Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Modernizando Aplicações Legadas para a Arquitetura Multi-Cloud: Benefícios, Desafios e Tendências Futuras

Ivon Miranda Santos

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Orientador
Prof.a Dr.a Carla Rocha Silva Aguiar

Brasília
2025

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

MM672m

Miranda Santos, Ivon
Modernizando Aplicações Legadas para a Arquitetura Multi-Cloud: Benefícios, Desafios e Tendências Futuras / Ivon Miranda Santos; orientador Carla Rocha Silva Aguiar. Brasília, 2025.
79 p.

Dissertação(Mestrado Profissional em Computação Aplicada) Universidade de Brasília, 2025.

1. Multi-cloud migration. 2. Multi-cloud management. 3. Multi-cloud architecture. 4. Cloud-native architecture. 5. Vendor lock-in. I. Rocha Silva Aguiar, Carla, orient. II. Título.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Modernizando Aplicações Legadas para a Arquitetura Multi-Cloud: Benefícios, Desafios e Tendências Futuras

Ivon Miranda Santos

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Prof.a Dr.a Carla Rocha Silva Aguiar (Orientador)
PPCA/UnB

Prof. Dr. Sérgio Antônio Andrade Freitas
PPCA/UnB

Prof. Dr. Sávio Salvarino Teles de Oliveira
INF/UFG

Prof. Dr. Gladston Luiz da Silva
Coordenador do Programa de Pós-graduação em Computação Aplicada

Brasília, 25 de Janeiro de 2025

Dedicatória

Eu dedico esse trabalho a minha *esposa* ***Patrícia***, que sempre me apoiou me motivou a continuar seguindo em frente e nunca desistir, mesmo quando os desafios se revelavam muito maiores do que os que eu esperava enfrentar. ***Meu sincero, muito Obrigado***!!

Agradecimentos

Agradeço primeiramente a Deus, que sempre me ajudou e permitiu que percorresse essa jornada.

Á minha orientadora prof.a. Dra. Carla Rocha que me ajudou a trilhar esse árduo caminho, me orientando e direcionando pelas melhores escolhas, que nem sempre as são as mais fáceis mas sempre são as mais importantes, aumentando consideravelmente a qualidade deste trabalho.

Aos professores Dr. Sergio e Dr. Sávio, que ofereceram contribuições valiosas e precisas, aprimorando e elevando a qualidade desse estudo.

Aos professores do PPCA - Programa de Pós-graduação em Computação Aplicada, por todo apoio e conhecimento compartilhado ao longo dessa trajetória acadêmica.

Aos meus familiares e amigos pelo apoio, compreensão, e pelos momentos de ausência que sempre estiveram acompanhados por motivos nobres.

E aos colegas de Trabalho, que me presentearam com a oportunidade de aplicar vários dos conhecimentos obtidos nessa jornada, além das boas e produtivas conversas sobre grandes ideias e soluções inovadoras.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

A computação em nuvem proporciona para as organizações recursos inovadores, flexíveis e economicamente vantajosos, que colaboram para as empresas serem mais céleres na construção de ferramentas inovadoras. Dentre as estratégias que visam otimizar os aplicativos de negócios, destaca-se a implementação de uma arquitetura multi-cloud, essa arquitetura multi-cloud surgiu como uma abordagem estratégica para aprimoramento do uso de recursos computacionais no desenvolvimento aplicativos de negócios, com a possibilidade do uso de diversos provedores. Contudo, há momentos em que a oportunidade sugere a modernização de sistemas legados, o que exige a reengenharia da aplicação e a seleção/configuração de novos provedores e serviços. Este trabalho de monografia tem como objetivo explorar e examinar de forma abrangente os desafios e soluções relacionadas à modernização de aplicações legadas para uma arquitetura multi-cloud, focando na importância da elaboração adequada da arquitetura da aplicação para o sucesso dessa transição. **A primeira fase** deste estudo realizou uma Revisão Sistemática da Literatura para identificar e mapear os principais desafios associados à migração de sistemas legados para ambientes multi-cloud. A análise de dados utilizou técnicas de Codificação e Codificação Focada para identificar tópicos de discussão e oportunidades de pesquisa. Foram recuperados 145 artigos e, após aplicação de rigorosos critérios de qualidade, 50 artigos foram analisados minuciosamente, resultando no agrupamento dos desafios em nove categorias principais. **A segunda fase** aplica um estudo de caso etnográfico para investigar um dos desafios destacados: ***a importância da arquitetura da aplicação na migração de uma aplicação legada***. Este estudo detalhou a modernização de um sistema legado, especificamente de mainframe para ***Multi-Cloud Native Architecture***, utilizando Domain-Driven Design (DDD), Saga Orchestration Pattern e o Hexagonal Architecture. A abordagem do DDD, combinada com Event Storming, facilita a modelagem dos domínios de negócio, enquanto o Saga Orchestration e a arquitetura Hexagonal implementa e gerencia as transações distribuídas, garantindo a integridade dos dados durante falhas operacionais. A pesquisa evidenciou desafios críticos relacionados à gestão, composição e configuração em ambientes multi-cloud, destacando a importância de projetar aplicativos nativos da nuvem para mitigar preocupações como a dependência de fornece-

dores. Na sequencia, com a realização do estudo de caso, a aplicação do DDD permitiu identificar e modelar os domínios do sistema legado, para possibilitar desenhar e implementar a reengenharia da aplicação. Este trabalho enfatizou que a natureza heterogênea dos ambientes multi-cloud apresenta desafios significativos, mas que podem ser superados através de um planejamento cuidadoso e uma arquitetura bem projetada. Mesmo assim, as descobertas sugeriram que **a modernização de sistemas legados com a migração para multi-cloud** permanecem áreas de pesquisa abertas, apresentando oportunidades valiosas para futuras investigações.

Palavras-chave: Multi-cloud migration, multi-cloud management, multi-cloud architecture, cloud-native architecture, vendor lock-in

Abstract

Cloud computing provides organizations with innovative, flexible, and cost-effective resources, enabling them to be more agile in developing innovative tools. Among the strategies aimed at optimizing business applications, the implementation of a multi-cloud architecture has emerged as a strategic approach to enhance business applications and enable the use of multiple providers. However, this approach often requires the migration of legacy systems, necessitating the reengineering of the application and the selection/configuration of new providers and services.

This monograph aims to comprehensively explore and examine the challenges and solutions for migrating legacy applications to a multi-cloud architecture, focusing on the importance of proper application architecture design for the success of this migration.

The first phase of this study conducted a Systematic Literature Review to identify and map the main challenges associated with migrating legacy systems to multi-cloud environments. Data analysis used Coding and Focused Coding techniques to identify discussion topics and research opportunities. A total of 145 articles were retrieved, and after applying rigorous quality criteria, 50 articles were thoroughly analyzed, resulting in the grouping of challenges into nine main categories. **The second phase** applied an ethnographic case study to investigate one of the highlighted challenges: **the importance of application architecture in the migration of a legacy application**. This study detailed the modernization of a legacy system, specifically from a mainframe to **Multi-Cloud Native Architecture**, using Domain-Driven Design (DDD), Saga Orchestration Pattern, and Hexagonal Architecture. The DDD approach, combined with Event Storming, facilitated the modeling of business domains, while the Saga Orchestration and Hexagonal Architecture implemented and managed distributed transactions, ensuring data integrity during operational failures.

The research highlighted critical challenges related to management, composition, and configuration in multi-cloud environments, emphasizing the importance of designing cloud-native applications to mitigate concerns such as ***vendor lock-in***. Subsequently, with the case study, the application of DDD enabled the identification and modeling of the legacy

system's domains, facilitating the design and implementation of the application's reengineering.

This study emphasized that the heterogeneous nature of multi-cloud environments presents significant challenges, but these can be overcome through careful planning and well-designed architecture. Nevertheless, the findings suggested that **the modernization of legacy systems with migration to multi-cloud** remains an open area of research, presenting valuable opportunities for future investigations.

Keywords: Multi-cloud migration, multi-cloud management, multi-cloud architecture, cloud-native architecture, vendor lock-in

Sumário

1 Introdução **1**
1.1 Esboço do Problema 4
1.2 Questões de Pesquisa 5
1.3 Projeto de Pesquisa 6

2 Fundamentação Teórica **8**
2.1 Computação em Nuvem 9
2.1.1 Nuvem privada 9
2.1.2 Nuvem pública 9
2.1.3 Nuvem Híbrida 10
2.1.4 Arquitetura Multi-Cloud 10
2.2 Design da Arquitura Multi-Cloud 11
2.2.1 Objetivo e Estratégia para Arquitetura Multi-Cloud 13
2.2.2 Plano de Implantação do Multi-Cloud Kubernetes Cluster 14
2.3 Arquitetura da Aplicação Multi-Cloud Native 19
2.3.1 Projeto de uma Aplicação Multi-Cloud 20
2.3.2 Domain-Driven Design 24
2.3.3 SAGA Orchestration Pattern 26
2.3.4 Hexagonal Architecture 27

3 Metódo de Pesquisa **29**
3.1 Revisão Sistemática da Literatura 29
3.1.1 Planejamento da Revisão Sistemática da Literatura 30
3.1.2 Executando a Revisão Sistemática da Literatura 31
3.1.3 Análise de dados e informações cruzadas da RSL 35
3.2 Estudo de Caso Etnográfico 35

4 Desafios da Migração para Multi-Cloud: uma perspectiva a partir da Revisão Sistemática da Literatura **37**
4.1 DESAFIOS DA MULTI-CLOUD 37

4.2 BENEFÍCIOS DA MULTI-CLOUD 38
4.3 GESTÃO EM NUVEM 39
4.4 ARQUITETURA CLOUD-NATIVE 40
4.5 MULTI-CLOUD ARCHITECTURE 42
4.6 PROVEDORES DE SERVIÇOS EM NUVEM 44
4.7 PROCESSO DE MIGRAÇÃO PARA NUVEM 45
4.8 INFRAESTRUTURA EM NUVEM 45
4.9 RISCO E SEGURANÇA 46
4.10 VENDOR LOCK-IN 46
4.11 ARQUITETURA LEGADA 47
4.12 Resultados da Revisão Sistemática da Literatura 48

5 Uma proposta de migração para Multi-Cloud Native Architecture: a perspectiva a partir de um Estudo de Caso Etnográfico 49
5.1 O Sistema Legado 50
5.2 Motivações e Objetivos da Modernização 51
5.3 O Processo de Modernização 52
5.3.1 Fase 1 54
5.3.2 Iteração 1 54
5.3.3 Iteração 2 55
5.4 Lições Aprendidas 56
5.5 Resultados 57

6 Conclusão 58

Referências 61

Apêndice 73

A Categories Table 74

B Fichamento de Artigo Científico 80

Anexo 84

I Documentação Original `UnB-CIC` (parcial) 84

Lista de Figuras

1.1 Processo de Pesquisa, combinação entre pesquisa teórica e estudo de caso prático. 7

2.1 A imagem demostra uma arquitetura *Multi-Cloud* através da implantação de um *Multi-Cluster Kubernetes*, composta por duas nuvens públicas, *Azure* e *AWS* respectivamente, integradas com um terceiro *cluster* instalado em nuvem privada, com três replicas de *Controls Planes*, uma replica implantada em cada nuvem do *cluster* para manutenção da alta disponibilidade, e ao lado direito, sugestões de ferramentas *opensource* aderentes ao padrão *CNCF* que complementam as funcionalidades necessárias para implantação de uma arquitetura *Multi-Cloud Opensource* mitigando o risco do *Vendor Lock-in*. 12

3.1 Processo da Metodologia de Pesquisa Passo a Passo 30
3.2 Compilação dos termos que compõem a frase de pesquisa e totalização dos documentos encontrados. 31
3.3 Processo da Metodologia de Pesquisa 32
3.4 Critérios de Inclusão 33
3.5 Critérios de Exclusão 34

4.1 Domínios ou categorias identificadas por *Focus Coding*. 38
4.2 Desafios relacionados à categoria "*Gestão em Nuvem*". Cada propriedade apresentou o número de citações encontradas no estudo e o percentual de citações quando considerado apenas nesta categoria. 40
4.3 Desafios relacionados à categoria "*Arquitetura Cloud-Native*". Cada propriedade apresenta o número de citações encontradas no estudo e o percentual de citação quando considerado apenas nesta categoria. 41
4.4 Desafios relacionados à categoria *Arquitetura Multi-Cloud*. Cada propriedade apresenta o número de citações encontradas no estudo e o percentual de cotação quando considerado apenas nesta categoria. 43

5.1 Sistema Legado. 51
5.2 Processo de Modernização de Sistema Legado. 53
5.3 Sistema Modernizado. 53

Lista de Tabelas

2.1 Descrição de algumas ferramentas Opensource aderentes às CNCF, que complementam a implantação da Arquitetura Multi-Cloud, com Multi-Cluster Kubernetes, suprindo necessidades como segurança, disponibilidade, gerenciamento, etc... 18

Lista de Abreviaturas e Siglas

API Interface de Programação de Aplicativos.

AWS Amazon Web Services.

Azure Microsoft's cloud computing platform.

CNCF Cloud Native Computing Foundation.

DevOps Development and Operations.

FIPS Federal Information Processing Standards.

GCP Google Cloud Platform.

IaaS Infrastructure as a Service.

LLT Long Live Transaction.

PaaS Platform as a Service.

RSL Revisão Sistemática da Literatura.

SaaS Software as a Service.

SLA Service Level Agreement.

Capítulo 1

Introdução

A transformação digital tem sido um tema presente nas empresas desde a década de 1990. Observa-se que a maioria ainda não possui uma base operacional robusta capaz de sustentar essa mudança necessária para aprimorar atividades primordiais do processo de negócio. Trata-se de um trabalho de longo prazo, complexo e com custos significativos [1]. No entanto, algumas empresas têm se destacado ao implementar transformações simultâneas digitalizando a organização e otimizando a eficiência operacional, buscam propostas disruptivas por meio da inovação e da agilidade. Essa abordagem é considerada essencial para manter a competitividade e preparar a empresa para os desafios organizacionais futuros [1].

