MÁRCIO HENRIQUE ALEXANDRE COSTA

DETECÇÃO DE ANOMALIAS EM APIS RESTFUL GOVERNADAS POR SOLUÇÕES DE IAM APLICADAS A TRANSAÇÕES PIX VIA MOBILE BANKING

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de Pós-Graduação em Engenharia de Dados e Big Data.

SÃO PAULO

2024

MÁRCIO HENRIQUE ALEXANDRE COSTA

DETECÇÃO DE ANOMALIAS EM APIS RESTFUL GOVERNADAS POR SOLUÇÕES DE IAM APLICADAS A TRANSAÇÕES PIX VIA MOBILE BANKING

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de Pós-Graduação em Engenharia de Dados e Big Data.

Área de concentração: Tecnologia da Informação – Engenharia/ Tecnologia/ Gestão

Orientador: Prof. Dr. Jorge Luis Risco Becerra

SÃO PAULO

2024

FICHA CATALOGRÁFICA

Costa, Márcio
Detecção de anomalias em APIs RESTful governadas por soluções de IAM aplicadas a transações Pix via mobile banking /
São Paulo, 2024.
70 p.

Monografia (Especialização em Engenharia de Dados e Big Data) – Escola Politécnica da Universidade de São Paulo. PECE – Programa de Educação Continuada em Engenharia.

1.Mineração de Dados

Universidade de São Paulo. Escola Politécnica. PECE – Programa de Educação Continuada em Engenharia II.t.

AGRADECIMENTOS

À minha família, cujo suporte incondicional e contínuo me permitiu a conclusão desta etapa em minha vida.

A todos os professores, que mostraram o caminho.

Ao meu orientador, Prof. Dr. Jorge Luis Risco Becerra, pelos inúmeros “ajustes de rumo” ao longo desta monografia.

CURSO ENGENHARIA DE BIG DATA

Coord.: Prof. Solange N. Alves de Souza
Vice-Coord.: Prof. Anarosa Alves Franco Brandão

Perspectivas profissionais alcançadas com o curso:

Venho atuando na área de software há muitos anos, principalmente através de empresas de consultoria. Há cerca de 2 anos passei a investir em um projeto pessoal e percebi que, para não ser apenas mais uma solução no mercado, seria necessária uma abordagem diferenciada, particularmente na área de aprendizado de máquina. Este curso me forneceu as bases e ferramentas para a sua materialização (ainda em construção, até o momento da escrita desse depoimento). Além disso, a formação em Engenharia de Dados e Big Data não somente abriu novas perspectivas de carreira, mas também representou uma evolução no projeto de soluções: os dados nunca mais serão vistos como meros resultados de transações.

RESUMO

Aplicativos de Mobile Banking modernos, executados a partir de smartphones, acessam os serviços bancários disponibilizados por seus respectivos bancos, como pagamentos via Pix, pelo uso de APIs REST e com autorização baseada em RBAC. Contudo, em caso de roubo dos aparelhos, suas contas podem ser comprometidas por transações não comumente realizadas por seus usuários legítimos, podendo ser classificadas como anomalias em função dos seus históricos de uso. Este trabalho tem como objetivo a construção de um software para a detecção de anomalias em transações bancárias via Pix baseado no algoritmo S-H-ESD (Seasonal Hybrid Extreme Studentized Deviate) combinado a tecnologias de Big Data.

Palavras-chave: ANOMALIA, API, REST, IAM, RBAC, S-H-ESD, PIX.

ABSTRACT

Modern mobile banking applications, running on smartphones, access banking services provided by their respective banks, such as payments via Pix, using REST APIs and with RBAC-based authorization. However, if the devices are stolen, their accounts may be compromised by transactions not commonly performed by their legitimate users, which may be classified as anomalies based on their usage history. This work aims to build software for detecting anomalies in banking transactions via Pix based on the S-H-ESD (Seasonal Hybrid Extreme Studentized Deviate) algorithm combined with Big Data technologies.

Keywords: ANOMALY, API, REST, IAM, RBAC, S-H-ESD, PIX.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIAÇÕES

API	Application Programming Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IAM	Identity and Access Management
JSON	JavaScript Object Notation
JWT	Json Web Token
NBDRA	NIST Big Data Reference Architecture
NIST	National Institute of Standards and Technology
PSP	Prestador de Serviços de Pagamento
RBAC	Role-Based Access Control
REST	Representational State Transfer
RPC	Remote Procedure Call
S-ESD	Seasonal Extreme Studentized Deviate
S-H-ESD	Seasonal Hybrid Extreme Studentized Deviate

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização

"*A Internet é, em tese, a mais poderosa ferramenta hoje disponível para reforçar a eficácia operacional [das organizações]*" (PORTER, 2001). Apesar desta afirmação ter sido cunhada há mais de 23 anos, é fácil perceber seu poder atualmente face às inúmeras facilidades (e riscos) que ela proporciona diariamente a todos nós, principalmente pelo surgimento das chamadas "plataformas digitais".

Conforme (HEIN et al., 2019), tais plataformas, aliadas a um ecossistema de atores sociais, continuam a mudar indústrias inteiras como serviços de acomodação (Airbnb), transporte (Uber), mídias sociais (Facebook) e bancários (Mobile Banking), apenas para citar alguns exemplos, influenciando diretamente nos mecanismos de criação de valor aos seus usuários. De fato, a tecnologia é uma poderosa ferramenta no aprimoramento da experiência do usuário e, consequentemente, na busca da vantagem competitiva por empresas concorrentes.

No contexto de Mobile Banking, a capacidade de se realizar transações bancárias com o uso de smartphones facilitou muito a vida das pessoas. Segundo (KOTLER; KELLER; CHERNEV, 2024), alguns clientes raramente pisam no *lobby* de um banco (ainda que este possua presença física). Com o advento do Pix, criado em novembro de 2020 para transferências instantâneas de valores, o número de transações usando essa modalidade vem aumentando. Conforme (MÁXIMO, 2024), foram registradas cerca de 224 milhões de transações Pix em um único dia em julho de 2024, demonstrando a popularidade do sistema.

Isso torna a segurança um aspecto fundamental, tanto para o cliente quanto para a instituição bancária, uma vez que tais transações estão sendo realizadas a partir de um dispositivo (*smartphone*) que pode estar comprometido com algum *malware* onde dados sigilosos podem estar sendo capturados para uso indevido por terceiros ou mesmo o próprio aparelho pode ter sido roubado. Segundo (BOLZANI, 2023), dados da Kaspersky, uma empresa voltada à segurança digital, mostram que já existiam em 2022 cerca de 200 mil *malwares* bancários. Como resultado do uso

indevido mencionado anteriormente, os criminosos acabam realizando transações Pix, a partir das contas bancárias dos reais proprietários, gerando movimentações atípicas em um curto período, desviando-se do padrão de uso normal do seu legítimo usuário. Esse desvio também pode ser chamado de anomalia. Somado a isso, conforme (BANCO CENTRAL DO BRASIL, 2024), as instituições financeiras participantes do sistema devem adotar medidas que mitiguem o risco de fraudes (identificadas como anomalias, no contexto desta monografia).

Detecção de anomalias não é um assunto novo e vem sendo objeto de pesquisa e desenvolvimento há vários anos, como se pode observar em (CARMINATI et al., 2015), onde os autores apresentam um sistema de suporte à decisão voltado exclusivamente à análise e investigação de anomalias em transações bancárias.

Em (SANOBER et al., 2021), os autores destacam a dificuldade na prevenção de transações anômalas, mas apontam para alguns algoritmos para a sua detecção como Floresta Aleatória, Regressão Logística, KNN (*K-nearest neighbor*) e SVM (*Support Vector Machine*).

1.2 Objetivo

O principal objetivo deste trabalho é construir um software voltado à detecção de anomalias, na forma de desvio comportamental em transações Pix via Mobile Banking, com base no histórico de requisições realizadas por seus usuários e usando uma abordagem de engenharia de dados e Big Data.

Espera-se obter um protótipo mínimo, na forma de um software funcional, capaz de detectar anomalias baseado no algoritmo S-H-ESD (*Seasonal Hybrid Extreme Studentized Deviate*), uma abordagem de aprendizado estatístico não supervisionado (HOCHENBAUM; VALLIS; KEJARIWAL, 2017), aliado a técnicas de Big Data voltadas à escalabilidade e desempenho.

Como objetivo secundário, espera-se dar visibilidade a estas anomalias através de notificações via e-mail e disponibilização de *dashboards* (representações gráficas das transações e anomalias detectadas, acessíveis via navegadores web), servindo como mecanismo de monitoração essencial para a tomada de decisões pelos

gestores da instituição bancária, segundo os princípios da Indústria 4.0 (SCHUH et al., 2020).

1.3 Justificativa

Ao longo do processo de revisão da literatura, foram avaliados alguns métodos de detecção de anomalias conforme estudos de seus respectivos autores e destacados a seguir.

Ronao e Cho (RONAO; CHO, 2016) destacam a necessidade de mecanismos de segurança para lidar tanto com ameaças externas (acessos via Internet, por exemplo) quanto internas (usuários internos mal-intencionados) no tocante à proteção de dados. Também citam que o modelo RBAC (Role-Based Access Control), padronizado pelo NIST (uma organização governamental dos Estados Unidos que, dentre outras competências, destina-se ao desenvolvimento e uso de padrões), não é suficiente para identificar anomalias em ações de usuários executadas dentro do seu papel, uma vez que tais ações seriam interpretadas como legítimas, não considerando o histórico de uso daquele usuário. Esse comportamento histórico do usuário, incluindo sazonalidades, representa importante fonte de dados para se traçar um padrão de uso como, por exemplo, transações Pix via Mobile Banking.

Hochenbaum, Vallis e Kejariwal (HOCHENBAUM; VALLIS; KEJARIWAL, 2017), engenheiros do antigo Twitter (hoje X), destacam a importância do desempenho e disponibilidade dos serviços disponibilizados pelas plataformas digitais como fatores de retenção de usuários. No contexto de Big Data, o aumento repentino no volume de mensagens, no caso do Twitter, classificadas como anomalias sazonais, pode impactar serviços e causar perdas de capital, além de reputação a longo prazo para a plataforma. Os autores utilizaram dois métodos para detecção de anomalias em séries temporais: S-ESD (Seasonal Extreme Studentized Deviate) e S-H-ESD (Seasonal Hybrid Extreme Studentized Deviate). Este último, apesar de consumir mais recursos computacionais, é recomendado pelos autores onde espera-se um maior volume de anomalias.

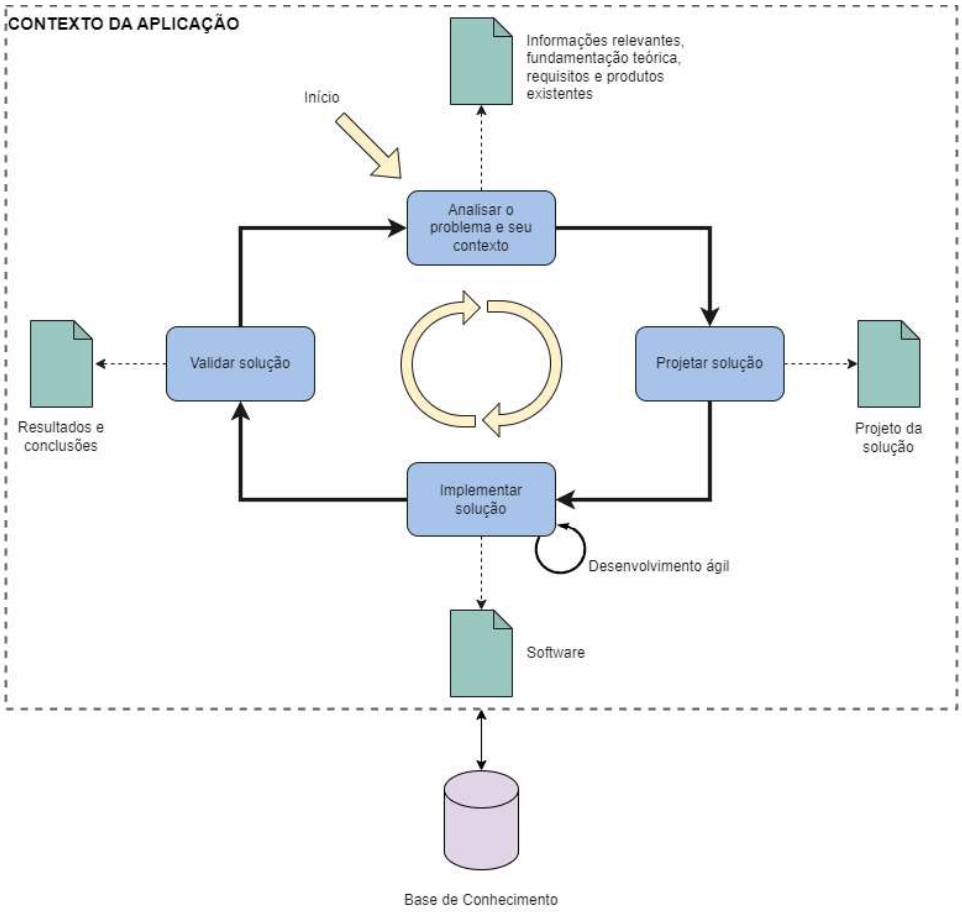
O software, produto desta monografia, utiliza como base o trabalho de (HOCHENBAUM; VALLIS; KEJARIWAL, 2017), dado o desempenho apresentado pelos algoritmos S-ESD e S-H-ESD, considerado como um ponto de partida para um protótipo funcional.

1.4 Metodologia

Este trabalho utilizou como metodologia o Design Science Research (DSR), adaptado de (WIERINGA, 2014), conforme ilustrado na Figura 1, devido às suas características de pesquisa e projeto de uma solução para um problema específico.

Deve-se observar a interação contínua e bidirecional de todas as etapas da metodologia com uma base de conhecimento, representada por literatura científica, projetos, produtos e melhores práticas.

Figura 1 – Método proposto para a condução do design science research (DSR)



Fonte: adaptado a partir de (WIERINGA, 2014).

A Tabela 1 detalha as etapas deste método, aplicadas a esta monografia, assim como as seções neste documento onde cada uma é abordada.

Tabela 1 – Etapas do método proposto

Etapa	Aplicação neste trabalho	Resultado	Seções
Analisar o problema e seu contexto	Realização de transações Pix, via Mobile Banking, geradas após a captura de dados ou roubo do smartphone do usuário.	Vulnerabilidades na autorização do Pix, fundamentação de anomalias e requisitos do Banco Central e da instituição bancária.	1 e 2
Projetar solução	Refinamento dos requisitos relacionados à detecção de anomalias e projeto arquitetural usando o NBDRA.	Arquitetura baseada no AWS e diagramas UML com as visões estruturais e comportamentais.	3.1 a 3.3
Implementar solução	Construção do software de detecção de anomalias conforme a plataforma tecnológica baseada no AWS e algoritmo S-H-ESD implementado em C++.	Software de detecção de anomalias em transações Pix em seu estado funcional e protótipo de	3.4

		tela para análise gráfica de dados.	
Validar solução	Provisionamento de infraestrutura, carga de base histórica, execução de requisições simuladas para teste de detecção e análise das métricas de detecção e consumo de recursos computacionais.	Obtenção do desempenho de detecção por F-Measure e avaliação de consumo de CPU e memória, assim como do tempo de resposta do software.	3.5 e 3.6

1.5 Organização

Este trabalho está organizado conforme os seguintes capítulos:

- Capítulo 1 – o trabalho é contextualizado em relação a um ambiente de negócio e problema específicos, são estabelecidos os seus objetivos, apontados relacionamentos com trabalhos de outros autores relevantes ao tema e definida a metodologia adotada para a pesquisa e desenvolvimento de um protótipo funcional na forma de software.
- Capítulo 2 – são abordados os principais conceitos relacionados ao tema deste trabalho, como estilos arquiteturais, Big Data, segurança e algoritmos para detecção de anomalias.
- Capítulo 3 – apresenta o contexto do software, produto deste trabalho, incluindo seus stakeholders, requisitos funcionais e de qualidade, arquitetura e decisões de projeto justificadas, detalhamento das tecnologias utilizadas e resultados obtidos.
- Capítulo 4 – apresenta as conclusões e próximos passos.

