1 QUADRO TEÓRICO

Neste capítulo serão descritos os principais conceitos e características das tecnologias a serem utilizadas para o desenvolvimento dos *softwares* propostos nos objetivos dessa pesquisa.

1.1 *Java*

Segundo Caelum (2015), a *Sun Microsystems*, no ano de 1992, criou um time sob liderança de James Gosling para desenvolver inovações tecnológicas, para rodar em pequenos dispositivos.

De acordo com Devmedia (2015), como os recursos dos equipamentos eram mínimos, o gestor tentou modificar e estender as linguagens C/C++, porém acabou mudando de ideia e criando uma nova linguagem que foi chamada de Oak, em português Carvalho, árvores existentes em frente ao seu escritório.

Contudo, os advogados responsáveis pelo registro da linguagem não aprovaram a denominação escolhida. Após alguns debates, levantaram possíveis nomes entre eles, *Java* (gíria americana para café), pois sua equipe se motivava com esse alimento.

Entre as principais características pode–se dizer que o *Java* é:

- Orientado a objeto;
- Seguro;
- Independente de plataforma.

Romanato (2015) afirma, que o Java usa a JMV (Java Virtual Machine), que é uma máquina virtual capaz de converter os Byte Codes para a linguagem do sistema operacional utilizado pelo cliente, sem a necessidade de compila–lo para cada plataforma.Dessa maneira um software que é executado no Windows, funcionará normalmente em qualquer outro sistema.

1.2 *Android*

Segundo Monteiro (2012), Android é um sistema operacional baseado no Lixux, feito especialmente para dispositivos móveis, o qual começou a ser desenvolvido no ano de 2003

pela então empresa Android Inc, que em 2005 foi agregada ao Google.

Por ser um software de código aberto, aliás, o primeiro para equipamentos mobile, Krazit (2009) publicou uma entrevista com Rubin, um dos idealizadores do Android, o qual afirma que o sistema pode rodar em equipamentos de diversos fabricantes, evitando assim ficar limitado a poucos dispositivos. Conforme informações do site Android (2015), hoje em dia existe mais de um bilhão de aparelhos espalhados pelo mundo com esse sistema operacional.

Conforme afirma Monteiro (2012) as aplicações são executadas em uma máquina virtual denominada Dalvik. Cada utilitário, usa uma instancia dessa máquina virtual tornando assim mais segura a aplicação, por outro lado os softwares só podem acessar algum recurso do dispositivo caso seja formalmente aceito pelo usuário ao instala-lo.

Um dos recursos muito utilizado no Andriod são as chamadas Intents, que de acordo com K19 (2015, p.29), "são objetos responsáveis por passar informações, como se fossem mensagens, para os principais componentes da API do Android, como as Activities, Services e BroadCast Receivers."

Monteiro (2012) diz que as Intents são criadas quando se tem a intenção de realizar algo como por exemplo compartilhar uma imagem, utilizando os recursos já existentes no dispositivo. Existem dois tipos de Intents:

- Intents implícitas – Quando não é informada qual atividade deve ser chamada, ficando assim por conta do sistema operacional verificar qual a melhor opção.
- Intents explicitas – Quando é informada qual atividade deve ser chamada. Usada normalmente para chamar activities da mesma aplicação.

Segundo K19 (2015) uma aplicação Android pode ser construída com quatro tipos de componentes:

- Activitie – São as telas com interface gráficas capazes de terem interações com os usuários.
- Services – São serviços executados em segundo plano, com tarefas que levam algum tempo sem comprometer a interação do usuário.
- Content providers – São os provedores de conteúdo que permitem o acesso e a modificação de dados.
- Broadcast receivers – São componentes capazes de responder a eventos do sistema operacional.

Com a ideia de desenvolver um aplicativo para dispositivos móveis, a plataforma Android foi escolhida devido ao seu destaque no mercado, pela facilidade que apresenta aos usuários e por ser um sistema operacional de código livre.

1.3 *Android Studio*

Umas das ferramentas mais utilizadas para o desenvolvimento em Android é o Eclipse IDE, contudo a Google criou um software especialmente para esse ambiente, chamado Android Studio. Segundo Gusmão (2014), ele é uma IDE baseado no ItelliJ Idea e foi apresentado na Conferência para desenvolvedores I/O de 2013.

