Aplicações Mobile Híbridas: Um Estudo de Caso do Framework Ionic para Construção de um Diário de Classe

Kamile A. Wahlbrinck[1], Bruno B. Boniati[2]

[1]Universidade Reginal Integrada do Alto Uruguai e das Missões (URI)
Caixa Postal 184 – 98.400-000 – Frederico Westphalen – RS

[2]Instituto Federal Farroupilha (IFFar)
Caixa Postal 169 – 98.400-000 – Frederico Westphalen – RS

{kamilewahlbrinck,brunoboniati}@gmail.com

Abstract*. Mobile apps are part of the daily lives of people, to check email, social networks, work agenda, among others. With increasing demand for mobile applications, a problem arises: porting these applications to the most varied platforms that exist, such as Android, iOS, Windows Phone. A solution for this problem are the hybrid applications, which are developed in a same language and ported to various platforms. The objective of this work is to develop a hybrid mobile application to help teachers in frequency control in the classroom, using the Ionic framework, analyzing their strengths and weaknesses.*

Resumo. *Aplicações móveis atualmente são parte do cotidiano das pessoas, para conferir e-mail, redes sociais, agenda de trabalho, entre outros. Com o aumento da demanda de aplicações móveis, surge um problema: portar essas aplicações para as mais variadas plataformas, como Android, iOS, Windows Phone. Uma solução para tal problema são as aplicações híbridas, que são desenvolvidas em uma mesma linguagem e portadas para diversas plataformas. Por meio deste trabalho objetiva-se desenvolver uma aplicação mobile híbrida para auxiliar docentes no controle de frequência em sala de aula, utilizando o framework Ionic, a fim de analisar os pontos positivos e negativos das aplicações híbridas bem como o referido framework.*

1. Introdução

Vivemos em uma sociedade rodeada de equipamentos tecnológicos. A adesão aos dispositivos móveis é evidente em nosso cotidiano, como aparelhos pessoais, de trabalho e lazer. O mercado de dispositivos móveis oferece atualmente vários sistemas operacionais (SO).

Os dois sistemas operacionais mais populares são *Android* e *iOS*, que compõem cerca de 96% do mercado, porém eles tem grandes diferenças quanto ao desenvolvimento de suas aplicações. Os 4% restantes são SO menos populares, como *Windows Phone* e *Black Barry* [Teleco 2016].

Para desenvolver aplicações para dispositivos móveis é necessário uma série de conhecimentos específicos a respeito de cada SO a fim de atingir o máximo de plataformas existentes. Pensando em suprir este mercado diversificado, empresas de *software* precisam de funcionários com competências em cada um dos diferentes sistemas operacionais, ou seja, que conheçam diversas linguagens de programação e particularidades das plataformas. Isto leva o desenvolvimento de aplicações móveis a se tornar caro e lento, com manutenções dispendiosas.

Observa-se no entanto, que não há um padrão de mercado em relação a plataforma de desenvolvimento a ser adotada, tornando o *software* desenvolvido incompatível para ser executado em uma plataforma diferente da qual foi projetado [Wahlbrinck 2015]. Diante de tal realidade, programadores procuram uma forma fácil e rápida para atender as necessidades do maior número possível de plataformas, pois desenvolver uma aplicação para cada dispositivo é uma tarefa economicamente desgastante. A solução é recorrer a *frameworks* de desenvolvimento multiplataforma, que a partir de um único código tornam possível que sua aplicação funcione em mais de uma plataforma.

Como essas tecnologias estão cada vez mais presentes no dia a dia das pessoas, é difícil imaginar uma sociedade sem o auxilio destes aparelhos, que contém diversos recursos, informações e funcionalidades. Sendo assim, não poderiam ficar de fora dos ambientes escolares, onde podem auxiliar no processo de ensino-aprendizagem, quando utilizados com objetivos específicos e bem definidos.

