Decisão

Segundo a análise AHP para chegar à metodologia de recolha de dados que melhor se adequa o próximo passo é criar uma matriz de comparações de importâncias dos parâmetros.

Tendo em conta a escala fundamental apresentada por Saaty (Saaty, 2008), é necessário apresentar valor avaliar cada critério quanto à sua importância, relativamente aos outros critérios. A escala fundamental de Saaty varia de 1 a 9 sendo o valor 1 correspondente a critérios de igual importância e o valor 9 a um critério com extrema importância, comparativamente. Com isso foi elaborada a Tabela 2 que ilustra os valores de importância dos diversos critérios comparados com os restantes, e utiliza as comparações das importâncias dos critérios apresentados nesta subsecção.

Pare concluir os dados presentes na Tabela 2 foi feita a seguinte análise comparativa:

- Os critérios relativos à fiabilidade de dados e aos dados disponíveis foram identificados como critérios com o mesmo grau de importância. Logo na Tabela 2 foram ambas comparadas usando o valor 1, que sugere que ambos os critérios têm a mesma importância.
- Os critérios relativos à quantidade de dados e à diversificação de dados foram critérios identificados com uma importância mais baixa que a fiabilidade ou diversificação de dados. Uma vez que a sua oscilação não impede por completo a retirada de conclusões deste estudo. Porém entre os critérios de quantidade e diversificação de dados o último é o mais importante. Uma vez que uma menor diversificação de dados pode, ou não, fazer com que não seja possível retirar conclusões para uma determinada tecnologia. Com isto em mente foi atribuída, na Tabela 2, uma importância ao critério de diversificação de dados, comparativamente à quantidade de dados, com valor de 2.
- Os critérios relativos à fiabilidade de dados e aos dados disponíveis têm uma importância ligeiramente superior à diversificação de dados e significativamente superior à quantidade de dados. Na Tabela 2, foram usados os valores 2 e 4, para descrever a importância dos critérios relativos à fiabilidade de dados e dados disponíveis, comparativamente com os critérios de diversificação de dados e quantidade de dados, respetivamente.

Tabela 2 – Matriz de comparações.

Critério	Fiabilidade dos dados	Dados disponíveis	Quantidade de dados	Diversificação de dados
Fiabilidade dos dados	1	1	4	2
Dados disponíveis	1	1	4	2
Quantidade de dados	1/4	1/4	1	1/2
Diversificação de dados	1/2	1/2	2	1

Seguindo com a análise AHP, é agora necessário a normalização da matriz Tabela 3.

Tabela 3 – Matriz de comparações normalizada.

Critério	Fiabilidade dos dados	Dados disponíveis	Quantidade de dados	Diversificação de dados
Fiabilidade dos dados	4/11	4/11	4/11	4/11
Dados disponíveis	4/11	4/11	4/11	4/11
Quantidade de dados	1/11	1/11	1/11	1/11
Diversificação de dados	2/11	2/11	2/11	2/11

Através da matriz normalizada é possível obter o vetor de prioridades:

$$\begin{bmatrix} 4/11 \\ 4/11 \\ 1/11 \\ 2/11 \end{bmatrix}$$

De acordo com a análise AHP é agora necessário calcular a razão de consistência, RC. O RC é um rácio que permite validar que os julgamentos de importâncias nos diferentes critérios foram consistentes. Para calcular o RC é necessário primeiro calcular o maior valor próprio da matriz de comparações normalizada.

Considerando que:

$$[Ax = \lambda max\ x]$$

Equação 1

Onde X é o próprio vetor e A é a matriz de comparações.

$$\begin{bmatrix} 1 & 1 & 4 & 2 \\ 1 & 1 & 4 & 2 \\ 1/4 & 1/4 & 1 & 1/2 \\ 1/2 & 1/2 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 4/11 \\ 4/11 \\ 1/11 \\ 2/11 \end{bmatrix} = \lambda max \begin{bmatrix} 4/11 \\ 4/11 \\ 1/11 \\ 2/11 \end{bmatrix}$$

$$\Leftrightarrow$$

$$\begin{bmatrix} 1.455 \\ 1.455 \\ 0.364 \\ 0.727 \end{bmatrix} = \lambda \ max \begin{bmatrix} 4/11 \\ 4/11 \\ 1/11 \\ 2/11 \end{bmatrix}$$

$$\Leftrightarrow$$

$$\lambda max = 4.003$$

Após o cálculo do maior valor próprio, e seguindo a análise AHP, é possível chegar RC usando a Equação 2.

$$RC=((\lambda max-n)/(n-1)) / IR$$

Equação 2

Onde IR corresponde ao índice aleatório que é referente ao número de comparações de critério efetuadas, neste caso IR = 0.9, (Saaty, 2008) .

$$RC=((4.003-4)/(4-1))/0.9= 0.001$$

Como a RC é inferior a 0.1 é possível concluir que os valores das prioridades relativas são consistentes.

É agora necessário a construção das matrizes de comparação para cada critério. Tendo em conta as observações feitas, em relação a cada critério e cada possibilidade, no início desta secção. As matrizes apresentadas já se encontram normalizadas.

É agora necessária a construção da matriz de comparação paritária para cada critério considerando as duas alterativas em estudo. Nas Tabela 4Tabela 5,Tabela 6Tabela 7 é possível verificar a avaliação a cada uma das alternativas, em comparação com a outra, para cada um dos critérios.

No fim resulta um valor de prioridade relativa por alternativa e por critério. Entende-se por prioridade relativa a avaliação ou vantagem relativa que uma alterativa tem sobre as restantes. A soma das prioridades relativa relativas de todas as alterativas por critério tem de ser igual a 1.

Tabela 4 – Matriz de comparação paritária da fiabilidade dos dados.

Estudo	Recolha de dados de sujeitos	Recolha de dados de empresas	Prioridade relativa
Recolha de dados de sujeitos	1	3/4	0.75
Recolha de dados de empresas	1/4	1	0.25

Tabela 5 - Matriz de comparação paritária das métricas recolhidas.

Estudo	Recolha de dados de sujeitos	Recolha de dados de empresas	Prioridade relativa
Recolha de dados de sujeitos	1	3/4	0.75
Recolha de dados de empresas	1/4	1	0.25

Tabela 6 – Matriz de comparação paritária da quantidade de dados.

Estudo	Recolha de dados de sujeitos	Recolha de dados de empresas	Prioridade relativa
Recolha de dados de sujeitos	1	1/6	0.17
Recolha de dados de empresas	5/6	1	0.83

Tabela 7 – Matriz de comparação paritária da diversidade dos dados.

Estudo	Recolha de dados de sujeitos	Recolha de dados de empresas	Prioridade relativa
Recolha de dados de sujeitos	1	3/5	0.6
Recolha de dados de empresas	2/5	1	0.4

Com os dados das prioridades relativas de cata alternativa é possível criar a matriz de prioridades.

$$\begin{bmatrix} 0.75 & 0.75 & 0.17 & 0.6 \\ 0.25 & 0.25 & 0.83 & 0.4 \end{bmatrix}$$

O último passo é multiplicar a matriz de prioridades pelo vetor próprio, que possui o peso de cada critério:

Prioridade composta= MP * X

Equação 3

Onde MP é a matriz de prioridades e X é o vetor próprio.

$$\begin{bmatrix} 0.75 & 0.75 & 0.17 & 0.6 \\ 0.25 & 0.25 & 0.83 & 0.4 \end{bmatrix} * \begin{bmatrix} 4/11 \\ 4/11 \\ 1/11 \\ 2/11 \end{bmatrix} = \begin{bmatrix} 0.67 \\ 0.33 \end{bmatrix}$$

Após a realização da análise AHP é possível concluir que a recolha de dados de sujeitos, é metodologia de recolha de dados mais adequada para este estudo.

3.3 Sumário

Neste capítulo foi feito um estudo sobre o método de classificação de *performance* recorrendo a KPIs. Foi também feito um levantamento e comparação das metodologias de recolha de dados possíveis de utilizar no decorrer deste estudo.

Com os estudos efetuados neste capítulo é agora possível responder às perguntas levantadas.

O que são *Key performance Indicators*?

KPIs são, utilizando a tradução direta, indicadores chave de *performance*. No contexto deste estudo um KPI é uma métrica que pode, de maneira direta, descrever a eficiência de desenvolvimento de aplicações móveis numa determinada tecnologia, este tema foi abordado na Secção 3.1.

Qual as possíveis metodologias adequadas para a recolha dos dados necessários para este estudo?

