Análise comparativa de desempenho de aplicação Android com persistência em Banco de Dados Relacional e Banco de Dados Orientado a Objetos

José Dijon de O. Neto[1], Iverton P. L. Maia[1], Emilio C. L. Lemos[1], Angélica Félix de Castro[1]

[1]Centro de Ciências Exatas e Naturais – Universidade Federal Rural do Semi-Árido (UFERSA) – Campus Mossoró
59.625-900 – Mossoró – RN – Brasil

{emiliolemos,iverton_croeger,jdoneto}@hotmail.com,
angelica@ufersa.edu.br

Abstract. *When it comes to databases on mobile devices, it is necessary to do a study and analysis to assess which technology is most effective for storing and processing information in mobile environments. The main objective of this work was to verify the feasibility of implementing a mobile application using a relational and object-oriented database. The same database was developed in SQLite (relational) and DB4o (object-oriented) in an Android environment; where the results showed that DB4o performed better than SQLite in most of the conditions tested.*

Resumo. *Em se tratando de bancos de dados em dispositivos móveis, é necessário fazer um estudo e uma análise para avaliar qual tecnologia é mais eficaz para armazenar e processar informações em ambientes móveis. O principal objetivo do presente trabalho foi verificar a viabilidade da implementação de um aplicativo móvel usando banco de dados relacional e orientado a objetos. O mesmo banco de dados foi desenvolvido em SQLite (relacional) e em DB4o (orientado a objetos) em um ambiente Android; onde os resultados mostraram que o DB4o teve melhor desempenho que o SQLite na maioria das condições testadas.*

1. Introdução

De acordo com Elmasri e Navathe (2010), os Bancos de Dados (BD´s) possuem as seguintes propriedades explícitas: (a) um banco de dados representa alguns aspectos do mundo real, sendo chamado, às vezes, de minimundo ou de universo de discurso, onde as mudanças no minimundo são refletidas em um banco de dados; (b) um banco de dados é uma coleção lógica e coerente de dados com algum significado inerente; e (c) um banco de dados é projetado, construído e povoado por dados, atendendo a uma proposta específica. Os BD´s possuem um grupo de usuários definido e algumas aplicações preconcebidas, de acordo com o interesse desse grupo de usuários. Em outras palavras, um banco de dados possui algumas fontes das quais os dados são derivados, alguns níveis de interação com os eventos do mundo real e um público efetivamente interessado em seus conteúdos.

Um Banco de Dados Relacional (BDR) é um banco de dados que modela os dados de uma forma que eles sejam percebidos pelo usuário como tabelas - ou, mais formalmente como relações [Elmasri e Navathe 2010]. Já um Banco de Dados Orientado a Objetos (BDOO) pode ser utilizado como alternativa aos Bancos de Dados Relacionais para armazenar objetos compartilhados entre diferentes aplicações. Os BDOO´s se tornaram conhecidos com o crescente uso de linguagens orientada a objetos [Mansueli 2016]. BDOOs partem de uma premissa simples: o que se persiste são os objetos e, portanto, o seu "estado", representado pelos atributos. Os atributos seriam equivalentes aos campos (ou colunas) de uma tabela. Já as associações entre objetos (atributos que referenciam outros objetos) podem ser comparadas aos relacionamentos em Sistemas Gerenciadores de Bancos de Dados Relacionais, criados como restrições de integridade referencial ("chaves estrangeiras"). Assim, o correspondente a uma "tabela-filha" em um BDOO seria um atributo que possui como valor outro objeto [Mansueli 2016].

Quando se trata de armazenamento de dados para o sistema operacional Android, é comum a adoção do banco de dados relacional conhecido por SQLite, que é gratuito, livre, portável e confiável [SQLite 2017]. Porém, um dos problemas envolvendo bancos de dados relacionais e aplicações orientadas a objetos é que as classes da aplicação não podem ser gravadas diretamente no banco de dados, sendo necessário definir um mapeamento entre cada classe e a tabela correspondente do banco de dados. Embora a maioria das aplicações desenvolvidas para Android utilize o paradigma de banco de dados relacional para persistência de dados, é possível adotar uma estratégia alternativa optando por bancos de dados orientados a objetos.

O principal objetivo deste trabalho consiste em implementar uma aplicação para Android, concomitantemente utilizar um banco de dados relacional (no caso, o SQLite) e um banco de dados orientado a objetos (o escolhido foi o Db4o), para, finalmente, fazer uma análise comparativa dos resultados obtidos.

