EasyChair Preprint

№ 4953

Presentation and Analysis of the Ecosystem for Mobile Development in Multiplatform

Ricardo Lucas Jardim

February 3, 2021

Apresentação e análise do ecossistema para desenvolvimento mobile em multiplataforma

1st Ricardo Lucas
Faculdade de Ciências Exatas e da Engenharia
Universidade da Madeira
Funchal, Portugal
ricardo.work.jardim@hotmail.com

***Resumo*—Esta tese foca-se em problemas identificados no mercado das tecnologias Cross-Platform, e na enorme variedade de soluções disponíveis tornando-se difícil escolher a melhor solução a utilizar.**

Diversos investigadores tentaram criar soluções para alguns destes problemas, criando comparações entre tecnologias de desenvolvimento mobile, identificando as suas vulnerabilidades e problemas que possam limitar alguns aspectos no desenvolvimento de aplicações. Do mesmo modo, dedicaram-se em criar métodos para avaliar uma tecnologia de desenvolvimento mobile, desde questionários ao publico alvo ao desenvolvimento de provas de conceito para realizarem testes de desempenho e benchmarks.

Nesta tese tentamos juntar contribuições de diversas áreas de estudo na avaliação de ferramentas cross-platform. Nomeadamente, desenvolvemos, uma revisão de literatura extensiva de forma a identificar as principais diferenças entre as tecnologias mobile nativas e Cross-Platform, uma análise individual das várias tecnologias Cross-Platform, uma análise comparativa entre várias tecnologias Cross-Platform em conjunto com uma componente quantitativa abordando os developers. Esperamos que com estes objetivos, possamos solucionar os problemas identificados, facilitando a escolha de uma tecnologia dependente da situação proposta, bem como assinalando quais as suas limitações e problemas que possam ser encontrados.

***Index Terms*—Mobile, Cross-Platform, Benchmark, Smartphone, IDE, API, Delevopers**

I. Introduction

Pelo motivo da rápida evolução da tecnologia e dos dispositivos móveis, o mercado de smartphones cresceu de forma exponencial. Pequenas e grandes empresas começaram a desenvolver os seus smartphones e sistema operativos (SO), como por exemplo, a Nokia e BlackBerry que foram das primeiras empresas a produzir um SO capaz de suportar aplicações realizadas por terceiros, utilizando os seus próprios equipamentos. Com esta evolução, o mercado tornou-se assim, um mercado mais acessível e abrangente, tanto a nível monetário como a nível computacional. Durante esta evolução, três empresas emergiram como líderes do mercado com os seus SO's, Google com o Android, Apple com o iOS e Microsoft com o WindowsPhone [1]. Neste momento, o mercado é liderado pela Google com 86,7% do mercado em 2019, [2]. Pois este sistema operativo não está bloqueado a um fabricante de hardware e poderá ser licenciado por qualquer fabricante que deseje oferecer um smartphone com o sistema Android. Logo de seguida, está a Apple e Windows com a menor percentagem do mercado, cada uma com o seu mercado privado de aplicações [3].

As diferenças entre estes SO's, significam que o desenvolvimento de aplicações nativas é também distinto. Utilizando-se o IDE Android Studio em conjunto com a linguagem JAVA e/ou Kotlin para construir aplicações para o sistema Android, o IDE Xcode em conjunto com Objetive-C e/ou Swift para aplicação iOS e por fim, o IDE Visual Studio, juntamente com a linguagem C# e/ou C++ para produzir aplicações para o WindowsPhone [2]. Através destas tecnologias, inúmeras aplicações foram construídas, contudo esta fragmentação revelou-se também problemática. Considerando que os conteúdos das plataformas são diferentes e comportam-se de forma diferente, a mesma aplicação terá que ser construída em várias linguagens e tecnologias para ser suportada em diferentes plataformas. Este facto poderá tornar o processo de construção de aplicações demoroso e dispendioso. De modo a solucionar este problema empresas começaram a construir tecnologias Cross-Platform [4]. O aparecimento de tecnologias Cross-Platform, veio a permitir reduzir o custo e esforço de desenvolvimento, maior rapidez na implementação e reutilização de código e redução de custos monetários, utilizando apenas uma linguagem [5].

A. Descrição do Problema

Nos dias de hoje, existe uma enorme variedade de soluções Cross-Platform para cada tipo de aplicação e feature a desenvolver, tornando difícil a escolha da melhor tecnologia a utilizar. Além disso, o suporte destas tecnologias é também fragmentado, visto que algumas não têm a versatilidade de apoiar uma vasta gama de SO's, limitando se apenas a algumas versões mais antigas destes, gerando assim, grandes problemas em termos de manutenção, suporte e segurança [6].

Visto que o mercado das tecnologias Cross-Platform cresceu exponencialmente, estas começaram a oferecer diferentes abordagens para o mesmo objectivo Cross-Platform, desde PWA's (Progressive Web App) a tecnologias que compilam para código nativo [2]. Tornando a escolha de uma framework para o desenvolvimento de uma aplicação muito importante, em função do facto de, todas estas possuírem diferentes comportamentos, estruturas, linguagens e o mais importante, custo e esforço de implementação em comparação com o

desempenho desejado [6]. Todas estas tecnologias contêm também limitações, inerentes à sua arquitectura, ambiente de desenvolvimento ou mesmo limitações relativas ao seu âmbito.

Este fracionamento poderá gerar grandes dificuldades para empresas e *developers* na escolha da melhor tecnologia para um problema inicial. A escolha da tecnologia certa é de grande importância. Se uma empresa escolhe uma tecnologia aleatória, posteriormente poderá encontrar graves problemas de desempenho, falta de suporte, entre outros problemas inesperados, sendo obrigada a recomeçar todo o processo de desenvolvimento em outra tecnologia [7], ou oferecer um produto que não vai de encontro aos requisitos iniciais.

Devido a estes problemas, e à falta de guias analíticos, não só a nível de implementação mas também de compatibilidade, torna-se difícil escolher qual é a melhor solução para um problema proposto. Muitas destas tecnologias têm problemas de escalabilidade e tornando apenas adequada para projectos pequenos, restringindo a evolução do sistema, [8]. Assim e pelo facto de ser necessário investir tempo para analisar e compreender as limitações de uma tecnologia Cross-Platform, muitas empresas continuam a ter equipas de desenvolvimento separadas pelos sistemas operativos para a construção de aplicações nativas [9].

B. Objetivos

Durante o período da tese proposta, temos como objetivos desenvolver uma revisão bibliográfica a partir do trabalho referido na motivação e descrição I-A, em conjunto com uma análise comparativa completa e válida entre as as várias tecnologias de desenvolvimento Cross-Platform, seleccionando as tecnologias através de um conjunto de critérios de inclusão e exclusão, de forma a eliminar tecnologias idênticas.

O foco final desta dissertação consiste na facilitação da escolha de uma tecnologia de desenvolvimento Cross-Platform, indicando qual é a melhor escolha para a situação proposta, bem como assinalando quais as suas limitações e problemas que possam ser encontrados. Através desta análise, será possível recolher dados ideais para facilitar uma futura construção de um sistema de recomendação de tecnologias para desenvolvimento Cross-Platform.

II. Revisão de Literatura

.

Esta secção será utilizada para apresentar uma contextualização das tecnologias de desenvolvimento mobile, sendo apresentado as diversas abordagens de desenvolvimento mobile, os diversos tipos de tecnologias e ferramentas, e as diferentes implementações de desenvolvimento. Apresentaremos também como surgiu a motivação para as de métricas, parâmetros e métodos utilizados na avaliação das tecnologias de desenvolvimento mobile Cross-Platform.

A. Trabalho Relacionado

O trabalho relacionado representará o estado da arte, identificando as diversas abordagens de desenvolvimento, em conjunto com as métricas e métodos para a avaliação das tecnologias de desenvolvimento mobile utilizadas. Dito isto, o trabalho relacionado foi dividido em quatro sub secções, as diversas abordagens de desenvolvimento mobile, o desenvolvimento nativo, desenvolvimento Cross-Platform, e por fim, métodos e métricas de avaliação.

1) Diversas Abordagens de Desenvolvimento Mobile: A expansão do desenvolvimento de aplicações mobile evoluiu de forma a resolver as necessidades dos *developers*, resolvendo problemas como separação de equipas por SOs, aprendizagem de várias linguagens, entre outros. Levando assim à criação de diversas abordagens de desenvolvimento de aplicações mobile, como podemos ver na figura 1 [10]. Independentemente do tipo de desenvolvimento, é sempre necessário ter em atenção aos desafios e às boas praticas de desenvolvimento mobile, proporcionando uma aplicação final agradável ao utilizador, e uma facilidade na manutenção e adição de funcionalidades [11].

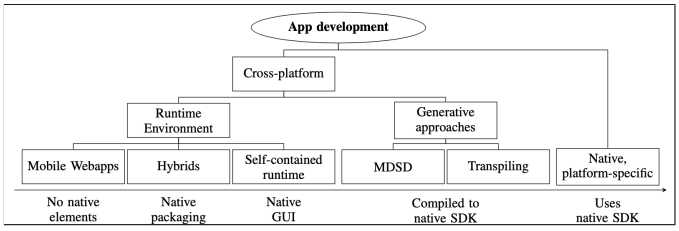

Figura 1. Esquema das possíveis abordagens de desenvolvimento mobile [12]

O desenvolvimento de aplicações mobile pode ser separado entre duas abordagens, as aplicações nativas que são desenvolvidas individualmente para cada SO, utilizando o SDK (Kit de desenvolvimento de software) fornecido, e o desenvolvimento Cross-Platform que utiliza uma linguagem base que pode ser executada em diferentes SOs [13].

Sendo que, o principal objectivo das tecnologias Cross-Platform's é possibilitar o desenvolvimento de aplicações a partir do mesmo código, torna-se necessário ter em conta alguns requisitos, de forma a proporcionar uma tecnologia abrangente e com um bom desempenho [7] [14].

Estes são os requisitos necessários de uma tecnologia Cross-Platform [15] [7] [14]:

- **Suporte a várias plataformas:** Uma tecnologia deve suportar múltiplas plataformas, essencialmente a mais utilizadas no mercado (iOS e Android)
- **Interface rica:** O sucesso de uma aplicação depende em grande parte da experiência do utilizador, logo é necessário ter incorporado uma interface rica e com apoio a animações, multimédia e gráficos 2D e 3D.
- **Segurança:** É essencial as aplicações serem as mais seguras possíveis.
- **Manutenção:** É crucial possibilitar a manutenção da aplicação, sem comprometer com funcionalidades anteriormente desenvolvidas
- **Consumo de recursos:** As aplicações desenvolvidas devem de ser optimizadas de modo a consumir menos recursos possível.