Por fim, a seção de referências possui a lista de artigos, livros e websites que serviram de base em todas as etapas da metodologia adotada neste projeto.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Principais Conceitos

Desde sua concepção em 1989 por Tim Berners-Lee, e sua implementação em 1990-91 (FIELDING et al., 2017), a Web democratizou o acesso à informação, começando pelos meios acadêmicos e alcançando todos os setores da sociedade desde então. Particularmente, o protocolo HTTP (Hypertext Transfer Protocol) serviu de base não apenas para a transferência de documentos em formato HTML (Hypertext Markup Language), principal uso da Web em seus primórdios, mas de uma grande variedade de conteúdo, servindo não apenas a pessoas usando seus navegadores para consultarem receitas culinárias, por exemplo, mas para troca de dados entre sistemas, dentre inúmeras outras aplicações.

Por outro lado, à medida que negócios aumentavam em complexidade, mais exigentes se tornavam em relação ao poder computacional para atenderem aos seus requisitos. Padrões e estilos arquiteturais de software tiveram de amadurecer para fazer frente à crescente demanda, tanto no aspecto funcional quanto de qualidade, assim como ao crescente volume de dados gerados.

Somado ao alto volume mencionado acima, a velocidade e a variedade dos dados caracterizam o que se chama de Big Data, uma área da Ciência da Computação dedicada a tratar, analisar e transformar em conhecimento tais dados. Para isto, técnicas de mineração de dados (HAN; PEI; TONG, 2023) são aplicadas usando-se diversos algoritmos especializados.

Um tema de fundamental importância que pode se beneficiar de alguns destes algoritmos é a segurança da informação, um dos aspectos de qualidade mencionados anteriormente. Nem todo dado pode ser acessado publicamente, sendo restrito somente a usuários devidamente credenciados e autorizados a este fim. Além disso, as ações tomadas por cada usuário, ao longo do tempo, podem descrever seu comportamento por meio de algoritmos especializados em reconhecimento de padrões (NEGRI, 2021), uma área do aprendizado de máquina (KELLEHER, 2020), como o uso de redes neurais (HAYKIN, 2000).

O correto entendimento do comportamento individualizado de cada usuário pode ajudar na identificação de possíveis desvios de conduta refletidos por ações que, embora autorizadas, caracterizam-se como anômalas, ou seja, algo que determinado usuário não costumaria fazer.

A capacidade em detectar tais anomalias pode ser muito útil no processo de autorização de acesso, não se limitando apenas aos direitos concedidos àquele usuário.

2.2 Revisão Bibliográfica

Detecção de anomalia é um tema abordado em diversas publicações, como livros e artigos, e em contextos variados, como acessos a bancos de dados, APIs e sistemas operacionais. Apesar deste trabalho possuir um foco bem definido sobre detecção de anomalias em APIs REST aplicadas a transações Pix, tais publicações, embora em áreas diversas, corroboram em aspectos comuns, sejam diretamente no tema, sejam abordando alguma característica similar.

Os trabalhos mencionados a seguir, classificados conforme suas principais áreas de pesquisa, foram escolhidos exatamente por tratarem de assuntos similares e que, de alguma forma, agregam ao tema.

2.2.1 APIs RESTful

O livro de Webber e Parastatidis (WEBBER; PARASTATIDIS, 2010), apoiado sobre a tese de doutorado de Roy Fielding (FIELDING, 2000), detalha a construção de sistemas distribuídos baseados na arquitetura (bem-sucedida) da web.

Embora originalmente criada para facilitar o acesso e compartilhamento de documentos sobre o protocolo HTTP, a Web acabou estabelecendo os princípios arquiteturais usados atualmente na construção de APIs, uma forma de exposição de serviços disponibilizados por softwares em um ambiente de rede (TAYLOR; MEDVIDOVIC; DASHOFY, 2010), dada a sua simplicidade e robustez.

Sendo o bloco de construção fundamental da Web, um recurso é qualquer coisa exposta de forma pública ou privada que possa ser submetida a alguma ação específica e pré-determinada. Tal ação, natural do protocolo HTTP, é conhecida como verbo (ou método, conforme consta em sua especificação) que, direcionada a um recurso, gera uma resposta. Exemplos de ações são: GET (obter um recurso existente), POST (incluir um novo recurso), PUT (alterar um recurso existente), PATCH (alterar parcialmente um recurso existente) e DELETE (excluir um recurso existente). Para que qualquer destas ações possa ser direcionada a um recurso específico, um identificador único é associado a ele.

REST, conforme definido por Roy Fielding, "*é um estilo híbrido derivado de vários estilos arquiteturais baseados em rede [comumente apoiado sobre o protocolo HTTP] e combinado a restrições adicionais que define uma interface uniforme*". Diz-se que uma API é RESTful quando esta adota tais princípios.

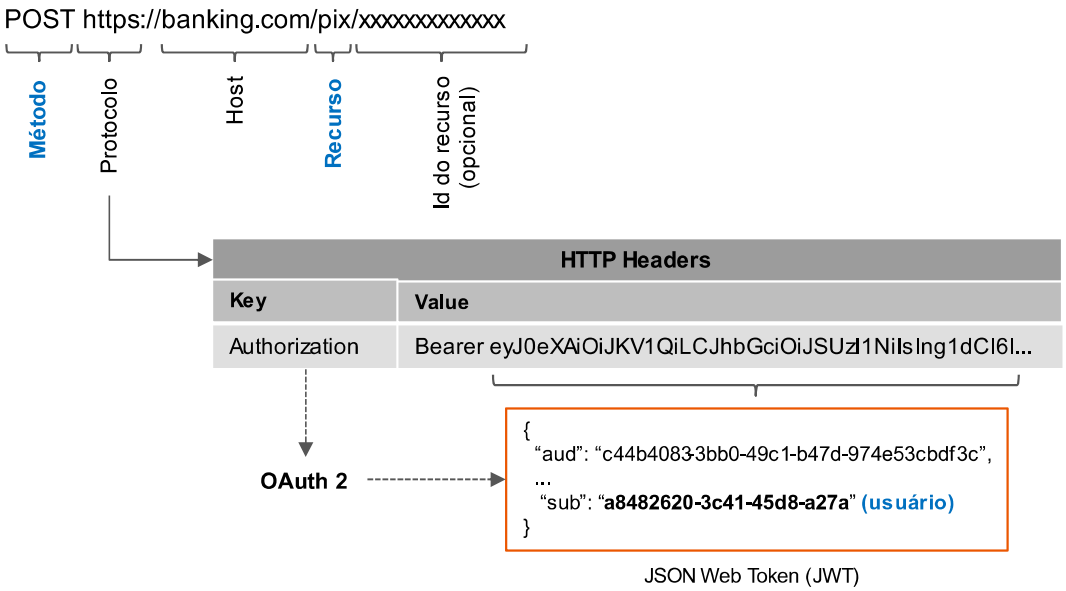
O trabalho de (FIELDING, 2000) destaca a importância das decisões de projeto, na escolha de seus componentes e como eles serão integrados, sob a ótica de duas disciplinas: software e rede. Tendo como foco o alcance mundial dos documentos hipermídia da Web, que é mais que uma simples questão de dispersão geográfica, Fielding analisa a interconexão da Internet e suas fronteiras organizacionais.

A partir daí, o autor introduz REST como um estilo arquitetural para sistemas hipermídia distribuídos, considerando fatores como escalabilidade, independência de implantação de componentes, latência de comunicação e segurança.

A dissertação de Fielding ainda serve de base para qualquer trabalho que envolva arquitetura de sistemas distribuídos baseados em protocolo HTTP, destacando as interações entre clientes (aqueles que efetuam requisições) e servidores (serviços que respondem às requisições a partir de suas APIs).

A Figura 2 ilustra um exemplo simples e fictício de uma requisição REST realizada no contexto de uma plataforma digital (Mobile Banking) a partir de um dispositivo (cliente) e direcionada a um serviço de transação bancária Pix (servidor).

Figura 2 – Anatomia de uma requisição REST



Fonte: elaborado pelo autor

Os elementos que compõem a requisição acima estão descritos na Tabela 2:

Tabela 2 – Elementos de uma requisição REST

Elemento	Descrição	Significado
Método	A ação sendo requisitada	O que fazer.
Protocolo	O protocolo de comunicação (HTTP seguro neste exemplo)	Usando qual canal de comunicação.
Host	O servidor ao qual a requisição está direcionada	Onde.
Recurso	O recurso, no domínio da aplicação, onde o método será aplicado	Qual recurso.
Id do recurso	Identificação do recurso (opcional, dependente do contexto)	Qual item específico.
JWT	Token gerado para o usuário no momento da autenticação no sistema	Quem está solicitando.

APIs RESTful, já no lado do servidor, e por adotarem este mesmo padrão, estabelecem contratos de interface que identificam e direcionam aos respectivos serviços da aplicação a requisição do cliente, cujo resultado do processamento é retornado respeitando-se este mesmo contrato.

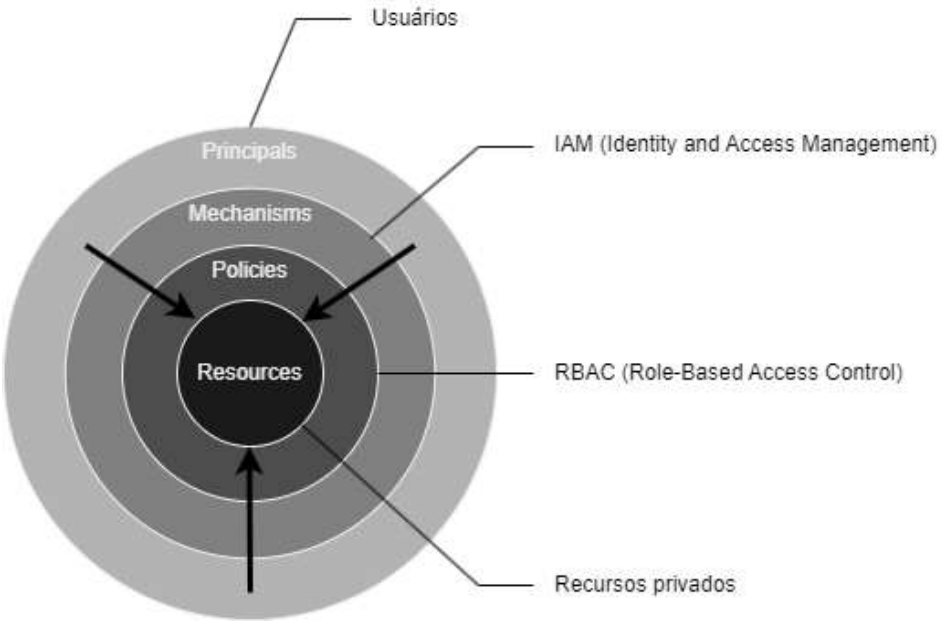
Aplicativos de Mobile Banking modernos usam APIs REST como meio de integração entre os diversos elementos do software, como a comunicação via Internet entre o

aplicativo sendo executado no smartphone do usuário e os serviços disponibilizados na infraestrutura do banco, como transações Pix. Dessa forma, tais APIs tornam-se a principal porta de entrada para acesso aos serviços da instituição, necessitando de atenção especial no que se refere à segurança.

2.2.2 Segurança e Controle de Acesso

Conforme (ROZANSKI; WOODS, 2012), são vários os fatores que levam à necessidade de se elaborar uma estratégia para a segurança de sistemas de informações, como a tendência crescente de sistemas distribuídos e o uso de redes públicas (Internet). Por segurança, os autores a definem como o conjunto de processos e tecnologias que permitam aos proprietários de recursos de um determinado sistema controlar quem poderá acessar qual recurso. Assim sendo, recurso é um conceito fundamental para a segurança do sistema. A Figura 3 abaixo ilustra este conceito.

Figura 3 – Elementos de segurança



Fonte: adaptado a partir de (ROZANSKI; WOODS, 2012).

De fato, arquiteturas de referência incluem elementos de segurança em sua composição, conforme ilustrado na Figura 8 (página 31), extraída de (“NIST Big Data Interoperability Framework”, 2019). Pode-se observar que todos os componentes da arquitetura estão dentro do que foi chamado de “Security and Privacy Fabric”, o que denota que tais componentes, sejam de processamento, comunicação ou armazenamento, estejam sob políticas e mecanismos de segurança bem definidos.

Segundo (WIEGERS; BEATTY, 2013), atributos de qualidade de software, como segurança, possuem interrelações que podem afetar tanto positiva quanto negativamente outros atributos, conforme pode-se observar na Figura 4.

Figura 4 – Relação entre atributos de qualidade

	Availability	Efficiency	Installability	Integrity	Interoperability	Modifiability	Performance	Portability	Reliability	Reusability	Robustness	Safety	Scalability	Security	Usability	Verifiability
Availability								+		+						
Efficiency	+				–	–	+	–			–		+		–	
Installability	+								+					+		
Integrity			–		–		–			–		+		+	–	–
Interoperability	+		–	–			–	+	+		+	–		–		
Modifiability	+		–				–		+	+			+			+
Performance		+			–	–		–			–		–		–	
Portability		–			+	–	–			+				–	–	+
Reliability	+	–		+		+	–				+	+		+	+	+
Reusability		–		–	+	+	–	+						–		+
Robustness	+	–	+	+	+		–		+			+	+	+	+	
Safety		–		+	+		–			+				+	–	–
Scalability	+	+		+			+	+	+		+					
Security	+			+	+		–	–	+		+	+			–	–
Usability		–	+				–	–	+		+	+				–
Verifiability	+		+	+		+			+	+	+	+		+	+	

Fonte: (WIEGERS; BEATTY, 2013)

Segurança, em particular, pode ter um impacto negativo no desempenho (*performance*, na figura acima). Desta forma, as decisões do projeto do sistema, representadas por sua arquitetura, deverão considerar tais impactos nas suas escolhas.

Já do ponto de vista dos elementos de processamento, as APIs desempenham um papel importante como conectores que expõem serviços em forma de contratos bem definidos (TAYLOR; MEDVIDOVIC; DASHOFY, 2010) que, em última instância, vão realizar ações sobre recursos mantidos pelo sistema. Esta exposição leva à necessidade de se criar mecanismos de autorização pelos motivos já mencionados por (ROZANSKI; WOODS, 2012).

No caso específico de APIs RESTful, os mecanismos mais comuns utilizados atualmente, segundo (QAZI, 2023), são HTTP Basic Authentication, API keys, JWT (JSON Web Token) e OAuth (Open Authorization), estando os dois últimos relacionados, sendo JWT um dos elementos comumente gerados durante um processo de autenticação configurado para Open Authorization. A Figura 2 (página 21) ilustra a anatomia de uma requisição REST com a presença de um JWT identificando o usuário solicitante.

Os elementos que constituem tal requisição, conforme descritos na Tabela 2 (página 21), permitem estabelecer a identidade do usuário, a ação solicitada e o recurso cuja ação afetará. A partir daí, uma solução de IAM (SHARMA; DHOTE; POTEY, 2016) pode proceder à autorização, ou não, do acesso solicitado

O mecanismo de autorização utilizado para tal, conforme (FERRAIOLO; KUHN, 1992) pode se basear em diversas abordagens, como MAC (Mandatory Access Control), DAC (Discretionary Access Control) e RBAC (Role-Based Access Control). Segundo Ferraiolo e Kuhn, e corroborado pela pesquisa posterior do NIST (SANDHU; FERRAIOLO; KUHN, 2000), RBAC demonstra ser mais apropriado como forma de definição de políticas de acesso e administração centralizada de IAM, convergindo com o conceito de segurança apresentado na Figura 3 (página 22), por (ROZANSKI; WOODS, 2012).

Contudo, os mecanismos de autorização mencionados acima (MAC, DAC e RBAC) se restringem à concessão de acesso a partir de regras pré-definidas, como papéis (roles) associados ao usuário participante da requisição.

Em cenários de roubo de identidade, ou de usuário com privilégios mas com más intenções, as ações tomadas podem fugir a um padrão de uso realizado historicamente, podendo mesmo serem classificadas como anomalias, mas serão autorizadas baseadas estritamente nas regras de acesso citadas.

2.2.3 Detecção de Anomalias

Segundo (HAN; PEI; TONG, 2023), detecção de outlier, também conhecido como detecção de anomalia, é o processo de localizar dados cujo comportamento difere

do esperado, tendo como desafio a interpretabilidade do que levou a alguns dados serem classificados como anomalias e outros não. Este é um requisito importante a ser considerado, e métodos estatísticos podem justificar o grau em que um determinado dado pode ser classificado como uma anomalia baseado na probabilidade de este ter sido gerado pelo mesmo mecanismo (no contexto deste trabalho, o usuário) que gerou a maioria dos dados ou não.

