DOI: 10.33947/2316-7394-v8n1-3898

ESTUDO COMPARATIVO ENTRE O XAMARIN E O CRONAPP NO DESENVOLVIMENTO DE APLICATIVOS NATIVOS E HÍBRIDOS

COMPARATIVE STUDY BETWEEN XAMARIN AND CRONAPP IN NATIVE AND HYBRID APPLICATION DEVELOPMENT

Fabio Fonseca Barbosa Gomes[1], Tiago Silva dos Santos[2]

RESUMO

Este trabalho visa fazer um estudo comparativo entre Xamarin e CronApp, fazendo com que possa criar uma aplicação usando uma única linguagem de programação, C# com o Xamarin e CronApp, ferramenta hibrida possibilitando construções de aplicações, sistemas web, mobile robustos de modo rápido, dinâmico e seguro. Deve-se notar que cada plataforma tem uma linguagem de programação e uma IDE específica. Além disso, é interessante apresentar as funcionalidades das duas ferramentas, que visam aliviar a necessidade de desenvolvimento duplicado para a criação de aplicativos e meios para criar uma aplicação nativa com o uso da ferramenta Xamarin para o desenvolvimento de aplicativos nativos e CronApp para desenvolvimentos híbridos. Nessa perspectiva, há uma tentativa de evitar o retrabalho, o que permitiu o surgimento das ferramentas. O desenvolvimento entre plataformas e, portanto, a Xamarin se destaca por ser capaz de gerar aplicativos para diferentes plataformas a partir da mesma base de código já o CronApp se destaca pelo desenvolvimento sem necessitar de conhecimentos específicos em abrangentes em desenvolvimento.

PALAVRAS-CHAVE: Sistemas de Informação. Software

ABSTRACT

This work aims to do a comparative study between Xamarin and CronApp, making it possible to create an application using a single programming language, C# with Xamarin and CronApp, hybrid tool enabling application constructions, web systems, and robust mobile quickly, dynamically and securely. It should be noted that each platform has a programming language and a specific IDE. In addition, it is interesting to present the functionalities of the two tools, which aim to alleviate the need for duplicate development for the creation of applications and means to create a native application using the Xamarin tool for the development of native applications and CronApp for hybrid developments. From this perspective, there is an attempt to avoid rework, which allowed the emergence of the tools. Cross-platform development and therefore Xamarin stands out for being able to generate applications for different platforms from the same code base, while CronApp stands out for development without requiring specific knowledge in developing comprehensives.

KEYWORDS: *Information System. Software*

1 [1] Centro Universitário UNIRB, Centro Universitário Dom Pedro II e IFBA.
2 [2] Centro Universitário UNIRB.

1. INTRODUÇÃO

Atualmente, existe uma gama de inovações para *smartphones*, o mercado demanda uma tendência para o desenvolvimento de aplicativos móveis, na qual cresce consideravelmente com a necessidade das pessoas se conectarem ao mundo digital (KRONBAUER *et al.,* 2015). Pensando neste contexto, os dispositivos móveis estão cada vez mais sofisticados e suas funções não são apenas para receber e efetuar chamadas. Eles visam agregar valor aos usuários, promover marcas, assim como, romper fronteiras e aproximar as pessoas, seja no trabalho, entre amigos ou na sociedade.

No mercado de dispositivos móveis os sistemas operacionais mais utilizados são o iOS, da Apple e o Android, da Google. Anteriormente existia o Windows Phone, sendo desativado em 2017 (CHOHFI, 2019). Cada um deles trabalham com uma linguagem de programação e um Ambiente de Desenvolvimento Integrado (*Integrated Development Environment* – IDE) específica. Com isso, o desenvolvimento de um programa utilizando uma única linguagem de programação para esses sistemas operacionais torna-se mais difícil. Isto ocorre porque a linguagem nativa do Android é em Java, enquanto para o iOS utiliza-se *Objective-C* ou *Swift*. Portanto, o trabalho de construção de um aplicativo para os dois ambientes é dobrado, e isso impacta diretamente em prazos e custos.

