UNIVERSIDADE DE SÃO PAULO

ESCOLA DE ENGENHARIA DE LORENA

DANIEL RODRIGO XIMENES AMARAL

Desenvolvimento de algoritmos com Apache Spark para tratamento de dados indústriais para Business Analytics com Tableau

Lorena - SP
2020

DANIEL RODRIGO XIMENES AMARAL

Desenvolvimento de algoritmos com Apache Spark para tratamento de dados indústriais para Business Analytics com Tableau

Trabalho de Graduação apresentado à Escola de Engenharia de Lorena - Universidade de São Paulo como requisito parcial para conclusão da Graduação do curso de Engenharia Física.

Prof. Dr. Fabiano Bargos

Lorena - SP
Junho, 2020

Ficha catalográfica elaborada pelo Sistema Automatizado
da Escola de Engenharia de Lorena,
com os dados fornecidos pelo(a) autor(a)

Amaral, Daniel Rodrigo Ximenes
Desenvolvimento de algoritmos com Apache Spark para tratamento de dados industriais para Bussines Analytics com Tableau / Daniel Rodrigo Ximenes Amaral; orientador Fabiano Bargos. - Lorena, 2020.
64 p.

Monografia apresentada como requisito parcial para a conclusão de Graduação do Curso de Engenharia Física - Escola de Engenharia de Lorena da Universidade de São Paulo. 2020

1. Industria 4.0. 2. Engenharia de dados. 3. Apache spark. 4. Big data. 5. Oee. I. Título. II. Bargos, Fabiano, orient.

Aos meus pais, irmão e amigos.

Agradecimentos

Este trabalho é resultado de anos de dedicação e estudo, sendo aluno de escola publica tentei entrar na USP 3 vezes, conseguindo uma vaga na terceira tentativa, permitindo assim a realização de um sonho. Educação no Brasil não é fácil, principalmente por não ser valorizada em famílias cuja a geração anterior não estudou numa universidade. Eu tive que ouvir algumas vezes a frase: "Este negócio de faculdade publica é coisa de filho de rico, filho de pobre tem que trabalhar pra pagar sua faculdade".

Felizmente tive o apoio da minha mãe, a pessoal mais sonhadora que existe que sempre acreditou e me apoiou incondicionalmente, mãe você é um exemplo e uma inspiração pra mim. Também agradeço aos meus irmãos, sem eles eu não teria a honra de ser o primeiro bacharel da família. Agradeço também aos meus avós, que sempre me receberam bem e me apoiaram mesmo me achando doido por "falar com o computador". Agradeço as minhas tias por me apoiarem mesmo sem entender o que eu faço. Sou grato também A Marcilene, cujo carinho, atenção e apoio me fizeram uma pessoa melhor, nossos momentos juntos foram inesquecíveis e vou sempre guarda-los comigo.

O ambiente universitário foi sem duvida o a experiência mais enriquecedora que tive em toda a vida. Conheci pessoas extraordinárias e pude construir amizades para a vida toda. Um agradecimento especial ao Lucas, sempre vou lembrar das novas conversas filosóficas naquele sofá, os 4 anos dividindo um apartamento fizeram da gente mais que amigos, irmãos. Agradeço ao Giovani por sempre se preocupar e cuidar de mim, e mesmo não morando oficialmente com a gente você praticamente morou na nossa casa, compartilhando esses 4 anos sensacionais. Outro amigo que tive o privilégio de morar junto foi o Rodrigo, foram 6 meses bem curtos mas bem divertidos, num período bem difícil de nossas vidas a gente ainda assim se divertia muito. Agradeço também ao João que sem dúvida faz parte desse time de irmãos, e mesmo tendo sumido por um tempo sua volta foi triunfal como uma fênix que renasce das cinzas, obrigado por compartilhar sua energia.

Gostaria também de agradecer a Universidade de São Paulo, que foi uma segunda mãe para mim, fornecendo acesso a oportunidades existentes em poucos lugar do Brasil. Tive acesso a todo o tipo de bolsa, permitindo assim a continuidade dos meus estudos e graças a esse apoio eu estou finalmente me formando.

“Something is only impossible until someone doubts and ends up proving the opposite.”.

ALBERT EINSTEIN

Resumo

O atual ambiente de produção industrial se caracteriza por uma profunda transformação com a integração do de produção industrial ao ambiente digital, conforme o conceito de indústria 4.0. O processo de automação gerou um cenário ao qual a indústria de manufatura se tornou a maior fonte de dados, porém descarta 99%. Apoiando-se nesse cenário é possível desenvolver soluções, que utilizam ferramentas de tratamento de dados, como o Apache Spark, que integrado com tecnologias de computação em nuvem como Google Cloud e ferramentas de análise de dados como Tableau permite desenvolver a integração entre o ambiente de produção e um sistema digital permitindo compreender a real situação da produção na fábrica. O presente trabalho se propõe a realizar o desenvolvimento de um algoritmo para transformação de dados industriais com linguagem de programação Python integrado no ambiente de computação em nuvem da Google. O algoritmo recebe os dados de máquinas industriais, e informações de contexto fornecidas por colaboradores da fábrica, executando as transformações e salvando em um banco de dados, onde uma ferramenta web acessa enviando uma requisição ao Tableau Server, que executa cálculos e gera gráficos sobre os dados processados apresentando informações estratégicas ao time de gerenciamento de planta. Os dados são utilizados para cálculo de métricas essenciais para o gerenciamento de planta industrial.

AMARAL, Daniel R. Ximenes. **Desenvolvimento de algoritmos com Apache Spark para tratamento de dados industriais para Business Analytics com Tableau**. 2020. 64 p. Monograph (Graduation) – University of São Paulo, Lorena - SP.

Palavras-chaves: indústria 4.0. Engenharia de Dados. Apache Spark. Big Data. OEE.

Abstract

The current industrial production environment is characterized by a profound transformation with the integration of industrial production into the digital environment, according to the concept of industry 4.0. The automation process generated a scenario to which the manufacturing industry has become the largest source of data, but discards 99 %. Based on this scenario, it is possible to develop solutions, which use data processing tools, such as Apache Spark, which integrated with cloud computing technologies such as Google Cloud and data analysis tools such as Tableau allows to develop the integration between the production system and a digital system to understand the real production situation at the factory. This work aims to develop an algorithm for transforming industrial data with Python programming language integrated in Google's cloud computing environment. The algorithm receives data from industrial machines, and context information provided by factory employees, executing the transformations and saving it in a database, where a web tool accesses it by sending a request to Tableau Server, which performs calculations and generates graphics about the processed data presenting strategic information to the plant management team. The data is used to calculate metrics essential for industrial plant management.

AMARAL, Daniel R. Ximenes. **Development of algorithms with Apache Spark for industrial data processing for Business Analytics with Tableau**. 2020. 64 p. Monograph (Graduation) – University of São Paulo, Lorena - SP.

Keywords: Industry 4.0. Data Engineer. Apache Spark. Big Data. OEE.

Lista de ilustrações

Figura 1 – Modelo tipico de um processo de BI. 28
Figura 2 – Módulos disponíveis na Plataforma Crave. 29
Figura 3 – Métricas KPI para cliente. 29
Figura 4 – Tecnologias e inovações essenciais para indústria 4.0. 34
Figura 5 – Comparação de dados gerados por diferentes setores, aproximadamente 2 pentabytes de dados industriais são armazenados a cada ano. 36
Figura 6 – Modelo tipico de um processo de BI. 37
Figura 7 – Valores de excelência para OEE 39
Figura 8 – Comparativo entre Engenharia de Dados e Ciência de Dados. 41
Figura 9 – Comparação da performance do Spark em relação a outras ferramentas comumente utilizadas para consultas SQL, Streaming e ML. 42
Figura 10 – Componentes do Apache Spark. 43
Figura 11 – Representação da estrutura de processamento do Spark, vemos que entre cada iteração o resultado é armazenado na memória volátil, e apenas no final com comando *write* o resultado é salvo em disco. 45
Figura 12 – Resumo do processo de tratamento de dados. 49
Figura 13 – Processo ETL a partir da extração dos dados, passando pelas transformações até o salvamento no banco de dados Os dados são segmentados por máquina e todos devem conter um *collector timestamp* que representa o momento de coleta do dado. 50
Figura 14 – Ferramenta Dataproc do Google Cloud mostrando um *cluster* criado com nome spark-faurecia-kpi. 51
Figura 15 – Criação de uma Cloud Function. 52
Figura 16 – Criação de uma Cloud Scheduler. 52
Figura 17 – Código para leitura de dados não processados do Amazon S3. 53
Figura 18 – Código para leitura de dados de contextualização do MongoDB. 53
Figura 19 – Tabelas de dados a partir do MongoDB. 53
Figura 20 – Contextualização de dados de máquina. 54
Figura 21 – Contextualização de dados de turno. 54
Figura 22 – Contextualização de dados de produto. 54
Figura 23 – Processamento de informações de status. 55
Figura 24 – Código para calcular a duração de parada. 56
Figura 25 – Código a contagem de produção em intervalos de uma hora. 56
Figura 26 – Montar o esquema de dados idêntico ao destino final. 56
Figura 27 – Código para salvar dados no Google BigQuery. 57
Figura 28 – Fluxo de interação do processo de BI. 57

Figura 29 – Pagina para consulta de OEE.. 58
Figura 30 – Lista de medidas calculadas pelo Tableau. 58
Figura 31 – Consulta de OEE por semana segmentado por turno. 59
Figura 32 – Apresentação de OEE por semana segmentado por turno para 1 mês de dados coletados. 59

Lista de tabelas

Tabela 1 – Revoluções de industriais por período e por tecnologias e capacidades . . . 33
Tabela 2 – Volume de dados processados por chamada e por dia. 60

Lista de abreviaturas e siglas

API: *Application Programming Interface*

BI: *Business Intelligence*

ETL: *Extract Transform Load*

IDE: *Integrated Development Environment*

ICT: *Internet Comunication Tecnology*

IOT: *Internet of Things*

KPI: *Key Performance Indicator*

ML: *Machine Learning*

NIST: *National Institute of Standards and Tecnology*

OCDE: *Organização para Cooperação e Desenvolvimento Econômico*

OEE: *Overall Equipment Effectiveness*

RAM: *Random Access Memory*

RDD: *Resilient Distributed Datasets*

SQL: *Sequal Query Language*

TI: *Tecnologia da Informação*

URI: *Uniform Resource Identifier*

URL: *Uniform Resource Locator*

VM: *Vitual Machine*

Sumário

Lista de ilustrações 17

Lista de tabelas 19

1 INTRODUÇÃO 25

2 APRESENTAÇÃO DA EMPRESA 27

3 OBJETIVO 31

4 FUNDAMENTAÇÃO TEÓRICA 33
4.1 indústria 4.0 33
4.2 *Big Data* na indústria 35
4.3 *Business Intelligence* (BI) 37
4.4 *Overall Equipment Effectiveness* (OEE) 38
4.4.1 Disponibilidade 39
4.4.2 Performance 40
4.4.3 Qualidade 41
4.5 Engenharia de Dados com Apache Spark 41
4.5.1 Definição de Engenharia de Dados 41
4.6 Apresentação do Apache Spark 42
4.6.1 RDDs 44
4.6.2 Processamento em Paralelo 44
4.7 Computação em nuvem 45
4.7.1 Características essenciais 46
4.7.2 Modelos de Serviços 47
4.7.3 Modelos de disponibilização 47
4.7.4 Vantagens da computação em nuvem 48

5 MÉTODOS COMPUTACIONAIS 49
5.1 Máquina e Parâmetros de Programação 49
5.2 Resumo do processo 49
5.3 Processamento ETL 50

6 APRESENTAÇÃO DOS RESULTADOS 51
6.1 Desenvolvimento do processo ETL 51
6.1.1 Configuração do ambiente 51

6.1.2 Extração 53
6.1.3 Transformação 54
6.1.4 Transformações especificas 55
6.1.5 Carregamento 56
6.2 Processo de BI 57
6.3 Volume de dados processados 60

7 CONCLUSÃO 61

8 TRABALHOS FUTUROS 63

REFERÊNCIAS 65

1 Introdução

Desde os primórdios da revolução industrial a forma pela qual produtos são desenvolvidos e manufaturados vem se transformado, conforme novas descobertas cientificas possibilitaram novas ferramentas, que aumentaram a capacidade de produção (SCHMIDT et al., 2015). Tais transformações são visíveis não só no aumento da disponibilidade de produtos industriais pelo mundo, como também na organização geral da indústria, passando a depender mais e mais da tecnologia para melhorar a rendimento. Conforme a indústria adota novas tecnologias e processos, novos paradgimas surgem permitindo dividir o processo em quatro revoluções industriais, e estamos passando nesse momento pela quarta revolução industrial, com características que combinam tecnologia da informação, robótica, internet das coisas, entre outras (LASI; KEMPER, 2014). Essas tecnologias combinadas formam um nicho de mercado que permite o surgimento de novas empresas, oferecendo serviços que integram o chão de fábrica com as ferramentas de Tecnologia da Informação (TI).