O presente trabalho teve como objetivo desenvolver uma aplicação móvel para realizar o controle de presença em sala de aula, explorando a programação multiplataforma, para simplificar o desenvolvimento da aplicação, visto que foi desenvolvida utilizando linguagens *web*. Para o desenvolvimento da parte lógica da aplicação foi utilizado o *framework* Ionic, já para compilar a aplicação para que seja portada para uma plataforma móvel foi utilizado o Cordova. Ambas as ferramentas foram escolhidas pois tem suporte a plataforma *Android*, são *open source* e utilizam tecnologias *web* para desenvolvimento do projeto.

A próxima seção (2) descreve os diferentes tipos de aplicações móveis e suas principais diferenças. Na sequência, a seção 3 demonstra como foi desenvolvida a aplicação bem como as ferramentas utilizadas no processo de desenvolvimento. Por fim, as consideração finais e indicações de trabalhos futuros.

2. Desenvolvimento Mobile

O desenvolvimento de aplicações para dispositivos móveis pode ser categorizado em três opções diferentes: desenvolvimento de aplicações nativas, que são instaladas no dispositivo, desenvolvimento de aplicações *web*, que rodam no navegador do aparelho e desenvolvimento de aplicações híbridas, que são uma mistura das duas anteriores, ou seja, serão instaladas nos dispositivos, porém desenvolvidas para rodar em um navegador *web*. Nas subseções seguintes cada opção será descrita com mais detalhes.

2.1. Aplicações Nativas

Aplicativos nativos representam a forma mais tradicional e comum de programar para dispositivos móveis. Desenvolver uma aplicação nativa exige a escolha de um sistema operacional alvo, determinando como consequência os dispositivos em que a aplicação poderá ser instalada. São adquiridos através de lojas ou portais eletrônicos. Aplicativos para *Android*, por exemplo, podem ser instalados a partir da *Google Play*, enquanto aplicativos para iOS podem ser obtidos na *Play Store*. Eles podem ser acessados através de ícones na tela do dispositivo e possuem acesso a recursos nativos do aparelho, como GPS (*Global Positioning System*) e câmera [da Silva et al. 2015].

Por ser desenvolvido para uma plataforma específica esses aplicativos apresentam uma performance maior que aplicativos *web*, porém o código da aplicação não pode ser reaproveitado a fim de portá-la para outras plataformas, tornando seu desenvolvimento mais custoso.

2.2. Aplicações Web

Aplicativos *Web*, na verdade, não são aplicativos, são sites responsivos que conseguem se adaptar aos diversos tamanhos de tela nos dispositivos *mobile*, o que faz com que sejam visualmente semelhantes a aplicativos nativos. A tarefa de desenvolvimento deste tipo de aplicação exige do desenvolvedor uma série de cuidados relacionados à escolha de determinadas funcionalidades bem como aspectos relacionados à apresentação da mesma. É preciso prever que nem todo dispositivo disponibiliza determinados recursos e também que em geral há uma diversidade bastante ampla de tamanhos e resoluções e tela e a aplicação precisa se adaptar a essa diversidade.

Esse tipo de aplicação é desenvolvido com tecnologias *web* em especial HTML5, CSS 3 e Java Script. Podem ser acessados e utilizados por computadores como por dispositivos móveis. Os aplicativos web não são instaláveis como os nativos, para acessá-los deve-se usar o navegador do dispositivo para conectar na URL do aplicativo *web* [Prezotto 2014].

2.3. Aplicações Híbridas

Os aplicativos híbridos são parcialmente nativos e parcialmente *web mobile*. Podem ser baseados em HTML5 e outros padrões web e exibidos através do navegador embutido no aplicativo. Assim como aplicativos nativos são baixados através de lojas de aplicativos, ficam disponíveis na tela principal do dispositivo e podem acessar recursos nativos do aparelho como câmera, GPS e sistema de arquivos.