A modernização de aplicações legadas, constitui parte dessa etapa de transformação digital com a migração de aplicações para nuvem, um processo complexo, longo e oneroso, que demanda uma abordagem minuciosa e embasada. Nesse contexto, pesquisadores têm se dedicado ao desenvolvimento de estratégias para auxiliar a tomada de decisões eficientes e sustentáveis, por meio de uma análise detalhada dos requisitos organizacionais, da adequação dos processos de negócio e da compreensão clara das implicações de custo. Esses estudos são fundamentais para garantir que a transição para ambientes de nuvem esteja alinhada às necessidades tecnológicas e aos objetivos estratégicos da empresa [2], [3].

Essa transformação evoluiu para um contexto multidisciplinar, diferentemente do cenário anterior, em que a escolha de um único fornecedor de nuvem e a integração de seus serviços e APIs com sistemas legados locais eram práticas comuns. Atualmente, a tendência se direciona para uma arquitetura multi-cloud, na qual os desenvolvedores selecionam e combinam os melhores serviços de diferentes provedores de nuvem. Essa abordagem otimiza a infraestrutura em nuvem ao permitir a oferta de recursos heterogêneos de diversos fornecedores, considerando características relevantes, como as influenciadas por restrições geopolíticas ou regulamentações regionais específicas [4], [5].

Implementar uma arquitetura multi-cloud pode proporcionar diversos benefícios, incluindo a diminuição de dependência de fornecedores únicos, com o desenvolvimento de sistemas mais resilientes capazes de suportar falhas e a redução de riscos de disponibilidade ligados a problemas de hardware ou interrupções de serviço [5]. Para possibilitar o uso de uma arquitetura multi-cloud, a seleção correta do padrão arquitetural tem papel fundamental no êxito da implantação, essa abordagem destaca a vantagem de poder escolher entre as melhores ofertas e recursos de cada fornecedor [6]. Porém, esse ambiente heterogêneo evidencia pontos de atenção como os riscos legais e de segurança cibernética, atributos essenciais para serem acompanhados [7].

Nesse contexto, existem várias aplicações legadas candidatas ao processo de modernização e migração para uma arquitetura Cloud-Native [8]. Porém, as aplicações legadas enfrentam problemas distintos pela geral ausência de documentação apropriada, dependências desatualizadas e fraca observância às metodologias modernas de engenharia de software, o que contribuiu para complicadores para equipes técnicas no tocante à manutenção e evolução destes sistemas [9]. Essas aplicações não foram projetadas para utilizar escalabilidade dinâmica, mas para operar em servidores físicos robustos. Assim, sua modernização requer modificações consideráveis na arquitetura. A migração para ambientes de nuvem, e a adoção de arquitetura nativa da nuvem, como os microsserviços, ainda que envolva complexidades, permite flexibilidade, escalabilidade e gestão otimizada de custos, apresentando vantagens técnicas e negociais quando implementadas corretamente [10], [11].

Mesmo com os desafios destacados, a adoção de soluções multi-cloud é recomendada na literatura especializada sobretudo pelos benefícios como maior resiliência, tolerância a falhas e flexibilidade operacional [6]. Nesse Contexto, a escolha do modelo arquitetural mais empregado para essa modernização tem sido a arquitetura de microsserviços, por apresentar uma abordagem para o desenvolvimento de software que foca na criação de pequenos serviços independentes, com a promoção de práticas ágeis e DevOps, com automação de infraestrutura para otimizar a entrega contínua, além de um gerenciamento e governança de dados descentralizados entre os serviços. Essa abordagem proporciona maior agilidade e escalabilidade em comparação com as arquiteturas tradicionais, podendo reduzir o custo total de operação e manutenção, além de acelerar o tempo de implementação [12].

A literatura destaca a ausência de um processo consolidado para a seleção de modelos de nuvem e a análise dos riscos e benefícios envolvidos, representando um desafio no processo de modernização [10]. Ainda assim, pesquisas que abordam a migração ou desenvolvimento de aplicações nativas em nuvem foram evidenciadas [13] [14] [15] [16] [17] [18] [19], indicando a relevância do tema e o interesse da comunidade científica em abordá-lo. Ademais, grande parte dos estudos concentram-se em aspectos específicos do

paradigma multi-nuvem. Dentre essas, a pesquisa de [13] destaca-se por abordar três pontos essenciais:

- A caracterização do conceito de multi-cloud.
- A definição arquitetural de aplicações multi-cloud.
- Elencar questões de investigação abertas que considera essenciais para serem trabalhadas no futuro

Com base nas contribuições dos trabalhos revisados, o presente estudo concentra-se nas necessidades específicas do setor financeiro, abordando os desafios envolvidos na migração de aplicações legadas Mainframe para aplicações nativas da nuvem, com a possibilidade de implantação em arquitetura Multi-Cloud, visando reduzir a dependência ou bloqueio de fornecedores (*vendor lock-in*). Essa abordagem coincide com uma das quatro principais tendências de pesquisa identificadas por [13], a saber:

1. Caracterização de aplicações Multi-Cloud e Multi-Cloud Native.
2. Projeto e design para Multi-Cloud.
3. DevOps para Multi-Cloud.
4. Segurança para aplicações Multi-Cloud Native.

A primeira parte deste trabalho apresentou uma Revisão Sistemática da Literatura (RSL) com o objetivo de identificar e mapear os principais desafios associados à modernização tecnológica, com ênfase na migração de aplicações legadas para ambientes multi-cloud. Foram recuperados 145 trabalhos de fontes selecionadas e analisados em detalhes 50 artigos. Na sequência, técnicas de codificação Inicial e Focada foram realizadas para mapear 9 categorias e 52 propriedades relacionadas aos desafios na migração de aplicações para uma arquitetura multi-cloud, resultando em 4.555 segmentos. Dessa forma, o trabalho contribuiu com o mapeamento dos desafios enfrentados na migração de aplicações legadas para arquitetura multi-cloud, com a identificação de oportunidades de pesquisa para questões ainda não resolvidas pelos especialistas.

Os resultados revelam desafios críticos, incluindo gestão em nuvens heterogêneas, tolerância a falhas, escalabilidade, questões de segurança dos dados e a dependência de fornecedores, além da necessidade de reengenharia arquitetural para adequar os aplicativos às exigências das plataformas nativas da nuvem. Esses desafios demandam atenção especial dos arquitetos de software. Contudo, a estratégia multi-cloud proporciona benefícios fundamentais, como independência de fornecedores, maior disponibilidade, escalabilidade inter-nuvens e otimização de custos. Embora os desafios na migração para nuvem

sejam significativos, os benefícios tendem a superar os riscos, desde que uma estratégia cuidadosamente detalhada e planejada seja implementada [13].

A segunda parte, descreve um estudo de caso real na modernização de uma aplicação bancária com mais de duas décadas em atividade, originalmente codificado em COBOL, e implantado em arquitetura Mainframe, para uma arquitetura Cloud-Native, mitigando os riscos que possam ocasionar o bloqueio do fornecedor (*vendor lock-in*) e impedir a implantação em uma arquitetura Multi-Cloud.

1.1 Esboço do Problema

O ritmo acelerado da evolução tecnológica impulsiona a constante atualização das empresas em busca de soluções que assegurem uma posição de vantagem competitiva diante dos concorrentes.

Visando por soluções, são desenvolvidos roteiros de atualizações tecnológicas, que abrange vários estágios, iniciando pela análise do ambiente atual, considerando as soluções e padrões disponíveis, até a implementação de tecnologias modernas e inovadoras [20].

Para efetivar essas transformações e alcançar o aumento de velocidade de entrega e produtividade, requer a adoção de novas tecnologias e da formação de um quadro de funcionários aptos para o seu uso [21]. No entanto a viabilidade técnica dessa tarefa perpassa a construção de uma arquitetura que seja tanto robusta quanto flexível [22, 23].

Dessa forma os sistemas legados tem se mostrado valiosos ao longo de décadas e contribuído para os resultados empresariais, apresentam sinais de exaustão, evidenciados pela elevação dos custos para manutenção ou incremento de novas funcionalidades, e pela perda de desempenho de processamento da aplicação [24].

Contudo, o cenário atual requer decisões e ferramentas específicas para que as empresas possam avançar e se adaptar às novas demandas. Há um consenso de que uma arquitetura em nuvem proporciona vantagens significativas, como escalabilidade e economia de recursos ociosos. No entanto para maximizar os benefícios desse ambiente, recomenda-se a implementação de uma arquitetura multi-cloud, que possibilite a escolha da melhor oferta de recursos de acordo com a disponibilidade de cada provedor [13, 22, 23].

Para tornar essa estratégia viável, é essencial desenvolver o design e a arquitetura da aplicação de forma a alcançar a compatibilidade do maior número possível de ***provedores*** mitigando o risco do ***Vendor Look-in*** e garantindo que a corporação possa escolher dentre as opções, sem ser restringida por decisões arquiteturais inadequadas que possam comprometer essa portabilidade. Assim, destaca-se o movimento visando a modernização de sistemas legados, incluindo a migração de grandes sistemas (Mainframe) para aplicações em nuvem, implementadas em Arquitetura de Microsserviços [25, 13, 26].

Observa-se características que aumentam o entusiasmo por essa transformação, como a escalabilidade horizontal e a maior liberdade na escolha entre provedores de computação em nuvem. Nesse contexto pode-se citar a estratégia da empresa de streaming de mídia Netflix, que utiliza os serviços dos provedores de Cloud AWS e Google para reduzir sua dependência de um único provedor, essa abordagem foca na recuperação de desastres e continuidade de negócios entre provedores aproveitando os recursos exclusivos de cada nuvem [27].

Apesar das vantagens observadas na utilização da arquitetura ***Cloud*** e ***Multi-Cloud***, a migração de aplicações legadas para essa arquitetura não é um processo trivial, na verdade tende a ser um processo com vários obstáculos. Neste contexto, a pesquisa focou em mapear os principais desafios enfrentados durante esse nesse processo de modernização a partir de uma Revisão Sistemática da Literatura, e observação e análise de um estudo de caso real na modernização e migração de um sistema legado essencial de uma grande ***Instituição Financeira***[1] para uma aplicação ***Multi-Cloud Native Architecture***. Esse sistema legado é um componente central no ecossistema de empréstimos dessa instituição, desenvolvido há mais de duas décadas, originalmente implantadas em Mainframe e escrito em linguagem COBOL com Banco de Dados DB2, que permanece em atividade com importância vital para a oferta e gestão de operações de crédito dessa instituição.

1.2 Questões de Pesquisa

Para atender à necessidade da Revisão Sistemática da Literatura (RSL), investigou-se a literatura visando identificar os desafios enfrentados por equipes na migração de aplicativos legados para uma arquitetura multi-cloud. Durante a etapa de planejamento, foi formulada a questão de pesquisa **RQ1** para guiar os esforços seguintes. O principal objetivo para a realização da RSL foi mapear o estado atual do conhecimento sobre o tema Multi-Cloud e encontrar oportunidades de pesquisa para questões não resolvidas sobre os desafios da migração no contexto da arquitetura Multi-Cloud.

RQ1 - ***Quais são os principais desafios que as equipes enfrentam ao implantar ou migrar aplicações no contexto da arquitetura multi-cloud?***

[1]Foram apresentadas observações com base em experiências reais de uma grande instituição financeira latino americana, doravante referida como ***Zetta Bank***. As informações disponibilizadas passaram por adaptações ou ajustes sempre que necessário, com o objetivo de proteger a confidencialidade e a privacidade dos dados corporativos. Essas modificações foram realizadas de forma criteriosa, de modo a não comprometer a integridade científica da pesquisa.

Esta fase da pesquisa identificou os objetivos, desafios e vantagens na adoção de uma arquitetura multi-cloud. Durante essa fase, houve um aprimoramento na compreensão dos riscos potenciais além do mapeamento dos desafios e as condições ideais para uma migração bem-sucedida. Ademais, motivações empíricas identificadas influenciaram a formulação da pergunta de pesquisa, destacando os obstáculos na migração de aplicações legadas. Portanto, a pergunta de pesquisa formulada serviu como guia para atingir eficazmente o objetivo principal.

Após a RSL, evidenciou-se que a categoria relacionada aos desafios da ***"ARQUITETURA CLOUD-NATIVE"*** foi a segunda mais proeminente. Esse fato motivou o direcionamento de novo esforço de pesquisa para esclarecer esse desafio, resultando na formulação da questão de pesquisa **RQ2**, com o intuito de guiar a segunda fase das investigações, buscando aprofundar a compreensão desse tema.

RQ2 - Como definir corretamente o padrão arquitetural para migrar uma aplicação legada para uma aplicação Cloud-Native de forma que ela mantenha a portabilidade e mitigue o Vendor Lock-in, habilitando a Arquitetura Multi-Cloud?

Para responder à RQ2, optou-se por explorar o tema por meio de um ***Estudo Etnográfico***, permitindo uma imersão profunda no processo de migração. Durante o estudo, foram empregadas técnicas de ***Domain Driven Design (DDD)*** e os estilos arquiteturais ***SAGA Orchestration Pattern e Hexagonal Architecture***, com o objetivo de criar uma aplicação escalável, resiliente e tolerante a falhas. A escolha por essa abordagem metodológica proporcionou uma visão detalhada e contextualizada dos desafios e soluções encontrados na prática de modernização de aplicações legadas.

1.3 Projeto de Pesquisa

O projeto de pesquisa integrou uma revisão sistemática da literatura e um estudo de caso etnográfico para proporcionar uma abordagem robusta na investigação de fenômenos complexos. Primeiramente, a revisão sistemática da literatura foi conduzida para identificar o estado atual do conhecimento sobre o tema específico de arquitetura Multi-Cloud. Esta fase contemplou uma busca minuciosa em bases de dados acadêmicas, seguida da análise das publicações relevantes para sintetizar o conhecimento existente, o objetivo foi mapear as principais descobertas, lacunas e debates na literatura, fornecendo uma base de conhecimento sólida para a próxima etapa da pesquisa.