2 Apresentação da empresa

A startup Crave Industry foi fundada em 2016 e conta atualmente com uma equipe de 8 pessoas, a maioria engenheiros e programadores. A empresa foi fundada com o objetivo de atender a necessidade de informações no chão de fabrica, ajudando na gestão de operações em ambientes industriais. Para tanto a empresa se propõe á:

> Conectar linhas de produção a uma plataforma personalizada que permite monitoramento dinâmico e preciso, buscamos ajudar indústrias de manufatura a digitalizar e conectar o chão de fábrica utilizando de tecnologias emergentes como: *Computing, Industrial Internet of Things, Big Data/Analytics e Security*. Ref: Crave Industry

Para tal é fornecido uma plataforma online que permite utilizar-se de serviços de análise avançada de dados para auxiliar na tomada de decisões. Podendo se conectar a qualquer fonte de dados, como:

- Historiadores (OSISoft PI)
- PLCs,
- Sistema Supervisório (SCADA)
- Sistemas de Qualidade
- Banco de Dados (SQL)
- MES
- ERP
- Logs

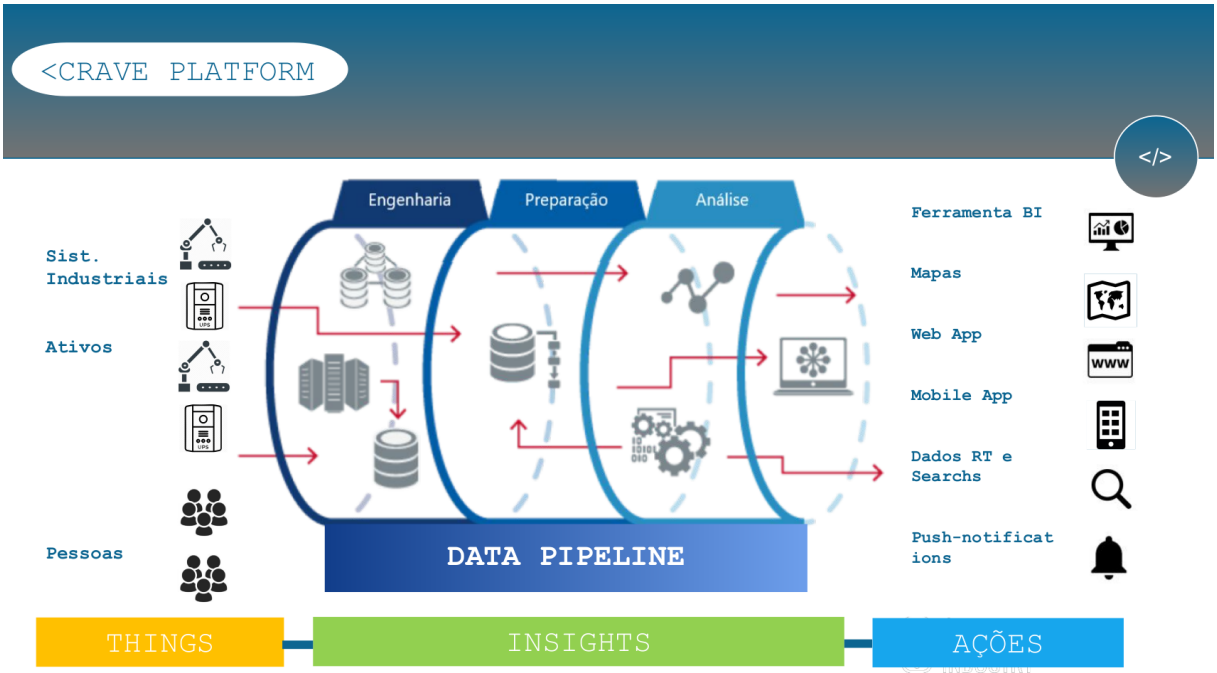
As ferramentas para análise de dados concentram-se em uma plataforma análises como:

- Causa-raiz
- Otimização
- Preditiva
- Controle Estatístico de Processo
- KPIs

- Detecção de Anomalia

A figura 1 resume a plataforma, apresentando desde a coleta de dados até o usuário final, o processo de data pipeline, entendido como ETL.

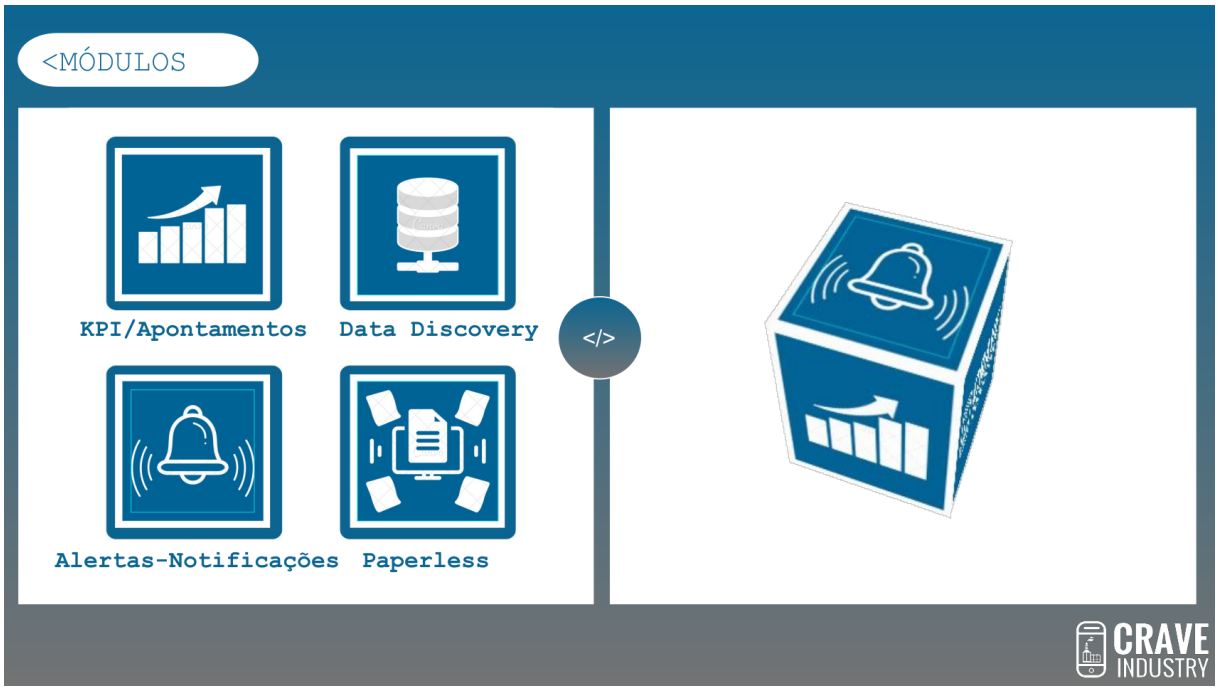
Figura 1 – Modelo tipico de um processo de BI.



Fonte: Crave Industry.

Existem quatro módulos, havendo a possibilidade de adquiri-los separadamente. A imagem 2 apresenta os quatro. O KPI/Apontamentos fornece de forma totalmente digital, rápida e precisa, índices essenciais para tomada de decisão dentro da fabrica. o Data Discovery fornece a capacidade de ler, interpretar e relacionar os dados gerados pelas máquinas durante o processo de fabricação, permitindo avaliar anomalias, fazer previsões, aplicar algoritmos de Inteligência Artificial, entre outros. Alertas e Notificações permite o envio automatizado de alertas customizados ao celulares dos gerentes de fabricas. O modulo Paperless permite eliminar o papel da linha de produção, permitindo gerar formulários de cadastro totalmente digitais para preenchimento em tablets, celulares ou computadores.

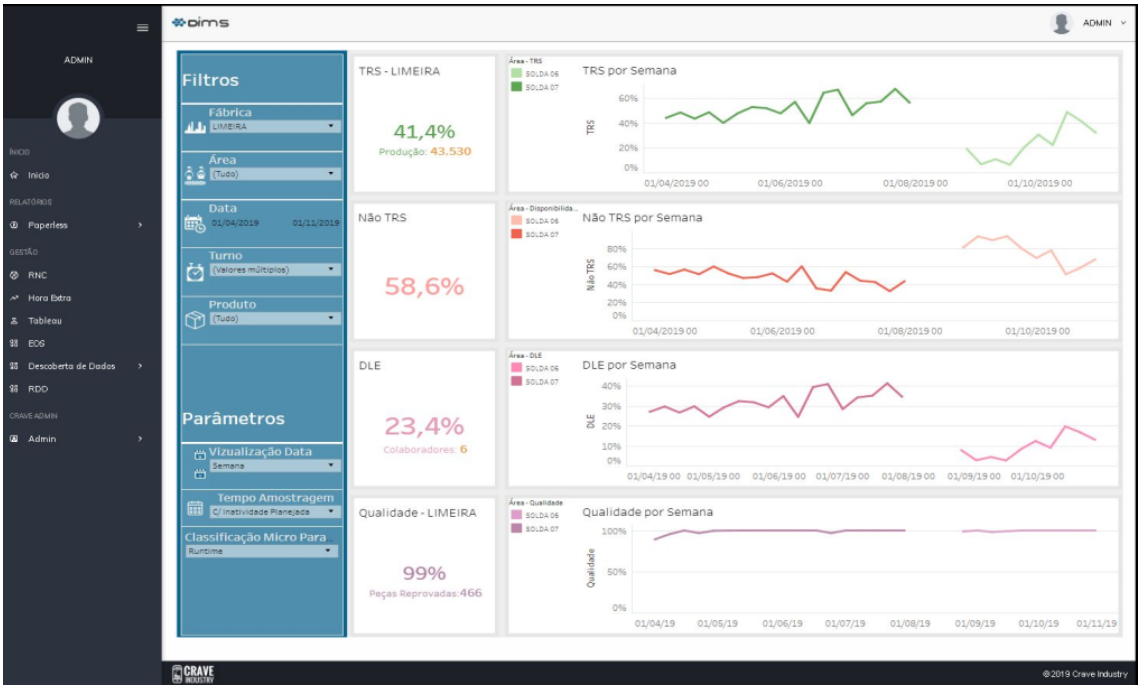
Figura 2 – Módulos disponíveis na Plataforma Crave.



Fonte: Crave Industry.

A figura 3 apresenta um exemplo do produto KPI para um cliente que possui fábrica em Limeira. A plataforma fornece a possibilidade de acompanhar os indicadores segundo as necessidades o cliente.

Figura 3 – Métricas KPI para cliente.



Fonte: Crave Industry.

3 Objetivo

O objetivo do presente trabalho é desenvolver um algoritmo receba os dados coletados das maquinas industriais e salvos na nuvem, faça todas as transformações e contextualizações necessárias e salve como uma tabela de dados para que visualizações permitam a análise desses dados o cálculo de KPIs, como índice OEE, e a criação de visualizações para o Gerente de Planta acompanhar a produção.

4 Fundamentação Teórica

4.1 indústria 4.0

A definição genérica de indústria 4.0 é apresentada como "A completa integração de canais horizontais e verticais entre empresas, departamentos e funções."(GILCHRIST, 2016). Para dar apoio a afirmação de que a quarta revolução industrial é a atual, é necessário apresentar os processos anteriores, com inovações que as distinguem das demais, conforme a tabela 1.

Tabela 1 – Revoluções de industriais por período e por tecnologias e capacidades

Revolução Industrial	Periodo	Tecnologias e Capacidades
Primeira	1784 até século 19	Manufatura impulsionada pelo motor a vapor.
Segunda	Final século 19th -1970s	Manufatura impusionada pela energia elétrica, divisão do trabalho (linha de produção).
Terceira	1970s-Atualmente	Eletrônica e tecnologia da informação impulsiona novos niveis e automação e complexidade.
Quarta	Atualmente	Tecnologia de sensores, interconectividade e análise de dados permitindo altos niveis de customização, integração nas cadeias de valor e maior eficiência.

Fonte: Adaptado de (DAVIES, 2015).

Vale citar que a terceira revolução industrial é diretamente responsável pela quarta, e afirmando que a automação da indústria iniciada nos anos 1970 é a responsável pela disponibilidade de dados observada atualmente na indústria de manufatura (GILCHRIST, 2016).

