Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia
COPPE UFRJ

UMA METODOLOGIA PARA MODELAGEM DA EXPERIÊNCIA DO CLIENTE ATRAVÉS DE PLATAFORMAS DE BIG DATA EM UNIDADE DE RESPOSTA AUDÍVEL

Vinicius Brito Rocha

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Civil, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Civil.

Orientador: Nelson Francisco Favilla Ebecken

Rio de Janeiro
Junho de 2019

UMA METODOLOGIA PARA MODELAGEM DA EXPERIÊNCIA DO CLIENTE ATRAVÉS DE PLATAFORMAS DE BIG DATA EM UNIDADE DE RESPOSTA AUDÍVEL

Vinicius Brito Rocha

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA CIVIL.

Examinada por:

_____
Prof. Nelson Francisco Favilla Ebecken, D.Sc.

_____
Prof. Rogério Pinto Espíndola, D.Sc.

_____
Prof. Marta Lima de Queirós Mattoso, D.Sc.

_____
Prof. Mário Antonio Ribeiro Dantas, Ph.D.

_____
Prof. José Luis Drummond Alves, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
JUNHO DE 2019

Rocha, Vinicius Brito
UMA METODOLOGIA PARA MODELAGEM DA EXPERIÊNCIA DO CLIENTE ATRAVÉS DE PLATAFORMAS DE BIG DATA EM UMA UNIDADE DE RESPOSTA AUDÍVEL/ Vinicius Brito Rocha. – Rio de Janeiro: UFRJ/COPPE, 2019.

XII, 103 p.: il.; 29,7 cm.

Orientador: Nelson Francisco Favilla Ebecken.

Tese (doutorado) – UFRJ/ COPPE/ Programa de Engenharia Civil, 2019.

Referências Bibliográficas: p. 74 – 79.

1. Big Data. 2. Experiência do Cliente. 3. Aprendizado de Maquina. 4. Unidade de Resposta Audível 5. Telecomunicações I. Ebecken, Nelson Francisco Favilla. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Civil. III. Título.

Dedico este trabalho aos meus pais. Minha mãe, Vera Lúcia Brito dos Santos (in memorian), que deu sua vida por mim e nunca deixou de acreditar no meu potencial. Ao meu pai José Aurimar Cunha da Rocha (in memorian), um grande homem e exemplo para mim. Ao meu sobrinho Yuri Rocha Nery (in memorian) que nos deixou durante a conclusão deste trabalho. E principalmente a minha eterna amada esposa, Alessandra Teixeira dos Santos Luiz Brito Rocha, sem ela não teria chegado tão longe.

Agradecimentos

A meu Orientador, Nelson Francisco Favilla Ebecken, pela forma que conduziu o processo de orientação, com sabedoria e parcimônia, fez tudo ser um prazer.

A Myrian Costa, que trabalhou ao meu lado como mentora, me ajudando em cada dificuldade.

Ao meu tio Luiz Fernando Pimenta Cardoso, um pai para mim, e a meu primo Felipe Cardoso, um irmão.

A Radakian Lino, um grande líder e amigo, que apoiou essa ideia desde o início. Sem isso essa tese não seria possível.

A Rafael Oliveira, Marina Barreto e Fernanda Alves, meus grandes amigos que sempre me apoiaram.

A empresa OI, seus colaboradores, e os amigos que fiz enquanto estive lá.

E a todos que fizeram parte desta jornada ao longo de todos esses anos. Muito obrigado.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

UMA METODOLOGIA PARA MODELAGEM DA EXPERIÊNCIA DO CLIENTE ATRAVÉS DE PLATAFORMAS DE BIG DATA EM UMA UNIDADE DE RESPOSTA AUDÍVEL

Vinicius Brito Rocha

Junho/2019

Orientador: Nelson Francisco Favilla Ebecken

Programa: Engenharia Civil

Este trabalho utilizou dados reais coletados de uma unidade de resposta audível, de uma companhia do setor de telecomunicações, para desenvolver um sistema para realizar o processamento de dados para fluxo contínuo (*Data Stream Processing System)* através tecnologias de Big Data resilientes, escaláveis e distribuídas. E com base nessa plataforma propor uma nova metodologia para medir e avaliar a experiência do cliente fazendo uso da teoria dos grafos e aprendizado de máquina. Auxiliando o processo de tomada de decisão no contexto do atendimento ao cliente, seja pela criação do sistema quanto pela metodologia, sendo ambos extensíveis a outros diferentes contextos de processamento de dados em tempo-quase-real.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A METHODOLOGY FOR MODELING CUSTOMER EXPERIENCE THROUGH BIG DATA PLATFORMS FOR AN INTERACTIVE VOICE RESPONSE

Vinicius Brito Rocha

June/2019

Advisor: Nelson Francisco Favilla Ebecken

Department: Civil Engineering

This work made use of data log collected from an Interactive Voice Response for customers service of a telecommunications company, to develop a Data Stream Processing System (DSPS) using technologies resilient, scalable and distributed for big data purpose. Based on the DSPS created was possible to purpose a new method to measure and evaluate customer experience using graph theory and machine learning for classification resulting in a better process for decision-makers of customer service. And beyond that, both, system and methodology can be used in other contexts of near of real-time data processing and customer experience respectively.

Sumário

1 Introdução.....1

1.1 A motivação a partir do conceito de experiência do cliente.....2
1.2 Motivação a partir das informações de URA.....3
1.3 A metodologia para medir a experiência do cliente.....5

2. Arquitetura DSPS e as tecnologias resilientes, escaláveis e distribuídas.....6

2.1 Hadoop.....10

2.1.1 MapReduce.....11
2.1.2 Hadoop Distributed File System – HDFS.....12
2.1.3 Namenodes and Datanodes.....13

2.3 Spark.....14

2.3.1 Interfaces de acesso e principais bibliotecas.....15

2.3.1.1 Spark SQL.....16
2.3.1.2 Spark Streaming.....17
2.3.1.3 Spark Mllib.....*19*
2.3.1.4 GraphX.....20
2.3.1.5 Bibliotecas terceiras.....20

2.4 NiFi.....21
2.5 Kafka.....23

2.5.1 Demais componentes do Kafka.....26

2.6 Impala.....27
2.7 Kudu.....29

2.7.1 Kudu e alguns tópicos relevantes.....31
2.7.1 Aplicações Kudu para problemas interativos.....32

2.8 Superset.....33
2.9 Aspectos sobre a arquitetura do DSPS.....35
3.1 Cliente no centro do negócio.....36
3.2 Modelo de atendimento SAC.....37
3.3 URA.....38

3.3.1 Aspectos do modelo de navegação em uma URA.....41
3.3.2 Aspectos gerais do processo de navegação em uma URA.....42

4 O sistema DSPS e a metodologia para avaliação da experiência do cliente.....44

4.1 A metodologia para medição da experiência do cliente.....44
4.2 Etapas para utilizar um modelo de agrupamento de chamadas.....46

4.3 Fluxo de dados.....50
4.4 Processamento do grupo inicial de logs.....53
4.5 Agrupamento e personas.....54
4.6 Implantação dos agrupamentos no DSPS.....58

5 Resultados da classificação em tempo quase real para log de chamadas.....60
6 Conclusões.....63
Referências Bibliográficas.....65
Anexo 1 Exemplo de log completo de uma chamada.....70
Anexo 2. Código para movimentar logs de URA simulando o processo de criação de logs de chamadas.....94
Anexo 3. Código Spark Streaming.....95

Lista de Figuras

Figura 1: Os 3 Vs do Big Data....6
Figura 2: Arquitetura conceitual utilizada....7
Figura 3: Arquitetura do GFS, (A.P.MITTAL; JAIN; AHUJA, 2015)....8
Figura 4: Arquitetura Final considerando plataformas RED....9
Figura 5: Distribuição de tipos de projetos da Fundação Apache. Fonte: site Fundação Apache....10
Figura 6: Fluxo do processo de calculo utilizando MapReduce, para o exemplo da temperatura....11
Figura 7: Comparativo alto nível da arquitetura Hadoop e Spark....14
Figura 8: Busca por tecnologias relacionadas com Big Data no forum Stack Overflow. 15
Figura 9: Esquema de módulos do Spark....16
Figura 10: Processamento de lotes em Streaming....17
Figura 11: Fonte de entrada e saída para o D-Stream....18
Figura 12: Overview de um sistema Spark Streaming....19
Figura 13: composição do GraphX....20
Figura 14: Arquitetura do Nifi. Fonte: site Nifi....21
Figura 15: Interface do NIFi, com os componentes utilizados neste trabalho....23
Figura 16: Esquema para um sistema ideal de assinante-publicação. Fonte: Cloudera Kafka Guide....24
Figura 17: Arquitetura esquemática do Kafka. Fonte Apache Kafka Guide....25
Figura 18: Interfaces de acesso ao Kafka....26
Figura 19: Esquema para os Brokers do Kafka....27
Figura 20: Esquema de funcionamento dos brooker....27
Figura 21: Fluxo durante o processamento de consultas através do Impala....28
Figura 22: Tabelas armazenadas no Kudu....31
Figura 23: Arquitetura de servidores Master e Tablets do Kudu....32
Figura 24: Imagem exemplo de alguns gráficos do Superset....34
Figura 25: Fluxo resumido representando o modelo SAC....38
Figura 26: Parte editada de um log de uma URA....39
Figura 27: Diagrama de navegação de uma chamada....41
Figura 28: Fluxo metodológico para mensuração da experiência do cliente....46
Figura 29: Exemplo de uma figura modelada como um grafo....47
Figura 30: Fluxo final para processamento de logs de URA....50
Figura 31: Telas auxiliares para ilustra o funcionamento do DSPS....53
Figura 32: Distribuição do tempo de processamento do primeiro grupo de logs....54
Figura 33: Simulação: número de clusters vs WSS - Elbow Method....56
Figura 34: Ilustração do uso da função pipeline....59
Figura 35: Primeiro painel: identificação da ultima chamada disponibilizada pelo sistema....61
Figura 36: Segundo painel: dados históricos da chamdas....62

Lista de tabelas

Tabela 1: Linha do tempo do Hadoop.....10
Tabela 2: Conexões extras para o Superset.....35
Tabela 3: Tabela de atributos disponíveis nos log, essenciais para experiência do cliente. 40
Tabela 4: exemplo de chamada navegada.....48
Tabela 5: Graus de entrada para uma chamada.....48
Tabela 6: Graus de saída para uma chamada.....49
Tabela 7: Atributos sumarizados para o K-Means.....49
Tabela 8: Exempo de chamada sumarizada para submissão ao K-Means.....50
Tabela 9: Atributos da tabela com a classificação de acordo com cluster.....52
Tabela 10: Centroides obtidos após execução do K-Means, para o conjunto com 17.646 logs.....57

Lista de abreviaturas

DSPS – Data Stream Processing System

HSPA – High Speed Packet Access

IOT – Internet of Things

LTE – Long-Term Evolution

NPS – Net Promoter Score

NTP – Network time protocol

RDBMS – Relational database management system

RED – Resiliente, escalável e distribuído

SAC – Serviço de atendimento ao consumidor

SMS – Short message service

SOA – Service Oriented Architecture

URA – Unidade Remota Audível

WSS – Within Sum of Squares

1 Introdução

A motivação inicial para este trabalho foi o de explorar tecnologias de Big Data, como Hadoop e Spark, e uma vez identificadas suas principais funcionalidades, utilizá-las para modelar um problema específico sobre a percepção do cliente ao utilizar um serviço de autoatendimento realizado através de uma Unidade de Resposta Audível – URA, que é oferecido por uma grande empresa do setor de telecomunicações.

Um dos resultados esperados pelo trabalho é responder se o uso desse tipo de tecnologia para processamento de dados, pode auxiliar durante a identificação de clientes que experienciaram negativamente o uso deste serviço de autoatendimento. Este tipo de pergunta pode ser respondida de inúmeras formas, mas no contexto específico do trabalho, visa-se obter a resposta sem que haja a necessidade de um questionamento direto ao cliente, por exemplo, através de algum tipo de questionário realizado após o atendimento, seja via URA ou por qualquer outro meio disponível. Para avaliar a experiência do cliente serão usados os registros da sua navegação no sistema, ou seja, os logs produzidos pela URA após o processo de atendimento.

O presente trabalho propões duas contribuições, a primeira é uma proposta de arquitetura para um sistema baseado em tecnologias resilientes, escaláveis e distribuídas – RED que seja capaz extrair as informações disponíveis sob a navegação do cliente e inferir sobre sua experiência. A segunda proposta é a criação de uma metodologia capaz de medir a experiência do cliente, conforme definida em (NASH; ARMSTRONG; ROBERTSON, 2013), para sistemas de URA, ou qualquer outra plataforma que se permita avaliar os agentes de atendimento. Em ambas contribuição, sob a ótica da arquitetura, ou sob a ótica da metodologia, podem ser estendida a diferentes aplicações onde o principal objetivo seja a definir e medir estruturas a partir da experiência do cliente.

A pesquisa aqui documentada, perseguiu continuamente a ideia da criação de um sistema para tomada de decisão em tempo-quase-real, a partir do uso dos logs criados ao final do procedimento de atendimento em uma URA.

O sistema de tomada de decisão foi desenvolvido em um formato de protótipo, sendo apenas capaz de processar logs de uma URA específica, neste caso, exclusivamente clientes da modalidade pré-pago. Porém, com muito pouco desenvolvimento a arquitetura aqui proposta pode ser utilizada em qualquer outro tipo de log de URA, assim como, em qualquer sistema que avalie um fluxo de atendimento de diversos canais.

Sistemas de processamento em tempo-quase-real são denominados como Data Stream Processing System – DSPS. Os autores (SAMOSIR; INDRAWAN-SANTIAGO;

HAGHIGHI, 2016) e (DE SOUZA, 2015), propõe soluções dessa natureza para auxiliar problemas que abordem um fluxo contínuo de dados (streaming). Sistemas de natureza DSPS passaram a receber mais destaque mediante ao seu uso para abordar problemas descritos como Big Data, nomenclatura dada para um gama de problemas que tratam questões envolvendo processamento de dados a partir de tecnologias resilientes, escaláveis e distribuídas, em contextos de aprendizado de máquina entre outros. Em seus trabalho, (GANDOMI; HAIDER, 2015), propõe uma revisão em conceitos que caracterizam o este tema, além disso, explicam como o termo Big Data se popularizou no segmento corporativo, mas que atualmente está em desuso no meio acadêmico, e sendo substituído por processamento intensivo e distribuído. Mediante a abordagem de temas interdisciplinares nesta pesquisa, o estado da arte deu-se na busca do máximo número de autores específicos cada um em suas áreas de pesquisa.

1.1 A motivação a partir do conceito de experiência do cliente

> A experiência do cliente é definida como a soma de todas as interações de um cliente com uma companhia, considerando todos os pontos de contato, incluindo: funcionários, canais, sistemas, produtos e serviços. Isto também inclui a percepção sobre sua marca. (NASH; ARMSTRONG; ROBERTSON, 2013, p.1)

A definição apresentada por (Nash et al. 2013), foi usada como referência, uma vez que o trabalho em questão apresenta uma definição clara sob o tema experiência do cliente, não define como este deve ser mesurado, abrindo então uma lista de possibilidades a serem exploradas. O DSPS criado permitiu propor um método de medição, assim com utilizá-lo para tomada de decisão.

A definição de Nash é facilmente adaptável para inúmeros tipos de situações envolvendo serviços de atendimento. Ao abordar uma URA através desse processo, a definição pode ser interpretada como a medição de cada uma das interações entre um cliente e a companhia, através do fluxo de navegação de menus em uma única chamada.

Mesmo com novos canais para realizar o atendimento, como páginas web, aplicativos para telefones móveis e redes sociais, a URA ainda ocupa uma relevante posição ao se tratar de plataformas para autoatendimento no contexto do serviço de atendimento ao consumidor – SAC. Neste modelo a URA é utilizada como a primeira camada, filtrando o número de clientes a terem contato com um funcionário ou prestador de serviço terceirizado. Isto é feito visando redução dos custos, com equipamento, pessoas e serviços (GANS; KOOLE; MANDELBAUM, 2003).

É comum estudos sobre experiência do cliente, atreladas ao modelo SAC, concentrarem seus esforços na etapa do atendimento realizado por seres humanos. Isso ocorre pois é neste elo da cadeia onde estão os maiores custos, e tem com alvo a utilização de métodos capazes de reduzir o tempo dedicado ao atendimento, outra vertente muito explorada é como a execução dos processos de atendimento são percebidos pelo cliente (PÉREZ; RODRÍGUEZ DEL BOSQUE, 2015).

Casos onde a experiência do cliente é direcionada ao estudo de URAs concentram-se em abordar o cliente diretamente, após a utilização do canal, através da realização de um questionário de perguntas elaborando uma pesquisa a partir do modelo de Net Promoter Score - NPS URA, ou de forma similar, como feito em (SCHRODER; JOHNSON, 2009), que utilizou uma URA para aplicar um questionário online com o intuito de coletar informações de pacientes portadores do vírus HIV, de forma que os mesmos pudessem responder de maneira anônima.

Em (VERHOEF et al., 2009), trata o estudo da experiência do cliente através das estratégias que devem ser adotadas para de forma a influenciar a lealdade que este desenvolve com a marca. Enquanto em (GENTILE; SPILLER; NOCI, 2007), a proposta é como avaliar a satisfação, de maneira quantitativa, a partir da metodologia *Net Promoter Score* – NPS, vista inicialmente em (REICHHELD, 2003). Neste trabalho a avaliação da experiência do cliente é realizada a partir da coleta dos logs, produzidos pela URA após a interação com um cliente qualquer, ou seja, sem a necessidade da apresentação direta de um questionário ao usuário após este utilizar do canal.

1.2 Motivação a partir das informações de URA

(JANARTHANAM; RAJA; CORPORATION, 2017) realizaram um trabalho a partir de logs de chamadas de uma URA, e propuseram um procedimento para otimizarão do seu funcionamento através da análise dos caminhos percorridos pelos clientes em uma ligação. Para isso, fizeram uso de algoritmos de recomendação implementados no software SAS[1]. Já (KHOTS; CORPORATION, 2015), apresentou uma arquitetura baseada em um modelo de dados relacional para processar dados não estruturados de uma URA e identificar padrões relevantes. Alguns dos cenários onde sua arquitetura propõe auxiliar são, a avaliação de melhores rotas para o encaminhamento de chamadas e uso de algoritmos de linguagem natural, para auxiliar o reconhecimento

[1] SAS – Statistical Analysis System, empresa de capital fechado. Referência em soluções para análises de dados.

de compras personalizadas de pacotes para TV por assinatura. Assim como (JANARTHANAM; RAJA; CORPORATION, 2017) seu trabalho também é baseado na plataforma SAS.

Note que, em ambos trabalhos mencionados acima os resultados foram obtidos a partir de software de propriedade intelectual privada, que não contempla o uso de tecnologias resilientes, escaláveis e distribuídas ou streaming de dados, como propõe o tema abordado neste trabalho, fazendo deste um assunto ainda pouco explorado.

Um sistema para processamento do fluxo contínuo para os logs, permite a realização de avaliações da percepção do cliente logo após a disponibilidade da informação, ou seja, assim que o serviço do canal de atendimento seja finalizado. Esse tipo de arquitetura aumenta a capacidade de atuação por parte da companhia em caso de uma experiência negativa por parte do cliente.

Para combinar as diversas tecnologias que forma utilizadas para construção do DSPS, foi necessário realizar uma ampla revisão das funcionalidades de diferentes tipos de plataformas RED de diferentes propósitos. Entre esses, estão a criação de roteamento de dados, algoritmos para aprendizado de máquinas interativos, armazenamento de dados em memória, serviço de mensagens em tempo real, visualização de dados e outras.

Este trabalho tem como destaque o uso de um conjunto de dados de grande volume, a resolução de um problema de uma empresa e todo um segmento de negócio, que ao combinar tecnologias com características RED produziu um sistema com capacidade de auxiliar a tomada de decisão em tempo quase real, baseado na informação do uso da URA armazenada nos seus logs. E a partir de disso desenvolveu uma metodologia capaz de modelar a experiência do cliente passiva de aplicação em qualquer sistema, ou grupo de sistemas, ou conjunto de canais de atendimento que sejam modeláveis a partir de um grafo matemático.

1.3 A metodologia para medir a experiência do cliente

Como já citado, a metodologia tem com base a definição vista em (NASH; ARMSTRONG; ROBERTSON, 2013), que combinada conceitos introdutórios da teoria dos grafos (COSTA, 2011; DIESTEL, 2000) possibilitando modelar cada um das interações entre um cliente e uma URA conforme um grafo. Um grafo é um par $G=(V,E)$ que satisfaz a relação $E\subseteq[V]^2$ de tal forma que o par de elementos que

pertencem ao conjunto *E* sejam elementos do subconjunto V. Onde os elementos do conjunto *V* são definidos como vértices, ou nós do grafo *G*, enquanto os elementos do conjunto *E* são definidos como arestas do grafo G.

Sob o aspecto probabilístico (DEGROOT; SCHERVISH, 2012) a metodologia toma como base a teoria de processos estocásticos (KNILL, 2009; ROSS, 1996), onde para cada possível caminho percorrido por um cliente ao navegar por uma URA, sendo que essa URA siga os mesmos critérios da que é avaliada neste trabalho, esta pode ser modelado como uma Cadeia de Markov. E cada menu da URA é um vértice e cada uma das interações realizadas entre esses menus é uma aresta. Diante desse modelo para URA é possível extrair um conjunto de informações, atributos, que podem ser utilizados para diferentes finalidades.

Para criar um conjunto padronizado, que seja represente grande parte do que ocorre durante a navegação em uma URA, como segunda etapa do processo de definição dessa metodologia, propõe-se o uso de algum algoritmo não supervisionado (EVERITT; DUNN, 2001b; MINGOTI, 2007; TAN et al., 2013), buscando assim encontrar agrupamentos naturais que sejam capaz de auxiliar na avaliação da percepção do cliente, de tal forma a generalizar o processo de tomada de decisão em tempo quase real.

Essa trabalho foi dividido em capítulos de tal forma que o capítulo 2 introduza uma breve revisão das tecnologias RED utilizadas, assim como a arquitetura do *Data Stream Processing System* desenvolvido. Além disso, este capítulo também inclui a descrição do processo de experimentação em se trabalhar com diferentes tecnologias. O Capítulo 3 detalha as questões referentes a URA, como os logs e a experiência do cliente no modelo de atendimento. O Capítulo 4 descreve os resultados vivenciados ao submeter os logs através do DSPS e como desse processo surgiu uma metodologia. O Capítulo 5 tece as conclusões obtidas com essa pesquisa, assim como as recomendações obtidas em trabalhos futuros. E o Capítulo 6, as referências bibliográficas.

2. Arquitetura DSPS e as tecnologias resilientes, escaláveis e distribuídas

Nos últimos anos o aparecimento de diversas novas tecnologias para acelerar a coleta e análise de dados foi exponencial. O seu uso em problemas simples como testes AB, ou em algoritmos para estimar parâmetros de uma complexa rede neural aumentaram significativamente.

Problemas dessa natureza passaram a ser nomeados pelo termo *Big Data*, uma expressão que vem sofrendo uma grande generalização no que diz respeito ao seu escopo de abrangência. O termo foi mencionado pela primeira vez em um trabalho da Nasa (COX; ELLSWORTH, 1997), que argumentava a respeito do crescimento do volume de dados, e da dificuldade para armazenar as informações atuais limitados pelas tecnologias existentes. Outros trabalhos apontavam para a diferença entre o volume de dados produzido e a capacidade de armazená-los, (HILBERT; LÓPEZ, 2011), e como isso estava produzindo um efeito perceptível em diversos setores econômicos, (DEMING; DRUCKER, 2012).

A dificuldade em se limitar o escopo de um problema representado pela expressão Big Data é algo que ocorre atualmente, em seu trabalho (GANDOMI; HAIDER, 2015), apresentaram conceitos e métodos visando auxiliar neste aspecto. Problemas que se enquadram dentre as características de alto volume, velocidade e variedade são classificados como questões a serem tratadas como Big Data, estas três características passaram a ser conhecidas como o 3 Vs. A Figura 1 toma como base o que foi descrito em seu trabalho e apresenta uma proposta que quantificar cada um dos 3 Vs.

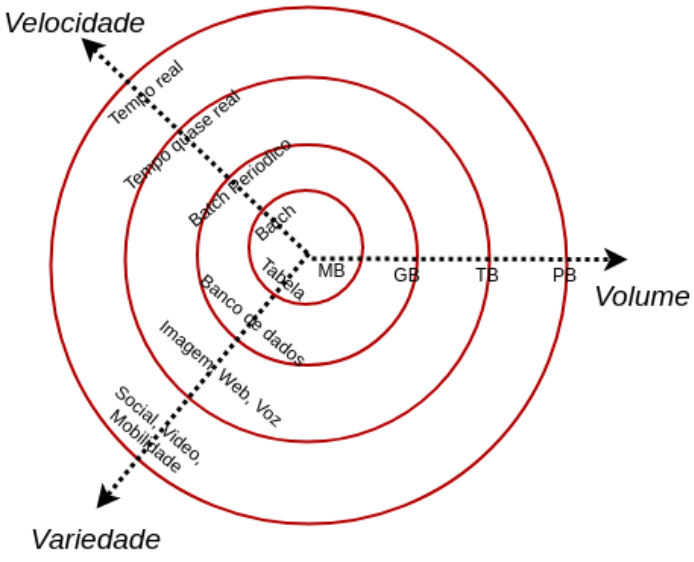

Figura 1: Os 3 Vs do Big Data.

A arquitetura para a construção do DSPS, representada pela Figura 2, tem como finalidade tratar algumas das características ilustradas pela figura acima, uma vez que os logs são dados não estruturados, o processamento para o sistema criado é em tempo quase real e o volume de dados processados pode chegar a uma escala de Terabytes em uma aplicação real (para o protótipo construído neste trabalho limita-se a um volume de Gigabyte).

Mesmo contemplando algumas das características representadas pelos 3 Vs, este trabalho define um problema de Big Data baseado no tipo de tecnologia que foi adotada para solucioná-lo. Neste caso, considerando apenas tecnologias resilientes, escaláveis e distribuída.

As etapas de desenvolvimento da arquitetura serão detalhadas no decorrer deste capítulo. É importante destacar que com base nesta estrutura e nas informações contidas nos logs foi possível formular a metodologia proposta nesta tese que permite modelagem da experiência do cliente, e que é capaz de atender a diversos cenários de maneira generalista.

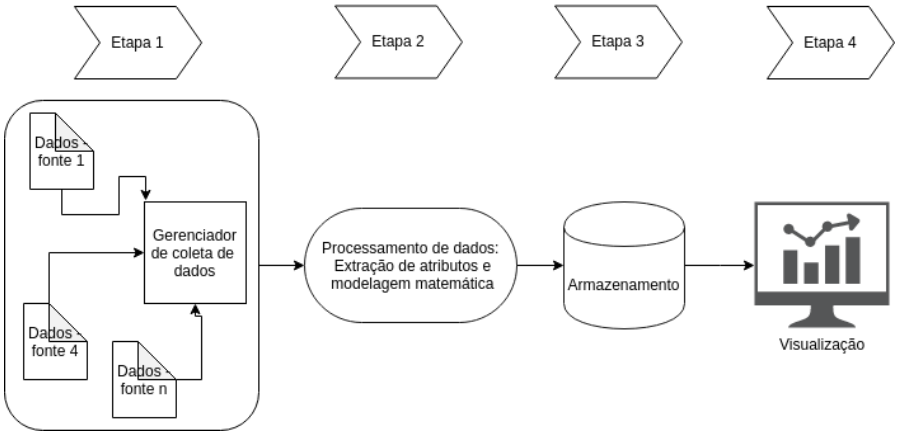

Figura 2: Arquitetura conceitual utilizada

O aspecto inicial da pesquisa envolveu buscar uma tecnologia RED capaz de realizar o processamento e modelagem dos logs. Desta forma, como base a Figura 2, o processo de pesquisa se iniciou pela etapa 2. A partir de uma revisão bibliográfica, baseada na história das plataformas RED, o trabalho de (GHEMAWAT; GOBIOFF; LEUNG, 2003), sobre o sistema de arquivos distribuído criado pelo Google, o GFS[2], é

[2] Google File System.

tido como o marco inicial, a Figura 3, (A.P.MITTAL; JAIN; AHUJA, 2015), ilustra a arquitetura do GFS. Ele é capaz de realizar o processamento de grandes volumes de dados e suas principais características são: monitoramento constante, tolerância a falhas com detecção de erros e recuperação automática e escalabilidade através de nós de um clusters dos computadores.

A partir desse momentos outros trabalhos surgiram baseados no artigo publicado pelo Google, onde o mais relevante foi o projeto de licença aberta Hadoop (PADHY, 2013; THE APACHE SOFTWARE FOUNDATION, 2011; WHITE, 2015), desenvolvido por Doug Cutting e Mike Cafarella enquanto ambos trabalhavam para o Yahoo. Ainda hoje o Hadoop é uma relevante e influência a muitos outros projetos, mais a frente neste capítulo existe uma sessão dedicada ao seu entendimento.

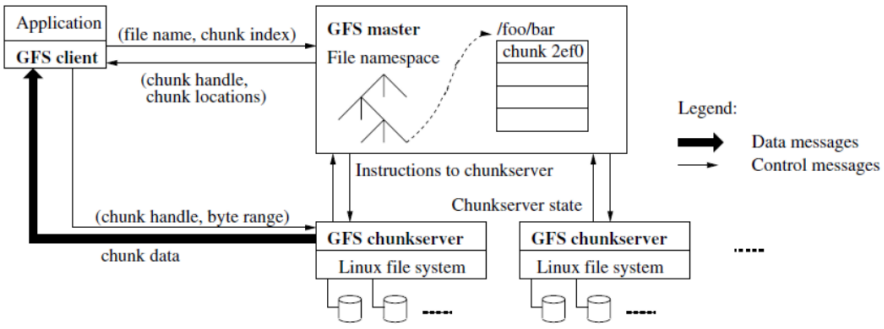

Figura 3: Arquitetura do GFS, (A.P.MITTAL; JAIN; AHUJA, 2015).

Mediante a necessidade de se processar logs em tempo quase real, e ao mesmo tempo modelar o processo de navegação no interior da URA a partir de algoritmos de aprendizado de máquina, o Hadoop não se mostra adequado. Isto se deu por alguns aspectos, primeiro pelo fato de sua estrutura armazenar arquivos em blocos de 128 Megabytes como tamanho mínimo, e cada log possui apenas 100 Kb em média. Além disso, a estrutura de processamento de dado do Hadoop, através do modelo de programação MapReduce (será apresentado a frente), implica que em cada etapa do processamento ocorra escrita em disco, tornando-o lento em execução de algoritmos iterativos.