Aplicações híbridas são populares porque permitem desenvolvimento multiplataforma, utilizando o mesmo código para mais de um sistema operacional. Como exemplos de ferramentas que permitem que isso seja possível podemos citar Cordova, Titanium, entre outros [Wahlbrinck 2015].

3. Desenvolvimento da Aplicação

O uso das tecnologias está transformando relações humanas em todas as suas dimensões, e no âmbito educacional não poderia ser diferente. Não é de hoje que se percebe a utilização de equipamentos eletrônicos em espaços escolares, no entanto, nos últimos anos observa-se uma ampliação na utilização de dispositivos móveis [Vergutz et al. 2014] [Sousa et al. 2011].

Tais dispositivos são ótimas ferramentas para otimizar o trabalho do professor e auxiliá-lo em sala de aula. Pensando nisso, a aplicação Diário de Classe foi desenvolvida para facilitar o trabalho do professor ao realizar a chamada, oferecendo uma alternativa aos cadernos de papel.

Para definir quais os serviços que o sistema deve oferecer e como deve reagir à interação do usuário, foi realizada a análise de requisitos antes do desenvolvimento do mesmo. Assim, a aplicação foi desenvolvida seguindo os requisitos descritos abaixo:

- Deverá ser permitido o cadastro de turmas e alunos, podendo vincular os alunos a uma turma, editar e excluir os registros;
- A chamada será realizada após o cadastro de uma nova aula, e o professor apenas precisa informar se o aluno está presente na aula ou não;
- As informações poderão ser exportadas a fim de visualização em outro dispositivo;
- A aplicação deverá gerar relatórios da frequência dos alunos divididos por turmas;

Requisitos não funcionais declaram as características da qualidade que o sistema possui e que estão relacionadas a sua funcionalidade. Para esta aplicação foram propostos os seguintes requisitos não funcionais:

- **Disponibilidade**: a aplicação deve funcionar independente de conexão com internet, para que possa ser usada em qualquer hora e lugar. Deve também, ser portada para mais de uma plataforma móvel.
- **Praticidade e Usabilidade**: como tem o objetivo de auxiliar o professor em sala de aula, a aplicação deve ser simples e prática, com um layout intuitivo, que não necessite de um período de adaptação.
- **Fluidez**: a aplicação deve ser leve e fluída, priorizando sempre seu desempenho.

3.1. Ferramentas de Desenvolvimento

A ideia principal é que a aplicação seja simples e prática. Pensando nisso foi desenvolvida para que possa ser utilizada *offline*, não necessitando de conexão com a internet para utilizá-la. Para isso ser possível, foi utilizado o banco de dados Sqlite, que cria o banco de dados no próprio dispositivo, dispensando o uso de *web services* ou outros meios para conexão com os dados da aplicação.

Para desenvolvimento da aplicação foi utilizado o *framework* Ionic, utilizado para criar a parte visual e lógica da aplicação. A compilação da mesma, foi realizada utilizando o Cordova, que já está integrado com o Ionic. As seções abaixo, descrevem mais detalhadamente cada um dos *framerowks* citados anteriormente.

3.1.1. Ionic v2

Ionic *Framework* é um SDK de código aberto que permite aos desenvolvedores criar aplicativos móveis de alta performance e qualidade utilizando tecnologias *web* já conhecidas (HTML, CSS e *JavaScript*). O Ionic é construído em cima do AngularJS e utiliza o Cordova para construção da aplicação de um *Webview*, capaz de renderizar as tecnologias previamente citadas. Devido a esta integração com o AngularJS o Ionic herdou várias diretivas, que é uma das *features* do ecossistema do AngularJS que permite extender o HTML de modo a fornecer reuso, componentização e lógica para a marcação.

Ionic é focado principalmente na aparência e no *Look and Feel*, ou seja, na experiência do usuário com a interface de uma aplicação, através da construção de *layouts* poderosos e de simples desenvolvimento. Existe uma dependência do AngularJS para desenvolvimento que permite consumir todos os recursos que o Ionic pode oferecer, porém pode-se utilizar apenas o CSS, perdendo assim componentes de interação, animação dentre outros já fornecidos na plataforma.