Do estudo realizado na Secção 3.2 foram identificadas duas metodologias possível de utilizar para a recolha de dados, recolha de dados de sujeitos e recolha de dados de empresas.

Das metodologias de recolha de dados identificadas, qual melhor se adequa a este estudo?

Na Secção 3.2 foi possível concluir, através de uma análise AHP, que a metodologia de recolha de dados que melhor se adequa a este estudo é a metodologia de recolha de dados de sujeitos.

4 Desenho da metodologia recolha de dados

Neste capítulo vai ser apresentada a proposta do desenho da metodologia que permite a recolha de dados correspondentes a um KPI, ainda a definir. O objetivo é que esta metodologia, depois de construída e aplicada, permita recolher dados relacionados com a eficiência de desenvolvimento das tecnologias em estudo.

Ao longo deste capítulo vão ser dadas respostas às seguintes perguntas:

Que métrica ou métricas serão usadas como KPI?

Que tecnologias vão ser estudadas?

Como é que os dados relativos à métrica escolhida serão recolhidos?

4.1 Seleção das tecnologias

De modo a selecionar as tecnologias mais relevantes, foi feito um estudo ao estado da arte no desenvolvimento de aplicações, para os dois sistemas operativos de *smartphones* mais populares, Android e iOS. Desse estudo, presente na Secção 2.1 dedicado ao contexto e análise do problema, surgiu a conclusão de que, das tecnologias presentes no mercado, as que melhores se enquadram no contexto deste estudo são:

- React Native;
- Flutter;
- Xamarin.

4.2 Teste da eficiência de desenvolvimento

Nesta secção será apresentada e justificada a métrica a usar como KPI da eficiência de desenvolvimento de aplicações nas tecnologias em estudo, referidas na Secção 4.1.

Neste estudo comparativo entre tecnologias, é necessário identificar um KPI que consiga representar a quantidade de trabalho efetuado numa tecnologia num período de tempo. Para ser possível comparar as diferentes tecnologias, quanto à sua eficiência de desenvolvimento, é importante que o KPI a usar seja uma métrica expressa de modo quantitativo. Podendo assim ser facilmente identificada a tecnologia mais eficiente no desenvolvimento de aplicações móveis.

Depois de escolhida e metodologia de recolha de dados na Secção 3.2, recolha de dados de sujeito, é necessário fazer um levantamento dos problemas a resolver por esses sujeitos. Da resolução desses problemas vão ser retirados dados relativos aos KPIs definidos. Isto vai de encontro à abordagem inicialmente planeada na Secção 1.5.

4.2.1 KPIs

Depois de identificados os requisitos que o KPI deve cumprir, de modo a ser de facto relevante a este estudo, foi um levantamento de indicadores geralmente usados em testes relacionados com a eficiência de desenvolvimento de software, (Linda M. Laird, 2006):

- *Source lines of code;*
- *Object points;*
- *Function points;*
- *Use case points.*

Programadores diferentes desenvolvem o seu código de diferentes maneiras. Podendo desenvolver código mais verboso, o que faz com que o número de linhas de código, *source lines of code,* seja mais elevado apesar do *use case* implementado ser o mesmo. Isto faz com que a métrica *source lines of code* não seja aplicável no contexto deste estudo. Pelo menos motivos foram também descartadas as métricas *Object points* e *Function points,* que usam, respetivamente, o número de objetos e o número de funções presentes no código.

Com este levantamento de métricas foi possível concluir que a métrica que tem mais relevância para este estudo é o *use case points*, UCPs. Pois é a única que permite avaliar o trabalho necessário para resolver uma tarefa, neste caso um *use case*, independentemente do programador que a executa.

4.2.1.1 Use Case Points

O UCP é uma métrica que permite expressar a dimensão ou complexidade de uma funcionalidade ou projeto de software a desenvolver. Para calcular a complexidade, este método baseia-se no número de interações entre o ator de um *use case* e o sistema, como é possível observar na Tabela 8. Um *use case* é considerado mais complexo quanto mais vezes o ator tiver de interagir com o sistema.

A par com o número de interações entre o ator de um *use case* diferentes tipos de atores do, *use case,* também têm influência na complexidade do mesmo, como é possível verificar na Tabela 9.

Os valores presentes na Tabela 8 e Tabela 9 são valores já previamente definidos e foram retirados do estudo publicado por Gautam Banerjee (Banerjee, 2004).

É ainda possível adicionar outros modificadores de complexidade, como por exemplo, ajustes à complexidade baseados na motivação e outros fatores relacionados com o programador (Banerjee, 2004). Para este estudo, estes fatores não vão ter sidos em conta.

Tabela 8- Fator de complexidade do caso de uso.

Classificação	Número de interações	Peso
Simples	1 – 3	5
Médio	4 – 7	10
Complexo	8 +	15

Tabela 9 - Fator de complexidade do ator.

Classificação	Tipo de ator	Peso
Simples	Sistemas a interagirem através de uma API.	1
Médio	Sistemas a interagirem através de protocolos de comunicação.	2
Complexo	Humano a usar uma interface gráfica.	3

Com isto, para este estudo a fórmula presente na Equação 4 será usada para calcular os UCPs de cada caso de teste. (Banerjee, 2004).

$$UCP = PUC + PAC$$

Equação 4

Onde PUC é o peso associado à classificação do use case, e PAC o peso associado à classificação do ator.

Desta fórmula resultará um valor em UCPs que corresponde à complexidade do *use case* em questão. Valores de UCPs superiores estarão associados a casos de uso mais complexos.

Se os UCPs forem usados como indicadores da complexidade de uma determinada tarefa e forem divididos pelo número de horas necessárias para completar cada tarefa é possível obter o número de UCPs desenvolvidos por hora, UCPs/h.

Se esse cálculo for feito para cada uma das tecnologias é possíveol obter KPIs, sob a forma de UCPs/h, que permitem comparar e eficiência do desenvolvimento de aplicações móveis em cada uma das tecnologias.

4.2.2 Casos de teste

Para testar a eficiência de desenvolvimento das tecnologias selecionadas, foi criada uma bateria de testes, composta por uma série de problemas a resolver. A listagem de problemas foi desenvolvida de modo a replicar tarefas ou casos de uso que estão presentes em grande parte das aplicações móveis presentes no mercado. Conforme referido na Secção 1.5 estes problemas,

ou tarefas, serão distribuídos por vários sujeitos que irão os resolver usando diferentes tecnologias. Dessa resolução serão retirados dados que permitam o cálculo dos UCPs/h.

Da lista de aplicações mais populares de 2019 (Jones, 2020), foi feito um levantamento de *use cases* simples que estão presentes na maior parte das aplicações. Esses *uses cases* serão usados na criação de uma listagem de problemas a resolver pelos sujeitos.

Neste contexto entende-se por *use case* simples uma funcionalidade que está presente numa aplicação móvel, mas que não define a mesma. Por exemplo uma aplicação móvel com capacidade de abrir a câmara não corresponde obrigatoriamente a uma aplicação de fotografia ou *social media*. Essa funcionalidade pode estar disponível em outro tipo de aplicações, como leitura de QR *codes* ou leitura de código de barras de produtos. O facto de apenas serem usados *use cases* simples para a criação da lista de problemas a resolver, faz com o período de implementação das soluções aos problemas não seja demasiado longo.

Se fossem utilizados *use cases* muito específicos resultariam tarefas muito mais extensas e complexas, o que faria com que a aceitação deste estudo por sujeitos fosse inferior. Por esse mesmo motivo não serão selecionados *use cases* que exijam algum equipamento ou serviço externo para além do seu computador, como por exemplo o uso de equipamentos *Bluetooth.*

Tendo em conta a informação e critérios apresentados nesta subsecção foi possível desenvolver uma listagem de *use cases* simples que existem na maior parte das aplicações móveis mais populares. Esses *use cases* serão utilizados como os problemas a resolver pelos sujeitos que farão parte desde estudo, conforme a abordagem definida na Secção 1.5. Essa listagem de problemas a resolver pelos sujeitos é apresentada na Tabela 10.

Para além dos *use cases* definidos irá também ser pedido aos sujeitos que participem neste estudo que completem as tarefas, consideradas importantes, de criação de um "*Hello World*" (Minott, 2020) e a ativação do modo *debug* (Min Xie, 2003). Apesar de não serem *use cases* são tarefas muito comuns no processo de desenvolvimento de software, principalmente em tecnologias novas.

Tabela 10 – Testes a usar como indicadores de velocidade de desenvolvimento por tecnolodia.