Em meados de 2009, grande parte da comunidade estudiosa da computação intensiva e de aprendizagem de máquina voltou sua atenção para o Spark, um motor para o processamento de dados em memória com caracteristicas generalista, mas que une diversos projetos especialistas. Spark estende o modelo de programação baseado em

fluxo de dados oferecido pelo Hadoop (MapReduce) (CHAMBERS; ZAHARIA, 2017; ZAHARIA, 2016), além de ser capaz de processar grandes volumes de dados com até 100 vezes mais velocidade que o Hadoop (“Apache Spark™ - Unified Analytics Engine for Big Data,” [s.d.]).

O Spark se adequou muito bem a necessidade apresentada pelo problema deste trabalho, sendo a principal plataforma utilizada. A partir dele, o DSPS foi sendo desenvolvido em etapas, onde outras tecnologias foram sendo incorporadas em diferentes momentos de tal forma a complementar o uso do Spark visando a arquitetura ilustrada pela Figura 2.

A Figura 4 apresenta antecipadamente todas as tecnologias RED já como parte da arquitetura final do DSPS, visando facilitar o entendimento do mesmo. Cada uma dessas tecnologias será detalhada ao longo deste capítulo.

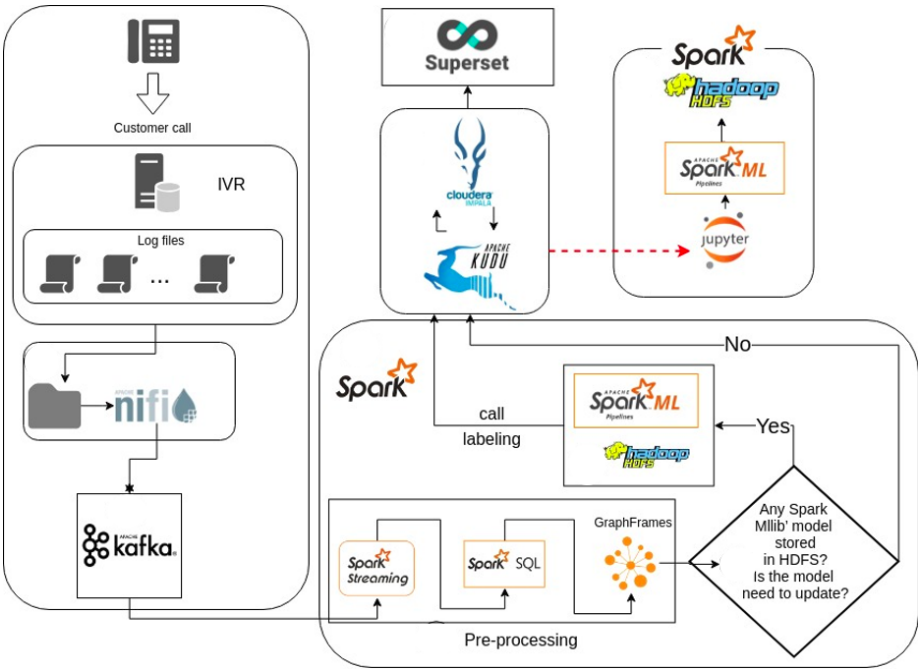

Figura 4: Arquitetura Final considerando plataformas RED

As tecnologias RED apresentadas na Figura 4 fazem parte da Fundação Apache – ASF (APACHE SOFTWARE FOUNDATION, 2019). A ASF foi criada em 1999 e atualmente possui mais de 350 projetos, dentre eles, aproximadamente 10% orientados a questões envolvendo Big Data, conforme ilustrado pela Figura 5. Dentre estes projetos

os mais relevantes atualmente são: Hadoop, Spark, Flume, Kafka, Impala, Kudu, Nifi, Sqoop, Hbase, Pig, Solr, Hive, Zookeper, Yarn (AGRAHARI; RAO, 2017).

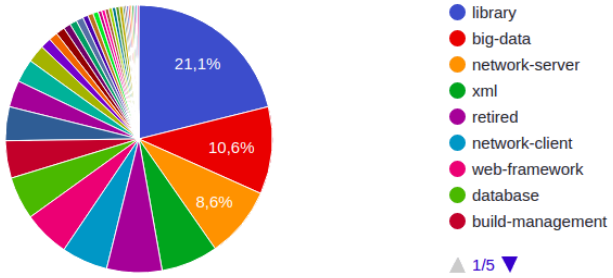

Figura 5: Distribuição de tipos de projetos da Fundação Apache. Fonte: site Fundação Apache

2.1 Hadoop

Como mencionado anteriormente o Hadoop, *(THE APACHE SOFTWARE FOUNDATION, 2011)*, foi criado pelo Yahoo em 2011, e teve como maior influência o artigo publicado pelo Google, (GHEMAWAT; GOBIOFF; LEUNG, 2003), *The Google File System – GFS.* Ambos projetos apresentam um sistema de arquivos distribuído com características como *escalabilidade*, e *alta disponibilidade*[3], capaz de processar grandes volumes de dados, a partir de uma infraestrutura de computação em cluster. *Hadoop* é um das tecnologias mais populares atualmente para processamento de dados através de computação em clusters.

Hadoop é composto pelos subprojetos Hadoop Common, HDFS (Hadoop Distributed File System), YARN (*Yet Another Other Negotiator)*, MapReduce, porém é comum utilizar o nome Hadoop para se referir a todo ecossistema de projetos RED. Os mais importantes projetos são, HDFS, que permite armazenamento de qualquer tipo de arquivo, e o MapReduce, que é o um paradigma de programação para o processamento de dados distribuídos em um cluster de computadores.

Versões iniciais do que é agora o Hadoop Distributed Filesystem e o Map-Reduce implementado por Doug Cutting e Mike Cafarella.	O Hadoop é executado de maneira confiável em 20 nós.	Doug Cutting se junta ao Yahoo !.	o projeto Apache Hadoop começou oficialmente a apoiar o desenvolvimento independente do MapReduce e do HDFS	Adoção do Hadoop pelo Yahoo! Equipe de grade	O benchmark Sort (10 GB / node) é executado em 188 nós em 47,9 horas.	Yahoo! configurar os nós do cluster de pesquisa do Hadoop-300	O benchmark Sort é executado em 500 nós em 42 horas (melhor hardware do que o benchmark de abril).	O benchmark Sort é executado em 20 nós em 1,8 horas, 100 nós em 3,3 horas, 500 nós em 5,2 horas, 900 nós em 7,8 horas.	Ganhou o benchmark de 1 terabyte em 209 segundos em 900 nós	Carregamento de 10 terabytes de dados por dia em clusters de pesquisa.	clusters com um total de 24.000 nós	Ganhou o tipo de classificação classificando 500 GB em 59 segundos (em 1400 nós) e a classificação de 100 terabytes em 173 minutos (em 3400 nós).

Tabela 1: Linha do tempo do Hadoop.

[3] O mesmo que Resiliente.

Como uma das primeiras versões para o DSPS foi utilizado o Hadoop, onde os logs foram armazenados no HDFS e processados com MapReduce. Porém esta solução foi abandonada mediante a necessidade de um processamento em tempo real e algumas características limitantes do Hadoop.

2.1.1 MapReduce

MapReduce é fundamentalmente um sistema capaz de realizar processamento em lotes de grandes volumes de dados, porém não é adequado em casos com processamento interativo, nem tão pouco permite a execução de busca em dados com baixa latência, ou seja, em poucos segundos.

MapReduce foi definido em (WHITE, 2015), como um modelo simples de programação para processamento de grandes volumes de dados, porém, não suficiente para a codificação de um programa útil por completo. Um programa MapReduce pode ser escrito em Java, Ruby e Python, por exemplo, pois o Hadoop permite diversas linguagens.

Um código submetido ao Hadoop, de acordo com o paradigma MapReduce, é automaticamente paralelizado conforme o número de clusters computacionais disponíveis. Isto permitiu que qualquer pessoa possa processar grandes volumes de dados, uma vez que disponha da quantidade de máquinas necessárias.

A Figura 6, apresenta as etapas de um processamento realizado através de um programa MapReduce (WHITE, 2015). Um programa MapReduce é composto das seguintes etapas: uma *Map*, onde cada registro é associado a uma chave, criando um par chave valor; uma *Shuffle* (aleatorização), que garante a entrega dos dados da etapa Map para Reduce; de uma etapa *Sort*, para reorganizar os dados ao final da etapa Map; e de uma *Reduce* para realização de operações matemáticas para o campo valor para as chaves criadas durante a fase *Map*. É importante ressaltar que a execução de um programa MapReduce está condicionado a um conjunto de dados de entrada, normalmente armazenados no HDFS.

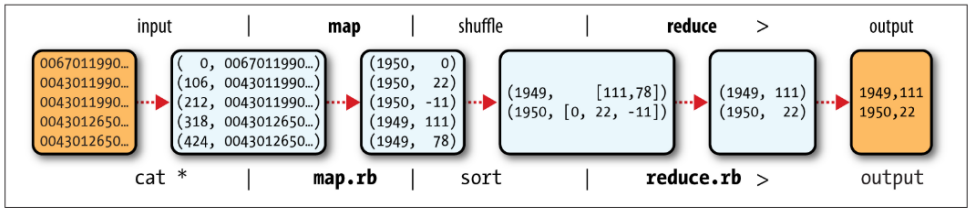

Figura 6: Fluxo do processo de calculo utilizando MapReduce, para o exemplo da temperatura.

É possível uma execução de um programa MapReduce com nenhuma tarefa do tipo *reduce*. Evitando assim a etapa de *shuffle*, *garantindo que nenhuma* transferência de dados ocorra para um nó externo, ocorrendo apenas quando a tarefa Map for finalizada em processo de escrita no HDFS.

Apesar de fora do escopo deste trabalho um resumindo entendimento do MapReduce se fez necessário neste trabalho, através desta sessão. Pois como já mencionado, o Hadoop foi considerado em um primeiro momento e o Spark possui uma modelo de programação similar.

Hadoop foi utilizado neste trabalho apenas somente através do HDFS, que serviu de repositório para armazenar os resultados obtidos com o modelo de clusterização, processo calculado durante a etapa 3, ilustrada pela Figura 4 e que será detalhado em capítulos posteriores. Já o processamento a partir do modelo de programação MapReduce foi inteiramente substituído pelo Spark.

2.1.2 Hadoop Distributed File System – HDFS

O HDFS é um sistema de arquivos, capaz de gerenciar todo o seu conteúdo de armazenamento através de um cluster de computadores. É mais complexo quando comparado a um sistema de arquivos convencionais, justamente por seu funcionamento de forma distribuída, que implica todos os prolemas desta natureza (WHITE, 2015).

Projetado para armazenar arquivos com grandes volumes através de um modelo de acesso aos dados onde a escrita é realizada uma única vez, e a leitura em múltiplos acessos. Desta forma, ao analisar um conjunto de dados é comum que grande parte dos dados, ou até mesmo todo ele, seja processado. Fazendo com que o tempo de leitura a todo o conjunto de dados seja mais relevante do que o tempo de leitura de uma única linha apenas.

O HDFS foi utilizado para armazenar os centroides resultantes do modelo de clusterização, uma vez que seu modelo de funcionamento permite múltiplos acessos aos dados, torna-o compatível com a arquitetura para o DSPS apresentada na Figura 4. Além disso, em um cenário onde a arquitetura deste trabalho esteja em funcionamento em um cluster de computadores, em caso de falha em um dos nós, os dados dos centroides ainda assim estariam disponíveis para uso.

O Hadoop trabalha com blocos de armazenamento de 128 megabytes, os arquivos no HDFS são divididos de acordo com o tamanho desses blocos e

armazenados em discos independentes. O resultado do modelo de clusterização é um arquivo com pouco mais de 1 kilobyte, porém não significa que seu armazenamento no HDFS ocupará um bloco inteiro de 128 megabytes de espaço em disco. Os blocos HDFS são construídos com este tamanho visando a minimização do custo com as buscas de dados. Porém, em um cluster de pequeno porte é interessante um tamanho de bloco menor dos que os 128 megabytes, para reduzir a taxa de transferência de dados, neste trabalho foram utilizados blocos de 32 megabytes, maiores detalhes em (WHITE, 2015).

2.1.3 Namenodes and Datanodes

Um cluster HDFS possui dois tipos de nós que trabalham em um padrão denominado mestre-escravo. Um nó recebe a atribuição de Name-node, tendo como função o gerenciamento central do cluster, enquanto os demais nós recebem a atribuição de *Data-nodes*, realizando o trabalho pesado. O *Name-node* gerencia o sistema de arquivos denominado Name-space, ou seja, a estrutura de árvore de arquivos e metadados, além disso, também é responsável por identificar os Data-nodes de todos os blocos os quais um arquivo está armazenado.

Em caso de falha do Name-node é impossível reconstruir os arquivos com origem nos blocos dos Data-nodes. São os Data-nodes que recuperam e armazenam cada bloco, seja sob demanda, via client, ou via Name-node. Além de periodicamente informem seu status para o Name-node.

Ainda sob o Name-node, é necessário garantir mecanismos para garantir a sua integridade, e o Hadoop possui dois. O primeiro é um backups periódico dos arquivos contidos no Name-node e o segundo é a execução de outro Name-node, que apesar do nome, não atua de fato como um Name-node. Sua principal função é registrar periodicamente a imagem do Name-space (a árvore do sistema de arquivos) em um log. Este segundo Name-node demanda um uso excessivo de processamento, praticamente a mesma quantidade de memória demandada pelo Name-node original em seu nó principal, por isso, geralmente sua execução é realizada em uma máquina separada. Além disso, o Name-node secundário também mantém um backup do Name-node principal, para utilização em caso de falha.

Como é possível verificar na arquitetura proposta na Figura 4, o DSPS faz uso do Hadoop somente através HDFS, com a finalidade de armazenar o resultado obtido pleo algoritmo K-Menas para considerando o cálculo dos centroides estimados via K-Means. Porém, mesmo com um escopo reduzido neste trabalho, o Hadoop possui uma grande importância no contexto de Big Data e por esta razão se dedicou uma sessão a um melhor entendimento de seu modelo de operação dentro deste trabalho.

2.3 Spark

Spark é um motor para a computação de alto desempenho em cluster, que consta com os benefícios da usabilidade e do desempenho, conforme apresentado em (ZAHARIA, 2016). Spark pode é dietamente comparado ao Hadoop, sob aspecto da sua proposta, tendo com principal diferença a sua arquitetura, que faz uso do processamento distribuído através da memória do tipo Random Access Memory – RAM, já o Hadoop utiliza o processamento distribuído através da escrita em disco, HDFS, para cada uma das suas operações MapReduce. A Figura 7 ilustra simplificadamente ambas arquiteturas.

Uma vez que faz uso da memória, Spark chegou a apresentar resultados até 100x mais eficientes que o Hadoop, com é apresentado em ("Apache Spark™ - Unified Analytics Engine for Big Data," [s.d.]), para uso em algoritmos interativos. (SHI et al., 2150), apresenta um comparativo mais detalhado entre Spark e o Hadoop, sob uma ótica das principais aplicações do primeiro, como por exemplo, capacidade para processamento de dados não estruturados, resolução de problemas de classificação supervisionada e não supervisionada e computação orientada a grafos.

Mediante aos melhores resultados do Spark em algoritmos interativos, com base nos trabalhos mencionados a ideia pelo uso do MapReduce foi substituída pelo Spark, que passou a ser a principal plataforma de Big Data para o desenvolvimento do DSPS criado nesta tese. Além disso, o MapReduce apresenta alta latência devido a escrita no HDFS, comprometendo seu uso em aplicações DSPS.

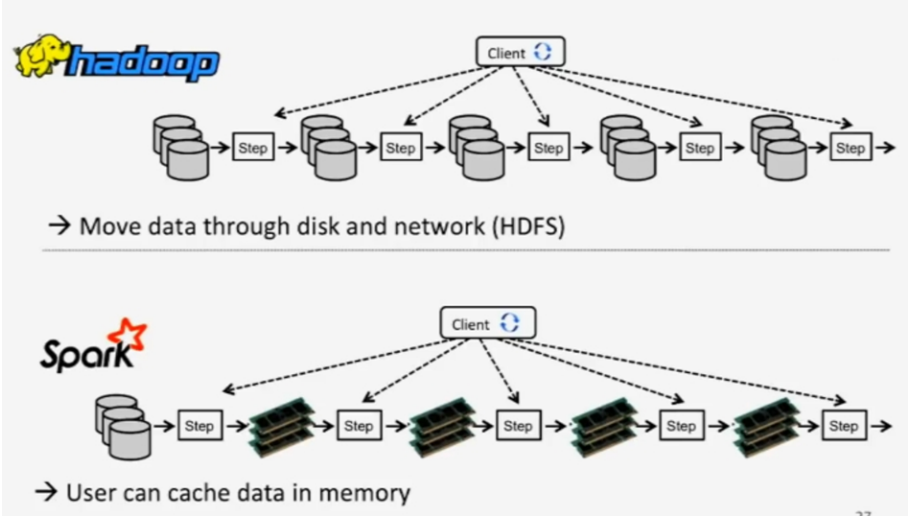

Figura 7: Comparativo alto nível da arquitetura Hadoop e Spark.

O Spark tem como principal abstração o *resilient distributed datasets* – RDD, uma estrutura de dados imutável, que faz uso de transformações para realizar operações

de alto nível, permitindo compartilhamento de dados em uma arquitetura de computação em cluster.

Entre algumas operações permitidas para um RDD estão map, filters e joins. Com uma das características de operações do Spark, para cada transformação de um RDD, envés de realizar o registro reais dos dados, apenas são registradas as transformações para criar de um novo RDD. Assim, em caso de perda de uma partição do cluster existem informações suficientes sobre o seu histórico, permitindo desta forma recalcular apenas a parte desejada, evitando que os RDDs perdidos sejam materializados a todo momento.

Apesar do principal benefício apresentado neste trabalho para o uso de RDD ser sua superioridade em velocidade comparado ao Hadoop, existem outros benefícios, que podem ser vistos em (CHAMBERS; ZAHARIA, 2017; CLOUDERA, 2019a; SAMOSIR; INDRAWAN-SANTIAGO; HAGHIGHI, 2016; ZAHARIA, 2016), entre outros. O Spark é o primeiro sistema de programação baseado em uma arquitetura distribuída em memória, com um propósito generalista para acelerar as interações dos algoritmos de aprendizado de máquina. O aumento de interesse em sua utilização pode ser visto na Figura 8, fonte (“Stack Overflow Trends,” [s.d.]), onde o crescimento da sua popularidade se dá tanto no meio acadêmico quanto nas emrpresas.

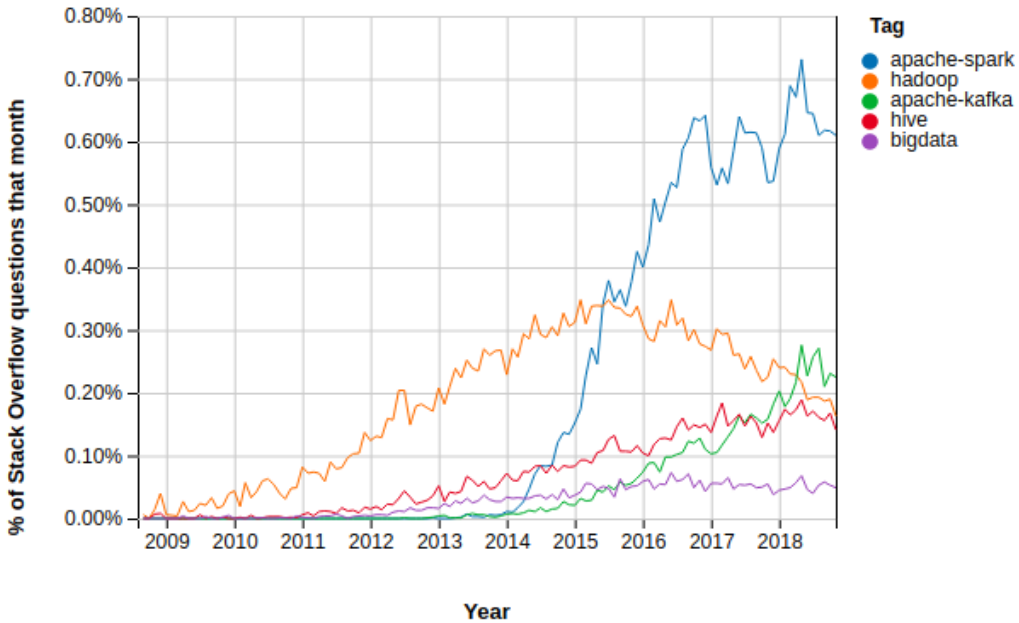

Figura 8: Busca por tecnologias relacionadas com Big Data no forum Stack Overflow.

2.3.1 Interfaces de acesso e principais bibliotecas

Atualmente entre as interfaces de acessos (API[4]), ou linguagens de programação, para uso do Spark estão: *Scala, Python, R e Java.* Sendo Scala a mais performática, sendo a linguagem original do Spark, mas é bastante comum seu uso a partir de Python e R, populares no uso de problemas de aprendizado de máquina. Este trabalho usou exclusivamente Python, e os códigos estão disponibilizados em Anexo.

Além da abstração RDD e das APIs, o Spark possui bibliotecas, ou projetos, que complementam seu conjunto de funcionalidades são elas: *Spark SQL, Spark Streaming, Mllib, e GraphX.* A Figura 9 apresenta as funcionalidades e projetos que o Spark engloba.

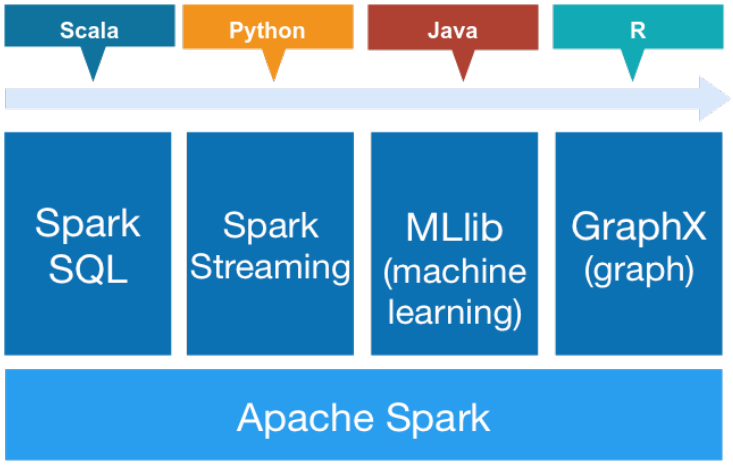

Figura 9: Esquema de módulos do Spark.

Além das bibliotecas disponíveis em seu núcleo, alguns outros projetos terceiros complementam o conjunto de soluções disponíveis para o Spark, como é o caso da biblioteca GraphFrames para modelam de grafos a partir de estruturas do tipo DataFrames.

Para processar os logs de URA foram utilizados, Spark Sql, Spark Streaming e Mllib simultaneamente. Porém, devido a uma limitação do GraphX que não possui suporte a API Python, a solução encontrada para a modelagem do fluxo de navegação das chamadas da URA como um grafo, foi feito a partir da biblioteca GraphFrames citada no paragrafo anterior.

2.3.1.1 Spark SQL

É um módulo para trabalho voltado ao uso de dados estruturados em formato de tabelas. Ao comparado com uma estrutura de dados modelada como um RDD, o *Spark*

[4] Application Programming Interface.

SQL oferece mais facilidade para manipulação dos dados, assim como performance em cálculos. Seu mecanismo interno utiliza algumas informações adicionais para acessar os dados de forma otimizada (ARMBRUST et al., [s.d.]). A motivação que fundamentou a criação desta biblioteca foi tornar a operação com um RDD mais eficientes, a partir da estrutura de dados DataFrames, que é uma coleção de objetos do tipo *Row*.

Junto com a estrutura de dados do tipo DataFrame (um Dataset organizado em colunas), foi introduzido o conceito de *schema*, que visava estabelecer uma maneira de descrever os dados. Com isso foi possível que Spark gerenciasse um *schema*, e assim, somente transmitisse os dados para os *Data-nodes* de forma mais eficiente.

O *Dataset* surgiu por último, oferecendo o melhor entre o modelo do RDD e do Dataframe. Isto porque, pois permite o uso de programação orientada a objetos, com forte tipagem e funções do tipo lambdas conforme um RDD, e ao mesmo tempo apresenta um desempenho otimizado visto em DataFrames.

Outra característica que motivou a transformação dos logs de URA de RDD para SparkSql foi sua capacidade de conexão via JDBC com vários tipos de fontes de dados. Dentre elas, Kudu, Hive e HDFS, além de suportar formatos de dados como Json, Orc, Avro ou Parquet.

2.3.1.2 Spark Streaming

A biblioteca Spark streaming permite a criação de fluxos de dados através de processamento em *micro-batching* de forma escalável. Essa projeto é uma extensão do Spark, porém, em vez do uso de RDDs sua principal abstração são os D-Streams, uma série continuada de RDDs ilustrado pela Figura 10 (CLOUDERA, 2019a).

Um D-Stream é criado a partir de uma entrada de fluxo de dados continuado, e submetido imediatamente a um processo de fracionamento (micro-batching), onde cada pequena partição dos dados é sequencialmente submetida ao motor de processamento do Spark, e então, trabalha de forma similar em um processamento com RDDs.

Figura 10: Processamento de lotes em Streaming.
Fonte Site Spark

Fontes de entrada suportadas pelo Spark Streaming são outros projetos RED, sistemas de arquivos distribuídos e redes sociais, e como destino alguns bancos de dados, painéis gráficos e o HDFS. A Figura 11 apresa um esquema com as principais fontes de entrada e de destino para o Spark Streaming. Conforme a arquitetura apresentada na Figura 4, os logs processados advindos do Kafka e armazenados no banco de dados RED Kudu. Outro repositório de entrada que alimenta DSPS é o HDFS para armazenar o modelo de clusterização

Figura 11: Fonte de entrada e saída para o D-Stream.
Fonte: site Spark

A estrutura de D-*Streams* evita problemas tradicionais visto em casos de processamento em streaming. Spark Streaming realiza o cálculo considerando um conjunto de tarefas curtas, determinísticas e sem origem, em vez de trabalhar com operadores contínuos e de origem determinada. Os RDDs são armazenados na memória através de tarefas, que podem ser recalculadas deterministicamente quando necessário. A decomposição dos cálculos em pequenas tarefas permite poderosas técnicas, como por exemplo, realizar a recuperação de dados em paralelo. O diagrama apresentado na Figura 12, obtido em (CLOUDERA, 2019a), apresenta uma visão geral do sistema de processamento do Spark Streaming, conforme descrito em (ZAHARIA, 2016). Note que desta forma, ao dividir os fluxos de dados em lotes de entrada estes são armazenados em memória, e assim, gerando as tarefas que o Spark Streaming precisa para realizar o processamento os lotes.

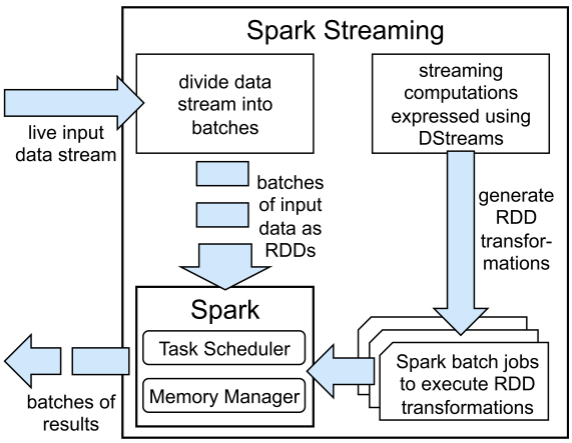

Figura 12: Overview de um sistema Spark Streaming.

Outra abordagem para tratar o fluxo de dados continuado através do Spark Streaming é a biblioteca *Structured Streaming*, que fornece uma estrutura baseada em uma tabela padronizada, onde os atributos de interesse são inseridos continuamente em um formato de dados do tipo Json em uma coluna específica. Apesar de ser um recurso mais atual esta biblioteca foi avaliada, porém descartada, uma vez que o formato D-*Streams* é mais adequado para o processamento de dados de natureza não estruturada, como é o log da URA. Para mais detalhes sobre o *Structured Streaming* consulte (“Structured Streaming Programming Guide – Spark 2.4.0 Documentation,” [s.d.]).

2.3.1.3 Spark Mllib

Esse é o subprojeto do Spark para aprendizado de máquina, com ele é possível realizar o pré-processamento dos dados, o treinamento e a validação dos modelos e a realização de previsões. Além disso, possui ferramentas para auxiliar a implantação dos modelos.

Mllib possui algoritmos para classificação supervisionada, não supervisionada, regressão linear multivariada, regras de associação, redução de dimensões (PCA), decomposição singular (SVD), mineração de texto e outros. Para mais detalhes (“MLlib: Main Guide – Spark 2.4.0 Documentation,” [s.d.]; TAN et al., 2013).

Para o desenvolvimento do DSPS, foram usadas cinco classes pertencentes a biblioteca Mllib: *ML Algorithms*, que contém os algoritmos citados acima, *Featurization*, para realizar o pré-processamento de dados; *Pipelines*, para construção de fluxos inteiros de aprendizado de máquina, incluindo a avaliação e o ajuste dos modelos; *Persistence*, para armazenar o resultado do modelo criado no HDFS; e por fim

Utilities, com funções estatísticas e de álgebra linear utilizadas durante o processo exploratório. Os códigos utilizados estão nos anexos ao final deste trabalho.

2.3.1.4 GraphX

Esta biblioteca permite a criação de grafos através da estrutura distribuída do Spark. Determinados problemas necessitam de um abordagem de modelagem não comportada pelo Mllib, muitas vezes adequadas ao uso de uma formato baseado em grafos, considerando vértices e arestas. O GraphX é um componente do Spark direcionado para esses casos, onde é possível criar grafos em alto nível e com o uso de computação distribuída, com funcionamento garantido pela abstração RDD.

GraphX tem como plano de fundo uma computação realizada através de um conjunto de operadores fundamentais como subgraph, joinVertices e aggregateMessages, advindas do projeto Pregel, conforme Figura 13 obtida em (CLOUDERA, 2019a), que foi renomeado como Apache Giraph (“Giraph – Welcome To Apache Giraph!,” [s.d.]). A principal vantagem do uso do GraphX é a possibilidade do uso de uma variedade de algoritmos e construtores, que visam simplificar a tarefa de se analisar uma estrutura de dados orientada a grafos.