Este trabalho utiliza a abordagem baseada em S-H-ESD (HOCHENBAUM; VALLIS; KEJARIWAL, 2017), um método não supervisionado de aprendizado estatístico para a detecção de anomalias, conforme citado na seção 1.3 (página 14) e, a seguir, são apresentados alguns dos conceitos utilizados em sua implementação.

2.2.3.1 Teste de hipóteses

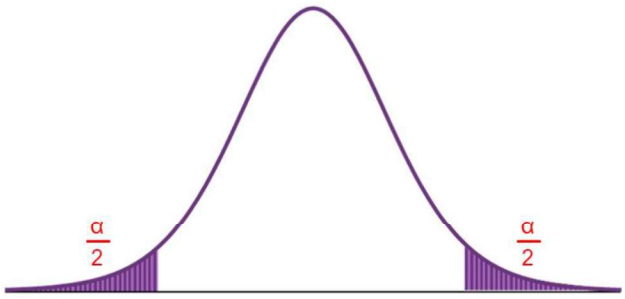
Conforme (TRIOLA, 2017), em estatística, uma hipótese é uma afirmativa sobre uma propriedade de uma população (no contexto deste trabalho, o histórico de ações de um determinado usuário); e um teste de hipótese, ou teste de significância, é um procedimento para o teste desta afirmativa. Tal teste considera uma hipótese nula (a afirmativa a ser testada) representada por H_0, e uma hipótese alternativa, representada por H_1, que difere de alguma forma da hipótese nula.

Do ponto de vista de detecção de anomalia, as hipóteses seriam:

H_0: não há outliers (anomalias) no conjunto de dados
H_1: há ao menos um outlier no conjunto de dados

A Figura 5 ilustra uma distribuição normal de dados, sendo α a probabilidade de cometermos o erro de rejeitar a hipótese nula quando ela é verdadeira (TRIOLA, 2017).

Figura 5 – Distribuição de probabilidade



Fonte: elaborado pelo autor

A área no gráfico da Figura 5, representada por $\frac{\propto}{2}$, identifica a região onde a hipótese nula é refutada e, consequentemente, validada a hipótese alternativa da presença de outliers.

Desta forma, a determinação do nível de significância definirá a sensibilidade na detecção de anomalias.

2.2.3.2 O caso Twitter (atual X)

O trabalho de (HOCHENBAUM; VALLIS; KEJARIWAL, 2017) é uma aplicação do S-H-ESD no Twitter para detectar anomalias baseadas nas seguintes métricas:

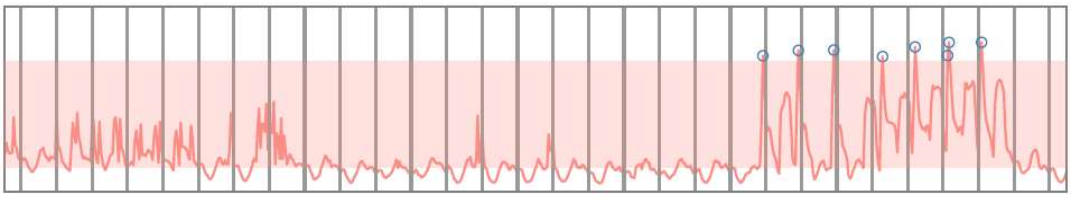
- Sistema
  - Uso de CPU
  - Uso de Heap
  - Tempo em GC (Garbage Collection)
  - Escrita em disco
- Aplicação
  - Taxa de requisições
  - Latência
- Core Drivers
  - Tweets por minuto
  - Retweets por minuto
  - Fotos únicas por minuto

O S-H-ESD tem como base o algoritmo S-ESD, que usa uma versão modificada de STL (Seasonal-Trend decomposition using LOESS – LOcal regrESSion) (OUYANG; RAVIER; JABLOUN, 2021) para lidar com sazonalidades nas séries temporais, o que torna mais robusta a detecção de anomalias. Esta característica é importante para este trabalho pois os usuários podem apresentar comportamentos sazonais.

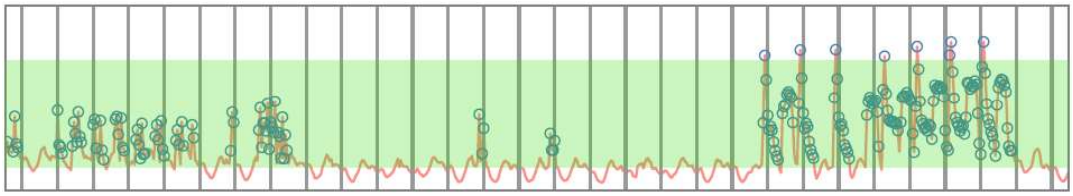
Além disso, substitui a média e desvio padrão usados no ESD pela mediana e MAD (Median Absolute Deviation) que, segundo os autores, aumenta a capacidade de detecção de anomalias pelo algoritmo, já que não são afetados pelos outliers.

A Figura 6, extraída de (HOCHENBAUM; VALLIS; KEJARIWAL, 2017), ilustra o desempenho na detecção de anomalias usando-se S-ESD (a) e S-H-ESD (b) em uma mesma série temporal. Os círculos representam anomalias detectadas.

Figura 6 – Desempenho na detecção de anomalias



(a) Anomalias detectadas via S-ESD: 1,11% ($\propto = 0{,}05$)



(b) Anomalias detectadas via S-H-ESD: 29,68% ($\propto = 0{,}05$)

Fonte: (HOCHENBAUM; VALLIS; KEJARIWAL, 2017)

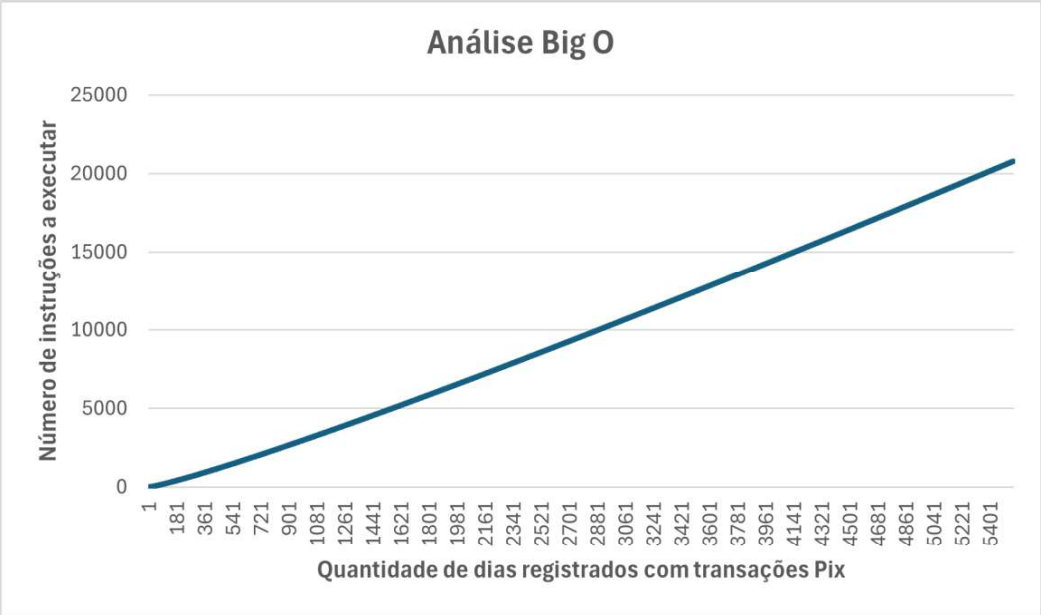
Como se pode observar nos gráficos, e conforme os próprios autores citam, S-ESD (Figura 6-a), que se baseia na média e desvio padrão, não detectou uma série de anomalias presentes na região destacada pelo retângulo vermelho, ao contrário do S-H-ESD (Figura 6-b), baseado em mediana e desvio absoluto mediano, agora destacado em verde.

Segundo os autores, a média pode ser distorcida por uma simples anomalia, com a distorção aumentando conforme $x_t \rightarrow \pm\infty$. Por outro lado, MAD é mais robusto em relação às anomalias, não sofrendo tais distorções.

2.2.3.3 Análise assintótica

A análise assintótica do S-H-ESD mostra que o tempo de processamento do algoritmo é aproximadamente linear em relação ao tamanho do conjunto de dados, tendo notação $O\ n.\log(n)$. Uma amostra dessa função pode ser vista na Figura 7:

Figura 7 – Análise assintótica (Big O)



Fonte: elaborado pelo autor

Do ponto de vista deste projeto, mantendo-se uma janela de 2 anos de dados históricos de transações Pix de cada usuário para a detecção de anomalias, prevê-se um desempenho aproximadamente constante com o uso deste algoritmo.

2.3 Big Data

Conforme (KRISHNAN, 2013), Big Data pode ser definido como "*alto volume de dados com variados graus de complexidade e gerados a diferentes velocidades que não podem ser processados usando-se tecnologias tradicionais ou soluções comerciais de prateleira*".

De fato, segundo (DAMA-DMBOK, 2017), até recentemente a análise de grandes conjuntos de dados vinha sendo limitada pela tecnologia. Novas tecnologias voltadas ao suporte de Big Data, como o ecossistema Hadoop e bancos de dados distribuídos, criaram oportunidades para a análise de dados massivos, não apenas para descreverem fatos ocorridos (efeito espelho retrovisor), mas para permitirem a predição de comportamentos (efeito para-brisa).

Alguns conceitos de Big Data, usados no contexto deste trabalho, são abordados nas seções seguintes.

2.3.1 Business Intelligence (BI)

Conforme (DAMA-DMBOK, 2017), BI refere-se a um tipo de análise de dados voltada à obtenção de conhecimento, essencial para a tomada de decisões em uma organização. Além disso, constitui o conjunto de tecnologias que suportam estas análises.

No contexto deste trabalho, o resultado da detecção de anomalias gera insumos que podem ser usados para a identificação de padrões de uso de recursos (transações Pix) pelos usuários, gerando conhecimento e servindo de base para a tomada de decisões.

2.3.2 Data Lake

Um Data Lake, conforme (DAMA-DMBOK, 2017), é um ambiente onde dados de vários tipos (texto, áudio, imagem e estruturas de dados diversas) podem ser armazenados para processamento com múltiplos propósitos.

Neste trabalho, este ambiente é utilizado pelo software de detecção de anomalias como área de armazenamento de estruturas JSON, sendo classificado como um estágio para o processamento posterior pelo pipeline de dados ilustrado na Figura 17 (página 49).

2.3.3 Data Warehouse (DW)

Segundo o (DAMA-DMBOK, 2017), um DW habilita as organizações a integrarem dados de diversas origens em um modelo de dados comum objetivando criar valor organizacional e se tornar fonte de informações confiáveis para a tomada de decisões. Baseia-se em dois componentes principais:

- Banco de dados integrado para o suporte à decisão
  Este trabalho usa um banco de dados colunar baseado em um modelo dimensional (Figura 16 na página 48) para análise de dados, permitindo a seleção de transações e anomalias (fatos) sob perspectivas diferentes, como recurso, usuário, ação e tempo (dimensões);

- Software para coleta, limpeza, transformação e armazenamento de dados
  Os resultados da detecção de anomalias, armazenados no data lake mencionado na seção 2.3.2, são processados por um serviço de ETL (Extract, Transform, Load) que compõe um pipeline de dados com o objetivo de adequar estes dados ao modelo dimensional para análise. Esta abordagem utiliza alguns conceitos importantes:

  - Processamento distribuído – disponibilização de vários "nós" escaláveis e de baixo custo ao invés de um único nó centralizado e de alto custo;
  - Redução de dados – um modelo de programação funcional baseado no mapeamento de atributos e sua redução (MapReduce) através de alguma técnica de agregação, como soma ou média (ELMASRI; NAVATHE, 2019).

2.3.4 Arquitetura de Referência

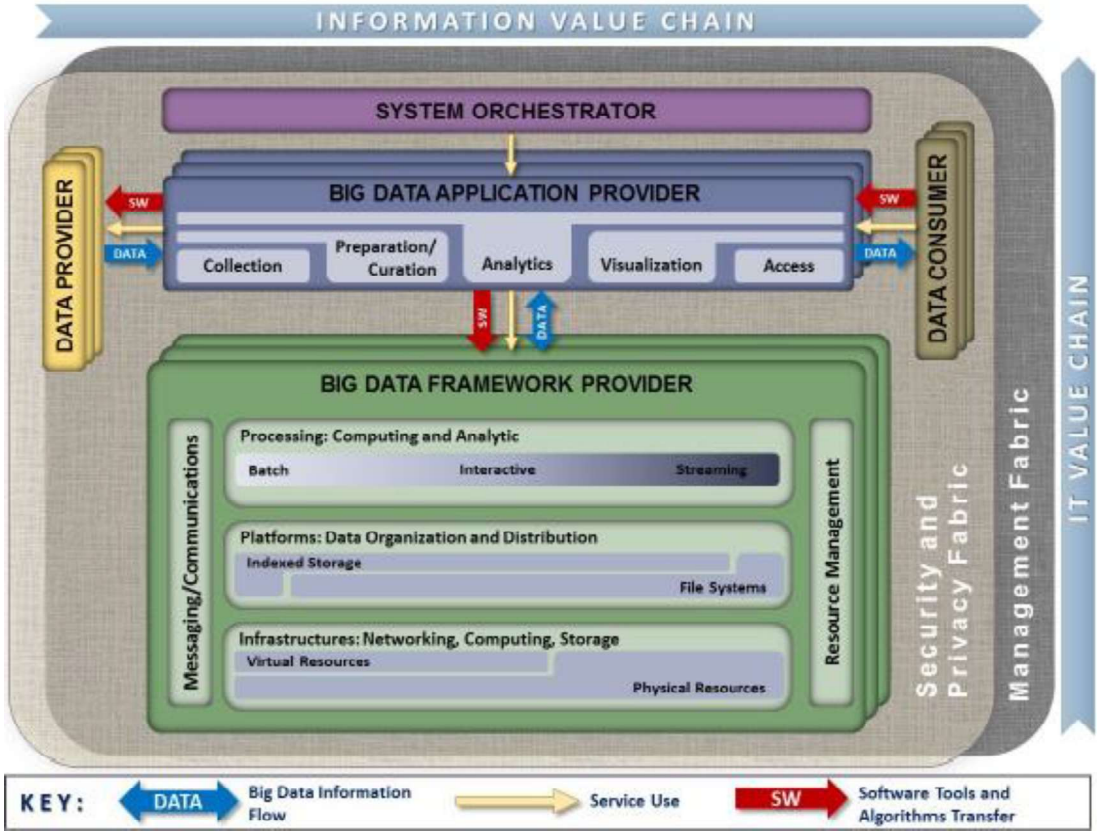
Segundo (CERVANTES; KAZMAN, 2016), uma arquitetura de referência é um "*blueprint*" que fornece uma estrutura lógica geral para um determinado tipo de aplicação. É um modelo de referência apoiado sobre um ou mais padrões

arquiteturais e tem se provado válido em contextos tanto de negócios quanto técnicos.

De fato, conforme (TAYLOR; MEDVIDOVIC; DASHOFY, 2010) sugerem, ao invés de se desenvolver novas arquiteturas para cada novo problema em um determinado domínio, pode-se derivar uma solução a partir de arquiteturas de referência.

No contexto de Big Data, o NIST possui em seu catálogo uma arquitetura de referência para este domínio, conforme ilustrada na Figura 8.

Figura 8 – NIST Big Data Reference Architecture (NBDRA)



Fonte: (“NIST Big Data Interoperability Framework”, 2019)

A Tabela 3 destaca os elementos que compõem esta arquitetura de referência e sua correlação com este trabalho.