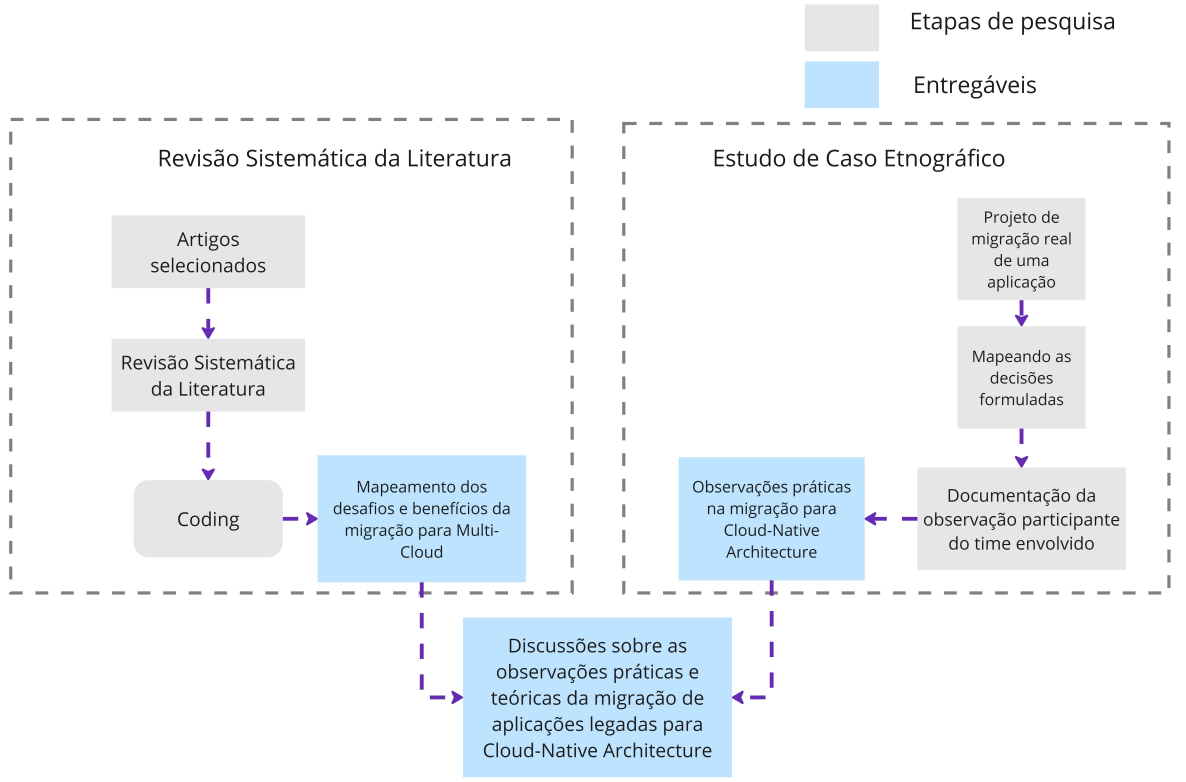


Figura 1.1: Processo de Pesquisa, combinação entre pesquisa teórica e estudo de caso prático.

Após a conclusão da revisão sistemática, a pesquisa avançou para a realização do estudo de caso etnográfico, que visou complementar e validar os dados e conhecimentos obtidos na revisão da literatura por meio da imersão profunda no contexto específico do caso estudado. Nessa etapa do processo o pesquisador assumiu o papel de observador participante, integrando-se à rotina do grupo ou comunidade alvo para observar diretamente os comportamentos, práticas culturais e interações sociais. Esta abordagem permitiu uma compreensão rica e detalhada dos fenômenos em questão, capturando nuances que não poderiam ser completamente apreciadas apenas pela revisão da literatura.

Este método de pesquisa proporcionou uma triangulação de dados que fortaleceu a validade e a confiabilidade das informações encontradas. A **revisão sistemática** ofereceu uma visão abrangente e teórica, enquanto o **estudo etnográfico** proporcionou ***insights*** empíricos e contextuais, a combinação desses métodos permitiu a compreensão mais aprofundada e integrada do tema investigado.

Capítulo 2

Fundamentação Teórica

Uma revisão da literatura foi conduzida para investigar a evolução da computação em nuvem, traçando sua trajetória desde os primórdios nos anos 2000 até as sofisticadas arquiteturas de microsserviços e multi-cloud que caracterizam o panorama atual. A computação em nuvem, com suas diversas modalidades (pública, privada, híbrida e multi-cloud), revolucionou a forma como as empresas gerenciam seus recursos de TI, oferecendo escalabilidade, flexibilidade e economia de custos. No entanto, a migração de aplicações legadas, especialmente aquelas com arquitetura monolítica, para ambientes de nuvem apresenta desafios significativos [28], [9].

Para superar esses desafios, abordagens como o Domain-Driven Design (DDD) e a combinação da arquitetura SAGA com a arquitetura hexagonal têm se mostrado valiosas. O DDD, ao enfatizar o alinhamento entre o software e o domínio de negócios, facilita a criação de microsserviços coesos e desacoplados, características que são ideais para ambientes de nuvem. Essa metodologia permite que os desenvolvedores se concentrem na lógica de negócios específica, promovendo uma arquitetura de software que reflete com precisão os processos e necessidades organizacionais. Além disso, a integração da arquitetura SAGA com a arquitetura hexagonal fornece um modelo robusto para gerenciar transações complexas e distribuídas, garantindo a consistência e a resiliência dos sistemas em ambientes altamente distribuídos [29], [30].

Este capítulo aprofundou a discussão sobre esses conceitos, explorando suas características, vantagens e desafios. Foram analisadas as particularidades de cada tipo de nuvem, os benefícios da arquitetura de microsserviços em relação à arquitetura monolítica, os princípios e práticas do DDD, e os mecanismos de funcionamento da arquitetura SAGA. Ao final, foi possível alcançar uma compreensão abrangente das tecnologias e abordagens que impulsionam a computação em nuvem moderna, bem como dos desafios e soluções para a migração e desenvolvimento de aplicações nesse ambiente.

2.1 Computação em Nuvem

Desde a introdução do conceito de "*computação em nuvem*", a compreensão dos serviços e recursos oferecidos evoluiu consideravelmente [4]. A computação em nuvem representa um paradigma destinado a solucionar desafios históricos relacionados à eficiência, eficácia e sustentabilidade nos investimentos em infraestruturas de informação e comunicação. Nesse contexto, a plataforma em nuvem disponibiliza recursos de computação, armazenamento e rede de forma virtualmente ilimitada [31]. É oferecida predominantemente através de modelos de entrega de serviços, como infraestrutura como serviço (IaaS), plataforma como serviço (PaaS) ou software como serviço (SaaS), permitindo a combinação ideal conforme as necessidades e as disponibilidades financeiras do cliente [31].

2.1.1 Nuvem privada

Uma nuvem privada é um serviço de computação fornecido pela Internet ou por uma rede interna privada, normalmente hospedado na infraestrutura interna da empresa [32]. Denominada nuvem interna ou corporativa, oferece diversos benefícios, incluindo controle e personalização aprimorados, obtidos por meio de recursos dedicados em uma infraestrutura de computação local. Possuem a capacidade de proporcionar maior segurança e privacidade [32]. Contudo, uma desvantagem é a responsabilidade do gerenciamento, manutenção e disponibilidade desse serviço recai sobre o departamento de TI da empresa, resultando em despesas de pessoal, administração e manutenção semelhantes às de um data center tradicional [33]. As nuvens privadas podem ser integradas a nuvens públicas para criar uma nuvens híbridas, permitindo que as empresas aproveitem os recursos da nuvem pública para liberar espaço e escalar serviços de computação conforme a necessidade [34] , [35].

2.1.2 Nuvem pública

Uma nuvem pública é uma configuração em que recursos, como capacidade de computação e armazenamento, são fornecidos por um provedor terceirizado pela Internet e compartilhados por organizações e indivíduos que desejam utilizá-los. Esses recursos podem ser obtidos de empresas como Microsoft, Google ou Amazon Web Services, alguns são gratuitos, enquanto outros podem ser adquiridos por meio de assinaturas ou modelos de pagamento conforme o uso [36], [11]. São oferecidos recursos e serviços como inteligência artificial, ferramentas para desenvolvedores, armazenamento e capacidade computacional para praticamente qualquer carga de trabalho.

A nuvem pública permitiu que as empresas utilizassem tecnologias avançadas e alcançassem uma escala global sem enfrentar os custos e o trabalho inicial normalmente associados às infraestruturas tradicionais. Ao contrário dos modelos de nuvem privada, em que os recursos eram restritos a uma única organização e administrados pelo fornecedor, tanto em data centers locais quanto externos, a nuvem pública ofereceu uma escalabilidade significativa. Esse modelo viabilizou o provisionamento de recursos de forma autônoma, o que possibilitou às organizações responderem de maneira eficiente às demandas variáveis de carga de trabalho e dos usuários. Para as organizações que buscavam alternativas às arquiteturas de TI tradicionais ou a outros tipos de computação em nuvem, a nuvem pública apresentou-se como uma solução viável, destacando-se por sua capacidade de adaptação e expansão conforme necessário [36].

No entanto, a adoção da infraestrutura de nuvem pública apresentou desafios significativos que precisaram ser superados. A migração de aplicativos existentes em arquiteturas *on-premises*, especialmente aqueles provenientes de arquiteturas legadas de mainframe, revelou-se bastante complexa. Ademais, à medida que mais recursos foram consumidos e os data centers se expandiram, os custos aumentaram de maneira significativa, impactando os orçamentos das organizações. Além do mais, surgiram preocupações relacionadas à segurança dos dados e das aplicações. Dessa forma, a adoção exclusiva da nuvem pública não se mostrou a solução predominante entre as organizações [37].

2.1.3 Nuvem Híbrida

Uma nuvem híbrida é caracterizada como uma combinação de nuvem pública e privada, conectadas, integrando um conjunto único de oferta de recursos e serviços [37], [38]. Essa combinação pode ser formada por uma nuvem privada interna de uma organização, com um ou mais provedores de nuvem pública, ou nuvem privada *on-premise* ou alocada em instalações terceirizadas em um ou mais provedores de nuvem pública [37], [38]. Essa composição promove a integração de serviços de nuvem pública, serviços de nuvem privada, além de infraestrutura local, oferecendo orquestração, gerenciamento e portabilidade de aplicativos entre esses ambientes [39]. A busca por um modelo adaptável e escalável impulsionou as empresas a adotar soluções que possibilitassem a alocação da infraestrutura de acordo com a demanda [37], [38].

2.1.4 Arquitetura Multi-Cloud

O aumento no número de fornecedores de nuvem impulsionou a disponibilidade de recursos e serviços, tornando a escolha do provedor um fator essencial para a otimização dos custos operacionais de uma organização. Essa questão suscitou diversas pesquisas

devido à importância de se compreender o processo de seleção de um provedor de nuvem adequado para a migração de aplicações, com o objetivo de alcançar custos reduzidos e alto desempenho. Dessa forma, se torna necessário, a definição uma estratégia criteriosa para a migração de aplicações entre diferentes provedores [11].

Para atender essa necessidade, surgiu um novo formato de arquitetura em nuvem: a ***Multi-Cloud***. A Multi-Cloud refere-se à utilização simultânea de dois ou mais provedores de computação em nuvem, possibilitando a implantação de aplicações em estruturas de nuvens públicas, privadas ou híbridas. As implantações em múltiplas nuvens visam fornecer maior redundância, melhor aproveitamento de desempenho, atender a restrições legais ou geopolíticas e evitar a dependência de um único fornecedor [35], [40].

Alguns autores consideram as implantações híbridas como uma forma de Multi-Cloud, onde um aplicativo é implantado tanto na infraestrutura local de uma nuvem privada quanto em plataformas de nuvens públicas. Contudo, outra abordagem encontrada na literatura refere Multi-Cloud como configurações que envolvem nuvem privada e nuvem pública entre vários provedores. [41].

2.2 Design da Arquitura Multi-Cloud

A utilização da arquitetura Multi-Cloud tende a contribuir para sucesso das empresas, porém há o desafio no sentido de iniciar uma estratégia Multi-Cloud mantendo as operações enxutas e a infraestrutura de TI consistente, criando plataformas de desenvolvimento que forneçam agilidade aos negócios, a implementação dessa estratégia exige um planejamento estratégico cuidadoso [42] [43].

Adotar a abordagem multi-cloud permiti a integração de serviços específicos de diferentes provedores, alinhando funcionalidades às estratégias de negócios. Essa prática oferece flexibilidade e promove a inovação, destacando-se como uma solução eficiente frente à dependência de um único provedor [42].

Essa abordagem destaca-se como uma solução eficaz para otimizar infraestruturas de TI e aprimorar a entrega de serviços, ao distribuir aplicativos e dados entre diferentes provedores, dessa forma as organizações tem a possibilidade de reduzir a dependência de um único fornecedor, evitando o **vendor lock-in** e fortalecendo sua capacidade de negociação [44, 45, 46]. Assim, a redundância inerente aos ambientes multi-cloud contribui para a manutenção da disponibilidade e do desempenho dos serviços diante de interrupções inesperadas [47].

Outro desafio observado é o incremento de ineficiências e sobrecargas oriundos da complexidade operacional do ambiente heterogêneo [47, 48]. A integração desses ambientes demanda ferramentas de orquestração avançadas para assegurar a interoperabilidade e a

gestão eficaz das cargas de trabalho [49, 50]. Mesmo assim, a segurança emerge como uma preocupação importante devido à distribuição de dados em múltiplas plataformas, podendo fragilizar a conformidade regulatória e elevar a exposição a ameaças cibernéticas [46, 51]. Para mitigar esses riscos, torna-se necessário implementar uma estrutura de governança, capaz de aplicar políticas e procedimentos uniformemente em todas as plataformas [48, 52].

A arquitetura multi-cloud, ilustrada na Figura 2.1, é configurada por meio de clusters Kubernetes distribuídos entre diferentes provedores, formando um ambiente de multi-cluster Kubernetes que integra a infraestrutura de cloud computing da organização [42, 16, 43]. A preferência pelo uso de ferramentas open source aderentes aos padrões da **CNCF** [53] como as descritas na tabela 2.1 contribui para a portabilidade e evita o aprisionamento a um fornecedor específico (vendor lock-in) [42, 16, 43].