ID	Problema	Tipo de teste	UCPs
1	Criação de um "*Hello World*".	Tarefa	-
2	Ativar e usar o modo *debug*.		-
3	Enviar dados de um *input* a um serviço *rest* e mostrar a resposta.	*Use case*	15
4	Abrir a câmara.		8
5	Tirar, mostrar e guardar uma fotografia.		18
6	Abrir um *url* no *in-app browser.*		6
7	Criar uma lista, maior que o ecrã, permitindo que seja possível fazer *scroll* na mesma. De modo a ver todo o seu conteúdo.		8
8	Criar dois ecrãs, navegar do primeiro para o segundo e vice-versa.		13
9	Passar informação entre duas *views*.		13
10	Guardar informação no smartphone. Ler a informação ao reiniciar a app.		11

Para cada um dos problemas identificados foi calculado o valor dos seus UCPs seguindo a Equação 4 apresentada na subsecção 4.2.2, os resultados estão presentes na Tabela 10. Estas serão distribuídas ao maior número de sujeito possível para que, das implementações das soluções aos problemas, seja possível recolher o maior número de dados possíveis.

4.3 Sumário

Neste capítulo foi possível estruturar um conjunto de problemas cujas soluções devem ser implementadas em cada uma das tecnologias identificadas. A cada problema foi atribuído um valor quantitativo de *use case points* que representa a sua complexidade.

Os problemas identificados serão resolvidos por um grupo de sujeitos que registará as horas necessárias para implementar a solução de cada problema. Esses dados serão utilizados para fazer uma avaliação da eficiência de desenvolvimento de aplicações móveis de cada tecnologia.

Em suma deste capítulo resultou o *design* da metodologia de recolha de dados idealizada na Secção 1.5. Com os estudos efetuados neste capítulo é agora possível responder às perguntas levantadas.

Que indicador será usada como KPI?

O indicador usado como KPI será a relação entre os UCPs atribuídos a cada tarefa e o tempo necessário para implementar a tarefa. Como explicado na subsecção 4.2.2, o indicador final pode ser representado por UCPs/h, correspondendo literalmente à velocidade de implementação de soluções aos problemas, que como definido na Secção 1.3 corresponde à eficiência de desenvolvimento.

Que tecnologias vão ser estudadas?

Tendo em conta o referido na secção 4.1, as três tecnologias finais escolhidas para serem estudadas são: React Native, Flutter e Xamarin.

Como é que os dados relativos à métrica escolhida serão recolhidos?

Conforme referido na subsecção 4.2.3, os dados relativos aos UCPs/h serão recolhidos através da criação de uma série de tarefas. Cada tarefa tem um número de UCPs que representa a sua complexidade. As tarefas serão distribuídas a um grupo de sujeitos que terá de implementar a solução das mesmas em diferentes tecnologias. Cada sujeito irá reportar as horas necessárias para a implementação da solução da tarefa.

Com os UCPs de cada tarefa e os dados relativos ao tempo necessário para as realizar, será possível calcular os UCPs/h de cada tecnologia.

5 Implementação da metodologia para recolha de dados

Neste capítulo será descrita a construção da metodologia desenhada no capítulo anterior, que tem como objetivo a recolha de dados, para que seja possível comparar a eficiência de cada tecnologia. Assim como identificadas, e justificadas, todas as diferenças entre o desenho da metodologia e a metodologia final.

Ao longo deste capítulo vão ser dadas respostas às seguintes perguntas:

Quais os problemas identificados durante o processo de construção da metodologia?

Quais as diferenças entre o desenho inicial da metodologia e o resultado final da mesma?

O resultado final, da implementação da metodologia, permite recolher dados de modo a ser possível retirar conclusões sobre as tecnologias em estudo?

5.1 Aplicação da metodologia

Quando a metodologia foi desenhada, como apresentada no capítulo anterior, a mesma tinha em mente a apresentação do projeto a vários sujeitos, aos quais seria apresentado e explicado o objetivo do estudo. Seriam também lançados vários desafios, a esses sujeitos, para que os mesmos pudessem fazer parte do estudo, como sujeitos.

5.1.1 Impacto do COVID-19 na implementação da metodologia

Devido ao estado atual de pandemia foi impossível apresentar este projeto a possíveis sujeitos, o que torna impossível a recolha de dados fazendo uso da metodologia escolhida durante o processo do desenho da metodologia.

Nesta situação, para que a metodologia fosse viável e permitisse recolher dados usáveis no teste das hipóteses, foi necessário fazer adaptações à metodologia previamente desenhada.

5.1.2 Adaptações à metodologia desenhada

Em suma a ideia inicialmente desenhada tinha como prossuposto a obtenção de dados através da resolução de problemas, previamente criados, por diferentes sujeitos. Os sujeitos desenvolveriam soluções para os diferentes problemas usando as diferentes tecnologias em estudo. Durante a resolução dos problemas seriam recolhidos vários dados que permitiriam

comparar e tirar conclusões sobre a eficiência das diversas tecnologias no contexto de desenvolvimento.

Sendo impossível apresentar este estudo e os respetivos problemas a resolver a sujeitos a obtenção de dados realizou-se de forma diferente. Foi feita uma parceria com uma empresa de desenvolvimento de software, DIGITAL SKWARE, que disponibilizou métricas sobre os diferentes projetos de desenvolvimento de aplicações móveis. As métricas em questão dizem respeito ao tempo utilizado, por cada colaborador da empresa, a resolver problemas relacionados com o desenvolvimento de aplicações móveis. Com isto o objetivo é associar casos de testes, apresentados na Tabela 10, a problemas para os quais a DIGITAL SKWARE implementou soluções e partilhou as métricas das mesmas. Uma versão desta metodologia de recolha de dados já tinha sido pensada durante o desenho da solução, mas não tinha sido escolhida como primeira opção.

Com isto, apesar da situação atual de pandemia, é na mesma possível a obtenção de métricas relativas ao tempo necessário para a implementação das soluções nas diferentes tecnologias estudadas.

Em contrapartida a quantidade de dados recolhidos será muito menor que o inicialmente previsto, isto representa um risco para o estudo. A baixa quantidade dados coloca em risco a relevância e, por consequência, a viabilidade do estudo apresentado neste documento.

Para combater este problema foi construído um inquérito com o objetivo de facilitar a extrapolação dos dados recebidos da DIGITAL SKWARE. Do inquérito faz parte uma série de questões relacionadas com a experiência no desenvolvimento de software usando as diferentes tecnologias em estudo. O mesmo será posteriormente distribuído em fóruns e grupos *online* específicos para programadores de aplicações móveis.

Apesar de destas medidas não resultarem os mesmos dados que na solução inicialmente proposta, será na mesma possível, com o auxílio do resultado dos inquéritos, extrapolar a partir dos dados recebidos da DIGITAL SKWARE.

5.2 Construção do inquérito

Com este inquérito é pretendido que seja recolhida a maior quantidade de dados possíveis. Porém, de modo aos mesmos terem alguma relevância, é necessário que os dados sejam recolhidos de sujeitos que já tenham tido contacto com pelo menos uma das tecnologias em estudo.

Reconhecendo os objetivos do inquérito foram delineadas algumas normas que o mesmo tem de respeitar:

- **Nenhuma das perguntas deve obrigar o sujeito a escrever a sua resposta:** Quando um sujeito abre o inquérito não é certo que o complete. Com perguntas de seleção, ordenação ou escolha múltipla é mais provável que o sujeito responda ao inquérito,

uma vez que não terá tanto trabalho como se tivesse de desenvolver respostas por escrito.

- **As perguntas devem ser curtas e claras:** A justificação desta norma é semelhante à anterior. Caso as perguntas sejam grandes ou de difícil compreensão é mais provável que o sujeito desista do inquérito a meio.
- **O inquérito deve apenas ser distribuído a sujeitos que já tenham tido contacto com as tecnologias em estudo:** Esta norma existe para que os dados sejam os mais relevantes possíveis.
- **O número de questões deve ser o mais curto possível:** Esta norma é necessária para que, uma vez que existe o objetivo de recolher informação do maior número possível de sujeitos, é necessário que o questionário seja curto para evitar a desistência a meio do questionário. Isto reduz também o risco de sujeitos selecionarem respostas aleatórias para terminar o questionário mais rápido.

5.2.1 Listagem das perguntas

Tendo em conta as normas e os objetivos do inquérito, foi desenvolvida uma série de questões, presentes na Tabela 11.

Tabela 11 - Questões presentes no questionário.