Tabela 3 – Elementos da arquitetura de referência NBDRA

Elemento	Descrição	Correlação
Data Provider	Os provedores que disponibilizam dados de interesse em sua forma crua	A solução de IAM da organização
Data Consumer	Os consumidores, internos ou externos, dos dados processados pela solução, segundo suas áreas de interesse, e com valor agregado	A solução de IAM da organização, software de notificação e ferramentas de análise de dados
Processing: Computing and Analytic	Software de processamento em batch, interativo ou de streaming de dados	Software detector de anomalias e ETL
Platforms: Data Organization and Distribution	Sistemas de gerenciamento e organização de dados, de sistema de arquivos a mecanismos de distribuição e indexação	Data Lake (S3) e Data Warehouse (Redshift)
Infrastructures: Networking, Computing, Storage	Gerenciamento de recursos de infraestrutura, como redes virtuais e armazenamento	Recursos gerenciados pelo AWS
Messaging/Communications	Recursos de comunicação e integração com elementos internos e externos	SQS, Kinesis Streams e SNS
Resource Management	Provisionamento de recursos conforme necessário	Recursos genreciados pelo AWS a partir de configurações

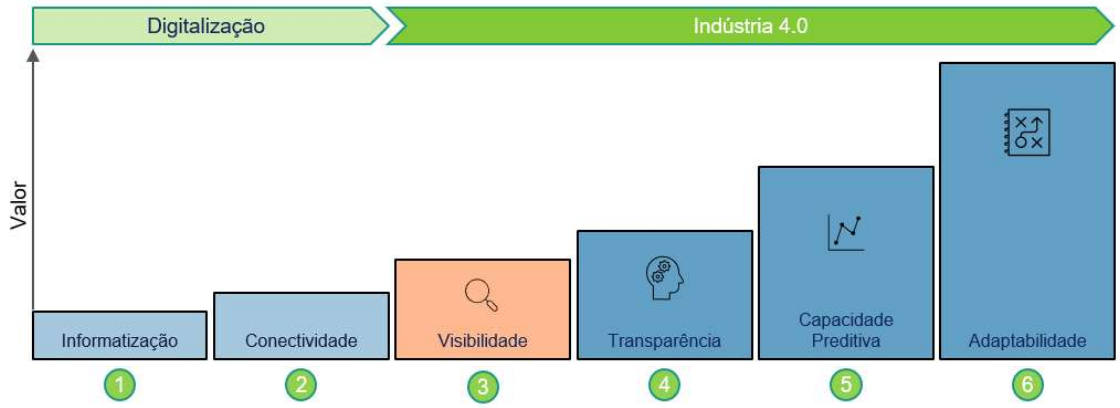
2.4 Indústria 4.0

O termo “Indústria 4.0”, conforme (CAVANILLAS; CURRY; WAHLSTER, 2016), se origina da digitalização e interligação de produtos, unidades de produção e infraestrutura de transporte, parte integrante da IoT (Internet of Things), que trazem novos desafios para simulação, planejamento, monitoração, controle de maquinário ou aplicações de dados.

A digitalização mencionada acima refere-se ao nível básico do processo de transformação digital que a maioria das empresas, engajadas em alcançar algum diferencial competitivo, vêm adotando já há alguns anos.

Segundo (SCHUH et al., 2020), níveis mais elevados de maturidade, conforme ilustrado na Figura 9, denotam que determinada organização alcançou algum estágio classificado como Indústria 4.0.

Figura 9 – Indústria 4.0



Fonte: adaptado a partir de (SCHUH et al., 2020).

Todos os estágios deste índice de maturidade estão descritos na Tabela 4, segundo os próprios autores:

Tabela 4 – Estágios do Índice de Maturidade da Indústria 4.0

Estágio	Cenário	Indústria 4.0
Informatização	Sistemas de informação em uso de forma isolada na organização para automação de atividades repetitivas.	
Conectividade	Sistemas isolados substituídos por componentes interconectados suportando processos de negócios.	
Visibilidade	Obtenção de dados de processos em tempo real favorecendo a tomada de decisões bem-informadas. É uma representação digital da empresa (*digital shadow*), servindo de base para os demais estágios da Indústria 4.0.	O que está acontecendo?
Transparência	Uso dos dados para criar informação e contextualização, gerando conhecimento dos processos para o suporte de decisões complexas rapidamente.	Por que está acontecendo?
Capacidade preditiva	Capacidade de simular diferentes cenários futuros e identificar o mais provável. Envolve projetar o *digital shadow* no futuro prevendo-se uma variedade de possibilidades e suas probabilidades.	O que vai acontecer?
Adaptabilidade	Quando a organização é capaz de usar os dados do *digital shadow* para tomada de decisões de forma automática, rápida e sem a intervenção humana.	Como uma resposta autônoma pode ser alcançada?

A capacidade de detecção de anomalias em transações Pix e a subsequente notificação automática, além da disponibilização dos resultados destas detecções para análise dimensional, servem de suporte para uma empresa alcançar o nível 3 (visibilidade), conforme destacado na Figura 9, auxiliando na tomada de decisões.

Conforme (CAVANILLAS; CURRY; WAHLSTER, 2016), Big Data é um habilitador para a Indústria 4.0, devendo ser visto como um ativo econômico de fundamental importância em iniciativas de inovação, eficiência e novas oportunidades.

3 CONTROLE DE ACESSO COM DETECÇÃO DE ANOMALIAS

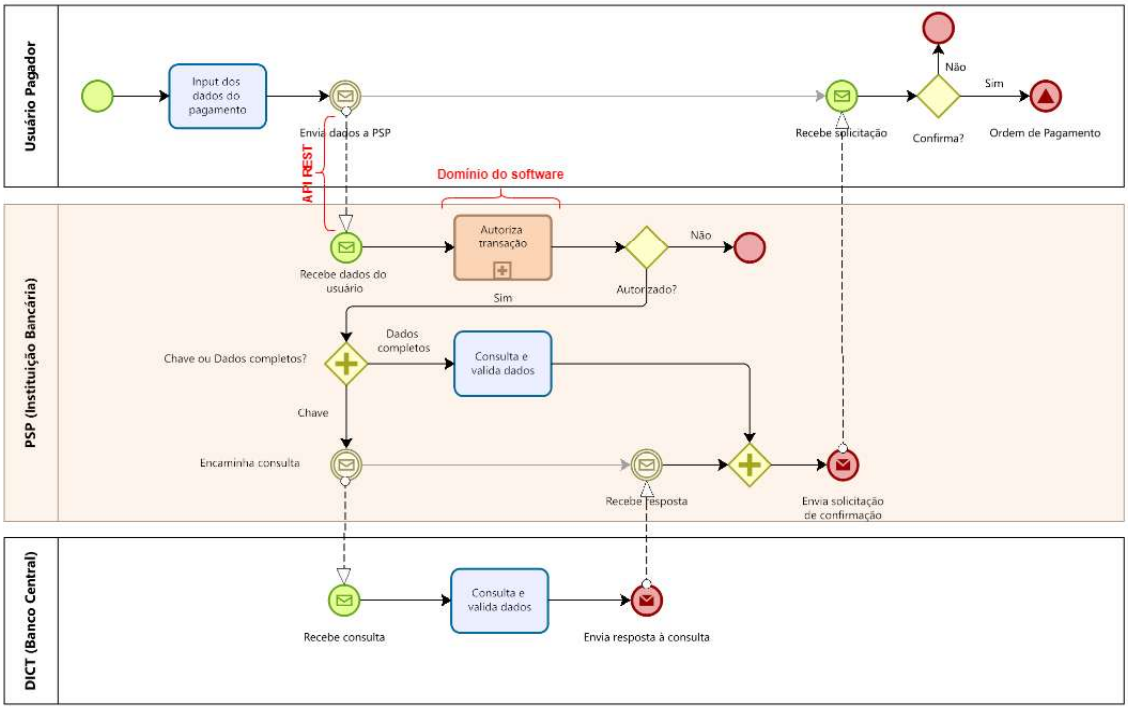
3.1 Contexto da Aplicação

As subseções a seguir são resultado da aplicação da fase “Analisar o problema e seu contexto” do DSR, conforme detalhado na seção 1.4 (página 15). Foram considerados documentos disponibilizados pelo (BANCO CENTRAL DO BRASIL, 2024), assim como informações levantadas junto a um banco digital de grande porte que opera no país.

3.1.1 Domínio

O banco digital mencionado acima possui cerca de 30 milhões de clientes e realiza até 28 milhões de transações Pix por dia, por meio de seu aplicativo de Mobile Banking. Este trabalho limita-se ao processo de pagamento por essa modalidade, conforme ilustrado na Figura 10 abaixo.

Figura 10 – Fluxo de geração de ordem de pagamento via Pix



Fonte: adaptado a partir de (BANCO CENTRAL DO BRASIL, 2024).

Conforme se pode observar na figura acima, a raia identificada como PSP (Instituição Bancária), de fundo mais escuro, apresenta as atividades relacionadas ao processo de geração de ordem de pagamento originada pelo usuário e sob a responsabilidade do banco. A atividade de nome “Autoriza Transação” representa o subprocesso responsável pela autorização da transação, que inclui aspectos como a autenticação do usuário e sua validação de acesso ao serviço. O software de detecção de anomalias passará a constituir uma parte deste subprocesso.

3.1.2 Stakeholders

A Tabela 5 abaixo lista os stakeholders identificados no contexto do subprocesso de autorização de uma transação Pix e suas respectivas áreas.

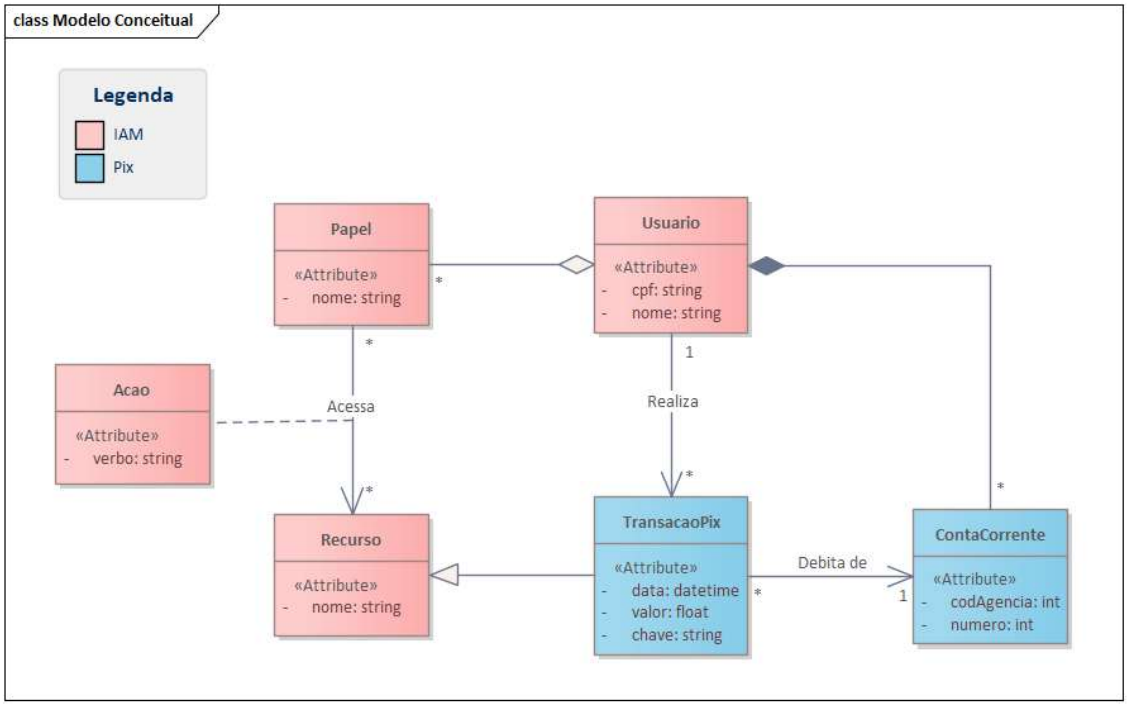
Tabela 5 – Stakeholders por área

Área	Profissionais
Processos de Negócios	Especialista de Negócio
Segurança da Informação e Dados	Analista de Segurança e Engenheiro de Dados
Serviços de TI	Arquiteto de Soluções e Engenheiro de Software

3.1.3 Modelo Conceitual

A Figura 11 representa o modelo conceitual simplificado das classes envolvidas no subprocesso “Autoriza Transação” destacando, em particular, transações Pix.

Figura 11 – Modelo Conceitual



Fonte: elaborado pelo autor

Uma transação Pix é “vista” pela solução de IAM como um recurso gerenciado por uma API específica, não incluída no modelo conceitual por se tratar de uma forma de implementação. O acesso a este recurso será concedido àqueles usuários que possuam algum papel associado a ele e para a ação desejada.

3.1.4 Requisitos

O subprocesso “Autoriza Transação”, implementado por uma solução de IAM interna do banco, tem como responsabilidade a identificação do usuário e a autorização do acesso à transação Pix utilizando uma política de acesso baseada em papéis (RBAC, seção 2.2.2 na página 22). O software de detecção de anomalias deverá fornecer a esta solução de IAM, após a correta identificação e autorização RBAC, a capacidade de verificar algum possível desvio comportamental do usuário considerando a quantidade histórica de transações Pix realizadas diariamente por ele. Caso a transação em curso seja classificada como anômala (quando a quantidade de transações Pix realizadas no dia seja classificada como uma anomalia), a solução de IAM poderá não a autorizar, encerrando todo o processo.

3.1.4.1 Requisitos Funcionais

A Tabela 6 consolida as funcionalidades do software de detecção de anomalias como parte do subprocesso de autorização de transação, conforme destacado na seção 3.1.1 (página 35).

Tabela 6 – Lista de requisitos funcionais

Requisito	Descrição	Observação
RF001	As APIs do software de detecção seguem o padrão REST com segurança baseada em OAuth	Facilidade de integração com a solução de IAM do banco
RF002	A quantidade mínima de dias de transações para formar base histórica é configurável	O histórico de transações é a base para caracterizar o comportamento dos usuários
RF003	O período padrão usado para decomposição da série histórica é de 7 dias, mas é configurável	O algoritmo S-H-ESD necessita do período para decomposição da série, cujo padrão também é de 7 dias
RF004	Outros algoritmos, além do S-H-ESD, poderão ser implementados para permitir a escolha de qual usar a partir de configuração do software	A substituição do algoritmo por configuração permite a análise de desempenho dos métodos sem que seja necessário recompilar e reimplantar o software
RF005	O software mantém base histórica de até 2 anos (configurável) das transações Pix realizadas por seus usuários	Essa base é usada como série temporal para a detecção de anomalias
RF006	Transações Pix analisadas como normais têm como código de retorno do software HTTP 200 (Ok)	"Ok" indica que nenhuma anomalia foi detectada no contexto da transação
RF007	Transações analisadas como anômalas têm como código de retorno do software HTTP 417 (Expectation Failed)	Este código, por convenção, indica anomalia detectada
RF008	Transações que não atendam ao padrão REST ou não incorporem OAuth têm como código de retorno do software HTTP 400 (Bad Request)	Este código indica transação inválida e não poderá ser processada
RF009	Um Data Warehouse deve permitir a análise do histórico de transações Pix e eventuais anomalias detectadas oferecendo, minimamente, as dimensões período e usuário	A visão multidimensional desses dados favorece a visibilidade necessária para o suporte à tomada de decisão

3.1.4.2 Requisitos de Qualidade

Por se tratar de um software que participará do processo de autorização on-line, ou seja, o usuário inicia uma transação Pix e aguarda por uma resposta, o principal requisito de qualidade, também conhecido como "requisito não funcional", é o desempenho.

A Tabela 7 descreve este e outros requisitos de qualidade considerados críticos para o software de detecção de anomalias.

Tabela 7 – Lista de requisitos de qualidade

Requisito	Descrição	Observação
RQ001	A análise para detecção de anomalia para qualquer transação Pix não poderá levar mais de 200ms	Interação síncrona com um usuário demanda uma resposta rápida, dado que a percepção de desempenho pelo usuário deve ser satisfatória, considerando que a autorização pelo IAM, por si só, já possui seu próprio custo em tempo
RQ002	É previsto momentos de pico com até 20.000 transações Pix simultâneas	Transações Pix oscilam em função de época e horário
RQ003	O software é tolerante a falhas: caso uma instância se torne indisponível, outra nova deverá tomar seu lugar automaticamente	Instabilidade de infraestrutura deve ser minimizada a fim de se garantir a disponibilidade do serviço de transação Pix aos usuários
RQ004	Garantia da privacidade e confidencialidade dos dados dos usuários, conforme legislação em vigor	Aderência à LGPD
RQ005	Anomalias detectadas são notificadas via e-mail	A notificação é parte essencial da monitoração das anomalias
RQ006	O software de detecção de anomalias é versionado	O algoritmo de detecção de anomalias deve evoluir com o tempo, sendo mantido registro dessa evolução e suas características conforme suas versões

3.1.5 Elementos Externos ao Software de Detecção de Anomalias

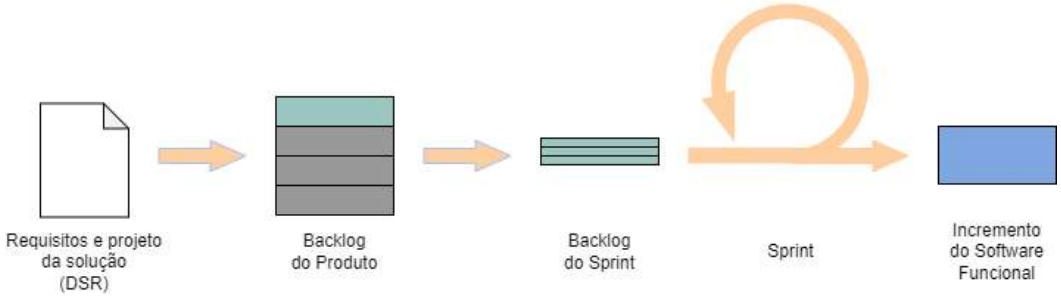
Por se tratar de um software participante do processo de autorização em transações Pix, conforme visto na seção 3.1.1 (página 35), deverá haver uma integração síncrona com a solução de IAM da instituição.