Carvalhos (2013) afirma que as maiores vantagens de se utilizar esse programa é a possibilidade de customizar o tema e os atalhos. A programação tornou-se mais rápida com o auto – complete que não necessita de nenhum comando, pois ele já vai completando de acordo com que é feita a digitação. Além disso, tem uma interface atraente com grande facilidade para a programação, sendo que é possível arrastar os elementos da view. Existe também uma forma simples para a integração com as ferramentas de controle de versão, como o GitHub. Os downloads necessários para um projeto podem ser feitos diretamente pela própria IDE sem ter a necessidade de ficar procurando nas páginas dos desenvolvedores.

Gusmão (2014) diz que a plataforma está disponível para Windows, Mac e Linux, e os programadores terão disponíveis uma versão estável e mais três versões que estarão em teste chamadas de Beta, Dev e Canary.

1.4 *Web Services*

Nos tempos atuais, com o grande fluxo de informação que percorre pelas redes da *internet* é necessário um nível muito alto de integração entre as diversas plataformas, tecnologias e sistemas. Como uma provável solução para esse ponto, já existem as tecnologias de sistemas distribuídos. Porém essas tecnologias sofrem demasiadamente com o alto acoplamento de seus componentes e também com a grande dependência de uma plataforma para que possam funcionar. Com intuito de solucionar a estes problemas e proporcionar alta transparência entre as várias plataformas, foram criados as tecnologias *web services*.

De acordo com Erl (2015):

> No ano de 2000, a W3C (*World Wide Web Consortium*) aceitou a submissão do *Simple Object Access Protocol* (SOAP). Este formato de mensagem baseado em XML estabeleceu uma estrutura de transmissão para comunicação entre aplicações (ou entre serviços) via HTTP. Sendo uma tecnologia não amarrada a fornecedor, o SOAP disponibilizou uma alternativa atrativa em relação aos protocolos proprietários tradicionais, tais como CORBA e DCOM.

Considera-se então a existência dos *web services* a partir daí. De acordo com Durães (2005), *Web Service* é um componente que tem por finalidade integrar serviços distintos. O que faz com que ele se torne melhor que seus concorrentes é a padronização do XML(*Extensible Markup Language*) para as trocas de informações. A aplicação consegue conversar com o servidor através do WSDL que é documento que contém as regras de funcionamento do *web service*.

Os *web services* além de fornecerem uma padronização de comunicação entre as várias tecnologias existentes, provê transparência na troca de informações. Isso contribui pelo fato de que as novas aplicações possam se comunicar com aplicações mais antigas ou aplicações contruídas sobre outras plataformas.

Além das tecnologias *web services* tradicionais, existe também os *web services* REST que também disponibilizam serviços, porém não necessitam de encapsulamento de suas mensagens assim como os *web Services* SOAP. Este fato influencia diretamente na performance da aplicação como um todo, haja vista que não sendo necessário o encapsulamento da informação requisitada ao *web service*, somente é necessário o processamento e tráfego da informação que realmente importa. As caracteristícas do padrão REST serão abordadas a seguir.

1.4.1 REST

Segundo Saudate (2012), REST é a sigla de *Representational State Transfer*, ou em português Transferência de Estado Representativo, desenvolvido por Roy Fielding na defesa de sua tese de doutorado. Segundo o próprio Fielding (2000) REST é um estilo que deriva do vários estilos arquitectónicos baseados em rede e que combinado com algumas restrições, fornecem uma interface simples e uniforme para fornecimento de serviços[1].

Rubbo (2015), afirma que os dados e funcionalidade de um sistema são considerados recursos e podem ser acessados através das URI's *(Universal Resource Identifier)*, facilitando dessa forma a comunicação do servidor com o cliente.

[1] Tradução e resumo de informações de responsabilidade do autor da pesquisa.

Saudate (2012), explica ainda que os métodos do HTTP[2] podem fazer modificações nos recursos através dos comandos:

- GET – Para recuperar algum dado.
- POST – Para criar algum dado.
- PUT – Para alterar algum dado.
- DELETE – Para excluir algum dado.

Segundo Godinho (2009), não há um padrão de formato para as trocas de informações, mas as que mais são utilizadas é o XML[3] e o JSON[4]. O REST[5] é o mais indicado para aplicações em dispositivos moveis, devido sua agilidade.

1.5 *Apache Tomcat*

De acordo com a Tomcat (2015), *Apache Tomcat* é uma implementação de código aberto das especificações *Java Servlet* e *JavaServer Pages*. O *Apache Tomcat* é um *Servlet Container*, que disponibiliza serviços através de requisições e respostas.

