**NÍCOLAS HENRIQUE VIEIRA TOLEDO**

**SEBASTIÃO BATISTA DE ANDRADE NETO**

**EXTRAINDO ESTATÍSTICAS DE**

**REDES SOCIAIS USANDO BIG DATA**

**UNIVERSIDADE DO VALE DO SAPUCAI**

**POUSO ALEGRE**

**2015**

**NÍCOLAS HENRIQUE VIEIRA TOLEDO**

**SEBASTIÃO BATISTA DE ANDRADE NETO**

**EXTRAINDO ESTATÍSTICAS DE**

**REDES SOCIAIS USANDO BIG DATA**

Pré-projeto de pesquisa apresentado à disciplina de TCC 1 do Curso de Sistemas de Informação como requisito parcial para obtenção de créditos.

**UNIVERSIDADE DO VALE DO SAPUCAI**

**POUSO ALEGRE**

**2015**

# QUADRO TEÓRICO

## *Big Data*

De acordo com Alecrim (2013, s.p.), a princípio, podemos definir o conceito de Big Data como sendo conjuntos de dados extremamente amplos e que, por este motivo, necessitam de ferramentas especialmente preparadas para lidar com grandes volumes, de forma que toda e qualquer informação nestes meios possa ser encontrada, analisada e aproveitada em tempo hábil.

De maneira mais simplista, a ideia também pode ser compreendida como a análise de uma massa de dados para a geração de resultados relevantes que, em volumes menores, dificilmente seriam alcançados.

## *Storm*

De acordo com Leibiusky, Eisbruch e Simonassi (2012), o *Storm* é um sistema distribuído, confiável e tolerante a falhas desenvolvido para processar fluxos de dados.

Sua estrutura consiste em diferentes tipos de componentes. Assim trabalho é dividido entre os componentes, cada qual é responsável por uma tarefa de processamento simples e específica. O fluxo de entrada do *cluster[[1]](#footnote-1)* do *Storm* é tratado por um componente chamado *spout*.

Segundo Apache (2015), um *spout* pode ser denominado como a fonte de fluxos em uma topologia. Geralmente, eles irão ler tuplas (sequencias de objetos) de uma fonte externa, posterirormente encaminhando-os para a topologia.

Continuando com Leibiusky, Eisbruch e Simonassi (2012), o *spout* por sua vez direciona os dados para um componente denominado *bolt*, o qual irá transformá-los de alguma maneira. Um *bolt* pode tanto persistir os dados em algum tipo de armazenamento, quanto direcioná-lo para outro *bolt*.

Para melhor entendimento, pode-se imaginar o *cluster* do *Storm* como uma corrente de *bolts*, cada qual executa algum tipo de transformação nos dados expostos pelo *spout*.

Dentre algumas propriedades do *Storm*, podem ser citadas:

* Simples de programar: A complexidade é reduzida;
* Suporte a várias linguagens de programação: É maior facilidade desenvolver linguagem baseada em JVM[[2]](#footnote-2), no entanto qualquer linguagem é suportada desde que se implemente uma pequena biblioteca intermediária;
* Tolerante a falhas: O *cluster* do *Storm* cuida dos trabalhos, realocando as tarefas quando necessário;
* Escalável: O *Storm* pode reatribuir as tarefas para as novas máquinas, caso seja necessário;
* Confiável: Todas as mensagens são obrigatoriamente processadas ao menos uma vez. Em caso de erros, a mensagem pode ser processada novamente. Assim, tem-se a garantia de nunca perder uma mensagem;
* Rápido: Velocidade é um dos fatores-chave do *Storm*;
* Transitivo: Pode-se obter exatamente a mesma semântica de mensagens para praticamente qualquer tipo de computação.

O *Storm* torna-se uma solução eficiente, rápida, confiável e de baixo custo para operações relacionadas à *Big Data*. Grandes massas de dados tem que ser processadas em tempo real, com a garantia que nenhum dado se perca. A ferramenta possibilita este tipo de solução, através de seu design e suas funcionalidades.

## Banco de dados NoSQL

Segundo DevMedia (2015), ao que tudo indica o termo NoSQL foi criado em 1998 para denominar os bancos de dados relacionais de código aberto que não possuíam uma interface. Hoje em dia esse nome é utilizado para bancos de dados que não fazem o uso somente de SQL.

De acordo com a revista Java Magazine (2013), NoSQL é um movimento que promove soluções de armazenamento de dados não relacionais. Ele é composto por diversas ferramentas que, de forma particular e específica, resolvem problemas como tratamento de grande volume de dados, execução de consultas com baixa latência e modelos flexíveis de armazenamento de dados, como documentos XML ou JSON.