Anomalias que sejam detectadas deverão ser notificadas por e-mail, de forma assíncrona, a usuários ou grupos de usuários definidos em configuração do software de detecção.

3.2 Processo de Desenvolvimento

O processo de desenvolvimento adotado foi o SCRUM, uma metodologia ágil amplamente usada e ilustrada na Figura 12.

Figura 12 – Metodologia ágil SCRUM



Fonte: elaborado pelo autor

O projeto do software e seus requisitos (Tabela 6 e Tabela 7 nas páginas 38 e 39, respectivamente), gerados nas fases iniciais do DSR (Figura 1 na página 16), foram usados como base para o backlog do produto (software) e, por conseguinte, para a formação do backlog de cada sprint de desenvolvimento, conforme detalhado na Tabela 8. Os sprints 1 e 2 foram concluídos, formando o conjunto mínimo de requisitos para a criação de um protótipo funcional. O sprint 3 será executado posteriormente.

Tabela 8 – Backlogs de sprints

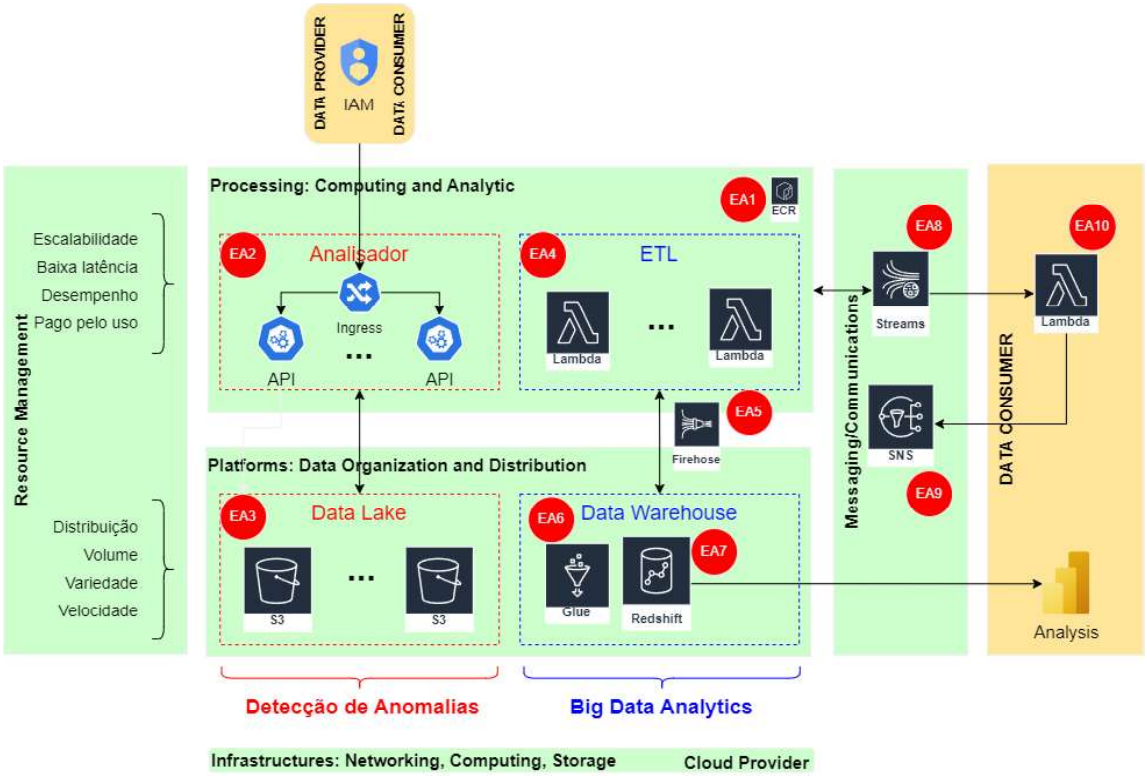
Sprint	Resumo	Requisitos
1	Implementação das classes e serviços de domínio (repositório de dados baseados em filesystem e AWS S3, serviço de detecção de anomalias baseado no algoritmo S-H-ESD).	RF002, RF003, RF004, RF005, RF006, RF007 e RQ004
	Implementação da API para atendimento a requisições REST (transações Pix)	RF001, RF008 e RQ001
	Implementação do manifesto Docker (Dockerfile) para build de imagem	RQ006
	Execução de testes locais	-
2	Implementação da API para carga de histórico de transações Pix	RF005
	Implementação do serviço de mensageria baseado no Kafka e AWS SQS	RF009 e RQ005 (parciais)
	Implementação do manifesto para implantação em clusters Kubernetes com critérios de elasticidade	RQ002 e RQ003
	Implementação do pipeline DevOps para implantação automática em cluster Kubernetes	RQ006
	Execução de testes em cluster Kubernetes	-
3	Implementar mecanismo ETL para carga de dados no Redshift via AWS Kinesis	Próximos passos
	Implementar integração com AWS SNS para envio de e-mail de notificação de anomalia detectada	Próximos passos

Pode-se observar que tanto o SCRUM quanto o DSR possuem natureza cíclica, favorecendo a execução de cada etapa de análise, projeto, implementação e validação de forma incremental e evolutiva, possibilitando entregas parciais do software de detecção de anomalias em um estado funcional. No contexto deste trabalho, foi possível realizar um teste inicial da implementação do algoritmo de detecção de anomalias (S-H-ESD) ao término do primeiro sprint e, ao término do segundo, foi possível realizar um teste de desempenho com base histórica de dois anos de transações Pix simuladas. Os resultados deste teste poderão ser vistos na seção 3.6, na página 55.

3.3 Projeto da Solução

Considerando os requisitos na seção 3.1.4, na página 37, espera-se um alto volume de transações Pix simultâneas, demandando uma solução compatível com essa carga e cuja detecção de anomalias não supere 200ms. Cenários como esse não são incomuns e as soluções criadas para atendê-los acabam tornando-se referências para aplicação em situações semelhantes. No contexto deste trabalho, a solução, na forma de uma arquitetura de referência, baseia-se no NBDRA (NIST Big Data Reference Architecture), conforme ilustrado na Figura 13.

Figura 13 – Arquitetura da solução



Fonte: elaborado pelo autor

Esta arquitetura é composta por 7 grandes blocos, separados por responsabilidades, conforme descritos abaixo:

- Data provider: IAM do banco como autorizador de transações Pix e fornecedor dos dados para o software de detecção de anomalias;
- Processing: software de detecção de anomalias (Analisador) das transações Pix e software de ETL para Big Data Analytics;
- Platforms: Data Lake para armazenamento de dados do software de detecção de anomalias e Data Warehouse para uso de análise de dados;
- Messaging/Communications: integração com software de preparação de e-mails sobre anomalias detectadas e serviço de notificação para envio destes e-mails;
- Data consumer: consumidores de dados disponibilizados no Data Warehouse e o próprio software de preparação de e-mails;

- Resource Management: gerenciamento de recursos, executado pelo provedor de nuvem (AWS), garantindo, por exemplo, a escalabilidade do software de detecção de anomalias conforme a demanda;
- Infrastructures: infraestrutura de rede, computacional (Analisador e ETL) e de armazenamento (Data Lake e Data Warehouse) sendo disponibilizadas como serviços pelo AWS.

Também pode-se observar duas regiões delimitadas por chaves, conforme descritas a seguir:

- Detecção de Anomalias: composto pelas regiões tracejadas “Analisador” e “Data Lake”, destinam-se ao software executável de detecção de anomalias e a sua área de armazenamento de dados, respectivamente;
- Big Data Analytics: composto pelas regiões tracejadas “ETL” e “Data Warehouse”, destinam-se ao software de extração, transformação e carga de dados provenientes do analisador a fim de alimentar o banco de dados para consultas analíticas.

A Tabela 9 descreve os elementos arquiteturais (círculos identificados por EAn), assim como a sua aplicação no contexto desta monografia.

Tabela 9 – Elementos arquiteturais

Id	Nome	Descrição
EA1	Elastic Container Registry (ECR)	O software compilado de detecção de anomalias é armazenado neste repositório, de forma versionada, a fim de ser executado no EKS como containers, descrito abaixo.
EA2	Cluster Kubernetes (Analisador)	Clusters EKS oferecem escalabilidade de containers e previsibilidade de custo. O software de detecção de anomalias, identificado aqui como “Analisador”, é executado neste ambiente, de forma elástica, atendendo às transações Pix, via API REST, conforme a demanda dos usuários do banco.
EA3	S3 Buckets (Data Lake)	Este é um serviço de armazenamento de dados, estruturados ou não. O software de detecção de anomalias o utiliza para a formação de um Data Lake, a fim de guardar e recuperar rapidamente os dados das transações Pix realizadas por seus usuários, formando as séries históricas utilizadas na detecção de anomalias.
EA4	Lambda Functions (ETL)	Lambda functions é uma tecnologia serverless e o mecanismo de ETL se beneficiará da capacidade elástica dessa plataforma para

		transformar os dados gerados pelo software de detecção de anomalias em uma estrutura baseada no Star Schema armazenada no Redshift, descrito abaixo.
EA5	Kinesis Firehose	Este componente realiza a ingestão de grandes quantidades de dados no Redshift a partir de comandos COPY, e seu uso é recomendado pelo AWS, que desencoraja o uso de comandos INSERT, cujo desempenho é muito inferior em larga escala. Sua aplicação neste projeto se justifica dada a alta demanda de transações Pix descrita na seção 3.1.4 (página 37).
EA6	Glue Data Catalog	Além de outras funções, o Glue oferece um serviço de catálogo de dados, essencial para a governança dos dados. Neste projeto, ele manterá o catálogo de dados sobre transações pix e anomalias detectadas.
EA7	Redshift	Um banco de dados colunar e distribuído, capaz de lidar com grandes quantidades de dados mantendo bom desempenho. Excelente recurso para cálculos de agregações em consultas analíticas. Neste projeto, será utilizado para viabilizar análise de dados a partir do modelo Star Schema mencionado na seção 3.3.2.1 (página 47).
EA8	Kinesis Streams	Broker de mensageria de alto desempenho capaz de se integrar com várias origens e destinos de dados. O software de detecção de anomalias o utilizará como mecanismo de integração de dados com o Big Data Analytics descrito na seção 3.3.2 (página 47).
EA9	Simple Notification Service (SNS)	Serviço de notificação capaz de envio de mensagens por vários meios, inclusive e-mail. A notificação de anomalias, identificadas pelo software de detecção, será realizada pelo uso deste serviço.
EA10	Lambda Function	Conforme descrito no EA4, o processamento das notificações se beneficia da elasticidade desse serviço, sendo usado aqui para a formatação dos e-mais relacionados às anomalias detectadas.

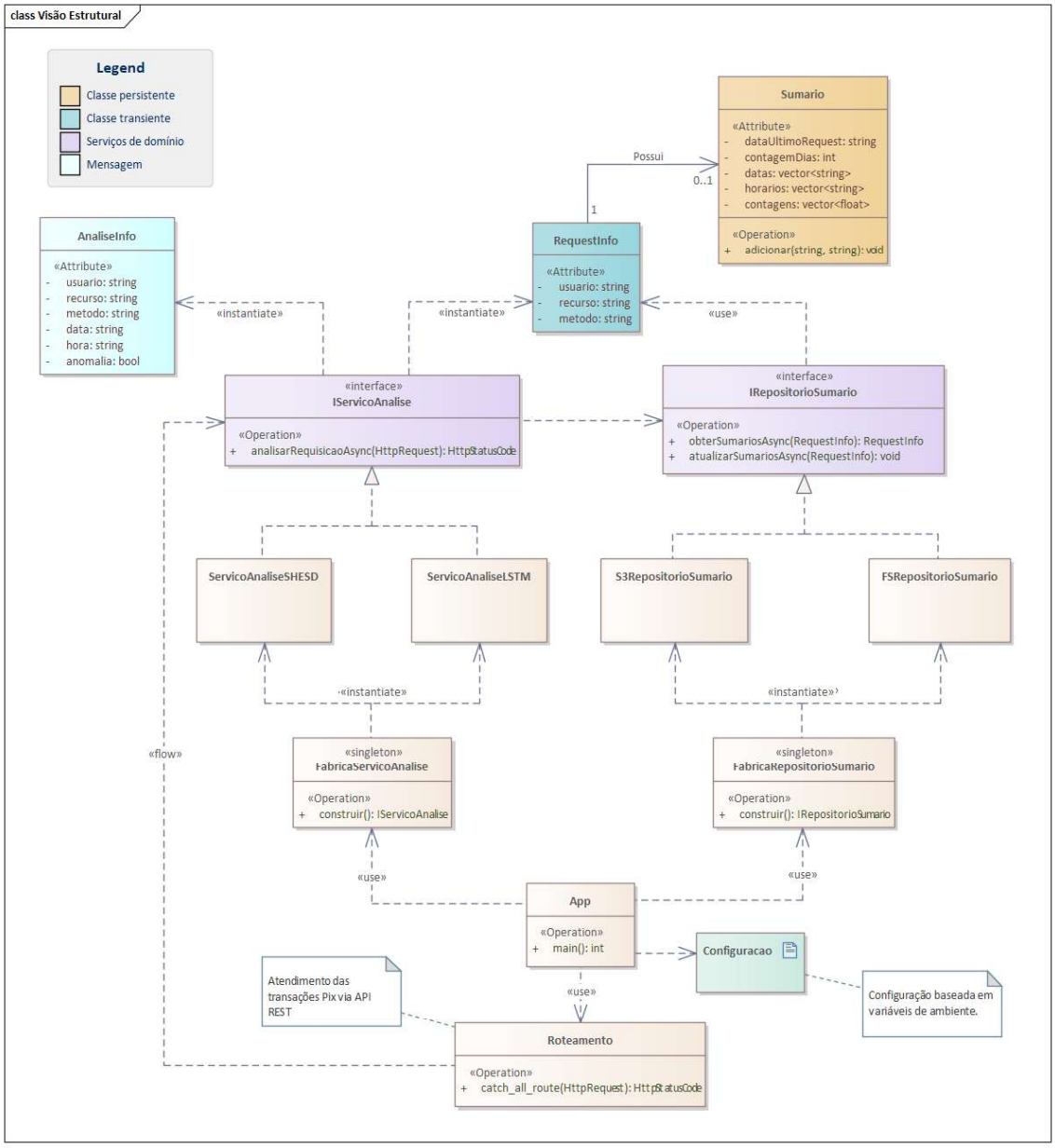
Todos os elementos que compõem a arquitetura devem estar associados a um ou mais requisitos, sejam funcionais ou de qualidade. O ANEXO A, na página 65, contém a matriz de rastreabilidade entre estes requisitos e os elementos arquiteturais que os endereçam.

3.3.1 Detecção de anomalias

3.3.1.1 Aspectos Estruturais

Os aspectos estruturais dizem respeito aos elementos que compõem o software de detecção de anomalias e suas associações, estando ilustrados na Figura 14.