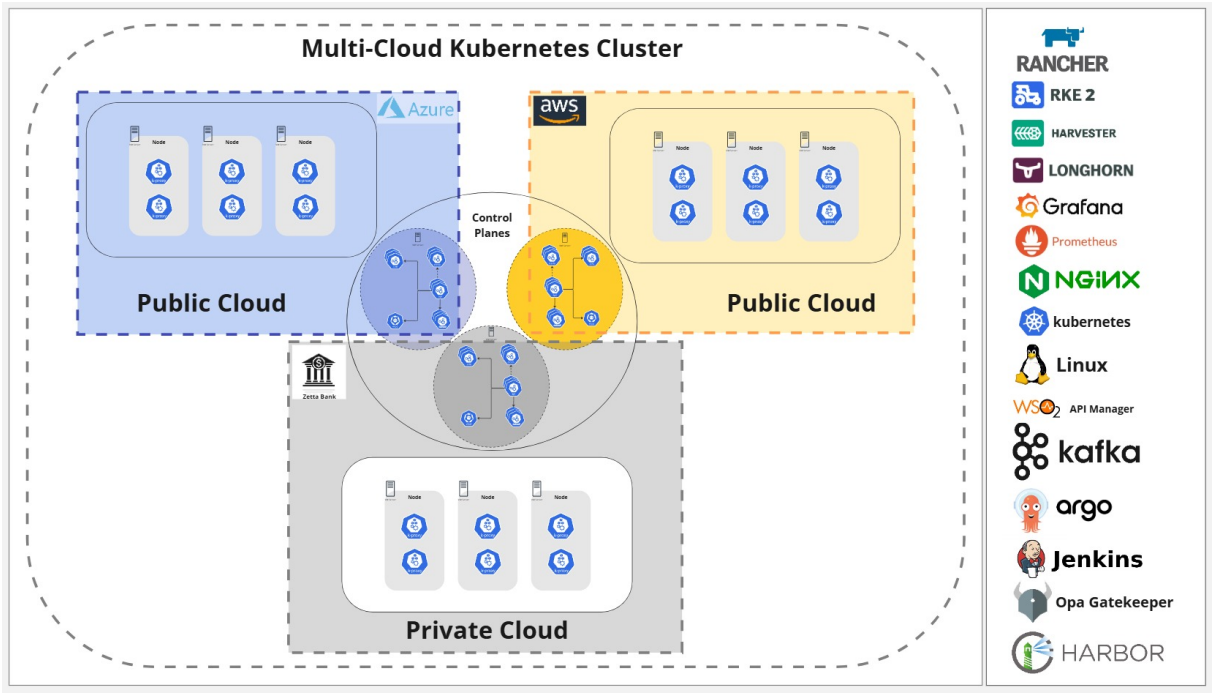


Figura 2.1: A imagem demostra uma arquitetura *Multi-Cloud* através da implantação de um *Multi-Cluster Kubernetes*, composta por duas nuvens públicas, *Azure* e *AWS* respectivamente, integradas com um terceiro *cluster* instalado em nuvem privada, com três replicas de *Controls Planes*, uma replica implantada em cada nuvem do *cluster* para manutenção da alta disponibilidade, e ao lado direito, sugestões de ferramentas *opensource* aderentes ao padrão *CNCF* que complementam as funcionalidades necessárias para implantação de uma arquitetura *Multi-Cloud Opensource* mitigando o risco do *Vendor Lock-in*.

2.2.1 Objetivo e Estratégia para Arquitetura Multi-Cloud

A implementação de uma estratégia de Kubernetes em Multi-Cloud exige a preparação do ambiente e da infraestrutura para suportar a implantação em múltiplos provedores. Essa etapa engloba a configuração de recursos como rede, segurança e armazenamento, garantindo a continuidade das operações em diferentes nuvens. O ***Cluster Kubernetes em Multi-Cloud*** visa a criação de um sistema unificado que explore os benefícios de múltiplas nuvens, evitando o vendor lock-in e promovendo flexibilidade, escalabilidade e resiliência. A preparação adequada do ambiente e da infraestrutura constitui a base para uma implantação bem-sucedida [43].

A seguir estão descritas algumas etapas necessárias para preparar o cenário para a realização uma estratégia de implantação de uma Arquitetura Multi-Cloud:

- **Definição dos Objetivos:** A reflexão sobre os objetivos que utilização da estratégia multi-cloud, é o ponto inicial do trabalho que define a direção dos próximos passos. Objetivos observados como por exemplo a necessidade de **performance**, **disponibilidade**, **escalabilidade** ou **otimização de custos**, influenciam na configuração e implantação do ***Multi-Cloud Kubernetes Cluster*** [43, 42]

- **Definição dos Requisitos**: A implementação de uma arquitetura Kubernetes multi-nuvem exige a definição prévia de requisitos específicos, incluindo a análise da carga de trabalho, restrições orçamentárias e o grau de controle desejado sobre o ambiente. Essa etapa visa garantir uma transição eficiente e um desempenho otimizado, além de auxiliar na alocação adequada de recursos e na seleção de provedores de serviços compatíveis com as necessidades da organização. A compreensão do orçamento e do nível de controle desejado previne custos imprevistos e garante a supervisão adequada da infraestrutura [43].

- **Seleção de provedores**: A seleção de provedores de nuvem deve ser conduzida com base em critérios objetivos que considerem demandas de carga de trabalho, restrições orçamentárias e as características dos serviços oferecidos. A adequação de cada provedor às necessidades específicas do sistema deve ser avaliada para assegurar a escolha mais alinhada aos requisitos técnicos e financeiros do projeto. Assim, o processo deve ser estruturado e fundamentado, priorizando a compatibilidade entre as exigências do sistema e as ofertas disponíveis no mercado [43], [54], [55], [56].

- **Seleção da Distribuição Kubernetes**: A escolha da distribuição Kubernetes adequada para um ambiente multi-cloud deve ser baseada em critérios técnicos como carga de trabalho, e grau de controle desejado sobre o ambiente e a capacidade da distribuição de operar em diferentes provedores de nuvem. A compatibilidade com

as necessidades específicas de cada organização, como segurança e conformidade, também deve ser considerada. Distribuições como o Kubernetes RKE2, mantido pela Rancher [57], que se concentra nos requisitos de segurança e conformidade dentro do setor do Governo Federal dos EUA, pode ser opções interessantes para ambientes com elevado grau de segunça e confiabilidade [43].

2.2.2 Plano de Implantação do Multi-Cloud Kubernetes Cluster

Definidos os objetivos, requisitos, distribuição Kubernetes e provedores de nuvem, a etapa subsequente consiste na elaboração de um plano de implantação para a configuração do cluster Kubernetes multi-cloud, a figura 2.1 ilustra um exemplo de um Multi-Cloud Cluster composto por duas nuvens públicas e uma privada. Esse plano deve abranger as etapas necessárias para a configuração da infraestrutura, a instalação e configuração do Kubernetes nos diferentes provedores, a integração entre os clusters e a implantação das aplicações [42, 43, 20].

- **Provisionar recursos na Nuvem**: com os provedores e a distribuição kubernetes selecionados, o próximo passo é planejar a infraestrutura, incluindo provisionar os recursos computacionais, armazenamento e rede necessários para executar o ambiente kubernetes em cada provedor [43].
- **Instalar e Configurar o Cluster Kubernetes**: o próximo passo é instalar e configurar o Kubernetes em cada provedor de nuvem. Isso normalmente envolve usar ferramentas como **Rancher** [57], **kubeadm** [58] ou **kops** [59] para configurar o cluster. O Cluster deve ser configurado para funcionar em vários provedores de nuvem [43].
- **Instalar e Configurar as ferramentas de gerenciamento Multi-Cloud**: a configuração de ferramentas de gerenciamento multi-cloud é essencial após a implantação dos clusters Kubernetes. Ferramentas como **Istio** [60], **KubeDirector** [61], **Ansible** [62] e **Terraform** [63] permitem o gerenciamento e a orquestração eficientes do ambiente multi-cloud, proporcionando escalabilidade, segurança e automação. A utilização dessas ferramentas habilita o controle e a otimização do ambiente Kubernetes, independentemente dos provedores de nuvem utilizados [43].
- **Implantação e Gerenciamento das Aplicações**: a implantação e o gerenciamento de aplicações no ambiente Kubernetes multi-cloud constituem a etapa final do processo, através da criação de manifestos Kubernetes, que definem os recursos necessários para a execução das aplicações, como *pods*, *services* e *deployments*.

Ferramentas como o **Helm** [64] podem auxiliar no gerenciamento dessa implantação, assim o planejamento e a execução criteriosa são essenciais para a construção de um **ambiente Kubernetes multi-cloud escalável, flexível e adaptado** às necessidades de carga de trabalho e orçamento da organização [43].

Observabilidade e Monitoração na Arquitetura Multi-Cloud

Manter a saúde e a disponibilidade em um ambiente multi-cloud requer vigilância constante e é imprescindível que uma estrutura adequada de observabilidade e monitoração seja implementada. Nesse cenário, onde as organizações utilizam serviços de múltiplos provedores de nuvem (como **AWS**, **Azure**, **GCP**), a observabilidade emerge como um pilar fundamental para garantir a saúde, o desempenho e a confiabilidade das aplicações e infraestruturas distribuídas. Diferente do monitoramento tradicional, que foca em métricas predefinidas, ***a observabilidade busca entender o motivo ou a causa raiz*** de um determinado comportamento do sistema, permitindo investigar questões desconhecidas e emergentes. Em um ambiente multi-cloud, a complexidade inerente à integração de diferentes plataformas, tecnologias e modelos de serviço torna a observabilidade ainda mais crítica. A capacidade de correlacionar dados de diversas fontes, desde logs e métricas até traces de requisições, é essencial para obter uma visão unificada e contextualizada do ambiente como um todo [43, 65].

Para implementar a observabilidade de forma eficaz em um ambiente multi-cloud, certas ferramentas tornam-se essenciais. Dessa forma, plataformas de observabilidade unificadas, como **Datadog**, **New Relic** e **Dynatrace**, **Prometheus** e **Grafana** oferecem a capacidade de coletar, analisar e visualizar dados de telemetria de múltiplos provedores de nuvem em um único painel. Outras ferramentas de coleta de dados agnósticas a provedores, como o **OpenTelemetry**, facilitam a instrumentação de aplicações e a padronização da coleta de logs, métricas e traces. Soluções de gerenciamento de logs centralizados, como o **ELK Stack (Elasticsearch, Logstash, Kibana)** ou **Splunk**, são cruciais para agregar e analisar grandes volumes de logs gerados em nuvens heterogêneas. Além disso, ferramentas de monitoramento de infraestrutura específicas de cada provedor (como **AWS CloudWatch**, **Azure Monitor** e **Google Cloud Monitoring**) também são importantes, desde que estejam integradas a uma visão centralizada [65, 66].

Os benefícios de observabilidade bem elaborada em uma arquitetura multi-cloud são vastos e impactam diretamente a eficiência operacional e a qualidade dos serviços. A observabilidade permite uma detecção e resolução de problemas mais rápidas, reduzindo o tempo de inatividade e o impacto em usuários e negócios. Proporciona *insights* profundos sobre o desempenho das aplicações e da infraestrutura em diferentes nuvens, possibilitando a otimização de recursos e a identificação de gargalos e pontos de atrito. Além da contri-

buição com o fortalecimento da segurança, permitindo a identificação de comportamentos anormais e potenciais ameaças em todo o ambiente distribuído. Em última análise, ***a observabilidade é uma necessidade obrigatória em arquiteturas multi-cloud***, pois a falta de visibilidade integrada pode levar a silos de informação, dificuldades na resolução de problemas, ineficiência de recursos e riscos de segurança aumentados, comprometendo a agilidade e a resiliência do negócio [67].

CNCF Software	Descrição
RANCHER	Rancher é uma plataforma de gerenciamento de contêineres que facilita a implantação, operação e escalabilidade de clusters Kubernetes em ambientes multi-cloud ou locais [57].
RKE2	RKE2 (Rancher Kubernetes Engine 2) é uma distribuição de Kubernetes otimizada para segurança, desenvolvida pela Rancher, projetada para atender a ambientes corporativos, com suporte a workloads críticos e conformidade com padrões como FIPS [68].
HARVESTER	Harvester é uma solução moderna de infraestrutura hiperconvergente (HCI) criada para servidores bare metal usando tecnologias de código aberto de nível empresarial, incluindo Linux, KVM, Kubernetes, KubeVirt e Longhorn [69].
LONGHORN	Longhorn é um sistema de armazenamento distribuído de código aberto para Kubernetes, projetado para oferecer volumes persistentes altamente disponíveis e resilientes, simplificando a gestão de armazenamento em clusters [70].
Prometheus	Prometheus é uma ferramenta de monitoramento de código aberto que coleta, armazena e analisa métricas de sistemas e aplicações em tempo real, utilizando um modelo baseado em séries temporais e uma linguagem de consulta poderosa [71, 72].
Continua na próxima página...	

CNCF Software	Descrição
	Grafana é uma plataforma de código aberto para visualização, análise e monitoramento de dados, que permite criar dashboards interativos e integrados com diversas fontes, como Prometheus, Elasticsearch e bancos de dados SQL [73, 72].
	NGINX é um servidor web de código aberto que atua como proxy reverso, balanceador de carga e gateway HTTP, projetado para alta performance, escalabilidade e eficiência no gerenciamento de conexões [74].
	Kubernetes é uma plataforma open-source que automatiza a implantação, o escalonamento e o gerenciamento de aplicações em contêineres [75].
	Linux é um sistema operacional de código aberto que serve como base para diversos sistemas, de servidores a dispositivos móveis, conhecido por sua flexibilidade e estabilidade [76].
	WSO2 API Manager é uma plataforma open-source que ajuda as empresas a gerenciar todo o ciclo de vida de suas APIs, desde a criação até a publicação e monitoramento, permitindo que elas compartilhem seus dados e serviços de forma segura e eficiente [77].
	Kafka é uma plataforma de streaming de eventos distribuída, de alta performance e tolerante a falhas, usada para publicar, assinar, armazenar e processar fluxos de dados em tempo real [78].
	Argo CD é uma ferramenta de entrega contínua para Kubernetes que usa GitOps para automatizar a implantação de aplicações, garantindo que o estado desejado do cluster corresponda ao definido no repositório Git [79].
Continua na próxima página...	

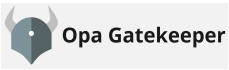
CNCF Software	Descrição
	Jenkins é uma ferramenta de automação de código aberto que ajuda a automatizar as etapas do processo de desenvolvimento de software, como compilação, teste e implantação, tornando mais fácil para os desenvolvedores integrar alterações de código e entregar software com mais rapidez [80].
	O OPA Gatekeeper é uma ferramenta que aplica políticas no Kubernetes usando Open Policy Agent (OPA), permitindo definir ou impor regras para controlar como os recursos são implantados e configurados no cluster kubernetes [81, 82].
	NeuVector é uma plataforma de segurança de contêineres que ajuda a proteger suas aplicações e infraestrutura em ambientes nativos da nuvem. Essa ferramenta trabalha como um "guarda-costas"para os contêineres, monitorando constantemente o ambiente, detectando ameaças e as neutralizando antes que causem danos. [81, 82].
	Harbor é uma plataforma open-source que garante a segurança das imagens de contêiner, oferecendo recursos como registro de imagens, varredura de vulnerabilidades, controle de acesso e gerenciamento de identidade [83].
	O Apache Cassandra é um banco de dados NoSQL distribuído, tolerante a falhas e altamente escalável, projetado para lidar com grandes volumes de dados em múltiplos servidores [84].