Para solucionar este problema, o Xamarin (plataforma de código aberto) foi desenvolvido para uma possibilidade inovadora. Ele tem a capacidade de criar uma aplicação nativa com um único algoritmo para as plataformas: Android e iOS. Desta forma, com a mesma tela e único código pode-se conseguir gerar para mesma aplicação para os dois sistemas operacionais.

A intenção ao utilizar o Xamarin é que os problemas serão atenuados, com o desenvolvimento da multiplataforma, sendo possível o desenvolvimento de uma aplicação eficiente e eficaz. Apesar das grandes vantagens apresentadas pelo Xamarin, este artigo também irá abordar os principais conceitos do CronApp que é o desenvolvimento utilizando a solução *low code* (desenvolvimento de forma RAD) e *high code* (desenvolvimento via código) essas duas soluções são independentes de tecnologias proprietárias de terceiros, tais como: servidores aplicações ou interpretadores (ARAÚJO, 2019).

Com o CronApp o desenvolvedor consegue gerar todo ciclo de vida do projeto, todos projetos desenvolvidos são compilados em nuvem, assim podendo desenvolver apenas pelo navegador bastando apenas ter acesso a internet, possuindo um ambiente pronto sem precisar nenhuma implementação no ambiente já configurado e pronto, incluindo o modo de Plataforma como Serviço (PaaS) para publicação dos projetos. A CronApp possui arquitetura MVC (*Model-View-Controller*) tanto na camada cliente quanto na camada servidor, ou seja, é possível qualquer desenvolvimento feito como modo de microserviços, possibilitando criar sistemas altamente integráveis (CRONAPP, 2019).

2. SISTEMAS OPERACIONAIS MÓVEIS

Com a evolução da tecnologia da informação e comunicações, os dispositivos móveis *smartphones, tablets* começaram a se tornar populares, houve a necessidade de aprimorar ou criar sistemas operacionais móveis. Destacam-se, entretanto, os sistemas operacionais móveis mais utilizados do mundo como: Android e IOS. Para HOLWERDA (2013), além dos sistemas operacionais móveis conhecidos, cada dispositivo móvel existente possui um sistema operacional invisível, considerado como um segundo sistema operacional que funciona de forma completamente silenciosa. Ele é conhecido como *Real Time Operating Systems* - Sistemas Operacionais de Tempo Real (RTOS) e será mais detalhado em uma das subseções posteriores.

2.1 ANDROID

O sistema Android, da Google encontra-se presente em grande parte dos dispositivos móveis atualmente. De acordo com IDC (2019) o sistema Android (figura 1) possui uma fatia de 85% do mercado, no total mais de 500 milhões de aparelhos e mais de 600 dispositivos diferentes que usam o sistema operacional. Ele foi criado no ano de 2005 por Andy Rubin, da empresa Android Inc. Foi desenvolvido a partir de um *kernel* com Linux, por conseguinte, é um sistema operacional de código aberto. A empresa Android Inc. foi comprada pela Google nesse mesmo ano e Rubin passou a ser o Diretor de Plataformas Móveis da Google (JACKSON, 2011).

Apesar do sistema ter surgido em 2005, ele passou a ter notoriedade no mercado apenas a partir de 2009. Segundo REED (2010), apenas 7% dos usuários americanos de smartphones utilizavam-no.

Porém, atualmente, esse sistema operacional é utilizado pela grande maioria dos usuários. Isso é reflexo do sistema de código aberto, o que determinou que diversos outros fabricantes (Samsung, Sony, LG, Nokia) passassem a desenvolver novos tipos de *smartphones* compatíveis com ele.

2.2 iOS

O iOS (*iPhone Operational System* – Sistema Operacional do iPhone) é um sistema operacional móvel criado pela Apple baseado em seus antigos modelos Macintosh. Ele é bastante conhecido pela sua Segurança, Estabilidade e Confiança (SEC). Isso significa que o sistema deve garantir ao seu usuário o conforto no uso de seu dispositivo móvel (APPLE, 2019). O sistema surgiu em 2007, mas sem suporte à conexão com função copiar e colar, mensagens multimídia e aplicações já instaladas de fábrica.