Figura 14 – Aspectos Estruturais



Fonte: elaborado pelo autor

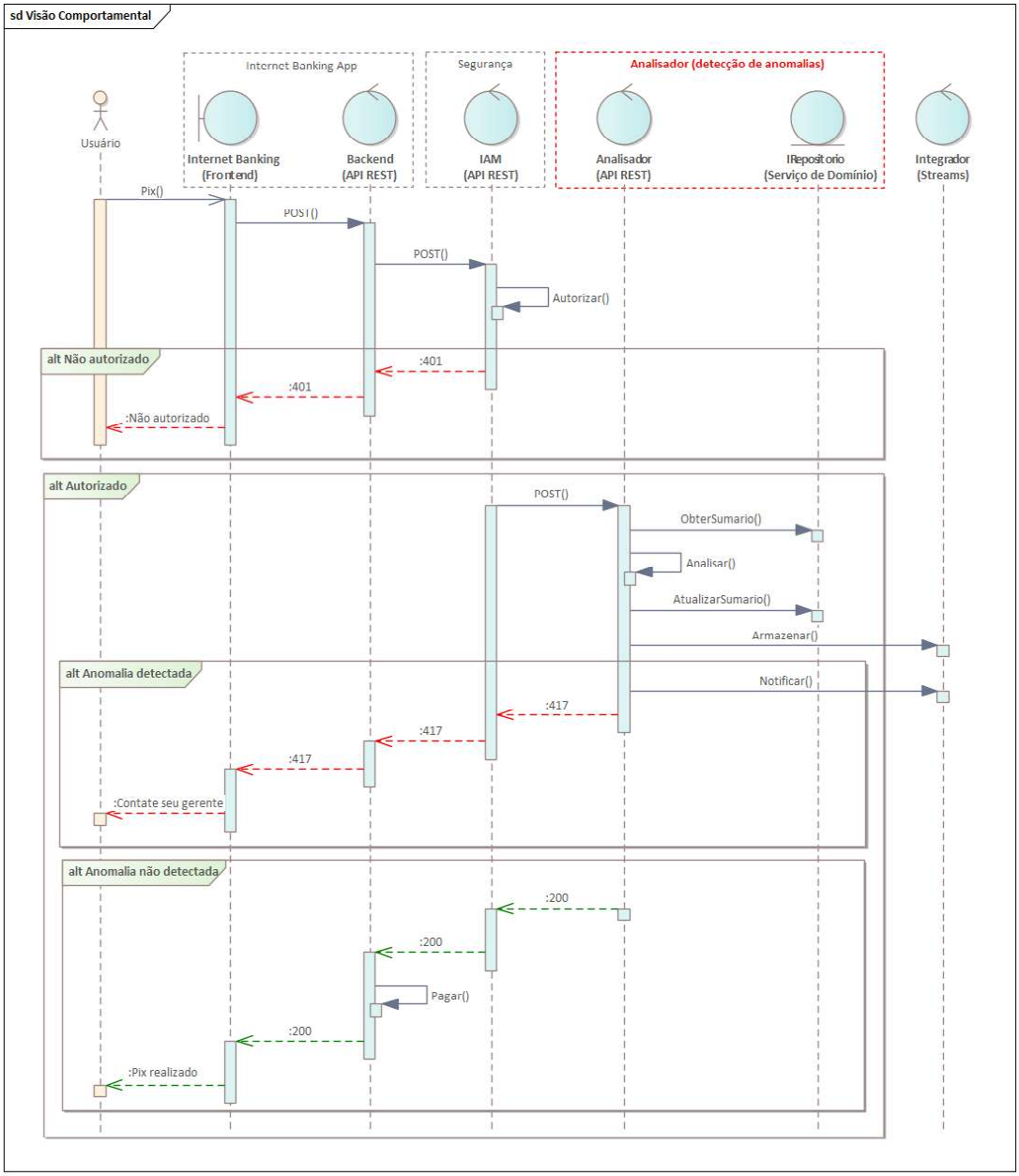
Dada a necessidade de desempenho, procurou-se racionalizar o I/O (Input/Output) envolvido na recuperação e gravação de dados no Data Lake pelo analisador. Dessa forma, a classe Sumario é a única classe persistente e foi concebida para agrupar todos os dados necessários para a detecção, otimizando a serialização (gravação) e desserialização (leitura) dessa estrutura, para e de JSON respectivamente.

Outro aspecto importante é a capacidade de escolha de serviços como o algoritmo de detecção e o tipo de Data Lake a usar baseado em configuração da aplicação.

3.3.1.2 *Aspectos Comportamentais*

Os aspectos comportamentais referem-se às interações do software de detecção de anomalias com os elementos externos a ele, assim como entre seus serviços internos, estando representados na Figura 15.

Figura 15 – Aspectos Comportamentais



Fonte: elaborado pelo autor

Pelo diagrama acima, pode-se observar que o analisador se encontra “atrás” da solução de IAM da instituição, sendo requisitada apenas após a autorização da

transação Pix, via API REST, iniciada pelo usuário no uso de uma aplicação de Mobile Banking.

Uma vez autorizada, o algoritmo de deteção de anomalias avalia a requisição com base no usuário, recurso (Pix) e método, assim como no histórico de transações Pix armazenado no Data Lake. Seu resultado é enviado ao mecanismo de mensageria (Kinesis Streams, no caso do AWS) para que seja processado por consumidores como o ETL.

Por fim, uma resposta é retornada ao IAM com os dois principais cenários ilustrados: anomalia detectada (HTTP status code 417) ou não detectada (HTTP status code 200).

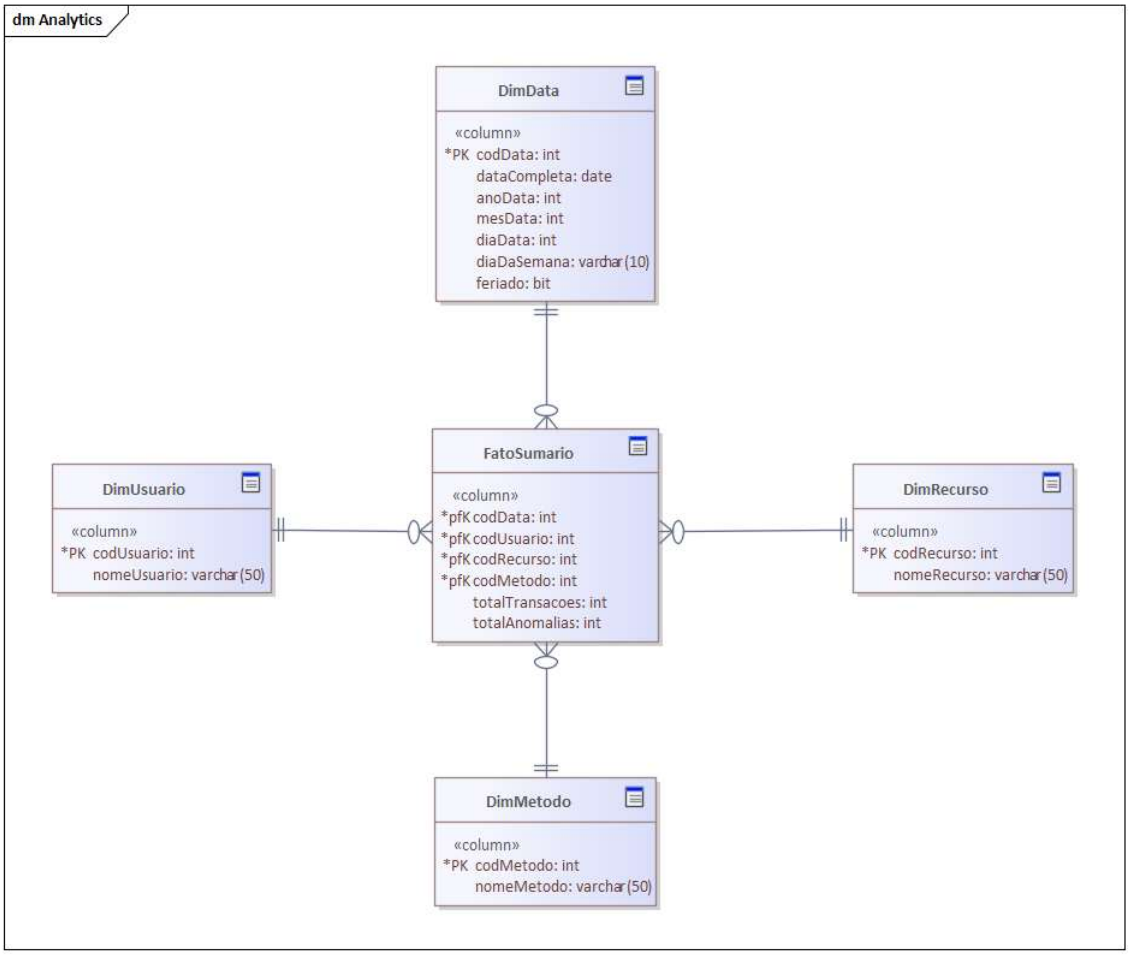
3.3.2 Big Data Analytics

3.3.2.1 Modelo de Dados

O modelo de dados para o Data Warehouse, ilustrado na Figura 16, foi concebido para dar visibilidade das anomalias envolvendo transações, como:

- Relação entre transações Pix e anomalias detectadas por período
- Total de anomalias envolvendo recursos específicos (Pix)
- Histórico comportamental de determinado usuário por período
- Período de maior incidência de anomalias por recurso (Pix)

Figura 16 – Modelo de Dados



Fonte: elaborado pelo autor

Para isso, adotou-se um modelo na forma Star Schema composta por uma tabela fato central (FatoSumario) associada a 4 dimensões (DimData, DimRecurso, DimMetodo e DimUsuario).

Softwares de análise de dados, como o Power BI, podem se conectar ao Data Warehouse e realizar consultas como as mencionadas acima, desde que com as devidas credenciais de acesso.

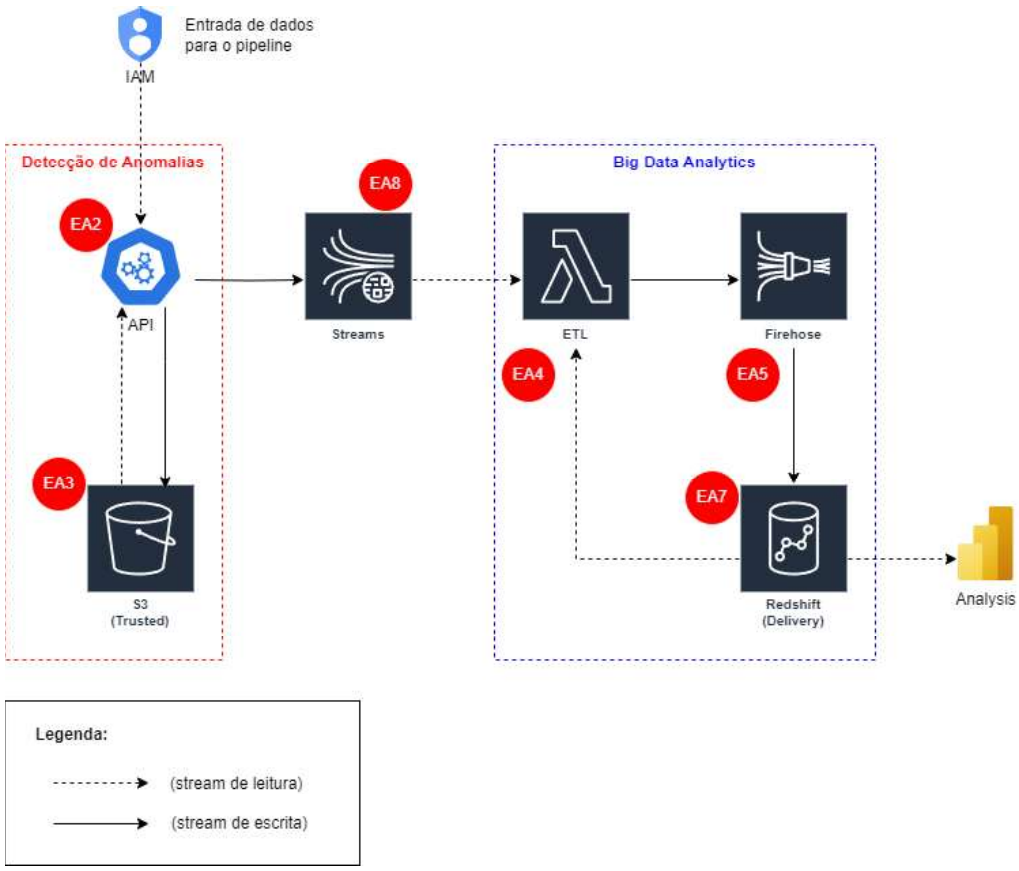
3.3.2.2 Pipeline de Dados

Os dados do Data Lake (vide item EA3 na Tabela 9, página 43), no contexto deste trabalho, são restritos ao software de detecção de anomalias, além de não estarem em um formato adequado à análise de dados multidimensional (consultas baseadas em um ou mais atributos das transações Pix). Por isso, torna-se necessário um

mecanismo que transforme os dados provenientes do analisador para que sejam carregados no Data Warehouse (item EA7 da Tabela 9, na página 43), segundo o modelo de dados mencionado acima.

A Figura 17 representa uma visão desse mecanismo, extraído a partir da arquitetura da solução (Figura 13, página 42), contendo apenas os elementos arquiteturais (EA) que constituem o pipeline de dados.

Figura 17 – Pipeline de dados



Fonte: elaborado pelo autor

O funcionamento desta pipeline de dados é descrito na Tabela 10:

Tabela 10 – Etapas da Pipeline de Dados

Etapa	Descrição
1	A solução de IAM autentica e aprova o acesso do usuário à transação Pix, repassando-a ao software de detecção de anomalias (EA2), garantindo a validade e integridade dos seus dados.
2	O software de detecção de anomalias (EA2) realiza a verificação, armazena os resultados no Data Lake (EA3) em formato JSON, envia uma mensagem com os dados da transação atual ao Streams (EA8) e responde ao IAM o resultado da detecção. Este Data Lake armazena dados já

	validados e confiáveis, por isso o nome “Trusted”.
3	O software ETL (EA4) recebe do Streams (EA8) a mensagem com o resultado da detecção (JSON) e o transforma para o modelo relacional (Figura 16, página 48), enviando-o ao Firehose (EA5).
4	O Firehose (EA5) realiza a inserção em lote dos resultados das detecções de anomalias no Data Warehouse (EA7), que possui o nome “delivery” por seus dados estarem prontos e em um formato adequado às consultas multidimensionais para os analistas.

Uma visão mais detalhada desta pipeline de dados está disponível no ANEXO B, na página 66.

3.4 Implementação

3.4.1 Detecção de anomalias

O software de detecção de anomalias, elemento arquitetural EA2 na Figura 13 (página 42), foi implementado utilizando-se as tecnologias listadas na Tabela 11:

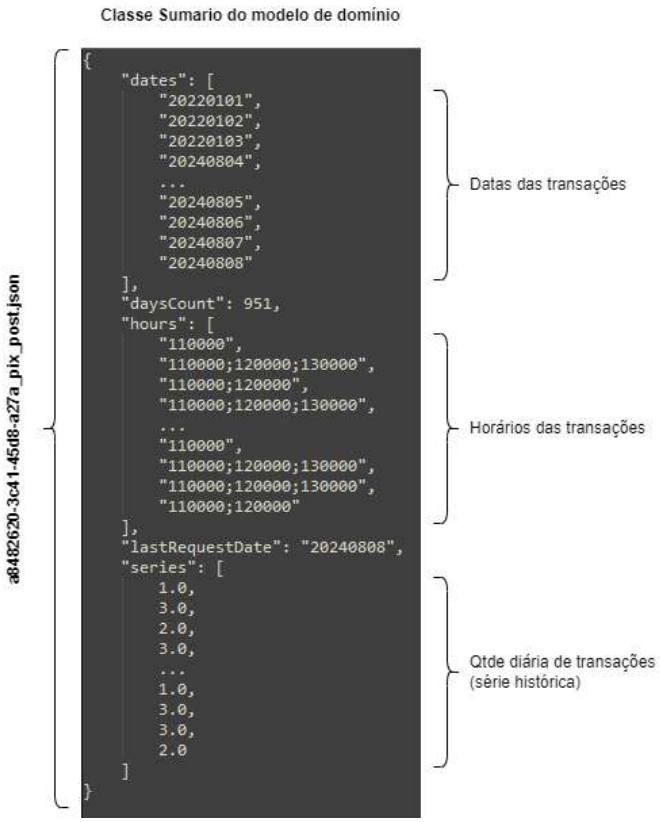
Tabela 11 – Descrição tecnológica do software de detecção

Elemento	Especificação
C++ 20	Linguagem usada para implementação, procurando atender ao requisito de qualidade RQ006 (Tabela 7, página 39).
Linux x86 64 bits	Plataforma de CPU e sistema operacional que compõem o ambiente de execução do analisador.
Gcc 11.4.0	Compilador C++.
CMake 3.22+	Linguagem de script usada para automatizar a compilação e linking dos componentes do analisador.
Crow 1.2.0	Componente web server para atender as requisições REST do IAM.
ASIO 1.30.2	Componente de infraestrutura de rede e I/O de sistema de arquivos.
JSON for Modern C++ versão 3.11.3	Componente usado para serialização em JSON do sumário de transações Pix (transformação do objeto sumário em memória para um formato que possa ser transferido por rede, assim como armazenado no Data Lake do analisador).
Seasonal-Trend Decomposition Procedure Based on Loess v0.1.2	Componente estatístico para implementação do algoritmo S-H-ESD.
Dist 0.3.0	Funções PDF e CDF para distribuições t e normais.
aws-sdk-cpp 1.11.428	Componente para acesso aos serviços do AWS, como S3 (Data Lake) e SQS (mensageria).
Docker 4.33.1	Software de conteinerização usado para testes locais do analisador.