O presente artigo está organizado da seguinte maneira: na Seção 2 é apresentado o referencial teórico e os trabalhos relacionados; na Seção 3 é apresentado o método utilizado neste trabalho; na Seção 4 desenvolvimento do aplicativo e projeto do banco de dados; na Seção 5 são discutidos os resultados; e na seção 6 são as conclusões e os trabalhos futuros.

2. Referencial Teórico

Segundo Lee (2011), o Android é o sistema operacional de código aberto da Google, o qual é projetado para dispositivos móveis e é baseado numa versão modificada do Linux, oferecendo uma abordagem unificada para desenvolvimento de aplicativos que podem executar em dispositivos diferentes como *smartphones* e *tablets*. O Android utiliza o SQLite como banco de dados padrão para armazenamento de dados e a linguagem Java para desenvolvimento de aplicativos.

Há quatro tipos de componentes de um aplicativo Android: *Activities*, *Services*, *Content providers* e *Broadcast receivers* [Monteiro 2012]. Cada tipo tem uma finalidade distinta e um ciclo de vida específico, que define a forma pela qual o componente é criado e destruído. A *Activity* é um componente que fornece uma tela com a qual os usuários podem interagir para fazer algo, como discar um número no telefone, tirar uma

foto, enviar um e-mail ou visualizar um mapa. Cada *activity* recebe uma janela que exibe a interface do usuário. A *UI thread* é a única *thread* que pode modificar a interface gráfica e a principal *thread* de aplicação. Se todo o fluxo de processamento for colocado na *thread* principal, em algum momento, a aplicação vai se tornar lenta ou travar.

Para contornar essa situação, foi disponibilizada a classe *AsyncTask* pela Google Inc. (2009) que permite a execução de processamento em *background* (em segundo plano) e a exibição do resultado na *thread* principal (que gerencia a interface gráfica da *activity*), sem que o desenvolvedor precise manipular *Threads* (responsáveis pelo gerenciamento dos processos) e/ou *Handlers* (classe que acessa o thread principal da interface gráfica). Assim, uma tarefa assíncrona, implementada na *AsyncTask*, é definida por três tipos genéricos, chamados *Params*, *Progress* e *Result*, e quatro eventos, chamados *onPreExecute*, *doInBackground*, *onProgressUpdate* e *onPostExecute*. Esses eventos representam as etapas antes, durante e depois da execução.

SQLite é uma biblioteca em linguagem C que implementa um banco de dados relacional SQL embutido; onde lê e escreve diretamente no arquivo de banco de dados no disco [SQLite 2017]. Programas que utilizam essa biblioteca podem ter acesso a banco de dados SQL sem executar um processo SGBD separado. O SQLite é *open-source*, multi-plataforma, portável e altamente confiável, que fornece suporte para comandos SQL, como *insert*, *select*, *update* e *delete*. O SQLite oferece suporte à transações, não exige configurações especiais ou administradores de banco de dados e já vem pré-instalado no Android. Além disso, disponibiliza uma interface para definição e manipulação de dados.

O Banco de Dados Orientado a Objetos DB4objects (DB4o) foi projetado para aplicações do tipo embarcada, cliente-servidor e desktop. Possui modo nativo para Java e .Net. Usa uma única biblioteca de desenvolvimento que se integra facilmente às aplicações e executa de forma confiável e escalável tarefas de persistência com somente algumas linhas de código, não importando o quanto são complexas as estruturas. Não necessita utilizar-se de consultas SQL para CRUD (Create, Read, Update, Delete). Apresenta algumas vantagens em relação ao Banco de Dados Relacional: oferece rapidez de inserção, utiliza pouco recurso computacional, é de fácil aprendizado, permite o acesso direto ao banco de dados sem utilizar Mapeamento Objeto-Relacional e ambiente de administração zero, já que as rotinas de melhoria podem ser feitas por um programador [Guerra 2007].

O DB4o é um "motor" de banco de dados orientado a objetos, *open source* e multiplataforma que consiste em um único arquivo *.jar*, projetado para aplicações embarcadas, cliente-servidor ou desktop. Permite armazenar objetos diretamente, não sendo necessárias consultas SQL ou de qualquer *framework* que faça o mapeamento objeto-relacional. Os objetos são persistidos em um arquivo com extensão *.yap*. Os métodos comumente utilizados são: *store*, *query* e *delete*.