Número	Questão	Tipo de resposta
1	Tem de desenvolver uma aplicação num período limitado. Ordene as tecnologias por preferência de uso	Ordenação
2	Um amigo seu, sem experiência no desenvolvimento de aplicações móveis, quer rapidamente criar uma aplicação simples para registar a sua lista de compras. Que tecnologias recomenda?	Ordenação
3	Que tecnologia gosta mais de usar, ou está mais entusiasmado por experimentar?	Seleção de opção

Nas perguntas de ordenação, 1 e 2, é possível para o sujeito indicar que não trabalhou com aquela tecnologia, ignorando-a assim na ordenação.

As questões, referidas na Tabela 11, respeitam as normas anteriormente descritas e permitem a recolha de informação sobre pontos relevantes para este estudo:

- Rapidez de desenvolvimento em cada uma das tecnologias, tendo em conta a experiência pessoal de cada indivíduo, pergunta 1.
- Curva de aprendizagem de cada uma das tecnologias, tendo em conta a experiência pessoal de cada indivíduo, pergunta 2.
- Qual a tecnologia que cada sujeito tem mais gosto ou vontade de usar no processo de desenvolvimento de aplicações móveis.

Com estes dados vai ser possível retirar conclusões sobre a experiência que cada sujeito teve com as tecnologias em estudo, quer sobre a velocidade de desenvolvimento quer sobre a curva de aprendizagem. Essas conclusões serão úteis para, por sua vez, ajudar na extrapolação dados recebidos da DIGITAL SKWARE.

5.2.2 Distribuição do inquérito

Como referido anteriormente o alvo deste inquérito são programadores que já tenham tido contacto com as tecnologias em estudo. Para que o inquérito chega a sujeitos que façam parte desse conjunto, o mesmo será partilhado e distribuído em grupo de redes sociais e fóruns destinados às tecnologias relevantes.

Foi feito um levantamento de várias opções para a distribuição do inquérito e foram selecionadas as seguintes plataformas:

- Reddit.com
- Facebook.com

Ambas as plataformas têm o conceito de comunidade, no caso do Reddit como “Subreddit” e no caso do Facebook como “Facebook Groups”. Isto foi uma das principais razões para a escolha destas plataformas, uma vez que permite a seleção das comunidades destinadas às plataformas em estudo para a distribuição dos inquéritos.

É também relevante o facto das comunidades em ambas as plataformas terem regras de partilha de conteúdo não muito restritas o que faz com que seja possível a divulgação dos inquéritos nas mesmas.

Apesar das perguntas na Tabela 11 estarem em português, o inquérito final será distribuído em Inglês, de modo a poder ser distribuído ao maior número de sujeitos.

Para facilitar a distribuição do inquérito, o mesmo será feito usado a ferramenta Google Forms, que para além de ser algo já conhecido pela comunidade facilita a exportação da informação. É também útil no sentido que permite limitar o número de respostas por pessoa que, no caso deste inquérito, é essencial que esse número seja um.

5.3 Resultado final da construção do inquérito

De todas as normas e objetivos delineados para o inquérito resultou o um conjunto de perguntas, inseridas na plataforma Google Forms. O conjunto de perguntas, apresentado nesta secção nas Figuras Figura 6Figura 7Figura 8, corresponde ao conjunto de perguntas identificadas na Tabela 11.

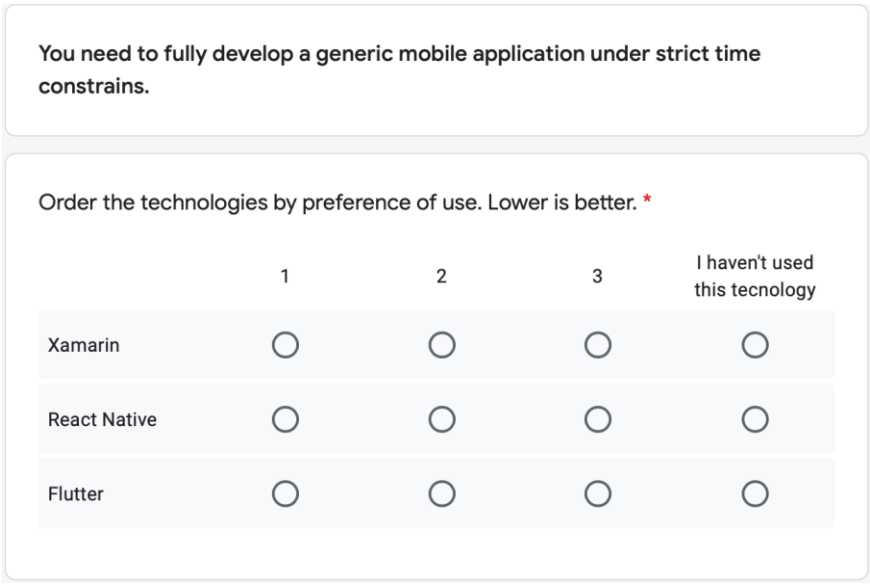

Figura 6 - Primeira pergunta do questionário

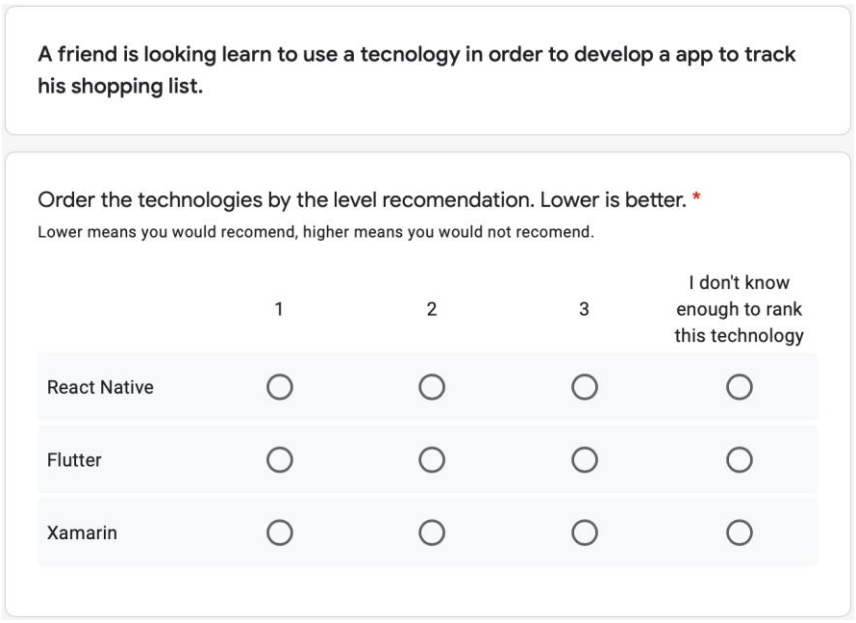

Figura 7 - Segunda pergunta do questionário

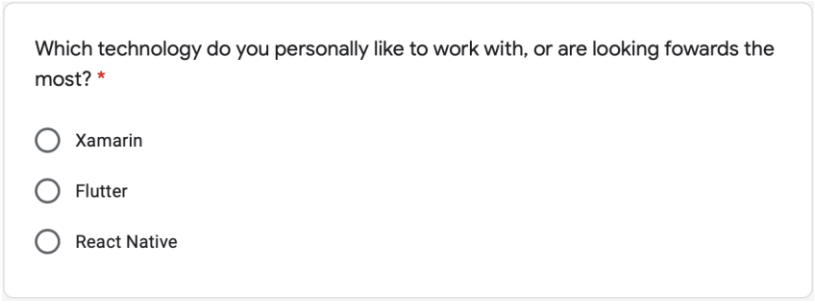

Figura 8 - Terceira pergunta do questionário

O questionário, apresentado nas Figuras Figura 6Figura 7Figura 8, foi distribuído a diferentes grupos de programadores que usam as tecnologias em estudo. A distribuição foi feita a partir da partilha do link do questionário, via Google Forms, (Cerdeira, 2020).

5.4 Sumário

Com os estudos efetuados neste capítulo é agora possível responder às perguntas levantadas.

Quais os problemas identificados durante o processo de construção da solução?

Como referido no capítulo 4, a solução envolvia a distribuição de um conjunto de problemas a sujeitos que por sua vez teriam de implementar a solução a esses problemas em diferentes tecnologias. Desse desenvolvimento seriam recolhidos dados que permitiriam o cálculo da métrica de *performance* desejada, UCPs/h.

Com o estado atual de pandemia, explicado na subsecção 5.1.1, não foi possível o contacto com programadores para a apresentação deste estudo. Impossibilitando a recolha de dados seguindo a metodologia apresentada e escolhida anteriormente no desenho da solução.