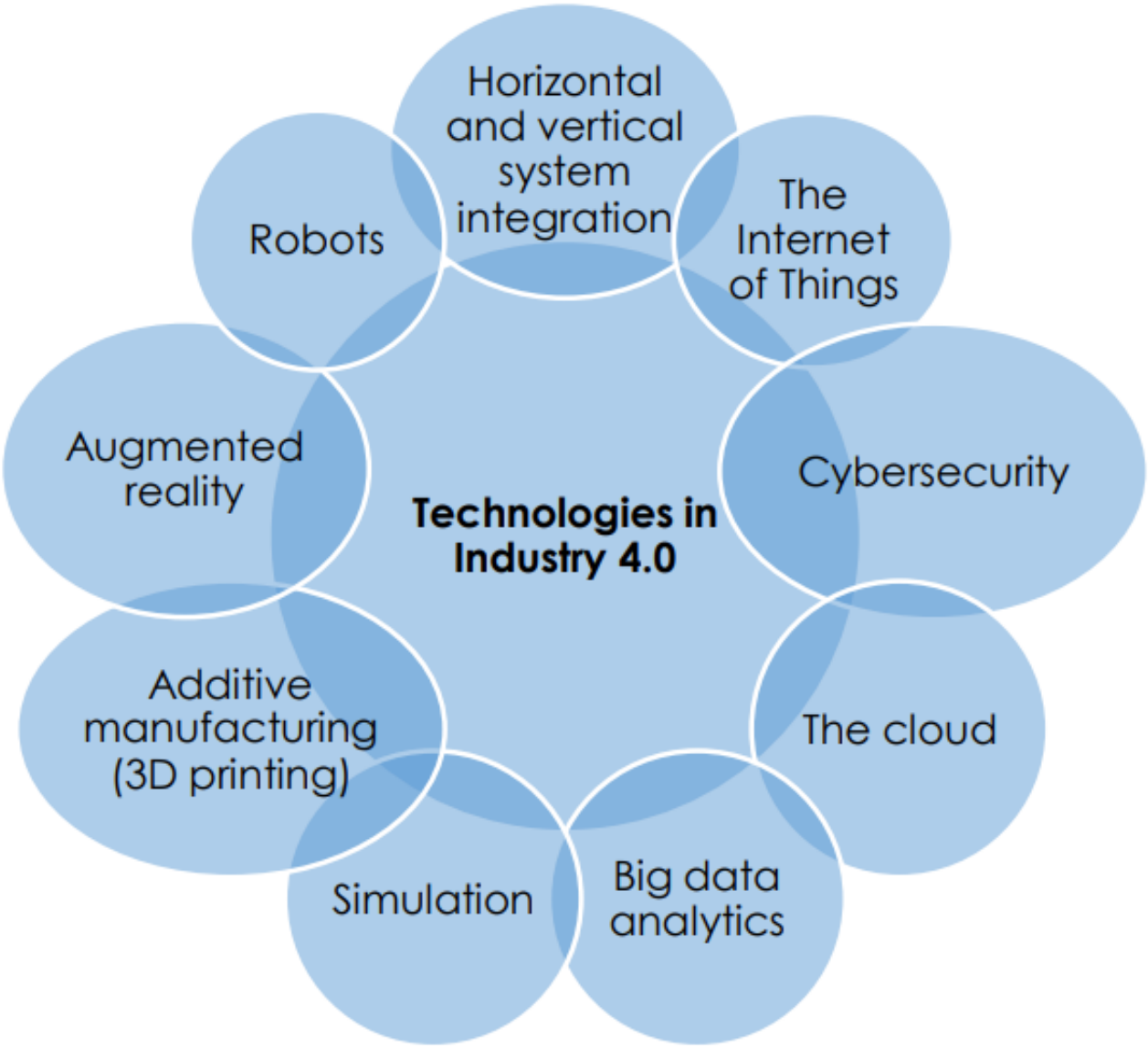
A primeira referencia ao conceito de indústria 4.0 foi apresentado na Conferência *Hannover Fair* em 2011 (Baygin et al., 2016), se referindo as tecnologias atualmente existentes ou em desenvolvimento que integradas ao ambiente industrial, resultariam no aumento da produtividade e da eficiência. Nas palavras da Chanceler Alemã Angela Merkel em discurso a

OCDE a indústria 4.0 é:

> A completa transformação de toda a esfera de produção industrial através da união da tecnologia digital e da internet com a indústria convencional

Entenda-se como a integração de todo o processo de produção, incluindo suprimento, gerenciamento de planta, distribuição e inclusive o produto (DAVIES, 2015). A Figura 4 ilustra as tecnologias que integram para formar o novo estágio de produção industrial.

Figura 4 – Tecnologias e inovações essenciais para indústria 4.0.



Fonte: (BAHRIN et al., 2016).

A maioria das tecnologias apresentadas na Figura 4 pertence a algum segmento da Tecnologia da Informação (TI) e precisam de adaptação para torna-las aplicáveis na indústria de manufatura, dentre essas adaptações pode-se citar:

- A aplicação das Tecnologias da Informação e da Comunicação (ICT), como a Internet das Coisas (IoT) para possibilitar que cada componente do sistema (cadeia de suprimentos, produto, entre outros) seja integrado e digitalizado, trocando e fornecendo informações em tempo real.

- Desenvolvimento de um sistema Cibernético-Físico que se utilize do ICT para monitorar todo o processo em tempo real, integrando sensores, dispositivos de comunicação, câmeras, robôs e todo o tipo de hardware que possa auxiliar na criação de uma versão digital de todo o processo.

- Desenvolvimento de ferramentas de *Big Data* para dados industriais. Necessário para armazenar e processar enormes quantidades de dados gerados por todos os componentes do processo, que devem ser processados, armazenados, e contextualizados para tornar possível obter informações sobre o processo.

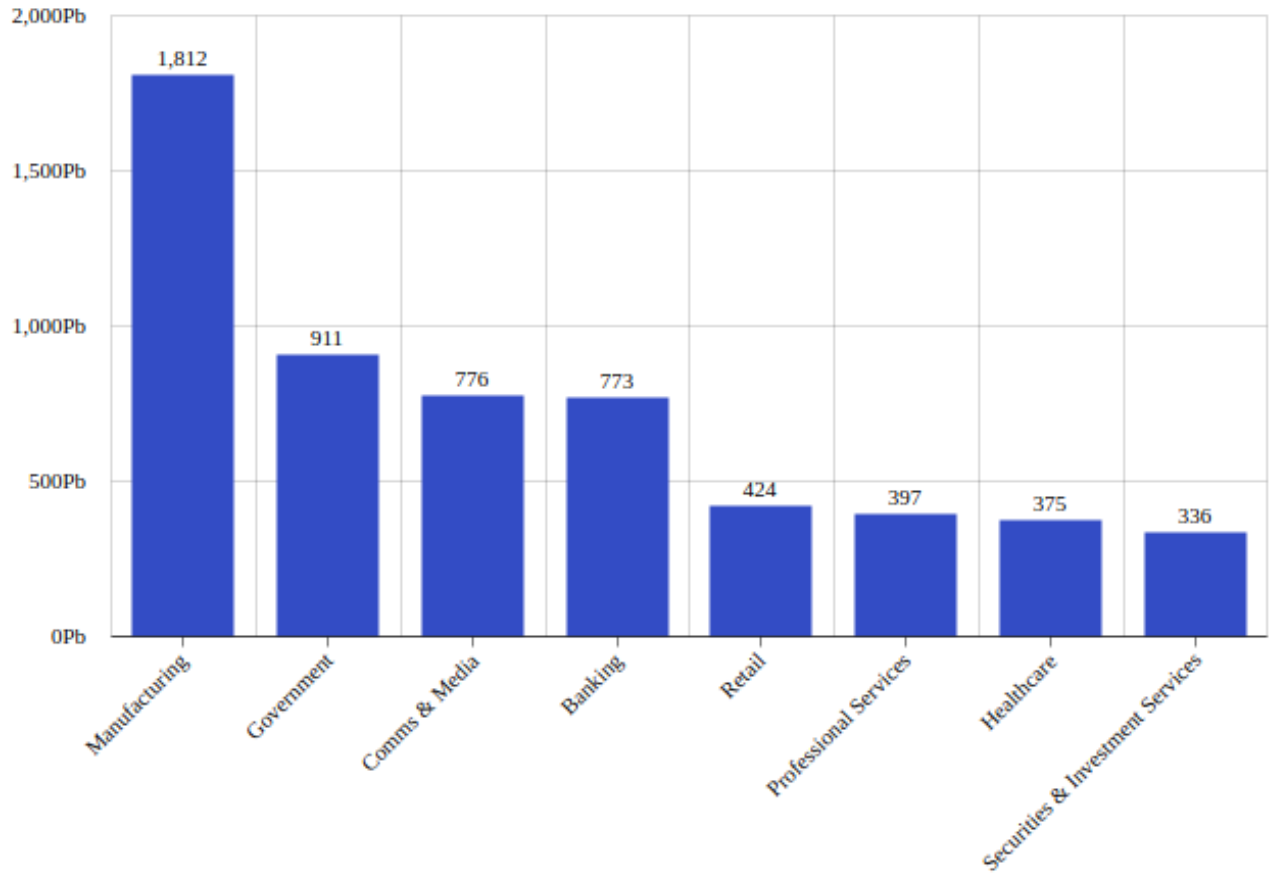
- Desenvolvimento de soluções de computação em nuvem para dados industriais. A computação em nuvem se tornou uma das ferramentas dominantes no armazenamento e processamento de grandes quantidades de dados, e atualmente fornece uma variedade de ferramentas que combinadas, habilitam a capacidade de armazenar, processar e analisar dados.

Espera-se com essas adaptações atingir um nivel de digitalização que permita desenvolver uma versão digital do produto, que acompanha todo o processo de desenvolvimento, cadeia de produção e distribuição ao consumidor final, sendo conhecido como *Digital Twin*. O objetivo é atingir o *Smart Manufacturing*, entendido como um conceito de concepção de fabrica com flexibilidade para atender as mudanças e necessidades dos clientes e do ambiente (Qi; Tao, 2018).

4.2 *Big Data* na indústria

Vários setores da economia são responsáveis por gerar grande quantidades de dados, que ajudaram a cunhar o termo *Big Data*, mas a indústria de manufatura é a que menos se beneficia dessa transformação. A figura 5 apresenta uma comparação dos dados gerados por diferentes setores.

Figura 5 – Comparação de dados gerados por diferentes setores, aproximadamente 2 pentabytes de dados industriais são armazenados a cada ano.



Fonte: McKinsey Global Institute.

O termo *Big Data* é comumente utilizado para referir-se a quantidade de dados exorbitante e não estruturado disponível graças ao advento da internet, dos smartphones e das redes sociais, mas há uma outra fonte de dados pouco aproveitada, segundo McKinsey Global Institute a indústria de manufatura é a maior coletora de dados, afirmando:

> Desde 2010, a indústria de manufatura coletou 2 mil pentabytes de dados potencialmente valiosos, mas descartou 99%.

Para aproveitar o potencial transformador que os dados podem agregar à indústria é necessário o desenvolvimento de ferramentas de captação, armazenamento e transformação de dados, auxiliando no processo de tomada de decisão. Qualquer modelo de produção que se apresente como indústria 4.0 deve considerar o aproveitamento dos dados para agregar valor no processo, sendo a maioria das propostas envolvendo o desenvolvimento de softwares, apontando assim para o processo de digitalização da indústria de manufatura como afirmado por McKinsey Global Institute.

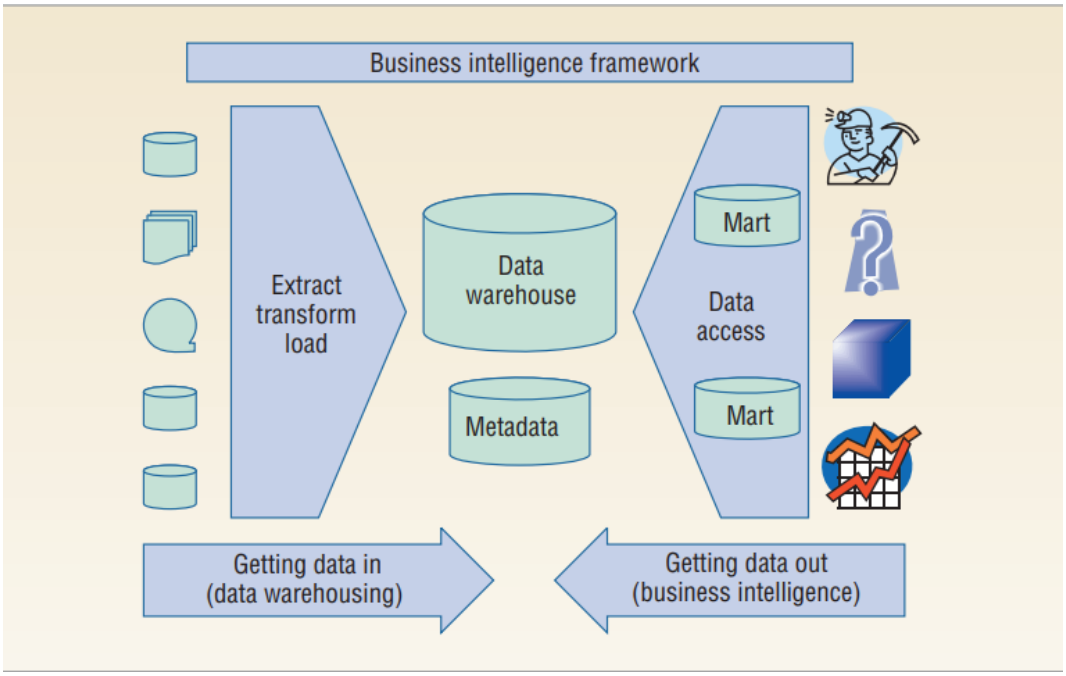
4.3 *Business Intelligence* (BI)

O termo BI é atribuído á Howard Dressner, no periodo em que trabalhava como analista na Gartner Group no inicio dos anos 1990 (Watson; Wixom, 2007) e desde então passou a ser utilizado para descrever aplicações analíticas.

Segundo (Watson; Wixom, 2007) BI é um processo que inclui duas atividades: Fazer dados entrarem e fazer dados saírem. A descrição mais detalhada afirma que um processo de BI tipico envolve a configuração de datacenters para armazenar grandes quantidades de dados estruturados, softwares de BI como Tableau, PowerBI, Qlink, Grafana, entre outros, possam executar manipulações sobre os dados, com o objetivo de gerar *Key Performance Indicator* (KPI) e *dashboards* (visualizações) possibilitando obter conhecimentos sobre empresas, mercados ou industriais e tomar decisões precisas fundamentada em dados (HOLSAPPLE; LEE-POST; PAKATH, 2014).

A figura 6 ilustra um processo tipico, observa-se no centro da figura o data warehouse, local onde os dados são armazenados. Os dados armazenados devem ser estruturados, o processo *Extract, Transform and Load* (ETL) devem ser executados sobre os dados não processados. Para obter conhecimento é necessário consultar a base de dados com um software de BI e desenvolver visualizações.