Esses bancos de dados foram escolhidos pelos seguintes motivos: o SQLite é o banco de dados relacional próprio para dispositivos móveis, o que se torna um grande facilitador no desenvolvimento de um aplicativo; e o DB4o é um dos BDOO´s mais consistente e mais robusto encontrados na literatura.

Na Figura 1 é ilustrado como o objeto é desestruturado para o armazenamento em tabelas no Banco de Dados Relacional. Já no DB4o, o objeto é diretamente armazenado como objeto.

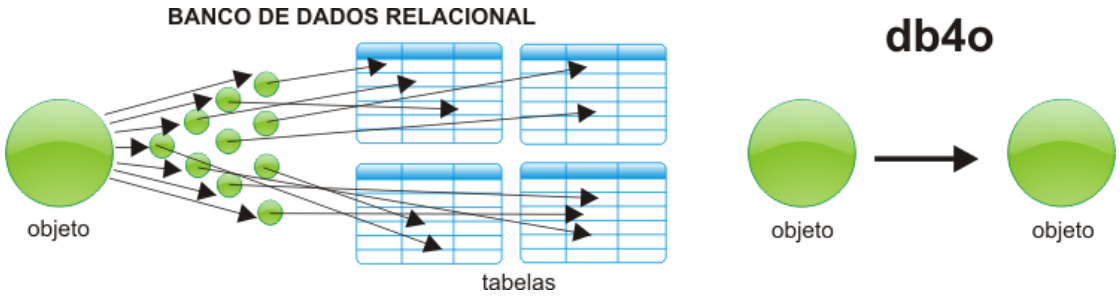


Figura 1. Armazenamento de objetos em um BDR e em um BD Db4o.

De acordo com Boscariolli *et al.* (2006) e Dias (2008), a utilização da tecnologia de banco de dados orientados a objeto permanece baixa, pois os custos envolvidos e o tempo necessário de adaptação à cultura empresarial são dispendiosos. Entretanto, Dias (2008) analisa com mais eficácia o desempenho das operações CRUD em um BDOO utilizando um Sistema Gerenciador de Banco de Dados (SGBD) Relacional, fornecendo números reais provenientes de testes em plataformas diferentes, concluindo que o paradigma Orientado a Objetos (OO) demonstrou um desempenho superior no ambiente configurado para os testes. O presente estudo objetiva avaliar as operações CRUD nos dispositivos móveis.

3. Trabalhos Relacionados

A maioria dos trabalhos encontrados na literatura mostra o desenvolvimento de aplicações em dispositivos móveis usando bancos de dados relacionais. Murakami *et al* (2004) desenvolveram e testaram o uso de dispositivos móveis e portáteis para o acesso a informações hospitalares usando BDRs.

Levis *et al.* (2008) criou um ambiente de aprendizagem móvel que proporciona recursos apropriados para o aprendiz no processo de ensino-aprendizagem. Do ponto de vista educacional, o desafio é a criação de um Nó cenário da educação ubíqua e a criação de um Perfil de Aprendiz, que integre suas localizações, interesses e informações referentes à aprendizagem.

Rovadosky *et al.* (2012) apresentou o uso de realidade aumentada em dispositivos móveis com sistema operacional Android, explorando as possibilidades de desenvolvimento de aplicações com essas tecnologias. Para avaliar as tecnologias, desenvolveu-se um protótipo de ferramenta com o intuito de proporcionar aos usuários maior acessibilidade na busca por informações em ambientes reais. A ferramenta auxilia seus usuários reproduzindo áudio com informações detectadas a partir de marcadores presentes no ambiente real. Os experimentos foram desenvolvidos com o intuito de auxiliar na localização de livros em uma biblioteca, em que se pôde comprovar que a realidade aumentada aliada ao uso de dispositivos móveis representa um grande potencial para o desenvolvimento de soluções nas mais diversas áreas.