Apesar de ter seus concorrentes bastante ativos no mercado, a Apple manteve o foco na usabilidade. Fazendo com que faça interação entre usuário e aplicação usada. Atualmente, o iOS está com uma visibilidade na interface com um ar de interação de fácil entendimento, simples comunicação, simples de usar e de fácil acesso as configurações. Adotam um visual limpo utilizando poucas cores, que fazem combinações e sejam de fácil entendimento também com que fique fácil de identificar as configurações para que possam fazer suas alterações desejáveis (APPLE, 2019).

Desta forma, devido as rápidas mudanças e previamente as necessidades de atualizações por diversas formas de uso, o iOS encontra-se na versão doze. Versão que aposta na experiência do usuário totalmente responsiva tanto no iphone quanto no iPad. Ações nativas como abrir uma câmera e digitar no teclado acontecem mais rápido depois da atualização. E na hora que o usuário faz diversas coisas ao mesmo tempo, as melhorias são muito mais percebidas. Essas melhorias aumentam o desempenho de todos os aparelhos compatíveis (APPLE, 2019).

A privacidade também é um quesito muito importante para atrair os usuários da Apple, ela foi pensada de forma que garante a segurança de seus dados de forma sigilosa. Apostando no direito humano fundamental, afirma que a privacidade é um direito humano fundamental. A novidade da atualização para a atualização 12 possui uma novidade com foco no compartilhamento de notícias, matérias, evita que anunciantes coletem as características únicas dos seus dispositivos móveis (APPLE, 2019).

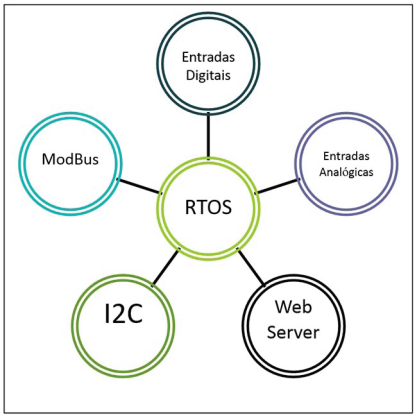
2.3 RTOS

O RTOS é um sistema operacional invisível, conhecido como um sistema multitarefa embarcado, que funciona em segundo plano nos dispositivos móveis e está instalado no *firmware* do processador, sendo desenvolvido pelo próprio fabricante da unidade central de processamento. É possível que um usuário possa desenvolver soluções de software livre para este sistema, porém as empresas de que montam os dispositivos móveis, têm preferência por adquirir os RTOS proprietários, pois eles já estão com todas as funcionalidades prontas (HOLWERDA, 2013).

Este sistema é considerado como multitarefa e funciona em vários tipos de dispositivos móveis, tais como atuadores, *smartphones, tablets*, dentre outros. A principal vantagem deste tipo de sistema operacional é que ele é muito flexível e consegue se comunicar com os mais diversos serviços de dados praticamente ao mesmo tempo. “As aplicações práticas para um RTOS são normalmente quando precisamos efetuar muitas tarefas ao mesmo tempo ou trabalhar com “Real-Time” onde falhas além do tempo definido é crítico” (MORAIS, 2018).

Como pode ser visualizado através da figura 1, o Sistema Operacional de Tempo Real deve se preocupar com a comunicação simultânea com várias funcionalidades diferentes, tanto como comunicação com o servidor, questões de entradas de dados analógicas ou digitais, além de protocolos de dados como o Modbus e o I2C

Figura 1: Exemplo de ambiente multitarefa do RTOS



Fonte: MORAIS (2018)

3. APLICATIVOS NATIVOS E HÍBRIDOS

As aplicações nativas (desenvolvidas para apenas um tipo sistema operacional) vêm com uma série de sensores, esses sensores possuem algumas funcionalidades como, GPS, giroscópio, acelerômetro e luminosidades. Os recursos nativos como, agenda, calendário, câmera, reconhecimentos de gestos na tela. É importante lembrar, que dependendo da funcionalidade da aplicação pode funcionar sem necessidade da internet.

O primeiro requisito de uma aplicação nativa, o usuário vai sentir-se familiar usando a aplicação. Levando em consideração o custo de desenvolvimento para aplicações nativas é muito maior, sendo necessário, equipes de conhecimentos específicos para cada tipo de plataforma. Em relação a capacidade de processamento das aplicações nativas, elas se mostram mais rápidas e eficientes do que uma aplicação não nativa. Visto que ela foi desenvolvida para aquele sistema operacional em que ela está executando, o que se torna extremamente vantajoso para quem deseja ter uma aplicação rápida em seu dispositivo móvel.