Segundo Tomcat (2015), o projeto desse *software* começou com a *Sun Microsystems*, que em 1999 doou a base do código para *Apache Software Foundation* onde foi lançada a versão 3.0.

Desta forma, o cliente envia uma requisição através do seu navegador, o servidor por sua vez a recebe, executa o *servlet* e devolve a resposta ao usuário.

1.6 PostgreSQL

De acordo com Sourceforge (2015), PostgreSQL é um Sistema Gerenciador de Banco de Dados Objeto-Relacional (SGBDOR) de código aberto desenvolvido na universidade da Califórnia em Berkeley.

[2] HTTP – Abreviação para *HyperText Transfer Protocol*
[3] XML – Abreviação para *Extensible Markup Language*.
[4] JSON – Abreviação para *JavaScript Object Notation*.
[5] REST – Abreviação para *Representational State Transfer*.

Segundo Milani (2008), a primeira versão foi lançada em 1987 com apoio de órgãos como *Army Research Office* (ARO) e a *National Science Foundation* (NSF). A primeira grande mudança ocorrida foi em 1994 criando o Postgres95 com a vantagem da incorporação da linguagem SQL (*Structured Query Language*).

Biazus (2015), afirma que com a popularização rebatizaram–o de PostgreSql com implementação e melhorias de recursos com padrão SQL.

Conforme Sourceforge (2015), o sistema suporta funcionalidades modernas como:

- Comandos complexos.
- Chaves estrangeiras.
- Integridade transacional.

Os programadores podem melhora-lo criando funcionalidades como:

- Funções.
- Operadores.
- Tipos de dados.

O PostgreSQL é um *software* de fácil utilização e multiplataformas o que leva a ser implantado em muitas empresas.

REFERÊNCIAS

BIAZUS, D. : **PostgreSQL Wiki.** 2015. Disponível em: <https://wiki.postgresql.org/wiki/Introdu%C3%A7%C3%A3o_e_Hist%C3%B3rico>. Acesso em: 11 de Março de 2015.

CAELUM. : **Java e Orientação a Objetos**. 2015. Disponível em: <http://www.caelum.com.br/apostila-java-orientacao-objetos/o-que-e-java/#2-2-uma-breve-historia-do-java>. Acesso em: 11 de Fevereiro de 2015.

DEVMEDIA. : **A história da tecnologia Java.** 2015. Disponível em: <http://www.devmedia.com.br/a-historia-da-tecnologia-java-easy-java-magazine-1/18446>. Acesso em: 20 de Fevereiro de 2015.

DURãES, R. : **Web Services para iniciantes**. 2005. Disponível em: <http://imasters.com.br/artigo/3561/web-services/web-services-para-iniciantes/>. Acesso em: 10 de Março de 2015.

ERL, T. : **Introdução às tecnologias Web Services:** soa, soap, wsdl e uddi. 2015. Disponível em: <http://www.devmedia.com.br/introducao-as-tecnologias-web-services-soa-soap-wsdl-e-uddi-parte1/2873>. Acesso em: 26 de Abril de 2015.

FIELDING, R. T. : **Architectural Styles and the Design of Network-based Software Architectures**. Tese (Doutorado) — University of California, 2000.

GODINHO, R. : **Criando serviços REST com WCF.** 2009. Disponível em: <https://msdn.microsoft.com/pt-br/library/dd941696.aspx>. Acesso em: 01 de Março de 2015.

MILANI, A. : **PostgreSQL.** São Paulo: Novatec, 2008.

ROMANATO, A. : **Entenda como funciona a Java Virtual Machine (JVM).** 2015. Disponível em: <http://www.devmedia.com.br/entenda-como-funciona-a-java-virtual-machine-jvm/27624>. Acesso em: 08 de Março de 2015.

RUBBO, F. : **Construindo RESTful Web Services com JAX-RS 2.0.** 2015. Disponível em: <http://www.devmedia.com.br/construindo-restful-web-services-com-jax-rs-2-0/29468>. Acesso em: 03 de Março de 2015.

SAUDATE, A. : **REST:**construa api's inteligentes de maneira simples. São Paulo: Casa do Código, 2012.

SOURCEFORGE. : **Documentação do PostgreSQL 8.2.0.** 2015. Disponível em: <http://pgdocptbr.sourceforge.net/pg82/intro-whatis.html>. Acesso em: 11 de Março de 2015.

TOMCAT, A. : **The Tomcat Story.** 2015. Disponível em: <http://tomcat.apache.org/heritage.html>. Acesso em: 08 de Março de 2015.