Este software alimenta um Data Lake, elemento arquitetural EA3 na Figura 13 (página 42), que armazena estruturas de dados JSON representando um agregado de transações Pix. Cada estrutura é identificada pelo usuário, recurso (transação Pix) e método da requisição, servindo como um particionamento de dados, favorecendo transações concorrentes e o desempenho geral da solução.

Uma amostra da estrutura de dados gerada para uma determinada chave (usuário + recurso + método) pode ser vista na Figura 18.

Figura 18 – Estrutura de dados particionada



Fonte: elaborado pelo autor

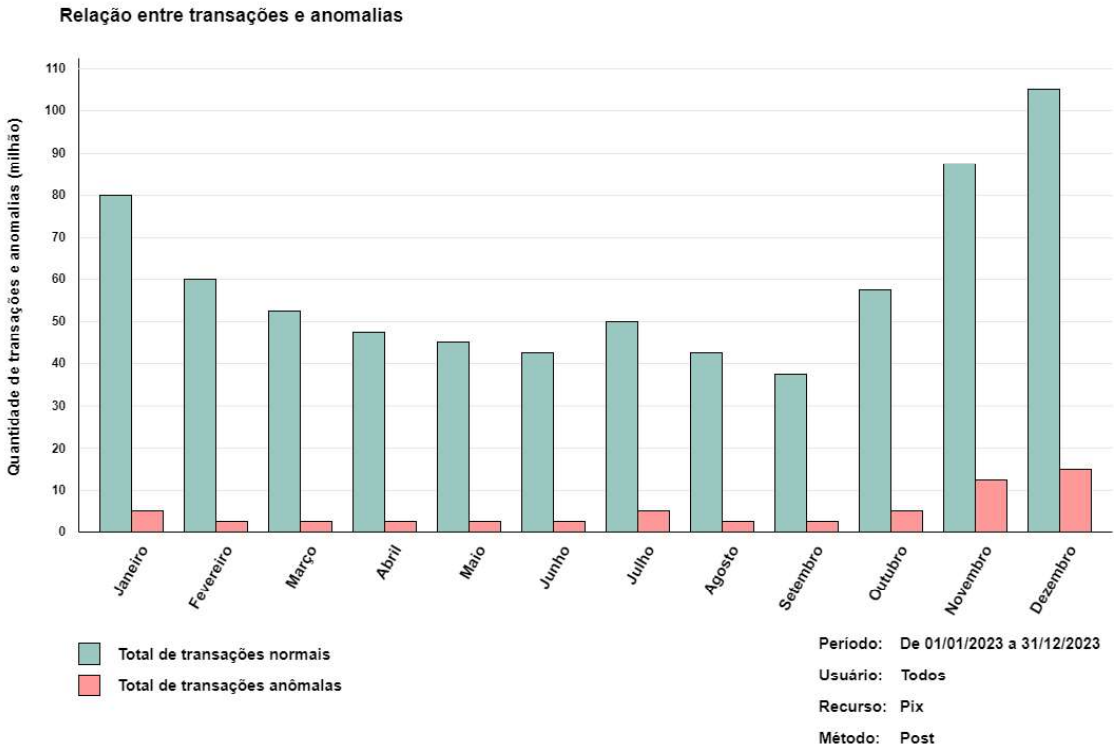
O ANEXO C, na página 67, representa o DevOps (CI/CD) criado para a automação da compilação, testes e implantação do software de detecção de anomalias, e o ANEXO D, na página 68, contém os links para o GitHub com códigos fontes e dados gerados.

3.4.2 Big Data Analytics

Além da detecção de anomalias no momento da transação Pix do usuário, a capacidade de análise e visualização dos dados históricos é parte fundamental para o entendimento do comportamento dos usuários. De fato, a partir destes dados, é possível fazer uso de algoritmos para se projetar tendências comportamentais destes usuários, permitindo o direcionamento de ações, sejam voltadas à mitigação de riscos, sejam ao provisionamento de recursos para atendimento à demanda.

A Figura 19 ilustra um protótipo de tela com a relação entre transações Pix e anomalias detectadas por período, uma das visões citadas na seção 3.3.2.1 (página 47), que podem ser obtidas a partir do Data Warehouse.

Figura 19 – Protótipo de tela para análise gráfica



Fonte: elaborado pelo autor

Neste protótipo em específico, optou-se por usar cores distintas para classificar total de transações Pix e anomalias, e o comprimento das barras para quantificar cada uma delas. Estas técnicas apresentam maior acurácia para a percepção humana.

3.5 Método de Avaliação

Para a avaliação do software de detecção de anomalias foi utilizado o seguinte ambiente computacional:

Hardware: CPU Intel (x84-64 bits), 8 GB de memória RAM e rede Gigabit Ethernet (10/100/1000 Mbps), equivalente a uma máquina “m5.large” no AWS.

Software: Sistema operacional Ubuntu 22.04.5 e um nó de um cluster Kubernetes versão 1.24.

Os seguintes passos foram adotados para a sua execução:

3.5.1 Passo 1 – Provisionamento de infraestrutura

- Disponibilização de armazenamento (Data Lake)
- Implantação do software de detecção de anomalias

3.5.2 Passo 2 – Geração de base histórica

- Período de 2 anos
- ~ 2.900.000 transações Pix via API REST
- 1.000 usuários
- Recurso fixo (Pix)
- Único método (POST – geração de ordem de pagamento)

3.5.3 Passo 3 – Execução de requisições com usuário teste

- Mantendo-se o padrão de transações Pix diárias
- Aumentando-se a quantidade de transações Pix diárias

A configuração utilizada para o algoritmo S-H-ESD pode ser vista na Tabela 12:

Tabela 12 – Configuração do algoritmo S-H-ESD

Parâmetro	Significado	Valor
MINDAYS	Número mínimo de dias para detecção de anomalias	730
PERIOD	Período usado na decomposição sazonal	7
ALFA	Nível de significância	0.05

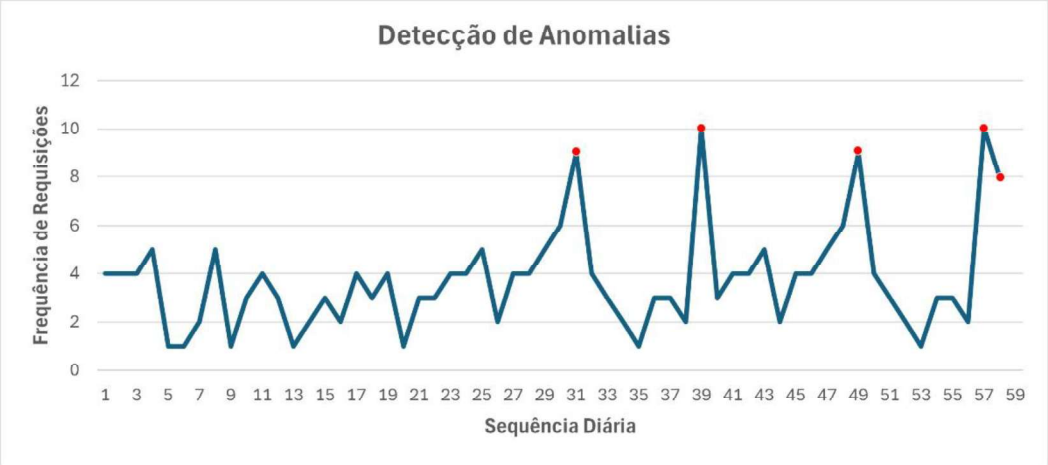
3.6 Resultados

Após a execução dos passos anteriores, foram observados os seguintes resultados:

3.6.1 Métricas de Detecção

A Figura 20 representa uma série temporal parcial de transações Pix efetuadas por um único usuário e direcionadas a um recurso específico (Pix) utilizando-se de uma mesma ação (pagar). Os pontos em vermelho sinalizam as anomalias detectadas nesta amostra.

Figura 20 – Detecção de anomalia em série histórica



Fonte: elaborado pelo autor

Aplicando-se as fórmulas de precisão, recall e F-Measure, obteve-se os seguintes resultados:

$$Precision = \frac{|\{S\} \cap \{G\}|}{|\{S\}|} = \frac{tp}{tp + fp} = \frac{3}{3+1} = 0{,}75$$

$$Recall = \frac{|\{S\} \cap \{G\}|}{|\{G\}|} = \frac{tp}{tp + fn} = \frac{3}{3+0} = 1$$

$$F = 2 \times \frac{precision \times recall}{precision + recall} = 2 \times \frac{0{,}75 \times 1}{0{,}75 + 1} = \mathbf{0,86}$$

Onde:

tp = true positive (acerto na detecção)
fp = false positive (erro na detecção)
fn = false negative (erro da detecção)

Isto significa que, para o conjunto de dados usados na simulação de transações Pix, o algoritmo apresentou um desempenho de 86% na identificação de anomalias. Deve-se considerar o uso de dados reais em um ambiente de homologação, assim como o ajuste do parâmetro ALFA (nível de significância da Tabela 12, na página 54) a fim de se avaliar o desempenho do algoritmo em um cenário o mais próximo possível de um ambiente de produção.

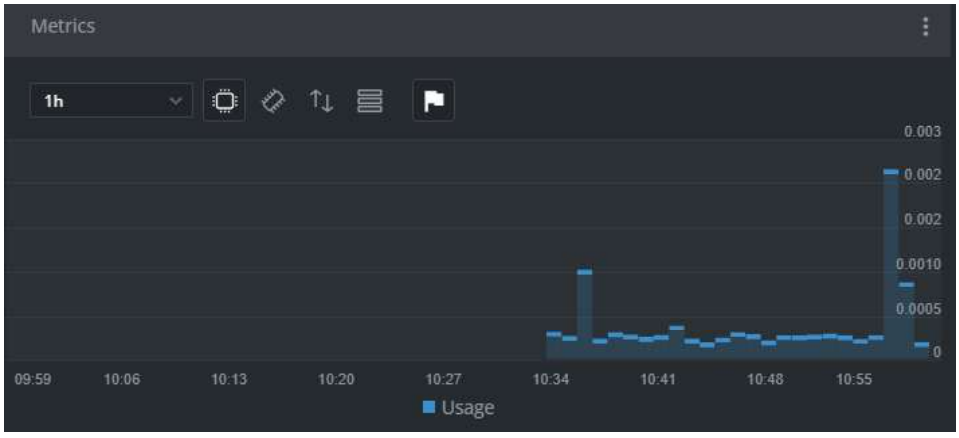
3.6.2 Métricas Computacionais

Para cada instância do componente analisador, atendendo a uma única requisição para a detecção de anomalia, foram consumidos os seguintes recursos computacionais, obtidos a partir do console de gerenciamento do Kubernetes:

3.6.2.1 CPU

Consumo de aproximadamente 2m (mili) core de uma CPU (física ou virtual), conforme a Figura 21:

Figura 21 – Consumo de CPU

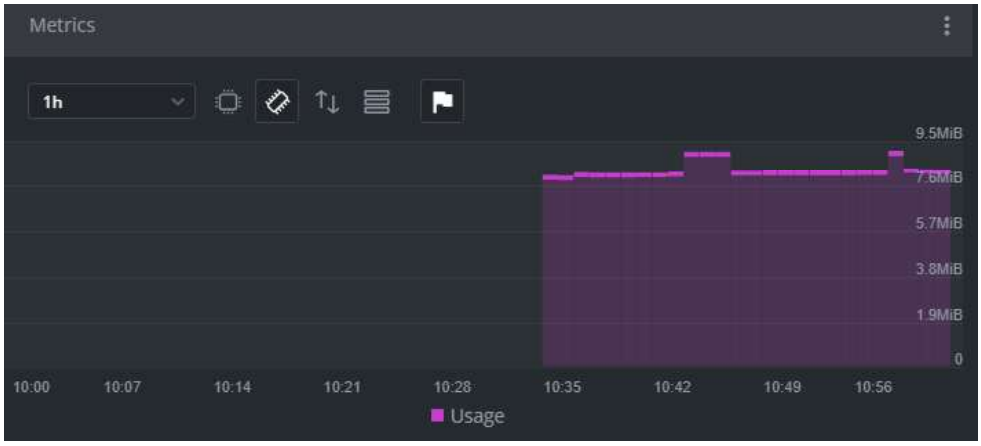


Fonte: elaborado pelo autor

3.6.2.2 Memória

Consumo de até 9,1MB de RAM, conforme a Figura 22:

Figura 22 – Consumo de memória



Fonte: elaborado pelo autor

3.6.2.3 I/O de rede

Tráfego de aproximadamente 150 bytes de dados, conforme a Figura 23:

Figura 23 – Consumo de rede

Fonte: elaborado pelo autor

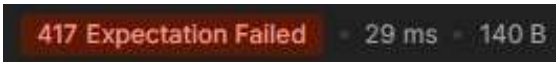
3.6.2.4 Tempo de resposta

O tempo de resposta para atendimento a uma requisição, incluindo a detecção de anomalia, foi de aproximadamente 30ms, conforme ilustrado abaixo:

Figura 24 – Tempo de resposta para uma requisição normal

Fonte: elaborado pelo autor

Figura 25 – Tempo de resposta para uma requisição anômala



Fonte: elaborado pelo autor

3.6.2.5 Tamanho da imagem

Imagem de cerca de 37MB, com código compilado nativamente, conforme a Figura 26:

Figura 26 – Tamanho da imagem Docker

Fonte: elaborado pelo autor

Do ponto de vista de recursos computacionais, o baixo consumo de CPU e memória obtidos permite que uma única instância do software de detecção de anomalias possa atender a várias transações Pix em paralelo. Para a formação do cluster de analisadores (Figura 13, página 42), cada instância deve ter uma configuração mínima de 2 vCPUs e 8 GB RAM, devendo escalar automaticamente ao atingir 80% ou mais de uso de CPU ou memória da instância.

O tráfego de rede também se mostrou bem reduzido, podendo ser suportado em uma infraestrutura Gigabit, como as existentes em provedores de nuvem (AWS e Azure).

O tempo de resposta para a detecção de anomalias atende ao requisito RQ001 (Tabela 7, página 39), cabendo um teste em ambiente de homologação com dados reais.

Por fim, uma imagem Docker pequena favorece o rápido provisionamento de novas instâncias do software de detecção de anomalias (analisadores), atendendo aos requisitos RQ002 e RQ003.

4 CONSIDERAÇÕES FINAIS

4.1 Conclusões

O tema central deste trabalho, detecção de anomalias, surgiu de uma necessidade real durante a concepção de um projeto de IAM (Identity and Access Management) que deverá ser disponibilizado como um SaaS (Software as a Service).

Todas as conclusões e lições aprendidas na realização deste trabalho serão melhor abordadas sob duas perspectivas: processo e produto.

4.1.1 Processo

O DSR (Design Science Research) estabeleceu a disciplina necessária para a construção do protótipo funcional, produto desta monografia, considerando a complexidade do tema abordado e seu contexto, transações Pix via Mobile Banking. Pode-se destacar que a busca pela fundamentação teórica (algoritmos de detecção de anomalias) e de trabalhos relacionados ao tema, assim como a etapa dedicada ao projeto da solução baseado nesse conhecimento, formaram uma sólida base para a sua construção, integrando-se bem ao SCRUM na forma de backlog de requisitos e entregas evolutivas.

4.1.2 Produto

Os requisitos funcionais e não funcionais, especificados a partir das informações sobre o problema e do projeto da solução (ambos provenientes do DSR), geraram expectativas altas em relação à capacidade de resposta do software de detecção de anomalias.