Consequentemente baseia-se nesses respectivos itens: Performance, Desenvolvimento, Interface, Recursos Disponíveis e Segurança:

- **Performance** – São desenvolvidas em códigos específicos para cada plataforma, o que traz benefício ao nível de performance.
- **Desenvolvimento** – Como o código tem de ser desenvolvido de raiz para cada plataforma, as aplicações nativas exigem maior investimento em desenvolvimento, tanto de tempo como financeiramente.
- **Interface** – O leque de recursos para aplicações nativas é, em geral, específico para cada plataforma, dificultando a sua integração para apps multiplataforma.
- **Segurança** – Mais seguro.

Os usuários de cada sistema operacional esperam uma forma de costume para usar as aplicações. Mão-de-obra cara é necessário e será preciso várias equipes de desenvolvimento. Ele fica mais trabalhoso e demorado, pela necessidade de criar um aplicativo para cada plataforma do mercado (ARAÚJO, 2019).

Os aplicativos *webs* podem ser baseados em HTML5 e exibidos através de um navegador embutido na aplicação, tendo parte ou total em seu conteúdo carregado pela internet. Eles são populares devido as suas características multiplataforma, isto é, se desenvolve apenas uma vez e executa em diferentes plataformas, permitindo redução dos custos de produção.

As aplicações hibridas são baseadas em uma solução genérica, que é usada em diversas plataformas. Atentam-se também em: *Performance*, Desenvolvimento, Interface, Recursos disponíveis, *Plug-ins* e Segurança:

- **Performance -** Aplicações híbridas aplicam camadas de tecnologia, como *web views* e outros plugins, nos APIs nativos, o que tem impacto na performance em comparação com aplicações nativas.
- **Desenvolvimento –** tendo por base tecnologias web familiares como HTML5, CSS e JavaScript, as aplicações híbridas são mais eficazes em termos de custo e têm a vantagem adicional de reutilizarem o mesmo código para diversas plataformas.
- **Interface –** São diversos recursos dirigidos a aplicações híbridas, como a vantagem de terem efeito em todas as plataformas. No entanto, aplicações híbridas estão também dependentes em grande parte de *plug-ins* para funcionarem ao nível de aplicações nativas.
- ***Plug-ins* –** Possui a vantagem de terem efeitos em todas as plataformas de desenvolvimento. Porém, aplicações híbridas estão também dependentes em grande parte de *plug-ins* para ter um ótimo funcionamento, da mesma forma de aplicações nativas.
- **Segurança –** Recorrem a *plug-ins* externos para terem como garantia maior segurança e utilizam código comum a tecnologias web, logo, são mais vulneráveis.

Para o desenvolvendo web para criação de aplicativos (*app*), existe o *app* híbrido que pode baixar pela *Play Store*. Comporta-se com aplicativo, porém no navegador web, responde como aplicativos. Existem *frameworks* que fornecem vários recursos para diversos dispositivos (HOLWERDA, 2013). O desenvolvimento pode ficar mais barato utilizando o seu desenvolvimento, por exemplo. O aplicativo pode abranger todas as plataformas no mercado, sem a necessidade de um retrabalho com alto custo. Um dos casos mais conhecidos é o recurso de *Touch*

Bluetooth, utilizado em *smartwatchs*. Nele, o usuário consegue realizar a comunicação entre o smartwatch e o *smartphone* através da tecnologia de *bluetooth* gerada a partir de uma aplicação híbrida instalada no *smartwatch* (SMITH, 2018).

3.1 VANTAGENS E DESVANTAGENS DOS TIPOS DE APLICATIVOS MÓVEIS

Aplicativos móveis nativos são aqueles desenvolvidos utilizando-se os componentes gráficos de interface de usuário nativos do ambiente operacional do *smartphone,* tipicamente Android e iOS. A alternativa aos aplicativos nativos são os chamados aplicativos de híbridos, que são construídos utilizando a base *Web*. Naturalmente, os aplicativos nativos oferecem uma experiência mais simples, por serem integrados ao ambiente operacional e oferecerem uma interface mais fluída e desempenho superior. Nesse sentido o Quadro 1 destaca os principais tipos de aplicativos móveis.