NoSQL não têm como objetivo substituir os bancos de dados relacionais, mas apenas propor algumas soluções que em determinados cenários são mais adequadas. Desta forma é possível trabalhar com tecnologias NoSQL e banco de dados relacionais dentro de uma mesma aplicação.

Os tipos de bancos de dados NoSQL que são mais usados são:

* Chave-Valor: Armazenam objetos indexados por chaves, e possibilitam sua busca a partir de sua respectiva chave.
* Documentos: Conjunto de documentos no formato JSON. Os documentos são tratados como objetos únicos, os quais possuem campos com os respectivos valores.
* Coluna: Formado por colunas que contem um conjunto de informações, semelhante a uma tabela.
* Grafos: Formado por dados distribuídos em forma de vértices e arestas, os quais possuem atributos tanto nas arestas quanto nos vértices.

Os bancos de dados NoSQL são implantados quando os bancos de dados relacionais já não estão suportando a demanda de dados, ou seja, estão apresentando lentidão em consultas e processamentos. Um NoSQL pode substituir um banco de dados relacional por completo, ou simplesmente, ser implantado como um “suporte”, realizando as consultas/processamentos em um grande volume de dados onde a velocidade é necessária. O tipo de NoSQL a ser é escolhido após uma análise crítica do gestor de TI da organização, dependendo da necessidade da organização.

### MongoDB

“*MongoDB* é o único banco de dados que aproveita as inovações do NoSQL (flexibilidade, escalabilidade, desempenho) e baseia-se na arquitetura de bancos de dados relacionais (consulta expressiva, índices secundários, forte consistência).” (MongoDB, 2015, s.p.).

## *Cloud Computing*

De acordo com Marinescu (2013), o ideal que a computação pode ser livremente distribuída como uma utilidade pública, do mesmo modo que a água e a eletricidade, foi formulado na década de 1960 pelo cientista da computação e visionário John McCarthy, o mesmo que defendeu a lógica matemática na inteligência artificial. Cerca de quatro décadas depois, com a internet já consolidada, a causa da “utilidade computacional” foi resgatada por grandes empresas de TI, tais como Amazon, Apple, Google, HP, IBM, Microsoft e Oracle.

Ainda de acordo com Marinescu (2013), o movimento da computação em nuvem foi um movimento iniciado em algum período em meados da primeira década do novo milênio, cuja ideia principal é que o processamento de informações pode ser feito com mais eficiência em métodos de armazenamento via internet, sendo assim alocados em um “*server farm*”, ou seja, um conjunto de servidores com capacidades computacionais extremamente superiores a de um computador convencional.

Uma das principais vantagens da computação em nuvem é que ela oferece serviços computacionais e de armazenamento escaláveis e elásticos. Todos os recursos utilizados para estes serviços são medidos. Com isso, os usuários serão somente cobrados pelos serviços que eles consomem. Aplicações científicas e de engenharia, mineração de dados, financiamento computacional, jogos, redes sociais, e muitas outras atividades computacionais de uso intensivo de dados podem se beneficiar da computação em nuvem.

A arquitetura da computação em nuvem oferece três tipos de serviços: Software como serviço (SaaS), Plataforma como serviço (PaaS) e Infraestrutura como Serviço (IaaS). Os serviços baseados em SaaS são mais designados ao usuário final. Os serviços baseados em PaaS são mais voltados a área de desenvolvimento, oferecendo assim poderosas ferramentas ao desenvolvedor. Já os serviços baseados em IaaS oferecem serviços de infraestrutura de TI completos, liberando assim a organização para focar nos negócios.

A computação em nuvem já é uma realidade empresarial em dias atuais. O número de organizações que estão adotando o paradigma é crescente.

## Implantação das Tecnologias

Segundo Pichatelli (2013), O Big Data pode se tornar caro de processar e armazenar se implantado em bancos de dados tradicionais. Para resolver esse problema novas tecnologias usam soluções *open source* e plataformas de *hardware* de custo acessível para armazenar os dados de maneira mais eficiente, paralelizar trabalhos e entregar poder de processamento.