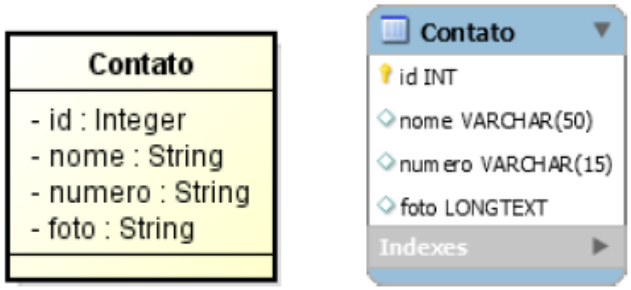
Vale ressaltar que não foram encontrados na literatura pesquisada, trabalhos científicos com o desenvolvimento de aplicativos em dispositivos móveis usando bancos de dados orientados a objetos.

4. Metodologia

Primeiramente foram desenvolvidos dois aplicativos de agenda de contatos para o sistema operacional de dispositivos móveis Android, um utilizando o banco de dados relacional SQLite e o outro utilizando banco de dados orientado a objetos DB4o.

Os aplicativos consistem na criação de apenas uma *Activity* para executar as operações de CRUD e depois exibe o resultado na caixa de diálogo chamada de *AlertDialog*.

A classe e a entidade de persistência ***Contato*** representa o armazenamento dos seguintes dados: identificador, nome, número do telefone e foto do perfil no dispositivo, conforme diagrama de classes – Figura 2(a) – e diagrama de entidade relacionamento – Figura 2(b).



(a) Diagrama de classes (b) Diagrama ER

Figura 2. Modelagem da Classe e entidade de persistência.

Para cada técnica de persistência foi utilizado um aplicativo de teste com a seguinte descrição: implementação de uma rotina que realiza automaticamente operações de inserção, exclusão, alteração e consulta repetidamente por 100, 1000 e 10000 vezes cada uma, obtendo-se o tempo médio para cada operação em cada caso.

O tempo de execução para cada operação CRUD e em cada uma das técnicas foi constatado a partir da diferença, em nanossegundos, entre o instante anterior e posterior à instrução que requisitava o acesso à base de dados.

Considerando que, em dispositivos móveis, um mesmo aplicativo possui apenas uma única instância que acessa o banco de dados local do dispositivo, acessos concorrentes não foram implementados neste estudo.

Os testes foram realizados em um dispositivo restaurado de fábrica, dedicado a este fim, sem nenhum aplicativo aberto e com o banco de dados iniciado sem registros. O dispositivo móvel utilizado foi o Samsung Galaxy SIII Mini (GT-I8190L) [Nogueira 2013].

5. Resultados Obtidos

Foram testadas a performance e o desempenho do SQLite e do DB4o. As Figuras de comparação exibem as duas versões da aplicação **Contato**, cada uma adotando um banco de dados diferente (i.e., SQLite e DB4o).

Na Figura 3 são exibidos os dados comparativos do CRUD nas operações repetidas 100 vezes. Percebe-se que na grande maioria das operações, o DB4o foi mais rápido do que o SQLite, somente na operação *delete*, o mesmo teve uma perda de desempenho.

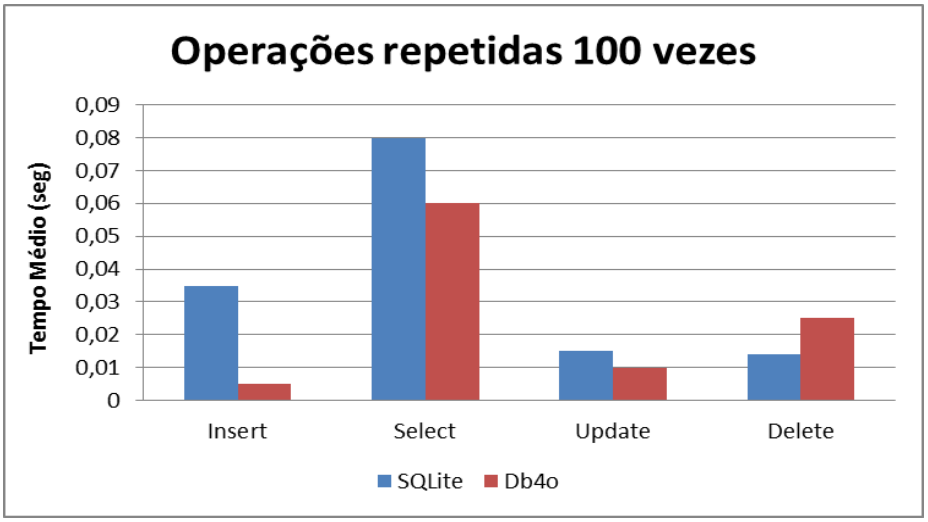


Figura 3. Comparativo do CRUD entre SQLite e DB4o em operações repetidas 100 vezes