Quadro 1: Tipos de Aplicativos Móveis

TIPO DE APLICAÇÃO	LINGUAGEM, BIBLIOTECAS, *FRAMEWORK*, SDK OU IDE
Web	HTML5 JQuery, CSS3, Javascript, JSON, NOTEPAD ++
Nativa	ObjetiveC, xCode, iOS Simulator
Híbrida	ObjetiveC, xCode, iOS Simulator, Cordova, HTML5, JQuery, CSS3, Javascript, JSON

Fonte: (Autoria própria, 2019)

Pode-se verificar, no Quadro 1, que os três tipos de aplicações são de extrema importância para o desenvolvimento de aplicações móveis, sendo que a aplicação *Web* é baseada na própria *web com a* utilizando das linguagens nativas, ou seja, resumidamente para desenvolvimento, o que fará é codificar em HTML, CSS3, *Javascript*, JQuery e *JSON*, o que fazem a comunicação entre eles para poder ser executada, também sendo adicionado os arquivos à um diretório local, semelhante a um site local.

A Nativa possui uma arquitetura diferente utilizando *ObjectiveC* como base, ela é utilizada como principal linguagem de programação para *MacOS X* e *IOS*, tendo também o *JSON* para fácil leitura e escrita, de fácil interpretação.

A Híbrida neste caso adota uma linha consistente no quesito mistura de linguagens com simulação voltada para iOS, podendo aplicar o desenvolvimento nativo de forma de hibrida com ajuda de *frameworks* baseado na *Web*. Observando o Quadro 2, percebe-se quais são as principais vantagens e desvantagens dos tipos de aplicações móveis.

Quadro 2: Tipos de aplicação e tecnologia utilizadas

TIPO DE LINGUAGEM	VANTAGENS	DESVANTAGENS
Webapps	- Executados pelo navegador, independentemente da plataforma; - Atualização e distribuição rápida, sem necessidade de *downloads* ou atualizações; - Acesso rápido e fácil pelo *smartphone*	- Pouca integração com o *hardware* do dispositivo; - Lento; - Menos funcionalidades; - Velocidade de execução depende da internet
Nativo	- Integração com o usuário mais rica em recursos - Velocidade de execução não depende da internet	- Cada plataforma terá sua versão do programa; - Distribuição e a atualização dependente de lojas *on-line*.
Híbrido	- Compartilha o código com outras plataformas; - Pode ser distribuído em lojas *on-line;* - Pode usar recursos de plataforma com código nativo.	Desempenho e *design* limitados

Fonte: (Autoria própria, 2019)

Com base no Quadro 2, realmente é executado pelo navegador, o que mais se destaca é que ele se comporta como um aplicativo, levando em conta a responsividade. A desvantagem principal é não ocupar um espaço na memória do aparelho, pois não necessita de *download*, conta apenas com o funcionamento, caso esteja conectado à internet.

A aplicação Nativa, geralmente, é a mais conhecida, podendo ser encontrada nas lojas *online* para *download* (Google Play Store e App Store). É escrito por uma linguagem específica de cada sistema ope-

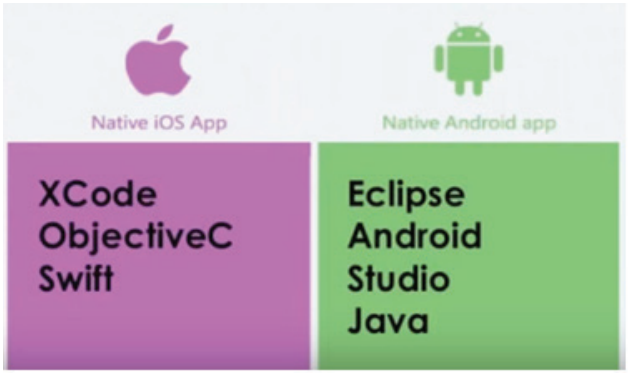
racional, podem ser acessados todos os dispositivos dos celulares (câmeras, GPS dentre outros). Funcionam *offline* (sem internet). Tem segurança garantida não possuindo um bom custo para desenvolvimento.