- **Ambiente de desenvolvimento:** A tecnologia deve prover um bom ambiente de desenvolvimento, em que integre os compiladores necessários e simuladores dos diferentes SOs que suporta

2) Desenvolvimento Nativo: As aplicações mobile nativas são concebidas para apenas serem executadas num SO específico, apenas considerando o tipo de dispositivo e a versão a ser utilizada. Deste modo, o developer tem uma selecção limitada de linguagens que diferem de SO e IDE, sendo necessário utilizar vários SDK's e IDE para desenvolver a mesma aplicação para várias plataformas. Por exemplo, o Android suporta Java e Kotlin , iOS o Objective-C e Swift, e por fim, o C# para Windows Phone, no qual todos estes geram código binário diferente, obtendo assim um executável, semelhante ao processo utilizado para as aplicações Desktop [13] [14].

Pelo motivo de este fornecer acesso direto as API's (Interface de Programação de Aplicações) fornecidas pelo SO (como GPS, lista de contactos, bilioteca HTTP, entre outras), apresenta um melhor desempenho quando comparado com as aplicações mobile desenvolvidas em tecnologias Cross-Platform [16], como podemos visualizar na figura 2. No entanto, o desenvolvimento nativo torna difícil reutilizar código, pois cada plataforma utiliza uma linguagem diferente, tornando difícil manter alterações em todas as plataformas [15].

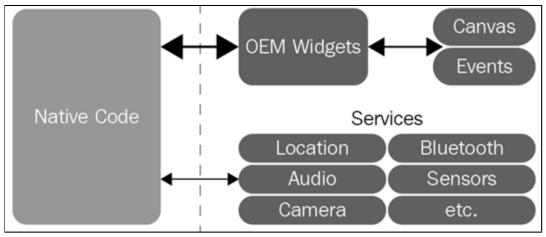

Figura 2. Arquitetura Native App

No momento de distribuir as aplicações, cada aplicação terá de ser transferida para as app stores (Loja de aplicações) de cada SO, sendo que cada loja tem um processo para avaliar se a aplicação cumpre com os requisitos da plataforma [16].

Esta abordagem de desenvolvimento mobile é a que envolve mais custos, uma vez que é necessário utilizar diferentes tecnologias e linguagens, sendo necessário realizar processos de codificação, manutenção e testes em cada plataforma [13].

3) Desenvolvimento Cross-Platform: Atualmente, das abordagens de desenvolvimento mobile, o desenvolvimento Cross-Platform é a mais utilizada, pelo motivo nos possibilitar desenvolver aplicações utilizando apenas uma linguagem, para varias plataformas [15].

Esta abordagem pode ser dividida em dois conceitos principais: abordagens em runtime (Tempo de execução) e abordagens generativas, ambas cujo o objetivo é construir uma aplicação mobile semelhante a uma aplicação nativa. A abordagem em runtime encontra-se dividida em três sub conceitos (WebApps, aplicações híbridas e *interpreted*), todas estas geram e processam o código em runtime. Por outro lado, a abordagem generativa encontra-se dividida em dois sub conceitos (Cross-Compiled e MDSD), onde utilizam compiladores para gerar o código em código nativo especifico do SO [2] [17].

a) Desenvolvimento WebApp (PWA): O conceito de desenvolvimento PWA iniciou-se em 2015, pela Google, em que pretendia juntar uma aplicação web com as aplicações mobile, permitindo desempenhar funções em modo *offline* em conjunto com notificações. Este conceito consiste numa aplicação web responsiva (Aplicação que se adequa a qualquer plataforma: Desktop, mobile e tablet) e progressiva (Aplicação que funciona independentemente do *browser* escolhido), no qual se adapta ao dispositivo que está a ser executado. E que é construída a partir de linguagens Web, JavaScript, HTML e CSS, como podemos visualizar na figura 3 [17].

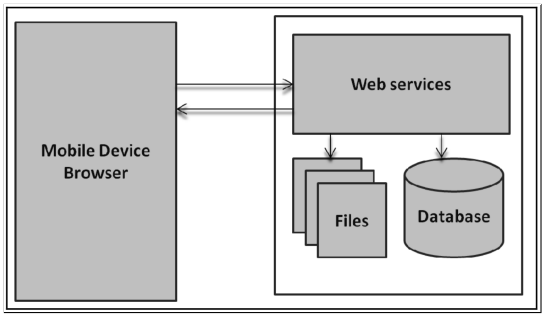

Figura 3. Arquitetura Web Application [18]

Esta abordagem distingue-se da abordagem nativa pela capacidade de ser executada em várias plataformas, independentemente do SO, pois é processada pelo *browser*. Pelo facto da aplicação ser executada no *browser*, não é possível instalar pelas app stores, só podem ser executado através de um URL (Uniform Resource Locator), ou instalada pelo próprio *browser* [18] [13].

O desenvolvimento PWA contem um conjunto de características que a torna uma abordagem adequada para desenvolvimento web e mobile [19] [20]:

1) O seu desenvolvimento progressivo, torna-o independente do *browser*.
2) É responsivo, adapta-se a qualquer dispositivo.
3) Tem funcionalidades *offine* através da utilização de *workers*
4) Pode ser descarregada do *browser*, e apresentado como um ícone no dispositivo
5) Utiliza apenas protocolos de comunicação seguros (HTTPS).

A vantagem da abordagem PWA é que estas comportam-se de forma semelhante em todos os *browsers* em todas as plataformas, possuindo interfaces idênticas [21]. Contudo, não permite o acesso às funcionalidades nativas do SO, como por exemplo, Gps ou camera, entre outros, limitando assim algumas funcionalidades que o developer necessita. Uma outra desvantagem é, pelo facto de utilizar um *browser*, é necessário considerar que o tempo que renderização das paginas web

poderá ser demorando e dispendioso, sendo necessário ter em consideração vários padrões de implementação [14] [20].

b) Desenvolvimento Híbrido: Através da abordagem de desenvolvimento WebApp, permitiu-se criar um outro conceito, que possibilitou os *developers* de aplicações web, criar aplicações mobile utilizando apenas linguagens de desenvolvimento web em conjunto com uma API em JavaScript, [10] onde esta API permite realizar chamadas as funcionalidades do SO e também utilizar os sensores disponíveis do dispositivo.

Com base no mesmo conceito de desenvolvimento WebApp, esta abordagem também é processada por um *browser*, mas é encapsulado numa base nativa chamada de WebView. A base nativa será encarregue de renderizar esta componente, tornando assim possível aceder aos recursos do SO, [3] como podemos ver na figura 4.

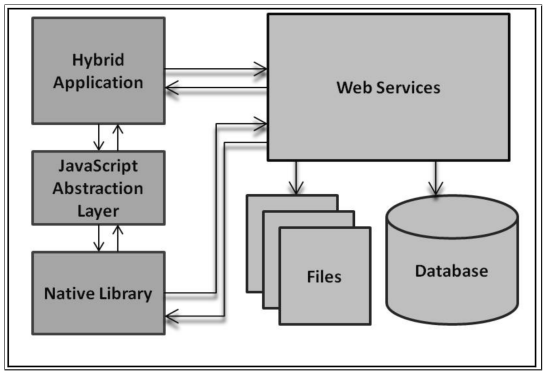

Figura 4. Arquitetura Hybrid [18]

A combinação do acesso às funcionalidades do SO, o comportamento nativo, e por ultimo, a facilidade de implementação da interface à custa das linguagens conhecidas, tornam esta abordagem muito popular [13]. De forma oposta às aplicações PWA, esta abordagem permite distribuir as aplicações pelas app stores, obtendo características nativas. Contudo estas aplicações têm um desempenho inferior em comparação com as abordagens nativas, uma vez que os processos são novamente realizados pelo *browser*, estando também sujeitas a vulnerabilidades na camada de abstração de JavaScript [14].

c) Abordagem Interpreted: Em contraste com a abordagem de desenvolvimento híbrida, as aplicações produzidas com esta abordagem contêm um runtime autónomo, isto é, não é utilizado um interpretador externo (como um browser). Deste modo o código é interpretado por diferentes plataformas sem utilizar um *browser*, [18] melhorando assim o desempenho e utilizando as API's através da camada de abstração, figura 5.

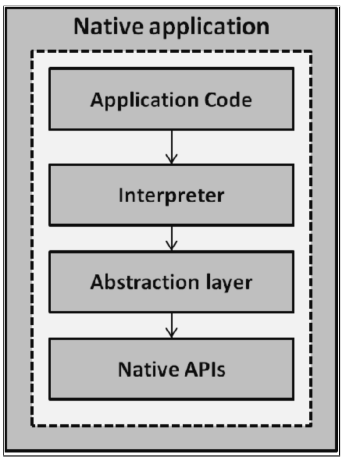

Figura 5. Arquitetura Interpreted [18]

Esta abordagem fornece um aspecto e sensação de aplicação nativa, sendo possível manter a mesma funcionalidade e estrutura de interface, mas pode ter um mau desempenho devido a interpretação em runtime [18]. As aplicações através desta abordagem são construídas a partir de um único projeto, em que a maior parte é transposto para código nativo e apenas uma pequena parte é interpretada. Assim, a sua implementação é independente da plataforma e do SO, permitindo a utilização de componentes de interface nativos [13] [10].

d) Abordagem Cross-Compiled: A abordagem *Cross-Compiled* consiste um criar uma aplicação completamente nativa utilizando a mesma linguagem entre várias plataformas, através do compilador que converte o código para código binário nativo do SO. O principal componente desta abordagem é o compilador que é responsável pela produção de código que possa ser executado numa determinada plataforma, ver figura 6.

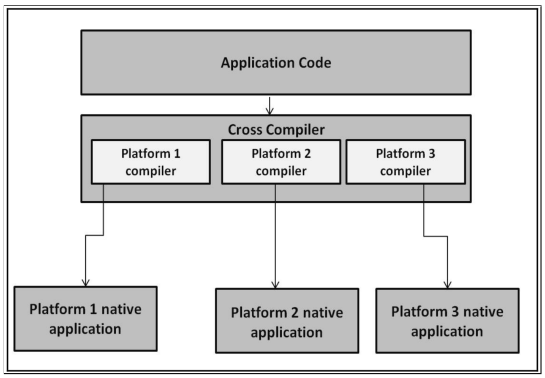

Figura 6. Arquitetura Cross Compiled [18]

Através do compilador, o código produzido torna-se nativo, desta forma todas as chamadas e funcionalidades do sistema tornam-se mais eficientes e fiáveis, como não depende de uma camada abstrata, todo a aplicação consegue ter um melhor desempenho [10].