Tabela 2.1: Descrição de algumas ferramentas Open-source aderentes às CNCF, que complementam a implantação da Arquitetura Multi-Cloud, com Multi-Cluster Kubernetes, suprindo necessidades como segurança, disponibilidade, gerenciamento, etc...

2.3 Arquitetura da Aplicação Multi-Cloud Native

As arquiteturas de software, em constante evolução, têm apresentado diferentes paradigmas para o desenvolvimento de aplicações [8]. O modelo monolítico, tradicionalmente utilizado em softwares empresariais, caracteriza-se por agregar todas as funcionalidades em um único módulo, o que pode resultar em alto acoplamento e dificuldades de escalabilidade e manutenção [8]. A crescente complexidade das aplicações e a necessidade de maior flexibilidade impulsionaram a busca por novas abordagens arquitetônicas [8].

Nesse contexto, a arquitetura de microsserviços emerge como uma alternativa promissora para superar as limitações dos sistemas monolíticos [41]. Baseada na decomposição da aplicação em serviços independentes e coesos, essa arquitetura oferece maior escalabilidade, flexibilidade e facilidade de manutenção [41]. Empresas como Netflix, Spotify e Uber têm adotado microsserviços com sucesso, demonstrando a eficácia dessa abordagem na construção de aplicações complexas e nativas em nuvem [41].

Arquitetura Monolítica

Arquiteturas monolíticas são frequentemente encontradas em softwares empresariais e apresentam características como alto acoplamento, baixa escalabilidade, complexidade de monitoramento e custos de manutenção elevados quando comparadas a microsserviços implementados em nuvem [41]. Nesse modelo arquitetural, o software é tipicamente construído em um único módulo, agregando diversas funcionalidades e responsabilidades, e qualquer manutenção exige a reimplantação de todo o pacote, mesmo para alterações mínimas. Tal característica aumenta o risco, pois qualquer mudança pode impactar todo o sistema [85]. A modernização dessas aplicações demanda tarefas complexas, incluindo refatoração e redesenho da arquitetura para um modelo nativo da nuvem. Além disso, a implantação em ambientes multi-cloud pode trazer benefícios como maior escalabilidade, alta disponibilidade e capacidade de manutenção, além da redução de custos. No entanto, a reengenharia da aplicação é um pré-requisito para viabilizar a migração e alcançar os benefícios da computação em nuvem [41].

Assim, as aplicações legadas em geral, não foram projetadas para aproveitar os recursos de aumento/redução dinâmicos de capacidade, comumente denominados escalabilidade vertical e horizontal. Frequentemente, essas aplicações dependem de elasticidade estática, suportada pelo provisionamento de servidores físicos mais robustos [8].

No entanto, sistemas com arquitetura monolítica impõem dificuldades significativas na migração para plataformas em nuvem e novas tecnologias, devido a limitações de escalabilidade, flexibilidade, desempenho e menor valor comercial [86]. A reengenharia

dessas aplicações para arquiteturas de computação em nuvem é um processo complexo, demandando diversas alterações nas camadas da aplicação [8].

Arquitetura de Microsserviços

A arquitetura de microsserviços apresenta-se como um estilo emergente de arquitetura de software para superar limitações arquiteturais monolíticas [86]. Os aplicativos que seguem o padrão de microsserviços consistem em serviços altamente especializados, pouco acoplados e coesos que trabalham juntos para fornecer determinadas funcionalidades. Comparado à arquitetura monolítica, cada serviço oferece sua lógica para uma pequena parte da aplicação, que de forma agregada compõem o sistema em sua totalidade[85] [23] [24].

Dessa forma, observa-se que empresas como a *Netflix*, *Spotify* e *Uber* utilizam arquitetura de microsserviços, implementando aplicações nativas em nuvem como base para infraestruturas complexas com sucesso [24]. O desenvolvimento de aplicativos para a nuvem torna-se mais modular, interoperável e distributivo, e tecnologias novas e em evolução, como contêineres *Docker*, microsserviços e arquitetura sem servidor, visam atingir esse objetivo e alcançar o resultado [87].

As arquiteturas de microsserviços envolvem colaborações entre diferentes módulos implantáveis de forma independente, muitas vezes sem um controlador centralizado, para alcançar a funcionalidade geral esperada do sistema [5].

Como os microsserviços podem ser lançados individualmente, o tempo necessário para corrigir *bugs* ou adicionar novos torna-se menor permitindo que as alterações sejam implantadas na produção com mais eficiência [88].

2.3.1 Projeto de uma Aplicação Multi-Cloud

O desenho arquitetural de aplicações com abordagem multi-cloud exige foco em resiliência, desempenho e alinhamento aos requisitos de negócio. Elementos como os princípios do 12-factor app, o uso de PaaS para aceleração, o design de soluções SaaS e a definição de KPIs de performance são essenciais. Além do mais, a otimização de ambientes multi-cloud e estratégias de recuperação de desastres complementam a abordagem para a criação de soluções de negócios contínuas e flexíveis [42]. O respeito a essas práticas colaboram para a eficácia de aplicações em ambientes multi-cloud, alinhando soluções tecnológicas às demandas empresariais [42].

1. Arquitetura para resiliência e performance

Uma arquitetura quando projetada para multi-cloud caracteriza-se pela capacidade de recuperação sob condições de alta carga, ataques e falhas em qualquer componente operacional. Avalia-se essa resiliência pela velocidade e pela abrangência da capacidade do sistema em retornar ao estado original sem perda de dados. Para evidenciar essas características, utiliza-se testes específicos, como o uso do *Chaos Monkey*, uma estratégia estruturada empregada para testar a robustez do sistema frente a falhas induzidas e verificar sua capacidade de recuperação [42].

Definir a resiliência dos sistemas inicia-se pela análise da relevância desses sistemas para o negócio e pelos níveis de serviço requeridos. Esse processo fundamenta-se em uma compreensão detalhada das necessidades empresariais. Ressalta-se que resiliência e desempenho não constituem apenas questões técnicas; esses aspectos englobam também a responsabilidade operacional, representando um compromisso compartilhado entre todas as equipes envolvidas na criação e gestão de aplicativos [42].

2. Requisitos de Negócio

A gestão da infraestrutura em ambientes multi-cloud exige o uso de estruturas arquiteturais bem definidas, incluindo aspectos como disponibilidade, backup e recuperação de desastres. Examina-se detalhadamente os requisitos e as soluções oferecidas pelas plataformas de nuvem para assegurar que os aplicativos permaneçam disponíveis, acessíveis e, sobretudo, seguros para uso. Antes de explorar essas soluções e tecnologias, torna-se essencial compreender os potenciais riscos empresariais associados à ausência de requisitos claramente definidos. Em um ambiente multi-cloud, esses riscos manifestam-se em múltiplos níveis, alinhando-se aos princípios fundamentais da arquitetura corporativa [42].

- Entendendo os Riscos com Dados

A gestão de riscos de dados exige a definição clara da propriedade dos dados, orientando a formalização em contratos com provedores de nuvem, em conformidade com leis como o GDPR na Europa. Diferentes tipos de dados, como comerciais, metadados e dados operacionais, requerem conformidade com regulamentações específicas [42].

Torna-se importante a localização exata dos dados, considerando que a otimização global de provedores como ***Azure***, ***AWS*** e ***GCP*** pode envolver movimentações de dados entre regiões, representando um risco para países que restringem a saída de dados de suas jurisdições. Recentemente, os EUA implementaram legislações para proteger dados de cidadãos europeus, garantindo supervisão independente para avaliar o uso adequado dessas informações por empresas americanas [42].

Dessa forma, evidencia-se a necessidade da integridade e legibilidade dos dados em processos de recuperação e durante a execução de transações, evitando duplicações. Para isso, testes regulares podem ser executados para garantir a qualidade dos dados, em especial em serviços SaaS [42].

Assim, a criação de um modelo de classificação de dados pode ser recomendado para garantir os níveis de confidencialidade, integridade e disponibilidade necessários para cada tipo de dado, com categorias como dados públicos, confidenciais corporativos e pessoais [42].

- Entendendo o Risco da Aplicação

O uso de SaaS tem se tornado comum, com empresas adotando-o preferencialmente em relação a PaaS e IaaS devido à sua facilidade operacional. No entanto, o SaaS envolve riscos específicos, pois toda a pilha, incluindo o próprio aplicativo, é gerida pelo provedor. Isso gera preocupações de segurança, como o acesso não autorizado a dados por meio de componentes compartilhados [42].

Outra vulnerabilidade observada é à sustentabilidade do provedor. Em crises como a pandemia de coronavírus, provedores menores enfrentaram dificuldades financeiras que os levem a descontinuar serviços. A continuidade e a proteção dos dados são essenciais caso o provedor suspenda a operação ou encerre o negócio [42].

- Entendendo os Riscos Tecnológicos

Ao configurar ambientes em plataformas de nuvem com componentes compartilhados, como data centers e camadas de armazenamento, computação e rede, o isolamento do ambiente cria uma área virtual separada, mas ainda dependente da infraestrutura subjacente de provedores como Azure, AWS e GCP. Cabe aos provedores garantir a disponibilidade desses ambientes, adotando estratégias de redundância que envolvem múltiplos data centers, zonas ou regiões globais [42].

Assim, a observabilidade e o monitoramento são essenciais, mas requerem configuração cuidadosa para evitar riscos associados a monitoramentos inadequados. Embora as plataformas forneçam ferramentas para essa tarefa, a responsabilidade pela configuração adequada é da empresa [42].

Em termos de segurança, embora as plataformas de nuvem pública ofereçam proteção robusta, a responsabilidade pela segurança do ambiente específico do cliente permanece com a empresa. A proteção eficaz envolve medidas como proteção de endpoint, segmen-

tação de rede, firewalls, varredura de vulnerabilidades e monitoramento contínuo contra intrusões [42].

A definição dos requisitos de segurança e proteção deve ser clara e gerenciável, equilibrando o que deve ser protegido com os custos associados. Os requisitos empresariais orientam o nível de proteção necessário para dados, aplicativos e infraestrutura, além de moldarem a arquitetura e políticas para ambientes multi-cloud. Esse processo de coleta de requisitos não é único, mas contínuo, adaptando-se à evolução das demandas e à velocidade das mudanças tecnológicas na era da nuvem [42].

3. Princípios 12-factor App

A metodologia de 12 fatores ***(12-factor)*** propõe diretrizes para o desenvolvimento de aplicativos modernos, preparados para implantação em nuvem e abstraídos da infraestrutura de servidor. Essa abordagem permite que os desenvolvedores construam aplicativos escaláveis e confiáveis, com dependências claramente definidas e isoladas da infraestrutura subjacente [89, 90, 91].

Além disso, o modelo de 12 fatores oferece uma gestão consistente de código por meio de uma base única e um processo de implantação definido e declarativo. Fundamentado nos livros de *Martin Fowler* [90] e [91], esses princípios mantêm sua relevância no desenvolvimento nativo de nuvem, sendo amplamente adotados para estruturar arquiteturas robustas e adaptáveis [89, 90, 91].

Os doze fatores (12-factor) são os seguintes [89]:

1. **Base de Código**: Uma base de código é rastreada no controle de revisão, com muitas implantações (incluindo Infraestrutura como Código) [89].
2. **Dependências**: Declare e isole explicitamente as dependências [89].
3. **Configurações**: Armazene a configuração no ambiente (Configuração como Código) [89].
4. **Serviços de apoio**: Trate serviços de apoio como recursos interligados [89]. .
5. **Construa, Lance, Execute**: Separe rigorosamente os estágios de construção e execução (gerenciamento de *pipeline*, trens de liberação) [89].
6. **Processos**: Execute o aplicativo como um ou mais processos sem estado armazenado [89].
7. **Vínculo de Porta**: Exportar serviços via vinculação de portas [89].
8. **Concorrência**: Escale horizontalmente por meio do modelo de processo [89].

9. **Descartabilidade**: Maximize a robustez com inicialização e desligamento rápido [89].
10. **Dev/prod semelhantes**: Mantenha o desenvolvimento, a preparação e a produção o mais semelhantes possível [89].
11. **Logs**: Trate os logs como fluxos de eventos [89].
12. **Processos de Admin**: Execute tarefas administrativas ou de gerenciamento como processos únicos [89].

2.3.2 Domain-Driven Design

O Domain-Driven Design (DDD), é uma abordagem de desenvolvimento de software centrada no domínio que reúne padrões, princípios e práticas, sendo particularmente útil na criação de arquiteturas de microsserviços [92].

A principal vantagem do DDD reside na capacidade de alinhar o código de produção com o domínio de negócios, auxiliando no gerenciamento da complexidade inerente ao domínio e na escrita de código mais sustentável e extensível. Esse enfoque permite a adição de novos recursos com menor esforço e promove uma arquitetura de alta coesão e baixo acoplamento [93], [94], [92]. Assim, literatura recomenda a utilização do DDD em domínios complexos, como e-commerce, sistemas fiscais e financeiros, mas não o considera particularmente útil para aplicações simples [92].

Ao percorrer o ***Domain-Driven Design (DDD)***, diversos passos devem ser executados, com modelos e elementos chaves para o domínio devidamente mapeados, como o modelo estratégico e o modelo tático, além de outros elementos essenciais para o domínio descritos a seguir [95], [30], [96]:

- **Event Storming**: técnica amplamente utilizada nas seções de modelagem do DDD, que auxilia na compreensão e representação visual dos eventos e processos de negócio [96].
- **Linguagem Ubíqua**: a utilização de termos e conceitos compreendidos por toda a equipe é essencial para evitar mal-entendidos na modelagem. No DDD valoriza-se o cuidado com a linguagem ubíqua para garantir uma comunicação nítida e abrangente entre os membros da equipe [94].
- **Eventos**: representam momentos decisivos em que ocorrem mudanças de estado significativas no sistema, refletindo as ações e interações que impulsionam o domínio do negócio [97].