Já o híbrido baseia-se como se fosse um remédio genérico comparado ao remédio original, é feito à base de *Frameworks* e garante uma perfeita comunicação. É uma junção de aplicativos nativos com os *web apps* e isso significa que o desenvolvedor utiliza o mesmo código para executar em aparelhos com Android ou iOS.

4 ESTUDO COMPARATIVO ENTRE O XAMARIN E O CRONAPP NO DESENVOLVIMENTO DE APLICATIVOS NATIVOS E HÍBRIDOS

Nesta seção, serão abordadas a ferramenta Xamarin e será feita uma análise sobre ela. Desta forma, será possível identificar qual das duas soluções é a mais vantajosa e eficiente. O Xamarin é uma plataforma de código aberto que significa dizer, para diferentes fins e, utilizando Mono (Implementação Código aberto e do NetFramework). Isto é diferente de soluções feitas exclusivamente para iOS ou para Android, em que um programador precisaria desenvolver a mesma solução de duas formas diferentes para cada sistema operacional, conforme pode ser visto na figura 2.

Figura 2: Linguagens de programação usadas no iOS e Androix



Fonte: CHOHFI (2019)

Apenas com conhecimento em C#, é possível realizar o desenvolvimento de aplicações para dispositivos móveis para os dois sistemas operacionais móveis. O reaproveitamento do código chega em torno de 75% a 100% do código do aplicativo, assim o desenvolvimento torna-se a interface nativa dos Sistemas Operacionais.

O foco principal no desenvolvimento de aplicações móveis com o Xamarin é diminuir a complexidade de desenvolvimento diretamente em interface de usuário para cada um dos Sistemas Operacionais, a ferramenta tem um framework, que se chama XamarinForms, faz com que as diferenças fiquem separadas e proporciona uma biblioteca comum. Assim, caso o aplicativo torne-se estreito aos recursos de interface (interface limitadas) pelo XamarinForms, assim vem o reaproveitamento e a totalidade do código, sabendo expor uma única versão para os dois ambientes de Sistemas Operacionais (ARAÚJO, 2019).

Xamarin é uma empresa fundada por duas pessoas que participavam de projetos *open source* (código aberto), com a tecnologia MONO que é a possibilidade de executar o C# em sistemas operacionais como Linux. Com isso, eles já trabalhavam na interoperabilidade na questão multiplataforma e então desenvolveram a Xamarin para que com a linguagem C# o desenvolvedor consiga desenvolver um aplicativo para Android e iOS.

A utilização do C# serviu como solução para o mercado de desenvolvimento multiplataforma. O diferencial do Xamarin é que o resultado gerado a partir do código do desenvolvedor não é um aplicativo híbrido, mas um ou mais aplicativos nativos gerados a partir da mesma estrutura de código. Tem-se, então, uma única linguagem responsável pela linguagem responsável pela lógica de programação, por exemplo: se precisar consultar um *web-service*, não precisa mais aprender Java ou Objective-c, poderá aprender C# e as bibliotecas padrões. Com isso, o desenvolvedor conseguirá consultar um *web-service* “apenas com os conhecimentos de C#” e fazer com que seu aplicativo comece a executar.

Por isso, foi necessário traduzir as bibliotecas de cada sistema operacional, então quando necessário acessar uma lista ou acessar a câmera de um dispositivo em cada uma dessas plataformas é diferente de como acessar a cada uma dessas plataformas pois eles tiveram que fazer uma tradução de tudo aquilo que era nativo (DA SILVA, 2017).

4.1.1 XAMARIN FORMS E XAMARIN STUDIO

O Xamarin *forms* é um *framework* onde consegue construir toda interface gráfica e automati-

camente vai traduzir essa interface para cada uma dessas plataformas. Então não tem só a linguagem, aquilo que faz a lógica do aplicativo unificada no C#, porém possui a interface unificada no Xamarin *forms.* Caso o desenvolvedor faça a criação de um botão, ele irá surgir com a interface gráfica do Android e assim para a outra plataforma. Desta maneira, uma série de componentes com características visuais de cada plataforma. Tudo isso é de certa forma configurável com o Xamarin (ARAÚJO, 2019). O Xamarin Studio faz parte do ambiente de desenvolvimento criada pela Xamarin para o desenvolvimento de aplicativos. Veio com objetivo de facilitar o processo de desenvolvimento facilitando o contato entre desenvolvedor e máquina.