A principal vantagem desta abordagem é o facto das aplicações serem capazes de atingir um desempenho nativo,

fornecendo todas as características das aplicações nativas, juntamente com os componentes nativos. Mas por outro lado, é extremamente difícil de identificar e corrigir erros na fase de compilação, e algumas funcionalidades terão de ser rescrevidas devido ao funcionamento diferente entre plataforma [18] [1].

e) Abordagem Model Driven Software Development: Por fim, a abordagem *Model Driven Software Development* é a menos popular de todas a outras abordagens, pois tem uma complexidade muito superior. Esta abordagem originou-se na engenharia de software orientada a modelo, que consiste numa metodologia de desenvolvimento que se foca principalmente na criação e exploração de modelos abstratos, modelos estes que possam representar problemas abstratos ou específicos [22]. Desta forma, o foco da abordagem orientada a modelos consiste em aumentar a produtividade, maximizando a compatibilidade entre sistemas.

Esta abordagem orientada por modelos existe há muitos anos, com o objetivo de gerir a variabilidade e centrando-se no modelo como representação abstrata de um sistema real. Como abordagem mobile, esta permite o desenvolvimento de aplicações Cross-Platform utilizando um nível de abstracção elevado, através de utilização de linguagens textuais, DSL (Linguagem de domínio específico) ou modelações como UML [10].

A grande vantagem desta abordagem, é o facto de utilizar atividades de modelação, assim o developer pode construir uma aplicação em alto nível, sem ter de lidar com técnicas de baixo nível, como por exemplo, a forma de guardar dados, notificações de sistema, entre outros [9]. Toda a interface da aplicação é construída somente por componentes nativos, e da mesma forma que a abordagem *Cross-Compiled*, todo o código produzido é compilado para o código binário nativo específico da plataforma, resultando assim, um bom desempenho e todas as funcionalidades do SO [14].

4) Métodos e Métricas de Avaliação: Na secção anterior foram apresentadas várias abordagens para desenvolvimento *Cross-Platform*, descrevendo e comparando as suas principais características e apontando as suas principiais diferenças. Considerando estes vários tipos de abordagens descritas, é de maior relevância compreender como poderão ser avaliados de uma forma analítica e rigorosa. Nesta secção será apresentada uma investigação das abordagens utilizadas pela industria e academia para analisar as a aplicações mobile *Cross-Platform*, os casos de estudo realizados, e formas de efetuar *benchmarks* para obter todos os dados relevantes sobre o consumo de recurso e outros aspectos relevantes para a avaliação de uma tecnologia *Cross-Platform*.

a) Métodos para avaliação: Atualmente existem vários métodos para avaliar e analisar tecnologias de desenvolvimento de software, deste a execução de um conjunto de questionários, à construção de outras tecnologias e ferramentas para realizar testes e medições [23]. Para avaliar as tecnologias de desenvolvimento de aplicações mobile, são propostos vários métodos e procedimentos, sendo que alguns destes são adaptações ou combinações de outros métodos.

Um dos métodos mais famosos na construção de investigações, são os questionários ao publico alvo, que permitem a colheita de informações, utilizando sondagens ou inquéritos. O questionário consiste num método de investigação, composto por um número de questões, com o objetivo de adquirir uma coleção de dados, necessários para formar componente qualitativa [24]. Os autores Catolino et al. do estudo [25] realizaram várias etapas para extrair métricas relevantes (características, interface, custo, funcionalidades, tamanho de aplicação) que possam ser encontradas na fase inicial de desenvolvimento de aplicações moveis, com o objetivo de comparar as aplicações móveis com as aplicações web. A primeira etapa consiste na recolha de métricas através da análise das citações online e a segunda uma entrevista a quatro *developers* experientes na área de mobile, de forma a validar o conjunto inicial de métricas obtidos na primeira etapa. No trabalho de Biørn-Hansen et al. [8], os autores elaboraram várias etapas para recolher dados de profissionais sobre conhecimento, utilização, questões relacionadas com o desenvolvimento Cross-Platform e os resultados da abordagem (Figura 7). Inicialmente criaram questões de investigação e de seguida um questionário de escolha múltipla, com um total de cinco perguntas de forma a recolher dados suficientes para responder as questões de investigação. Antes da escolha dos participantes, realizaram-se testes para assegurar a fiabilidade do questionário. Já, na dissertação de Santos [15], foi elaborado um questionário com questões de escolha múltipla, que se dividem em três grupos, informações pessoais, informação sobre a experiência do participante e informações acerca das tecnologias de desenvolvimento mobile Cross-Platform. O objetivo deste questionário foi obter dados relevantes para construir um sistema de recomendação baseado em conhecimento. O resultado do questionário permitiu a escolha de 12 critérios relevantes aos participantes expostos na tabela abaixo I. Por fim, na investigação de Angulo et al. [26], após uma série de testes numa aplicação nativa e numa aplicação Cross-Platform, foi pedido aos participantes para preencherem uns questionários de comparação final, sendo estes o Escala de Usabilidade do Sistema (SUS) de forma a recolher primeiras ideias dos participantes, e o Questionário de Experiência do Utilizador (UEQ), focando nos únicos aspectos aplicáveis ao contexto de utilização, e um questionário final onde compararam o aspecto e comportamento das tecnologias.

Tabela I
LISTA DE CRITÉRIOS OBTIDOS PELOS PARTICIPANTES, PARA A CONSTRUÇÃO DE UM SISTEMA DE RECOMENDAÇÃO BASEADO EM CONHECIMENTO ELABORADO NA DISSERTAÇÃO DE SANTOS [15]

Numero	Critério
1º	Plataformas suportadas
2º	Linguagem de programação
3º	Tipo de aquisição (Grátis, Flexível ou Pago)
4º	Licenças
5º	Suporte a compras internas
6º	Facilidade na distribuição para as app stores
7º	Utilização de IDE de desenvolvimento
8º	Recursos suportados
9º	Geração de código nativo
10º	Tempo de adopção da versão mais recente das plataformas suportadas
11º	Atualizações constantes da tecnologia
12º	Suporte a criação de interface

	Familiarity *(Q#1)*		Interest *(Q#2)*		Usage *(Q#3–4)*	
Framework *(alphabetic order)*	Totals	Averages *(max 5)*	Totals	Averages *(max 5)*	*N (Hobby)*	*N (Prof.)*
Cordova ∗	-	-	-	-	2	0
Fuse by Fusetools	124	1.23	162	1.6	3	1
Intel XDK	132	1.31	147	1.46	2	0
Ionic Framework	289	2.86	281	2.78	37	17
jQuery Mobile	280	2.77	164	1.62	12	9
Meteor ∗	-	-	-	-	13	12
NativeScript by Telerik	159	1.57	199	1.97	5	1
PhoneGap	338	3.35	306	3.03	47	32
React Native by Facebook	302	2.99	432	4.28	46	26
Reapp	135	1.34	174	1.72	4	2
Sencha Touch	174	1.72	138	1.37	1	1
Tabris.js	112	1.11	145	1.44	1	0
Titanium by Appcelerator	173	1.71	149	1.48	3	3
Touchstone.js	123	1.22	164	1.62	2	1
Xamarin Forms	166	1.64	166	1.64	6	2
None		-	-	-	14	30

Figura 7. Resultados dos questionários na investigação de Biørn-Hansen et al. [8] realizados em várias etapas para recolher dados de profissionais sobre popularidade, utilização e questões relacionadas com o desenvolvimento Cross-Platform

A revisão sistemática é uma investigação cientifica que agrega vários estudos relevantes à investigação permitindo adquirir vários dados úteis para a construção de uma analise comparativa [27]. Através da revisão de literatura, vários investigadores conseguiram obter dados suficientes para construir listas de critérios, atribuindo pesos a cada item escolhido, avaliando assim uma abordagem ou tecnologia. Na investigação de Rieger et al. [28], construíram uma lista de critérios essenciais para caracterizar e avaliar as abordagens e tecnologias de desenvolvimento mobile, no qual os pesos de 1 a 5 foram atribuídos por investigadores e experientes na área. A lista de critérios foi criada através de dados de vários artigos e separada em 4 categorias, infraestrutura, perspectiva de desenvolvimento, perspectiva de aplicação e perspectiva de utilização. Um estudo similar de Heitkötter et al. [29], utilizou uma lista de critérios semelhante, mas apenas separou a lista em 2 categorias, perspectiva de infraestrutura e perspectiva de desenvolvimento, avaliando cada critério através de conclusões de vários estudos. Da mesma forma que as investigações anteriores, Xanthopoulos et al. [30], propuseram um conjunto de critérios para avaliar e comparar as abordagens de desenvolvimento mobile, sendo os critérios distribuição no mercado, tecnologias generalizada, hardware e acesso a dados, interface do utilizador e percepção de desempenho pelo utilizador. Outras contribuições de Bernardes et al. [31], Amatya et al. [32] , Charkaoui et al. [33] e Ahmad et al. [34] analisaram e compararam as várias abordagens de desenvolvimento mobile Cross-Platform através das informações obtidas de vários artigos. As investigações de Bernardes et al. [31] e Charkaoui et al. [33] indicaram quais os pontos fracos e fortes de cada abordagem de desenvolvimento, e os estudos de Ahmad et al. [34] e Amatya et al. [32] assinalaram os desafios encontrados em cada uma das abordagens de desenvolvimento.

De um modo mais prático, o desempenho e o consumo de recursos de uma aplicação têm uma enorme importância na escolha de uma tecnologia, sendo que existem atualmente muitos estudos e artigos que analisam vários parâmetros do dispositivo ao executar uma determinada aplicação, como podemos ver na tabela II. O estudo desenvolvido por Dorfer et al. [35], utilizou dois cenários de teste controlados, o primeiro media o consumo de CPU e RAM, e o segundo o consumo de bateria, executando os dois cenários 10 vezes. Todos os testes foram criados utilizando a ferramenta UI *automator*, que permite executar comandos de interface autonomamente, e seguindo um conjunto de passos definidos pelos autores, mostrados na figura 8. Dalmasso et al. [7] realizaram testes de desempenho de forma a comprar a aplicação criada com o Titanium e outras com o PhoneGap, utilizando apenas o sistema Android. Representando uma aplicação empresarial, tinha vários botões na interface, cada um com um tipo de pedido web (AJAX, REST e SOAP) e analisava e exibia diferentes formatos de respostas (Texto, XML, JSON). A investigação de Biørn-Hansen et al. [17], realizou testes em seis dispositivos moveis, de modo a abranger várias gamas de dispositivos, efetuando cinco testes com focos diferentes, analisando os parâmetros, CPU, RAM em estado de inactividade e ocupação e tempo de conclusão de uma certa tarefa. Os testes foram escolhidos para serem executáveis na sua maioria isoladamente, evitando configurações complexas que dependem de factores externos, tais como a qualidade da rede. Estes métodos fornecem uma estrutura para uma execução de benchmarking em conjunto com ferramentas de monitorização, que será referida na secção II-A4c.