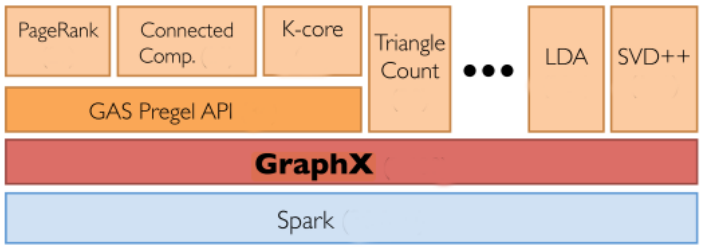

Figura 13: composição do GraphX.

Uma das principais limitações enfrentadas neste trabalho para se usar o GraphX, é seu suporte restrito a linguagem Scala, limitando seu uso. Outro aspecto contrário, é sua necessidade de uso com a abstração do RDD, uma vez que a estrutura de um log é organizada em DataFrames após seu processamento. Note que, essa última característica poderia ser contornada, porém em conjunto com a primeira, foi o suficiente para a busca de uma alternativa.

2.3.1.5 Bibliotecas terceiras

Além destas bibliotecas nativas do Spark, exitem diversas contribuições de projetos terceiros, (“Third-Party Projects | Apache Spark,” [s.d.]). Algumas se tornam muito populares e são incorporadas ao Spark em versões futuras. A spark-excel da crealytics, (“Maven Repository: com.crealytics » spark-excel_2.11 » 0.8.2,” [s.d.]) (GONZALEZ et al., [s.d.]), é muito conhecida.

Diante das restrições apresentadas com o GraphX, uma alternativa ao seu uso que tem ganhado destaque é o GraphFrames (BRADLEY, 2016). Essa foi adotada neste trabalho por ser compatível com a linguagem Python, e suportar uma estrutura de dados baseada no GraphFrames.

Ainda sobre mais contribuições de terceiros que foram relevantes para o desenvolvimento deste trabalho estão o conector SparkStreaming-Kudu, e SparkStreaming-Kafka, ambos podem ser obtidos no repositório de artefatos Maven (“Maven – Introduction,” [s.d.]).

2.4 NiFi

A primeira camada de inserção de dados para o DSPS ocorre em tempo real, sempre em que um log é disponibilizado em um local pré determinada no sistema de arquivos. Este processo foi realizado fazendo uso do projeto NiFi, uma plataforma RED cuja principal função é simplificar a construção, automatização e o gerenciamento de fluxos de dados entre diferentes sistemas. A Figura 14 ilustra a arquitetura sob a qual o NiFi opera, conforme descrito em (“Apache NiFi,” [s.d.]).

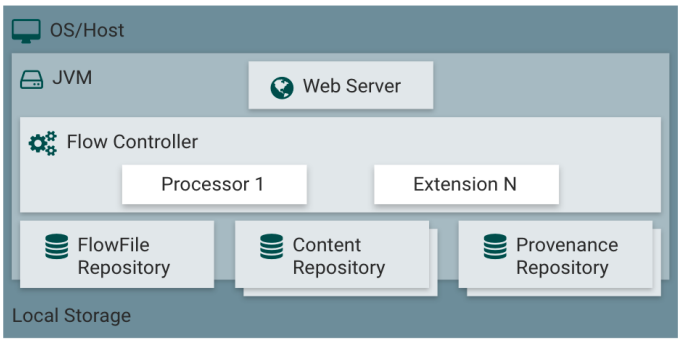

Figura 14: Arquitetura do Nifi. Fonte: site Nifi

Sua execução ocorre a partir de um servidor web em uma máquia virtual Java – JVM, com um modelo de funcionamento simplificado, através de uma interface visual

que permite a construção de fluxos de dados usando operadores denominados *Processor,* e cada operador é conectado através de *Extensions*.

Um *Processor* pode receber dados de uma fonte externa, previamente configurada, ou do resultado de um outro *Processor*. Além dos Processors, existem outros componentes que compõe o NiFi, conforme ilustrado a partir da Figura 14, e para cada um deles é dada uma breve descrição a seguir.

O componente *FlowFile Repository* tem como finalidade o armazenar arquivos *FlowFile* criados, esses arquivos contêm as instruções de execução para cada um dos fluxos de dados.

O *Content Repository* é o local responsável por realizar o armazenamento do conteúdo de todos os *FlowFile* criados.

O *Provenance Repository* é responsável por armazenar todo o histórico de execuções realizadas pelos FlowFiles.

A camada denominada como *Flow Controler,* é o centro operacional do NiFi, e através dela é realizado o gerenciamento dos conjuntos de extensões e plug-ins, permitindo a comunicação do NiFi com outros sistemas.

Sob o aspecto de performance, o Nifi depende do hardware disponível, porém, o NiFi foi desenvolvido para maximizar a utilização do hardware, e assim, operar com mais desempenho a medida que exitam mais recursos.

Uma vez entendido os componentes que são parte do NiFi, é possível compreender como o NiFi compõe o ecossistema de fluxo de dados continuo criado para este trabalho. Conforme já explicado, o NiFi é a camada de entrada para o DSPS, possuindo duas atribuições, a primeira relacionada com o processo de coleta dos log, mensionada no início desta sessão, e a segunda, com a entrega dos logs para a próxima camada, que é realizada a partir da integração com Kafka, outra tecnologia RED baseada em streaming para mensagens, que será introduzida a seguir.

Alguns aspectos finais que caracterizam o NiFi são, sua facilidade para a criação e gerenciamento de fluxos de dados através de uma interface web, sua segurança, uma arquitetura extensível e escalabilidade para computação em cluster. A Figura 15 apresenta o uso do NiFi neste trabalho, monitorando os logs e os entregando para o Kafka assim que disponíveis. Nesta figura é possível avaliar que o componente GetFile,

um Processor, é utilizado para realizar a coleta enquanto o PublishKafka, um outro Processor, o envia ao Kafka.

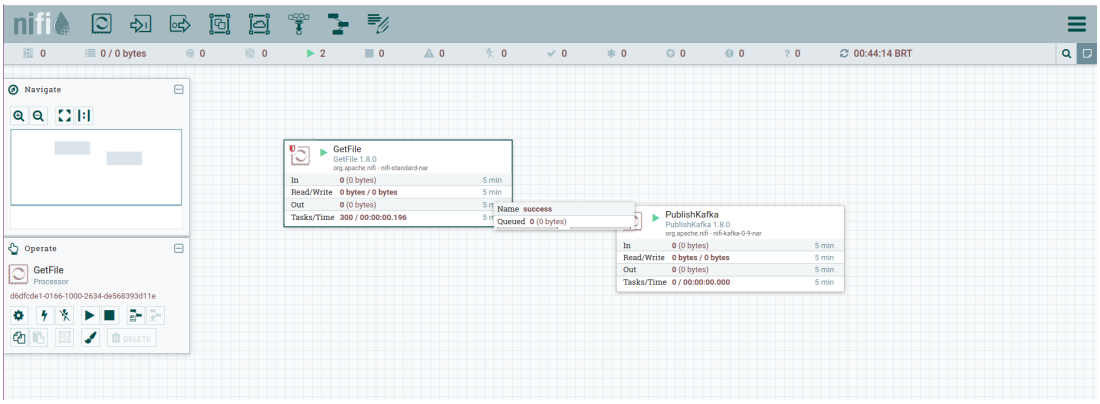

Figura 15: Interface do NIFi, com os componentes utilizados neste trabalho.

2.5 Kafka

O Kafka é um sistema de mensagens de baixa latência, capaz de gerenciar fluxos de dados contínuos através de um sistema de filas. Seu funcionamento segue um modelo do tipo assinante-publicação, que funciona a partir de uma arquitetura distribuída, escalável e resiliente (“Apache Kafka,” [s.d.]).

O problema apresentado neste trabalho é caracterizável como um problema de fluxo de dados contínuos a partir da ocorrência de eventos. Em seu trabalho, (WIATR; SŁOTA; KITOWSKI, 2018), descreve cenários para uso do Kafka, onde demostra que este é uma plataforma ideal, em casos onde se deseja realizar a coleta otimizada da resposta de uma ocorrência a eventos.

Kafka esta ativamente preparado para receber os logs, que são enviados através do NiFi. Uma vez recebidos são tratados como mensagens e direcionados para um Topics, que é um canal personalizado do Kafka, responsável pela gestão da fila caso essa exista. Neste trabalho o Kafka entrega o log para o Spark Streaming, conforme apresentado pela Figura 4.

Algumas outras aplicações para o Kafka são:

- Internet das Coisas (coleta de logs de televisões inteligentes, geladeiras, lavadoras, secadoras, sensores de temperatura, monitores de saúde pessoal e relógios inteligentes).

- Sensores e alarmes diversos. (monitoramento de áreas em fazendas, parques de diversões, florestas, etc. Além do monitoramento de equipamentos complexos como motores e sensores médicos).

- Georreferenciamento. (monitoramento do deslocamento realizado por veículos, assim como avaliação de comportamento em jogos on-line com múltiplos participantes).

- Outras aplicações (coleta de informação espacial de satélites).

As principais atribuições as quais o Kafka é uma alternativa estão relacionadas à leitura, a publicação e o armazenamento de mensagens.

Uma mensagem publicada no Kafka pode vir de diferentes assinantes, fontes de dados, assim como pode ser proveniente do próprio Kafka. Seu sistema de gerenciamento do processamento de mensagens é similar a um sistema de mensagens corporativas, considerando um formato de fila do tipo First-in-First-out – FIFO, onde o primeiro a entrar na fila será o primeiro a ser atendido (BOROVKOV, 1976).

Sistemas de gerenciamento de mensagens possuem uma arquitetura similar à apresentada pela Figura 16, considerando um modelo do tipo assinante-publicação, onde a fonte de dados de origem está representada pelo publicador A, que é responsável por enviar a mensagem com destino ao assinante A. Da mesma forma, a fonte de dados com origem no o publicador B é responsável por entregar a mensagem ao assinante B, e assim por diante, (CLOUDERA, 2019b).

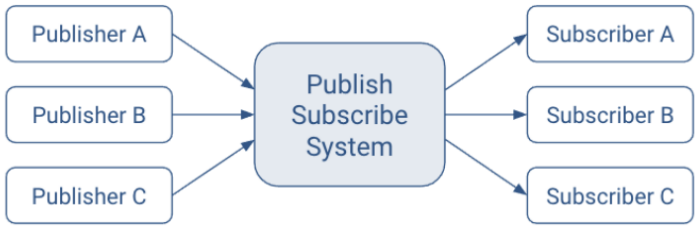

Figura 16: Esquema para um sistema ideal de assinante-publicação.
Fonte: Cloudera Kafka Guide.

O Kafka possui uma arquitetura diferente do que foi introduzida pela Figura 16, nele os responsáveis por publicar as mensagens são denominados *producer (*produtores) e os assinantes *consumers* (consumidores), além disso, Kafka introduziu o conceito de *topic (*tópico), que tem como função agrupar um conjunto de mensagens com as características similares. De forma resumida, um *topic* é uma fila de mensagens recebidas por um ou mais *producers*, e que pode ser lida por um ou mais *consumers*, conforme apresentado na Figura 17 e como pode ser visto, o Kafka gerencia suas filas de mensagens de forma independente, evitando que impactos em um *topic* afete outro.

Cada mensagem trafegada através do Kafka é um registro, que é representado pela plataforma a partir de um par de chave-valor que também incluí informação de data e hora. Esta chave não é um componente necessário e normalmente é utilizada apenas

para identificar mensagens de uma mesma fonte de dados. É possível incluir cabeçalhos para armazenar alguma informação específica. Este tipo de artifício não foi utilizado neste trabalho, pois havia apenas um fonte de dados sendo consumida pelo DSPS. Além disso ao se usar partições é importante ter em mente que não existe garantia de que os registros enviados a diferentes partições mantenham sua ordenação original. Situação que adiciona uma complexidade à lógica de processamento em um modelo de streaming.

O t*opic* é a principal abstração do Kafka, e a forma como os fluxos de dados são organizados a partir de diferentes *producers permite com cada fluxo de mensagem seja entregue baseado em regras específicas.* O funcionamento do Kafka ocorre a partir de quatro interfaces, *Kafka Streams, Kafka Producer, Kafka Producer, Kafka Consumer,* Figura 18, que interagem com os *producers*, *topics* e *consumers*. Essas interfaces são introduzidas a seguir (CLOUDERA, 2019b).

Figura 17: Arquitetura esquemática do Kafka. Fonte Apache Kafka Guide.

- *Kafka Streams:* permite transformar fluxos contínuos de entrada de dados em fluxos contínuos de saída. Isto é feito a partir do consumo as mensagens de cada uma das filas de um ou mais t*opics*, e da mesma forma permite produzir os fluxos de saída para um ou mais *topics*.
- *Kafka Producer:* permite realizar a publicação de um determinado fluxo de mensagens em um ou mais *topics*.
- *Kafka Consumer:* realiza o monitoramento dos *topics*, permitindo a leitura dos fluxos de mensagens através de cada producers.
- *Kafka Connect:* realiza a conexões de baixa latência entre o Kafka e diversos sistemas de banco de dados, e assim, permite que o Kafka trafegue altos volumes de dados.

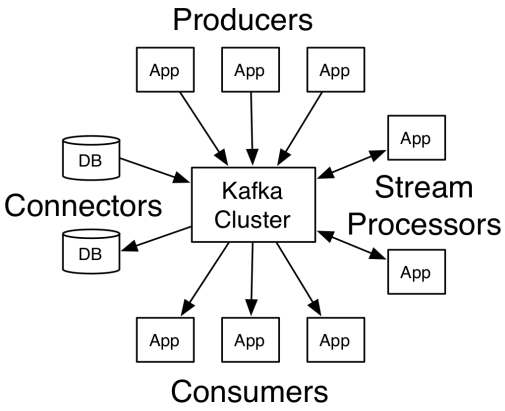

Figura 18: Interfaces de acesso ao Kafka.
Fonte: website Apache Kafka.

2.5.1 Demais componentes do Kafka

Kafka ainda é composto por Brokers, servidores que armazenam as mensagens uma vez enviadas aos *topics* mediante a solicitações dos *consumers*. O Kafka permite a execução de paralelos hospedeiros, considerando um *broker* por hospedeiro. Desta forma, caso um hospedeiro seja interrompido, o Kafka garante que os demais continuem em funcionamento, ou seja, uma plataforma resiliente de funcionamento contínuo, que permite dimensionamento ilimitado de *produces* e *consumers*. As informações de cada fluxo de dados, armazenada em cada um dos *topics*, é replicada em múltiplos *brokers*, como pode ser visto pela Figura 19 (CLOUDERA, 2019b).

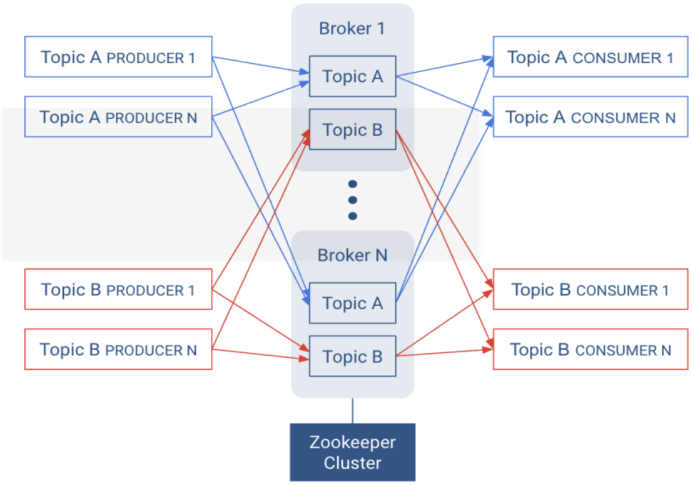

Figura 19: Esquema para os Brokers do Kafka.

Além do entendimento dos brokers, o conceito de partições é relevante ao se fazer uso do Kafka. Neste contexto cada registro é dividido em partições, onde cada uma dessas partições pode ser considerada um subconjunto dos registros dentro de um *topic*. Ao se criar um *topic* duas propriedades são definidas, o número de partições, e o fator de replicação dos registros. Neste trabalho por se ter um único servidor contendo todas as plataformas RED, em um modelo simplificado de operação, utilizou-se uma partição e uma replicação. A Figura 20 ilustra um cenário com N = 2 *brokers* e um fator de replicação M = 2, onde cada um dos *producers (CLOUDERA, 2019b)*.

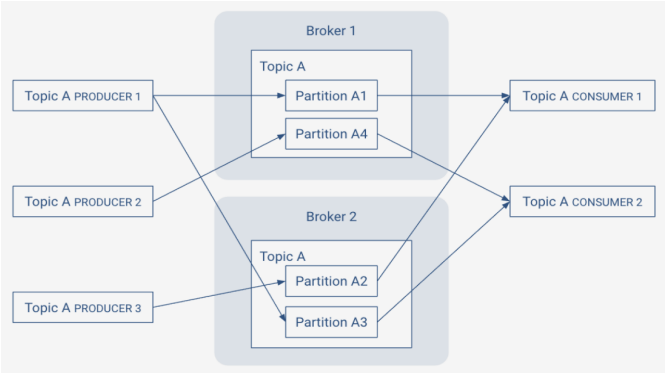

Figura 20: Esquema de funcionamento dos brooker.

2.6 Impala

O Cloudera Impala é um moderno mecanismo Massive Parallalel Processing – MPP através de sintaxe SQL, que foi projetado para operar em conjunto com dados do

ecossistema Hadoop. Sua principal caraterística é a baixa latência ao realizar um alto volume de consultas simultâneas para leitura de dados, que podem estar armazenados em fontes como HDFS, Hbase, Amazon S3, Kudu etc. Um exemplo de arquitetura de uso do Impala com HDFS e o Hbase pode ser visto na Figura 21, (CLOUDERA, 2019c). Impala foi utilizado neste trabalho para permitir o acesso aos logs, uma vez processados (KORNACKER et al., [s.d.]).

Impala é composto por três serviços, o daemon Impala (impalad), responsável por aceitar as consultas de processos enviado pelos clientes e orquestrar as execuções no cluster, além de ser responsável pela execução individualizada de qualquer fragmento de outra consulta realizada por outros daemons. O daemon do Statestore (statestored) é o serviço de publicação-assinatura de metadados do Impala, ele distribuí os metadados através do cluster para todos os processos do Impala. O daemon Catalog (catalogd) é responsável pelo repositório de acesso aos metadados da Impala, através dele os daemons podem executar comandos que são refletidos no Hive Metastore. O fluxo de operação do Impala pode ser visto na Figura 21.

O Impala suporta os formatos de dados como Avro, RC, sequencia de texto, texto simples e Parquet. Além disso, esses formatos podem ser combinados com algoritmos de compressão como gzip, bz2 entre outros.

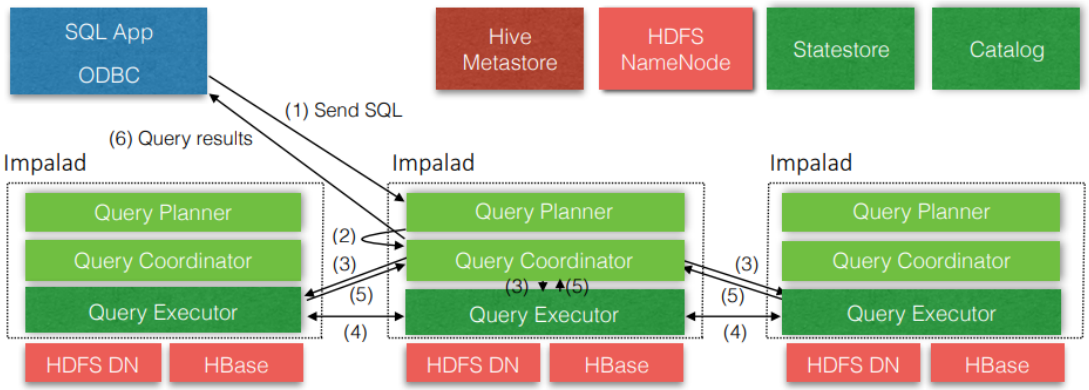

Figura 21: Fluxo durante o processamento de consultas através do Impala.

Uma consulta realizada a partir do Impala é executada em cada DataNodes do cluster, distribuída através dos nós disponíveis para o Impala. Não serão discutidas questões sobre a configuração do impala, independente de qual seja o local de armazenamento de dados, HDFS, Hbase ou Kudu. Sendo o Impala um componente da

distribuição Cloudera, isto o torna facilmente gerenciado pelo *Cloudera Manager*[5] *(CLOUDERA, 2019c; ROAD, [s.d.])*

O Impala inclui uma sintaxe para recursos comuns de manipulação de dados, simplificando de forma geral o uso do ecossistema Hadoop, através de comandos SQL como, SELECT, WHERE, GROUP BY, ORDER BY, WITH, IN, BETWEEN, entre outros. Funções internas para processamento de sequências, números, datas, funções agregadas, sub-consultas e operadores de comparação, também estão presentes. Além disso o Impala permite o uso de funções definidas pelo próprio usuário.

O uso do impala no DSPS foi através do Spark, para inserção de dados, ao final do processamento em formato streaming. Além disso durante o processo de cálculo para do centroides do K-Means, o Imapala foi usado para acessar dados já armazenados no Kudu. Um fato interessante que antes do Kudu, o uso Impala é altamente associado aos formatos de dados Parquet e Avro em conjunto com armazenamento no HDFS, e menos usado com o Hbase. As principais vantagens no uso do Impala comparado aos demais projeto RED de mesma natureza, ou seja, consulta de dados são:

- Processamento nos nós de datanodes, evitando assim demasiado tráfego de rede;
- Metadados unificado;
- Dispensa da necessidade de conversão entre diversos formato de dados, reduzindo processamento;
- Dados disponíveis direto na fonte de origem, reduzindo processo de extração, transformação e carga.

Mais detalhes sobre acesso a dados armazenados no ecossistema Hadoop, podem ser vistos em trabalhos como (FLORATOU; MINHAS; OZCAN, 2150)

2.7 Kudu

O armazenamento no ecossistema Hadoop costuma ser realizado de duas formas, a primeira é caracterizada por dados com puco atualização, ou seja, de natureza estática, enquanto a segunda demanda acesso aleatorizado de maneira eficiente. Para o primeiro caso é comum realizar o armazenado no HDFS, a partir de formatos binários como Avro, (“Welcome to Apache Avro!,” [s.d.]) ou Parquet, (“Apache Parquet,” [s.d.]). No entanto, para o segundo cenário, o HBase (“Apache HBase – Apache HBase™

[5] Gerenciado de cluster desenvolvido pela empresa Cloudera.

Home," [s.d.]; GEORGE, 2011) ou Apache Cassandra, (CARPENTER; HEWITT, [s.d.]), que permitem leitura e escrita considerando baixa latência, mas mesmo neste cenário não são melhores que os formatos anteriores, quando considerado aplicações analíticas SQL, ou de aprendizado de máquina.

Kudu é um sistema de armazenamento colunar que considera dados estruturados, projetado e implementado para operar considerando altas taxas de transferências. Fornecendo um sistema de armazenamento de acesso sequencial, como HDFS, e um sistema de acesso aleatório de baixa latência, como HBase ou Cassandra. Sendo um modelo intermediário entre ambos cenários.

Kudu distribui os dados através de particionamento horizontalizado, replicando cada uma das partições, e usando o algoritmo Raft para escolha dos servidores líderes e seguidores. Kudu foi desenvolvido como parte do ecossistema Hadoop e suporta diferentes modos de acesso, como os motores Impala e Spark, e o paradigma MapReduce (LIPCON; CRUTCHER, 2015). Além de possuir características como resiliência, escalabilidade e processamento distribuído ao longo de um cluster de computadores.

Cada tabela criada no Kudu necessita de uma chave primária, conforme o modelo de dados de um banco de dados relacional. Outras características do Kudu são: Processamento altamente performático similar a um modelo OLAP[6], considerando um volume de dados comportados em memória. Integração com ecossistema Hadoop; Forte integração com o Impala. Simplificação da arquitetura, mediante seu modelo de execução de trabalhos sequenciais e aleatórias simultâneos, entre outras. Mais detalhes podem ser vistos em (CLOUDERA, 2019d).

A Figura 22, ilustra algumas tabelas armazenadas no Kudu, para a criação do processo de fluxo de dados gerados no DSPS. Neste caso a interface de acesso é o Imapal-shell.

[6] Online Analytical Processing, do inglês processamento analítico online, é um método tradicional para acessar e visualizar dados, que permite realizar análises com flexibilidade e performance.

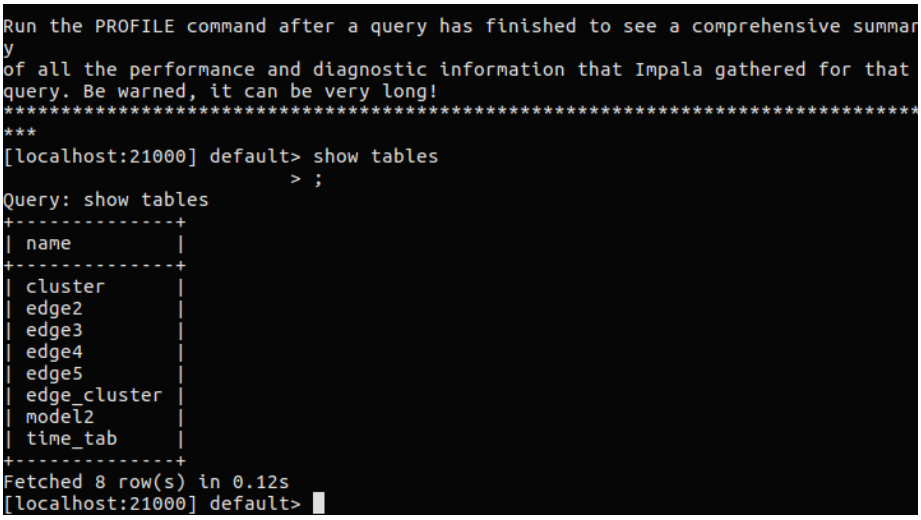

Figura 22: Tabelas armazenadas no Kudu.

2.7.1 Kudu e alguns tópicos relevantes

Algumas características do Kudu merecem um destaque, dentre elas está o algoritmo Raft, que auxilia a escolha dos nós que terão papel de líder entre cada servidor *master* e *tablets*.

Tabelas:

As tabelas são os locais onde os dados são armazenados. Estas possuem uma estrutura similar a encontrada em banco de dados, além disso, possuem esquema e chave primária. Para garantir uma estrutura resiliente, cada tabela é dividida pequenos em segmentos denominados *tablets.* Note que, este processo é realizado a partir das características que foram definidas para o campo da chave primária.

O algoritmo Raft:

O algoritmo Raft é utilizado para escolher quem terá papel de lider, no contexto dos servidores *masters* e *tablets*, além disso, auxilia no processo que determina se uma operação de escrita de dados em uma tabela ocorre com sucesso ou não.

Tablet:

Um *tablet* é um segmento de uma tabela, similar ao conceito de uma partição. Um *tablet* é replicado em servidores denominados *tablet servers*. Cada uma dessas réplicas pode assumir a função de um líder. As réplicas de um *tablet* permitem a realização apenas etapas para leitura dos dados.

Master:

O *master* controla todos, *tablets*, *tablet server*, a *Catolog Table* e qualquer outro metadado relacionados ao cluster, só pode haver um nó master. Em caso de um nó lider desaparecer, um novo nó master é escolhido a partir do uso do algoritmo *de Raft. E a*inda neste cenário é importante destacar que *o nó master* é responsável pela coordenação das operações envolvendo os metadados.

A seguir a Figura 23 ilustra o modelo de funcionamento para os servidores Master e Tablets, assim como ilustra como o Kudu escolhe os nós líderes e seguidores através do algoritmo Raft. Note que, na figura em questão os líderes estão identificados em tom amarelo, enquanto os seguidores estão em cinza. Figura retirada de (CLOUDERA, 2019d).

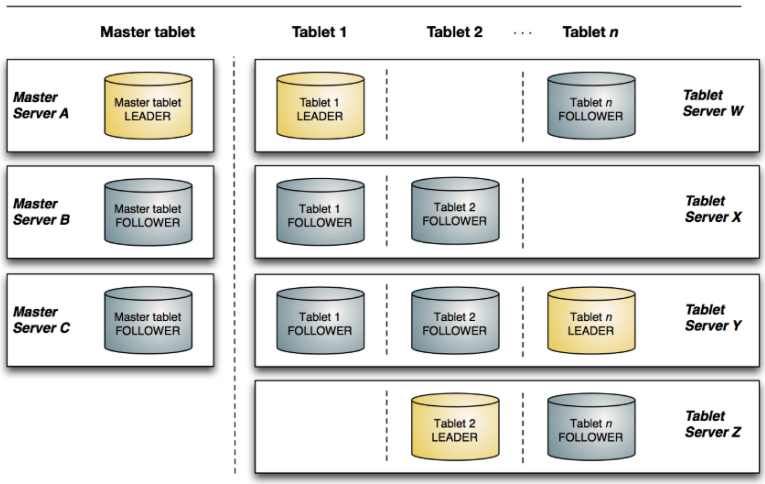

Figura 23: Arquitetura de servidores Master e Tablets do Kudu.

2.7.1 Aplicações Kudu para problemas interativos

Fluxo de dados disponibilizados em tempo-quase-real:

Situações onde existe inserção de dados em tabelas, onde as mesmas são disponibilizadas para leituras assim que estas chegam, considerando um sistema com alta taxa de escrita e leitura. Conforme é o ocorrido pela aplicação desenvolvida neste trabalho.

Séries temporais:

Dados que consideram uma sequência de observações codificados conforme o momento de sua ocorrência, são definidos como séries temporais. Esse modelo de dados tem como principal aplicação a realização de previsões futuras, tomando com base comportamentos e padrões históricos. Kudu por armazenar dados em formato colunar contribuí em um processo de análise com múltiplas séries temporais, isto porque, em

uma requisição a um grupo de dados, a busca é realizada limitadamente apenas a cada coluna envolvida. Diferentemente de uma arquitetura tradicional para um banco de dados relacional, que ao sofrer uma requisição de mesma natureza, necessita varrer todas as colunas, pois seu processo de busca ocorre linha por linha.

Modelagem Preditiva:

Cenários que requeiram dados de entrada para modelos matemáticos, estatísticos ou computacionais que demandem alterações constantes do seu processo de aprendizado, ou qualquer outra situação externa, que justifique algumas alterações na modelagem.