Quais as diferenças entre o desenho inicial da solução e o resultado final da mesma?

Como identificado na subsecção 5.1.2, a principal diferença entre o desenho da solução e a solução implementada é a metodologia de recolha de dados. Na solução implementada os dados relativos aos UCPs/h são recolhidos diretamente das métricas entregues por uma empresa.

Essa mudança fez com que o número de sujeitos que efetivamente participam neste estudo diminuíam, fazendo com isso com que os dados recolhidos tenham menos relevância. Para combater essa diferença foi desenvolvido um inquérito que será distribuído com o intuito de recolher informação sobre a experiência de cada sujeito no desenvolvimento de aplicações móveis nas diferentes tecnologias.

O resultado final, da implementação da solução, permite recolher dados de modo a ser possível retirar conclusões sobre as tecnologias em estudo?

Sim. Apesar de o número de sujeitos que contribuirá com as suas métricas de *performance* na velocidade de desenvolvimento de software ser menor, os dados recebidos do inquérito construído nas secções 5.2 e 5.3 ajudarão a dar relevância estatística aos dados recebidos da empresa. Permitindo assim retirar conclusões sobre as tecnologias em estudo.

6 Avaliação dos dados recolhidos

Neste capítulo serão identificados os dados necessários para que sejam possível tirar conclusões relativamente às hipóteses em testes. Bem com a identificação dessas mesmas hipóteses.

Ao longo deste capítulo vão ser dadas respostas às seguintes perguntas:

Com os dados recolhidos, foi possível obter o KPI para cada tecnologia?

6.1 Apresentação dos dados recolhidos

Nesta secção serão apresentados todos os dados recolhidos ao longo do estudo, bem como a comparação dos dados recebidos da DIGITAL SKWARE e os dados recolhidos através dos inquéritos distribuídos.

6.1.1 Dados recebidos da DIGITAL SKWARE

No decorrer da duração do estudo foram recolhidos dados relativos à duração da implementação de soluções, relacionados com desenvolvimento de aplicações móveis, nas diferentes tecnologias.

Os dados foram recolhidos ao longo dos últimos dois anos, em projetos desenvolvidos pela DIGITAL SKWARE. Três programadores desenvolveram soluções para React Native e, dois desses mesmos programadores, também realizaram tarefas em Flutter. Os mesmos registaram as horas, com uma granularidade de 0.5 horas, utilizadas na implementação de inúmeras tarefas. O Xamarin não faz parte das tecnologias usadas pela DIGITAL SKWARE, por isso não foram recebidos quaisquer dados relativos a essa tecnologia.

Dos dados recebidos da DIGITAL SKWARE nem todos foram utilizados neste estudo, foi feito sim um mapeamento das tarefas, e respetivos dados, apresentadas pela empresa e as tarefas apresentadas na Tabela 10 da Secção 4.2 deste estudo. Esse mapeamento foi feito comparando as descrições das tarefas entregues pela DIGITAL SKWARE e os *use cases* presentes na Tabela 10.

De seguida foi calculada a média do tempo que os programadores usaram para cada tarefa. Esses dados encontram-se na Tabela 12. É importante realçar que, tendo em conta os dados recebidos da DIGITAL SKWARE, três programadores realizaram trabalho para React Native e dois para Flutter. Para os cálculos da média de horas necessárias para completar as tarefas presentes na Tabela 12 foi sempre possível usar dados de no mínimo, dois programadores.

Tabela 12 – Horas necessárias para cada tarefa, por tecnologia

	Média de horas necessárias para completar a tarefa			UCPs
ID	React Native	Flutter	Xamarin	
1	1	1	-	-
2	1	1	-	-
3	2	2	-	15
4	2	1	-	8
5	5	3	-	18
6	1.5	0.5	-	6
7	1	1	-	8
8	0.5	0.5	-	13
9	0.75	1	-	13
10	1	1	-	11

Como é possível verificar, na Tabela 12, não foi possível preencher os dados relativos ao Xamarin. Sendo assim não vai ser possível retirar conclusões, relativas ao Xamarin, diretamente dos dados recebidos da DIGITAL SKWARE.

Os dados recebidos da DIGITAL SKWARE correspondem à média do tempo utilizado para desenvolver uma solução para cada tarefa. É relevante referenciar que, segundo a DIGITAL SKWARE, aquando do desenvolvimento das soluções para as tarefas das quais resultaram estes dados os programadores que a concluíram não tinham qualquer experiência na tecnologia. Esta informação é positiva uma vez que garante que todos os dados recolhidos foram oriundos de programadores que possuíam o mesmo grau de experiência com a tecnologia em questão.

Dos dados presentes da Tabela 12, foram calculados os UCPs por hora de cada uma das tarefas e tecnologia, os resultados são visíveis na Tabela 13.

Tabela 13 - UCPs por hora, para cada uma das tencologias.

	UCPs/hora		
ID	React Native	Flutter	Xamarin
3	7.5	7.5	-
4	4	8	-
5	3.6	6	-
6	4	12	-
7	8	8	-
8	26	26	-
9	17.3(3)	13	-
10	11	11	-

Como é possível observar na Tabela 13 e na Figura 9, a tarefa com o id 9 possui um valor de UCPs/hora bastante distante dos outros valores. Como tal a mesma vai ser excluída das próxima visualizações, cálculos e observações retiradas a partir dos dados.

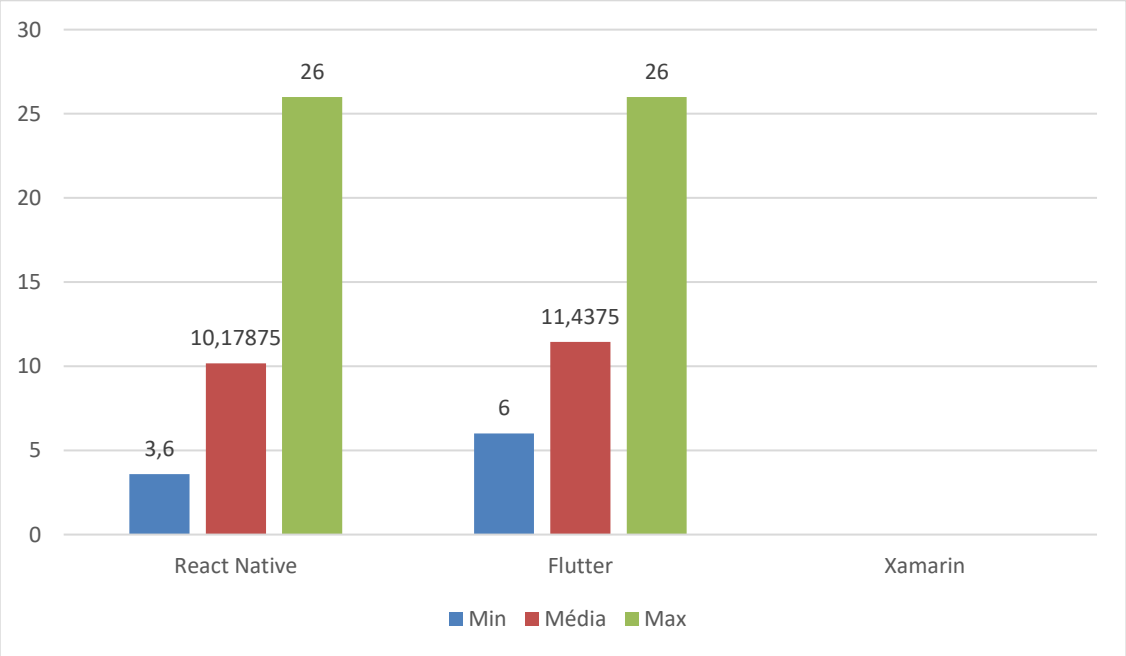

Figura 9 - Comparação entre os valores de UCPs por hora de cada tecnologia

Na Figura 10 é possível verificar que, tendo em conta os dados recolhidos, o Flutter possui uma vantagem na velocidade de desenvolvimento. Tendo um valor médio de UCPs por hora superior, um acréscimo de cerca de 18.4% face ao React Native. Apesar de os dados não terem sido recebidos em grande quantidade é possível fazer uma extrapolação com o auxílio dos resultados dos inquéritos.