Steps	CPU and memory test	Battery test
Step 1	Start the measurement tool (Recording time: 3 min. 16 sec.).	Start the measurement tool (Recording time: 15 min. 16 sec.).
Step 2	Press the home screen button and wait 2 seconds.	Press the home screen button and wait 2 seconds.
Step 3	Start of the installed app by an intent.	Start of the installed app by an intent.
Step 4	Run the app for 3 minutes.	Run the app for 15 minutes.
Step 5	Press the home screen button and wait 2 seconds.	Press the home screen button and wait 2 seconds.
Step 6	Open recently opened apps and wait 2 seconds.	Open recently opened apps and wait 2 seconds.
Step 7	Close the app.	Close the app.

Figura 8. Lista de passos utilizados nos dois cenários de teste controlados no estudo de Dorfer et al. [35]

De uma forma diferente dos estudos anteriores. Delia et al. [13] elaborou experiências em todas as abordagens de desenvolvimento mobile, e em ambos os SOs (Android e iOS). Os testes foram realizados em seis dispositivos, num total de 42 casos de teste, e estes testes consistiam num calculo matemático incluindo várias iterações, avaliando assim a ve-

locidade de processamento e o tempo de execução necessário para efectuar cálculos matemáticos intensivos. Para cada caso de teste, foram realizadas 30 execuções separadas, obtendo em cada caso uma amostra de tempo, e para caracterizar cada uma das amostras obtidas, foram calculadas variáveis estatísticas, como a média da amostra e o desvio padrão da amostra. De forma a conseguir medir o consumo de energia de um telemóvel e a energia consumida por cada aplicação sem a utilização de ferramentas de software, Ciman et al. [36] decidiu utilizar dispositivos que tivessem acesso direto à bateria para usar um Monsoon PowerMonitor8 externo. A principal função do PowerMonitor8 seria medir a energia do telemóvel, controlando o nível exigido de energia ao realizar as funcionalidades das aplicações mobile.

Outros estudos e contribuições propostas por Willocx et al. [37], Willocx et al. [38], Rösler et al. [39], Jia et al. [40], Bekkhus et al. [41] e Corbalan et al. [42], focaram-se numa aplicação empresarial, sendo que os teste de desempenho consideraram apenas as funcionalidades da aplicação, sem seguir um processo específico ou algum padrão para recolher e analisar os dados obtidos pelos testes.

Tabela II
PARÂMETROS DE DESEMPENHO AVALIADOS PELOS ESTUDOS

Estudo	Parâmetros
Bekkhus et al. [41]	CPU, RAM, Tempo médio de execução, Bateria
Biørn-Hansen et al. [17]	CPU, RAM, API's(Aceletometro, Contactos, Sistema de ficheiros, Localização)
Corbalan et al. [42]	CPU, RAM e Bateria
Ciman et al. [36]	Bateria
Dalmasso et al. [7]	CPU, RAM e Bateria
Dorfer et al. [35]	CPU, RAM e Bateria
Jia et al. [40]	Tempo de renderização, RAM, Tempo de resposta, Espaço no disco
Que et al. [43]	CPU, RAM, Bateria, Tempo de instalação, Tempo para iniciar, Fluxo de rede
Rawassizadeh et al. [44]	CPU, RAM, Espaço no disco, Bateria, Rede
Ryan et al. [45]	CPU, RAM e Bateria, Fluxo de rede, ECS, SOS, OMS, EMS, NEI, SSO, NI
Willocx et al. [37]	CPU, RAM, Tempo para iniciar, Espaço no disco, Tempo de pausa e resumo
Willocx et al. [38]	CPU, RAM, Espaço no disco, Tempo de resposta, Bateria

Um outro modo de avaliar o desempenho de uma tecnologia ou abordagem de desenvolvimento mobile, consiste na criação de um software ou uma tecnologia que realiza testes automáticos, semelhantes entre várias tecnologias. O estudo de Dhillon et al. [46], apresentou uma framework (figura 9), com o objetivo de realizar testes de desempenho para avaliar as tecnologias de desenvolvimento mobile Cross-Platform. O âmbito da estrutura é limitado a aplicações que não são graficamente intensivas, sendo que não permite aplicações com o foco em jogos ou processamento gráfico intensivo (aplicações de realidade aumentada, realidade virtual e realidade mista). A avaliação consiste em três fases, a primeira centra-se na determinação das capacidades da estrutura e caracteristicas das ferramentas (ambiente de desenvolvimento, acesso ao telemovel, sensores, segurança, entre outros), a segunda fornece um conjunto de parâmetros de referencia que devem ser implementados utilizando as tecnologias mobile Cross-Platform (Benchmarks de desempenho e usabilidade), e por fim a terceira fornece os dados da ferramenta estudada, onde são discutidos itens de experiências de desenvolvimento a fim de compreender o resultado.

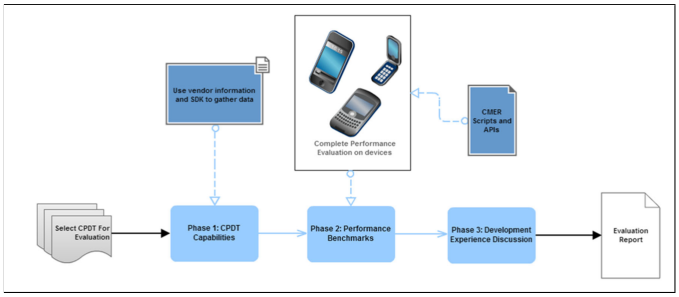

Figura 9. Framework apresentada por Dhillon et al. [46] para avaliar certos tipos de tecnologias de desenvolvimento mobile Cross-Platform

Os testes de desempenho têm uma grande influencia na avaliação de uma tecnologia, contudo muitos investigadores também dão importância aos testes de usabilidade e funcionalidade. O estudo realizado por Hussain et al. [47], reuniu 26 artigos para obter orientações sobre a construção de um modelo de avaliação de usabilidade. Para assegurar que o modelo construído pelos autores é fiável, realizaram experiências para testar a usabilidade das aplicações em dois telemóveis diferentes. Por outra lado, o trabalho de Angulo et al. [26] focou-se na avaliação do impacto da interface gráfica, entre uma aplicação nativa e outra criada pelo Titanium. Para decidir qual a abordagem que tem uma melhor usabilidade, realizaram dois estudos com 37 participantes. O primeiro estudo era laboratorial, que avaliava a primeira impressão dos utilizadores, realizando cinco tarefas especificas que abrangiam todas as funcionalidades da aplicação. O segundo estudo longitudinal, onde os participantes utilizavam a aplicação durante cinco dias em circunstancias reais, utilizando os seus próprios telemóveis. Por ultimo, Sommer et al. [4] realizou uma investigação, onde criaram cinco aplicações idênticas, três com tecnologias Cross-Platform e duas em tecnologias nativas, com o foco de conhecer a utilidade das soluções Cross-Platform para o developer e as vantagens para as pequenas e médias empresas que desenvolvem aplicações móveis. Utilizando o modelo FURPS+ (Funcionalidade, usabilidade, fiabilidade, desempenho, suporte ao desenvolvimento, implementação, suporte, custos) avaliaram cada aplicação dando um peso entre uma a cinco valores.

Category	*Weight*	*Titanium*	*Rhodes*	*PhoneGap and Sencha Touch*	*Android SDK*	*iOS SDK*
Functionality	4	3	2	3	**4**	**4**
Usability features	3	4	3	3	**5**	**5**
Developer support	2	**4**	3	**4**	**4**	**4**
Reliability & Performance	3	3	2	2	4	**5**
Deployment, Supportability, Costs	5	**4**	**4**	**4**	**4**	3
Rounded final score:		**3.6**	**2.9**	**3.2**	**4.2**	**4.1**

Figura 10. Aplicação do modelo FURPS+ para avaliar cada aplicação, realizado na investigação de Sommer et al. [4]

b) Provas de Conceito: Provas de Conceito podem ser a base de toda a analise de uma comparação, sendo possível investigar ao pormenor, eliminando possíveis ambiguidades [48]. Esta abordagem tem uma grande importância pelo facto de ser possível visualizar as diferenças e as limitações que são implementadas pelas tecnologias. Os casos de estudo são muito utilizados para a realização de testes no desenvolvimento de software, da mesma forma, são também utilizados no desenvolvimento mobile, principalmente para garantir que a tecnologia suporta todos os requisitos necessários para resolver o problema inicialmente proposto e se a linguagem e estrutura de código não dificultam a manutenção e adição de funcionalidades [49].

Um simples caso de estudo foi realizado por Xanthopoulos et al. [30] que construiu uma aplicação utilizando a tecnologia Titanium, com o objetivo que a aplicação realizasse pedidos HTTP, mostrando os dados em uma lista, utilizando apenas JavaScript e menos que 150 linhas de código. Através deste caso de estudo conseguiu demonstrar que apenas com uma linguagem e poucas linhas de codigo, era possível criar a mesma aplicação para dois sistemas diferentes (Android e iOS). Um caso de estudo semelhante foi realizado por Smutný [50], com o objetivo de comprovar que as linguagens nativas podiam ser substituídas por linguagens Web. Utilizando a tecnologia jQuery mobile, construiu uma aplicação (tradutor automático de termos das línguas Inglês-Checo) utilizando código já existente da aplicação web (figura 11), apenas ajustando o tamanho da janela e modificando os eventos do rato para para os eventos customizados (tap, swipe, orientação, etc...).