2.8 Superset

Ao se tratar de todos os temas relacionados com análise de dados, geralmente, aspectos envolvendo a visualização de resultados são negligenciados. Com a criação de plataformas dedicadas a visualização de dados, o assunto passou a receber mais atenção inclusive no meio acadêmico (ECKERSON, 2011). Como exemplo é possível destacar o software Tableau, que reúne uma grande rede de usuários. Sua principal característica é a facilidade na criação de gráficos organizadso em painéis, a partir de diferentes fontes de dados (MURRAY, [s.d.]).

Durante a última década os projetos a respeito do assunto de Big Data (plataformas RED) mantiveram sua atenção em questões como latência de processamento, volume de dados, formatos para armazenamento, integração, etc. Nenhuma iniciativa relevante foi criada no contexto da visualização de dados. Isso mudou a partir do Superset, que é um projeto pertencente a Apache Software Foundation, desenvolvido em Python, que tem como principal característica auxiliar a construção de painéis, a partir de gráficos interativos e de interface amigável. Para se dar uma dimensão da capacidade de visualização do Superset, a figura Figura 24 ilustra um painel com algumas possibilidades gráficas permitidas. Esta imagem foi extraída da página web oficial do Superset.

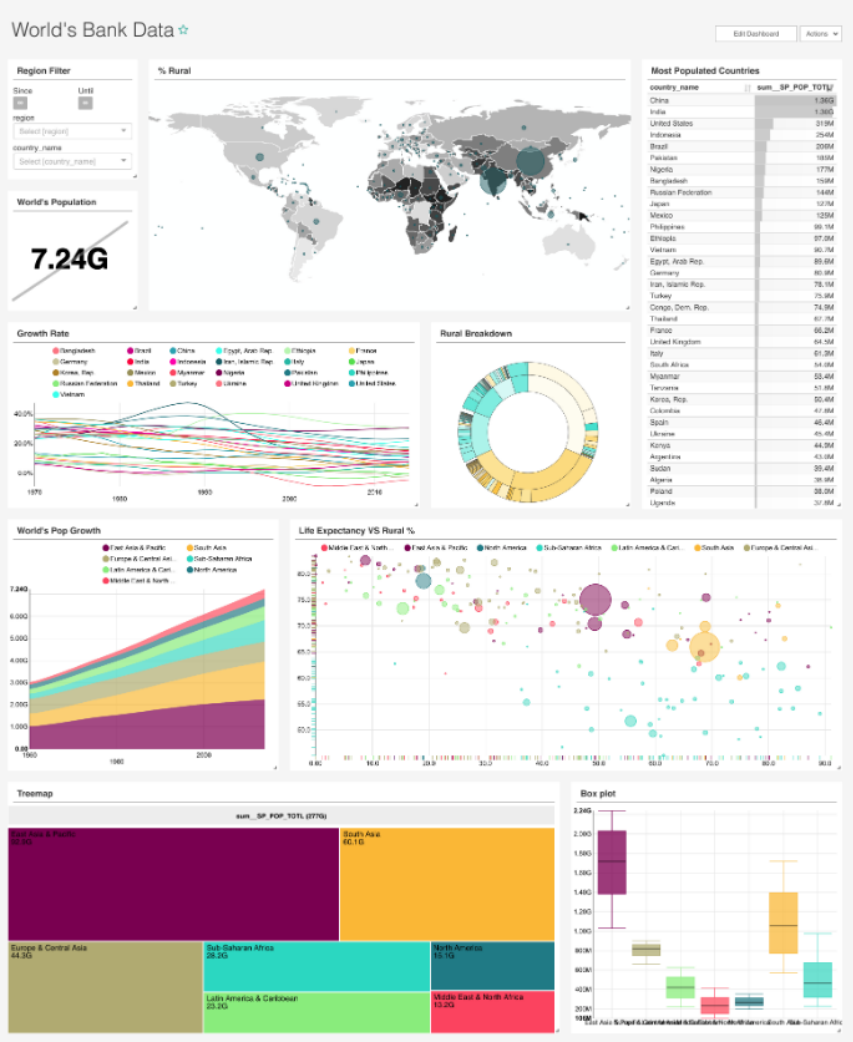

Figura 24: Imagem exemplo de alguns gráficos do Superset

Além de possuir diversos modelos preconcebidos o Superset possui segunda característica, uma interface para realização de consultas analíticas a partir do uso de SQL. Essa interface é o *SQL Lab*, que pode se conectar a diversos bancos de dados, e além disso, disponibiliza uma interface facilitada para coleta dos resultados, incluindo suporte a longas consultas e possibilidade de navegação através dos metadados das tabelas (“Apache Superset (incubating) — Apache Superset documentation,” [s.d.]).

Sobre as possibilidades de conexão do Superset, este trabalho desenvolveu uma arquitetura baseada em apenas uma fonte de dados, o Kudu, introduzido na sessão 2.7. Para que o Superset se conecte ao Kudu é necessário utilizar o motor Impala, introduzido na sessão 2.6. A biblioteca que realiza a conexão é a impyla, listada na Tabela 2, conjuntamente com todas as demais bibliotecas de conectividade.

Tabela 2: Conexões extras para o Superset

Database	pypi package	SQLAlchemy URI prefix
MySQL	pip install mysqlclient	mysql://
Postgres	pip install psycopg2	postgresql+psycopg2://
Presto	pip install pyhive	presto://
Hive	pip install pyhive	hive://
Oracle	pip install cx_Oracle	oracle://
sqlite		sqlite://
Snowflake	pip install snowflake-sqlalchemy	snowflake://
Redshift	pip install sqlalchemy-redshift	redshift+psycopg2://
MSSQL	pip install pymssql	mssql://
Impala	pip install impyla	impala://
SparkSQL	pip install pyhive	jdbc+hive://
Greenplum	pip install psycopg2	postgresql+psycopg2://
Athena	pip install "PyAthenaJDBC>1.0.9"	awsathena+jdbc://
Athena	pip install "PyAthena>1.2.0"	awsathena+rest://
Vertica	pip install sqlalchemy-vertica-python	vertica+vertica_python://
ClickHouse	pip install sqlalchemy-clickhouse	clickhouse://
Kylin	pip install kylinpy	kylin://
BigQuery	pip install pybigquery	bigquery://
Teradata	pip install sqlalchemy-teradata	teradata://
Pinot	pip install pinotdb	pinot+http://controller:5436/ query? server=http://controller:5983/

2.9 Aspectos sobre a arquitetura do DSPS

Neste capítulo foram introduzidas as tecnologias RED que compuseram o DSPS, apresentadas em sessões. A partir do que foi apresentado neste capítulo é possível compreender a arquitetura apresentada pela Figura 4. Possibilitando assim, compreender futuramente o fluxo de dados para avaliar a experiência do cliente através dos logs.

É interessante destacar que a abordagem para processar dados em fluxo contínuo, foi descrita por (Miloslavskaya & Tolstoy, 2016), como um, dos três tipos de processamento de dados relativo a Big Data. Denominado como, *Stream processing in hard real-time.* Duas das características que o destacam são, o processamento de dados em tempo real e armazenamento de resultados apenas ao final do fluxo. Exatamente como foi feito neste trabalho. Os outros dois modelos para Big Data são, *Batch processing in pseudo real or soft real-time e Hybrid processing.*

3. A Experiência do Cliente e o autoatendimento

O capítulo 1 contextualizou o problema que motivou essa pesquisa através da necessidade de uma empresa brasileira, no setor de telecomunicações, que desejava

conhecer o padrão de comportamento dos seus clientes ao usar o seu canal de autoatendimento URA, e avaliar as percepções sobre o serviço.

Um dos principais aspectos a respeito do processo de avaliação é que este não deve ser realizado a partir de uma abordagem direta ao cliente, ou seja, através de um questionamento direto, que é a forma tradicional de se realizar este tipo de levantamento pois acredita-se que uma abordagem desta natureza produz resultados enviesados.

Um dos primeiros aspectos identificados foi a necessidade de se modelar os possíveis caminhos a serem percorridos dentro da URA. Porém, esses caminhos são alterados com uma alta frequência, mediante as inúmeras necessidades internas ou externas a companhia, desta forma, optou-se pode modelar a experiência de usabilidade dos clientes ao utilizarem o serviço de autoatendimento da URA.

A cada contato realizado com o serviço de atendimento com consumidor final (SAC), inicialmente o cliente é conectado com uma URA compatível com seu perfil de cliente, onde esta oferece opções de serviços para a realização do autoatendimento baseado no serviço contratado, seja ele telefônico, dados, ou TV por assinatura. Caso a URA não seja capaz de solucionar a demanda de serviços do cliente, este é redirecionado para uma célula com atendentes, para a realização de um processo personalizado.

Ao final da etapa envolvendo a URA, o registro das interações que o cliente realizou são armazenados em um arquivo de registro, um log do sistema. Nestes arquivos estão as informações relacionadas a chamada, assim como ao cliente, sendo a única fonte de dados disponível para se modelar o comportamento de utilização em uma URA.

Em uma grande empresa de telecomunicações, diariamente milhares de clientes entram em contato com serviço SAC para tratar inúmeros assuntos. E cada desses contatos é interceptado inicialmente por uma URA, produzindo um grande volume de logs de natureza não estruturada, de difícil coleta e interpretação. Mediante esses fatores e a necessidade de responder a alguma questão especial no menor tempo possível, ou seja, em tempo quase real, este trabalho optou por desenvolver um sistema exclusivamente a partir de tecnologias para Big Data, para processar e modelar tais logs, tomando como base nas tecnologias introduzidas no capítulo 2. E combinadas para produzir um sistema único, definido como *Data Streaming Processing System* - DSPS, ilustrado a partir do esquema proposto a partir da Figura 2 e da Figura 4.

3.1 Cliente no centro do negócio

Diversos setores do mercado têm dedicado seus esforços tentando se adequar a uma nova realidade mercadológica, onde a percepção dos clientes passou a influenciar

diretamente a estratégia das empresas, este atualmente vem sendo colocado no centro do negócio. Isto é consequência direta do aumento da acesso a informação, provocado pela massificação da internet, que atualmente está presente em diferentes tipos de dispositivos, como eletrodomésticos, televisões e principalmente nos telefones pessoais.

Assim como os demais setores, as empresas de telecomunicações redirecionam seus esforço na direção de uma melhor relação com seus clientes. A preocupação em como seus canais de atendimento têm sido utilizados representam parte desse novo modelo. Por muitos anos a estratégia de empresas desse setor no mercado brasileiro restringia-se a realização de investimentos em suas redes de comunicações, que mesmo assim se utilizava de tecnologias atrasadas quando comparadas com outros países.

É interessante destacar que o setor de telecomunicações brasileiro foi responsável por criar, ou ao menos fomentar, parte dos desafios que atualmente necessita solucionar. Ao estimular o consumo de novos serviços relacionados ao consumo de dados, como por exemplo a internet de alta velocidade e a internet móvel, com a intenção de aumentar suas receitas, as empresas criaram um efeito inverso. O aumento no consumo desses serviços possibilitou o uso de aplicativos para comunicação via mensagens de áudio e chamadas de vídeo, gerando uma redução no consumo de seus serviços vendidos de alto valor agregado, como o *Short Message Service* – SMS e as intermediações de chamadas de longa distância, loais e internacionais.

A redução das receitas e o aumento de custos envolvendo o excessivo uso rede de dados, provocou queda na qualidade dos serviços, que foi rapidamente percebida pelos clientes. Esse efeito culminou em uma alta de reclamações no órgão regulador e em medidas judicias contra as empresas.

Mediante as mudanças no perfil do consumidor, motivadas pelo novo cenário de acesso a informação, as operadoras de telecomunicações iniciaram um processo de revisão de suas estratégias, concentrando seus esforços em digitalização e experiência do cliente. Desta forma aumentando a relevância dos canais de autoatendimento, dentre eles a URA, alvo desse trabalho.

3.2 Modelo de atendimento SAC

A estrutura convencional utilizada para atendimento receptivo é baseada no modelo SAC, com três etapas. A primeira é o autoatendimento, realizado através de uma plataforma URA de autoatendimento. A segunda é o atendimento personalizado, via atendimento de uma operação de call center, esta etapa ocorre imediatamente após o cliente ser submetido a primeira. Ou seja, este canal é apresentado somente a clientes que não tiveram suas demandas solucionadas no autoatendimento com a URA. E por fim, oferecido em paralelo aos canais telefônicos, também são disponibilizadas

estabelecimentos presencias, em formato de lojas. Porém este último modelo tem sido gradativamente reduzido mediante a necessidade constante das companhias em reduzir sues custos. Em grande parte este tipo de canal vem sendo substituídos por plataformas digitais, como aplicativos para telefones móveis e páginas de conforme apresentado no relatório (LEGGETT; COM, 2017).

Além das lojas, o processo de digitalização dentro das instituições vem impactando o próprio modelo SAC, que tem sofrido com a redução das operações de call center, uma vez que esta parte concentra o maior investimento, quando comparada a URA.

Uma ilustração do fluxo telefônico de um modelo de atendimento SAC pode ser visto representado pela ilustração da Figura 25, em operadoras de telefonia frequentemente usa-se esse modelo, replicado-o em função da quantidade de produtos para os quais a companhia ofereça atendimento.

Como já citado, este trabalho se concentra em uma única URA de uma operadora de telefonia, dedicada ao atendimento de clientes da modalidade telefonia móvel pré-pago. Porém sem perda de generalidade, tanto a metodologia para medição da experiência do cliente e a arquitetura para DSPS, apresentada no capítulo 2, são extensíveis para qualquer tipo de URA.

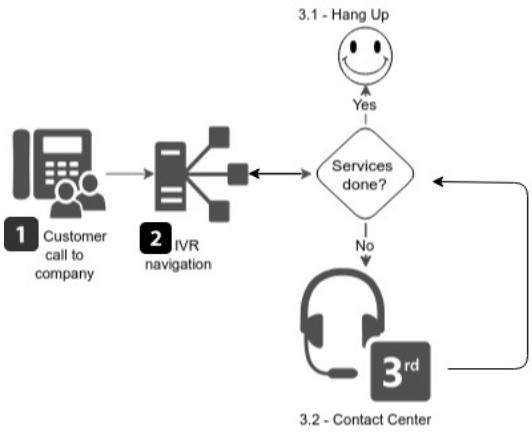

Figura 25: Fluxo resumido representando o modelo SAC.

Enquanto o autoatendimento através de URAs é algo consolidado, mas carece de trabalhos relacionando-o aos conceitos de experiência do cliente e Big Data.

3.3 URA

A primeira utilização para URAs foi em um banco, por volta dos anos 70. Na época a aplicação visava permitir que os clientes verificassem seus saldos em conta-corrente. Com passar dos anos a URA foi ganhando outras funcionalidades, atualmente é praticamente indispensável no atendimento empresarial, se consolidando como uma plataforma inteligente e versátil, porém nem sempre eficaz. Outras finalidades onde a URA tem obtido destaque são vendas e levantamento de dados para pesquisas (SCHRODER; JOHNSON, 2009).

Dados de relacionamento com o cliente, em geral, ainda são poucos explorados em sua plenitude. Com o crescimento do uso de mídias sócias, os trabalhos que abordam este tema, o fazem a partir de redes de relacionamento, dessa forma, o uso dos dados gerados internamente nas empresas, ainda são pouco exploradas. Por exemplo, logs de URA são classificadas como *dark data,* um termo usado em casos onde o propósito de uso não é estabelecido, mas entende-se que existe um alto potencial para geração de valor em fazê-lo. Em seu trabalho, (CIMIANO, 2018) reforça que tanto dark data, quanto dados não estruturado, são responsáveis por 80% da informação atualmente disponível em uma empresa.

```
;;; Host: URACTX11RJOED
;;; Base path: D:/NVP/logs/../callLogs/master/UraCampanha
;;; Program: NVP
;;;
;;;;;;;;;;
;;;;;;;;;;
start{
        TIME                = xxxx
        BEGIN_TIME          = xxxx
        SESSION_ID          = xxxx
        RECORD_NUMBER       = 1
        CALL_SEIZURE_TIME   =  xxxxxxx
        BROWSER             =  xxxx
        BROWSER_ID          = 1687565

.
.
.

GRAMMAR_LABEL       = xxxxPromoxxxx
        PROMPT_DELAY        =xxxx
        PROMPT_DURATION     = xxxx
        LAST_USER_INPUT_TIME    =xxxx

        STATE_DURATION      = xxxx
}end
;;;;;;;;;;
;;;;;;;;;;
start{
        TIME                =xxxx
        BEGIN_TIME          = xxxx
        SESSION_ID          = xxxx

.
.
.
}end
```

Figura 26: Parte editada de um log de uma URA.

Um log da URA a qual estamos utilizando apresentam características conforme a apresentada na Figura 26, que retrata a estrutura de uma chamada, registrada em um log. Neste caso a figura foi alterada, mas é possível ver um log completo no Anexo 1. Note que, nesta figura, cada passo é registrado através de delimitadores representados através dos termos *start{*, e *}end*. E toda a informação de um menu. São blocos com as

informações do menu, que foi acessado pelo cliente. E para uma visão mais detalhada, um exemplo com um log inteiro pode ser visto no Anexo 1, página 70.

Ao avaliar todo um log de uma chamada, foram mapeados as informações de maior relevância e com potencial para explicar a experiência do cliente ao mesmo tempo em que fosse possível avaliar a URA. E estes atributos estão listados a seguir na Tabela 3. Não necessariamente todos listados na tabela foram utilizados. Alguns foram transformados ou serviram como uma base para a construção e avaliação de outras medidas. Porém são estes atributos que fundamentaram o processo decisório.

Field Name	Field Description
Key	Primary key
Session id	Call key
Tel number	Identification number
Step	Position of Menu in Navigation
Source	Name of origem Menu
Destiny	Name of origem Menu
Relationship	Name of response chosen by customer
Day	Day of call
Month	Month of call
Year	Year of call
Time	Time of call
State duration	Time of duration in each Menu
Prompt delay	Time of delay to start audio in each Menu
Prompt duration	Tempo de duração total
Event Date	Date of call

Tabela 3: Tabela de atributos disponíveis nos log, essenciais para experiência do cliente.

Para melhor ilustrar, a Figura 27 apresenta um diagrama simulando um processo de navegação de uma chamada em um atendimento por uma URA. Neste exemplo são considerandos quatro passos, onde em cada um dos menus existe uma estrutura completa de informações delimitadas pelas expressões *start{*, e *}end* citados anteriormente.

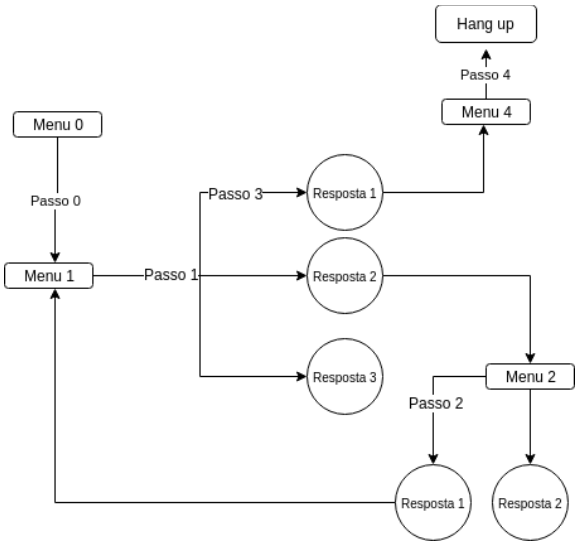

Figura 27: Diagrama de navegação de uma chamada.

No cenário da Figura 27, durante a transição realizada o passo 2, note que, a escolha pela resposta de número 1, mais uma vez encaminhou o cliente ao menu de número 1. A possibilidade de revistar menus, foi uma das características que influenciaram ao se pensar uma chamada como um grafo matemático, em vez de uma árvore de decisão.

3.3.1 Aspectos do modelo de navegação em uma URA

Ainda tomando como referência a Figura 27, ao analisá-la sob o aspecto das transições entre cada um dos menus, nota-se que, apenas o último menu escolhido influenciará na escolha do próximo. Essa propriedade, somada a característica já ciada, da possibilidade de revisitação dos menus, permitem que cada chamada possa ser modelada de acordo com uma cadeia de Markov, um caso particular de um processo estocástico.

Segundo (ROSS, 1996), um processo estocástico é uma família de variáveis aleatórias $\{X_t\}_{t\in T}$, definida sobre um espaço de probabilidade, indexado pelo parâmetro t, onde $t\in T$. $\{X_t\}_{t\in T}$ é uma função de dois argumentos $\{X(t,\omega), t\in T, \omega\in\Omega\}$, onde Ω é o espaço amostral. Ou seja, dado um t fixo, tem-se $X(t_0,\omega)=X_{t_0}(\omega)$ é uma variável aleatória denotada por, pois. Da mesma forma ao fixar-se $\omega=\omega_0$ tem-se $X(t,\omega_0)=X_{\omega_0}(t)$ é uma função que só depende de t, ou uma realização no processo. Para t e ω fixados $X(t,\omega)$ é um número real.

Ao modelar a URA através de uma cadeia de Markov, cada menu é um espaço de estado e as conexões entre menus são representadas pelas probabilidades de transição da cadeia.

Uma vez que os logs processados são equiparáveis a observar realizações de eventos de uma cadeia de Markov, cada um desses eventos são representados a partir de grafos orientados, onde, menus são os vértice a transição entre cada passo, conectando a outro menu são as arestas. Em um cenário hipotético onde todas as combinações de escolhas possíveis fossem realizadas seria possível determinar a estrutura principal da URA, ou seja, da cadeia de Markov.

Outra abordagem para o mesmo cenário é imaginar a URA como um grafo, e cada possível conexão entre menus, como arestas com propriedades representadas por probabilidades. Desta forma, a cada nova chamada, um subgrafo seria gerado e registrado no log e assim, da mesma forma que anteriormente em um cenário hipotético, caso todas as combinações de chamadas, acessando todos os menus, então, seria possível recriar a estrutura principal da URA.

3.3.2 Aspectos gerais do processo de navegação em uma URA

Uma etapa importante a respeito do processo de navegação, e que merece ser detalhado, é o evento que determina o momento de término de uma interação entre um cliente e a URA. A finalização de uma chamada decorre conforme uma dentre as quatro formas descritas a seguir.

1. O cliente realiza todas as etapas que lhe são oferecidas durante o fluxo de navegação, soluciona suas demandas e desconecta a ligação;
2. O cliente está no meio de uma etapa de navegação entre os menus, então, desconecta o contato com a URA.
    1. A desconexão pode ocorrer pois o cliente encontrou o que buscava e não deseja mais continuar navegando;
    2. A desconexão pode ocorrer pois o cliente não obtém nenhuma resposta, mediante as opções de menus que lhe são apresentados. E além disso, não conseguiu ser transferido para o call center;
3. Ocorre uma desconexão involuntária da ligação mediante a um erro;
4. Após navegar por uma sequência de menus, porém sem sucesso na resolução que motivou o contato, o cliente é transferido para o segundo nível de atendimento, em uma célula de call center, como apresentado no fluxo da Figura 25. Desta forma, o atendimento realizado pela URA é finalizado, porém a demanda do cliente não foi atendida e o processo de atendimento continua em um nível de maior personalização, e com isso de maior custo para companhia.

Dentre os motivos de desconexão apresentadas acima, o número 2 é o único que se destaca por apresentar uma ramificação em duas opções, onde a primeira tem como resposta um evento de sucesso, do ponto de vista da experiência do cliente, equanto o

segundo de insucesso. Já o motivo de número 3 é definitivamente destaca-se negativamente como um ponto de atenção, uma vez que não foi uma escolha do cliente e sim um erro no processo de atendimento. E por fim, os motivos de finalização de números 1 e 4, mesmo com naturezas diferentes, não podem ser tipificados de forma negativa. No de número 1, a URA atendeu ao seu propósito, mesmo que sem que isso tenha sido registrado corretamente. Enquanto que no motivo de número 4, mesmo sem satisfazer a demanda do cliente a URA possibilitou que este fosse transferido para modelo mais adequado e com maior probabilidade de sucesso no atendimento.

É interessante destacar que, em ambos os sub casos de motivos representados pelo item de número 2, o cliente apresentou maiores chances de sucesso no primeiro, onde a chamada foi terminada corretamente.

Interpretar a experiência de um cliente somente através da forma a qual um atendimento foi finalizado pode gerar muitas incertezas a respeito da eficacia da plataforma no autoatendimento.

Descartando o processo de abordagem direta ao cliente, dentre as abordagens mias seguras de avaliação da plataforma da URA, via experiência do cliente, avaliar todo o processo de navegação é sem dúvida uma das formas mais seguras e precisas de se inferir a respeito do processo.

4 O sistema DSPS e a metodologia para avaliação da experiência do cliente.

Este capítulo detalha a metodologia proposta para medir a experiência do cliente, assim como o algoritmo que descreve o fluxo de dados. Note que, durante o processamento de dados faz uso da metodologia, por esse motivo foram apresentados juntos neste capítulo, porém, é importante destacar que ambos podem ser usados de forma distintas.

4.1 A metodologia para medição da experiência do cliente

A definição de (NASH; ARMSTRONG; ROBERTSON, 2013), citada no capítulo 2, define o conceito de experiência do cliente, mas não estabelece nenhuma métrica.

Com base nesta definição propõe-se um modelo com base em todos pontos de contato, que será apresentado a seguir em tópicos organizados em duas etaps. A primeira visa definir um modelo baseado em grafos e quais atributos devem ser destacados. Enquanto a segundo propõe o uso de um algoritmo de agrupamento (cluster) para identificar clientes de comportamento similar, visando determinar uma abordagem adequada para cada grupo criado.

Metodologia Geral

Etapa 1:

1. Um período pré-determinado de tempo para realizar a medição;
2. Ordenar o fluxo de canais aos quais o cliente mantém contato, no período definido, permitindo assim conectá-los em ordem cronológica.
3. Assumir uma estrutura baseada em um grafo orientado, G=(V,A) para modelar todos os eventos de contato com os canais;
4. Extrair os atributos:
    1. Contatos que o cliente realiza com cada um dos canais;
    2. Tempo desprendido pelo cliente em cada um dos contatos;
    3. Número de conexões de saída que um canal realiza com todos os outros (graus de saída de cada um dos vértice);

4. Número de conexões de entrada que um canal recebe vindo de todos os outros (graus de entrada de cada um dos vértice).

Etapa 2:

A partir da etapa anterior:

1. Para cada cliente, determinar o valor médio dos atributos:
    1. Quantidade de contatos que o cliente realiza com os canais;
    2. Tempo total utilizado nos canais;
    3. Número médio de graus de saída de cada canal;
    4. Número médio de graus de entrada de cada canal.
2. Definir um algoritmo de agrupamento;
3. Determinar os parâmetros para o algoritmo de agrupamento;
4. Interpretar os grupos com base na natureza do atendimento e dos dados.

As duas etapas acima determinam a metodologia de forma generalista para medir a experiência do cliente. Adaptá-la para o modelo da URA é uma tarefa bastante direta, cada canal de atendimento torna-se um menu possível de ser navegado em um contato com a URA.

1. O período de tempo é definido por uma chamada, ou seja, cada log determina um período.
2. Numerar cada menu de acordo com sua aparição no log;
3. Assumir um grafo orientado, G=(V,A) para a estrutura de menus, onde cada menu é um vértice e cada transição entre menus são as arestas;
4. Extrair os atributos listados a seguir, de cada um dos blocos de menus, dentro de um log:
    1. Posição do menu acessado pelo cliente;
    2. Tempo desprendido em cada um dos menus;
    3. Número de conexões de saída que cada menu realizou em relação aos outros menus (graus de saída de cada um dos vértices);
    4. Número de conexões de entrada que cada menu recebeu de todos os outros menus (graus de entrada de cada um dos vértice).

Etapa 2:

Neste trabalho foi utilizado o algorítimo K-Means para agrupar as chamadas em clusters, porém qualquer outro algoritmo se enquadra no método proposto.

1. Para cada log, determinar o valor médio dos atributos:
    1. Quantidade de interações ou menus acessados;

2. Tempo total para a chamada;
3. Número médio de graus de saída, considerando todos menus da chamada;
4. Número médio de graus de entrada, considerando todos menus da chamada.

2. Definir um algoritmo de agrupamento (K-Means);
3. Determinar o número de grupos a serem criados com K-Means (método Elbow);
4. Interpretar os agrupamentos tomando como base em situações previamente conhecidas no autoatendimento via URA.

A Figura 28 apresenta um fluxo para a metodologia.

As etapas para estimação dos clusters, descritas na metodologia acima, serão detalhadas em próximas sessões.

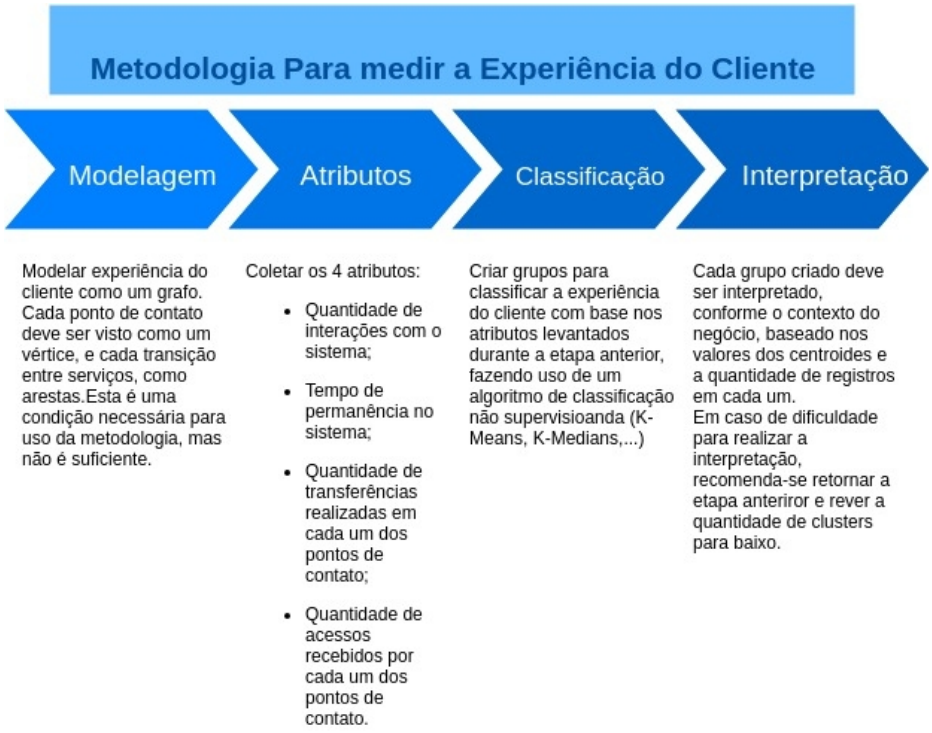

Figura 28: Fluxo metodológico para mensuração da experiência do cliente.