4.1.2 COMPARATIVO ENTRE XAMARIN E UMA FERRAMENTA DE DESENVOLVIMENTO HÍBRIDO.

Muitos autores discutem a respeito do desenvolvimento de aplicativos feitos de forma ágil e eficaz. Em relação ao Xamarin e o CronApp, percebe-se que as duas ferramentas de desenvolvimento são de extrema importância para os desenvolvedores. Os dois dividem muito a opiniões de desenvolvedores para que possa adequar a ferramenta com a sua necessidade referente a aplicação. O Xamarin foi criado como solução para desenvolvimento de aplicativos móveis. Visa amenizar essa necessidade de desenvolvimento em duplicidade, permitindo o desenvolvimento para Android e IOS. A CronApp é uma ferramenta para desenvolvedores que possibilita construção de aplicações, sistemas *web* e *mobile* robustos de modo rápido, construída em camadas, tendo toda evolução até chegar na maturidade de codificação avançada, após isso, foi inserido a programação visual, evoluindo então para uma ferramenta *Low code.*

O ambiente existente na nuvem, possui todas as funcionalidades prontas, permitindo que o usuário consiga criar um projeto com poucos cliques e sem dificuldade. A CronApp tem arquitetura MVC tanto na parte cliente quanto na camada de servidor, também é adotado dois estilos de desenvolvimento *Low code* que é de baixa codificação e possui mais simplicidade, menos liberdade e alta produtividade. É uma ferramenta de desenvolvimento Híbrido que atende as duas formas de desenvolvimento. O Quadro 3 apresenta uma comparação entre as duas ferramentas.

Quadro 3: Comparativo entre Xamarin e CronApp

CRITÉRIO	XAMARIN	CRONAPP
Performance Nativa	- Rápido para executar em IPhone 4 e Androids antigos	- Bom para o desenvolvimento de aplicativos atuais
Performance Híbrida		
Característica Nativa	- Utiliza C# para desenvolvimento	- *Back End* em Java, baseado em nuvem; - Vários *frameworks* para implementação
Característica Híbrida		
Data Amount	- Exibe qualquer quantidade de dados	
Launch Time	- Lançamento em alta velocidade	

Fonte: (Autoria Própria, 2019)

Conforme pode ser visto no Quadro 3, no quesito *performance,* o Xamarin não funciona de forma eficiente, ou seja, ele passa toda essa eficiência para os celulares mais antigos, já o CronApp acaba tendo a possibilidade de inspecionar pelo *browser* o que está funcionando ou não, também se tornando mais útil para aplicações mais recentes.

A característica nativa da Xamarin aposta que a solução de tudo no quesito desenvolvimento aposta no C# com ajuda do Xamarin Forms para o visual. O CronApp tem toda sua programação *back end* em Java, sendo realizada na nuvem, possuindo a tecnologia *blocky* com vários *frameworks* para integração no desenvolvimento.

O *Data amount* deixa entender que tanto para o Xamarin quanto para o CronApp, tem a possibilidade de consulta de dados ou transmissão no qual não podem ser contabilizados. Conclui-se que o Xamarin é mais vantajoso nesses quesitos porque oferece uma performance para celulares atualizados, na característica nativa o Xamarin ganha, porém, a atualidade referente a celulares torna-se mais vantajoso perante a atualidade.

4.1.3 GENYMOTION

Para saber como está o processo de desenvolvimento de software é extremamente necessário para um sistema de computação equipado para emular um outro sistema operacional, assim dentre

vários tipos de emuladores disponíveis e gratuitos para uso e testes da aplicação. Tem um muito usado que é o Genymotion.

O Genymotion é um emulador de sistema operacional Android que vem ganhando espaço no mercado devido ao desempenho e sua extensa variedade de versões de Sistemas Operacionais disponíveis por ele. É possível usar um *smartphone* com Android simulado em um computador comum. Para fazer isso é necessário ter um programa virtualizador como o VirtualBox, por exemplo.