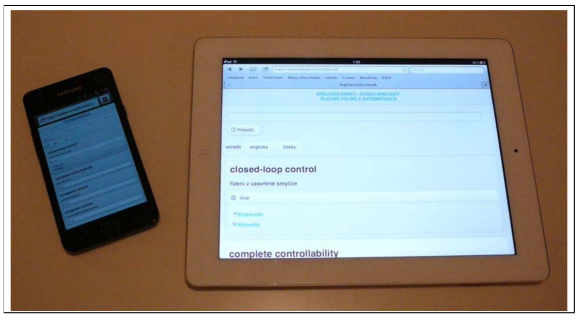

Figura 11. Caso de estudo realizado por Smutný [50] utilizando código já existente da aplicação web, transformando numa aplicação mobile

Uma das formas de comparar as diferenças entre as tecnologias de desenvolvimento de aplicações nativas e Cross-Platform consiste na construção de duas aplicações em tecnologias diferentes com o mesmo intuito. Angulo et al [26], realizou um estudo em que construiu dois protótipos em que ambos foram sujeitos a uma avaliação, sendo que o prototipo mais benéfico seria depois utilizado pelos estudantes para estes terem acesso à informação universitária em tempo real, incluindo dados dos professores, entre outras informações. A primeira aplicação foi criada com tecnologias nativas e a segunda com a tecnologia Titanium. Num estudo semelhante conduzido por Dorfer et al. [35], os autores desenvolveram a mesma aplicação duas vezes, uma delas utilizando a abordagem nativa do Android, e a outra utilizando a tecnologia React Native (figura 12). Esta aplicação consiste na procura de restaurantes, utiliza o GPS para determinar a posição atual, a API do Google Places para encontrar os restaurantes próximos sendo mostrados num mapa e numa lista, e ligação a Internet para realizar pedidos HTTP. O objetivo das aplicações era obter resultados de qual a que consumia mais recursos. O artigo realizado por Que et al. [43], construiu um caso de estudo similar aos anteriores, utilizando a tecnologia Cordova e tecnologias nativas, com o foco de testar as funcionalidades do sistema, como a camara, GPS, entre outros, de forma a obter uma indicação de qual a melhor abordagem.

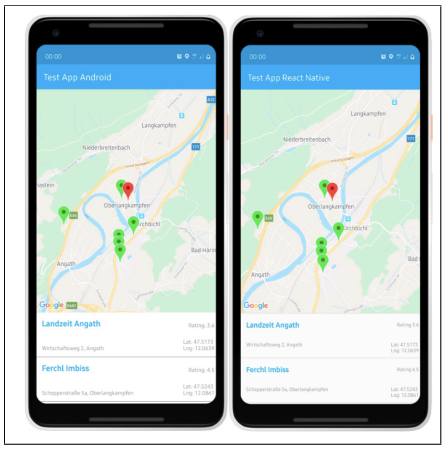

Figura 12. Caso de estudo construído por Dorfer et al. [35], com o objetivo de comparar uma aplicação nativa com uma Cross-Platform

Da mesma forma, como os estudos anteriores, os casos de estudo também são utilizados para realizar comparações entre diferentes tecnologias de desenvolvimento de aplicações mobile Cross-Platform. Biørn-Hansen et al. [17] desenvolveu um estudo sobre as aplicações WebApp, no qual construiu três aplicações como caso de estudo (figura 13), a primeira aplicação construída com a tecnologia Ionic (hibrida), a segunda criada com o React Native (*Interpreted*) e por fim, a terceira com o React (PWA), com o objetivo de realizar comparações técnicas entre as tecnologias que utilizam linguagens web. Um outro estudo realizado por Sommer et al. [4], construiu como caso de estudo, uma aplicação comercial "MobiPrint"para apoiar os clientes de lojas de impressão fotográfica, criando

a mesma aplicação em três tecnologias, Titanium, Rhoads e PhoneGap. Entre outros pontos, a aplicação testa, a integração com serviços web (HTTP), a localização (GPS), acesso ao sistema de ficheiros e por ultimo, funcionaliades de usabilidade. Em comparação aos artigos anteriormente descritos, existem outros mais abrangentes, como os estudos de Ciman et al. [36], Corbalan et al. [42], Delia et al. [51] e Willocx et al. [37], em que construíram um caso de estudo nas várias abordagens de desenvolvimento de aplicações Cross-Platform em conjunto com a abordagem nativa, seguindo um conjunto de requisitos impostos pelos autores. A aplicação de Ciman et al. [51], consiste numa versão mobile de um ambiente virtual de ensino (WebUNLP) que não estava adaptado para dispositivos moveis. Após a construção dos casos de estudo, os autores construíram uma análise com as limitações e problemas de cada abordagem ao encontro com os requisitos. Por outro lado, Corbalan et al. [42] construíram três aplicações diferentes com objectivos, processamento intensivo, reprodução de vídeo e reprodução de áudio, a fim de medir o consumo de energia causado por cada abordagem.

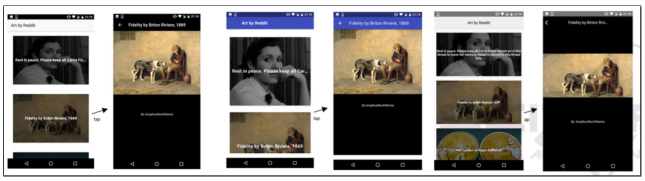

Figura 13. Caso de estudo desenvolvido por Biørn-Hansen et al. [17], com o objetivo de realizar comparações técnicas entre as tecnologias que utilizam linguagens web (Ionic, PWA e React Native)

c) Ferramentas de Benchmarks: Sendo que os casos de estudo são utilizados para analisar as limitações de uma tecnologia, muitas vezes é necessário avaliar as características de performance do software no hardware. Dito isto, o Benchmarking consiste numa avaliação de características de desempenho de um software em um especifico hardware, como por exemplo, o consumo de recursos [52]. Nos dias de hoje, existem muitas ferramentas para avaliar a performance de um software, como a ferramenta ADB do Android que permite verificar a utilização do CPU, memoria, entre outros [35]. Para algumas empresas, a escolha de uma tecnologia depende grande parte, dos resultados dos Benchmarks, geralmente quando necessitam de um software rápido e estável.

Sendo o SO Android um sistema Open-Source, o processo de Benchmark torna-se mais controlado e detalhado em comparação com outros SOs, possuindo uma grande variedade de ferramentas para analisar os sensores e componentes de hardware [42]. Através da ferramenta ADB (Android Debug Bridge) é possível verificar aspectos de performance de um software num especifico hardware. O trabalho realizado por Biørn-Hansen et al. [17] utilizou esta ferramenta para monitorizar o tempo de lançamento da atividade [1]. Os estudos de Dorfer et al. [35], Dalmasso et al. [7], Willocx et al. [37], Willocx et al. [38] e Bekkhus et al. [41] utilizaram esta ferramenta para executar medições contínuas do CPU, memoria e utilização de bateria, sendo que os resultados foram produzidos pelos casos de teste em intervalos de segundo, realizando reset na ferramenta antes de cada teste. Uma outra ferramenta fornecida pelo Android é o Android Profiler que apresenta dados de perfil, utilizada pelo artigos Bekkhus et al. [41], Biørn-Hansen et al. [10], Willocx et al. [38] e Dalmasso et al. [7], que é semelhante à anterior, mas permite uma analise pormenorizada da actividade do CPU, memória, tempos de resposta e tráfego de rede, em conjunto com uma interface gráfica. Contudo, existem outras ferramentas de Benchmark criadas por terceiros, utilizadas em artigos, como:

- Power Tutor, utilizado na investigação de Dalmasso et al. [7], para medir o consumo de energia de aplicações individuais.
- Trepn Profiler da Qualcomm, utilizado no trabalho de Corbalan et al. [42], para medir e analisar o consumo de energia em cada um dos componentes de hardware.
- Little Eye4, utilizado no artigo de Willocx et al. [37], para monitorizar um conjunto de parâmetros relacionados com a performance para qualquer processo em execução no dispositivo.
- Uma ferramenta de monitorização de recursos desenvolvida por Rawassizadeh et al. [44], que rastreia e regista o consumo de recursos de um determinado processo, sendo executado em paralelo como uma aplicação alvo.

Por outro lado, sendo o SO iOS um sistema Close-Source, torna processo de Benchmark mais dificultado em comparação ao SO Android, possuindo apenas algumas ferramentas para analisar o comportamento de hardware. Os estudos de Willocx et al. [37] e Willocx et al. [38] utilizam a ferramenta Activity Monitor (Monitor de atividade) fornecida pela Apple, para monitorizar alguns aspectos de performance em certos componentes de hardware, como o CPU e memoria. Uma outra ferramenta fornecida pela Apple, é o Time Profiler do Xcode, que permite analisar a performance de uma aplicação e encontrar problemas de memória, tais como fugas e zombies [2] [54].

B. Contribuições

Dado o problema proposto na secção I-A, esta dissertação possui objectivos para resolver o problema apresentado tanto a nível académico, como empresarial. Estas contribuições resultarão da concretização dos objectivos descritos na secção I-B, no qual irão facilitar futuras investigações a respeito ao tema proposto (CONT1, CONT2), e irá promover uma melhor escolha entre as tecnologias de desenvolvimento mobile Cross-Platform analisadas (CONT3, CONT4). Através dos dados adquiridos

Esta dissertação pretende contribuir com os seguintes pontos:

[1]Uma atividade fornece a janela na qual a aplicação desenha a própria interface gráfica. Essa janela normalmente preenche o ecrã, mas pode ser menor do que o ecrã e flutuar sobre outras janelas. [53]

[2]Zombies são valores que foram retirados da RAM mas ainda existe referencia a este, levando a problemas inesperados.

1) **CONT1:** Revisão de literatura sobre o desenvolvimento mobile Cross-Platform, avaliando e comparando as diferentes tecnologias.
2) **CONT2:** Análise individual de várias tecnologias Cross-Platform, utilizando um estudo de caso e ferramentas benchmark.
3) **CONT3:** Análise comparativa entre várias tecnologias Cross-Platform.
4) **CONT4:** Elicitar uma série de recomendações para a seleção de uma tecnologia de desenvolvimento mobile para um problema proposto.

III. Solução Proposta

De forma a resolver os problemas detalhados na secção I-A e de modo a concretizar os objetivos descritos na secção I-B, esta dissertação irá se basear numa revisão de literatura, sobre as várias abordagens de desenvolvimento de aplicações moveis, além dos métodos e métricas utilizadas para as avaliar. Será realizado uma análise comparativa com objectivos académicos e empresariais, entre as várias tecnologias disponíveis, seleccionadas por critérios de inclusão e exclusão, de modo a fornecermos uma avaliação completa. Para esta analise será produzido um caso de estudo, que consiste no desenvolvimento de uma aplicação móvel semelhante em todas as tecnologias selecionadas, realizando assim uma analise individual, testando e recolhendo métricas como consumo de recursos, esforço de desenvolvimento, limitações entre outros parâmetros apresentados na tabela II. Além desta analise, a avaliação das tecnologias irá também ter uma componente qualitativa, em que serão realizados questionários ao publico alvo, de forma a aferir como estas ferramentas são utilizadas no dia a dia, por equipas que desenvolvem aplicações mobile. Com isto, esperamos recolher informação técnica e qualitativa sobre as tecnologias escolhidas, obtendo os dados necessários de forma a contribuir para uma melhor escolha da tecnologia Cross-Platform para solucionar um problema proposto, expondo as suas limitações.