Na Figura 4 são demonstrados os dados das operações repetidas 1000 vezes. O BDOO continua se sobressaindo havendo um desempenho similar na operação de *delete.*

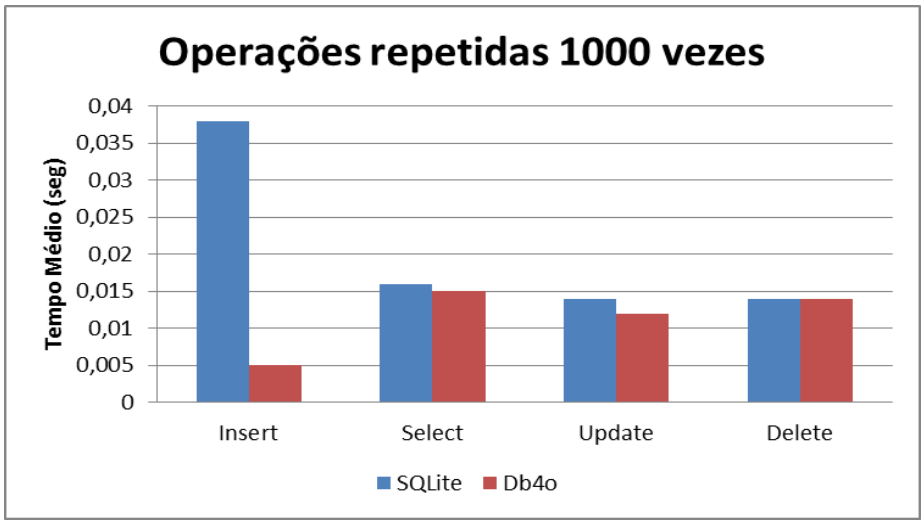


Figura 4. Comparativo do CRUD entre SQLite e DB4o em operações repetidas 1000 vezes

E por fim, na Figura 5 são demonstrados os dados das operações repetidas 10000 vezes. Na operação *select,* o BDOO DB4o apresentou um aumento considerável de tempo se comparado às operações *select* anteriores e ao próprio tempo do SQLite.

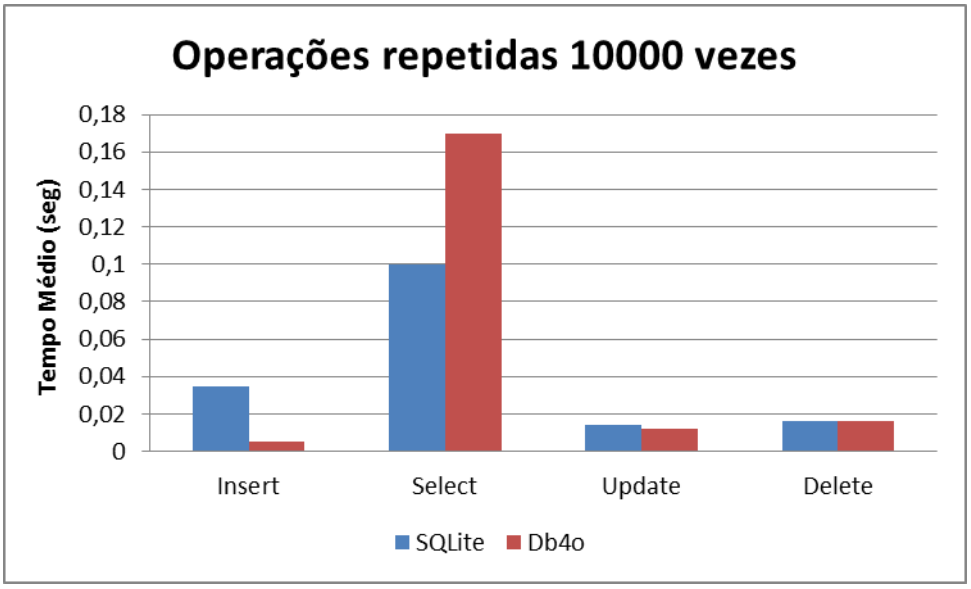


Figura 5. Comparativo do CRUD entre SQLite e DB4o em operações repetidas 10000 vezes

Dessa maneira, é possível perceber que o DB4o obteve um desempenho superior ao SQLite, em quase todos os casos, exceto em dois: *delete* por 100 vezes e *select* por 10000 vezes.