- **Comandos**: representam ações realizadas em momentos específicos do negócio, desencadeando mudanças de estado e gerando eventos [97].
- **Agregados**: são agrupamentos lógicos de elementos importantes para uma determinada seção ou assunto. Um agregado recebe comandos, processa informações e produz eventos, encapsulando a lógica de negócio relacionada [30].
- **Políticas**: constituem decisões e regras impostas pelas condições do negócio, que podem alterar o fluxo de execução do processo ou sistema, influenciando o comportamento e as respostas do domínio [30].
- **Sistemas Externos**: referem-se as integrações com sistemas ou entidades externas, permitindo a troca de informações e a interação com outros componentes do ecossistema de software [30].

Quando os tópicos são suficientemente independentes, assemelhando-se a subsistemas dentro de um sistema maior, são estabelecidas fronteiras de domínios (*Bounded Context*) que se relacionam entre si. Essas relações entre fronteiras que são denominadas de mapa de contexto (*Context Map*), são compostas por domínios e subdomínios identificados que são classificados conforme sua relevância, complexidade e impacto [30], [92]:

- **Domínio Principal**: Tipicamente construído pela própria empresa devido à sua relação direta com o valor competitivo do negócio. Caracteriza-se por alta complexidade e alta volatilidade, demandando atenção e desenvolvimento interno para garantir a diferenciação competitiva [95].
- **Domínio Genérico**: Geralmente, são domínios que apoiam o negócio, mas não são determinantes para sua sobrevivência. Podem ser adquiridos de fornecedores externos, por meio de compra ou aluguel. Apresentam baixa volatilidade, mas podem possuir alta complexidade, tornando a aquisição externa uma opção viável [95].
- **Domínio de Suporte**: Desempenha um papel importante no suporte ao negócio, mas não é crucial para a vantagem competitiva da empresa. Pode ser adquirido de fornecedores externos ou desenvolvido internamente. Caracteriza-se por baixa complexidade e baixa volatilidade, oferecendo flexibilidade na escolha da estratégia de implementação [95].

É importante ressaltar que o DDD representa o do estado atual do negócio, incorporando elementos essenciais que orientam a construção do sistema de forma a apoiar o negócio da maneira mais eficaz possível [92]. No entanto, o negócio é dinâmico e pode

sofrer alterações a qualquer momento, seja para atender a novas demandas de mercado ou para refletir avanços na compreensão de um domínio de negócio ao longo do tempo [95]. Dessa forma, o DDD deve ser periodicamente revisado e atualizado para assegurar que continue refletindo com precisão os domínios de negócio. Seria equivocado presumir que os elementos modelados não necessitam de atualizações contínuas, visto que a natureza dinâmica do negócio exige adaptações e refinamentos constantes no modelo [30], [92], [95].

2.3.3 SAGA Orchestration Pattern

SAGA é uma arquitetura algorítmica concebida para orquestrar múltiplas transições de estado. Inicialmente projetada para gerenciar *Long Live Transactions (LLTs)*, cada transação deve ser executada de forma autônoma, podendo variar de um período breve ou longo para sua conclusão. É indicado fragmentar as LLTs em transações menores, desmembrando um único processo de negócios em um conjunto de chamadas atendidas por serviços colaborativos [98] ,[29].

Uma característica importante, é que uma arquitetura distribuída não implementa o gerenciamento de transações da mesma forma que observa-se em banco de dados, o chamado ACID (*Atomicidade, Consistência, Isolamento, Durabilidade*), assim o gerenciamento de transações em um sistema distribuído, é uma funcionalidade complexa de implementar, mas necessária. O SAGA Pattern resolve essa questão com gerenciamento de eventos de transações e ações compensatórias, utilizadas para desfazer processamentos anteriores [98], [29].

Os processos de recuperação que foram desenhados na arquitetura SAGA, para garantir a confiabilidade dos dados e recuperação em eventual falha, são implementados de duas forma, a recuperação retroativa (backward recovery) e a recuperação proativa (forward recovery) [98]:

- **backward recovery** envolve reverter através de ações compensatórias.
- **forward recovery** envolve seguir adiante com o processamento, fazendo novas tentativas de execução.

Pode ser descrito como uma abordagem para o gerenciamento de transações em sistemas distribuídos, em que cada transação é dividida em uma série de passos menores e compensáveis. Cada passo é uma ação independente passível de reversão por uma ação compensatória correspondente. Essa arquitetura é particularmente relevante em sistemas que podem estar submetidos à ocorrência de falhas e instabilidades, onde é necessário manter a resiliência e a confiabilidade das informações, independentemente das flutuações de estabilidade do ambiente [29].

Essa arquitetura sobressai pela capacidade de aprimorar a resiliência do sistema em um ambiente distribuído, onde falhas podem ocorrer em qualquer ponto da comunicação entre os serviços. Essas falhas são administradas, de forma a permitir que cada serviço execute suas operações de maneira autônoma e registre pontos de compensação, possibilitando a reversão total ou parcial de quaisquer ações realizadas. Esse aspecto é importante para sistemas que necessitam manter alta disponibilidade, minimizando os impactos decorrentes de falhas ou indisponibilidades. Ademais, um benefício expressivo do SAGA é a melhora na escalabilidade, uma vez que a fragmentação de transações em partes menores permite que cada serviço seja escalado de maneira independente. Isso é particularmente vantajoso em arquiteturas de microsserviços, nas quais cada serviço pode ser dimensionado conforme a demanda específica, sem a necessidade de escalar toda a aplicação, como ocorre em uma aplicação monolítica. Além disso observa-se incremento da flexibilidade na implantação e atualização dos componentes do sistema, contribuindo para uma melhor manutenção e evolução contínua da aplicação [99].

Embora a implementação do SAGA ofereça vantagens, ela impõe desafios significativos, como a complexidade adicional na orquestração da lógica de negócios, em que cada operação exige uma ação compensatória correspondente. A gestão dessas operações pode rapidamente tornar-se complexa, especialmente em sistemas com inúmeras dependências entre serviços, intensificando a dificuldade no desenvolvimento e exigindo validação e testes rigorosos. Além disso, assegurar a consistência dos dados em um sistema distribuído com o SAGA apresenta desafios, mesmo o SAGA sendo projetado para lidar com consistência eventual, é imprescindível que todas as ações compensatórias necessárias sejam executadas corretamente em caso de falhas. Isso requer um mecanismo robusto de monitoramento e recuperação de falhas para garantir que todas as transações sejam concluídas ou revertidas adequadamente [98].

Dessa forma, esse estilo arquitetural oferece uma abordagem resiliente, confiável e escalável para gerenciar transações em sistemas distribuídos, com benefícios evidentes em termos de disponibilidade e desempenho. No entanto, os desafios de complexidade na implementação e na garantia de consistência dos dados demandam uma abordagem cuidadosa e um planejamento criterioso. Quando implementado corretamente, o SAGA pode proporcionar uma estrutura poderosa, flexível e escalável para manter a integridade e a consistência dos dados em ambientes distribuídos [100], [29].

2.3.4 Hexagonal Architecture

A arquitetura hexagonal, ou padrão ***Ports and Adapters***, visa a separação de responsabilidades em aplicações, isolando a lógica de negócio de componentes externos como interface de usuário e bancos de dados [101]. Essa abordagem promove a sustentabilidade

e adaptabilidade do software, com a interação entre o núcleo da aplicação e os componentes externos se dá por meio de **portas** e **adaptadores**, facilitando testes e elevando a qualidade do software [102, 101].

Observa-se como um dos benefícios da arquitetura hexagonal, a possibilidade de adaptação de novas tecnologias, como bancos de dados, sem alterar a lógica principal da aplicação [103]. Essa flexibilidade facilita a atualização tecnológica em cenários dinâmicos. Além disso, a organização do código proporcionada por essa arquitetura facilita a colaboração entre equipes e o gerenciamento de projetos complexos [104].

Contudo, a implementação da arquitetura hexagonal apresenta desafios, como a complexidade na definição de limites entre componentes e o gerenciamento de interfaces e adaptadores [105]. A curva de aprendizado pode ser acentuada para equipes com experiência em modelos tradicionais, exigindo familiaridade com conceitos como inversão de dependência e segregação de interface [105, 106].

Dessa forma, a arquitetura hexagonal constitui uma estrutura robusta para o desenvolvimento de aplicações adaptáveis e sustentáveis. A separação de responsabilidades inerente a esse modelo facilita a integração de novas tecnologias e a realização de testes. Porém a adoção dessa arquitetura exige atenção às complexidades e à curva de aprendizado. Em suma, a arquitetura hexagonal representa uma abordagem promissora para a construção de aplicações resilientes e flexíveis, aptas a suprir as demandas do desenvolvimento moderno [107, 106, 108].

Capítulo 3

Metódo de Pesquisa

Uma investigação multi métodos foi conduzida para abranger as perspectivas de acadêmicos e profissionais sobre os desafios encontrados na migração de aplicações legadas para ambientes ***Multi-Cloud***. Utilizou-se uma ***Revisão Sistemática da Literatura***, examinando artigos revisados por pares, para mapear com precisão os desafios observados nesse processo. Adicionalmente, incluiu-se um ***Estudo de Caso Etnográfico***, com observação participante, sobre o trabalho realizado em uma empresa do mercado financeiro para migrar uma aplicação legada tradicional, implantada em plataforma Mainframe, para uma arquitetura multi-cloud. Como resultado, esta pesquisa evidenciou os desafios da migração de aplicações legadas para multi-cloud, utilizando dois métodos de pesquisa distintos, com o propósito de integrar o conhecimento acadêmico com o conhecimento empírico observado em trabalhos reais de migração.

3.1 Revisão Sistemática da Literatura

A utilização da Revisão Sistemática da Literatura (RSL) contribuiu para o mapeamento do conhecimento acadêmico existente permitindo identificar as lacunas ainda não preenchidas na migração de aplicações legadas para ambientes multi-cloud. A Figura 3.1 apresenta uma visão geral do desenho do estudo. As seções seguintes detalham a codificação inicial/aberta, a seleção das categorias principais e a codificação focada.

Essas fases resultaram na análise de documentos, codificações, memorandos, categorias e desafios mapeados [17]. Assim, buscou-se fornecer uma síntese dos desafios associados à migração de aplicações para multi-cloud destinadas a gestores, bem como os profissionais e as organizações.

3.1.1 Planejamento da Revisão Sistemática da Literatura

Alinhados ao objetivo definido da questão de pesquisa ***RQ1***, foram selecionadas palavras-chave que delinearam a frase de pesquisa. Este processo tornou-se fundamental para identificar artigos que abordam o problema e podem contribuir para investigação, colaborando com a definição do protocolo de revisão.

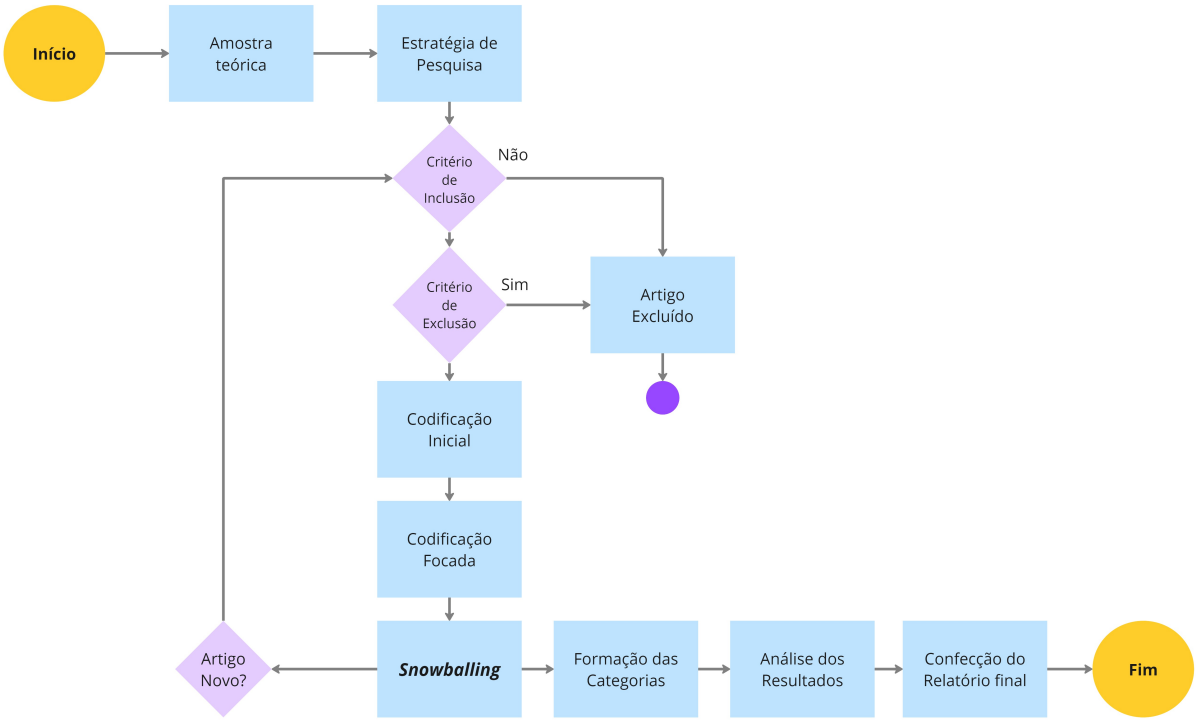


Figura 3.1: Processo da Metodologia de Pesquisa Passo a Passo

As fontes de pesquisa deste estudo foram constituídas por bases de dados digitais online, especificamente Scopus e IEEE Xplore. A base Scopus foi escolhida por oferecer um mecanismo de busca abrangente, que engloba publicações da Science Direct, ACM Digital Library e Springer Link. Foram utilizados os termos "Multi-Cloud"e "Migração", bem como suas variações, para obter uma compreensão mais abrangente de como os autores se referem a esses assuntos. A partir da análise, foram selecionados alguns artigos e extraídas as palavras-chave mais utilizadas associadas ao tema [23], [16], [109], [41]. Posteriormente, formulou-se a questão de pesquisa e definiu-se o protocolo de pesquisa com base nos *insights* extraídos desses artigos iniciais.