Ionic é modelado em SDKs de desenvolvimento mobile nativos conhecidos, o que facilita o entendimento para qualquer pessoa que já tenha desenvolvido algo para Android ou iOS. Seu *design* foi projetado para ser simples, limpo e funcional. Possui tipografia, componentes e paradigmas interativos baseados em plataformas populares e foi pensado e desenvolvido para que funcione perfeitamente em todos os tipos de dispositivos [IONIC 2016].

3.1.2. Cordova

Apache Cordova é um *framework* de desenvolvimento móvel de código aberto. Ele permite que sejam utilizadas tecnologias *web* padrão, como HTML5, CSS3 e *JavaScript*,

para desenvolvimento multiplataforma. Os aplicativos são executados em invólucros direcionados para cada plataforma, e contam com *plugins* para acessar os recursos de cada dispositivo, tais como GPS, sistema de arquivos e câmera.

Cordova envolve um app HTML/*JavaScript* em um recipiente nativo que pode acessar as funções do dispositivo de várias plataformas. Estas funções são acessadas através de uma API (*application program interface*) *JavaScript* unificada, permitindo que com um único código seja possível atingir quase todos os telefones e/ou tablets no mercado hoje e publicar as aplicações em suas lojas de aplicativos. As plataformas suportadas pelo *framework* são: Android, iOS, BlackBarry10, Ubuntu, Windows Phone 8, Bada, FireOs e LG We OS.

A principal vantagem do Cordova é o pacote de *Plugins* que ele oferece. Estes permitem que a aplicação acesse recursos nativos do dispositivo. Além daqueles oferecidos pelo Cordova, também existem *plugins* de terceiros compatíveis com a plataforma, ou ainda, a possibilidade de criar-se um *plugin* próprio para integrar a aplicação [Cordova 2016].

3.1.3. Sqlite

Sqlite é uma biblioteca que implementa um banco de dados SQL embutido e não precisa de instalação para ser utilizado. Seu uso é recomendado para aplicações onde a simplicidade, implementação e manutenção são mais importantes que os incontáveis recursos que os sistemas de gerenciamento de banco de dados (SGBDs) implementam.

Entre as vantagens de utilizar SQLite pode-se citar que o mesmo é um *software* livre e multiplataforma, possui armazenamento seguro, permite guardar o banco de dados em um único arquivo e não possui dependências externas [Sqlite 2016].

3.2. Ambiente de Desenvolvimento e Testes

Partindo do ponto de que aplicações híbridas são desenvolvidas com tecnologias *web*, não é necessária uma ferramenta específica para desenvolvimento do sistema e o mesmo pode ser testado e depurado em navegadores *web*.

Como ferramenta de desenvolvimento foi utilizado o editor de texto *Sublime*, para edição dos códigos e desenvolvimento da aplicação. Os testes iniciais, principalmente de *layout*, foram realizados no navegador Chrome, que permite que o código seja depurado, a fim de identificar onde está ocorrendo as falhas do sistema, o que possibilita uma manutenção mais rápida da aplicação.

Para versionamento de código foi utilizado o GitHub, que serve como repositório para a aplicação, possibilitando voltar o código a versões anteriores caso exista necessidade. Os testes com a aplicação foram realizados em dois emuladores do Android, com versões do sistema operacional diferentes, Android 4.4 (Kitkat) e Android 5.0 (Lollipop), e em um *smartphone* Samsung Galaxy com sistema operacional Android versão 4.1.1 (Jelly Bean).

3.3. Ambiente de Desenvolvimento e Testes

Ao final de todo o estudo realizado foi implementado um protótipo da aplicação de Diário de Classe, que permite que o professor cadastre turmas, alunos e aulas, e a partir destes cadastros realize a chamada em sala de aula.