Figura 6 – Modelo tipico de um processo de BI.



Fonte: (Watson; Wixom, 2007).

As visualizações geradas comumente são obtidas a partir dos KPIs (Key Performance Indicator). Definidos como métricas de negócios customizadas, utilizadas para apresentar o atual status ou avaliar tendencias dentro de uma organização (NADA, 2010). Para atingir objetivos

uma empresa apresenta métricas de resultados esperados e compara com os obtidos, assim o KPI apresenta dois valores, o esperado e o real.

4.4 *Overall Equipment Effectiveness* (OEE)

Desenvolvida pelo *Japan Institute of Plant Maintenance* (JIPM) o índice OEE foi concebido para avaliar a eficiência de uma maquina industrial, com o objetivo de avaliar de forma quantitativa seu funcionamento, indicando se havia ou não necessidade de manutenção (NAKAJIMA, 1988), mas com o tempo esse o índice passou a ser usado em uma linha ou mesmo uma planta inteira.

O índice OEE é comumente usado como KPI em conjunto com ferramentas de gestão para funcionar com um indicador do sucesso do gerenciamento de planta (STAMATIS, 2010). Em essência o índice pode ser entendido como: (NAKAJIMA, 1988)

- Uma medida que identifica potencial de produção por equipamento
- Medida que rastreia as perdas
- Identifica janela de oportunidade

O índice pode ser aplicado desde uma área individual de trabalho, a um departamento ou planta inteira, independente da área os principais objetivos são:

- Aumentar produtividade
- Diminuir custo
- Aumentar a vida útil do equipamento

O cálculo OEE depende de outros três indicadores apresentados na Equação 4.1, cada um fornecendo uma informação diferente e combinadas fornecem o entendimento geral da planta, area ou departamento. Para calcular o OEE a equação fornecida por (STAMATIS, 2010) é utilizada.

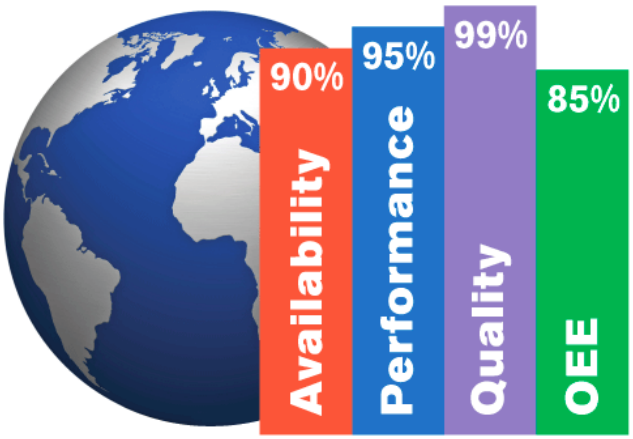
$$OEE = Disponibilidade * Performance * Qualidade \tag{4.1}$$

Onde disponibilidade, performance e qualidade são métricas que compõem o OEE, calculadas com métodos específicos.

Os três fatores variam de 0 a 1 ou de 0% até 100% possibilitando assim um valor máximo para o indice OEE de 100%. O valor médio OEE para as empresas é de 60%, e plantas

extremamente produtivas possuem um valor igual ou maior a 85% (NAKAJIMA, 1988). A figura 7 fornece os valores considerados de excelência para cada um dos fatores que compõem o OEE.

Figura 7 – Valores de excelência para OEE .



Fonte: World Class OEE.

4.4.1 Disponibilidade

A disponibilidade representa a quantidade de tempo de produção efetivo, pois durante a produção o total de tempo disponível para produzir não é totalmente aproveitado sendo comum ocorrerem paradas programadas e não programadas. (STAMATIS, 2010) As paradas programadas se referem a manutenção periódica, mudança de turno, horário de almoço ou qualquer outra parada prevista. Já as paradas não programadas ocorrem por decorrência de falhas, defeitos ou qualquer outra parada inesperada. O cálculo é feito segundo (NAKAJIMA, 1988).

$$Disponibilidade = \frac{TTP - TP}{TTP} \tag{4.2}$$

Sendo o *TTP* o tempo total de produção, *TP* o tempo parado.

A maioria dos autores considera como tempo parado apenas paradas não programadas, mas é comum algumas empresas utilizarem o tempo programado como parada também. No caso vamos apenas considerar como parada apenas o tempo não programado, sendo assim uma maquina teria como *TTP* numa empresa que apresenta dois turnos de 8 horas cada com 1h de almoço e 1h para manutenção o valor somado dos 2 turnos (16h) excluindo o horário de almoço dos dois turnos (2h) e também as paradas para manutenção (2h), resultando no tempo total de produção de 12h como exemplo.

O *TP* é qualquer parada que esteja fora da prevista, ou seja não é parada para almoço e nem para manutenção, por consequência no modelo ideal o tempo parado em valor nulo e a disponibilidade é de 100%.

A disponibilidade é uma das medidas mais fundamentais para avaliar o desempenho geral de uma fabrica, sendo a acuracidade dessa informação de vital importância para o acompanhamento realista da planta. (NAKAJIMA, 1988).

> Algumas empresas não armazenam paradas que não excedam trinta minutos. Esta é uma prática não saudável. O tempo de operação baseado em dados tão rudes onde paradas por falhas de dez ou vinte minutos não são registradas pode levar a um gerenciamento ineficiente

Fonte: (NAKAJIMA, 1988)

4.4.2 Performance

Durante o desenvolvimento de uma linha de montagem, ou de uma máquina de produção é necessário definir o **Tempo de Ciclo Padrão**, que pode ter nomenclaturas variadas mais representa a velocidade de produção, ou o tempo no qual uma peça ou conjunto de peças consome dentro de um processo ou máquia. Idealmente qualquer processo deve consumir sempre a quantidade de tempo esperada, mas na pratica o tempo médio real pode variar. Assim a equação para calcular a performance sugerida por (AMES et al., 1995):

$$Performance = taxP * TOL \tag{4.3}$$

Sendo *taxP* a Taxa de Performance e *TOL* o tempo de operação liquida om obtidos segundo as equações 4.4 e 4.5

$$taxP = \frac{TCP}{TCR} \tag{4.4}$$

Onde *TCP* é o tempo ciclo padrão e *TCR* o tempo de ciclo real

A variável *TOL* da equação 4.5 é tempo de operação liquida, é utilizado para calcular a performance, e fornece um valor que avalia o quanto do tempo utilizado na linha foi realmente aproveitado para produzir, sendo obtido segundo a equação:

$$TOL = \frac{Quant.\ Peças * TCR}{TTP} \tag{4.5}$$

Multiplicando a equação 4.3 e 4.4 obtem-se a performance como:

$$Performance = \frac{Quant.Peças * TCP}{TTP} \tag{4.6}$$

4.4.3 Qualidade

A qualidade é o mais simples de calcular e bem direto, obtido como a subtração entre o total de peças processadas e o total de peças rejeitadas dividoido pelo total de peças processadas, como sugerido por (AMES et al., 1995)

$$Qualidade = \frac{TPP - PR}{TPP} \tag{4.7}$$

Onde *TPP* é o total de peças processadas e *PR* é o total de peças rejeitadas.

4.5 Engenharia de Dados com Apache Spark

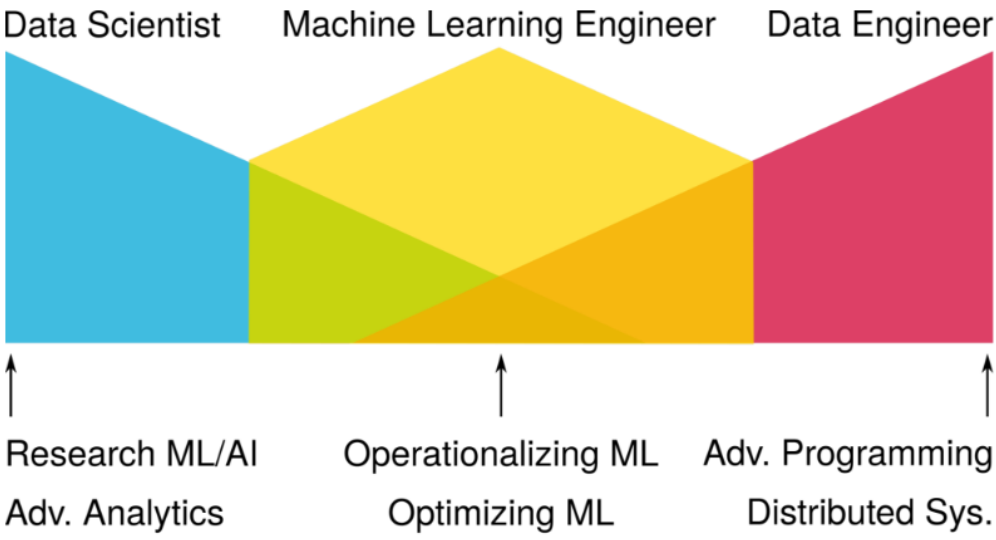
4.5.1 Definição de Engenharia de Dados

Muitas vezes entendido como análogo a Ciência de Dados, na pratica é uma profissão totalmente diferente. Uma definição apresentada for James Furbush define que a engenharia de dados deve construir e manter o pipeline de dados de uma organização, limpando e organizando dados obtidos a partir de uma fonte não estruturada

É comum que inicialmente a fonte de dados esteja desestruturada tornando necessário um conjunto de manipulações para torna-los utilizáveis. Comumente esta é regra, então o Engenheiro de Dados é o profissional que desenvolve algoritmos para fazer as manipulações necessárias tornando possível que um Cientista de Dados ou um Analista BI extraia informações.

A imagem 8 fornece um comparativo das duas profissões, onde podemos notar que há uma sobreposição de competências em comum, mas outras são especificas de cada área.

Figura 8 – Comparativo entre Engenharia de Dados e Ciência de Dados.



Fonte: *Big Data* Institute.

Entre as duas profissões há um meio termo, conhecido como Engenheiro de Aprendizagem de Máquina. Sendo o responsável por transformar os soluções do Cientista de dados em produto, aprimorando e otimizando os softwares desenvolvidos para torna-los funcionais em escala. Conclui-se que a principal atividade da engenharia de dados é o desenvolvimento de algoritmos que processem, de forma eficiente, conjuntos de dados para permitir o trabalho de outros profissionais.

4.6 Apresentação do Apache Spark

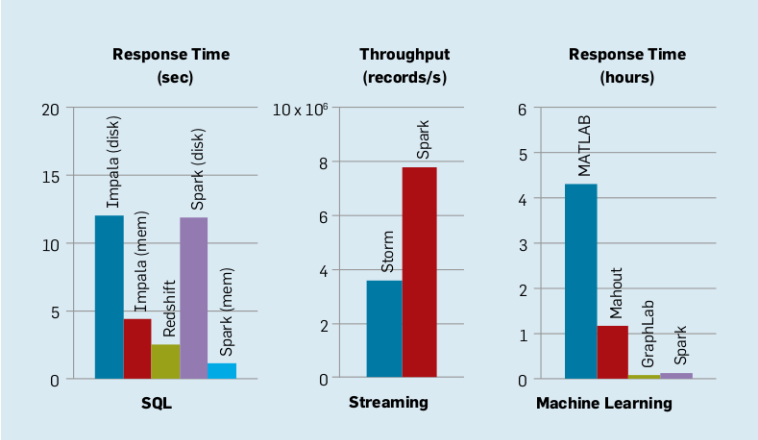
Spark é uma ferramenta de programação com código aberto que permite computação distribuída em *cluster*s unida com computação em tempo real. Foi inicialmente desenvolvido por Matei Zaharia (ZAHARIA et al., 2010) em sua tese de doutorado na *Berkeley AMPLab* em 2009, sendo posteriormente entregue a Apache Software Foundation.

O surgimento do *Big Data* impulsionou o desenvolvimento de ferramentas para processar grandes volumes de dados, e para permitir a escalabilidade, o modelo de computação distribuída (clusterização) foi adotada, pois permite aumentar o desempenho do sistema sem precisar troca-lo, apenas adicionando mais computadores e conectando-os a rede (ZAHARIA et al., 2010).

Para manipular o grande volume de dados disponíveis no começo da década muitas ferramentas distintas foram desenvolvidas, porém a diversidade de fonte de dados e variedade de estruturas tornou a manipulação um trabalho extensivo e custoso. Assim o grupo da Universidade de Berkeley na Califórnia iniciou um projeto que unificaria muitas funcionalidade, e que funcionaria com computação em paralelo.

A figura 4.6 fornece uma comparação do tempo de resposta entre as principais ferramentas para cada aplicação. O Spark fornece velocidades até dez vezes maiores para tratamento de consultas SQL (ZAHARIA et al., 2010).

Figura 9 – Comparação da performance do Spark em relação a outras ferramentas comumente utilizadas para consultas SQL, Streaming e ML.



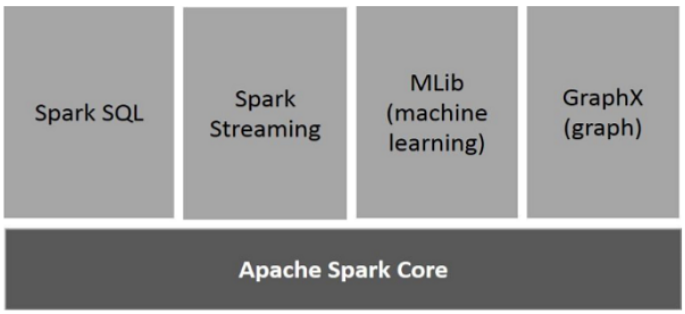
Fonte: (ZAHARIA et al., 2016).