Após uma análise mais aprofundada sobre a abordagem do tema, foram selecionadas as palavras-chave mais relevantes e definidos os termos utilizados nas bases de pesquisa conforme mostrado na Figura 3.2. Dessa forma, estabeleceu-se o protocolo de pesquisa composto por duas fases: a Condução da Revisão e a Documentação da Revisão, denominadas como Resultados e Discussões.

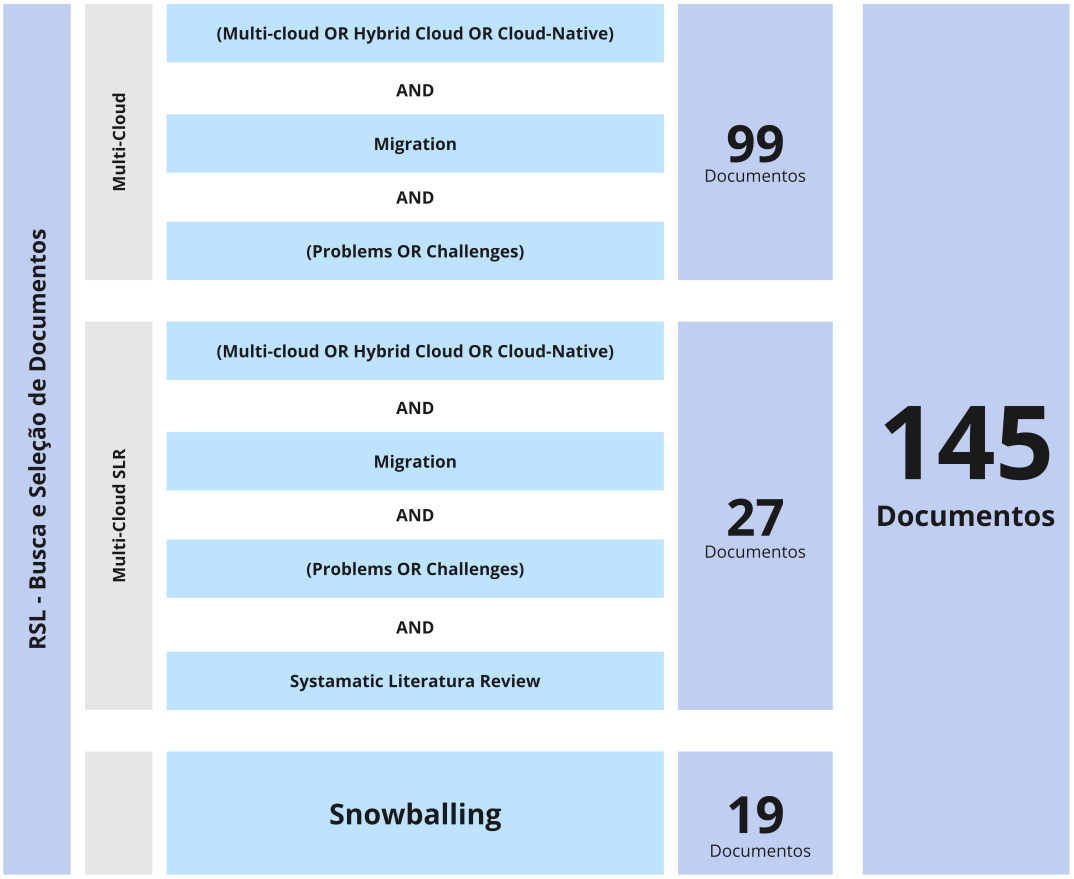


Figura 3.2: Compilação dos termos que compõem a frase de pesquisa e totalização dos documentos encontrados.

3.1.2 Executando a Revisão Sistemática da Literatura

A realização da Revisão Sistemática da Literatura RSL teve como objetivo identificar obras, terminologias, práticas e técnicas pelo ambiente acadêmico. Esse processo estabeleceu um percurso preciso e rigoroso, com etapas cuidadosamente mapeadas e documentadas, além de um protocolo bem definido para extração, análise e relato dos resultados [16], [110], [111].

Seguindo o protocolo definido na fase anterior, foram selecionados os artigos que constituíram a base de dados para esta pesquisa. Assim, antes de iniciar o processo de análise e síntese dos dados, foram aplicados filtros de inclusão e exclusão visando selecionar os artigos mais representativos e pertinentes para análise.

Seleção dos Estudos Primários

A aplicação dos termos de busca "Multi-cloud"e "Multi-cloud RSL"resultaram em 99 e 27 artigos, respectivamente, totalizando 126 artigos, conforme ilustrado na Figura 3.2.

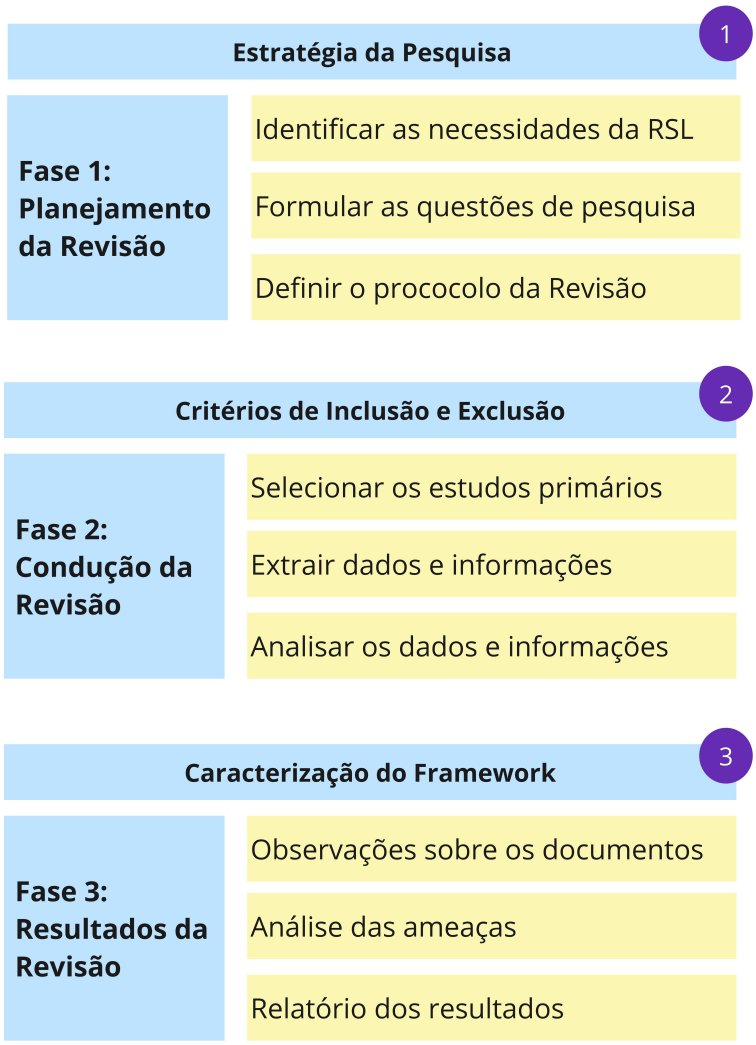


Figura 3.3: Processo da Metodologia de Pesquisa

O processo de pesquisa teve início com base nesse conjunto preliminar de dados, os quais foram submetidos à critérios de inclusão e exclusão para refinar a seleção. Durante essa fase analítica, foram identificados artigos adicionais referenciados pelos autores do conjunto inicial os quais continham informações pertinentes ao estudo. Esse método de expansão iterativa do conjunto de literatura é conhecido como *Snowballing*[112]. A seção a seguir descreve as etapas desse processo. As quatro etapas a seguir detalham os passos executados para delimitar a seleção de artigos objetos do estudo.

1. **Busca Avançada em Repositórios Científicos:** O processo de pesquisa foi realizado com base em critérios científicos, considerando exclusivamente artigos de periódicos, conferências, congressos ou simpósios.

2. **Filtros:** Foram aplicados filtros para excluir artigos com base em seus títulos,

utilizando os seguintes critérios: (1) "Fonte imprópria", (2) "Repetidos", (3) "Estudos não escritos em inglês"e (4) "Artigos com menos de três páginas".

3. **Seleção:** A seleção dos artigos foi realizada com base na leitura e análise do ***abstract*** e, quando necessário, no conteúdo integral dos textos. Adicionalmente, foi verificado se os artigos abordavam o problema e as questões de pesquisa relevantes para o estudo em questão.

4. **Conduzindo o Snowballing:** Para garantir a inclusão do maior número possível de estudos relevantes, incluindo aqueles não capturados pela frase de pesquisa inicial, empregou-se a abordagem de *snowballing* [112] de referência .

Após a aplicação da técnica descrita e a realização de testes em bases de dados de artigos científicos, foram formulados os termos de busca que produziram os resultados mais pertinentes. Estes termos, denominados critérios de inclusão, são apresentados na Tabela 3.4.

Critérios de Inclusão	
IC1	Os documentos devem abordar os seguintes assuntos em seus títulos, resumos ou palavras chaves: - Multi-cloud - OR Cloud-Native - OR Cloud - OR Private Cloud - OR Hybrid Cloud - OR Migration Between Clouds - OR Challenges of Deployment in Clouds - OR Migration from Legacy Systems to Cloud - OR Transformation from Legacy Systems to Cloud - OR Challenges of System Deployments in - Cloud Architecture - OR Microservices Architecture.
IC2	As publicações devem estar escritas em lingua Inglesa.

Figura 3.4: Critérios de Inclusão

Na fase subsequente de seleção e filtragem dos artigos, realizou-se um exame minucioso dos resumos e palavras-chave dos artigos coletados. A finalidade da análise foi identificar e eliminar artigos cujo conteúdo divergisse significativamente da questão de pesquisa. Nesta fase, utilizou-se um recorte da metodologia de pesquisa ***PRISMA*** para garantir a

aplicação constante e sistemática dos critérios de exclusão, conforme ilustrado na Figura 3.5 [113].

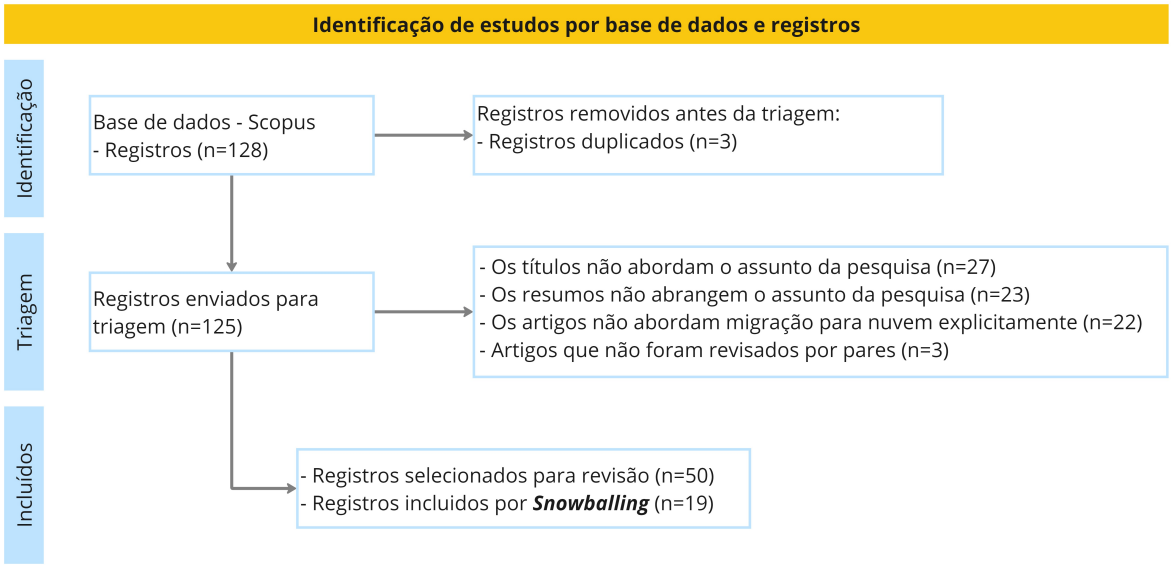


Figura 3.5: Critérios de Exclusão

Após a conclusão das etapas mencionadas, a seleção foi reduzida a cinquenta artigos essenciais, as quais foram encaminhados para a fase subsequente de análise e síntese aprofundada dos dados.

Extração e Análise dos Dados

No presente estudo, técnicas de codificação foram empregadas para realizar uma análise exaustiva dos artigos selecionados. Essa abordagem mostrou-se valiosa para mitigar, de forma sistemática, o viés do pesquisador, através da revisão colaborativa do livro de códigos [111]. Desse modo, durante o processo de codificação, foram recuperadas citações mencionadas diretamente no texto ou listadas nas bibliografias dos artigos. A identificação dessas referências resultou na recuperação e subsequente incorporação dos artigos à análise aprofundada de conteúdo . Essa estratégia está alinhada com o método *Snowballing*, conforme descrito por [114].

Para estruturar os conceitos emergidos da análise, foi empregado um conjunto de técnicas de codificação, como *Codificação Aberta*, *Codificação Inicial* e *Codificação Focada* [114, 115]. O processo de codificação envolveu a atribuição de rótulos aos segmentos de dados, facilitando a classificação, o resumo e a representação de cada ponto [115]. Assim, os dados analisados foram organizados em categorias que refletiram os temas de interesse dos pesquisadores envolvidos. Um livro de códigos abrangente foi elaborado,

contendo as definições dos códigos e incluindo trechos relevantes dos textos analisados. Esse procedimento possibilitou a identificação dos códigos e das categorias com maior frequência de ocorrência, auxiliando na categorização simplificada dos dados (Codificação Focada). A partir da análise comparativa dos dados e das categorias primárias, buscou-se compreender as relações entre esses componentes e integrar as percepções obtidas aos desafios identificados, conforme descrito no resultado da RSL.

3.1.3 Análise de dados e informações cruzadas da RSL

Todos os dados utilizados na Revisão Sistemática da Literatura, destinados a fundamentar as conclusões do estudo, fornecendo uma cadeia detalhada de evidências, estão disponíveis em `https://tinyurl.com/multicloudslr`.

3.2 Estudo de Caso Etnográfico

A etnografia constitui-se como uma metodologia de pesquisa na antropologia cultural além de representar o registro escrito que documenta os resultados dessa pesquisa. Frequentemente identificada com a própria antropologia cultural. No campo da pesquisa qualitativa, o estudo etnográfico é interpretado como um método observacional realizado em ambientes específicos, que não se limita à na observação, mas envolve a participação ativa nas atividades diárias do objeto do estudo [116].