Pensando na usabilidade da aplicação a primeira tela da mesma Figura 1, apresenta as funções que serão mais utilizadas no sistema, seguindo um padrão de *layout* já conhecido por usuários móveis, tornando o uso da ferramenta intuitivo e prático.

Seguindo este mesmo pensamento, o processo de realização da chamada, representado na Figura 2, também deve ser algo prático e sem maiores complicações. Assim, todos os alunos já vem marcados com a presença, sendo apenas necessário dar ausência para alunos que faltaram a aula, visto que o número de ausências tende sempre a ser menor que o de presenças.

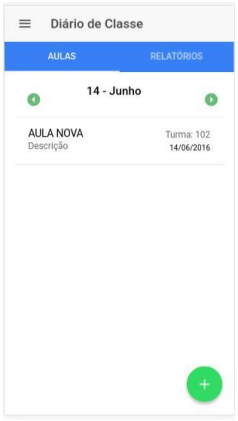

Figura 1. Cadastro de Turmas

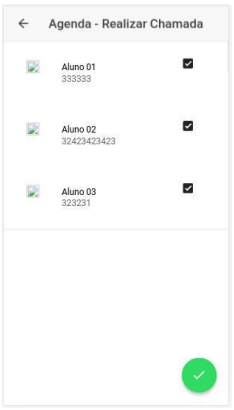

Figura 2. Cadastro de Alunos

O cadastro de turmas, visto na Figura 3, alunos e aulas segue um mesmo padrão de *layout*, feitos ambos em janelas modal que abrem sobre a tela principal da aplicação. Os dados requisitados são apenas os necessários para identificação da turma, alunos e aula, não sendo necessário cadastro de informações irrelevantes ao sistema.

O cadastro de alunos, apresentado na Figura 4, permite acesso a câmera do dispositivo utilizada para tirar a foto do aluno que está sendo cadastrado. Isto é possível através de plugins disponibilizados pelo Cordova, que permitem acesso a recursos nativos do dispositivo, neste caso a câmera.

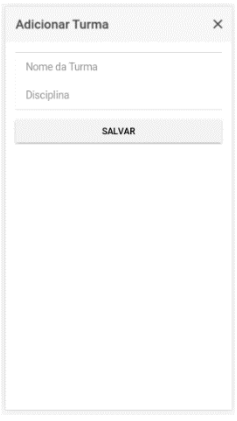

Figura 3. Cadastro de Turmas

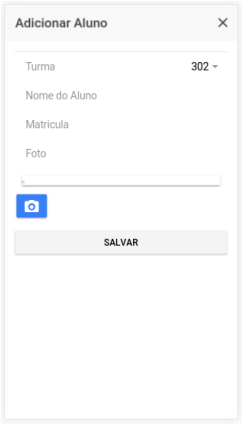

Figura 4. Cadastro de Alunos

Os relatórios de frequência são apresentados separados por turmas. A partir da escolha da turma é apresentado o resumo de todas as aulas já cadastradas para a mesma e o relatório de presenças em cada aula. A aplicação permite também a exportação dos dados, para que possam ser visualizados em outro dispositivo.

6. Conclusão

A construção de aplicações multiplataformas que utilizam linguagens *web* corresponde a uma tendência dentro do mercado de produções de aplicativos móveis. No entanto, a aplicação das tecnologias advindas da programação web no processo de desenvolvimento de aplicativos multiplataformas precisa ser feita de maneira cuidadosa, pois apesar de serem mais flexíveis podem enfrentar problemas de desempenho quando não implementados de maneira eficiente. No futuro, pode-se supor, que o aumento típico da velocidade de computação e capacidade de memória vai levar aplicações híbridas a terem melhorias em seus desempenho, porém até la, é necessário analisar todos os fatores na escolha do desenvolvimento de uma aplicação [Roslër et al. 2014].