As principais vantagens do Spark são listadas abaixo: (ZAHARIA et al., 2016)

- **Velocidade** - Velocidades de até 100 vezes em memória se comparado com processamento de dados em disco.
- **Multiplas linguagens** - Spark fornece uma *Application Programming Interface* (API) em Java, Scala ou Python, permitindo a escrita de algoritmos em qualquer linguagem suportada.
- **Analise Avançada** - Permite a utilização de algoritmos SQL, dados em tempo real, tem sua própria ferramenta de Machine Learning (ML), e algoritmos de grafos.

O Apache Spark é composto por quatro componentes principais que fornece capacidade de desenvolver algoritmos com diferentes abordagens, além de análise com sua própria ferramenta ML. Há também o Spark Core que integra as quatro soluções. A figura 10 apresenta a estrutura do Spark, e como os quatro componentes estão conectados ao Apache Spark Core, que permite manipular um mesmo conjunto de dados utilizando módulos do Spark distintos.

Figura 10 – Componentes do Apache Spark.



Fonte: (ZAHARIA et al., 2016).

Cada componente fornece soluções completas, e serão detalhadas a seguir:

- **Apache Spark Core** - É o executor general em que todas as funcionalidades são construídas. Fornece a capacidade de processamento em memoria e mantém os registros das posições de cada componente dos dados RDDs distribuídos pelo *cluster*
- **Spark SQL** - Componente montado sobre Spark Core, fornece uma abstração alternativa camada SchemaRDD, que permite tratar dados estruturados e não estruturados com consultas SQL.
- **Spark Streaming** - Utilizando das funcionalidades mais rápidas do Spark Core para fazer análises em tempo real, transformando uma fonte de dados continua em pequenos RDDs para fazer os tratamentos e análises em tempo real.

- **MLib** - É uma biblioteca de ML que processa os dados em memória permitindo assim obter resultados muito mais rapidamente se comparado a outras ferramentas.

- **GraphX** - É uma ferramenta que fornece uma API para computação em grafos, o usuário pode criar seus próprios modelos. Fornece também um otimizador de resultados para permitir velocidades de processamento mais altas.

A maior vantagem de usar o Spark para processamento de dados é sua capacidade de fazer computação em paralelo, permitindo assim enorme potencial de escalabilidade, já que para aprimorar o sistema de processamento basta adicionar mais computadores sem precisar trocar o sistema inteiro por um mais potente.

4.6.1 RDDs

Spark fornece duas abstrações para processamento em paralelo, uma chamada RDD abreviação para *Resilient Distributed Datasets* e outra de operações paralelas. RDD pode ser definido como uma abstração do conjunto de dados que tem a característica de ser distribuído por todo o *cluster* e por isso é resiliente pois a distribuição garante redundância no caso de alguma das máquinas falhar.

Qualquer conjunto de dados alimentado em uma aplicação Spark é transformado em RDDs, quebrando os dados originais em pequenos conjuntos que são processados em paralelo por todo o *cluster*. Outra característica importante é que RDDs são *read-only*, não podendo ser alterados depois de criados, apenas lidos.

Uma das características do Spark que o diferencia de outras ferramentas é o processamento em memória, mantendo os dados salvos na memória volátil do computador (RAM). Essa característica o diferencia do Hadoop MapReduce que entre cada transformação precisava salvar os dados na memória permanente.

4.6.2 Processamento em Paralelo

A capacidade de executar processamento em paralelo é um dos grandes diferencias, pois permite que uma operação feita em um enorme conjunto de dados seja distribuída em dezenas e até mesmo milhares de computadores viabilizando o rápido processamento de grande quantidades de dados

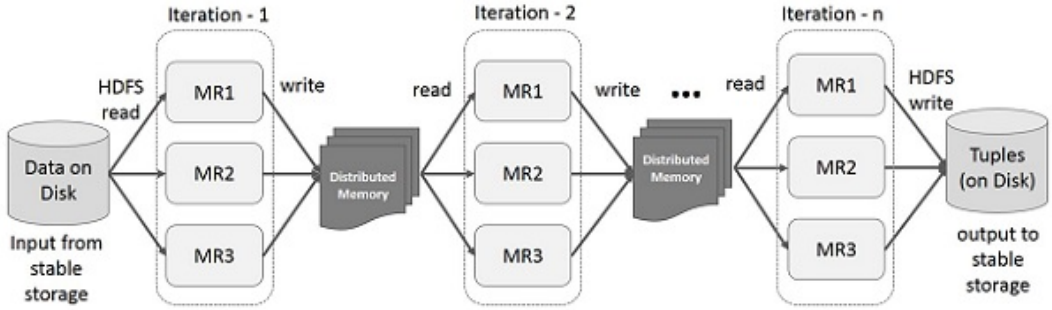
Uma outra funcionalidade é a capacidade de criar variáveis distribuídas por todas as máquinas do *cluster*, reduzindo a necessidade de comunicação entre maquinas. Isto torna algumas transformações, como agrupamentos de dados, união de dados, entre outros mais eficientes.

Outra característica fundamental é que Spark utiliza *lazy evaluation* na hora de processar instruções do usuário. Ao fornecer um conjunto de instruções (transformações sobre um conjunto

de dados) o Spark não as executa linha por linha, e sim regrista todas as operações e apenas ao receber uma **ação** o algoritimo avalia, utilizando teoria de grafos, qual o caminho mais otimizado para executar toda aquela lista de transformações. Na pratica isso significa que todo o algoritmo escrito para Spark deve terminar com ao menos uma ação, caso o contrário o conjunto de transformações não é materializado.

Para ilustrar como o Spark processa informação a figura 11 serve como referência:

Figura 11 – Representação da estrutura de processamento do Spark, vemos que entre cada iteração o resultado é armazenado na memória volátil, e apenas no final com comando *write* o resultado é salvo em disco.



Fonte: Tutorials Point

4.7 Computação em nuvem

A Computação em Nuvem ou *Cloud Computing* é uma das tecnologias que emergiram no inicio da segunda década do século XXI impulsionada pela disseminação da internet, pelo surgimento de novas tecnologias em TI. A computação em nuvem tem a capacidade de transformar o desenvolvimento de software, tornando-o mais acessível a empresas de TI, mas também a empresas de outros seguimentos, mas que podem se beneficiar do poder de processamento de informações em nuvem.

Anterior ao advento da computação em nuvem, qualquer empresa com interesse de aumentar sua capacidade de processar e armazenar informações, bem como desenvolver ou customizar aplicações para uso próprio precisava investir em servidores para armazenar dados, grandes equipes de TI para desenvolver e integrar as aplicações da empresa nesses servidores, entre outras necessidades.

Grandes empresas de TI, como Google, Microsoft e Amazon perceberam que é possível fornecer um conjunto de tecnologias de TI como serviço disponibilizado por uma plataforma web ou um software próprio, que auxilia no gerenciamento das aplicações customizadas desenvolvidas por seus clientes, estendendo também para armazenamento de dados, entre outros serviços.

Existe uma grande confusão sobre o que é exatamente computação em nuvem, pois muitos dos serviços hoje entendidos como "Computação em Nuvem"já eram fornecidos por

muitas empresas de TI. O CEO da Oracle Larry Ellison demostrou sua frustração com a seguinte frase.

> O interessante da computação em nuvem é que foi definida para incluir tudo o que já fazemos ... Não entendo o que faríamos de maneira diferente à luz da computação em nuvem, além de alterar a redação de alguns de nossos artigos publicitários.

Adaptado de: The ACM Digital Library

Uma definição focada no impacto que essa tecnologia tem sobre o ambiente de negócios é apresentada como uma modelo de serviço de tecnologias da informação, agregando hardware e software, entregues por demanda à clientes em uma rede de serviços. Os recursos requisitados devem ser dinamicamente escaláveis, rapidamente disponibilizados e requerindo minima interferecia do provedor de serviço (MARSTON et al., 2011).

Com objetivo de esclarecer a NIST (*National Institute of Standards and Tecnology*) desenvolveu uma definição de Computação em Nuvem através de um documento de 2011 que apresenta a definição, e também características essenciais, como a maneira com o qual os serviços devem ser distribuídos, como deve ser a interação do usuário, quais capacidades e permissões são disponibilizadas e quais serviços podem ser disponibilizados.

Computação em nuvem permite por meio da internet, de forma onipresente, conveniente e acessível por demanda, compartilhar um conjunto de recursos computacionais configuráveis (redes, servidores, armazenamento, aplicações e serviços) que podem ser providenciados rapidamente e lançados com mínimo esforço gerencial ou interação com provedor de serviço. Modelo composto por **cinco características essências**, **três modelos de serviços** e **quatro modelos de disponibilização** (MELL; GRANCE, 2010).

4.7.1 Características essenciais

Cada ponto apresentado acima será descrito , conforme apresentado por (MELL; GRANCE, 2010)

- **Serviços sob demanda**. O próprio consumo define o quanto será consumido do armazenamento ou do tempo de servidor, conforme a necessidade sem qualquer interação. Na prática significa escalonamento automático.
- **Amplo acesso a rede**. Capacidade de acesso amplo a plataforma que controla a nuvem através de uma gama de dispositivos, como computadores, laptops, telefones celulares e estações de trabalho.
- **Agrupamento de recursos**. Os serviços em nuvem devem disponibilizar uma gama de recursos que forneçam ao usuário a capacidade de definir quais recursos consumir. São

definidos para servirem a vários clientes, assim um *datacenter* poderá armazenar dados de inúmeros clientes, estes que definem quais recursos acessar. Geralmente o consumidor não deve ter conhecimento sobre a localização física dos recursos, mas deve ter a capacidade de definir a localização sob um nível de abstração, como exemplo: país, estado ou *datacenter*.

- **Rápida elasticidade**. As capacidades dos serviços são dinâmicas, e podem ser rapidamente providenciadas ou liberadas, podendo ser automaticamente escaladas baseado em demanda, assim o consumidor enxerga a capacidade dos serviços como ilimitada.

- **Serviços sob medida**. Sistemas em nuvem podem automaticamente controlar e otimizar seus recursos, verificando a demanda em tempo real e readequando os sistemas para providencias apenas os recursos necessários. Para isso o consumo de recursos deve ser monitorado, controlado e reportado com transparência para o providenciador e consumidor.

4.7.2 Modelos de Serviços

Os modelos de serviços que um provedor de computação em nuvem deve fornecer são: (MELL; GRANCE, 2010)

- **Software as a Service (SaaS)** Traduzido como Software como Serviço, o provedor de serviço deve disponibilizar um conjunto de aplicações que podem ser usados por vários dispositivos, normalmente através do navegador web.

- **Platform as a Service (PaaS)** Traduzido como Plataforma como Serviço, o provedor de serviços em nuvem deve fornecer a capacidade ao consumidor de desenvolver e inserir dentro do sistema da nuvem suas próprias aplicações criadas usando alguma linguagem de programação.

- **Infrastructure as a Service (IaaS)** Traduzido como Infraestrutura como Serviço o provedor de serviços em nuvem deve fornecer a capacidade de armazenar e processar dados, entre outros serviços de computação essenciais, como *Virtual Machine* (VM) no qual o consumidor tem a capacidade de rodar softwares arbitrários. O provedor também deve fornecer a segurança das informações armazenadas em seus datacenters.

4

4.7.3 Modelos de disponibilização

Os serviços e tecnologias de computação em nuvem podem ser disponibilizados de formas diferentes, seguindo a definição da NIST há 4 formas (MELL; GRANCE, 2010).

- **Nuvem privada** Uma infraestrutura de nuvem é providenciada de forma exclusiva por uma organização para ser utilizado por múltiplos consumidores. Pode ser operado, gerenciado

ou propriedade da organização, uma empresa terceira, ou uma combinação de ambos, e pode existir em alguma outra premissa.

- **Nuvem comunitária** A infraestrutura em nuvem é providenciada para uso exclusivo de uma comunidade de consumidores por uma organização que compartilha interesses (missão, requerimentos de segurança, entre outros). Pode ser operado, gerenciado ou propriedade da organização, uma empresa terceira, ou uma combinação de ambos, e pode existir em alguma outra premissa.

- **Nuvem publica** A infraestrutura da nuvem é aberta ao publico para ser usado sobre licença publica (GNU). Pode ser operado, gerenciado ou propriedade da organização, uma empresa terceira, ou uma combinação de ambos, e pode existir em alguma outra premissa.

- **Nuvem hibrida** A infraestrutura em nuvem é composta de duas ou mais infraestruturas em nuvem distintas (privada, comunitária ou publica) que permanecem como entidades únicas, mas são integradas por alguma tecnologia padrão ou proprietária, que permite portabilidade de dados ou aplicações.

4.7.4 Vantagens da computação em nuvem

A coleta de dados da indústria de manufatura não é devidamente aproveitada, pois a indústria não tem a infraestrutura necessária para armazenar e processar os dados. Para habilitar a capacidade de obter valor dos dados gerados é necessário uma tecnologia de computação que seja rapidamente escalável, forneça capacidade de customização e seja economicamente acessível. Essas são características definidas como vantagens da computação em nuvem, segundo (MARSTON et al., 2011) são:

- Redução dramática de custo para pequenas empresas, que passam a ter acesso a grande poder computacional.