Dito isto, as próximas subsecções expõem os métodos a seguir para a avaliação e comparação das tecnologias escolhidas, em conjunto com uma revisão de literatura extensiva de forma a identificar as principais diferenças entres as tecnologias mobile nativas e Cross-Platform. Todos os parâmetros para a avaliação foram escolhidos de forma a tornar essa avaliação mais precisa possível. De igual forma, todos estes métodos resultaram da avaliação às abordagens realizadas por investigadores, identificados e discutidos na subsecção II-A4.

A. Revisão de literatura extensiva

Após a identificação e discussão das abordagens de desenvolvimento mobile nativo e Cross-Platform, torna-se necessário especificar os tipos de tecnologias e ferramentas que são utilizados nas diversas abordagens de desenvolvimento mobile, e as diferentes implementações nativas e Cross-Platform. Nesta secção serão apresentados os vários tipos de tecnologias e ferramentas e de seguida, os diversos tipos de implementações de tecnologias de desenvolvimento mobile, identificando as principais características e diferenças.

B. Questionários

Seguidamente à revisão de literatura extensiva e aos métodos e métricas estudadas, será realizado o primeiro método de avaliação, sendo este um questionário ao publico alvo, incluindo na analise uma componente quantitativa. Esta componente têm um grande nível de importância, pois permite recolher informações relevantes sobre a utilização das tecnologias escolhidas no dia a dia, por equipas de desenvolvimento de aplicações mobile. O questionário proposto será baseado no questionário realizado por Biørn-Hansen et al. [8], pelo motivo que esta investigação possuía objectivos semelhantes aos esperados nesta componente (dados de profissionais sobre conhecimento, utilização, questões relacionadas com o desenvolvimento Cross-Platform), e antes de escolherem os participantes, realizaram testes para assegurar a fiabilidade do questionário. Assim sendo, o questionário será composto por seis perguntas de escolha múltipla, divido em três categorias, conhecimento, interesse e utilização, e como o publico alvo, *developers* de aplicações mobile.

C. Caso de Teste

Para construir uma análise individual de cada tecnologia escolhida, é necessário realizar métodos de teste, de forma a obter dados significativos sobre o desempenho da aplicação, o consumo de recursos e usabilidade. Este método é muito importante, pelo facto de ser executado na prova de conceito realizada em cada tecnologia. Este caso de teste irá consistir em dois cenários de teste controlados, sendo executados um numero indefinido de vezes e isolados, evitando configurações que dependam de factores externos, tais como a qualidade da rede, entre outros.

O primeiro cenário será um calculo matemático (semelhante ao utilizado no artigo de Delia et.al**??**) incluindo várias iterações, avaliando assim a velocidade de processamento e o tempo de execução necessário para efectuar cálculos intensivos, sempre monitorizando o consumo de recursos. Através do primeiro cenário, é possível obter dados significativos sobre como a tecnologia processa os dados e utiliza a memoria para guarda-los.

O segundo cenário consistirá numa lista de passos em que serão realizados sequencialmente. Este cenário será capaz de exigir o máximo da tecnologia, monitorizando o consumo de recurso e estabilidade da aplicação. A lista de passos a seguir será baseada na que foi realizada na investigação de Dorfer et al. [35], pelo motivo que conseguiu obter dados tecnológicos significativos para efectuar uma comparação entre tecnologias.

A lista de passos será a seguinte:

1) **Passo 1:** Iniciar a ferramenta de monitorização
2) **Passo 2:** Clicar no botão *home screen* e esperar 2 segundos
3) **Passo 3:** Abrir a aplicação
4) **Passo 4:** Realizar os seguintes testes:

a) **Passo 4.1:** Abrir janela de consumo de dados do servidor, esperar pelo resultado e depois fechar.
b) **Passo 4.2:** Abrir janela de CPU intensivo, esperar pelo resultado e depois fechar.
c) **Passo 4.3:** Abrir janela de consumo de API's do SO, esperar pelo resultado e depois fechar.

5) **Passo 5:** Fechar aplicação
6) **Passo 1:** Terminar a monitorização e fechar a ferramenta

D. Prova de conceito

Para realizar os cenários de teste acima descritos, será construído uma prova de conceito, que irá possibilitar uma investigação ao pormenor. Esta prova de conceito ajudará a eliminar possíveis ambiguidades entre as tecnologias, visualizando as suas limitações, permitindo verificar se a tecnologia suporta todos os requisitos necessários e se a linguagem e estrutura de código não dificultam a manutenção e adição de funcionalidades.

Dito isto, a prova de conceito proposta consistirá na construção de uma aplicação móvel em todas as tecnologias seleccionadas realizando assim uma analise individual, testando e recolhendo métricas como consumo de recursos, custo, esforço de desenvolvimento, limitações impostas pela arquitectura, desempenho, segurança, entre outros parâmetros representados na tabela III. Através dos dados obtidos pela construção da prova de conceito, será possível realizar uma análise comparativa completa, entre as várias tecnologias de desenvolvimento mobile Cross-Platform.

A aplicação terá de ir em contra com os seguintes requisitos:

1) **R1:** Janela principal com várias opções de escolha
2) **R2:** Área para executar um calculo matemático com várias iterações, expondo o valor final e com o tempo total de processamento
3) **R3:** Área para comunicação com servidor, consumindo vários dados obtidos por este
4) **R4:** Área para realizar acesso as API's fornecidas pelo sistema operativo e dispositivo.
5) **R5:** Área para gerar vários botões e janelas

E. Benchmarks

Sendo que a prova de conceito será utilizada para analisar as limitações das tecnologia tanto a nível de aplicação mobile como nível de ambiente de construção, é necessário avaliar as características de performance da aplicação num especifico hardware. Um outro método a ser utilizado será o Benchmarking, que irá monitorizar o consumo de recursos e características de desempenho durante a execução dos casos de teste descritos na subsecção III-C. Os parâmetros a monitorizar estão descritos na tabela III.

Para a realização dos benchmarks, para a prova de conceito em Android, será utilizado as ferramentas, ADB (Android Debug Bridge) que irá monitorizar os consumos de recursos da prova de conceito num especifico hardware, como CPU, memoria e utilização de bateria, em intervalos de tempo fornecido pelas ferramentas do CLI (Interface de Linhas de Comandos), e Android Profiler caso a tecnologia a suporte, para obter uma analise pormenorizada da actividade do CPU, memória, tempos de resposta e tráfego de rede, em conjunto com uma interface gráfica.

Já para a prova de conceito em iOS um será utilizado um conjunto de ferramentas chamado de *Instruments* fornecidas pela Apple, no Xcode. Este conjunto de ferramentas irão permitir monitorizar certos componentes de hardware, como CPU, Memoria e consumo de bateria.

Devido ao facto de alguns sistemas realizados por terceiros prejudicarem o bom funcionamento da prova de conceito, afetando os resultados obtidos, foram escolhidas estas ferramentas pelo motivo que foram utilizadas em vários estudos, sem comprometer a aplicação.

F. Métricas para avaliação

Ao longo da execução dos métodos descritos anteriormente, para cada tecnologia escolhida serão observado vários parâmetros de avaliação. Estes parâmetros de avaliação, foram obtidos e selecionados através da realização da revisão de literatura descrita na secção II e revisão de literatura extensiva apresentada na secção III-A, verificando quais os pontos mais importantes numa tecnologia de desenvolvimento mobile Cross-Platform. Estes parametros estão separados por categorias e sub categorias, em que serão importantes para realizar analise completa de cada tecnologia escolhida, e por fim, uma comparação entre tecnologias.

Os parâmetros de avaliação propostos são os seguintes:

Tabela III
PARÂMETROS DE AVALIAÇÃO PROPOSTOS

Categoria	Sub Categoria	Parâmetro
Aplicação	-	Tamanho final da aplicação (pronto para distribuição nas app stores)
		Estabilidade da aplicação entre plataformas
	Consumo de Recursos	CPU
		Bateria
		Memoria (RAM)
	Desempenho	Tempo de compilação
		Tempo de construção do APK
		Tempo médio da realização de testes
		Tempo médio para renderizar a aplicação
Ambiente	Desenvolvimento	Tamanho final do projeto
		Suporte da comunidade
		Suporte na resolução e demonstração de erros
		Dificuldades de ambiente e desenvolvimento
		Limitações e restrições da tecnologia
		Ferramentas disponibilizadas para desenvolvimento
		Segurança e Encriptação
	Dependências	Dependências de outras tecnologias
		Ferramentas disponibilizadas para gestão de dependências e aplicações

G. Planeamento

O planeamento de uma dissertação têm um grande nível de importância, pelo facto que de possibilita identificar e gerir as horas de trabalho necessárias desde o inicio até à entrega final da tese. Dito isto, foi realizado o planeamento da tese utilizando o diagrama de Gantt (figura 14,15 e 16), que permite-nos visualizar o cronograma de um projeto, ajudando na gestão e nas tarefas realizadas, identificando quando foram iniciadas, a sua duração e previsão de término.