4.2 Etapas para utilizar um modelo de agrupamento de chamadas

Ao utilizar um modelo com base em um grafo orientado para descrever uma chamada, supõe-se que cada um dos menus é um vértice desse grafo, e cada uma das conexões entre os menus são as arestas. A Tabela 4 apresenta um exemplo de uma chamada com oito interações me menus. Essa mesma chamada pode ser avaliada a partir de um modelo baseado em um grafo a partir da Figura 29.

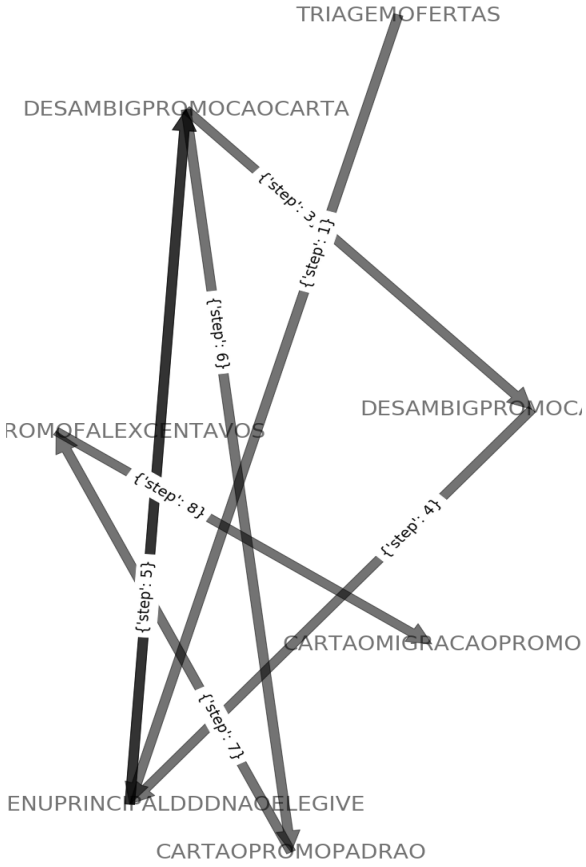

Figura 29: Exemplo de uma figura modelada como um grafo.

Analisando o exemplo utilizado é possível verificar que existem dois menus que são percorridos duas vezes, sendo que o primeiro possui o nome CARTAOMENUPRINCIPALDDDNAOELEGIVEL e o segundo, DESAMBIGPROMOCAOCARTAO. Note que, isso ocorre durante o passo 2 e o passo de número 5, vide Figura 29 ou Tabela 4.

Tabela 4: exemplo de chamada navegada.

Step Number	Source	State Duration
1	TRIAGEMOFERTAS_SALDOVALIDO_INIT1	28.75
2	CARTAOMENUPRINCIPALDDDNAOELEGIVEL	17.27
3	DESAMBIGPROMOCAOCARTAO	6.25
4	DESAMBIGPROMOCAOCARTAO_DESAMBIGOP1_TUDOPORDIA	11.5
5	CARTAOMENUPRINCIPALDDDNAOELEGIVEL	5.36
6	DESAMBIGPROMOCAOCARTAO	26.44
7	CARTAOPROMOPADRAO_MENU	44.31
8	NAOCLIENTEPROMOFALEXCENTAVOS_MENUPRINCIPAL	47.72

Uma vez que a chamada esteja representada pelo formato da Tabela 4, não é possível fazer uso do algoritmo de agrupamento para identificar chamadas de comportamentos similares. Este algoritmo trata as linhas como entidades a serem agrupadas conforme sua similaridade, assim sendo, é fundamental representar cada uma das chamadas em uma única linha.

Retomando a metodologia descrita na sessão anterior, o processo que descreve a transformação aplicada aos quatro atributos para representar a chamada a partir de uma única observação foi descrito na segunda fase da própria metodologia. Durante as operações de processamento de dados, duas tabelas auxiliares são criadas para calcular a média dos graus de saída e a média dos graus de entrada dos menus, uma vez que esses são os vértices do grafo. Ambas tabelas estão representadas pela Tabela 5 e Tabela 6, baseada na chamada ilustrada através da Figura 29.

Note que, para representar a chamada em uma única linha, conforme descrito na metodologia da sessão anterior, para os atributos baseados nos graus dos vértices (Tabela 5 e Tabela 6) basta calcular o valor médio dos graus.

Tabela 5: Graus de entrada para uma chamada.

Vertex	inDegree
CARTAOPROMOPADRAO_MENU	1
NAOCLIENTEPROMOFALEXCENTAVOS_MENUPRINCIPAL	1
DESAMBIGPROMOCAOCARTAO_DESAMBIGOP1_TUDOPORDIA	1
CARTAOMENUPRINCIPALDDDNAOELEGIVEL	2
DESAMBIGPROMOCAOCARTAO	2
CARTAOMIGRACAOPROMO_NAOELEGIVELMIGRACAOPROMO	1

Dessa forma, o valor médio dos graus de saída dos menus de uma chamada pode ser interpretado como, o valor médio de encaminhamentos que um menu realiza para os demais. E o valor médio dos graus de entrada dos menus de uma chamada pode ser interpretado como, o valor médio de transferências recebidas em um menu de todos os outros menus.

Tabela 6: Graus de saída para uma chamada.

Vertex	outDegree
TRIAGEMOFERTAS_SALDOVALIDO_INIT1	1
CARTAOPROMOPADRAO_MENU	1
NAOCLIENTEPROMOFALEXCENTAVOS_MENUPRINCIPAL	1
CARTAOMENUPRINCIPALDDDNAOELEGIVEL	2
DESAMBIGPROMOCAOCARTAO	2
DESAMBIGPROMOCAOCARTAO_DESAMBIGOP1_TUDOPORDIA	1

Os dois atributos detalhados acima, têm sua origem na modelagem da chamada pela estrutura de um grafo orientado. Os outros dois atributos que compõe a metodologia da sessão 4.1, são obtidos a partir de valores registradas no interior dos log, são eles, a quantidade de menus navegados (ou número de passos) e o tempo total da chamada.

Definir um modelo baseado em grafos para o processo de navegação em uma URA é uma abordagem interessante, porém, caso não fosse a utilização das propriedades dos graus dos vértices, como atributos para o algoritmo de agrupamento, a estrutura de grafos não traria benefício alguma a metodologia apresentada. Além disso, no contexto da mensuração da experiência do cliente, a informação referente aos graus permite identificar padrões muito específicos de agrupamentos, e de características mais complexas.

Os quatro atributos utilizados estão sempre presentes em estruturas baseadas em grafos para descrever a experiência do cliente, que tomem como referência a definição de (NASH; ARMSTRONG; ROBERTSON, 2013).

A Tabela 7 lista esses atributos sumarizados para a etapa de agrupamento. Nesta mesma tabela existe a identificação para a chamada através de um código interno criado no momento que um log é registrado, representado pelo campo *session ID*. Visando facilitar o entendimento o mesmo código foi utilizado para identificar as chamadas, uma vez que essas estejam armazenadas em estruturas tabulares.

Tabela 7: Atributos sumarizados para o K-Means.

Field Name	Field Description
key	Primary key
session ID	Call key
Step Count	Count of interations
Total Duration	Sum of time in call
Average Indegree	Average number of indegree
Average Outdegree	Average number of outdegree

Detalhando os atributos da Tabela 7, *Step Count* é a quantidade de menus navegados, *Total Duration* ao tempo total de navegação. Os atributos, *Average Indegree*

e *Average Outdegree*, são as médias dos graus de entrada e de saída de todos os menus. E a Tabela 8 é um exemplo desses atributos em sua forma reduzida para uso no algoritmo de agrupamento.

Tabela 8: Exempo de chamada sumarizada para submissão ao K-Means.

Session ID	Step Count	Total Duration	Average Outdegree	Average Indegree
0A063C1E_00001290_ 551BF6C6_B7B4_0070	8	158,85	1,4	1,4

4.3 Fluxo de dados

Introduzidas as tecnologias individuais e a arquitetura para o DSPS no capítulo 2, as informações para caracterização do problema e a metodológica proposta no capítulo 3. Es É possível retornar a figura da arquitetura, para detalhar o fluxo de dados percorrido ao longo do sistema. A Figura 30 retoma a arquitetura apresentada no início e fim do capítulo 2, incluindo cada quatro etapas para percorrer o processo do fluxo contínuo de dados.

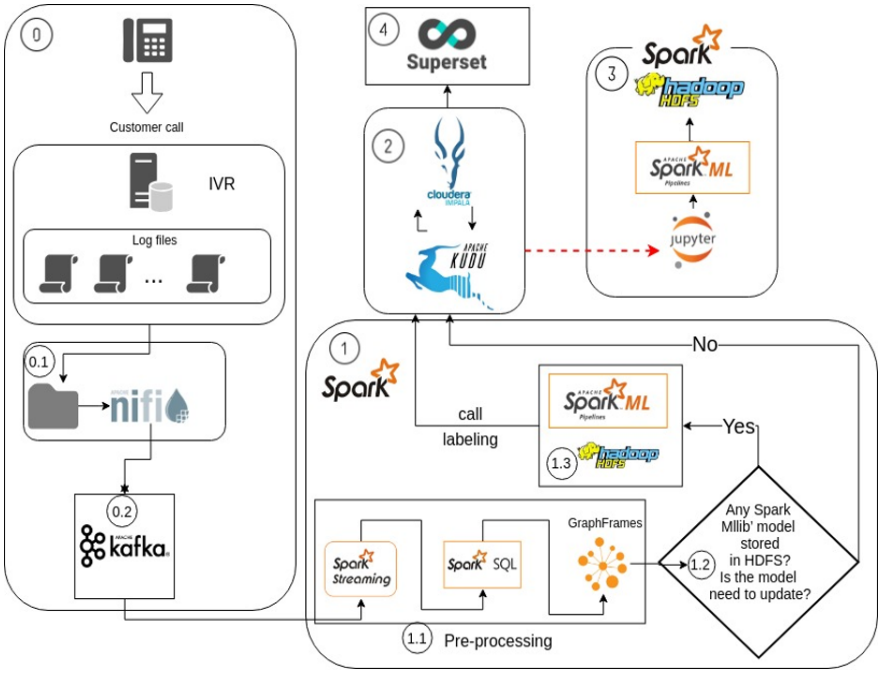

Figura 30: Fluxo final para processamento de logs de URA

As quatro etapas estão descritas a seguir, em formato de uma estrutura de um algoritmo.

(0) Inicialização dos sistemas: Inicialize os serviços, NiFi e Kafka; Crie duas tabelas no banco de dados Kudu, conforme os atributos das Tabela 3, Tabela 7 e Tabela 9; Identifique o local onde os log de chamadas são criados e configure no Nifi; e inicie o código Spark.

(0.1) NiFi monitora origem dos logs e transfere para o Kafka.

(0.2) Kafka Topic recebe a mensagem pelo *Producer* e retransmite via *Consumer* para o Spark Streaming.

(1) Spark Streaming:

(1.1) Realizar atividade de parsing dos log em dados de chamadas, incluindo extração de atributos, modelagem para grafos (Spark streaming, SQL e GraphFrames);

(1.2) Verifica a existência de um modelo K-Means da biblioteca Mllib armazenado no HDFS. Ele Existe? Está atualizado?

Se sim, vá para o passo (1.3).Caso contrário vá para o passo (2).

(1.3) Classifique a chamada em um dos clusters, através da menor distância entre, os seus atributos Tabela 7, e o modelo armazenado no HDFS. Armazene o resultado no banco de dados Kudu, de acordo com a estrutura apresentada em Tabela 9.

(2) Armazene os resultados dos dados processados conforme as Tabela 3, Tabela 7.

(2.1) Verifique. Existem registros de chamadas o suficiente para computar os centroides do K-Means? Caso sim, vá para o passo (3). Caso contrário, retorne para o passo (0.1).

(3) A partir dos dados previamente processados e armazenados no banco de dados Kudu, com o Spark, faça usos do Jupyter[7] notebook e do SparkMlib; e compute o K-Means, através dos passos a seguir.

(3.1) Padronize os atributos a serem utilizados;

(3.2) Avalie a quantidade de clusters a serem criados;

(3.3) Execute K-Means;

(3.4) Interprete os clusters criados, a partir dos centroides e quantidade de registros;

(3.5) Armazene o modelo K-Means criado no HDFS;

(3.6) Volte para o passo (0.1)

Como resultado desse fluxo, o atributo referentes ao agrupamento a qual uma chamada pertence passa a incluir a sua caracterização. Ilustrando isso segue a Tabela 9, com os atributos listados anteriormente na Tabela 3 e incluindo uma identificação

[7] O Jupyter Notebook foi incluído no detalhamento, por ter sido uma interface auxiliar utilizada durante a construção do processo, auxiliando durante a avaliação do K-Means.

numérica para o agrupamento (Cluster) e o nome dado (Cluster Name), além dessas duas novas informações, também foi incluído o registro referente ao momento de processamento do log.

Tabela 9: Atributos da tabela com a classificação de acordo com cluster.

Field Name	Field Description
Key	Primary key
Session_id	Call key
Tel number	Identification number
Step	Position of Menu in Navigation
Source	Name of origem Menu
Destiny	Name of origem Menu
Relationship	Name of response chosen by customer
Day	Day of call
Month	Month of call
Year	Year of call
Time	Time of call
State Duration	Time of duration in each Menu
Prompt Delay	Time of delay to start audio in each Menu
Prompt Duration	Tempo de duração total
Event Date	Date and time of call envent
Processing Date	Date and time of processing
Cluster	Cluster number
Cluster Name	Interpretation of the meaning for each cluster group created

Em um breve detalhamento sobre o algoritmo apresentado acima, destaca-se a importância do passo (1), responsável por centralizar o Spark e as atividades de:

- Processar e armazenar o primeiro grupo de logs, utilizados para calcular os centroides do K-Means.

- Uma vez diante das informações dos agrupamentos, o passo (1.3) é responsável por atribuir novas chamadas a um dos clusters;

- Todo o processo de extração, transformação, modelagem, ocorre no passo.

Ainda com o intuito de detalhar o fluxo de dados, a Figura 31 apresentam algumas imagens de telas para diferentes etapas de funcionamento do processo de coleta, processamento e armazenamento.

A tela de número 1, retrata o momento em que os logs são registrados em local físico no sistema, logo após a finalização do contato entre o cliente e a URA. Imediatamente após serem criados, são transportados para o Kafka através do NiFi (ambos sendo executados em segundo plano).

Em paralelo ao processo descrito acima, o Spark está coletando as mensagens (logs) enfileirados no Kafka, conforme a tela 2. E por fim, após a execução do processamento pelo Spark, de acordo com códigos no Anexo 3, as chamadas são armazenadas no banco de dados Kudu, conforme os exemplos descritos pela Tabela 4 e Tabela 8, ilustrados pela tela 3.

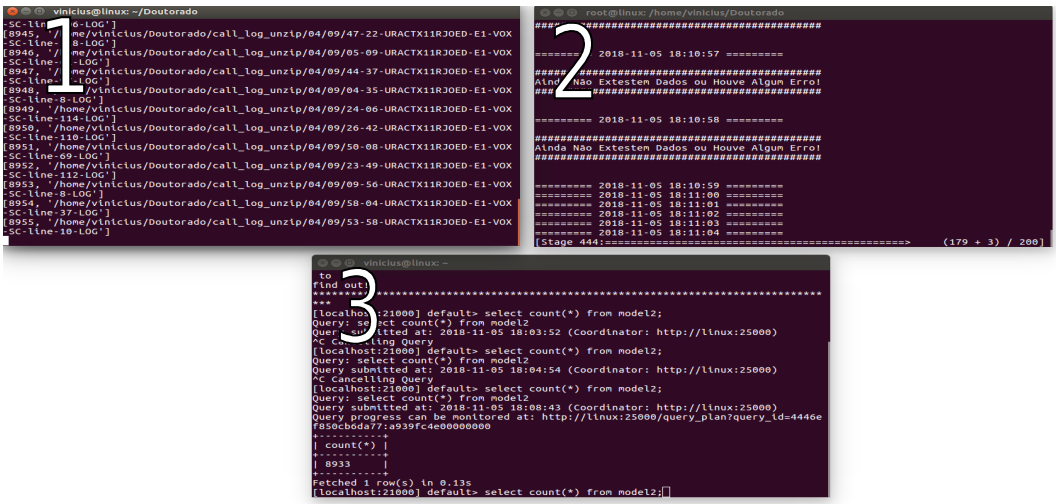

Figura 31: Telas auxiliares para ilustra o funcionamento do DSPS.

É importante destacar que a tela 1 exibe a gravação dos logs a partir de um código (Anexo 2) desenvolvido para simular o funcionamento da URA, uma vez que os logs foram coletados com a finalidade de desenvolver esse trabalho.

O próximo capítulo apresenta como resultado, a utilização da metodologia proposta para medir a experiência do cliente em conjunto com o processamento dos logs em tempo-quase-real, com base na arquitetura criada (Figura 30) e o fluxo de processamento, ilustrado pelo pelo algoritmo apresentado neste capítulo.

4.4 Processamento do grupo inicial de logs

Foram disponibilizados para realizar esse trabalho 3GB de logs, equivalente a dois dias das experiências de navegação dos clientes, da telefonia móvel do tipo pré-pago, isso é equivalente a dois dias de funcionamento da URA.

Os logs foram separados em dois grupos de dados, tomando como base o dia em que foram gerados pela URA e essa sessão tratará o primeiro grupo. O segundo será abordado a frente.

O processamento dos logs resultou em uma quantidade de 17.646 (1,5GB) chamadas validas, que foram armazenadas em formato tabular no banco de dados Kudu,

seguindo a estrutura de dados conforme apresentado no exemplo ilustrado pela Tabela 4.

Essas chamadas, após processadas, foram utilizadas para estimar os centroides do algoritmo de agrupamento (K-Means). O processamento foi realizado em um hardware com as seguintes especificações; processador Intel Core i7, com oito núcleos; 16 GB de memória RAM e um disco rígido de 250 GB M2-SSD.

O tempo de processamento está representado pelo histograma da Figura 32, em média de 3,47 segundos, tomando a mediana como medida, uma vez que a distribuição do tempo de processamento é assimétrica.

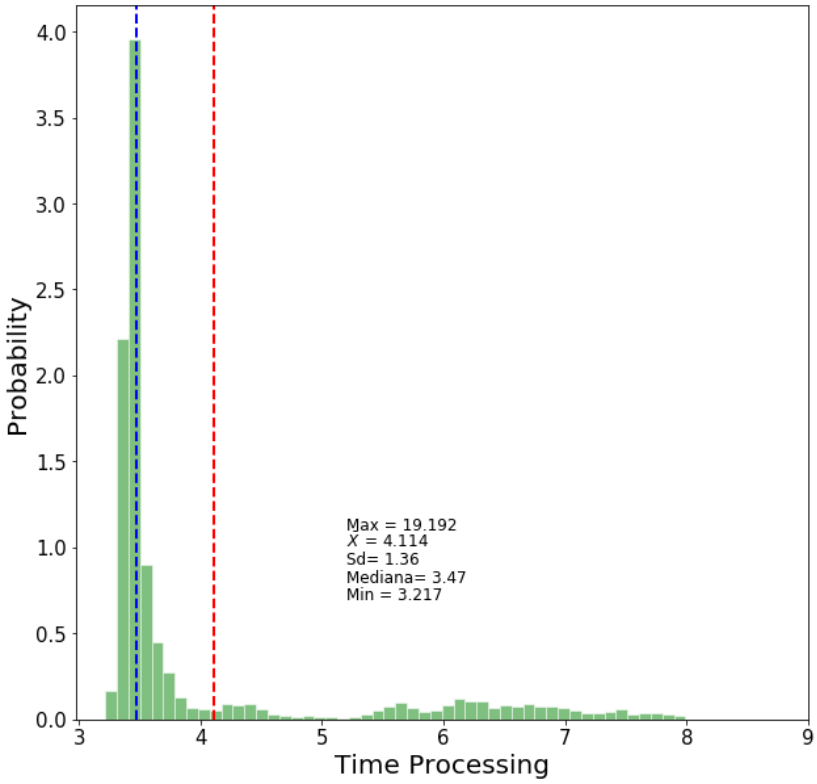

Figura 32: Distribuição do tempo de processamento do primeiro grupo de logs.

A intenção ao se separar os logs em dois grupos foi utilizar o primeiro grupo para estimar os centroides do algoritmo de agrupamento, já o segundo grupo, foi utilizado para simular o funcionamento do sistema em sua totalidade. As sessões apresentadas a seguir detalham as etapas e resultados para o primeiro grupo.

4.5 Agrupamento e personas

Uma vez que o K-Means foi o algoritmo escolhido[8] para realizar os agrupamentos, fez se necessário determinar a quantidade de agrupamentos a serem criados. Para isso, foi utilizado o método de Elbow (método do cotovelo), que tem como proposta avaliar a variação total interna dos clusters (Within Sum of Squares – WSS), a partir da soma de todas as distâncias entre cada uma das observações e seus respectivos centroides. Para isso, um grupo de cenários foram simulados variando a quantidade de agrupamentos criados em cada um deles. Dessa forma foi possível avaliar em que momento o aumento no núemro de agrupamentos tornou-se menos eficiente no cálculo que visa reduzir a variabilidade total (EVERITT; DUNN, 2001a; HAIR, 2010).

O número de simulações realizadas variou o número de agrupamentos de 2 à 20 grupos, sendo que, para medir cada um desses cenários fez-se o uso do WSS, conforme apresentado pela equação (1). O processo final decisório a respeito da quantidade de grupos foi realizado comparando as diferenças sequências entre os WSS, equação (2), dado o número de agrupamentos, abordagem bastante comum descrita em (TAN et al., 2013).

$$WSS = \sum \sum Dist_{euclidian}\left(X_{ij}, c_j\right)^2 ; i = 1, \ldots, n; j = 1, \ldots, K \quad (1)$$

$$\Delta WSS = WSS_c - WSS_{c-1}; c = 2, \ldots, 20 \quad (2)$$

Note que, a Figura 33 apresenta o resultado para cada uma das medidas WSS em função do número de agrupamentos propostos. No eixo das abscisas estão as quantidades de agrupamentos simulados e no eixo das ordenadas os resultados dos WSS.

Uma redução acentuada na taxa do WSS ocorre em torno de cinco e seis agrupamentos. O uso do método Elbow para determinar o número de grupos foi utilizado de forma a direcionar a escolha, uma vez que o conhecimento a respeito do problema em questão limita a quantidade de grupos, pois sabe-se que não existem muitos cenários no processo de atendimento resultante da navegação na URA, assim sendo, o uso cenários com uma alta quantidade de agrupamentos para o K-Means não são uma opção pois tendem a aumentar a dificuldade em se interpretar os resultados.

[8] A escolha do K-Means foi orientada a sua popularidade para realizar os agrupamentos, uma vez que o foco do trabalho não é o processo de agrupamento e sim a arquitetura e a metodologia proposta.

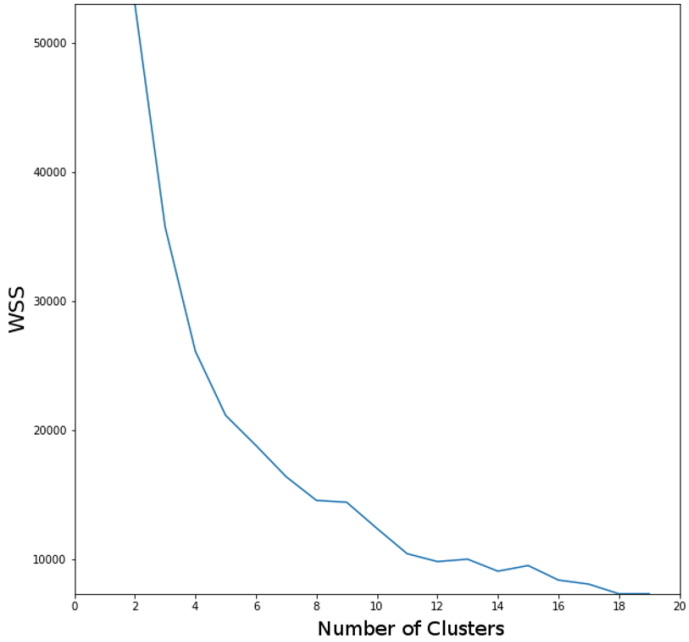

Figura 33: Simulação: número de clusters vs WSS - Elbow Method

Em cada um dos cenários os atributos foram padronizados tornando-os comparáveis em uma mesma escala, evitando assim distorções, prática comum ao se tratar problemas dessa natureza. A padronização ocorre a partir da média e desvio padrão, equações (3) e (4), respectivamente (MINGOTI, 2007).

$$\bar{X} = \sum x_i / n \tag{3}$$

$$\sigma = \sqrt{(1/(n-1)) \sum (x_i - \bar{x})^2} \tag{4}$$

A padronização dos atributos é realizada subtraindo a respectiva média em cada um desses atributos, em seguida para cada atributo resultante, dividindo-os pelo desvio padrão respectivo a cada atributo. Dessa forma cada um dos atributos padronizados passa apresentar valores centrados em zero. Este procedimento é utilizado para estimar os centroides, e consequentemente identificar a qual agrupamento cada uma das chamadas pertencem.

Os cinco centroides do algoritmo K-Means estão exibidos na Tabela 10, com base neles foram definidas as personas. Isso foi feito em conjunto com um especialista em atendimento.

Tabela 10: Centroides obtidos após execução do K-Means, para o conjunto com 17.646 logs.

Cluster number	Average for steps number	Total time duration over a call	Average in-degree for vertices	Average out-degree for vertices
1	18,4	110,4	9,3	8,8
2	17,8	390,9	2,2	2,2
3	3,1	64,2	1,2	1,4
4	110,5	438,7	44,5	44,0
5	8,3	179,7	1,5	1,5

Para cada das um das personas definidas para os agrupamentos, os nomes referentes a interpretação de negócio foram definidos pelo autor deste trabalho.

Cluster 1: apresenta um alto valor médio para o número de graus, seja de entrada ou saída, além de uma média de passos navegados elevada. A partir desses resultados é possível concluir que os clientes experimentaram navegar apenas entre um grupo restrito de menus, assim, vivenciando uma experiência negativa e não atingindo o objetivo do contato. Este cluster foi classificado como *Warning: High Menu Repetition.*

Cluster 2: Este grupo apresenta um valor médio para a quantidade de passos navegados (Average for steps number) alto, uma estatística baixa de transferências, representada pelos atributos (Average in-degree for vertices e Average out-degree for vertices), além de um tempo médio de navegação (Total time duration over a call) igual a 6,5 minutos. Este grupo se destaca pelo elevado número de passos, que sugerem esforço para realizar o atendimento. Uma experiência negativa, que não chega a ser grave. Cluster classificado como *Warning: Many Steps.*

Cluster 3: apresenta um comportamento com ligações de curta duração. Este é um padrão comum visto em clientes com planos pré-pagos, que em grande maioria entram em contato buscando informações a respeito a respeito do saldo de créditos em seu plano. Cluster classificado como, *Regular: Fast Call.*

Cluster 4: Outliers. Atributos com valores extremados. Geralmente visto quando ocorre um falha na URA, ou em casos de testes simulados dentro do sistema da URA. Este tipo padrão é praticamente impossível de ser criado por uma cliente, porém cada ocorrência de erro no sistema consome recursos computacionais da URA, que deveriam ser dedicadas ao cliente para melhorar sua experiência. Cluster classificado como *Warning: Outlier / Error*.

Cluster 5: Os atributos deste centroide apresentam atributos com valores dentro da média. Indicando um atendimento padrão na solicitação de serviços mais complexos, como, por exemplo, informações a respeito do consumo, compra de novos créditos,

aquisição de novos produtos, cancelamento ou migração de plano, etc. Cluster classificado como *Regular: Common call*.

4.6 Implantação dos agrupamentos no DSPS

Método de aprendizado de máquina precisam auxiliar o processo de tomadas decisões de forma automática, desta forma é preciso garantir que os resultados obtidos para estimar os centroides dos agrupamentos, detalhados na sessão anterior, sejam incorporados ao fluxo de processamento de dados. Desta forma, essa sessão introduz como isso foi realizado no contexto do DSPS apresentado.

Cada novo log submetido ao DSPS, em tempo de processamento, deve ser associado a um agrupamento. O Spark é uma plataforma RED que tem com uma de suas bibliotecas centrais o Mllib, que é orientado ao desenvolvimento de aprendizado de máquina. Nela, além de algoritmos como o K-Means existem outras funções, partes de APIs, que permitem um grande número de funcionalidades como por exemplo, facilitar a construção de fluxos contendo inúmeras regras de aprendizado de máquina, e exportá-las de maneira simplificada.

O *pipeline* é uma API da biblioteca Mllib (“MLlib: Main Guide – Spark 2.4.0 Documentation,” [s.d.]) que permite encadear funções, utilizada para inserir os resultados obtidos através do K-Means, Tabela 10, ao DSPS ilustrado pela Figura 30, Dessa forma foi possível fornecer de forma rápida e simplificada as informações contidas no processo de modelagem ao sistema.

Dentro da biblioteca Spark MLlib existem cinco APIs, para serem utilizadas em conjunto no Spark, para criação de modelos de aprendizado de máquina. São elas, *Utilities, Featurization, Pipelines, Persistence e ML Algorithms.*

Featurization é uma API que visa fornecer ferramentas para transformar e padronizar os dados em fase de preparação, por exemplo a função *VectorAssembler,* que permite transformar conjunto de atributos em um único vetor, modelo de trabalho do Sparl. A função StandardScaler padroniza um atributo em uma tabela em uma escala em torno do zero e com desvio padrão unitário, visando reduzir problemas de diferentes escalas entre atributos. Ambas funções fazem parte do pipeline.

Por fim, *Persistence,* que permite escrever e ler pipelines criados no HDFS, onde estes podem ser requisitados quando necessário. Por exemplo, neste trabalho o pipline foi utilizado através da biblioteca Spark Streaming.

Uma vez apresentada cada uma das APIs, é possível descrever de forma simplificada as funções que foram utilizadas para realizar o processo de classificação das chamadas, baseado nos agrupamentos da sessão introduzidos na sessão 4.6, são elas:

StandardScaler, VectorAssembler e K-means.fit. A partir delas, uma representação do fluxo de utilização pode ser visto na Figura 34.