5. CONSIDERAÇÕES FINAIS

Conforme pode-se ver neste artigo, o Xamarin apresenta-se como uma solução de baixo custo que promete otimizar o tempo do programador no desenvolvimento de novas soluções. Hoje, esta solução se apresenta como uma enorme vantagem que é a facilidade no compartilhamento dos códigos-fonte para soluções em Android ou iOS.

O Xamarin permite a execução de linguagem de programação C# e, como ferramenta de desenvolvimento, tem a visão de diminuir a quantidade de desenvolvimento dividido. O diferencial do Xamarin é que o resultado gerado a partir do código do desenvolvedor é uma aplicação nativa. Sendo que a equipe de desenvolvimento utilizará apenas uma linguagem de programação por projeto, precisará escrever apenas uma vez o aplicativo para atingir diversas plataformas.

O CronApp é uma ferramenta de desenvolvimento *Low code* (desenvolvimento orientado a componentes), tendo toda evolução até chegar no modo desenvolvimento de codificação. Oferece uma forma de desenvolvimento para vários níveis de desenvolvedores, oferecendo até mesmo para o desenvolvedor sem experiência em programação, utilizando a forma de desenvolvimento RAD, unindo o desenvolvimento de aplicações híbridas.

Para trabalhos futuros, pretende-se realizar mais testes com o desenvolvimento de um sistema em cada uma das ferramentas e depois, fazer um teste comparativo com este sistema, sendo testado em condições iguais, para se verificar por qual das ferramentas desenvolvidas ele se mostrou mais eficaz e eficiente.

REFERENCIAS

ANDROID. (2019). **Android.** Disponível em: https://www.android.com/. Acesso em: 16 jul. 2019.

APPLE. (2019). **IOS. iOS 13.** Disponível em: https://www.apple.com/br/ios/ios-12. Acesso em: 25 nov. 2019.

ARAÚJO, E. C. (2019). **Xamarin Forms e MVVM**. Casa do Código.

CHOHFI, A. (2019). **Desenvolvimento Multiplataforma com Xamarin e MVVM, da arquitetura UI específica, Microsoft.**

CRONAPP. **Cronapp.** Disponível em: https://www.cronapp.io/. Acesso em: 16 Jul 2019.

DA SILVA, L.L.B. (2015). **Desenvolvimento de Aplicações para Dispositivos Móveis:** tipos e exemplo de aplicação na plataforma iOS II Workshop de Iniciação Científica em Sistemas de Informação, Goiânia.

HOLWERDA, T. (2013). **The second operating system hiding in every mobile phone.** Disponível em: https://www.osnews.com/story/27416/the-second-operating-system-hiding-in-every-mobile-phone/. Acesso em: 25 nov. 2019.

IDC. (2019). **Smartphone Market Share** – *OS Data Overview*. Disponível https://www.idc.com/promo/smartphone-market-share/os. Acesso em: 16 jul. 2019.

JACKSON, W. **Android Apps for Absolute Beggin-ers.** Apress. Nova York, 2011.

KRONBAUER, A.; GOMES, F.F.B.; ARAÚJO, B. (2015) Remote Home: A Universal Control for Residential Environments. In: **Proceedings of the 21st Brazilian Symposium on Multimedia and the Web, Manaus.**

MACHADO, H. **Desenvolvimento multiplataforma com Xamarin. Devmedia [site]**, s. d. Disponível em: http://www.devmedia.com.br/desenvolvimento-multiplataforma-com- xamarin/33467. Acesso em: 25 ago. 2017.

MORAIS, J. (2018). **RTOS:** um ambiente multi-tarefas para sistemas embarcados. Disponível em: https://www.embarcados.com.br/rtos-sistema-operacional-de-tempo-real/. Acesso em: 25 nov. 2019.

REED, B. **A Brief History of Smartphones** - *How the smartphone went from a high-end enterprise device to an everyday consumer staple.* Disponível em: http://www.pcworld.com/article/199243/a_brief_history_of_smartphones.html. Acesso em: 16 jul. 2019.

SMITH, D. H. (2018). Wireless Device Connection Management. U.S. **Patent Application**, n. 15/612,749, 6 dez. 2018.