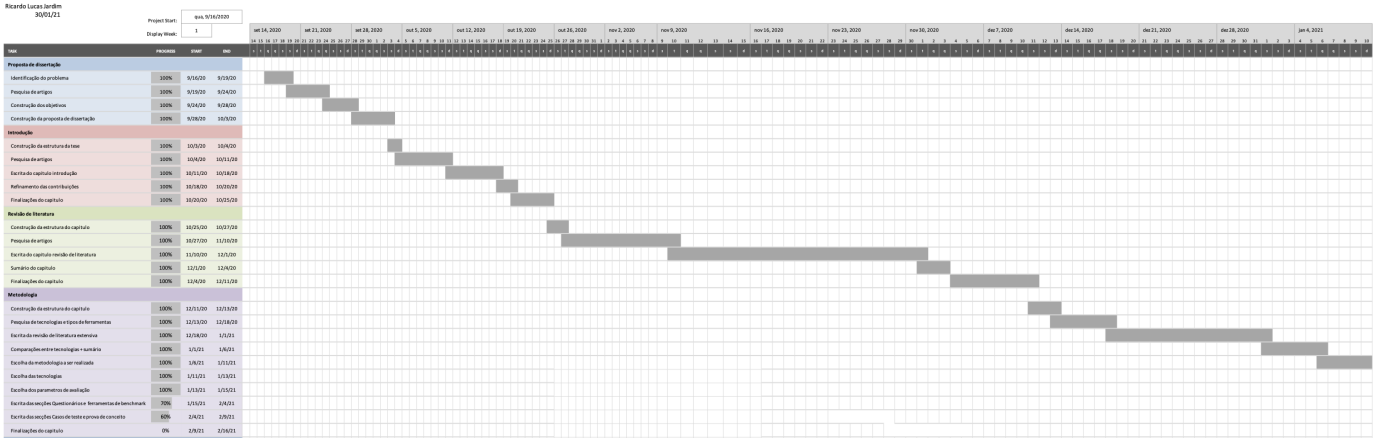

Figura 14. Planeamento de Gannt, parte 1 (14 de Setembro a 10 de Janeiro)

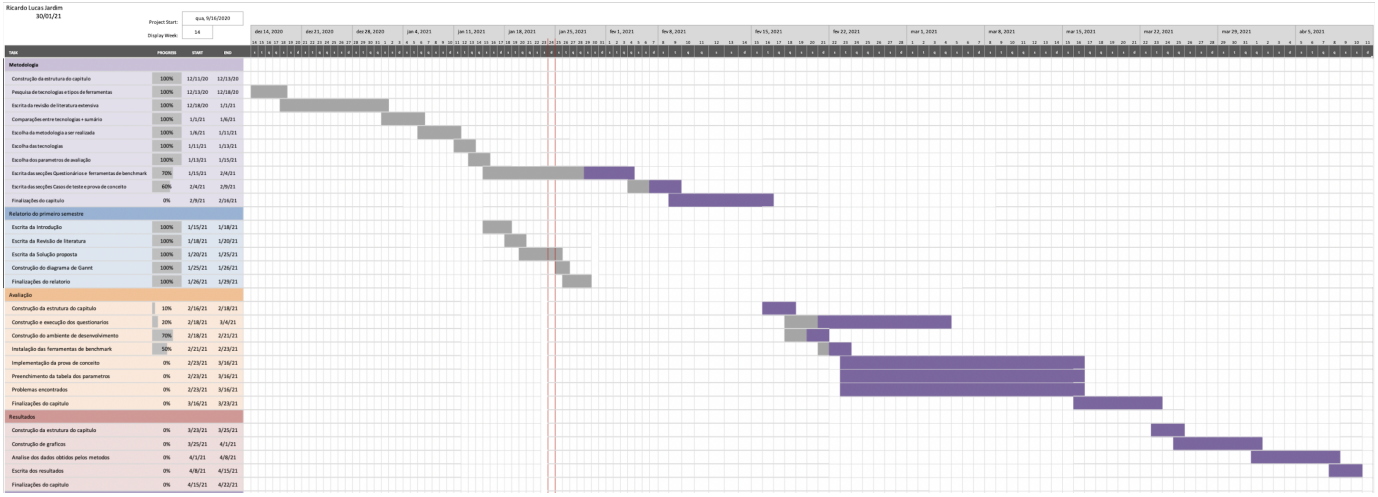

Figura 15. Planeamento de Gannt, parte 2 (14 de Dezembro a 10 de Abril)

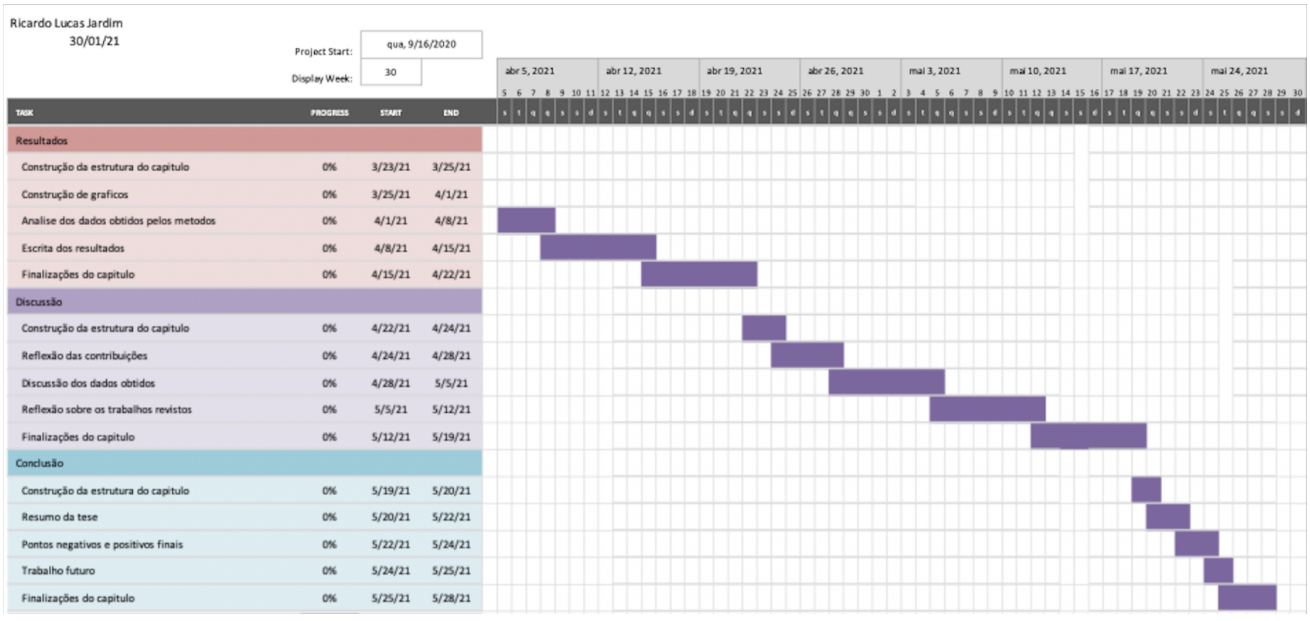

Figura 16. Planeamento de Gannt, parte 3 (5 de Abril a 28 de Maio)

IV. Conclusão

A escolha de uma tecnologia para o desenvolvimento de uma aplicação é muito importante, em função do facto de, todas estas possuírem diferentes comportamentos, estruturas, limitações inerentes à sua arquitectura, linguagens e o mais importante, custo e esforço de implementação em comparação com o desempenho desejado. Esta tese foca-se nestes problemas, e na falta de guias analíticos, não só a nível de implementação mas também de compatibilidade, tornando-se difícil escolher qual é a melhor solução para um problema proposto.

Neste primeiro semestre, já foram finalizado os dois primeiros capítulos, a introdução e revisão de literatura, e grande parte da metodologia, já estando concluída a primeira contribuição (1), sendo a revisão de literatura extensiva. Da mesma forma, já está concluindo a escolha das tecnologias e os parâmetros a avaliar de cada tecnologia, e estando também concluído a escolha dos métodos de avaliação, isto é, quais os questionários a realizar, como serão realizados os casos de testes, quais as ferramentas para benchmarks a utilizar e como será a prova de conceito a ser desenvolvida em cada tecnologia selecionada.

Após estar concluído o terceiro capítulo, a metodologia, iniciarei a realizar os questionários, e de seguida, a construção do ambiente de desenvolvimento para a implementação da prova de conceito e utilização das ferramentas para benchmark. De seguida, começarei no desenvolvimento da prova de conceito anotando problemas encontrados, e por fim, a obtenção dos resultados e discussão.

Agradecimentos

Após cinco anos, a minha jornada académica chega ao fim. Durante estes anos conheci algumas pessoas importantes a que devo agradecer por fazerem parte desta grande jornada, em que sempre me incentivaram a continuar e nunca em desistir.

Começo assim por agradecer aos meus pais e à minha namorada por me apoiarem sempre, confiando sempre que iria chegar ao fim de mais uma etapa da minha vida. Agradeço também a todos os meus amigos que me ajudaram sempre ao longo destes cinco anos, tornando este caminho mais fácil e alegre nas situações mais difíceis.

Por fim, gostaria de agradecer a todos os professores que fizeram parte desta jornada, mas mais importante ainda ao meu orientador Filipe Quintal, por toda a dedicação, paciência, empenho, todas as criticas e pelo conhecimento que me passaram, tornando possível a concretização desta dissertação.

Referências

[1] W. S. El-Kassas, B. A. Abdullah, A. H. Yousef, and A. M. Wahba, "Taxonomy of Cross-Platform Mobile Applications Development Approaches," *Ain Shams Engineering Journal*, vol. 8, no. 2, pp. 163–190, Jun. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2090447915001276

[2] A. Korchi, M. K. Khachouch, Y. Lakhrissi, and A. Moumen, "Classification of existing mobile cross-platform approaches," in *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, Jun. 2020, pp. 1–5.

[3] D. S. dos Santos, H. D. Nunes, H. T. Macedo, and A. C. Neto, "Recommendation System for Cross-Platform Mobile Development Framework," in *Proceedings of the XV Brazilian Symposium on Information Systems*, ser. SBSI'19. New York, NY, USA: Association for Computing Machinery, May 2019, pp. 1–8. [Online]. Available: https://doi.org/10.1145/3330204.3330279

[4] A. Sommer and S. Krusche, "Evaluation of cross-platform frameworks for mobile applications," Mar. 2013.

[5] N. Boushehrinejadmoradi, V. Ganapathy, S. Nagarakatte, and L. Iftode, "Testing cross-platform mobile app development frameworks," in *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering*, ser. ASE '15. Lincoln, Nebraska: IEEE Press, Nov. 2015, pp. 441–451. [Online]. Available: https://doi.org/10.1109/ASE.2015.21

[6] M. Palmieri, I. Singh, and A. Cicchetti, "Comparison of cross-platform mobile development tools," in *2012 16th International Conference on Intelligence in Next Generation Networks*, Oct. 2012, pp. 179–186.

[7] I. Dalmasso, S. K. Datta, C. Bonnet, and N. Nikaein, "Survey, comparison and evaluation of cross platform mobile application development tools," in *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, Jul. 2013, pp. 323–328, iSSN: 2376-6506.

[8] A. Biørn-Hansen, T.-M. Grønli, G. Ghinea, and S. Alouneh, "An Empirical Study of Cross-Platform Mobile Development in Industry," *Wireless Communications and Mobile Computing*, vol. 2019, pp. 1–12, Jan. 2019.