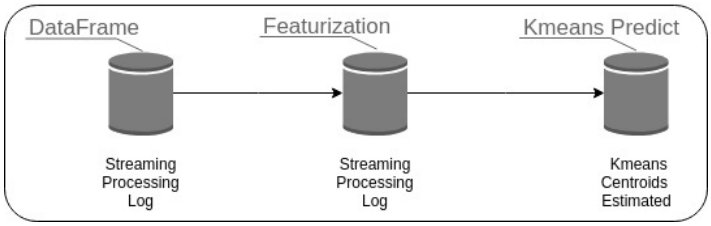

Figura 34: Ilustração do uso da função pipeline.

5 Resultados da classificação em tempo quase real para log de chamadas

Visando simular um cenário completo para tomada de decisão em tempo quase real com base nas informações da URA, o capítulo 5 combinou um segundo conjunto de logs, com 9.384 arquivos (aproximadamente 1,2 GB), os centroides calculados a partir do K-Means considerando o primeiro conjunto de logs, a metodologia apresentada capaz de mensurar a experiência do cliente, e a arquitetura desenvolvida a partir de tecnologias RED, para realizar o processamento de dados em um fluxo contínuo (DSPS).

Neste novo grupo de logs de chamadas, cada um delas é classificada conforme um dos agrupamentos introduzidos na sessão 4.5. Isto ocorre no momento em que são submetidas ao sistema, permitindo uma abordagem diferente para o processo de atendimento baseado nos perfis previamente identificados.

Neste novo cenário os resultados foram apresentados no Superset, tecnologia RED para criação de gráficos e painéis introduzida na sessão 2.8. O Superset permite visualizar em tempo quase real os resultados, uma vez que os dados são consumidos sequencialmente após seu armazenamento do banco de dados Kudu. Ou seja, as tabelas que armazenam os resultados produzidos pelo Spark, são uma fonte imediata para o Superset.

Foram construídos dois painéis com o auxílio do Superset, o primeiro contempla as informações referentes a chamada mais recente[9] disponibilizada pelo sistema DSPS. Enquanto o segundo painel disponibiliza informações acumuladas, permitindo avaliar historicamente comportamentos dos clientes que navegaram pela URA.

O DSPS é dinâmico, e isto inclui os painéis do Superset, essa característica não pode ser retratada neste documento, e esta fracamente representada por duas imagens dos painéis, a partir da Figura 35 e Figura 36, que estão detalhados a seguir.

Uma demostração do DSPS em funcionamento pode ser verificada no link: https://www.youtube.com/watch?v=CW3bcnbKSwI&t=38s

A Figura 35, contém três gráficos:

- A representação da chamada em evidência através de um grafo orientado;
- A classificação atribuída a esta chamada com base no perfil de agrupamento estipulado na sessão 4.5.
- Os atributos da mesma chamada, apresentados em formato tabular.

[9] Uma vez que essa esteja armazenada no Kudu.

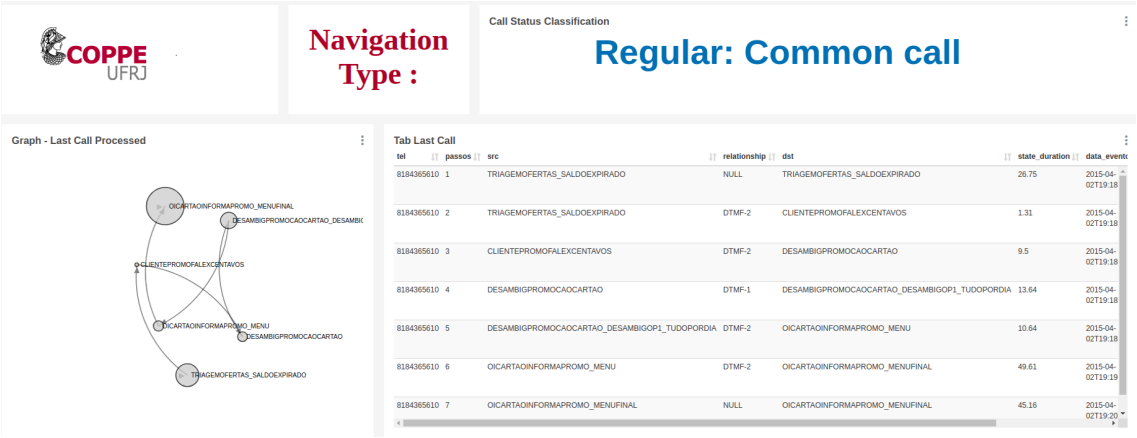

Figura 35: Primeiro painel: identificação da ultima chamada disponibilizada pelo sistema.

A Figura 36 está organizada da seguinte forma:

Primeira linha: o primeiro objeto é uma janela que permite filtrar todos os objetos do painel. Em seguida, à direita, está uma janela que apresenta o resultado total de chamadas já processadas e armazenadas. Em seguida está um gráfico representado os log processados ao longo do tempo em intervalos de um minuto.

Segunda linha: o primeiro gráfico é um boxplot, para avaliar a dispersão da quantidade de chamadas em cada um dos agrupamentos. Em uma análise preliminar sobre este gráfico é possível notar que, o grupo denominado *Regular: Quick Call* apresenta a maior mediana, indicando que este grupo é responsável pela maioria das chamadas, em seguida está o agrupamento *Regular: Common Call*, que possui a maior variabilidade dentre os outros agrupamentos. Note que, o alto valor para mediana em ambos os grupos está dentro do resultado esperado, uma vez que ambos deveriam ser responsáveis pela maioria das chamadas, uma vez que se assume um bom funcionamento da URA. E ao final da segunda linha foi disponibilizado um vídeo introdutório sobre Big Data.

Terceira linha: o primeiro gráfico é um histograma para o tempo de processamento. Em seguida uma nuvem de palavras, para todos os menus navegados. Em uma análise preliminar, existem dois nomes dento da nuvem que se destacam, são eles, UNCAPTURADATAVALIDADE e DESAMBIGPROMOCAOCARTAO. Em ambos os casos estão relacionados com a temática, promoção de um plano do cliente. O menu UNCAPTURADATAVALIDADE verifica a existência de alguma promoção, enquanto o DESAMBIGPROMOCAOCARTAO avalia como deve ser encaminhada a chamada em questão para outros menus com base nas regras existentes.

Figura 36: Segundo painel: dados históricos da chamdas.

6 Conclusões

Este trabalho contribuiu em duas direções, sendo a primeira, apresentando uma metodologia para avaliar e medir a experiência do cliente, que pode ser expansível a diferentes abordagens, e não somente ao universo de uma URA. A segunda contribuição está na arquitetura proposta do sistema DSPS, que possui as características de plataformas resilientes, escaláveis e distribuídas, permitindo implantar a metodologia citada, no contexto de navegação em uma URA. O DSPS também permite uma utilização ampla em circunstâncias variadas apenas com pequenos ajustes no que tange o processamento dos dados.

O uso da teoria dos grafos para modelar o processo de navegação em uma URA é uma abordagem alternativa, que permitiu o desenvolvimento da metodologia generalista para medir a experiência do cliente. E do ponto de vista computacional, o DSPS desenvolvido neste trabalho viabilizou o uso da modelagem baseada em grafos e a aplicabilidade da metodologia para experiência do cliente em tempo quase real.

Outras contribuições foram, propor uma solução para a companhia de telecomunicações que cedeu os dados para a realização dessa pesquisa. A possibilidade de explorar e avaliar projetos RED, voltados a pesquisa em Big Data, relevantes para a comunidade como Spark, Kafka, Impala, Kudu entre outros.

O Hadoop-MapReduce foi uma das primeiras alternativas avaliadas para processamento de logs de URA, porém uma vez que os principais objetivos eram, velocidade e escalabilidade, o Spark foi adotado pela facilidade proposta por suas bibliotecas Spark Streamig e Spark Mllib.

Dentre as plataformas de streaming, o projeto Flume (“Welcome to Apache Flume — Apache Flume,” [s.d.]; WHITE, 2012) foi avaliado previamente frente ao Kafka. A mudança ocorreu por uma revisão na utilização do Spark a partir da biblioteca Spark-Structured-Streaming, (“Structured Streaming Programming Guide – Spark 2.4.2 Documentation,” [s.d.]), compatível apenas com Kafka. Porém, o Spark-Structured-Streaming não possuía as componentes vistas no Spark Streaming e não foi adotado, porém o Kafka mantido.

Durante a avaliação sob como armazenar os resultados, o Apache Solr foi considerado por ser um projeto integrante da distribuição Cloudera com características para armazenar dados não estruturados, indexando-os em tempo real, (“Apache Solr - Ref Guide,” [s.d.]). Outro projeto foi o Neo4j, um banco de dados orientado a grafos (“Neo4j Graph Platform – The Leader in Graph Databases,” [s.d.]), que permitiria manter o modelo de grafos das chamadas. Porém, o primeiro apresentou dificuldade ao Spark Streaming, e o segundo ao uso do Python com o API.

Entre os desafios aprendidos, o Spark se mostrou capaz de integrar-se com diversas tecnologias e, de forma geral, uma solução poderosa em questões que abordem a necessidade de se tratar fluxos contínuos de dados (streaming). Dentre aspectos negativos experimentados, a necessidade de realizar um processo de configuração específico para melhorar o gerenciamento no uso de memória. O uso de muitas bibliotecas externas. E até o momento dessa pesquisa, problemas de incompatibilidade entre essas as bibliotecas não nativas e a versão do Spark contida na distribuição Cloudera, problema que foi contornado através da obtenção de uma versão direta de https://spark.apache.org/.

Sobre o Kafka, Kudu e Impala, não apresentaram dificuldades de uso, assim como o Nifi, em sua versão 1.8 que não é um componente integrante da distribuição Cloudera. E por fim, o Apache Superset mostrou-se uma plataforma útil para construção de painéis e gráficos, sendo capaz de conectar-se com diferentes fontes de dados.

Como tema de trabalhos futuros, algumas sugestões são, utilizar a metodologia desenvolvida em um outro contexto envolvendo a medição da experiência do cliente, assim como adaptar a arquitetura do DSPS para tratar problemas mais amplos.

Realizar a atualização dos centroides continuamente, a cada novo log processado pelo DSPS, e simultaneamente garantir o significado determinado para os agrupamentos.

Avaliar novos métodos para o algoritmo de classificação não supervisionado dentro contexto da metodologia proposta.

Referências Bibliográficas

A.P.MITTAL; JAIN, V.; AHUJA, T. Google File System and Hadoop Distributed File System- An Analogy. **International Journal of Innovations & Advancement in Computer Science (IJIACS)**, v. 4, n. March, p. 626–636, 2015.

AGRAHARI, A.; RAO, D. A Review paper on Big Data: Technologies, Tools and Trends. **International Research Journal of Engineering and Technology (IRJET)**, v. 4, n. 10, p. 640–649, 2017.

Apache HBase – Apache HBase™ Home. Disponível em: <https://hbase.apache.org/>. Acesso em: 11 aug. 2019.

Apache Kafka. Disponível em: <https://kafka.apache.org/>. Acesso em: 5 mar. 2019.

Apache NiFi. Disponível em: <https://nifi.apache.org/>. Acesso em: 5 mar. 2019.

Apache Parquet. Disponível em: <https://parquet.apache.org/>. Acesso em: 11 aug. 2019.

APACHE SOFTWARE FOUNDATION. **Apache Projects List**. Disponível em: <https://projects.apache.org/projects.html>. Acesso em: 19 mar. 2019.

Apache Solr - Ref Guide. Disponível em: <http://lucene.apache.org/solr/guide/>. Acesso em: 5 mar. 2019.

Apache Spark™ - Unified Analytics Engine for Big Data. Disponível em: <https://spark.apache.org/>. Acesso em: 4 mar. 2019.

Apache Superset (incubating) — Apache Superset documentation. Disponível em: <https://superset.incubator.apache.org/>. Acesso em: 5 mar. 2019.

ARMBRUST, M. et al. Spark SQL: Relational Data Processing in Spark. [s.d.].

BOROVKOV, A. A. Systems with Autonomous Service. In: **Stochastic Processes in Queueing Theory**. New York, NY: Springer New York, 1976. p. 235–242.

BRADLEY, J. K. GraphFrames About the speaker : Joseph Bradley. 2016.

CARPENTER, J. (SYSTEMS ARCHITECT); HEWITT, E. **Cassandra : the definitive guide**. [s.l: s.n.].

CHAMBERS, B.; ZAHARIA, M. Spark The Definitive GuideExcerpts from the upcoming book on making big data simple with Apache Spark. p. 1–127, 2017.

CIMIANO, P. . H. M. Taming Dark Data : Data-Driven Insights For Smart Decision Making. **Semalytix**, 2018.

CLOUDERA. **Spark Guide**. Palo Alto: [s.n.]. Disponível em: <https://opensource.org/licenses/Apache-2.0>. Acesso em: 4 mar. 2019a.

CLOUDERA. **Apache Kafka Guide**. Palo Alto: [s.n.]. Disponível em: <https://opensource.org/licenses/Apache-2.0>. Acesso em: 5 mar. 2019b.

CLOUDERA. **Apache Impala Guide**. Palo Alto: [s.n.]. Disponível em: <https://opensource.org/licenses/Apache-2.0>. Acesso em: 5 mar. 2019c.

CLOUDERA. **Apache Kudu Guide**. Palo Alto: [s.n.].

COSTA, P. P. DA. Teoria de Grafos e suas Aplicações. p. 77, 2011.

COX, M.; ELLSWORTH, D. **Application-Controlled Demand Paging for Out-of-Core Visualization**. [s.l: s.n.]. Disponível em: <https://www.nas.nasa.gov/assets/pdf/techreports/1997/nas-97-010.pdf>. Acesso em: 19 mar. 2019.

DE SOUZA, D. DA C. R. **Uma arquitetura de referência para o processamento distribuído de stream de dados em soluções analíticas de near real-time**. [s.l: s.n.].

DEGROOT, M. H.; SCHERVISH, M. J. **Probability and statistics**. [s.l.] Addison-Wesley, 2012.

DEMING, E.; DRUCKER, P. Big Data : The Management Revolution. **Havard Business Review**, n. October, p. 1–12, 2012.

DIESTEL, R. **Graph Theory, Electronic**. [s.l: s.n.]. v. 173

ECKERSON, W. W. **Performance dashboards : measuring, monitoring, and managing your business**. [s.l.] Wiley, 2011.

EVERITT, B.; DUNN, G. **Applied multivariate data analysis**. [s.l.] Wiley & Sons, 2001a.

EVERITT, B. S.; DUNN, G. **Applied Multivariate Data Analysis**. West Sussex, United Kingdom: John Wiley & Sons, Ltd,., 2001b.

FLORATOU, A.; MINHAS, U. F.; OZCAN, F. **SQL-on-Hadoop: Full Circle Back to Shared-Nothing Database Architectures**. [s.l: s.n.]. Disponível em: <http://www.tpc.org/tpch/>. Acesso em: 11 aug. 2019.

GANDOMI, A.; HAIDER, M. Beyond the hype: Big data concepts, methods, and analytics. **International Journal of Information Management**, v. 35, n. 2, p. 137–144, 2015.

GANS, N.; KOOLE, G.; MANDELBAUM, A. Telephone Call Centers: Tutorial, Review, and Research Prospects. **Manufacturing & Service Operations Management**, v. 5, n. 2, p. 79–141, 2003.

GENTILE, C.; SPILLER, N.; NOCI, G. How to Sustain the Customer Experience:: An Overview of Experience Components that Co-create Value With the Customer. **European Management Journal**, v. 25, n. 5, p. 395–410, 1 Oct. 2007.

GEORGE, L. **HBase : the definitive guide**. [s.l.] O'Reilly, 2011.

GHEMAWAT, S.; GOBIOFF, H.; LEUNG, S.-T. The Google file system. **ACM SIGOPS Operating Systems Review**, v. 37, n. 5, p. 29, 2003.

Giraph - Welcome To Apache Giraph! Disponível em: <http://giraph.apache.org/>. Acesso em: 29 mar. 2019.

GONZALEZ, J. E. et al. **GraphX: Graph Processing in a Distributed Dataflow Framework**. [s.l: s.n.]. Disponível em: <https://amplab.cs.berkeley.edu/wp-content/uploads/2014/09/graphx.pdf>. Acesso em: 5 mar. 2019.

HAIR, J. F. **Multivariate data analysis**. [s.l.] Prentice Hall, 2010.

HILBERT, M.; LÓPEZ, P. The World's Technological Capacity to Store, Communicate, and Compute Information. **Science Express**, p. 13, 2011.

JANARTHANAM, P.; RAJA, V. K.; CORPORATION, W. Optimization of Interactive Voice Response Systems using SAS ® Padmashri Janarthanam , University of Nebraska Omaha ; p. 1–6, 2017.

KHOTS, D.; CORPORATION, P. D. W. Unstructured Data Mining to Improve Customer Experience in Interactive Voice Response Systems. p. 1–11, 2015.

KNILL, O. **Probability Theory and Stochastic Processes with Applications**. 1. ed. NewDelhi-: Overseas Press, 2009.

KORNACKER, M. et al. **Impala: A Modern, Open-Source SQL Engine for Hadoop**. [s.l: s.n.]. Disponível em: <http://impala.io/>. Acesso em: 11 aug. 2019.

LEGGETT, K.; COM, F. Forrester2017. 2017.

LIPCON, T.; CRUTCHER, M. **Apache Kudu**. [s.l: s.n.].

Maven – Introduction. Disponível em: <https://maven.apache.org/what-is-maven.html>. Acesso em: 1 apr. 2019.

Maven Repository: com.crealytics » spark-excel_2.11 » 0.8.2. Disponível em: <https://mvnrepository.com/artifact/com.crealytics/spark-excel_2.11/0.8.2>. Acesso em: 1 apr. 2019.

MILOSLAVSKAYA, N.; TOLSTOY, A. Big Data, Fast Data and Data Lake Concepts. **Procedia Computer Science**, v. 88, p. 300–305, 1 Jan. 2016.

MINGOTI, S. A. **Análise de Dados Através de Métodos de Estatística Multivariada: Uma Abordagem Aplicada**. Belo Horizonte: UFMG, 2007.

MLlib: Main Guide - Spark 2.4.0 Documentation. Disponível em: <https://spark.apache.org/docs/latest/ml-guide.html>. Acesso em: 5 mar. 2019.

MURRAY, D. **Tableau your data! : fast and easy visual analysis with tableau software**. [s.l: s.n.].

NASH, D.; ARMSTRONG, D.; ROBERTSON, M. Customer Experience 2.0: How Data, Technology, and Advanced Analytics are Taking an Integrated, Seamless Customer Experience to the Next Frontier. **Journal of Integrated Marketing Communications**, v. 1, n. 1, p. 32–39, 2013.

Neo4j Graph Platform – The Leader in Graph Databases. Disponível em: <https://neo4j.com/>. Acesso em: 5 mar. 2019.

PADHY, R. P. Big Data Processing with Hadoop-MapReduce in Cloud Systems. **International Journal of Cloud Computing and Services Science (IJ-CLOSER)**, v. 2, n. 1, p. 3337, 2013.

PÉREZ, A.; RODRÍGUEZ DEL BOSQUE, I. An Integrative Framework to Understand How CSR Affects Customer Loyalty through Identification, Emotions and Satisfaction. **Journal of Business Ethics**, v. 129, n. 3, p. 571–584, 26 Jul. 2015.

REICHHELD, F. F. The One Number You Need to Grow. 2003.

ROAD, P. M. **Cloudera Enterprise 6 Release Guide**. [s.l: s.n.].

ROSS, S. M. **Stochastic processes**. [s.l.] Wiley, 1996.

SAMOSIR, J.; INDRAWAN-SANTIAGO, M.; HAGHIGHI, P. D. An evaluation of data stream processing systems for data driven applications. **Procedia Computer Science**, v. 80, p. 439–449, 2016.

SCHRODER, K. E. E.; JOHNSON, C. J. Interactive voice response technology to measure HIV-related behavior. **Current HIV/AIDS Reports**, v. 6, n. 4, p. 210–216, 2009.

SHI, J. et al. **Clash of the Titans: MapReduce vs. Spark for Large Scale Data Analytics**. [s.l: s.n.]. Disponível em: <http://aws.amazon.com/ec2/instance-types/>. Acesso em: 30 jul. 2019.

Stack Overflow Trends. Disponível em: <https://insights.stackoverflow.com/trends?tags=apache-spark%2Chadoop%2Capache-kafka%2Chive%2Cbigdata>. Acesso em: 30 mar. 2019.

Structured Streaming Programming Guide - Spark 2.4.0 Documentation. Disponível em: <https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html#overview>. Acesso em: 2 apr. 2019.

Structured Streaming Programming Guide - Spark 2.4.2 Documentation. Disponível em: <https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html>. Acesso em: 6 may. 2019.

TAN, P.-N. et al. **Introduction to data mining**. [s.l: s.n.].

THE APACHE SOFTWARE FOUNDATION. **Apache Hadoop**. Disponível em: <https://hadoop.apache.org/>. Acesso em: 4 mar. 2019.

Third-Party Projects | Apache Spark. Disponível em: <https://spark.apache.org/third-party-projects.html>. Acesso em: 1 apr. 2019.

VERHOEF, P. C. et al. Customer Experience Creation: Determinants, Dynamics and Management Strategies. **Journal of Retailing**, v. 85, n. 1, p. 31–41, 1 Mar. 2009.

Welcome to Apache Avro! Disponível em: <http://avro.apache.org/>. Acesso em: 11 aug. 2019.

Welcome to Apache Flume — Apache Flume. Disponível em: <https://flume.apache.org/>. Acesso em: 29 mar. 2019.

WHITE, T. **Hadoop : The Definitive Guide**. 4 th ed. Sebastopol: O'Reilly Media, Inc, 2012.

WHITE, T. **Hadoop: The Definitive Guide FOURTH EDITION**. 4 th ed. Sebastopol: O'Reilly Media, Inc, 2015.

WIATR, R.; SŁOTA, R.; KITOWSKI, J. Optimising Kafka for stream processing in latency sensitive systems. **Procedia Computer Science**, v. 136, p. 99–108, 1 Jan. 2018.

ZAHARIA, M. An Architecture for Fast and General Data Processing on Large Clusters. **An Architecture for Fast and General Data Processing on Large Clusters**, 2016.