- Habilita acesso a recursos de hardware quase que imediatamente, sem nenhum investimento de capital graças a flexibilidade dos recursos computacionais na nuvem o cliente paga pelo que usa.

- Reduz as barreiras para inovações em TI, permitindo ou mesmo viabilizando o surgimento de novas empresas.

- Permite a empresas escalarem seus serviços de forma rápida e precisa.

- Permite o surgimento de novos produtos e serviços que antes não eram possíveis.

5 Métodos Computacionais

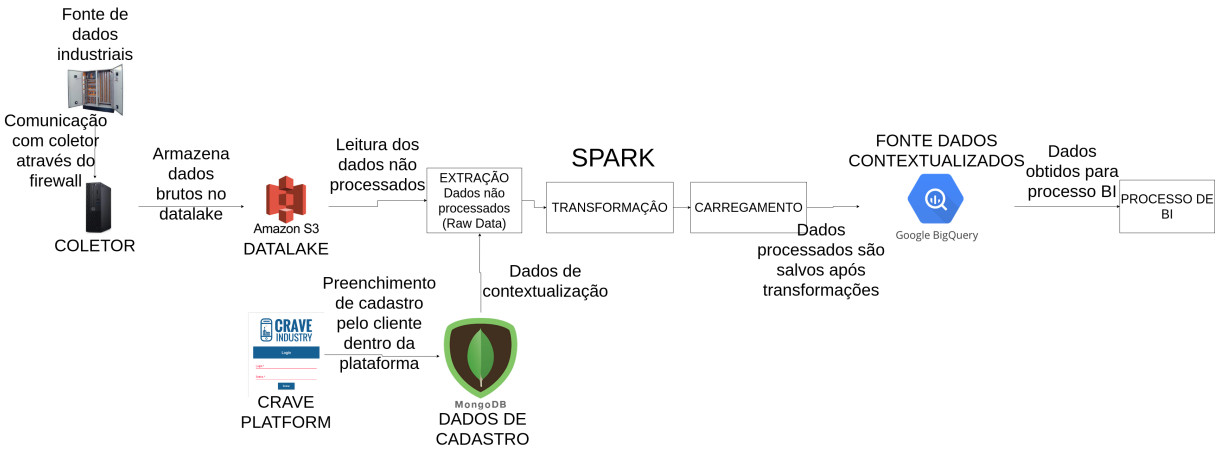
5.1 Máquina e Parâmetros de Programação

Os algoritmos para tratamento de dados foram desenvolvidos em um laptop convencional Dell, com 16Gb de memória RAM e um processador Intel i7 sexta geração. O ambiente de desenvolvimento escolhido foi o PyCharm Profissional, uma IDE da Jetbrains pois permite o desenvolvimento de softwares em linguagem Python, consultas SQL e controle de versão com Git. Para utilizar Apache Spark com linguagem python é necessário instalar a biblioteca PySpark, juntamente com outras bibliotecas usuais de Python para manipulação de dados.

5.2 Resumo do processo

O processo de transformação de dados que o algoritmo deve efetuar é longo e complexo, com uma lista de transformações, mas que podem ser resumidas em três passos: Extração, Transformação e carregamento, ou ETL (Extraction Transformation Load). A figura 12 apresenta o escopo geral, mostrando a fonte de dados para extração até o final no qual dados serão armazenados após tratamento.

Figura 12 – Resumo do processo de tratamento de dados.



Fonte: Autor

O algoritmo Spark deve buscar os dados não processados do Amazon S3, assim como os dados de cadastro do MongoDB. Os dados de cadastro são preenchidos pelo cliente na Crave Platform e armazenados no MongoDB, e servirão para enriquecer os dados não processados com informações sobre turno, tempo de ciclo padrão, paradas programadas, entre outras informações que podem variar dependendo do cliente.

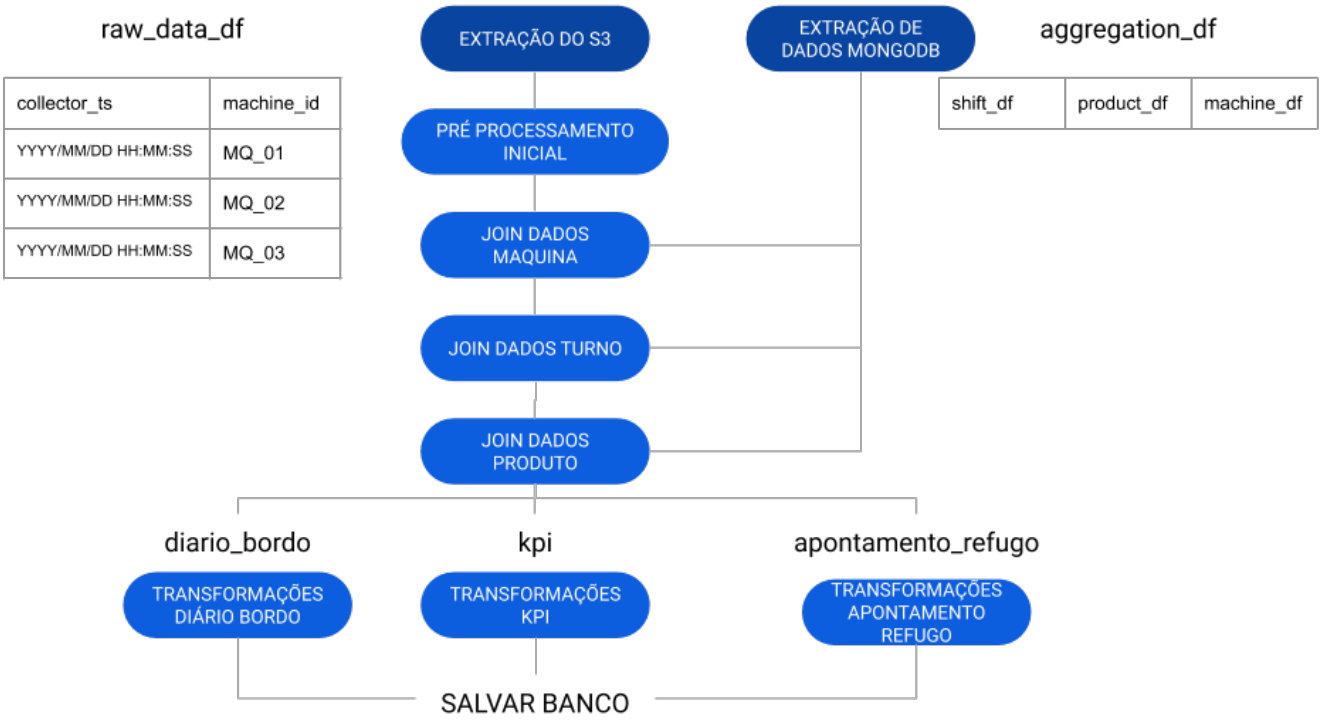
5.3 Processamento ETL

Para exemplificar o processo ETL apresentado utilizou-se dados do cliente que fabrica peças automotivas. Há três algoritmos Spark que rodam de forma independente, mas apresentam transformações em comum e também são utilizados para os KPIs e visualizações.

O processo ETL é desenvolvido nas seções 6.1.1 à 6.1.5, começando com a configuração do ambiente, seguindo por todas as transformações em comum, e a segunda parte as transformações especificas de cada processo.

A figura 13 resume os passos do processo ETL partindo da extração dos dados, transformações e manipulações até a ponta final onde os dados são salvos no banco de dados. Os detalhes de cada transformação estão disponíveis nas seções referentes ao processo ETL.

Figura 13 – Processo ETL a partir da extração dos dados, passando pelas transformações até o salvamento no banco de dados Os dados são segmentados por máquina e todos devem conter um *collector timestamp* que representa o momento de coleta do dado.



Fonte: Autor

6 Apresentação dos Resultados

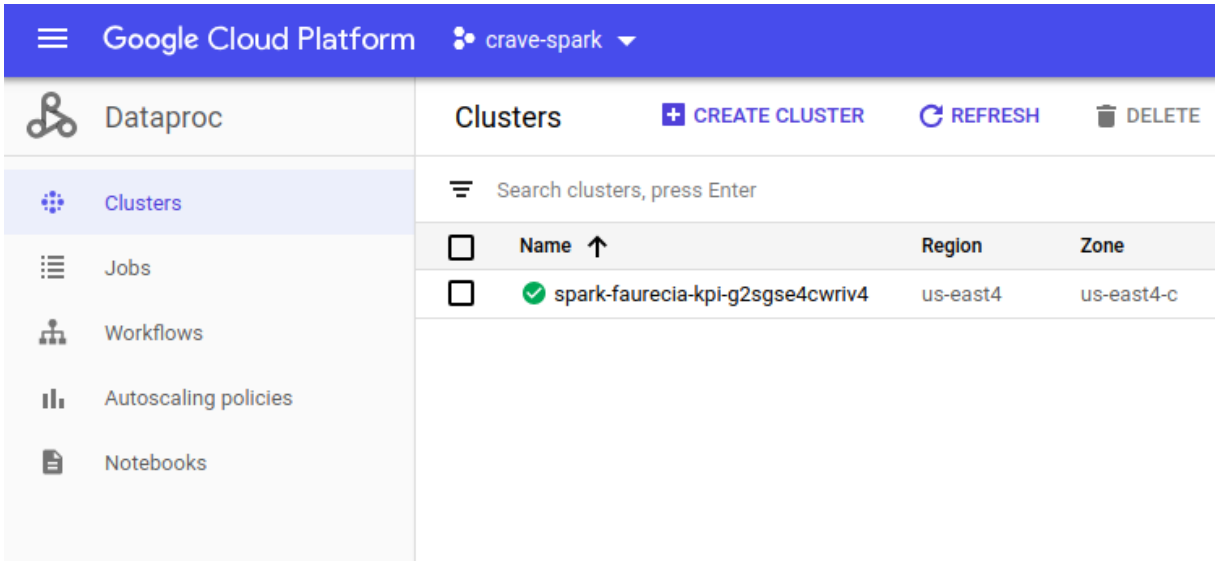
6.1 Desenvolvimento do processo ETL

6.1.1 Configuração do ambiente

O ambiente é configurado no *Google Cloud Platform*, uma ferramenta online acessível por qualquer navegador de internet onde é possível configurar o ambiente de computação em nuvem, ajustar todas as ferramentas disponíveis, encontrar a documentação das ferramentas, entre outras funcionalidades.

O *cluster* é criado com a ferramenta Dataproc, que permite gerenciar *cluster* e também criar VMs para aumentar o poder de processamento do *cluster*. A figura 14 apresenta a pagina de gerenciamento de *cluster*s para o projeto crave-spark.

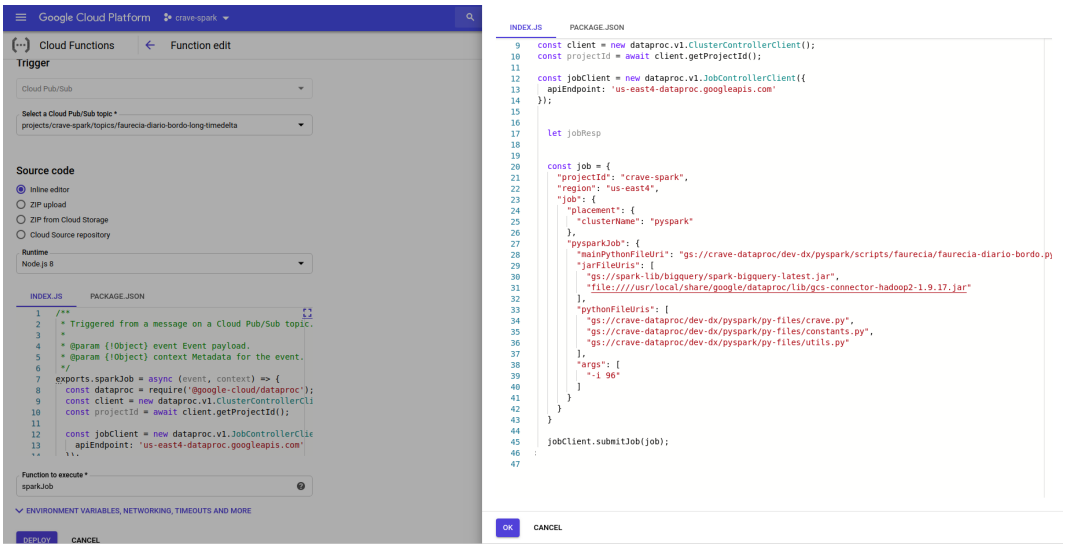
Figura 14 – Ferramenta Dataproc do Google Cloud mostrando um *cluster* criado com nome spark-faurecia-kpi.



Fonte: Autor

Com o *cluster* criado é necessário adicionar uma função no *Cloud Function*, que executa o código escrito em Pyspark dentro do *cluster*. Para cada algoritmo de Spark criado deve haver uma *function*. A figura 15 mostra um exemplo de *function* criada utilizando javascript como linguagem de programação.

Figura 15 – Criação de uma Cloud Function.