[9] J. Dehlinger and J. Dixon, "Mobile Application Software Engineering : Challenges and Research Directions," 2011. [Online]. Available: /paper/Mobile-Application-Software-Engineering-%3A-and-Dehlinger-Dixon/3ffc4f867e45f8c8e9019726b351172ed77aca4f

[10] A. Biørn-Hansen, C. Rieger, T.-M. Grønli, T. A. Majchrzak, and G. Ghinea, "An empirical investigation of performance overhead in cross-platform mobile development frameworks," *Empirical Software Engineering*, vol. 25, no. 4, pp. 2997–3040, Jul. 2020. [Online]. Available: https://doi.org/10.1007/s10664-020-09827-6

[11] A. Aldayel and K. Alnafjan, "Challenges and Best Practices for Mobile Application Development: Review Paper," in *Proceedings of the International Conference on Compute and Data Analysis*, ser. ICCDA '17. New York, NY, USA: Association for Computing Machinery, May 2017, pp. 41–48. [Online]. Available: https://doi.org/10.1145/3093241.3093245

[12] T. A. Majchrzak, A. Biørn-Hansen, and T.-M. Grønli,

"Progressive Web Apps: the Definite Approach to Cross-Platform Development?" *Hawaii International Conference on System Sciences 2018 (HICSS-51)*, Jan. 2018. [Online]. Available: https://aisel.aisnet.org/hicss-51/st/mobile_app_development/7

[13] L. Delía, N. Galdamez, L. Corbalan, P. Pesado, and P. Thomas, "Approaches to mobile application development: Comparative performance analysis," in *2017 Computing Conference*, Jul. 2017, pp. 652–659.

[14] M. Latif, Y. Lakhrissi, E. H. Nfaoui, and N. Es-Sbai, "Cross platform approach for mobile application development: A survey," 03 2016, pp. 1–5.

[15] D. S. d. Santos, "Sistema de recomendação de frameworks para desenvolvimento multiplataforma em dispositivos móveis," Aug. 2018, accepted: 2019-03-15T12:50:37Z Publisher: Pós-Graduação em Ciência da Computação. [Online]. Available: https://ri.ufs.br/jspui/handle/riufs/10685

[16] H. Charaf, "Developing mobile applications for multiple platforms," in *2011 Second Eastern European Regional Conference on the Engineering of Computer Based Systems*, 2011, pp. 2–2.

[17] A. Biørn-Hansen, T. A. Majchrzak, and T.-M. Grønli, "Progressive Web Apps: The Possible Web-native Unifier for Mobile Development," Jan. 2017, pp. 344–351.

[18] C. P. R. Raj and Seshu Babu Tolety, "A study on approaches to build cross-platform mobile applications and criteria to select appropriate approach," in *2012 Annual IEEE India Conference (INDICON)*, Dec. 2012, pp. 625–629, iSSN: 2325-9418.

[19] S. Tandel and A. Jamadar, "Impact of progressive web apps on web app development," 09 2018.

[20] D. Fortunato and J. Bernardino, "Progressive web apps: An alternative to the native mobile apps," in *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*, 2018, pp. 1–6.

[21] P. Egli, "Software framework for web-based applications," Jun. 2009. [Online]. Available: https://patents.google.com/patent/US7546576B2/en

[22] S. Sendall and W. Kozaczynski, "Model transformation: the heart and soul of model-driven software development," *IEEE Software*, vol. 20, no. 5, pp. 42–45, 2003.

[23] J. Dujmovi'c, "A Method For Evaluation And Selection Of Complex Hardware And Software Systems," in *CMG 96 Proceedings*, 1996, pp. 368–378.

[24] J. A. Krosnick, "Survey research," *Annual Review of Psychology*, vol. 50, no. 1, pp. 537–567, 1999, pMID: 15012463. [Online]. Available: https://doi.org/10.1146/annurev.psych.50.1.537

[25] "A Set of Metrics for the Effort Estimation of Mobile Apps - IEEE Conference Publication." [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7972739

[26] E. Angulo and X. Ferre, "A Case Study on Cross-Platform Development Frameworks for Mobile Applications and UX," in *Proceedings of the XV International Conference on Human Computer Interaction*, ser. Interacción '14. New York, NY, USA: Association for Computing Machinery, Sep. 2014, pp. 1–8. [Online]. Available: https://doi.org/10.1145/2662253.2662280

[27] J. Biolchini, P. Mian, A. Candida, and C. Natali, "Systematic review in software engineering," 01 2005.

[28] C. Rieger and T. A. Majchrzak, "Towards the definitive evaluation framework for cross-platform app development approaches," *Journal of Systems and Software*, vol. 153, pp. 175–199, Jul. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121219300743

[29] H. Heitkötter, S. Heierhoff, and T. A. Majchrzak, "Evaluating Cross-Platform Development Approaches for Mobile Applications," vol. 140, Jan. 2013, pp. 120–138.

[30] S. Xanthopoulos and S. Xinogalos, "A comparative analysis of cross-platform development approaches for mobile applications," in *Proceedings of the 6th Balkan Conference in Informatics*, ser. BCI '13. New York, NY, USA: Association for Computing Machinery, Sep. 2013, pp. 213–220. [Online]. Available: https://doi.org/10.1145/2490257.2490292

[31] T. Freitas Bernardes and M. Miyake, "Cross-platform Mobile Development Approaches: A Systematic Review," *IEEE Latin America Transactions*, vol. 14, pp. 1892–1898, Apr. 2016.

[32] S. Amatya and A. Kurti, "Cross-Platform Mobile Development: Challenges and Opportunities," Jan. 2014, vol. 231, pp. 219–229.

[33] S. Charkaoui, Z. Adraoui, and E. H. Benlahmar, "Cross-platform mobile development approaches," in *2014 Third IEEE International Colloquium in Information Science and Technology (CIST)*, Oct. 2014, pp. 188–191, iSSN: 2327-1884.

[34] A. Ahmad, C. Feng, M. Tao, A. Yousif, and S. Ge, "Challenges of mobile applications development: Initial results," in *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Nov. 2017, pp. 464–469, iSSN: 2327-0594.

[35] T. Dorfer, L. Demetz, and S. Huber, "Impact of mobile cross-platform development on CPU, memory and battery of mobile devices when using common mobile app features," *Procedia Computer Science*, vol. 175, pp. 189–196, Jan. 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050920317099

[36] M. Ciman and O. Gaggi, "An empirical analysis of energy consumption of cross-platform frameworks for mobile development," *Pervasive and Mobile Computing*, vol. 39, pp. 214–230, Aug. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574119216303170

[37] M. Willocx, J. Vossaert, and V. Naessens, "A Quantitative Assessment of Performance in Mobile App Development Tools," in *2015 IEEE International Conference on Mo-*

bile Services, Jun. 2015, pp. 454–461, iSSN: 2329-6453.

[38] ——, "Comparing performance parameters of mobile app development strategies," in *Proceedings of the International Conference on Mobile Software Engineering and Systems*, ser. MOBILESoft '16. New York, NY, USA: Association for Computing Machinery, May 2016, pp. 38–47. [Online]. Available: https://doi.org/10.1145/2897073.2897092

[39] F. Rösler, A. Nitze, and A. Schmietendorf, "Towards a Mobile Application Performance Benchmark," Jul. 2014.

[40] X. Jia, A. Ebone, and Y. Tan, "A performance evaluation of cross-platform mobile application development approaches," in *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems*, ser. MOBILESoft '18. New York, NY, USA: Association for Computing Machinery, May 2018, pp. 92–93. [Online]. Available: https://doi.org/10.1145/3197231.3197252

[41] M. Bekkhus and L. Arvidsson, *Resource utilization and performance : A comparative study on mobile crossplatform tools*, 2020. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:hj:diva-50431

[42] L. Corbalan, J. Fernandez, A. Cuitiño, L. Delia, G. Cáseres, P. Thomas, and P. Pesado, "Development frameworks for mobile devices: a comparative study about energy consumption," in *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems*, ser. MOBILESoft '18. New York, NY, USA: Association for Computing Machinery, May 2018, pp. 191–201. [Online]. Available: https://doi.org/10.1145/3197231.3197242

[43] P. Que, X. Guo, and M. Zhu, "A Comprehensive Comparison between Hybrid and Native App Paradigms," in *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, Dec. 2016, pp. 611–614, iSSN: 2472-7555.

[44] R. Rawassizadeh, "Mobile Application Benchmarking Based on the Resource Usage Monitoring," *IJMCMC*, vol. 1, pp. 64–75, Oct. 2009.

[45] C. Ryan and P. Rossi, "Software, performance and resource utilisation metrics for context-aware mobile applications," in *11th IEEE International Software Metrics Symposium (METRICS'05)*, Sep. 2005, pp. 10 pp.–12, iSSN: 1530-1435.

[46] S. Dhillon and Q. H. Mahmoud, "An evaluation framework for cross-platform mobile application development tools," *Software: Practice and Experience*, vol. 45, no. 10, pp. 1331–1357, 2015, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/spe.2286. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2286

[47] A. Hussain, N. L. Hashim, N. Nordin, and H. M. Tahir, "A METRIC-BASED EVALUATION MODEL FOR APPLICATIONS ON MOBILE PHONES," *Journal of Information and Communication Technology*, vol. 12, pp. 55–71, Apr. 2013. [Online]. Available: http://e-journal.uum.edu.my/index.php/jict/article/view/8137

[48] "What Is a Case Study and What Is It Good for? on JSTOR." [Online]. Available: https://www.jstor.org/stable/4145316?seq=1

[49] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, p. 131, Dec. 2008. [Online]. Available: https://doi.org/10.1007/s10664-008-9102-8

[50] P. Smutný, "Mobile development tools and cross-platform solutions," in *Proceedings of the 13th International Carpathian Control Conference (ICCC)*, May 2012, pp. 653–656.

[51] L. Delia, N. Galdámez, P. Thomas, L. Corbalán, and P. Pesado, "Multi-platform mobile application development analysis," May 2015, pp. 181–186.

[52] S. E. Sim, S. Easterbrook, and R. C. Holt, "Using benchmarking to advance research: a challenge to software engineering," in *25th International Conference on Software Engineering, 2003. Proceedings.*, 2003, pp. 74–83.

[53] "Introdução a atividades | Desenvolvedores Android." [Online]. Available: https://developer.android.com/guide/components/activities/intro-activities?hl=pt-br

[54] "Instruments Overview - Instruments Help." [Online]. Available: https://help.apple.com/instruments/mac/8.0/#/dev7b09c84f5