Ainda segundo Pichatelli (2013) alguns passos necessários para se conseguir o pleno potencial de Big Data:

* **Coletar:** O dado é coletado das fontes de informação e distribuído por meio de múltiplos nós, por exemplo em um arquitetura *grid*, cada um dos quais processa um subconjunto de dados em paralelo.
* **Processar:** O sistema então usa o mesmo paralelismo gerenciado para ter um desempenho computacional mais rápido em cada nó. Depois, cada nó transforma os resultados das pesquisas em informações mais consumíveis para serem usadas tanto pelos seres humanos (em caso de análise) quando pelas máquinas (em caso de interpretação de resultados em larga escala).
* **Gerenciar:** Geralmente o processamento de Big Data é heterogêneo, originado a partir de diferentes sistemas transacionais. Quase todos os dados precisam ser entendidos, definidos, anotados, limpos e auditados para fins de segurança.
* **Medir:** As análises de negócios devem determinar uma métrica e devem ser acompanhadas constantemente. Geralmente as companhias medem o quanto um dado pode ser integrado/relacionado com um comportamento de consumo ou registro histórico; e como essa integração ou correção aumenta ou diminui com o tempo.
* **Consumir:** O resultado da análise dos dados deve atender a demanda original. Por exemplo, se o resultado for de algumas centenas de *terabytes* de interações em redes sociais, ele pode demonstrar como seus clientes compram produtos complementares. Então, deve haver regras de como os dados de mídias sociais são acessados e atualizados. O mesmo serve para o acesso de dados máquina-a-máquina (M2M).
* **Armazenar:** Como a tendência “*data-as-a-service*” ainda toma forma, cada vez mais os dados permanecem em um único lugar, enquanto os programas de acesso a essas informações se movem. Mesmo que os dados sejam armazenados para o curto prazo de processamento em lote ou para o longo prazo de retenção, as soluções de armazenamento devem ser deliberadamente dirigidas.
* **Governar:** A governança de dados engloba as políticas e fiscalização de informações por meio de uma perspectiva de negócios. Como definido, a governança de dados se aplica a cada um dos seis estágios de entrega de Big Data.

Realmente, o Big Data não é uma solução barata e de fácil implantação. É necessário haver um estudo de caso, para saber se a organização realmente precisa dessa tecnologia. Além do mais, é necessário ter profissionais especializados na área para ocorrer a implantação, e obter a vantagem competitiva desejada.

## HTML

Segundo W3C[[3]](#footnote-3) (2015), HTML é uma abreviação de *Hypertext* *Markup* *Language* - Linguagem de Marcação de Hypertexto. Resumindo em uma frase: o HTML é uma linguagem para publicação de conteúdo (texto, imagem, vídeo, áudio e etc) na Web.

O HTML é baseado no conceito de Hipertexto. Hipertexto são conjuntos de elementos – ou nós – ligados por conexões. Estes elementos podem ser palavras, imagens, vídeos, áudio, documentos etc.

Entre 1993 e 1995, o HTML ganhou as versões HTML+, HTML2.0 e HTML3.0, onde foram propostas diversas mudanças para enriquecer as possibilidades da linguagem. Contudo, até aqui o HTML ainda não era tratado como um padrão. Apenas em 1997, o grupo de trabalho do W3C responsável por manter o padrão do código, trabalhou na versão 3.2 da linguagem, fazendo com que ela fosse tratada como prática comum.

### HTML5

Ainda segundo W3C (2015), um dos principais objetivos do HTML5 é facilitar a manipulação do elemento possibilitando o desenvolvedor a modificar as características dos objetos de forma não intrusiva e de maneira que seja transparente para o usuário final.

Ao contrário das versões anteriores, o HTML5 fornece ferramentas para a CSS[[4]](#footnote-4) e o Javascript[[5]](#footnote-5) fazerem seu trabalho da melhor maneira possível.

## CSS

De acordo com W3C (2015, p. sp)

O CSS formata a informação entregue pelo HTML. Essa informação pode ser qualquer coisa: imagem, texto, vídeo, áudio ou qualquer outro elemento criado.

Essa formatação na maioria das vezes é visual, mas não necessariamente. No CSS Aural, nós manipulamos o áudio entregue ao visitante pelo sistema de leitura de tela.

O CSS prepara essa informação para que ela seja consumida da melhor maneira possível.

### Bootstrap

Segundo Bootstrap (2015, p. sp),

*Bootstrap* é o *framework*[[6]](#footnote-6) mais popular de HTML, CSS e JavaScript para desenvolvimento responsivo na web.

Ele é feito para pessoas de todos os níveis, dispositivos de todas as formas e os projetos de todos os tamanhos.

## JavaScript

De acordo com Microsoft (2015, p. sp),

A linguagem JavaScript é uma linguagem de script interpretada com base em objetos. JavaScript é uma linguagem fracamente tipada, o que significa que você não declara os tipos de dados das variáveis explicitamente. Em muitos casos o JavaScript executa conversões automaticamente quando são necessárias.