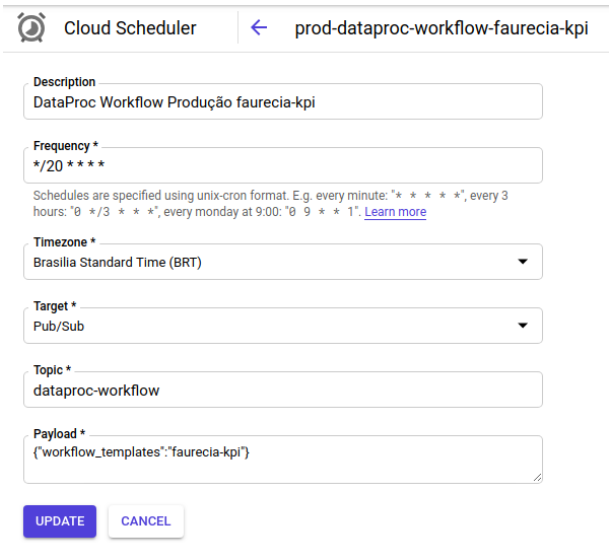


Fonte: Autor

A imagem é divida em dois trechos, a esquerda que apresenta as opções para configurar a *Cloud Function*, e a direita que apresenta o código que será executado dentro do *cluster* onde é informado o projeto, a região, o nome do *cluster* e também a localização do código Spark como uma *Uniform Resource Identifier* (URI).

O próximo passo é configurar um *Cloud Scheduler*, que define a frequência com o qual o algoritmo do Spark é chamado para processar os dados. A figura 16 apresenta a pagina onde se configura um *Cloud Scheduler*.

Figura 16 – Criação de uma Cloud Scheduler.



Fonte: Autor

6.1.2 Extração

Para fazer a extração dos dados o Pyspark fornece um conector para o Amazon S3 e também para o MongoDB. A figura 17 apresenta o trecho de código para coletar dados do Amazon S3.

Figura 17 – Código para leitura de dados não processados do Amazon S3.

```
# LOAD DATA FROM S3
raw_data_df = load_from_s3(s3_bucket=cc.S3A_FAUR_BUCKET,
                           s3_machine_preffix=TEST_MACHINE_SUFFIX,
                           job_start_utc=job_start_utc)
```

Fonte: Autor

A função que faz a leitura dos dados recebe três parâmetros, o primeiro é o endereço de acesso ao banco, o segundo é um sufixo que identifica o cliente e o ultimo é o inicio da execução do código.

Os dados de cadastros obtidos do MongoDB são obtidos pela figura 18 de código.

Figura 18 – Código para leitura de dados de contextualização do MongoDB.

```
# LOAD DIM TABLE FROM MONGO DB
mongo_source = spark.read.format('mongo') \
    .option('uri', mongo_coll_conn) \
    .option("pipeline", complex_pipeline) \
    .load()
```

Fonte: Autor

A função que executa a leitura de dados recebe um parâmetro, porém pode receber outros parâmetros dentro do método opções, no caso há duas opções. A primeira define o endereço do banco e a seguida é uma consulta de dados onde é possível definir quais dados serão extraídos.

A tabela gerada a partir dessa leitura é uma mistura de informações de cadastro, então é necessário separar os dados de acordo com as informações distintas que serão utilizadas para enriquecer os dados não processados.

Figura 19 – Tabelas de dados a partir do MongoDB.

```
# LOAD DIMENSION DATA (FOR JOINS)
machines_df = spark.sql(
    'select distinct machine_id, area_name_code, area_name, factory_name, minor_downtime from mongo')

shifts_df = spark.sql(
    'select distinct factory_name, weekDay, shift_start_hour, shift_end_hour, is_production_shift, shift_name from mongo')

product_df = spark.sql(
    'select distinct product_code, standard_cycle_seconds, product_work_content, area_name_code from mongo')
```

Fonte: Autor

As três tabelas geradas representam respectivamente, informações de máquinas, como nome de área, parada mínima, entre outras. A segunda tabela representa os turnos horário de inicio e fim, e por fim a tabela de produto que fornece as informações do produto.

6.1.3 Transformação

Após a extração os dados é necessário fazer as transformações e contextualização. Inicialmente as três tabelas de dados de contextualização são adicionadas a tabela de dados não processados. Os trechos de código na figura 20 apresentam a função que executa a contextualização.

Figura 20 – Contextualização de dados de máquina.

```
# JOIN MACHINE INFO
stage_df = ci.crave_join(
    spark_session=spark,
    join_metadata=kpi_pipeline['join_machine_dimension'],
    left_df=stage_df,
    right_df=machines_df)
```

Fonte: Autor

Figura 21 – Contextualização de dados de turno.

```
# JOIN SHIFT INFO
stage_df = ci.crave_join(
    spark_session=spark,
    join_metadata=kpi_pipeline['join_shift'],
    left_df=stage_df,
    right_df=shift_df)
```

Fonte: Autor

Figura 22 – Contextualização de dados de produto.

```
# JOIN PRODUCT INFO
stage_df = ci.crave_join(
    spark_session=spark,
    join_metadata=kpi_pipeline['join_area_id'],
    left_df=stage_df,
    right_df=product_df)
```

Fonte: Autor

A função que adiciona os dados de cadastro recebe quatro parâmetros, o primeiro fornecendo a sessão do atual do Spark, o segundo recebe os nomes das colunas que serão processadas, e os dois últimos referem-se a a tabela de dados não processados e a tabela de cadastro. Cada tabela de cadastro deve adicionar colunas novas a tabela de dados.

Para definir o status atual da máquina, que pode variar de "Produzindo", "Parado"ou "Perda de Comunicação"é necessário fazer algumas transformações, a imagem 23 apresenta o trecho de código que executa as transformações necessárias para obter o status da máquina.

Figura 23 – Processamento de informações de status.

```
# CREATE MAX_STD_CYCLE DATAFRAME GROUPED BY MACHINE
max_cycle = collect_max_std_cycle(stage_df, 'machine_id')

# CREATE ELLAPSED_TIME_AVG COLUMN GROUPED BY MACHINE
ellapsed_time = collect_elapsed_time(stage_df, 'COLLECTOR_TS', 'machine_id')

# CREATE OFFSET COLUMN ON DATAFRAME
stage_df = set_offset(
    max_standard_cycle_df=max_cycle,
    ellapsed_time_avg_df=ellapsed_time,
    dataframe=stage_df)

# LEAD FUNCTION BASED ON OFFSET COLUMN
stage_df = offset_lead(
    df=stage_df,
    machines=distinct_machines)

# CREATE STATUS COLUMN
stage_df = spark.sql(f'SELECT *, \
    {build_select_expr(kpi_pipeline["create_status_column"])} \
    FROM stage WHERE')
```

Fonte: Autor

O primeiro trecho apresenta o cálculo do ciclo padrão máximo por máquina, valor necessário para cálculo do OEE. Cada produto processado tem seu próprio ciclo padrão mas é feito uma aproximação com base no caso em que o valor é mais alto.

O segundo trecho de código verifica o tempo decorrido entre as linhas, que pode variar dependendo da coleta de dados, sendo no caso a cada 1 segundo.

Por fim uma função chamada *lead* verifica se houve incremento do contador de produção. Essa verificação é feita linha a linha, comparando a linha atual com a linha que está um ciclo padrão a frente. Por exemplo se o ciclo padrão é de 100 segundos, a linha atual deve ser comparada com todas as linha que até 100 segundos a frente e deve encontrar um incremento no contador de produção, se esse incremento não ocorrer o status é definido como "Parado"mas a busca continua até encontrar o próximo incremento, caso a busca continua até um valor de tolerância (300 segundos) e o incremento não ocorrer o status da máquina é definido como "Perda de Comunicação", caso ocorra um incremento de produção o status é definido como "Produzindo".

6.1.4 Transformações especificas

As tabelas complementares, diário de bordo e apontamento de refugo armazenam informações sobre a duração das paradas e a contagem de produção respectivamente.

A figura 24 apresenta o trecho de código onde o tempo de parada é obtido filtrando os dados onde o *status* é "Parado"ou *substatus* é "Perda de Comunicação. O *substatus* serve para registrar *microparadas*, que são paradas tão curtas que o status é considerado como produzindo, normalmente são paradas para tirar ou colocar uma peça dentro da máquina. No final o filtro sobre

end-time (tempo final) é necessário pois o algoritmo salva apenas dados de paradas concluídas, onde o tempo parado pode ser calculado.

Figura 24 – Código para calcular a duração de parada.

```
# DOWNTIME VALUES
stage_df = spark.sql(
    f'SELECT * FROM stage WHERE sub_status = "Parado" OR sub_status = "Perda_Comunicação"')

stage_df = build_query(stage_df, 'stage', build_select_expr(apont_parada["diario_bordo"]))

stage_df = spark.sql(f'SELECT * FROM stage WHERE end_time IS NOT NULL')
```

Fonte: Autor

O trecho de código na figura 25 é utilizado para calcular a produção da linha agregado por hora, somando o incrementador de produção em intervalos de 1h. Há um filtro para evitar o salvamento de linhas onde o incrementador de produção é 0.

Figura 25 – Código a contagem de produção em intervalos de uma hora.

```
    stage_df = spark.sql(
        f"""select area_id, area_name_code, MACHINE_ID, model,
        timestamp_hour, sum(incr_production_counter) as sum_counter
        from stage
        group by area_id, MACHINE_ID, area_name_code, model, timestamp_hour""")

    stage_df = spark.sql(
        f'select * from stage where sum_counter > 0')
```

Fonte: Autor

A coluna *timestamp-hour* é o tempo de coleta truncado em horas, permitindo assim somar todos os incrementador de produção que correspondentes á hora especifica.

6.1.5 Carregamento

Para enviar os dados ao destino final é necessário garantir que o modelo esquemático dos dados a serem salvos é idêntico ao banco de dados. Para isso o trecho de código 26 é utilizado.

Figura 26 – Montar o esquema de dados idêntico ao destino final.

```
# FIT SCHEMA TO BIGQUERY
schema = StructType.fromJson(json.loads(KPI_SCHEMA))

stage_df = spark.createDataFrame(stage_df.rdd, schema).repartition(col("area_code"))
```

Fonte: Autor

Por fim os dados estão prontos para serem armazenados no *Google BigQuery*, o seguinte trecho de código adiciona os dados no banco.

Figura 27 – Código para salvar dados no Google BigQuery.

```
# WRITE ON BIGQUERY
stage_df.write \
    .format('bigquery') \
    .option('table', 'crave.kpi') \
    .mode('append') \
    .save()
```

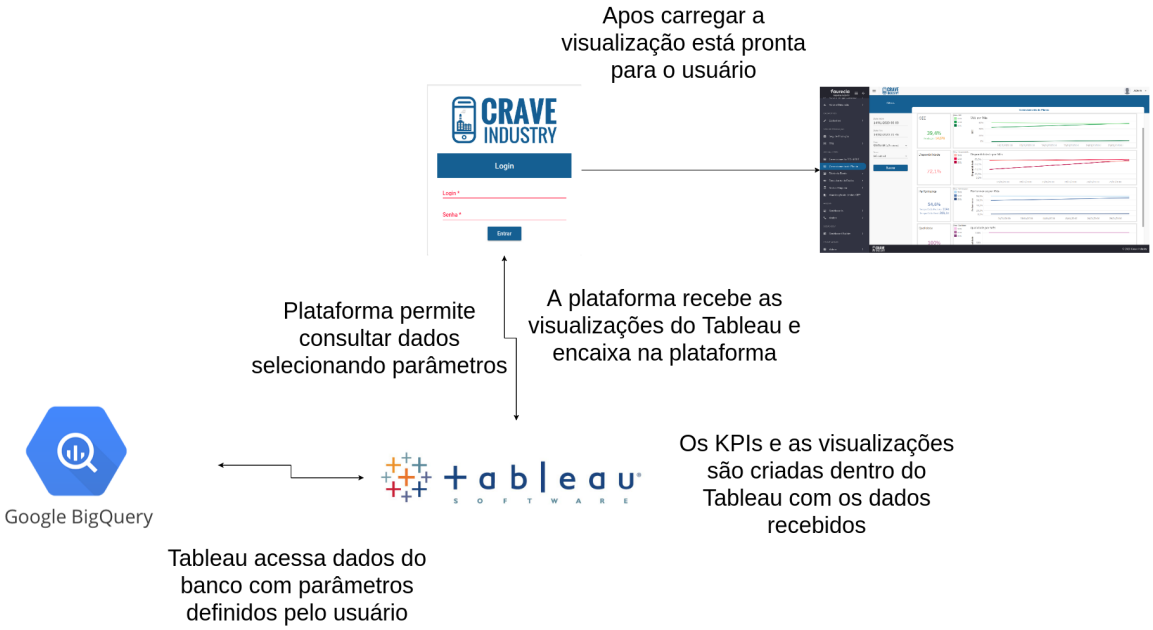
Fonte: Autor

6.2 Processo de BI

Com os dados processados e contextualizados é possível fazer os cálculos para obter KPIs, e apresentar visualizações que descrevam a linha de produção para acompanhamento do gerente.

A figura 28 ilustra o processo de BI a partir do ETL.