Ficou claro que uma abordagem diferenciada seria necessária, levando à escolha de técnicas de Big Data que pudessem entregar esta capacidade. Em particular, o uso de armazenamento não estruturado, como S3 Buckets, a redução de dados na computação das requisições, e o uso de tecnologias escaláveis, como containers

Kubernetes e Lambda Functions, foram fundamentais na viabilização da solução, pois permitem o processamento paralelo massivo de dados.

O software executável de detecção de anomalias apresentou um baixo consumo de CPU e memória, favorecendo a economia de energia.

Por outro lado, o algoritmo implementado nessa plataforma demonstrou ter um bom desempenho em termos de detecção, embora os testes tenham sido realizados com dados simulados.

4.2 Trabalhos Futuros

Apesar do algoritmo S-H-ESD ter apresentado um bom desempenho com dados simulados, torna-se necessária a implantação do software de detecção de anomalias em um ambiente com dados reais com o único objetivo de se avaliar os resultados em um cenário próximo ao de produção, sem interferir com o processo de autorização porventura implantado.

Outros algoritmos, como LSTM, estarão sendo implementados e configurados para uso alternativo ao S-H-ESD. Dessa forma, será possível fazer uma análise de desempenho entre diversas abordagens.

REFERÊNCIAS BIBLIOGRÁFICAS

BANCO CENTRAL DO BRASIL. **Sistema de Pagamentos Brasileiro (SPB) - Pix**. Disponível em: <https://www.bcb.gov.br/estabilidadefinanceira/pix>. Acesso em: 31 out. 2024.

BOLZANI, I. Golpes e fraudes associados ao PIX crescem no país, dizem especialistas. **g1.globo.com**, 15 maio 2023.

CARMINATI, M. et al. BankSealer: A decision support system for online banking fraud analysis and investigation. **Computers and Security**, v. 53, p. 175–186, 1 set. 2015.

CAVANILLAS, J. M.; CURRY, E.; WAHLSTER, W. **New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe**. Saarbrucken: Springer, 2016.

CERVANTES, H.; KAZMAN, R. **Designing Software Architectures**. Boston: SEI, 2016.

DAMA-DMBOK. 2nd. ed. Basking Ridge: Technics Publications, 2017.

ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. 7ª ed. São Paulo: Pearson Education, 2019.

FERRAIOLO, D. F.; KUHN, D. R. **Role-Based Access Controls**. Gaithersburg, 1992.

FIELDING. **Architectural Styles and the Design of Network-based Software Architectures**. Irvine, 2000.

FIELDING, R. T. et al. **Reflections on the REST architectural style and "principled design of the modern web architecture"**. Paderborn, 2017.

HAN, J.; PEI, J.; TONG, H. **Data Mining: Concepts and Techniques**. 4. ed. Cambridge: Elsevier, 2023.

HAYKIN, S. **Redes Neurais: Princípios e Prática**. 2. ed. Porto Alegre: Bookman, 2000.

HEIN, A. et al. **Digital platform ecosystems**. Garching bei München, 2019.

HOCHENBAUM, J.; VALLIS, O. S.; KEJARIWAL, A. **Automatic Anomaly Detection in the Cloud Via Statistical Learning**. , 2017. Disponível em: <http://arxiv.org/abs/1704.07706>

KELLEHER, J. D. **Fundamentals of Machine Learning for Predictive Data Analytics**. 2. ed. Cambridge: MIT Press, 2020.

KOTLER, P.; KELLER, K. L.; CHERNEV, A. **Administração de Marketing**. 16. ed. Porto Alegre: Bookman, 2024.

KRISHNAN, K. **Data Warehousing in the Age of Big Data**. Waltham: Elsevier, 2013.

MÁXIMO, W. **Pix bate recorde e supera 224 milhões de transações em um dia**. Disponível em: <https://agenciabrasil.ebc.com.br/economia/noticia/2024-07/pix-bate-recorde-e-supera-224-milhoes-de-transacoes-em-um-dia>. Acesso em: 18 dez. 2024.

NEGRI, R. G. **Reconhecimento de Padrões: um Estudo Dirigido**. São Paulo: Blucher, 2021.

NIST Big Data Interoperability Framework. Gaithersburg, 2019. Disponível em: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-6r2.pdf>

OUYANG, Z.; RAVIER, P.; JABLOUN, M. STL Decomposition of Time Series Can Benefit Forecasting Done by Statistical Methods but Not by Machine Learning Ones †. **Engineering Proceedings**, v. 5, n. 1, 1 jul. 2021.

PORTER, M. E. **Competição**. Rio de Janeiro: Campus, 2001.

QAZI, F. Application Programming Interface (API) Security in Cloud Applications. **EAI Endorsed Transactions on Cloud Systems**, v. 7, n. 23, p. e1, 17 out. 2023.

RONAO, C. A.; CHO, S. B. Anomalous query access detection in RBAC-administered databases with random forest and PCA. **Information Sciences**, v. 369, p. 238–250, 10 nov. 2016.

ROZANSKI, N.; WOODS, E. **Software Systems Architecture**. 2. ed. Boston: Addison Wesley, 2012.

SANDHU, R.; FERRAIOLO, D.; KUHN, R. **The NIST model for role-based access control**. Association for Computing Machinery (ACM), 26 jul. 2000.

SANOBER, S. et al. An Enhanced Secure Deep Learning Algorithm for Fraud Detection in Wireless Communication. **Wireless Communications and Mobile Computing**, v. 2021, 2021.

SCHUH, G. et al. **acatech STUDY Industrie 4.0 Maturity Index**. Munich, 2020.

SHARMA, D. H.; DHOTE, C. A.; POTEY, M. M. **Identity and Access Management as Security-as-a-Service from Clouds**. (Elsevier, Ed.), 2016.

TAYLOR, R. N.; MEDVIDOVIC, N.; DASHOFY, E. M. **Software Architecture - Foundations, Theory and Practice**. New York: Wiley, 2010.

TRIOLA, M. F. **Introdução à Estatística**. 12ª ed. Barueri: Pearson Education, 2017.

WEBBER, J.; PARASTATIDIS, S. **REST in Practice: Hypermedia and Systems Architecture**. Newton: O'Reilly, 2010.

WIEGERS; BEATTY. **Software Requirements**. 3rd. ed. Redmond: Microsoft Press, 2013.

WIERINGA, R. J. **Design Science Methodology for Information Systems and Software Engineering**. Enschede: Springer, 2014.

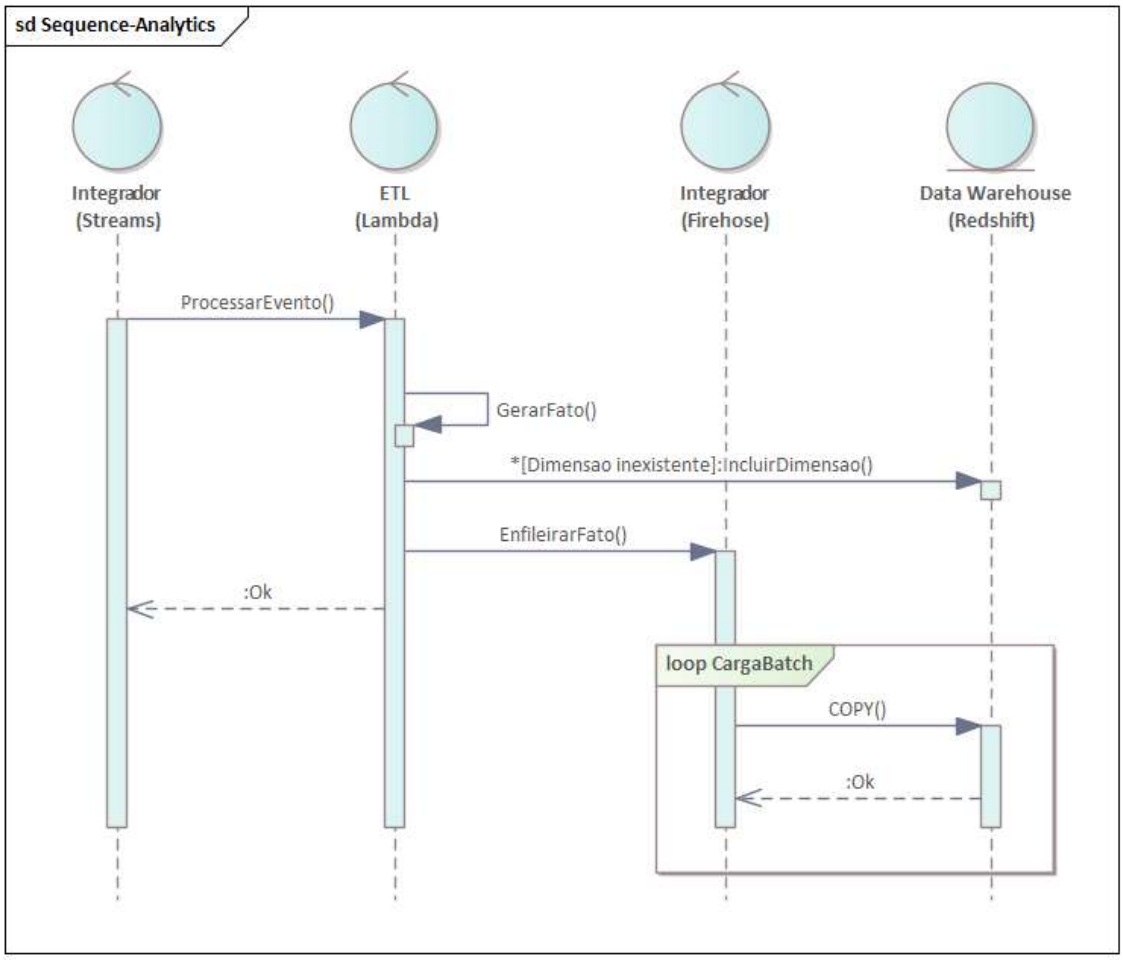
ANEXO A – MATRIZ DE RASTREABILIDADE

A matriz abaixo representa as associações entre os requisitos e os elementos arquiteturais da solução, conforme mencionado na seção 3.3 (página 41), permitindo sua rastreabilidade.

Elemento Arq. / Requisito	EA1 (ECR)	EA2 (Analisador)	EA3 (Data Lake)	EA4 (ETL)	EA5 (Firehose)	EA6 (Glue)	EA7 (Redshift)	EA8 (Streams)	EA9 (SNS)	EA10 (Lambda)
RF001 - APIs seguem padrão REST e OAuth		✓								
RF002 - Qtde mínima para formação de base histórica configurável		✓								
RF003 - Período para decomposição de 7 dias configurável		✓								
RF004 - Algoritmo usado para detecção de anomalias configurável		✓								
RF005 - Manutenção de base histórica de 2 anos		✓	✓							
RF006 - Requisições classificadas como normais retornam HTTP 200		✓								
RF007 - Requisições classificadas como anômalas retornam HTTP 417		✓								
RF008 - Requisições que não atendam ao padrão REST retornam HTTP 400		✓								
RF009 - Histórico de transações e anomalias disponibilizado para análise				✓	✓	✓	✓	✓		
RQ001 - O tempo para a detecção de anomalia deve ser inferior a 200ms		✓	✓							
RQ002 - Previsão de até 20.000 transações Pix simultâneas		✓	✓							
RQ003 - Analisador deve apresentar resiliência		✓								
RQ004 - Manutenção da privacidade e confidencialidade dos dados dos usuários			✓							
RQ005 - Notificação de anomalias por e-mail								✓	✓	✓
RQ006 - Analisador deve ser versionado	✓	✓								

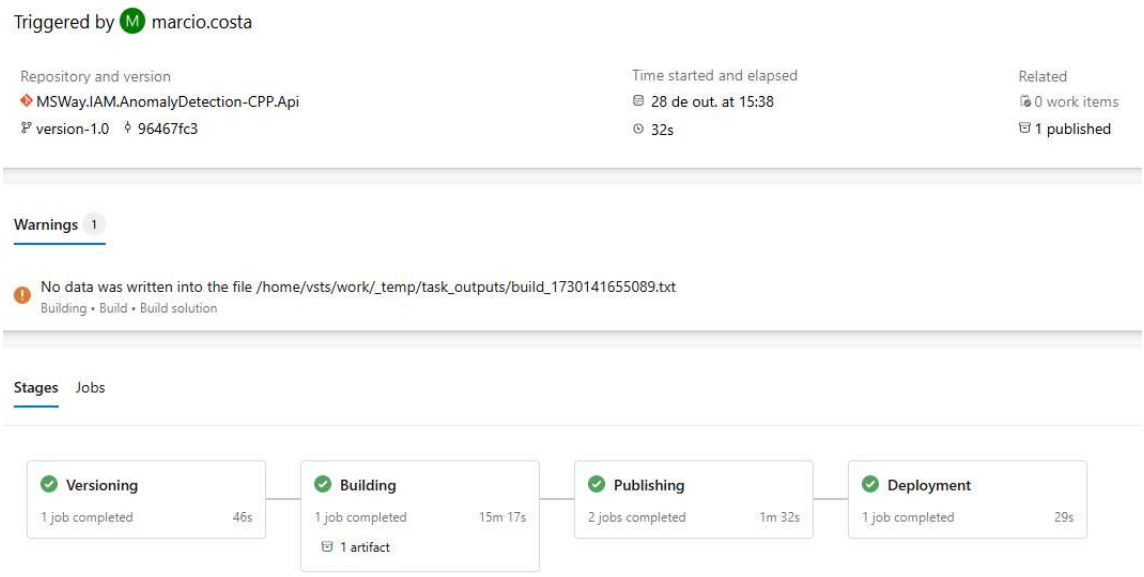
ANEXO B – DIAGRAMA DE SEQUÊNCIA DO BIG DATA ANALYTICS

O diagrama abaixo representa as interações entre os diversos elementos arquiteturais da solução de Big Data Analytics. Deve-se notar que a ingestão dos fatos é realizada pelo Kinesis Firehose, conforme consta na Tabela 9 (página 43), elemento arquitetural EA5.

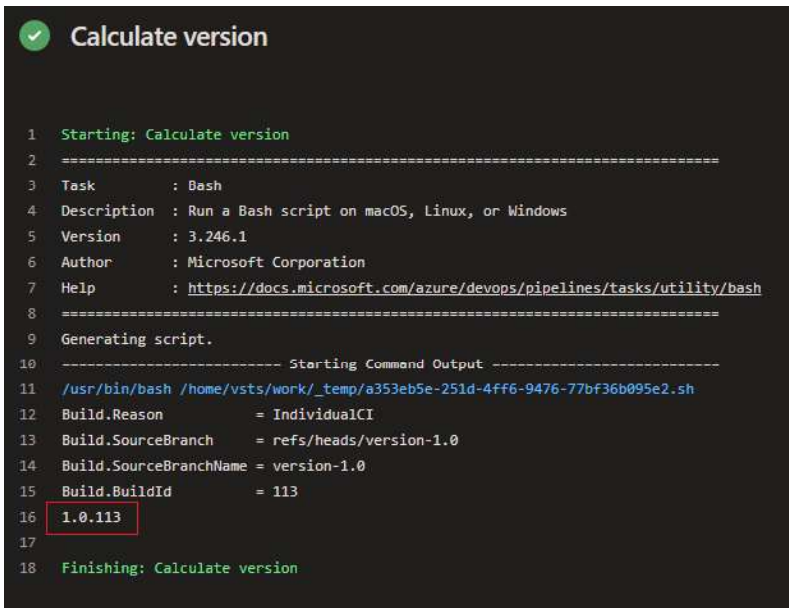


ANEXO C – PIPELINE DE DEVOPS

A automação da compilação, testes e implantação é tão importante em projetos de engenharia de dados quanto em engenharia de software tradicional. A imagem abaixo ilustra a pipeline de automação do software de detecção de anomalias (elemento arquitetural EA2 da Figura 13, página 42) e seus estágios: versionamento, compilação, publicação e implantação em ambiente de desenvolvimento.



A imagem a seguir destaca o versionamento aplicado ao software (neste exemplo, versão 1.0.113), atendendo ao requisito de qualidade RQ006 (Tabela 7, página 39).



ANEXO D – CÓDIGO FONTE E DADOS

A tabela abaixo contém os links com a localização dos softwares construídos ao longo desta monografia, assim como dos dados gerados para a avaliação do algoritmo de detecção de anomalias.

Tabela 13 – Links para o GitHub

Conteúdo	Link
Gerador de dados simulados	https://github.com/marciohacosta/anomaly-detection-loader
Dados gerados	https://github.com/marciohacosta/anomaly-detection-data
Detector de anomalias	https://github.com/marciohacosta/anomaly-detection