Anexo 1 Exemplo de log completo de uma chamada

Exemplo de um log de chamada limitado as restrições de divulgação de informação confidencial.

```
      ;;;
      ;;; Type: calllog
      ;;; Version: 1
      ;;; Locale: English_United States.1252
      ;;;                                                                    Opened
D:/NVP/logs/../callLogs/master/Accenture/UraCampanha/2015/04April/01/00/00-24-
URACTX11RJOED-E1-VOX-SC-line-104-LOG
      ;;; Time: Wed Apr 01 00:00:24 2015
      ;;; Host: URACTX11RJOED
      ;;; Base path: D:/NVP/logs/../callLogs/master/Accenture/UraCampanha
      ;;; Program: NVP
      ;;;
      ;;;;;;;;;;
      start{

            BEGIN_TIME       70         = 0.984
            SESSION_ID                = 0a063c1e_00001598_551b5f47_ade6_0160
            RECORD_NUMBER             = 1
            LAST_USER_INPUT_TIME      = 2.85538e+006
            CALL_SEIZURE_TIME         = 2015/04/01 00:00:23.957
            BROWSER                   = Browser 113
            BROWSER_ID                = 1691670
            CALL_REMOTE_URL           = tel:3488001222
            CALL_LOCAL_URL            = tel:12066
            TRANSITION                              = InitialPage resulted in
http://10.6.60.42:8090/uracampanha/vxml/index.jsp?ura=URACTX11RJOED
            EXECUTING_URL                                                  =
http://10.6.60.42:8090/uracampanha/vxml/index.jsp?
ura=URACTX11RJOED<>http://10.6.60.42:8090/uracampanha/vxml/oiatende-root.jsp
            ENTERING_FORM             = #Index
            ENTERING_FORM_ITEM        = block:_0
            APP_AppSrv                = APPCTX19RJOED
            APP_Extension             = 104
            APP_Appl                  = PROMO
            TRANSITION                                                     =
http://10.6.60.42:8090/uracampanha/vxml/index.jsp?ura=URACTX11RJOED   resulted
in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=Inicio&ura=URACTX11RJOED&ctiport=104&ani=tel:34880012
22&dnis=tel:12066
            EXECUTING_URL                                                  =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=Inicio&ura=URACTX11RJOED&ctiport=104&ani=tel:34880012
22&dnis=tel:12066<>http://10.6.60.42:8090/uracampanha/vxml/oiatende-root.jsp
```

```
ENTERING_FORM             = #manutencao
ENTERING_FORM_ITEM        = block:_0
ENTERING_FORM_ITEM        = block:_1
ENTERING_FORM             = #SaudacaoInicio
ENTERING_FORM_ITEM        = block:_2
APP_varsRoot.isPossuiDDDAtivoMigracaoPreControle = true
APP_varsRoot.possuiPromocaoIN = false
APP_varsRoot.constaTabelaOfertaDegrau7 = false
APP_varsRoot.bolsoRecargaExpirado = false
APP_varsRoot.ddd          = 34
APP_varsRoot.possuiDDDAtivoOpcaoRecarga = true
APP_varsRoot.erroCadastroPromo = false
APP_varsRoot.isNovoOiControle = false
APP_varsRoot.atingiuRecargaMinMesAtual = false
APP_varsRoot.isInativo    = false
APP_varsRoot.isOiControle = false
APP_varsRoot.isPossuiDDDFaleIlimitado = false
APP_varsRoot.isDataCadastroMigracaoM4Umaior24H = false
APP_varsRoot.isClienteIsentoCobrancaTaxaFranquiaMigracao = false
APP_varsRoot.existeRecargaMax = false
APP_varsRoot.isConstaTabelaOfertaFaleIlimitado = false
APP_varsRoot.isMaillingParalelaPromo = false
APP_varsRoot.isElegivelNovaPromo = false
APP_varsRoot.possuiDDDPacoteBonusMix = true
APP_varsRoot.isOiContaTotal = false
APP_varsRoot.possuiDDDPacoteBonusExtra = true
APP_varsRoot.possuiDDDPacoteSMS = true
APP_varsRoot.possuiCreditoEspecialIN = false
APP_varsRoot.possuiPredileto = false
APP_varsRoot.dddInternetMovelPrePaga = true
APP_varsRoot.isPosPago    = false
APP_varsRoot.isDddElegivelPromoSorteioOiPaggo = false
APP_varsRoot.isInvertido  = false
APP_varsRoot.bolsoOiFixoPromocionalComecaHoje = false
APP_varsRoot.isPrePago    = true
APP_varsRoot.possuiCadastroM4U = false
APP_varsRoot.atingiuRecargaMaxMesAtual = false
APP_varsRoot.elegivelInternetMovelPrePaga = false
APP_varsRoot.constaTabelaOfertaSuperBonus = false
APP_varsRoot.isElegivelBCE = false
APP_vars.constaNaBase     = true
APP_vars.M4UBeanExecutorExecutionTime = 31
APP_vars.idBeneficio      = 44
APP_vars.parametroInformaDnis = false
APP_vars.M4URecargaPrePagoExecutor = Success
APP_vars.possuiPromo31Anos = false
APP_vars.erroConsultaSiebel = false
APP_vars.SiebelPrincipalExecutorExecutionTime = 141
APP_vars.statusCode       = 0
APP_vars.promocaoNaoIdentificada = true
APP_vars.M4URecargaPrePagoExecutorExecutionTime = 63
APP_vars.isElegivelPesquisa = false
APP_vars.M4URecargaPrePagoExecutorRestExecutionTime = 312
APP_vars.isCanalDiretoRecarga = false
APP_vars.SiebelInativosExecutor = Success
APP_vars.isErrorM4uPrepago = false
APP_vars.PerfilInExecutorExecutionTime = 422
```

```
APP_vars.M4UBeanExecutor  = Success
APP_vars.SiebelInativosExecutorExecutionTime = 0
APP_vars.SiebelPrincipalExecutor = Success
APP_vars.isSucessoConsultaM4U = true
APP_vars.dnis             = tel:12066
APP_vars.M4URecargaControleExecutor = Success
APP_vars.aniDisponivel    = true
APP_vars.INChamadaConsulta = true
APP_vars.ani              = 3488001222
APP_vars.hour             = 0
APP_vars.isPesquisaOn     = false
APP_vars.PerfilInExecutor = Success
APP_vars.erroConsultaIN   = false
APP_vars.emManutencao     = false
APP_vars.INConsultaMonUra = true
APP_vars.M4URecargaControleExecutorExecutionTime = 203
APP_vars.M4URecargaPrePagoExecutorRest = Success
APP_vars.isClienteCorporativo = false
APP_vars.idPrograma       = 35
ENTERING_FORM_ITEM        = block:playBlock
APP_AppSrv                = APPCTX19RJOED
APP_Ani                   =
APP_Dnis                  = tel:12066
APP_INCONSULTAMONURA      = true
APP_IDPROGRAMA            = 35
APP_IDBENEFICIO           = 44
APP_INPERFORMANCE         = 0
APP_INCHAMADACONSULTA     = true
APP_PROMO                 =
ENTERING_FORM             = #Inicio
ENTERING_FORM_ITEM        = block:_3
APP_varsRoot.isPossuiDDDAtivoMigracaoPreControle = true
APP_varsRoot.possuiPromocaoIN = false
APP_varsRoot.constaTabelaOfertaDegrau7 = false
APP_varsRoot.bolsoRecargaExpirado = false
APP_varsRoot.ddd          = 34
APP_varsRoot.possuiDDDAtivoOpcaoRecarga = true
APP_varsRoot.erroCadastroPromo = false
APP_varsRoot.isNovoOiControle = false
APP_varsRoot.atingiuRecargaMinMesAtual = false
APP_varsRoot.isInativo    = false
APP_varsRoot.isOiControle = false
APP_varsRoot.isPossuiDDDFaleIlimitado = false
APP_varsRoot.isDataCadastroMigracaoM4Umaior24H = false
APP_varsRoot.isClienteIsentoCobrancaTaxaFranquiaMigracao = false
APP_varsRoot.existeRecargaMax = false
APP_varsRoot.isConstaTabelaOfertaFaleIlimitado = false
APP_varsRoot.isMaillingParalelaPromo = false
APP_varsRoot.isElegivelNovaPromo = false
APP_varsRoot.possuiDDDPacoteBonusMix = true
APP_varsRoot.isOiContaTotal = false
APP_varsRoot.possuiDDDPacoteBonusExtra = true
APP_varsRoot.possuiDDDPacoteSMS = true
APP_varsRoot.possuiCreditoEspecialIN = false
APP_varsRoot.possuiPredileto = false
APP_varsRoot.dddInternetMovelPrePaga = true
APP_varsRoot.isPosPago    = false
```

```
            APP_varsRoot.isDddElegivelPromoSorteioOiPaggo = false
            APP_varsRoot.isInvertido  = false
            APP_varsRoot.bolsoOiFixoPromocionalComecaHoje = false
            APP_varsRoot.isPrePago    = true
            APP_varsRoot.possuiCadastroM4U = false
            APP_varsRoot.atingiuRecargaMaxMesAtual = false
            APP_varsRoot.elegivelInternetMovelPrePaga = false
            APP_varsRoot.constaTabelaOfertaSuperBonus = false
            APP_varsRoot.isElegivelBCE = false
            APP_vars.constaNaBase     = true
            APP_vars.M4UBeanExecutorExecutionTime = 31
            APP_vars.idBeneficio      = 44
            APP_vars.parametroInformaDnis = false
            APP_vars.M4URecargaPrePagoExecutor = Success
            APP_vars.possuiPromo31Anos = false
            APP_vars.erroConsultaSiebel = false
            APP_vars.SiebelPrincipalExecutorExecutionTime = 141
            APP_vars.statusCode       = 0
            APP_vars.promocaoNaoIdentificada = true
            APP_vars.M4URecargaPrePagoExecutorExecutionTime = 63
            APP_vars.isElegivelPesquisa = false
            APP_vars.M4URecargaPrePagoExecutorRestExecutionTime = 312
            APP_vars.isCanalDiretoRecarga = false
            APP_vars.SiebelInativosExecutor = Success
            APP_vars.isErrorM4uPrepago = false
            APP_vars.PerfilInExecutorExecutionTime = 422
            APP_vars.M4UBeanExecutor  = Success
            APP_vars.SiebelInativosExecutorExecutionTime = 0
            APP_vars.SiebelPrincipalExecutor = Success
            APP_vars.isSucessoConsultaM4U = true
            APP_vars.dnis             = tel:12066
            APP_vars.M4URecargaControleExecutor = Success
            APP_vars.aniDisponivel    = true
            APP_vars.INChamadaConsulta = true
            APP_vars.ani              = 3488001222
            APP_vars.hour             = 0
            APP_vars.isPesquisaOn     = false
            APP_vars.PerfilInExecutor = Success
            APP_vars.erroConsultaIN   = false
            APP_vars.emManutencao     = false
            APP_vars.INConsultaMonUra = true
            APP_vars.M4URecargaControleExecutorExecutionTime = 203
            APP_vars.M4URecargaPrePagoExecutorRest = Success
            APP_vars.isClienteCorporativo = false
            APP_vars.idPrograma       = 35
            ENTERING_FORM_ITEM        = block:playBlock
            TRANSITION                                                  =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=Inicio&ura=URACTX11RJOED&ctiport=104&ani=tel:34880012
22&dnis=tel:12066                          resulted                        in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=TriagemInativos
            EXECUTING_URL                                               =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
```

```
action=transition&dialog=TriagemInativos<>http://10.6.60.42:8090/uracampanha/v
xml/oiatende-root.jsp
             ENTERING_FORM             = #TriagemInativos
             ENTERING_FORM_ITEM        = block:_0
             ENTERING_FORM_ITEM        = block:playBlock
             TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=TriagemInativos          resulted          in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=TriagemDesbloqueios
             EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=TriagemDesbloqueios<>http://10.6.60.42:8090/uracampan
ha/vxml/oiatende-root.jsp
             ENTERING_FORM             = #TriagemDesbloqueios
             ENTERING_FORM_ITEM        = block:_0
             APP_varsRoot.constaTabelaOfertaDegrau7 = false
             APP_varsRoot.promoExpirada = false
             APP_varsRoot.isPossuiPromoParalela = false
             APP_varsRoot.isPossuiOfertaFaleIlimitado = false
             APP_vars.isPossuiPromoTemplateSecundariaControle = false
             APP_vars.isControleSemFatura = false
             APP_vars.ClienteJaOuviu2Vezes = false
             APP_vars.isPrePago        = true
             APP_vars.isPossuiPromoTemplateSecundariaCartao = true
             APP_vars.promocaoIdentificada = true
             APP_vars.isSemPromo       = false
             APP_vars.isBloqueioCadastro = false
             ENTERING_FORM_ITEM        = block:playBlock
             TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=TriagemDesbloqueios        resulted        in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=Welcome
             EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=Welcome<>http://10.6.60.42:8090/uracampanha/vxml/oiat
ende-root.jsp
             ENTERING_FORM             = #Welcome
             ENTERING_FORM_ITEM        = block:_0
             APP_vars.isPossuiPromoTemplateSecundariaControle = false
             APP_vars.aptoNotificacaoBolhaEstimuloRecargaControle = true
             APP_vars.isFinalAniWelcome = false
             APP_vars.isPossuiOferta   = true
             APP_vars.isPossuiPromoTemplateSecundariaCartao = true
             APP_vars.isDddWelcome     = true
             APP_vars.isMailingNotificacaoBolhaEstimuloRecargaControle = false
             APP_vars.isElegivelBolhaRecargaWelcome = true
             APP_vars.uraNovaPacote    = true
             APP_vars.isMailingControleConectado = false
             APP_vars.ativaIncentivoOferta = true
             APP_vars.isSaldoExpirado  = false
             APP_vars.isMailingOiGaleraInformaAdesao = false
             APP_vars.bolsoRecargaExpiraUmDia = false
             APP_vars.statusClienteUraPersonalizadaOK = false
```

```
            APP_vars.isIncentivoOferta = false
            APP_vars.incentivoRecargaDDD = true
            APP_vars.aptoNotificacaoIncentivoBolhaControleConectado = true
            APP_vars.isSaldoMenorQueDezReais = true
            APP_vars.isDataDisponibilidade = false
            APP_vars.bolsoRecargaExpiraDoisDias = false
            ENTERING_FORM_ITEM        = block:_1
            APP_controle              = ICR1_00001
            APP_controle              = ICR1_00011
            APP_controle              = ICR1_00013
            APP_controle              = ICR1_00024
            APP_controle              = ICR1_00028
            APP_controle              = ICR1_00014
            ENTERING_FORM             = #TriagemOfertas_SaldoValido_init1
            ENTERING_FORM_ITEM        = block:_3
            APP_vars.isPossuiPromoTemplateSecundariaControle = false
            APP_vars.aptoNotificacaoBolhaEstimuloRecargaControle = true
            APP_vars.isFinalAniWelcome = false
            APP_vars.isPossuiOferta   = true
            APP_vars.isPossuiPromoTemplateSecundariaCartao = true
            APP_vars.isDddWelcome     = true
            APP_vars.isMailingNotificacaoBolhaEstimuloRecargaControle = false
            APP_vars.isElegivelBolhaRecargaWelcome = true
            APP_vars.uraNovaPacote    = true
            APP_vars.isMailingControleConectado = false
            APP_vars.ativaIncentivoOferta = true
            APP_vars.isSaldoExpirado  = false
            APP_vars.isMailingOiGaleraInformaAdesao = false
            APP_vars.bolsoRecargaExpiraUmDia = false
            APP_vars.statusClienteUraPersonalizadaOK = false
            APP_vars.isIncentivoOferta = false
            APP_vars.incentivoRecargaDDD = true
            APP_vars.aptoNotificacaoIncentivoBolhaControleConectado = true
            APP_vars.isSaldoMenorQueDezReais = true
            APP_vars.isDataDisponibilidade = false
            APP_vars.bolsoRecargaExpiraDoisDias = false
            ENTERING_FORM_ITEM        = block:playBlock
            ENTERING_FORM_ITEM        = field:option
            PROMPTER_PATH             = None
            PROMPTS                   = -tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 76

      http://10.6.60.42:8090/uracampanha/prompts/UPR1_Inicio_SaudacaoInicial_c
.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 9

      pause:250;-tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 70

      http://10.6.60.42:8090/uracampanha/prompts/UPR1_Inicio_Abertura_v2.wav;-
tts_text: SPEAK
```

```
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 9

       pause:250;-tts_text: SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 84

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_TriagemOfertas_SaldoVali
do_init1.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 48

       http://10.6.60.42:8090/uracampanha/prompts/2.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 52

       http://10.6.60.42:8090/uracampanha/prompts/reais.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 50

       http://10.6.60.42:8090/uracampanha/prompts/And.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 49

       http://10.6.60.42:8090/uracampanha/prompts/84.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 52

       http://10.6.60.42:8090/uracampanha/prompts/Cents.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 87

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_TriagemOfertas_SaldoVali
do_init2_V3.wav
              GRAMMAR                    = RECOGNIZE
       Content-Type:application/x-nuance-gsl
       Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#TriagemOfertas
_SaldoValido_init1<>field:option
       Nuance-Package-Name:pt-BR
       Nuance-Config-Name:pt-BR
       Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=Welcome
       Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                          birth,                          prePaid,
expirationDate,                                                   paymentType,
prePaid,                                                        valueRecharge,
```

```
endCardNumber,                    validationValue,                    cpf,
cvv,                          randomChoose,                        endCard1,
endCard2,                                                      endCardNumberUser,
paymentTypeUser,                                              M4URecargaBandeira,
bandeira1,                      bandeira2,                          bandeira3,
numerocartao,                                                 dataValidadeCartao,
dtNasc,                                                          cpfClienteNovo,
scvv";client.InputModes="dtmf";client.TooMuchSpeechTimeoutSecs="10.0";rec.Conf
idenceRejectionThreshold="34";vrs.ConnectAt="start-
utt";vxml.interdigittimeout="1.5";
       Cookie-Store:http://URACTX11RJOED:16063/cookies/

       ;GSL2.0
       _VWS_:public[
       [(dtmf-1 )] {<vws_id 0>}
       [(dtmf-2 )] {<vws_id 1>}
       ]
              PARAMETERS                                                    =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
              GRAMMAR_LABEL                          =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#TriagemOfertas
_SaldoValido_init1<>field:option
              PROMPT_DELAY              = 0.125
              PROMPT_DURATION           = 25.813 (COMPLETED)
              STATUS                    = NO_SPEECH_TIMEOUT
              SERVER_HOSTNAME           = Unknown <connected via resource mgr>
              RECRESULT_ID              = id_1_-1_-1
              BARGE_IN                  = FALSE
              STATE_DURATION            = 30.969
       }end
       ;;;;;;;;;;
       start{
              TIME                      = Wed Apr 01 00:00:55 2015
              BEGIN_TIME                = 31.953
              SESSION_ID                = 0a063c1e_00001598_551b5f47_ade6_0160
              RECORD_NUMBER             = 2
              ENTERING_FORM_ITEM        = block:playBlock
              ENTERING_FORM_ITEM        = field:option
              PROMPTER_PATH             = None
              PROMPTS                   = -tts_text: SPEAK
       Resource-Name: AudioServer
```

```
       Content-Type: text/uri
       Content-Length: 66

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_OpcaoInvalida1.wav;-
tts_text: SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 84

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_TriagemOfertas_SaldoVali
do_init1.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 48

       http://10.6.60.42:8090/uracampanha/prompts/2.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 52

       http://10.6.60.42:8090/uracampanha/prompts/reais.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 50

       http://10.6.60.42:8090/uracampanha/prompts/And.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 49

       http://10.6.60.42:8090/uracampanha/prompts/84.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 52

       http://10.6.60.42:8090/uracampanha/prompts/Cents.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 87

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_TriagemOfertas_SaldoVali
do_init2_V3.wav
              GRAMMAR                    = RECOGNIZE
       Content-Type:application/x-nuance-gsl
       Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#TriagemOfertas
_SaldoValido_init1<>field:option
       Nuance-Package-Name:pt-BR
       Nuance-Config-Name:pt-BR
       Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=Welcome
       Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                          birth,                          prePaid,
expirationDate,                                                   paymentType,
prePaid,                                                        valueRecharge,
```

```
endCardNumber,                    validationValue,                    cpf,
cvv,                          randomChoose,                        endCard1,
endCard2,                                                      endCardNumberUser,
paymentTypeUser,                                              M4URecargaBandeira,
bandeira1,                      bandeira2,                          bandeira3,
numerocartao,                                                 dataValidadeCartao,
dtNasc,                                                          cpfClienteNovo,
scvv";client.InputModes="dtmf";client.TooMuchSpeechTimeoutSecs="10.0";rec.Conf
idenceRejectionThreshold="34";vrs.ConnectAt="start-
utt";vxml.interdigittimeout="1.5";
       Cookie-Store:http://URACTX11RJOED:16063/cookies/

       ;GSL2.0
       _VWS_:public[
       [(dtmf-1 )] {<vws_id 0>}
       [(dtmf-2 )] {<vws_id 1>}
       ]
              PARAMETERS                                                    =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
              GRAMMAR_LABEL                          =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#TriagemOfertas
_SaldoValido_init1<>field:option
              PROMPT_DELAY               = 0.094
              PROMPT_DURATION            = 3.937 (BARGE_IN)
              LAST_USER_INPUT_TIME       = 4.063
              STATUS                     = RECOGNITION
              SERVER_HOSTNAME            = Unknown <connected via resource mgr>
              RECRESULT_ID               = id_1_-1_-1
              NUM_RESULTS                = 1
              RESULT[0]                  = dtmf-2
              CONFIDENCE[0]              = 100
              PROBABILITY[0]             = 0
              NUM_NL_INTERPRETATIONS[0] = 1
              NL_INTERPRETATION[0][0]    = {<vws_id 1>}
              CONFIDENCE[0][0][vws_id]   = 0
              BARGE_IN                   = TRUE
              STATE_DURATION             = 4.063
       }end
       ;;;;;;;;;;
       start{
              TIME                       = Wed Apr 01 00:01:00 2015
```

```
            BEGIN_TIME               = 36.109
            SESSION_ID               = 0a063c1e_00001598_551b5f47_ade6_0160
            RECORD_NUMBER            = 3
            APP_controle             = ICR1_00008
            TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=Welcome             resulted             in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=MenuTipoDePlano
            EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=MenuTipoDePlano<>http://10.6.60.42:8090/uracampanha/v
xml/oiatende-root.jsp
            ENTERING_FORM            = #MenuTipoDePlano
            ENTERING_FORM_ITEM       = block:_0
            APP_varsRoot.promoExpirada = false
            APP_vars.isPossuiPromoTemplateSecundariaControle = false
            APP_vars.isPossuiPromoTemplateSecundariaCartao = true
            APP_vars.promocaoIdentificada = false
            APP_vars.possuiPromoOiGalera = false
            ENTERING_FORM_ITEM       = block:playBlock
            APP_controle             = ICR1_06702
            APP_controle             = ICR1_06704
            TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=MenuTipoDePlano         resulted          in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=ClientePromoFaleXCentavos
            EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=ClientePromoFaleXCentavos<>http://10.6.60.42:8090/ura
campanha/vxml/oiatende-root.jsp
            ENTERING_FORM            = #ClientePromoFaleXCentavos
            ENTERING_FORM_ITEM       = block:_0
            APP_vars.isDailyfeeFixo  = false
            APP_vars.isBolsoDeRecargaMaiorQue4_98 = false
            APP_vars.isDailyfeeOi    = false
            APP_vars.dddFCODailyFee  = true
            APP_vars.isRemainMinutesNull = true
            APP_vars.isTipoBonusIgualTarifaDiariaWifi = true
            APP_vars.isRecargaMininaNull = true
            APP_vars.isFCORchgInfoDtMaiorOuIgualAHj = true
            APP_vars.p3oferta        = true
            APP_vars.isOpcaoSomenteRecarga = false
            APP_vars.isDailyfeeOiFixo = false
            APP_vars.isClienteFCORchgInfo2Bolso = false
            APP_vars.isDailyfeeHoje  = false
            APP_vars.isClienteFCORchgInfo1Bolso = true
            APP_vars.dddOfertaTudoPorDia = false
            APP_vars.finalAniOfertaTudoPorDia = false
            APP_vars.dddFale10SaldoCartao = false
            APP_vars.dddFale10ValidadeCartao = false
            APP_vars.isBolsoDeRecargaMaiorQueTarifaSec = true
            ENTERING_FORM_ITEM       = block:playBlock
            ENTERING_FORM_ITEM       = field:option
```

```
              PROMPTER_PATH             = None
              PROMPTS                   = -tts_text: SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 95

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_ClientePromoFaleXCentavo
s_MenuPrincipal_op2.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 95

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_ClientePromoFaleXCentavo
s_MenuPrincipal_op3.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 95

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_ClientePromoFaleXCentavo
s_MenuPrincipal_op4.wav;-tts_text: SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 98

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_ClientePromoFaleXCentavo
s_MenuPrincipal_op5_V1.wav;-tts_text: SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 95

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_ClientePromoFaleXCentavo
s_MenuPrincipal_op6.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 95

       http://10.6.60.42:8090/uracampanha/prompts/UPR1_ClientePromoFaleXCentavo
s_MenuPrincipal_op7.wav
              GRAMMAR                   = RECOGNIZE
       Content-Type:application/srgs+xml
       Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#ClientePromoFa
leXCentavos<>field:option
       Nuance-Package-Name:pt-BR
       Nuance-Config-Name:pt-BR
       Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=ClientePromoFaleXCentavos
       Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                          birth,                          prePaid,
expirationDate,                                                   paymentType,
prePaid,                                                        valueRecharge,
endCardNumber,                    validationValue,                    cpf,
cvv,                         randomChoose,                        endCard1,
endCard2,                                                  endCardNumberUser,
paymentTypeUser,                                          M4URecargaBandeira,
```

```
bandeira1,                          bandeira2,                          bandeira3,
numerocartao,                                                  dataValidadeCartao,
dtNasc,                                                           cpfClienteNovo,
scvv";client.InputModes="dtmf";client.TooMuchSpeechTimeoutSecs="10.0";rec.Conf
idenceRejectionThreshold="34";vrs.ConnectAt="start-
utt";vxml.interdigittimeout="1.5";
       Cookie-Store:http://URACTX11RJOED:16063/cookies/

       <?xml version="1.0"  encoding="ISO-8859-1" standalone="no"?>
       <grammar                        xmlns="http://www.w3.org/2001/06/grammar"
xmlns:nuance="http://voicexml.nuance.com/grammar"   root="_VWS_"   xml:lang=""
version="1.0" tag-format="Nuance" mode="voice">
       <rule id="_VWS_" scope="public"><one-of>
       <item><tag>&lt;vws_id                                    0&gt;</tag><ruleref
uri="#UPR1_rootCapturaClientePromoFaleXCentavos1"/></item>
       <item><tag>&lt;vws_id 1&gt;</tag><one-of>
       <item>dtmf-2 </item>

       </one-of></item>
       <item><tag>&lt;vws_id 2&gt;</tag><one-of>
       <item>dtmf-3 </item>

       </one-of></item>
       <item><tag>&lt;vws_id 3&gt;</tag><one-of>
       <item>dtmf-4 </item>

       </one-of></item>
       <item><tag>&lt;vws_id 4&gt;</tag><one-of>
       <item>dtmf-5 </item>

       </one-of></item>
       <item><tag>&lt;vws_id 5&gt;</tag><one-of>
       <item>dtmf-6 </item>

       </one-of></item>
       <item><tag>&lt;vws_id 6&gt;</tag><one-of>
       <item>dtmf-7 </item>

       </one-of></item>
       </one-of></rule>
       <rule                    id="UPR1_rootCapturaClientePromoFaleXCentavos1"
scope="public"><one-of><item                                  repeat="1-4"><ruleref
uri="#UPR1_digitsRootCapturaClientePromoFaleXCentavos1"/></item><item>dtmf-2
</item><item>dtmf-3 </item><item>dtmf-4 </item><item>dtmf-5 </item><item>dtmf-
6 </item></one-of></rule>
       <rule              id="UPR1_digitsRootCapturaClientePromoFaleXCentavos1"
scope="public"><one-of><item>dtmf-8      </item><item>dtmf-0      </item></one-
of></rule>
       </grammar>
             PARAMETERS                                                        =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
```

```
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
            GRAMMAR_LABEL                    =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#ClientePromoFa
leXCentavos<>field:option
            PROMPT_DELAY             = 0.094
            PROMPT_DURATION          = 18.516 (BARGE_IN)
            LAST_USER_INPUT_TIME     = 18.641
            STATUS                   = RECOGNITION
            SERVER_HOSTNAME          = Unknown <connected via resource mgr>
            LINK_REPORT                                                  =
{<http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971
B40EC44FFC?action=transition&dialog=ClientePromoFaleXCentavos>              =>
{<special:NULL> => {<special:NULL>}}}
            RECRESULT_ID             = id_1_-1_-1
            NUM_RESULTS              = 1
            RESULT[0]                = dtmf-4
            CONFIDENCE[0]            = 100
            PROBABILITY[0]           = 0
            NUM_NL_INTERPRETATIONS[0] = 1
            NL_INTERPRETATION[0][0]  = {<vws_id 0>}
            CONFIDENCE[0][0][vws_id]  = 0
            BARGE_IN                 = TRUE
            STATE_DURATION           = 18.641
      }end
      ;;;;;;;;;;
      start{
            TIME                     = Wed Apr 01 00:01:18 2015
            BEGIN_TIME               = 54.797
            SESSION_ID               = 0a063c1e_00001598_551b5f47_ade6_0160
            RECORD_NUMBER            = 4
            TRANSITION                                                   =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=ClientePromoFaleXCentavos    resulted    in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=DesambigServicos
            EXECUTING_URL                                                =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=DesambigServicos<>http://10.6.60.42:8090/uracampanha/
vxml/oiatende-root.jsp
            ENTERING_FORM            = #DesambigServicos
            ENTERING_FORM_ITEM       = block:_0
            TRANSITION                                                   =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=DesambigServicos         resulted         in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPacotes
```

```
            EXECUTING_URL                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=UNPacotes<>http://10.6.60.42:8090/uracampanha/vxml/oi
atende-root.jsp
            ENTERING_FORM            = #UNPacotes
            ENTERING_FORM_ITEM       = block:_0
            APP_vars.parametroDDDPacoteIncentivoAtivo = true
            ENTERING_FORM_ITEM       = subdialog:ConsultaGetCustomerM4u
            EXECUTING_URL                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=UNSubM4uGetCustomer<>http://10.6.60.42:8090/uracampan
ha/servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC?
action=transition&dialog=UNSubM4uGetCustomer
            ENTERING_FORM            = #UNSubM4uGetCustomer
            ENTERING_FORM_ITEM       = block:_0
            APP_vars.statusCode      = 1000
            APP_vars.isFalhaM4u      = false
            APP_vars.isTemCartaoCadastrado = false
            APP_vars.isRetornoInvalido = false
            APP_vars.isSucesso       = true
            APP_retorno              = Retorno consulta GetCustomerM4u:1000
            ENTERING_FORM_ITEM       = block:playBlock
            APP_controle             = ICPA_MP001
            APP_controle             = ICPA_001
            ENTERING_FORM_ITEM       = field:option
            PROMPTER_PATH            = None
            PROMPTS                  = -tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 65

      http://10.6.60.42:8090/uracampanha/prompts/IncentivoPacote_01.wav;-
tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 58

      http://10.6.60.42:8090/uracampanha/prompts/pacotes_op1.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 58

      http://10.6.60.42:8090/uracampanha/prompts/pacotes_op2.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 58

      http://10.6.60.42:8090/uracampanha/prompts/pacotes_op3.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 58

      http://10.6.60.42:8090/uracampanha/prompts/pacotes_op4.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
```

```
        Content-Length: 69

        http://10.6.60.42:8090/uracampanha/prompts/tratar_outros_assuntos.wav
               GRAMMAR                    = RECOGNIZE
        Content-Type:application/x-nuance-gsl
        Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPacotes<>fie
ld:option
        Nuance-Package-Name:pt-BR
        Nuance-Config-Name:pt-BR
        Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=UNPacotes
        Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                            birth,                            prePaid,
expirationDate,                                                        paymentType,
prePaid,                                                             valueRecharge,
endCardNumber,                       validationValue,                        cpf,
cvv,                            randomChoose,                           endCard1,
endCard2,                                                         endCardNumberUser,
paymentTypeUser,                                                 M4URecargaBandeira,
bandeira1,                          bandeira2,                           bandeira3,
numerocartao,                                                    dataValidadeCartao,
dtNasc,                                                             cpfClienteNovo,
scvv";client.InputModes="dtmf";client.NoSpeechTimeoutSecs="3.0";client.TooMuch
SpeechTimeoutSecs="10.0";rec.ConfidenceRejectionThreshold="34";rec.Interpretat
ionEngine="full";vrs.ConnectAt="start-utt";vxml.interdigittimeout="1.5";
        Cookie-Store:http://URACTX11RJOED:16063/cookies/

        ;GSL2.0
        _VWS_:public[
        [(dtmf-1 )] {<vws_id 0>}
        [(dtmf-2 )] {<vws_id 1>}
        [(dtmf-3 )] {<vws_id 2>}
        [(dtmf-4 )] {<vws_id 3>}
        [(dtmf-0 )] {<vws_id 4>}
        ]
               PARAMETERS                                                        =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
```

```
            GRAMMAR_LABEL                       =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPacotes<>fie
ld:option
            PROMPT_DELAY              = 0.094
            PROMPT_DURATION           = 24.828 (BARGE_IN)
            LAST_USER_INPUT_TIME      = 24.937
            STATUS                    = RECOGNITION
            SERVER_HOSTNAME           = Unknown <connected via resource mgr>
            RECRESULT_ID              = id_1_-1_-1
            NUM_RESULTS               = 1
            RESULT[0]                 = dtmf-2
            CONFIDENCE[0]             = 100
            PROBABILITY[0]            = 0
            NUM_NL_INTERPRETATIONS[0] = 1
            NL_INTERPRETATION[0][0]   = {<vws_id 1>}
            CONFIDENCE[0][0][vws_id]  = 0
            BARGE_IN                  = TRUE
            STATE_DURATION            = 24.953
      }end
      ;;;;;;;;;;
      start{
            TIME                      = Wed Apr 01 00:01:43 2015
            BEGIN_TIME                = 79.766
            SESSION_ID                = 0a063c1e_00001598_551b5f47_ade6_0160
            RECORD_NUMBER             = 5
            APP_controle              = ICPA_DM001
            TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPacotes            resulted            in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPctDadosMontagem
            EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=UNPctDadosMontagem<>http://10.6.60.42:8090/uracampanh
a/vxml/oiatende-root.jsp
            ENTERING_FORM             = #UNPctDadosMontagem
            ENTERING_FORM_ITEM        = block:_0
            APP_vars.saldoValidoMaiorIgualPctDadosExtra15MB = true
            APP_vars.saldoValidoMaiorIgualPctDadosExtra50MB = false
            APP_vars.saldoValidoMaiorIgualPctDadosExtra10MB = true
            APP_vars.saldoValidoMaiorIgualMenorPacote = true
            APP_vars.saldoValidoMaiorIgualPctDados40MB = false
            ENTERING_FORM_ITEM        = block:playBlock
            ENTERING_FORM_ITEM        = field:option
            PROMPTER_PATH             = None
            PROMPTS                   = -tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 70

      http://10.6.60.42:8090/uracampanha/prompts/Pct-Dados-Montagem_1_V3.wav;-
tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 81
```

```
      http://10.6.60.42:8090/uracampanha/prompts/Pct-dados-Ja-sabe-como-
funciona_V2.wav
            GRAMMAR                    = RECOGNIZE
      Content-Type:application/x-nuance-gsl
      Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPctDadosMont
agem<>field:option
      Nuance-Package-Name:pt-BR
      Nuance-Config-Name:pt-BR
      Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=UNPctDadosMontagem
      Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                          birth,                          prePaid,
expirationDate,                                                  paymentType,
prePaid,                                                       valueRecharge,
endCardNumber,                    validationValue,                    cpf,
cvv,                          randomChoose,                        endCard1,
endCard2,                                                   endCardNumberUser,
paymentTypeUser,                                           M4URecargaBandeira,
bandeira1,                       bandeira2,                       bandeira3,
numerocartao,                                              dataValidadeCartao,
dtNasc,                                                       cpfClienteNovo,
scvv";client.InputModes="dtmf";client.NoSpeechTimeoutSecs="3.0";client.TooMuch
SpeechTimeoutSecs="10.0";rec.ConfidenceRejectionThreshold="34";rec.Interpretat
ionEngine="full";vrs.ConnectAt="start-utt";vxml.interdigittimeout="1.5";
      Cookie-Store:http://URACTX11RJOED:16063/cookies/

      ;GSL2.0
      _VWS_:public[
      [(dtmf-1 )] {<vws_id 0>}
      [(dtmf-2 )] {<vws_id 1>}
      [(dtmf-3 )] {<vws_id 2>}
      [(dtmf-4 )] {<vws_id 3>}
      ]
            PARAMETERS                                                    =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
            GRAMMAR_LABEL                        =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPctDadosMont
agem<>field:option
```

```
            PROMPT_DELAY              = 0.078
            PROMPT_DURATION           = 38.703 (BARGE_IN)
            LAST_USER_INPUT_TIME      = 38.828
            STATUS                    = RECOGNITION
            SERVER_HOSTNAME           = Unknown <connected via resource mgr>
            RECRESULT_ID              = id_1_-1_-1
            NUM_RESULTS               = 1
            RESULT[0]                 = dtmf-3
            CONFIDENCE[0]             = 100
            PROBABILITY[0]            = 0
            NUM_NL_INTERPRETATIONS[0] = 1
            NL_INTERPRETATION[0][0]   = {<vws_id 2>}
            CONFIDENCE[0][0][vws_id]  = 0
            BARGE_IN                  = TRUE
            STATE_DURATION            = 38.828
      }end
      ;;;;;;;;;;
      start{
            TIME                      = Wed Apr 01 00:02:22 2015
            BEGIN_TIME                = 118.609
            SESSION_ID                = 0a063c1e_00001598_551b5f47_ade6_0160
            RECORD_NUMBER             = 6
            APP_controle              = ICDM_PSA004
            TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPctDadosMontagem        resulted        in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPctDadosAssinaturaMontagem
            EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=UNPctDadosAssinaturaMontagem<>http://10.6.60.42:8090/
uracampanha/vxml/oiatende-root.jsp
            ENTERING_FORM             = #UNPctDadosAssinaturaMontagem
            ENTERING_FORM_ITEM        = block:_0
            APP_vars.saldoValidoMaiorIgualPctDAM2 = false
            APP_vars.saldoValidoMaiorIgualPctDAM1 = false
            APP_vars.saldoValidoMaiorIgualPctDAM3 = false
            ENTERING_FORM_ITEM        = block:playBlock
            APP_controle              = ICDAM015
            ENTERING_FORM_ITEM        = field:option
            PROMPTER_PATH             = None
            PROMPTS                   = -tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 61

      http://10.6.60.42:8090/uracampanha/prompts/funciona_assim.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 81

      http://10.6.60.42:8090/uracampanha/prompts/Pct-Dados-Recorrente-
Montagem_3_V3.wav, SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 60
```

```
       http://10.6.60.42:8090/uracampanha/prompts/menu_montagem.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 69

       http://10.6.60.42:8090/uracampanha/prompts/tratar_outros_assuntos.wav
              GRAMMAR                    = RECOGNIZE
       Content-Type:application/x-nuance-gsl
       Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPctDadosAssi
naturaMontagem<>field:option
       Nuance-Package-Name:pt-BR
       Nuance-Config-Name:pt-BR
       Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=UNPctDadosAssinaturaMontagem
       Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                          birth,                          prePaid,
expirationDate,                                                   paymentType,
prePaid,                                                        valueRecharge,
endCardNumber,                    validationValue,                    cpf,
cvv,                           randomChoose,                       endCard1,
endCard2,                                                    endCardNumberUser,
paymentTypeUser,                                            M4URecargaBandeira,
bandeira1,                        bandeira2,                        bandeira3,
numerocartao,                                               dataValidadeCartao,
dtNasc,                                                        cpfClienteNovo,
scvv";client.InputModes="dtmf";client.NoSpeechTimeoutSecs="3.0";client.TooMuch
SpeechTimeoutSecs="10.0";rec.ConfidenceRejectionThreshold="34";rec.Interpretat
ionEngine="full";vrs.ConnectAt="start-utt";vxml.interdigittimeout="1.5";
       Cookie-Store:http://URACTX11RJOED:16063/cookies/

       ;GSL2.0
       _VWS_:public[
       [(dtmf-1 )] {<vws_id 0>}
       [(dtmf-2 )] {<vws_id 1>}
       [(dtmf-0 )] {<vws_id 2>}
       ]
              PARAMETERS                                                    =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
```

```
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
             GRAMMAR_LABEL                        =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPctDadosAssi
naturaMontagem<>field:option
             PROMPT_DELAY              = 0.063
             PROMPT_DURATION           = 44.094 (BARGE_IN)
             LAST_USER_INPUT_TIME      = 44.188
             STATUS                    = RECOGNITION
             SERVER_HOSTNAME           = Unknown <connected via resource mgr>
             RECRESULT_ID              = id_1_-1_-1
             NUM_RESULTS               = 1
             RESULT[0]                 = dtmf-1
             CONFIDENCE[0]             = 100
             PROBABILITY[0]            = 0
             NUM_NL_INTERPRETATIONS[0] = 1
             NL_INTERPRETATION[0][0]   = {<vws_id 0>}
             CONFIDENCE[0][0][vws_id]  = 0
             BARGE_IN                  = TRUE
             STATE_DURATION            = 44.188
      }end
      ;;;;;;;;;;
      start{
             TIME                      = Wed Apr 01 00:03:06 2015
             BEGIN_TIME                = 162.812
             SESSION_ID                = 0a063c1e_00001598_551b5f47_ade6_0160
             RECORD_NUMBER             = 7
             ENTERING_FORM             = #UNPctDadosAssinaturaMontagem_01
             ENTERING_FORM_ITEM        = block:_1
             APP_vars.saldoValidoMaiorIgualPctDAM2 = false
             APP_vars.saldoValidoMaiorIgualPctDAM1 = false
             APP_vars.saldoValidoMaiorIgualPctDAM3 = false
             APP_controle              = ICDAM014
             APP_controle              = ICDAM008
             TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPctDadosAssinaturaMontagem   resulted   in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPctDadosSemSaldoAssinatura
             EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=UNPctDadosSemSaldoAssinatura<>http://10.6.60.42:8090/
uracampanha/vxml/oiatende-root.jsp
             ENTERING_FORM             = #UNPctDadosSemSaldoAssinatura
             ENTERING_FORM_ITEM        = block:_0
             ENTERING_FORM_ITEM        = block:playBlock
             APP_controle              = ICDAM_MP001
             ENTERING_FORM_ITEM        = field:option
             PROMPTER_PATH             = None
             PROMPTS                   = -tts_text: SPEAK
      Resource-Name: AudioServer
      Content-Type: text/uri
      Content-Length: 78

      http://10.6.60.42:8090/uracampanha/prompts/PctDadosSemSaldoAssinatura_1_
V2.wav, SPEAK
```

```
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 75

       http://10.6.60.42:8090/uracampanha/prompts/PctDadosSemSaldoAssinatura_2.
wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 75

       http://10.6.60.42:8090/uracampanha/prompts/PctDadosSemSaldoAssinatura_3.
wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 78

       http://10.6.60.42:8090/uracampanha/prompts/PctDadosSemSaldoAssinatura_4_
V2.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 78

       http://10.6.60.42:8090/uracampanha/prompts/PctDadosSemSaldoAssinatura_5_
V2.wav, SPEAK
       Resource-Name: AudioServer
       Content-Type: text/uri
       Content-Length: 75

       http://10.6.60.42:8090/uracampanha/prompts/PctDadosSemSaldoAssinatura_0.
wav
              GRAMMAR                    = RECOGNIZE
       Content-Type:application/x-nuance-gsl
       Nuance-Grammar-Label:/uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPctDadosSemS
aldoAssinatura<>field:option
       Nuance-Package-Name:pt-BR
       Nuance-Config-Name:pt-BR
       Content-
Base:http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B
971B40EC44FFC?action=transition&dialog=UNPctDadosSemSaldoAssinatura
       Vendor-Specific-
Parameters:behavior.calllog.ConfidentialURLParameters="state,
creditCard,                          birth,                          prePaid,
expirationDate,                                                   paymentType,
prePaid,                                                        valueRecharge,
endCardNumber,                    validationValue,                      cpf,
cvv,                           randomChoose,                        endCard1,
endCard2,                                                  endCardNumberUser,
paymentTypeUser,                                          M4URecargaBandeira,
bandeira1,                        bandeira2,                        bandeira3,
numerocartao,                                             dataValidadeCartao,
dtNasc,                                                      cpfClienteNovo,
scvv";client.InputModes="dtmf";client.NoSpeechTimeoutSecs="3.0";client.TooMuch
SpeechTimeoutSecs="10.0";rec.ConfidenceRejectionThreshold="34";rec.Interpretat
ionEngine="full";vrs.ConnectAt="start-utt";vxml.interdigittimeout="1.5";
       Cookie-Store:http://URACTX11RJOED:16063/cookies/
```

```
      ;GSL2.0
      _VWS_:public[
      [(dtmf-1 )] {<vws_id 0>}
      [(dtmf-2 )] {<vws_id 1>}
      [(dtmf-3 )] {<vws_id 2>}
      [(dtmf-4 )] {<vws_id 3>}
      [(dtmf-0 )] {<vws_id 4>}
      ]
            PARAMETERS                                                    =
audio.Provider="dialogic";audio.ReadyForNextCall="TRUE";audio.dialogic.Channel
ID="1691670";audio.dialogic.Lines="91-
120";audio.dialogic.SecondaryDevice="";audio.sip.Lines="91-
120";audio.sip.UserAgentPort="5060";audio.sip.UserAgentURI="sip:nvp@URACTX11RJ
OED";client.ChannelId="0160";client.KillPlaybackOnDTMF="FALSE";client.PopConte
xt="";client.PushContext="client.InputModes=\"dtmf\";client.KillPlaybackOnDTMF
=\"TRUE\"";client.TTSAddresses="10.6.60.30:27404";config.ClapiConfigFile="D:\N
VP\vws/conf/clapi_conf.xml";config.LogFileNamePrefix="vbs4-
jni";config.LogFileRootDir="D:\NVP/logs";config.LoggingLevel="STATUS";config.M
anagementStationHost="10.6.60.34:8080";config.RecClientPort="7882";config.Serv
iceID="daae7e42-0a06-3c22-00d1-
c7788221088e";egr.builtin.context="http://localhost:8120/builtin/query/";ep.Ba
rgeInInitialNoiseFloor="-22";ep.BargeInMinNoiseFloor="-
42";rec.RouterDirectory="D:\NVP\vws/data/lang/Router/callrouter.rtr";rm.Addres
ses="GERCTX01RJOED,GERCTX02RJOED";sc.UsedInJVM="TRUE";vrs.CompilationServerTim
eoutMs="120000";vrs.DatabaseOperationTimeoutMs="120000";vrs.InstallSignalHandl
er="FALSE";
            GRAMMAR_LABEL                        =  /uracampanha/vxml/oiatende-
root.jsp<>servlet;jsessionid=443824B2736D1AD432B971B40EC44FFC<>#UNPctDadosSemS
aldoAssinatura<>field:option
            PROMPT_DELAY             = 0.094
            STATUS                   = HANG_UP
            BARGE_IN                 = FALSE
            STATE_DURATION           = 26.641
      }end
      ;;;;;;;;;;
      start{
            TIME                     = Wed Apr 01 00:03:33 2015
            BEGIN_TIME               = 189.453
            SESSION_ID               = 0a063c1e_00001598_551b5f47_ade6_0160
            RECORD_NUMBER            = 8
            PROMPT_DURATION          = 2.85557e+006 (HANG_UP)
            TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=UNPctDadosSemSaldoAssinatura   resulted   in
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=HangUp
            EXECUTING_URL                                                 =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?
action=transition&dialog=HangUp<>http://10.6.60.42:8090/uracampanha/vxml/oiate
nde-root.jsp
            ENTERING_FORM            = #HungUp
            ENTERING_FORM_ITEM       = block:_0
            ENTERING_FORM_ITEM       = block:_1
            APP_hup                  = USR
```

```
             TRANSITION                                                    =
http://10.6.60.42:8090/uracampanha/servlet;jsessionid=443824B2736D1AD432B971B4
0EC44FFC?action=transition&dialog=HangUp resulted in exit
             STATUS                    = LEFT_OVERS
             BARGE_IN                  = FALSE
             STATE_DURATION            = 0
       }end
       ;;;
       ;;;                                                                Closed
D:/NVP/logs/../callLogs/master/Accenture/UraCampanha/2015/04April/01/00/00-24-
URACTX11RJOED-E1-VOX-SC-line-104-LOG
       ;;; Wed Apr 01 00:03:33 2015
       ;;;
```