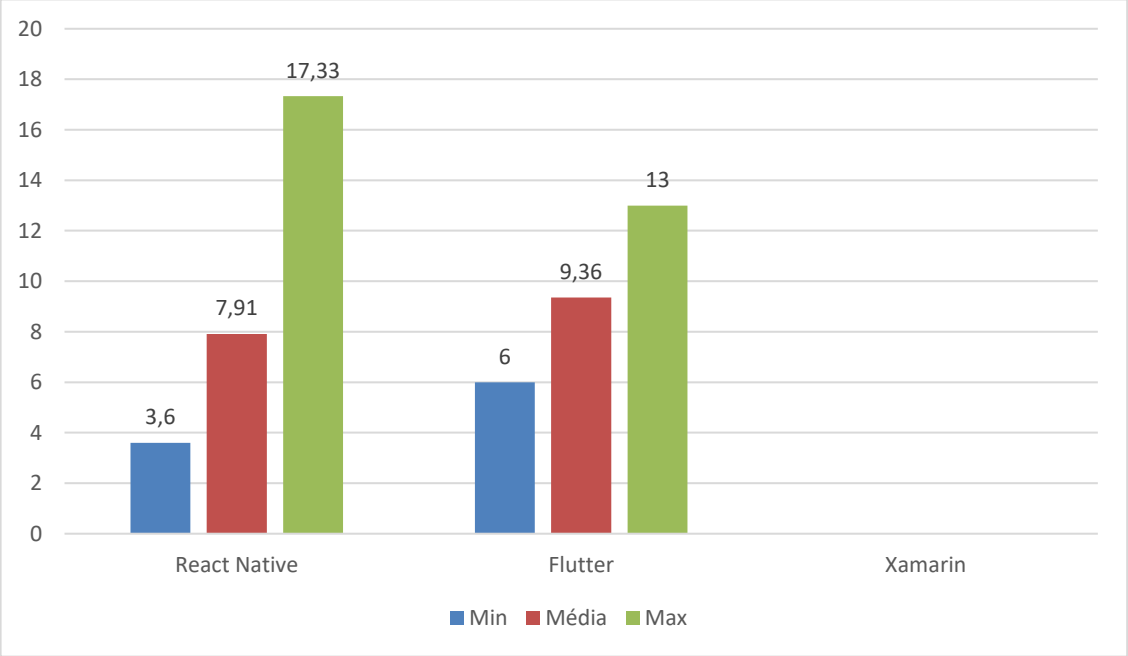

Figura 10 - Comparação entre os valores de UCPs por hora de cada tecnologia (atualizado)

6.1.2 Dados recolhidos do inquérito realizado

Durante o período de recolha de dados foram submetidas um total de 124 respostas únicas ao inquérito. No momento da partilha do inquérito foi feita uma publicação nas plataformas Reddit e Facebook onde o objetivo do estudo era apresentado. A publicação foi feita em grupos de comunidades de programadores das várias tecnologias em estudo. Todas as respostas foram obtidas de forma voluntária e gratuita. Os dados foram diretamente exportados da plataforma Google Forms, estando os mesmos apresentados nas Figuras Figura 11Figura 12Figura 13.

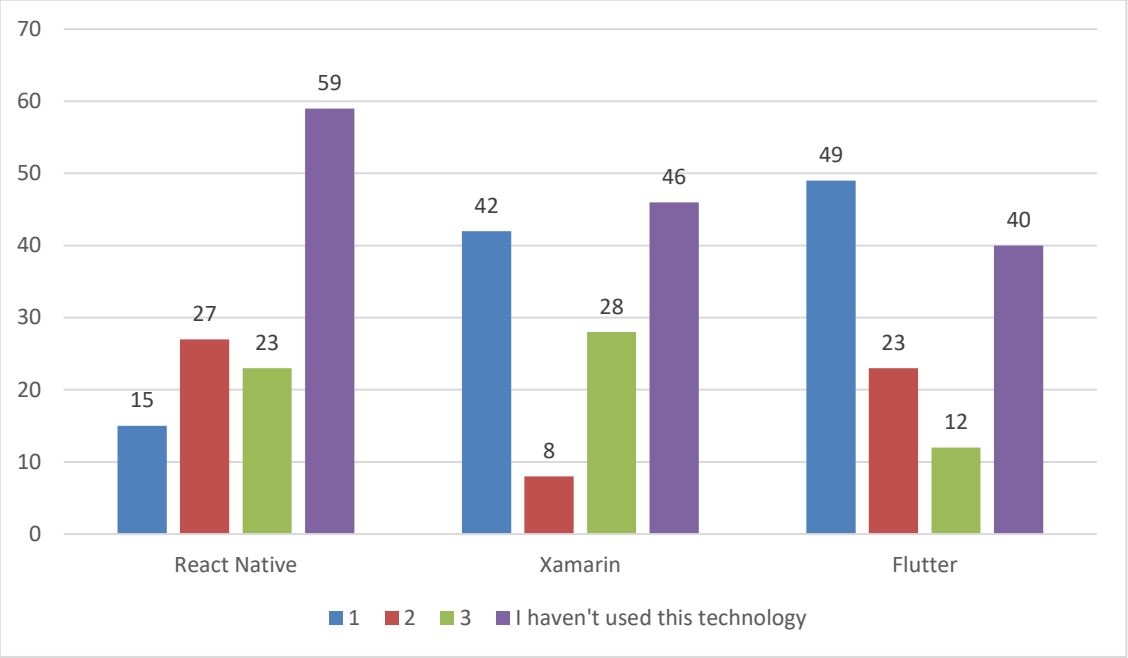

Figura 11 - Respostas à pergunta número 1 do questionário - "*Order the technologies by preference of use. Lower is better.*"

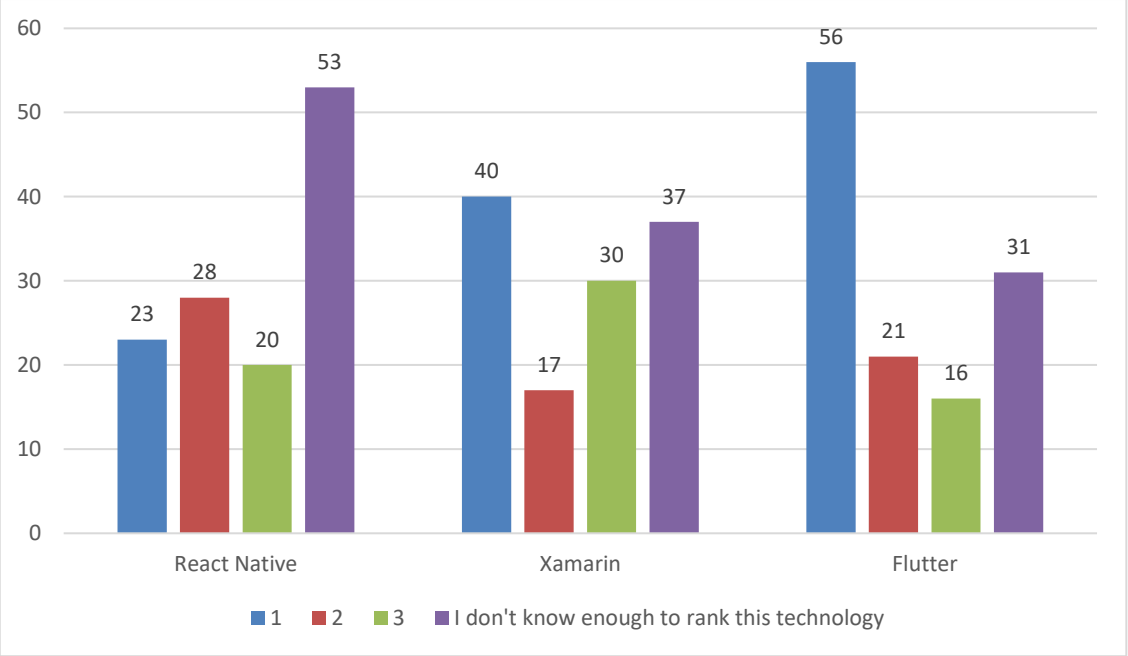

Figura 12 - Respostas à pergunta número 2 do questionário - "*Order the technologies by the level recomendation. Lower is better*."