### JQuery

De acordo com JQuery (2015, p. sp),

jQuery é uma biblioteca JavaScript rápida, pequena e rica em recursos. Ele faz coisas como conversão de documentos HTML e manipulação, tratamento de eventos, animação, e Ajax[[7]](#footnote-7) mais simples com uma API[[8]](#footnote-8) fácil de usar que funciona em uma gama de navegadores. Com uma combinação de versatilidade e capacidade de extensão, jQuery mudou a maneira que milhões de pessoas escrever JavaScript.

# REFERÊNCIAS

APACHE. **Apache Storm**. Disponível em: <https://storm.apache.org/documentation/Concepts.html>. Acessado em 06/03/2015

BOOTSTRAP, <http://getbootstrap.com/>, acessado em 21 de fevereiro de 2015.

DEVMEDIA, O que é NoSQL?. <http://www.devmedia.com.br/o-que-e-nosql-java-magazine-86/18777>, acessado em 03 de março de 2015.

DEVMEDIA, NoSQL: Conheça mais sobre o mongoDB [http://www.devmedia.com.br/nosql-conheca-mais-sobre-mongodb-revista-sql-magazine-95/23318](http://www.devmedia.com.br/nosql-conheca-mais-sobre-mongodb-revista-sql-magazine-95/23318.com/), acessado em 21 de fevereiro de 2015.

INFO WESTER, O que é Big Data?. <http://www.infowester.com/big-data.php>, acessado em 07 de fevereiro de 2015.

INFO WESTER, O que é Cloud Computing (Computação nas Nuvens)?.  [http://www.infowester.com/cloudcomputing.php](%20http://www.infowester.com/cloudcomputing.php), acessado em 04 de março de 2015.

JQUERY, <http://jquery.com/>, acessado em 21 de fevereiro de 2015.

LEIBIUSKY, J.; EISBRUCH, G.; SIMONASSI, D. **Getting Started Whith Storm.** Sebastopol: O’Reilly, 2012.

MARINESCU, D. C. **Cloud Computing Theory And Pratice.** Waltham: Morgan Kaufmann, 2013.

MICROSOFT, Java Script princípios básicos. <https://msdn.microsoft.com/pt-br/library/ie/6974wx4d%28v=vs.94%29.aspx>, acessado em 21 de fevereiro de 2015.

MONGODB, <http://www.mongodb.org/>, acessado em 26 de fevereiro de 2015.

MOZILLA DEVELOPER NETWORK, O que é AJAX?. https://msdn.m https://developer.mozilla.org/pt-BR/docs/AJAX/Getting\_Started, acessado em 21 de fevereiro de 2015.

PICHATELLI, M . Big Data: Sete Passos iniciais. <http://b2bmagazine.consumidormoderno.uol.com.br/index.php/negocios/item/2768-big-data-sete-passos-iniciais>, acessado em 07 de fevereiro de2015.

TAURION, C. Big Data. Editora Brasport: Rio de Janeiro, 2013. Disponível apenas em formato de livro eletrônico.

W3C, Visão Geral do HTML5. <http://www.w3c.br/cursos/html5/conteudo/capitulo1.html>, acessado em 21 de fevereiro de 2015.

W3C, CSS – Curso W3C Escritório do Brasil. <http://www.w3c.br/pub/Cursos/CursoCSS3/css-web.pdf.com/>, acessado em 21 de fevereiro de 2015.

1. Relacionamento entre dois ou mais computadores para que estes trabalhem de maneira conjunta no intuito de processar uma tarefa. [↑](#footnote-ref-1)
2. *Java Virtual Machine* - Máquina Virtual Java. [↑](#footnote-ref-2)
3. *World Wide Web Consortium*, consórcio internacional no qual organizações filiadas, uma equipe em tempo integral e o público trabalham juntos para desenvolver padrões para a Web. [↑](#footnote-ref-3)
4. *Cascading Style Sheets*, formata a informação entregue pelo HTML. [↑](#footnote-ref-4)
5. Linguagem de programação baseada em *scritps*. [↑](#footnote-ref-5)
6. Conceito que une códigos comuns entre vários projetos de software provendo uma funcionalidade genérica. [↑](#footnote-ref-6)
7. *Asynchronous JavaScript and XML*, ou seja, uso do XML para se comunicar com os scripts do lado do servidor. [↑](#footnote-ref-7)
8. Conjunto de rotinas e padrões de programação para acesso a um aplicativo de software ou plataforma baseado na Web. [↑](#footnote-ref-8)