Entre as vantagens das aplicações híbridas pode-se citar a necessidade de uma curva de aprendizagem baixa, já que na maioria dos casos os *frameworks* utilizam linguagens *web*, a portabilidade da mesma aplicação para vários sistemas operacionais, baixo custo das ferramentas de desenvolvimento, que em sua grande maioria são *free* e *open source* e dependem apenas de um editor de código para desenvolvimento, e desenvolvimento mais rápido, visto que se utiliza o mesmo código para gerar o *build* da aplicação para diversos SO.

Porém, como desvantagem pode-se citar a performance da aplicação, que em muitos casos é o ponto decisivo entre a escolha do desenvolvimento multiplataforma e nativo, já que aplicações híbridas utilizam um *WebView* do *browser*, perdendo a fluidez dos aplicativos nativos. Outra desvantagem é a incompatibilidade de *plugins* com as plataformas nativas, que podem ser incompatíveis em alguns casos.

Como um caso de sucesso de aplicação híbrida, com foco em educação e utilizando o *framework* Ionic, podemos citar o aplicativo Moodle Mobile, com versões para Android e iOS. O Moodle Mobile é a versão móvel do site do Moodle, que é uma plataforma de aprendizagem projetada para educadores, administradores e alunos com um sistema robusto, seguro e integrado para criar ambientes de aprendizagem personalizados. Ele foi desenvolvido pensando em permitir acesso ao sistema em qualquer lugar, tanto que disponibiliza várias funções *offline*, permite receber notificações e mensagens instantâneas, além de disponibilizar toda a grade de cursos e atividades do usuário [Moodle 2016].

Portanto, a escolha em utilizar ou não um *framework* de desenvolvimento multiplataforma *mobile* deve levar em consideração muitos requisitos, entre eles o tempo disponível para desenvolver a aplicação, capital para investimento, número de desenvolvedores que compõe o time de programação, plataformas a serem atingidas, público alvo da aplicação e linguagem utilizada para desenvolvimento. Estes são fatores essenciais para serem analisados no desenvolvimento de um aplicativo móvel, para que a versão final atenda as necessidades do mesmo.

Como trabalho futuro, pretende-se integrar a aplicação com alguma plataforma educacional já existente, para que seja mais simples o processo de cadastro dos alunos e exportação dos dados.

Referências

Cordova, A. (2016). Apache cordova. https://cordova.apache.org/docs/en/latest/guide/overview/index.html. Acessado em 23-05-2017.

da Silva, L. L. B., Pires, D. F., and Neto, S. C. (2015). Desenvolvimento de aplicações para dispositivos móveis: Tipos e exemplo de aplicação na plataforma ios.

IONIC (2016). Ionic 2. http://ionic.io/2. Acessado em 23-05-2017.

Moodle (2016). Moodle mobile. https://docs.moodle.org/31/en/Moodle_ Mobile. Acessado em 11-06-2017.

Prezotto, E. D. (2014). Estudo de frameworks multiplataforma para desenvolvimento de aplicações mobile híbridas. Universidade Federal de Santa Maria.

Roslër, F., Nitze, A., and Schmietendorf, A. (2014). Towards a mobile application performance benchmark. In The Ninth International Conference on Internet and Web Applications and Services (ICIW 2014), pages 55–59.

Sousa, R. P. d., Moita, F. d., Carvalho, A. B. G., et al. (2011). Tecnologias digitais na educação.

SQLite (2016). Sqlite. https://www.sqlite.org/about.html. Acessado em 23-05-2017.

Teleco (2016). Teleco, inteligência em telecomunicações. http://www.teleco.com.br/sist_operacional.asp. Acessado em 12-05-2017.

Vergutz, A., Boniati, B. B., and Wahlbrinck, K. A. (2014). Utilização de tablets por docentes em espaços escolares.

Wahlbrinck, K. A. (2015). Análise de performance de frameworks para desenvolvimento multiplataforma mobile. Universidade Federal de Santa Maria.