Anexo 2. Código para movimentar logs de URA simulando o processo de criação de logs de chamadas

```
# -*- coding: utf-8 -*-
import os
import shutil
import time
import datetime
import time
init=datetime.datetime.now()

allfiles=os.listdir('caminho de origem')


k=0
for i in allfiles:
	print [k,i]
	try:
		shutil.copy(os.path.join("caminho de origem'", i), "caminho  de destino")
		time.sleep(1)


	except:
		pass
	k=k+1
fim=datetime.datetime.now()
delta=fim-init
print delta.total_seconds()
```

Anexo 3. Código Spark Streaming

```
# -*- coding: utf-8 -*-
##############
# bibliotecas:
##############
from pyspark import SparkContext,SparkConf
from pyspark.streaming import StreamingContext
from pyspark.streaming.kafka import KafkaUtils
from pyspark.sql.types import *
from pyspark.sql.types import Row as Row
from pyspark.sql import functions as F
from pyspark.sql.window import Window
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType
from pyspark import sql as sql
from kafka import SimpleProducer, KafkaClient
import pandas as pd
import numpy as np
from operator import add
import pandas as pd
from graphframes import GraphFrame
from pyspark.sql.functions import concat, col, lit,unix_timestamp,udf,monotonically_increasing_id,col, round,lag, lead, first, last, desc
from time import strptime
import datetime
import time
from pyspark.sql.types import DoubleType,StringType,LongType
from pyspark.ml import Pipeline,PipelineModel
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler,StandardScaler
from pyspark.ml.clustering import KMeans,KMeansModel
from pyspark.ml.tuning import ParamGridBuilder, TrainValidationSplit
import pyspark.ml.evaluation as evaluation



init=datetime.datetime.now()

##########
#funcoes:
##########
# funcao 1
# lipeza de espacos e caracteres especiais
def clean_all(x):
    return x.strip(';').strip('\t').strip(' ').strip('').upper()

# funcao 2
# funcao Tag alvo
def interest(x0):
    x=x0.upper()
        if u'RECORD_NUMBER' in x or  u'SESSION_ID' in x  or  u'GRAMMAR_LABEL' in x or u'RESULT[0]' in x or u'STATE_DURATION' in x or u'PROMPT_DELAY' in x or  u'PROMPT_DURATION' in x  or u'TIME                =' in x  or 'CALL_REMOTE_URL' in x:
```

```
        return x

# funcao 3
def grammar_limpa(x):
    x1=x[x.find('#'):]
    return str(x1[1:x1.find('<')])

# funcao 4
def session_id_limpa(x):
    x1=x[x.find('=')+1:].strip()
    return x1

# funcao 5
def prompt_delay_limpa(x):
    return x[x.find('=')+1:].strip()

# funcao 6
def prompt_duration_limpa(x):
    x1=x[x.find('=')+1:].strip()
    return x1 #str(x1[1:x1.find('(')])

def prompt_tel_limpa(x):
    x1=x[x.find('=')+6:].strip()
    return x1 #str(x1[1:x1.find('(')])


# funcao 7
# ifelse
def ifelse(x):
    if  u'SESSION_ID' in x:
            return 'SESSION_ID='+ session_id_limpa(x)
    if  'RESULT[0]' in x:
            return 'RESULT='+prompt_delay_limpa(x)
    if u'GRAMMAR_LABEL' in x:
            return 'GRAMMAR_LABEL='+grammar_limpa(x)
    if  u'RECORD_NUMBER' in x:
            return 'RECORD_NUMBER='+prompt_delay_limpa(x)
    if  u'PROMPT_DELAY' in x:
            return 'PROMPT_DELAY='+prompt_delay_limpa(x)
    if  u'PROMPT_DURATION' in x:
            return 'PROMPT_DURATION='+prompt_duration_limpa(x)
    if  u'STATE_DURATION' in x:
            return 'STATE_DURATION='+prompt_delay_limpa(x)
    if  u'TIME' in x.upper():
            return 'TIME='+prompt_delay_limpa(x)
    if  u'CALL_REMOTE_URL' in x.upper():
            return 'TEL='+prompt_tel_limpa(x)

    else:
            return x


# funcao 8
# conta passos sessao
def contasession(x):
             session_cont=[k  for k in range(len(x)) if 'SESSION_ID' in str(x[k])]
```

```
                saida=[]
                for i in range(len(session_cont)):
                        if i==max(range(len(session_cont))):
                                atual=[i+1]+x[session_cont[i]:]
                                saida.append(atual)
                        else:
                             atual=[i+1]+x[session_cont[i]:session_cont[i+1]]
                             saida.append(atual)
                return saida


        #funcao 9
        def validaposicao(x):
              reposicao=['SESSION_ID','RECORD_NUMBER','GRAMMAR_LABEL','PROMPT_DELAY','PRO
MPT_DURATION','RESULT','STATE_DURATION','TIME','TEL' ]
                store=[]
                     index=[]
                contador_passo=0
                     linha=x[0]
                     store.append(linha)
                     for i in reposicao:
                             try:
                                     store.append(x[[i in str(s) for s in x].index(True)])
                             except ValueError:
                                     if i=='RECORD_NUMBER':
                                     store.append(str(i)+'NULL'+str(contador_passo))
                             if i in ['GRAMMAR_LABEL','RESULT']:
                                     if (True in ['HANG_UP' in str(s) for s in x ]):
                                                     store.append(str(i)+'='+str('HANG_UP'))
                                     else:
                                          store.append(str(i)+'='+str('NULL'))

                             else:
                                             store.append(str(i)+'=0')
                     contador_passo+=1
                     return     store

        ########################
        #### classe especial ###
        ########################
        class especial(object):
                def get_value(self,x):
                   tmp_store=[]
                   for i in x:
                             try:
                                tmp_store.append(str(i)[str(i).index('=')+1:])
                             except:
                                tmp_store.append(str(i))
                   return tmp_store
                def map_get_value(self,x):
                             return map(especial().get_value,x)
                def find_complete(self,x):
                             tmp_store2=[]
                             insta=especial().get_value(x)
                             for i in insta:
                                try:
```

```
                        tmp_store2.append(str(i)[:str(i).index('(COMPLETED)')].strip(' '))
                    except:
                        tmp_store2.append(str(i))
                return tmp_store2
            def map_complete(self,x):
                store3=[]
                for i in range(len(x)):
                    try :
                        store3.append(especial().find_complete(x[i]))
                    except:
                        store3.append(especial().get_value(x[i]))
                return store3
            def barge_in(self,x):
                store4=[]
                insta2=especial().find_complete(x)
                for i in insta2:
                    try:
                        store4.append(i[0:i.index('(BARGE_IN')].strip(' '))
                    except:
                        store4.append(i)
                return store4
            def map_barge_in(self,x):
                store5=[]
                for i in range(len(x)):
                        try:
                            store5.append(especial().barge_in(x[i]))
                        except:
                            store5.append(especial().barge_in(x[i]))
                return np.array(store5).transpose().tolist()
            def toint(self,x):
                insta3=especial().barge_in(x)
                store6=[]
                for i in insta3:
                    try:
                        store6.append(int(i))
                    except:
                        store6.append(i)
                return  store6
            def map_toint(self,x):
                store7=[]
                for i in range(len(x)):
                        try:
                            store7.append(especial().toint(int(x[i])))
                        except:
                            store7.append(especial().toint(x[i]))
                return np.asmatrix(store7).tolist()

class Hang_treat(object):
            def treat(self,x):
                saida=[]
                for i in x:
                        try:
                                posit=i.index(' (HANG_UP)')
                                saida.append(i[0:(posit-1)])
                        except:
                                saida.append(i)
```

```
                return saida
        ##########################
        # Convert para DtaFrame  #
        ##########################
        def getSparkSessionInstance(sparkConf):
          if ("sparkSessionSingletonInstance" not in globals()):
                                                globals()["sparkSessionSingletonInstance"]      =
SparkSession.builder.config(conf=sparkConf).getOrCreate()
          return globals()["sparkSessionSingletonInstance"]


        def change(x):
                tam=(len(x)-1)
                penultimo=x[tam-1][5]
                ultimo=x[tam][5]
                x[tam-1][5]=ultimo
                x[tam][5]=penultimo


                penultimo=x[tam-1][6]
                ultimo=x[tam][6]
                x[tam-1][6]=ultimo
                x[tam][6]=penultimo
                return x

        def trata_hang_up(x):
                        try:
                                return      x.replace(' (HANG_UP)','0')
                        except:
                                return x


        def try_int(x):
                try:
                        return int(x)
                except:
                        return x

        def reply_tel(x):
                tel=x[0][9]
                for i in range(len(x))[1:]:
                        x[i][9]=tel
                return x

        ############################
        # para criar chave primaria #
        ############################
        def float2int(x):
                return int(x)
        udffloat2int = udf(float2int, IntegerType())

        def datetime_to_float(d):
          epoch = datetime.datetime.utcfromtimestamp(0)
          total_seconds =  (d - epoch).total_seconds()
          # total_seconds will be in decimals (millisecond precision)
          return total_seconds
```

```
#pega data em python
def getdate():
	import datetime
	return datetime.datetime.now().strftime("%Y%m%d%H%M%S")

getdate_udf = udf(getdate, StringType())

def getdate2():
	import datetime
	return datetime.datetime.now()

def timeexec():
	import datetime
	return datetime.datetime.now()
#timeexec_udf = udf(timeexec, FloatType())

getdate_udf2 = udf(getdate2, TimestampType())

timestamp = datetime.datetime.fromtimestamp(time.time()).strftime('%Y-%m-%d %H:%M:%S')

def process(time, rdd):
   print("========= %s =========" % str(time))
   init=datetime.datetime.now()
   try:
      # Get the singleton instance of SparkSession
      spark = getSparkSessionInstance(rdd.context.getConf())
      # Convert RDD[String] to RDD[Row] to DataFrame

	rowRdd =rdd.map(change).flatMap(lambda x: x ).map(lambda x : Row(SESSION_ID=x[1],
PASSOS=int(x[0]),ED=x[3].strip(''),PROMPT_DELAY=float(x[4]),PROMPT_DURATION=float(x[7])-
float(x[4]),RESULT=x[6],STATE_DURATION=float(x[7]),WEEK=x[8][0:3],MONTH=x[8][4:7],DAY=try_int(x[8]
[8:10]),YEAR=x[8][20:25],TIME=x[8][11:19],TEL=x[9]))


	df_tmp = rowRdd.toDF()
	df_tmp2=df_tmp.withColumn('NO',concat(col('ED'),lit('-'),
col('RESULT'))).withColumn('PROMPT_DURATION',round(df_tmp["PROMPT_DURATION"],2).cast('double')).w
ithColumn('STATE_DURATION',round(df_tmp["STATE_DURATION"],2).cast('double'))

	df_tmp3=df_tmp2.withColumn('dst', lead('ED').over(Window.orderBy('PASSOS')))

	df_tmp4=df_tmp3


#df_tmp5=df_tmp4#.withColumn('data',getdate_udf2()).withColumn('step',1+monotonically_increasing_id())
	df_tmp5=df_tmp4.withColumn('step',df_tmp4['PASSOS'])


df_tmp6=df_tmp5.withColumn('key',concat(df_tmp5['PASSOS'].cast('string'),df_tmp5['SESSION_ID'])).withColumn
('DAY', df_tmp5['DAY'].cast('int')).withColumn('YEAR', df_tmp5['YEAR'].cast('int'))
	df_tmp6.createOrReplaceTempView("tab")

	v=spark.sql("select distinct ED as id, ED as name from tab")
```

```
e_tmp=spark.sql("select key, SESSION_ID as session_id, TEL as tel, PASSOS as passos, step, ED as src,  dst, RESULT as relationship, DAY as day ,MONTH as month,YEAR as year,TIME as time,STATE_DURATION as state_duration,PROMPT_DELAY as prompt_delay,PROMPT_DURATION as prompt_duration from tab order by key,passos,time").withColumn('num_mes',F.when(col('month')=='JAN','01').when(col('month') == 'FEB', '02').when(col('month') == 'MAR', '03').when(col('month') == 'APR', '04').when(col('month') == 'MAY', '05').when(col('month') == 'JUN', '06').when(col('month') == 'JUL', '07').when(col('month') == 'AUG', '08').when(col('month') == 'SEP', '09').when(col('month') == 'OCT', '10').when(col('month') == 'NOV', '11').when(col('month') == 'DEC', '12').otherwise('ERROR')).withColumn('num_day',F.when(col('day')>=10,col('day')).otherwise(concat(lit('0'),col('day') )))
                ##############
                # cria datas #
                ##############
                e_tmp1=e_tmp.withColumn('data_evento0',concat(e_tmp['year'],lit('-'),
e_tmp['num_mes'],lit('-'),e_tmp['num_day'],lit(' '),e_tmp['time']))
e_tmp2=e_tmp1.withColumn('data_evento1',col('data_evento0').cast('timestamp')).drop(e_tmp1.data_evento0)
                e_tmp3=e_tmp2.withColumn('data_evento',(F.unix_timestamp(e_tmp2['data_evento1'])-
10800).cast('timestamp')).drop(e_tmp2.data_evento1).withColumn('data_process_tmp',F.unix_timestamp(getdate_udf2()).cast('timestamp')).withColumn('data_process',(F.unix_timestamp(col('data_process_tmp'))-
7200).cast('timestamp')).drop(col('data_process_tmp'))
                e=e_tmp3.filter((e_tmp3['src']!='HANG_UP') & (e_tmp3['dst']!='NULL') | (e_tmp3['src']!
='NULL') & (e_tmp3['dst']!='NULL'))
                auxiliar=spark.sql("select distinct ED as id, SESSION_ID from tab")
                #print e.show(truncate=False)
                grafo=GraphFrame(v, e)
                graus=grafo.degrees.repartition(4).alias('graus')
                graus_in=grafo.inDegrees.repartition(4).alias('graus_in')
                graus_out=grafo.outDegrees.repartition(4).alias('graus_out')
                tempo=spark.sql("select ED as id,sum(STATE_DURATION) as state_duration,sum(PROMPT_DELAY) as prompt_delay,sum(PROMPT_DURATION) as prompt_duration from tab group by ED")
                join_tmp = graus.join(graus_in, graus.id == graus_in.id,'inner').drop(graus_in.id)
                join_tmp2= join_tmp.join(graus_out, join_tmp.id == graus_out.id,'inner').drop(graus_out.id)
                join_tmp3= join_tmp2.join(tempo, join_tmp2.id == tempo.id,'inner').drop(tempo.id)
                join_tmp4= join_tmp3.join(auxiliar, join_tmp3.id == auxiliar.id,'inner').drop(auxiliar.id)
                join_tmp5= join_tmp4
                join_output=join_tmp5.select(["SESSION_ID","id",
"degree","inDegree","outDegree","state_duration","prompt_delay","prompt_duration"]).withColumn("session_id",join_tmp5['SESSION_ID']).withColumn("indegree",join_tmp5['inDegree']).withColumn("outdegree",join_tmp5['outDegree'])

                #print join_output.dtypes
                join_output.createOrReplaceTempView("tab_model_tmp")
                tab_model_tmp=spark.sql("select SESSION_ID as session_id, sum(outDegree) as qtd_passos ,Max(outDegree) as max_outdegree, avg( outDegree ) as avg_outdegree,Max(inDegree) as max_indegree,AVG(inDegree) as avg_indegree, sum(STATE_DURATION) as total_duration from tab_model_tmp group by SESSION_ID").withColumn('step', lit('1'))#.withColumn('step',1+monotonically_increasing_id()).withColumn('data',getdate_udf2())


tab_model=tab_model_tmp.withColumn('key',concat(tab_model_tmp['step'],tab_model_tmp['SESSION_ID'])).drop(tab_model_tmp.step).select(["key","session_id","qtd_passos","total_duration","max_outdegree","avg_outdegree","max_indegree","avg_indegree"])
                ######################
                # kmeans cluster p-1#
```

```
modeldata=spark.read.format('org.apache.kudu.spark.kudu').option('kudu.master',"localhost:7051").option('kudu.table',"impala::default.model2").load()
                    kmeans = Pipeline.load('/user/vinicius/pipelinekmeans')

cluster=kmeans.fit(modeldata).transform(tab_model).select('session_id','cluster').withColumn('segment',F.when(col('cluster') == 0, 'Warning: High Menu Repetition').when(col('cluster') == 1, 'Warning: Many Steps').when(col('cluster') == 2, 'Regular: Quick call').when(col('cluster') == 3, 'Warning: Outlier/Error').otherwise('Regular: Common call')).alias('cluster')
                    #cluster_tmp2=cluster_tmp.withColumn('time',unix_timestamp(lit(timestamp),'yyyy-MM-dd HH:mm:ss').cast("timestamp")).withColumn('data_process',(F.unix_timestamp(col('time'))-7200).cast('timestamp')).drop(col('time'))
                    #cluster=cluster_tmp2.drop(col('data_process')).alias('cluster')
                    #print cluster.show(truncate = False)
                    #print cluster_tmp2.show(truncate = False)
                    ####################################
                    # marca a chamada com o cluster          p-2 #
                    edge_cluster=e.join(cluster, e.session_id == cluster.session_id,'inner').drop(cluster.session_id)
                    #print edge_cluster.show(truncate = False)
                    #print edge_cluster.dtypes
                    ######################################################################

#e.write.format('org.apache.kudu.spark.kudu').option('kudu.master',"localhost:7051").option('use_local_tz_for_unix_timestamp_conversions','true').option('kudu.table',"impala::default.EDGE2").mode("append").save()

#tab_model.write.format('org.apache.kudu.spark.kudu').option('kudu.master',"localhost:7051").option('kudu.table',"impala::default.model2").mode("append").save()
                    ##############################
                    # escreve tabelas cluster p-3#

#cluster_tmp2.write.format('org.apache.kudu.spark.kudu').option('kudu.master',"localhost:7051").option('kudu.table',"impala::default.cluster").mode("append").save()

edge_cluster.write.format('org.apache.kudu.spark.kudu').option('kudu.master',"localhost:7051").option('kudu.table',"impala::default.edge_cluster").mode("append").save()
                    ##########################################################################
                    #print e.show()
                    #print e.dtypes
                    #print chave_tmp.show()
                    timeexec=datetime.datetime.now()-init
                    timeexecrdd=timeexec.total_seconds()

id_tb=tab_model.select('session_id').withColumn('id2',lit('1').cast('string')).select(['id2','session_id'])
                    id_tb_tmp=id_tb.repartition(4).alias('tab1')
                    time_tb=sc.parallelize([timeexecrdd]).map(lambda x : Row(timeexecrdd=x)).toDF().withColumn('id2',lit('1').cast('string')).select(['id2','timeexecrdd'])
                    time_tb_tmp=time_tb.repartition(4).alias('tab2')

tempojoin=time_tb_tmp.crossJoin(id_tb_tmp).drop(id_tb_tmp.id2).drop(time_tb_tmp.id2).select("session_id","timeexecrdd")

tempojoin.write.format('org.apache.kudu.spark.kudu').option('kudu.master',"localhost:7051").option('kudu.table',"impala::default.time_tab").mode("append").save()
                    #print 'tempojoin'
                    #print tempojoin.show()
                    #print tempojoin.dtypes
```

```
            except:
                    pass
                    print
                    print "##############################################"
                print "Ainda Não Extestem Dados ou Houve Algum Erro!"
                    print "##############################################"
                    print
                    print



            ###################################
            # contexto streaming RDD read Kafka:
            ###################################

            print('Init ================== {} {}')
            conf = SparkConf().setMaster("yarn").setAppName("Streaming").set("spark.sql.crossJoin.enabled", "true")
            sc = SparkContext(conf=conf)
            ssc = StreamingContext(sc,1)
            sqlsc=sql.SQLContext(sc)
            kstream = KafkaUtils.createDirectStream(ssc, topics = ['new2'],kafkaParams = {"metadata.broker.list":
'localhost:9092',"consumer.timeout.ms":"100", "auto.offset.reset" :"largest" })
            print('Start ================== {} {}')



            ##########################
            # map, reduce e transform:
            ##########################


            log = kstream.map(lambda x: x[1])
            log_p1 = log.map(lambda x : filter(None,map(clean_all,x.split('\r\n'))))
            log_p2 = log_p1.map(lambda x: filter(interest,x))
            log_p3 = log_p2.map(lambda x: map(ifelse,x)).map(lambda x: contasession(x)).map(lambda x:
map(validaposicao,x)).map(lambda x: especial().map_toint(x)).map(reply_tel)
            log_p4 = log_p3.reduce(add).cache().foreachRDD(process)


            ################
            # print screen #
            ################
            #log_p3.pprint()
            ssc.start()             # Start the computation
            #
            ssc.awaitTermination()  # Wait for the computation to terminate

            ssc.stop()
```