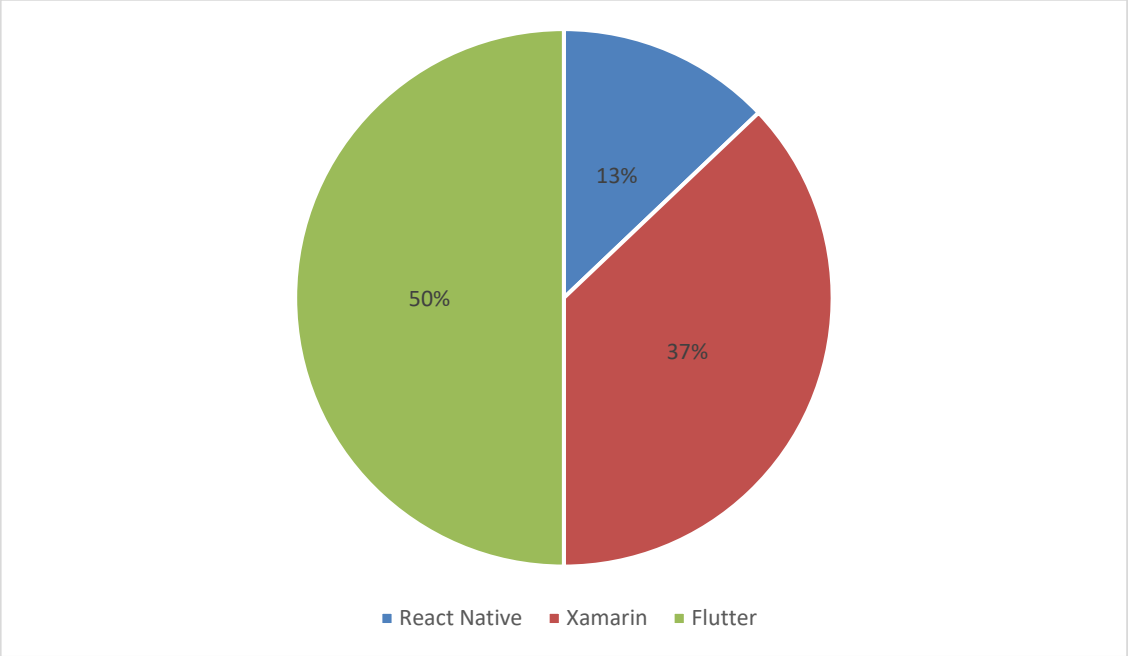

Figura 13 - Respostas à pergunta número 2 do questionário - "*Which technology do you personally like to work with, or are looking forward the most?*"

Nos dados recolhidos da DIGITAL SKWARE foi possível observar que os programadores necessitaram de mais horas para concluir as mesmas tarefas no React Native do que no Flutter. Isto é algo que vai de encontro aos dados recolhidos no inquérito. Na pergunta relacionada com a escolha de uma tecnologia para realizar uma tarefa num período limitado é evidente a preferência por Flutter face ao React Native. Estas afirmações podem ser validadas na tabela seguinte.

Tabela 14 – Comparação entre UCPs por hora e respostas ao inquérito

Tecnologia	Média UCPs/h	Média do valor de resposta à pergunta 1
React Native	7.91	2,12
Xamarin	N/A	1.82
Flutter	9.36	1.56

Para calcular a média do valor da resposta 1, para cada tecnologia foi usada a fórmula presente na Equação 5.

$$\text{MédiaR1} = (\text{NumOpt1} * 1 + \text{NumOpt2} * 2 + \text{NumOpt3} * 3) / (\text{NumResp})$$

Equação 5

Onde NumOpt1, NumOpt2 e NumOpt3 são o número de respostas com a opção “1”, ”2” e “3” selecionadas, respetivamente. NumResp é igual ao total de respostas onde uma das três opções anteriormente mencionadas foi selecionada.

$$\text{MédiaR1\_ReactNative} = (15*1+27*2+23*3)/65 = 2.12$$

Para este indicativo, MédiaR1, um valor mais perto de 1 seria indício de uma tecnologia cujos sujeitos que responderam ao inquérito reconhecem como mais rápida no desenvolvimento. Um valor mais perto de 3 seria indicativo de uma tecnologia cujos sujeitos assumiam como mais lenta para desenvolvimento.

Esta constatação permite, de certa forma, dar relevância aos dados de UCPs por hora recolhidos da empresa DIGITAL SKWARE, uma vez que a mesma relação entre os dados da mesma é visível também nos dados oriundos do inquérito. Este facto vai fazer com que seja viável, até um certo ponto, fazer uma extrapolação a partir de uma quantidade, inicialmente pequena de dados. A relação entre os diferentes conjuntos de dados é estudada na Secção 6.2.

6.2 Relação dos UCPs por hora recolhidos e os dados do inquérito

Nesta secção é pretendido chegar, matematicamente, a uma relação entre os valores de UCPs/hora recolhidos, e a médico do valor de resposta à pergunta 1 do inquérito. Essa relação pode apenas ser calculada usando os dados das tecnologias em estudo, React Native e Flutter, uma vez que não existem dados de UCPs/hora para o Xamarin. Porém a relação identificada entre os dois conjuntos de dados é também útil para ser possível extrapolar relativamente a valores de UCPs/hora esperados no Xamarin. Possibilitando uma comparação entre as três tecnologias em estudo.

Para facilitar a comparação entre os dois conjuntos de dados, foi feita uma transformação relativa à apresentação da média do valor de resposta à pergunta 1. Será agora representada por um valor percentual, em que 0% corresponderia a uma média de valor igual a 3 e 100% uma média de valor igual a 1. O cálculo desse novo valor é feito seguindo a seguindo a fórmula presente na Equação 6.

$$\text{PercentagemR1} = (100 – 100*(\text{MédiaR1} – 1)/2) *100$$

Equação 6

Tabela 15 – Relação PercentagemR1 e UCPs/hora

Tecnologia	Média UCPs/h	PercentagemR1	UCPs/hora / PercentagemR1
React Native	7.91	44%	0.18
Xamarin	N/A	59%	N/A
Flutter	9.36	72%	0.13

Por observação da informação presente na Tabela 15, é possível afirmar que para cada ponto percentual de PercentagemR1 para o React Native correspondem um total de 0.18 UCPs/hora. Já para o Flutter, para cada ponto percentual de PercentagemR1, correspondem um total de 0.13 UCPs/hora, significando que existe uma relação em média de 0.155 UCPs/hora por cada valor percentual de PercentagemR1 atribuído a cada tecnologia.

6.3 Dados finais e estimativa dos UCPs por hora do Xamarin

Apesar de não ter sido possível obter informações relativas a valores de UCPs/hora na tecnologia Xamarin, foi possível obter dados da mesma através das respostas dos inquéritos.

Após ter sido calculada a média do rácio entre os valores da média de UCPs/hora e PercentagemR1, para o React Native e Flutter, é possível calcular uma estimativa para o valor de UCPs/hora do Xamarin. Fazendo uso do valor PercetagemR1 da tecnologia Xamarin.

Tabela 16 - Estiamtiva UCPs por hora para a tecnologia Xamarin.

Tecnologia	Média UCPs/h	PercentagemR1	UCPs/hora / PercentagemR1
React Native	7.91	44%	0.18
Xamarin	**9.15**	**59%**	**0.155**
Flutter	9.36	72%	0.13

6.4 Análise dos dados e teste das hipóteses

Para testar as hipóteses serão feitas análises aos dados recolhidos em cada tecnologia e tiradas conclusões relativas ao tempo necessário para a conclusão de um UCP.

Como verificado na Tabela 16 foi possível obter os valores do KPI para cada uma das tecnologias. Com esses dados foi possível concluir que a hipótese em teste, referida na Secção 6.1, é de facto

verdadeira. Para as três tecnologias estudadas, tendo em conta os dados recolhidos, existem diferenças substanciais na eficiência de desenvolvimento de aplicações móveis.

Como é possível observar nos dados presentes na Tabela 16, o Flutter e o Xamarin obtiveram valores de UCPs/h bastante mais elevados com o React Native. Comparativamente ao React Native, o Xamarin e o Flutter obtiveram valores de UCPs/h 16.

% e 18%, respetivamente, mais elevados.

6.5 Sumário

Com os estudos efetuados neste capítulo é agora possível responder às perguntas levantadas.

Com os dados recolhidos, foi possível obter valores do KPI, neste caso UCPs/h, para cada tecnologia?

Sim. Apesar da DIGITAL SKWARE não ter disponibilizado dados relativos à tecnologia Xamarin, foi possível relacionar os restantes dados recebidos da DIGITAL SKWARE, dois quais resultaram os valores de UCPs/h para React Native e Flutter, com os dados obtidos do inquérito realizado. Com isso foi possível realizar uma estimativa dos UCPs/hora para a tecnologia Xamarin.

7 Conclusões

Neste capítulo será apresentado um resumo do estudo descrito neste documento, bem como os objetivos realizados. Serão também apresentadas algumas das limitações da solução, assim como algum trabalho que pode ser realizado futuramente na metodologia apresentada.