6. Conclusões

De acordo com os resultados obtidos, é possível concluir que o DB4o teve melhor desempenho do que o banco de dados relacional SQLite. Deve-se enfatizar que uma grande vantagem oferecida por esta tecnologia é a otimização do tempo e dos recursos de OO, pois o DB4o permite persistir os objetos conservando todas as suas características, e ainda dispensa todas as tarefas árduas nas atividades com dois modelos diferentes. É importante salientar também que o objeto de estudo se deu pelos testes e observações realizadas em um ambiente isolado, ou seja, os resultados obtidos foram empiricamente produzidos a partir do funcionamento de um dispositivo rodando apenas as configurações padrões.

É importante destacar que o DB4o teve melhor desempenho que o SQLite nas condições em que os testes foram realizados.

Como trabalhos futuros, será testada a influência da herança de classes ou a utilização de classes complexas nos resultados obtidos, diferentemente do atual estudo, que foi realizado apenas com uma única tabela e uma única classe.

Referências

Boscariolli C., Bezerra A., Benedicto M., Delmiro G. (2006) Uma reflexão sobre bancos de dados orientados a objetos. In IV CONGED - Congresso de Tecnologias para Gestão de Dados e Metadados do Cone Sul.

Dias, D. R. C. (2008) Avaliação do uso de sistemas orientados a objetos utilizando BDOO (Java x .Net), 2008. Disponível em: <http://aberto.univem.edu.br/handle/11077/274>. Acesso em: 02 de nov. 2016.

Elmasri, R. e Navathe S. B. (2010), Sistemas de Banco de Dados. 6ª edição. Editora Addison-Wesley.

Google Inc. (2009). AsyncTask - Reference API - Android Developers, 2009. Disponível em: <https://developer.android.com/reference/android/os/AsyncTask.html>. Acesso em: 02 de nov. 2016.

Guerra, G. (2007) DB4Objects na terra de gigantes do BD relacional com Java – Parte I. Disponível em: <http://www.devmedia.com.br/db4objects-na-terra-de-gigantes-do-bd-relacional-com-java-parte-i/4121>. Acesso em: 01 mar 2017.

Lee, Wei-Meng. (2011). Introdução ao Desenvolvimento de Aplicativos para o Android. Rio de Janeiro. Editora Ciência Moderna Ltda, 2011.

Levis D., Barbosa J. L. V. e Pinto S. C. C. S. (2008) Aperfeiçoamento automático do perfil do aprendiz em ambientes de educação ubíqua. In: *Brazilian Journal of Computers in Education (RBIE)*, Vol. 16, Nº 1.

Mansueli, V. A. P. (2016) Bancos de Dados Orientados a Objetos - SQL Magazine 78. Disponível em: <http://www.devmedia.com.br/bancos-de-dados-orientados-a-objetos-sql-magazine-78/17717>. Acesso em: 01 mar 2017.

Monteiro, J. B. (2012), Google Android – crie aplicações para celulares e tablets. 1ª edição. Editora Casa do Código.

Murakami A., Kobayashi L.O.M., Tachinardi U., Gutierrez M. A., Furuie S.S., e Pires F.A. (2004) Acesso a Informações Médicas através do Uso de Sistemas de Computação Móvel. In: IX Congresso Brasileiro de Informática em Saúde CBIS'2004. Disponível em: <http://telemedicina.unifesp.br/pub/SBIS/CBIS2004/trabalhos/arquivos/18.pdf>. Acesso em: 01 mar 2017.

Nogueira R.M., (2013) Review: Samsung Galaxy SIII Mini (GT-i8190L). Disponível em: <http://www.showmetech.com.br/review-samsung-galaxy-s3-mini-gt-i8190l/>. Acesso em: 17 mar 2017.

Rovadosky D.S., Pavan W., Dalbosco J. e Cervi C.R. (2012) Uma aplicação de realidade aumentada para dispositivo móvel com sistema operacional Android. In: Revista Brasileira de Computação Aplicada, Vol. 4, Nº 1.

SQLite (2017). Site oficial do SQLite. Disponível em: <http://www.sqlite.org/>. Acesso em: 01 mar 2017.