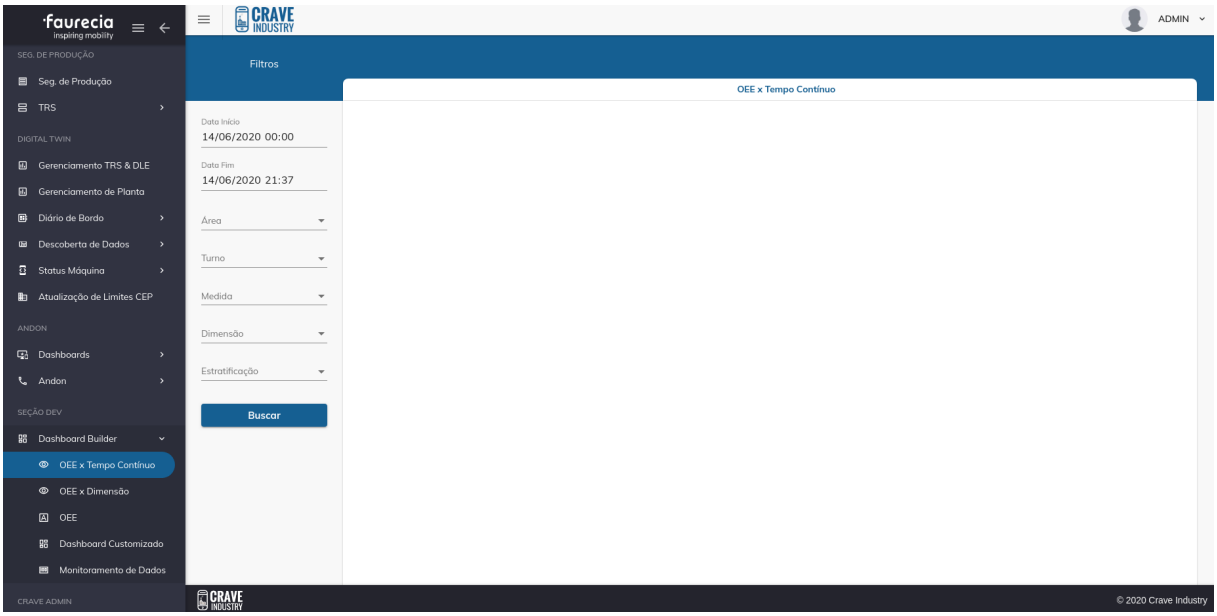
Figura 28 – Fluxo de interação do processo de BI.



Fonte: Autor

O cliente pode obter visualizações acessando a Plataforma Crave, preenchendo dados de busca, como numero de máquina, área da fabrica e intervalo de tempo. Esses dados são utilizados para gerar uma *Uniform Resource Locator* (URL) que dá acesso ao *Tableau Server* recebendo os dados preenchidos e utiliza para buscar os dados solicitados ao *Google BigQuery*.

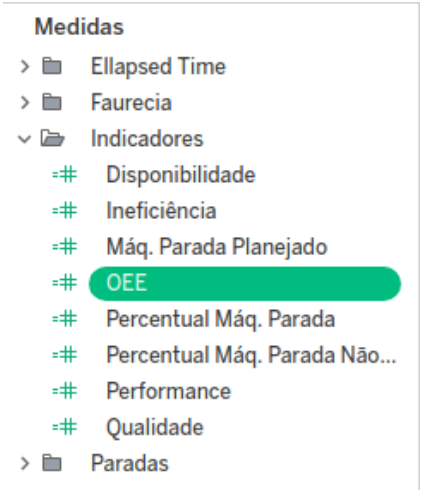
Figura 29 – Pagina para consulta de OEE..



Fonte: Autor

Ao receber os dados o Tableau calcula os KPIs e gera as visualizações que são encaixadas dentro da Plataforma Crave onde o cliente pode analisar e tomar decisões. Os KPIs são calculados sobre o conjunto de dados obtidos pela consulta, assim para cada consulta de dados o valor é calculado e apresentado. A figura 30 apresenta a lista de medidas calculadas dentro do Tableau.

Figura 30 – Lista de medidas calculadas pelo Tableau.



Fonte: Autor

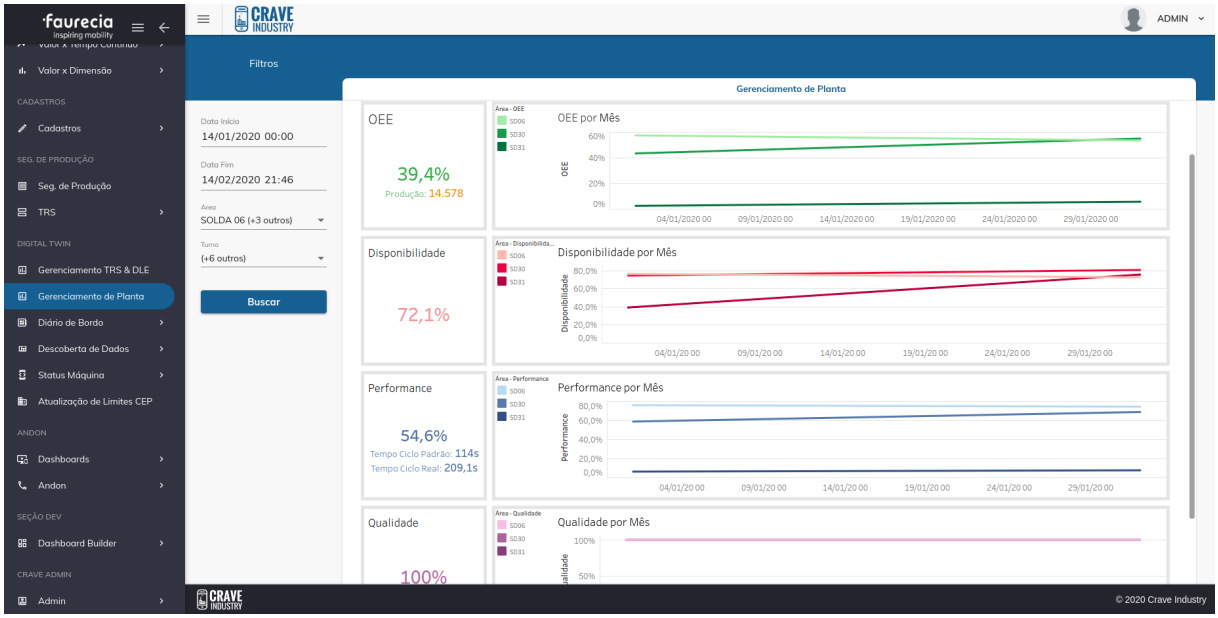
As medidas *OEE, Disponibilidade, Performance e Qualidade* são os KPIs. A figura 31 apresenta um exemplo de consulta de OEE por semana, segmentado por turno para um mês de dados.

Figura 31 – Consulta de OEE por semana segmentado por turno.

Fonte: Autor

Há inclusive visualizações de gerenciamento de planta, como a apresentada pela figura 32 que também recebe dados processados pelo Spark.

Figura 32 – Apresentação de OEE por semana segmentado por turno para 1 mês de dados coletados.



Fonte: Autor

6.3 Volume de dados processados

A tabela 2 apresenta os resultas do volume de dados obtidos do banco Amazon S3 para cada chamada do algoritmo.

Tabela 2 – Volume de dados processados por chamada e por dia.

Algoritmo Spark	Frequência de chamada	Volume de dados processados por cada chamada	Volume de dados processados por dia
KPI	10 min	21,6kB	3.110,4kB
Apontamento Refugo	1h	130,0kB	3.120,0kB
Diario Bordo	30min	1.560,0kB	74.880,0kB

O total de dados processados por é de 81,0Mb/dia ou de 2.430,0Mb/mês aproximadamente. Segundo Amazon AWS o custo de acesso de acesso ao S3 é de $0.09 centavos de dólar por GB, o que resulta num custo de $0,21 centavos de dólar por mês. Há também o custo de armazenamento, segundo a mesma fonte citada anteriormente é de $0.023 centavos de dólar por GB, representando um aumento mensal de $0,05 centavos de dólar.

O custo de acesso ao *Google BigQuery* segundo *Google Cloud* só é aplicado caso a consulta de dados ultrapasse 1TB por mês, e de armazenamento o custo só é aplicado quando ultrapassa os 10GB ao mês.

O custo para manter a atual capacidade de processamento é de aproximadamente 600R$, com um *cluster* de três máquinas recebendo dados não apenas do KPI, mas também de outros produtos.

Para avaliar a performance de leitura de dados armazenados no *Google BigQuery* é definido como um fator em conjunto com a escalabilidade. Para o algoritmo Spark será avaliado o tempo de processamento de cada algoritmo.

A consulta de dados do *Google BigQuery* para tabela KPI, coletando todos os dados de um cliente equivalente a um ano e seis meses de dados consome 16.9 segundos e processa 2.1 Gb de dados. Sendo está a pior situação de uma consulta, o tempo consumido foi considerado suficiente, concluindo que uma consulta de dados na plataforma deve consumir no máximo 20 segundos.

O ponto fundamental para utilizar o Spark é a escalabilidade, assim conforme mais clientes surgem e mais poder de processamento se torna necessário, com as tecnologias em nuvem é possível aumentar a capacidade de processamento com apenas alguns comandos.

7 Conclusão

O objetivo de desenvolver um algoritmo para processamento de dados foi atingido, permitindo a consulta de dados e obtenção de indicadores. A utilização da computação em nuvem para acionar o algoritmo desenvolvido permite que, conforme o aumento da demanda por poder de processamento, seja possível aumentar a capacidade do sistema, justificando assim o potencial transformador da computação em nuvem.

8 Trabalhos Futuros

Os resultados apresentados atenderam ao objetivo, porém há alguns questionamentos a se fazer, principalmente em relação a quantidade de dados, já que o Apache Spark é uma ferramenta para tratamento de enormes volumes de dados, até mesmo de terabyte de dados processados em tempo real. Atualmente há apenas um cliente que utiliza o produto KPI, por isso não há como saber se os algoritmos estão genéricos o suficiente para tratar dados de outros clientes pois este cenário é desconhecido. O Google Cloud fornece a capacidade de desenvolver softwares completos para funcionarem dentro da nuvem com o Cloud App Engine, que permite customização muito maior se comparada com o Dataproc, além de custar menos. Considerando o apresentado propôs-se uma lista de melhorias a serem implementadas em trabalhos futuros:

- Aprimorar a biblioteca crave.py criando mais funções genéricas para tratar qualquer fonte de dados industriais
- Desenvolver uma API para todo o processo ETL e utilizar o Google Cloud App Engine, que permite desenvolver aplicações customizadas dentro da nuvem.
- Avaliar a utilização da biblioteca Pandas considerando o volume de dados processados.

Referências

Peter Mell e Timothy Grance. PETER Mell e Timothy Grance. 2010. US 7,716.253 B2. Disponível em: <https://patentimages.storage.googleapis.com/8c/06/4d/0ec46c2ba01f00/US7716253.pdf>. Citado na página 37.

AMES, V. A. et al. Semiconductor manufacturing productivityoverall equipment effectiveness (oee) guidebookrevision 1.0. *SEMATECH*, 1995. Citado 2 vezes nas páginas 40 e 41.

BAHRIN, M. A. K. et al. Industry 4.0: A review on industrial automation and robotic. *Jurnal Teknologi*, 2016. Citado na página 34.

Baygin, M. et al. An effect analysis of industry 4.0 to higher education. In: *2016 15th International Conference on Information Technology Based Higher Education and Training (ITHET)*. [S.l.: s.n.], 2016. p. 1–4. Citado na página 33.

DAVIES, R. Industry 4.0 digitalisation for productivity and growth. *EPRS | European Parliamentary Research Service*, 2015. Disponível em: <https://www.europarl.europa.eu/RegData/etudes/BRIE/2015/568337/EPRS_BRI(2015)568337_EN.pdf>. Citado 2 vezes nas páginas 33 e 34.

GILCHRIST, A. *Industry 4.0: The Industrial Internet of Things*. [S.l.]: APRESS, 2016. ISBN 978-1-4842-2046-7. Citado na página 33.

HOLSAPPLE, C.; LEE-POST, A.; PAKATH, R. A unified foundation for business analytics. *Decision Support Systems*, v. 64, p. 130 – 141, 2014. ISSN 0167-9236. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167923614001730>. Citado na página 37.

LASI, H.; KEMPER, H.-G. Industry 4.0. *Springer Fachmedien Wiesbaden*, 2014. Citado na página 25.

MARSTON, S. et al. Cloud computing — the business perspective. *Decision Support Systems*, v. 51, n. 1, p. 176 – 189, 2011. ISSN 0167-9236. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167923610002393>. Citado 2 vezes nas páginas 46 e 48.

MELL, P.; GRANCE, T. *The NIST Definition of Cloud Computing*. [S.l.], 2010. Disponível em: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>. Citado 2 vezes nas páginas 46 e 47.

NAKAJIMA, S. *Introduction to TPM (Total Productive Maintenance)*. [S.l.]: Productivity Press, 1988. Citado 3 vezes nas páginas 38, 39 e 40.

Qi, Q.; Tao, F. Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. *IEEE Access*, v. 6, p. 3585–3593, 2018. Citado na página 35.

SCHMIDT, R. et al. Industry 4.0 - potentials for creating smart products: Empirical research results. In: ABRAMOWICZ, W. (Ed.). *Business Information Systems*. Cham: Springer International Publishing, 2015. p. 16–27. ISBN 978-3-319-19027-3. Citado na página 25.

STAMATIS, D. *The OEE Primer Understanding Overall Equipment Effectiveness, Reliability, and Maintainability*. [S.l.]: Productivity Press, 2010. Citado 2 vezes nas páginas 38 e 39.

Watson, H. J.; Wixom, B. H. The current state of business intelligence. *Computer*, v. 40, n. 9, p. 96–99, 2007. Citado na página 37.

ZAHARIA, M. et al. Spark: Cluster computing with working sets. *University of California, Berkeley*, 2010. Citado na página 42.

ZAHARIA, M. et al. Apache spark: A unified engine for big data processing. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 59, n. 11, p. 56–65, out. 2016. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/2934664>. Citado 2 vezes nas páginas 42 e 43.