7.1 Resumo

Como referido na Secção 1.2 existem cada vez mais tecnologias que permitem um desenvolvimento de aplicações móveis, para diferentes sistemas operativos, a partir de uma base de código. Estas tecnologias têm como objetivo permitir um desenvolvimento mais eficiente, porém não existem dados que comparem as tecnologias mais relevantes relativamente à sua eficiência de desenvolvimento.

O objetivo deste estudo era o desenvolvimento de uma metodologia de recolha de dados que permitissem comparar as diferentes tecnologias usadas no desenvolvimento de aplicações móveis, relativamente à sua eficiência de desenvolvimento.

7.2 Objetivos realizados

Com a implementação da metodologia de recolha de dados era pretendido dar resposta à questão levantada na Secção 1.3. E após o desenho e implementação da metodologia foi possível responder à pergunta.

Qual a tecnologia de desenvolvimento de aplicações móveis que permite um desenvolvimento de aplicações para Android e iOS mais eficiente?

Dos dados obtidos relativos aos UCPs/h de cada tecnologia é evidente o destaque do Flutter como a tecnologia mais eficiente. Porém, devido ao número baixo de dados que foi possível recolher, sem investigação adicional não é sensato assumir que o Flutter é de facto a tecnologia mais eficiente.

Foi também apresentado como objetivo deste estudo, na Secção 1.3, a criação de uma metodologia que permitisse a recolha de dados com os quais fosse possível comparar a eficiência de desenvolvimento de aplicações móveis de diferentes tecnologias. Apesar de não ter sido totalmente posta em prática, devido ao atual estado de pandemia, a metodologia foi desenhada e, até um certo ponto, implementada. A metodologia permitiu, de facto, a recolha de dados relacionados com a eficiência do desenvolvimento, este objetivo foi cumprido.

Em circunstâncias normais, do uso da metodologia apresentada no Capítulo 4 resultará um conjunto de dados suficientemente vasto e relevante para que sejam retiradas conclusões relativamente à eficiência de desenvolvimento das tecnologias que se pretenderem estudar.

Em suma, apesar de não ter sido possível obter dados suficientes para ser possível afirmar com segurança qual a tecnologia mais eficiente para o desenvolvimento de aplicações móveis a metodologia para tal foi desenhada e implementada. Dadas as circunstâncias de pandemia que existiram durante o desenvolvimento deste estudo não foi possível tirar proveito total da metodologia desenvolvida, resultando num número de dados mais baixo que o esperado.

7.3 Limitações e trabalho futuro

Apesar da metodologia desenhada ter funcionado ficou provado com este estudo que ainda existem limitações. A metodologia não tinha previso a impossibilidade do uso de sujeitos para a implementação das soluções aos problemas proposto. Algo que ficou bastante evidente quando foi necessário fazer uso de sujeitos e o estado atual de pandemia o impediu.

Como trabalho futuro seria interessante a realização de algumas melhorias à metodologia proposta, bem como a realização de uma nova recolha de dados. Uma vez que em situações normais, e não de confinamento, a mesma teria possibilitado uma quantidade de dados superior. Algo que tinham permitido retirar uma conclusão mais assertiva relativamente à tecnologia mais eficiente no desenvolvimento de aplicações móveis.

Referências

AltexSoft. (2019). *The Good and The Bad of Xamarin Mobile Development*. Obtido de https://www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/

AltextSoft. (2019). *The Pros and Cons of Flutter App Development*. Obtido de Hackernoon: https://hackernoon.com/the-good-and-the-bad-of-flutter-app-development-45f27b1caa36

Banerjee, G. (2004). Use Case Estimation. Obtido de https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.110.453&rep=rep1&type=pdf

Bershadskiy, S. (2015). *Modus*. Obtido de https://moduscreate.com/blog/leverage-existing-ios-views-react-native-app/

Blair, I. (Fevereiro de 2020). *Mobile App Download and Usage Statistics*. Obtido de https://buildfire.com/app-statistics/

Cerdeira, H. (2020). *Mobile Development Technologies*. Obtido de https://forms.gle/zaEwNsQdGkZmS87n7

Costello, S. (10 de Janeiro de 2020). *How Many Apps Are in the App Store?* Obtido de lifewire: https://www.lifewire.com/how-many-apps-in-app-store-2000252

Developer Survey Results. (2018). Obtido de Stackoverflow: https://insights.stackoverflow.com/survey/2018/#overview

Facebook. (2020). Obtido de https://github.com/facebook/react-native

Feoktistov, I. (2020). Obtido de https://relevant.software/blog/react-native-vs-flutter-which-to-choose-for-cross-platform-development/#Apps_Built_with_React_Native

Flutter architectural overview. (2020). Obtido de Flutter: https://flutter.dev/docs/resources/architectural-overview

Github. (Fevereiro de 2020). *Mobile on Github*. Obtido de Mobile: https://github.com/topics/mobile?o=desc&s=stars

Google. (2020). *FAQ about Google Trends data*. Obtido de Google Forms: https://support.google.com/trends/answer/4365533?hl=en

GoogleTrends. (2020). Obtido de https://trends.google.com/trends

Green, A. (2019). Obtido de https://get-a-wingman.com/top-7-programming-jobs-that-will-be-most-in-demand-in-2020/

Jones, K. (25 de Janeiro de 2020). *The World's Most Downloaded Apps*. Obtido de https://www.visualcapitalist.com/ranked-most-downloaded-apps/

Keerti. (Dezembro de 2016). *React Native at WalmartLabs*. Obtido de https://medium.com/walmartlabs/react-native-at-walmartlabs-cdd140589560#.ueonqqloc

Linda M. Laird, M. C. (2006). *Software Measurement and Estimation: A Practical Approach.* Wiley-IEEE Computer Society Pr; 1ª Edição (5 junho 2006).

Martin, S. (Abril de 2020). Obtido de https://medium.com/datadriveninvestor/7-popular-cross-platform-app-development-tools-that-will-rule-in-2020-349c80fb51

Mawhinney, J. (28 de Maio de 2015). *How to Choose the Right KPIs for Your Business*. Obtido de https://blog.hubspot.com/marketing/choosing-kpis

Microsoft. (2020). Obtido de What is Xamarin?: https://docs.microsoft.com/en-us/xamarin/get-started/what-is-xamarin

Microsoft. (2020). *Xamarin Tutorial - Hello World in 10 minutes*. Obtido de https://dotnet.microsoft.com/learn/xamarin/hello-world-tutorial/modify

Min Xie, B. Y. (Maio de 2003). A study of the effect of imperfect debugging on software development cost. Obtido de A study of the effect of imperfect debugging on software development cost: https://blog.bitlabstudio.com/the-importance-of-debugging-886df73427ea

Minott, Z. (2020). Obtido de Why "Hello World" Is The Most Important Program You'll Write: https://medium.com/dev-genius/why-hello-world-is-the-most-important-program-youll-write-8d5b1a59cc30

Occhino, T. (Março de 2015). *React Native: Bringing modern web techniques to mobile*. Obtido de https://engineering.fb.com/android/react-native-bringing-modern-web-techniques-to-mobile/

Parmenter, D. (2015). *Key Performance Indicators: Developing, Implementing, and Using Winning KPIs.* WILEY. Obtido de https://kpi.org/KPI-Basics

Saaty, T. L. (2008). Relative Measurement and Its Generalization in Decision Making Why Pairwise Comparisons are Central in Mathematics for the Measurement of Intangible Factors The Analytic Hierarchy/Network Process. *RACSAM - Revista de la Real Academia de Ciencias Exactas, Físicas y Naturales. Serie A. Matemáticas*.

Szczepański, M. (2018). European app economy State of play, challenges and EU policy. Obtido de European app economy: http://www.europarl.europa.eu/RegData/etudes/BRIE/2018/621894/EPRS_BRI(2018)621894_EN.pdf

Thompson, E. (2017). *Discover the Countries Leading in App Usage*. Obtido de https://www.appannie.com/en/insights/market-data/global-consumer-app-usage-data/

Turner, A. (Fev de 2020). *HOW MANY SMARTPHONES ARE IN THE WORLD?* Obtido de bankmycell: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world

TWIN, A. (19 de Setembro de 2019). *Key Performance Indicators (KPIs)*. Obtido de https://www.investopedia.com/terms/k/kpi.asp

Vuk, A. (2017). *Difference between Xamarin.Forms and Xamarin Traditional*. Obtido de https://almirvuk.blogspot.com/2017/07/difference-between-xamarinforms-and.html