A etnografia dedicou-se a examinar indivíduos em seus ambientes naturais, empregando métodos de coleta de dados que procuravam compreender seus contextos sociais e atividades cotidianas. Tornou-se necessário que os pesquisadores se engajassem diretamente com o ambiente e com as atividades, além reunir dados sem aplicar interpretações sistemáticas [116]. Dessa forma, o estudo de campo concentra-se na análise das pessoas em seus ambientes de vida reais, observando-as em seus locais de residência e imergindo em suas atividades cotidianas.

Metodologias flexíveis foram utilizadas, não estruturadas e abertas, permitindo uma compreensão aprofundada das experiências vivenciadas [116]. Um estudo de caso pode ser entendido como a análise de um fenômeno específico, abrangendo um programa, evento, indivíduo, processo, instituição ou grupo social. Trata-se de uma investigação empírica que aborda um fenômeno atual em seu contexto real de vida, particularmente quando as delimitações entre o fenômeno e o contexto não são claras [117]. Em resposta, os engenheiros de software utilizaram técnicas de modelagem avançadas, e a experiência prática dos profissionais para direcionar e orientar a pesquisa acadêmica [118].

Propôs-se, uma abordagem metodológica mais rigorosa e detalhada, a qual deveria ser seguida estritamente, destacando-se a importância da flexibilidade na investigação etno-

gráfica. Argumentou-se que essa flexibilidade era necessária para ajustar-se à realidade e alcançar os objetivos pretendidos, desde que fossem empregadas as técnicas tradicionalmente associadas à etnografia, como a observação participante, a entrevista intensiva e a análise documental [119], [120].

Dessa forma, descrevem-se a seguir os passos essenciais para a condução de uma pesquisa etnográfica [121]:

1. **Definição do problema e seleção do caso**: A seleção deve ser embasada na relevância do caso para o fenômeno investigado e na sua potencialidade de proporcionar *insights* significativos. A justificativa para a escolha do caso deve ser meticulosa, detalhando de maneira abrangente por que tal caso é representativo ou excepcional dentro do contexto do tema em questão [121].

2. **Preparação para entrada em campo**: Garantir as autorizações necessárias e entender o ambiente cultural e social do grupo que será investigado. O pesquisador deve realizar uma imersão eficiente e respeitosa [121].

3. **Coleta de dados**: O envolvimento do pesquisador nas atividades diárias do grupo permite a coleta de observações participantes e o registro detalhado de comportamentos, interações e eventos. A triangulação de dados proporciona a integração as informações de diferentes fontes [121].

4. **Análise e interpretação dos dados**: A codificação dos dados, o reconhecimento de temas e padrões recorrentes, a análise reflexiva considerando as perspectivas e interações do pesquisador, e a interpretação contextualizada dentro do quadro teórico da pesquisa bem como das evidências obtidas na revisão sistemática da literatura. O relatório final deve apresentar uma narrativa detalhada e abrangente, que compreenda a complexidade e as sutilezas do caso analisado [121].

Capítulo 4

Desafios da Migração para Multi-Cloud: uma perspectiva a partir da Revisão Sistemática da Literatura

A migração de aplicações legadas para a nuvem mantém-se como foco nas equipes de TI, enquanto os benefícios e desafios da arquitetura multi-cloud persistem em debate no meio científico e corporativo [13].

A busca com a frase de pesquisa ilustrada na Figura 3.2 retornou 126 trabalhos. Após a aplicação dos critérios de inclusão e exclusão, e a adição de 19 artigos via processo *Snowballing* [112], 50 artigos foram selecionados para análise detalhada na primeira fase da pesquisa. A análise da migração de aplicações para multi-cloud revelou a predominância de desafios sobre os benefícios, com 54% das citações codificadas associadas a "desafios"e 46% a "benefícios". Assim, com o uso da a Codificação Focada gerou-se nove categorias semânticas, representadas na Figura 4.1 e detalhadas nas subseções subsequentes. Os desafios de cada categoria e os respectivos trabalhos constam na Tabela do Apêndice A.

4.1 DESAFIOS DA MULTI-CLOUD

Abordando os desafios destacados no contexto Multi-Cloud, ficou evidente a complexidade de gerenciar e organizar essa infraestrutura heterogênea; assim a seleção de provedores de nuvem capazes de atender a todos os requisitos da aplicação não é uma tarefa trivial [9]. Dessa forma, a implementação de uma estratégia Multi-Cloud demanda um planejamento meticuloso e uma análise criteriosa dos recursos oferecidos por cada provedor para garantir a conformidade com as especificações técnicas e operacionais das aplicações envolvidas.

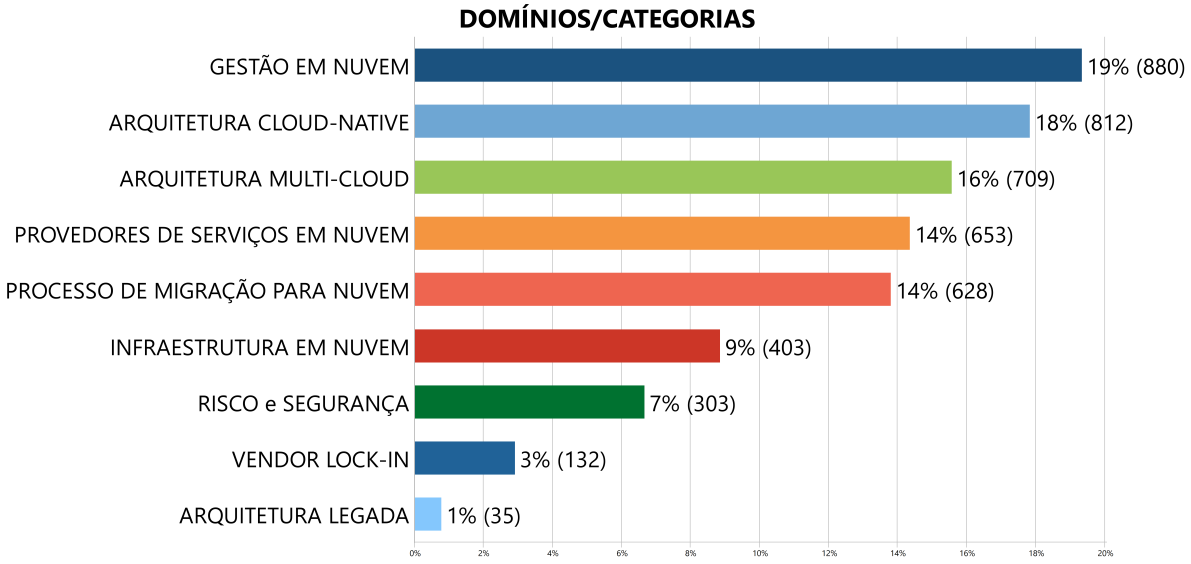


Figura 4.1: Domínios ou categorias identificadas por *Focus Coding*.

[31] , [38]. Especialistas propõem combinar provedores de nuvem para criar soluções multi-cloud personalizadas [9]. Essa abordagem considera o alinhamento entre dados históricos e estrutura otimizada, buscando o provedor mais adequado às necessidades do cliente para maximizar a eficiência na infraestrutura multi-cloud. As abordagens de migração existentes requerem ajustes para acomodar a heterogeneidade das plataformas e as exigências das aplicações modernizadas [122].

A implementação de uma solução para o uso de múltiplas nuvens introduz pontos críticos, incluindo a portabilidade entre provedores, essa característica pode prejudicar a interoperabilidade entre nuvens e resultar no bloqueio do fornecedor, também conhecido como ***vendor lock-in***, logo portabilidade é definida como a capacidade de migrar dados e sistemas de uma nuvem para outra, enquanto a interoperabilidade é a capacidade de utilizar serviços em várias nuvens de maneira integrada [12].

Além disso, a insuficiência de documentação emergiu como um obstáculo significativo, impondo dificuldades aos times no processo de adaptação de sistemas legados à nuvem. Essa carência demandou uma análise minuciosa em atividades como reengenharia e refatoração. Nesse contexto, pesquisas apresentaram metodologias e estruturas para endereçar os desafios inerentes à migração de aplicações, oferecendo suporte para a execução dessa tarefa. [9].

4.2 BENEFÍCIOS DA MULTI-CLOUD

Em referência aos benefícios observados, a computação em nuvem, quando corretamente dimensionada, contribuiu para uma solução acessível para organizações com orçamento

limitado, oferecendo acesso a serviços e recursos tecnológicos sem a necessidade de investimentos significativos [123]. Ao expandir as possibilidades, a arquitetura Multi-Cloud permitiu a implantação de microsserviços em áreas específicas, respeitando tanto os interesses organizacionais quanto as restrições legais de cada região, sendo uma das razões pelas quais as empresas estão optando por arquiteturas de microsserviços em vez das tradicionais arquiteturas monolíticas, proporciona uma transição mais eficiente entre as diferentes nuvens disponíveis [5].

A adoção da nuvem como plataforma para o desenvolvimento de aplicações proporcionou uma série de benefícios, como maior tolerância a falhas, redução de custos, acesso facilitado a dados, aprimoramento da segurança, além de escalabilidade e resiliência, esses fatores favoreceram o avanço das aplicações nativas da nuvem [23]. Por sua vez, a adoção de uma arquitetura Multi-Cloud, caracterizada pela utilização de múltiplos provedores de serviços em nuvem, ampliou a disponibilidade e diversificação de recursos, mitigando a dependência de um único provedor. Porém, apesar dos avanços observados nas estratégias de migração para esse tipo de arquitetura, um modelo ideal para a implementação de Multi-Cloud ainda não foi plenamente definido [8].

Dessa forma, observou-se a migração de aplicativos legados para soluções baseadas em nuvem, incluindo o modelo de Software como Serviço (SaaS), consolidando-se como uma tendência. A adoção da arquitetura de microsserviços desempenhou um papel fundamental nesse processo, permitindo ciclos de lançamento mais ágeis e a implementação gradual de novas funcionalidades. Esse movimento impulsionou a modernização e a migração de sistemas legados, orientando-os em direção a modelos que proporcionaram benefícios como economia de escala, eficiência energética, escalabilidade e elasticidade [124], [85], [87].

4.3 GESTÃO EM NUVEM

Com a adoção de estratégias Multi-Cloud pelas organizações, a gestão eficaz durante o processo de migração torna-se uma questão relevante. Na Figura 4.2, são evidenciadas as propriedades relacionadas ao **Categoria Gestão em Nuvem**: *Performance*, *Custo*, *Portabilidade*, *Monitoração*, *Múltiplos locatários*, *DevOps*.

Conforme evidenciado na Figura 4.2, a propriedade *Performance* foi sucedida por *Custos* como principais preocupações nessa categoria de pesquisa. A literatura destacou a seleção de um provedor de armazenamento em nuvem adequado implicava na busca por um equilíbrio entre custo e desempenho, como apontado por [11].

No entanto, o progresso das configurações das arquiteturas de nuvem, passando de nuvens privadas para híbridas e de federação até Multi-Cloud, permitiu a otimização de

custos e a adaptação de cargas de trabalho a diferentes cenários. Diante disso, a gestão de um ambiente heterogêneo exigiu a consideração de aspectos técnicos, estratégicos e financeiros.

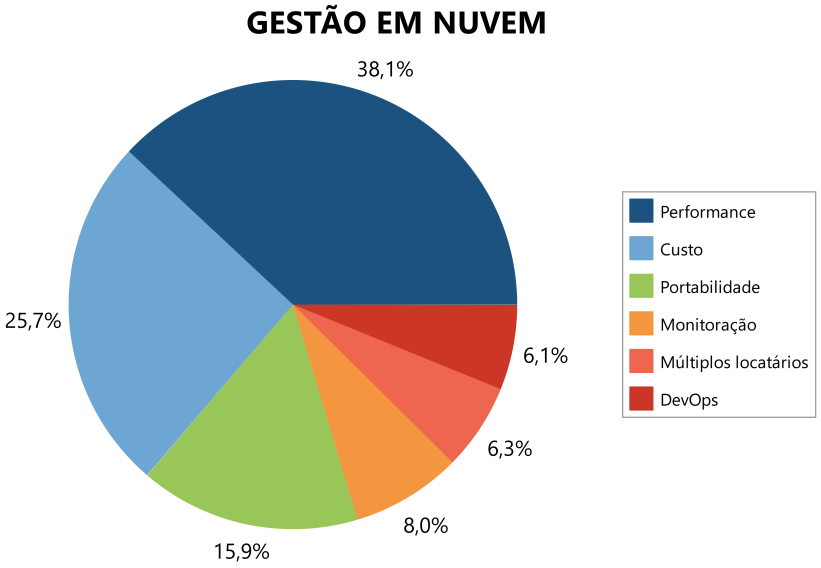


Figura 4.2: Desafios relacionados à categoria "*Gestão em Nuvem*". Cada propriedade apresentou o número de citações encontradas no estudo e o percentual de citações quando considerado apenas nesta categoria.

O gerenciamento de um ambiente heterogêneo requer o monitoramento constante dos aplicativos e microsserviços. Embora existam diversas ferramentas e estruturas de rastreamento. Com a variedade de opções de monitoramento disponíveis e às complexidades do monitoramento de aplicativos de microsserviços, uma estrutura de monitoramento deve ser projetada para alertar as partes interessadas sobre o desempenho e as falhas de um aplicativo com base nos dados capturados [5]. No entanto, os recursos oferecidos pelos provedores, são disponibilizados através serviços em suas plataformas, podendo incluir recursos de monitoração com garantias de acordos de níveis de serviço (*SLAs*) [31].

4.4 ARQUITETURA CLOUD-NATIVE

A arquitetura Cloud-Native abrange aspectos gerais de design, incluindo abordagens de arquitetura orientada a microsserviços, implementados como unidades independentes encapsuladas em contêineres [87]. A adoção da nuvem tem incrementado sua relevância globalmente, com empresas concentrando-se na migração de aplicações legadas e, no desenvolvimento de aplicativos nativos da nuvem (*cloud-native*) evidenciando os benefícios dessa estratégia. A implantação de aplicativos projetados para a nuvem promove a intero-

perabilidade ou portabilidade em cenários híbridos e Multi-Cloud, essencial para superar as vulnerabilidades do *vendor lock-in* [23].

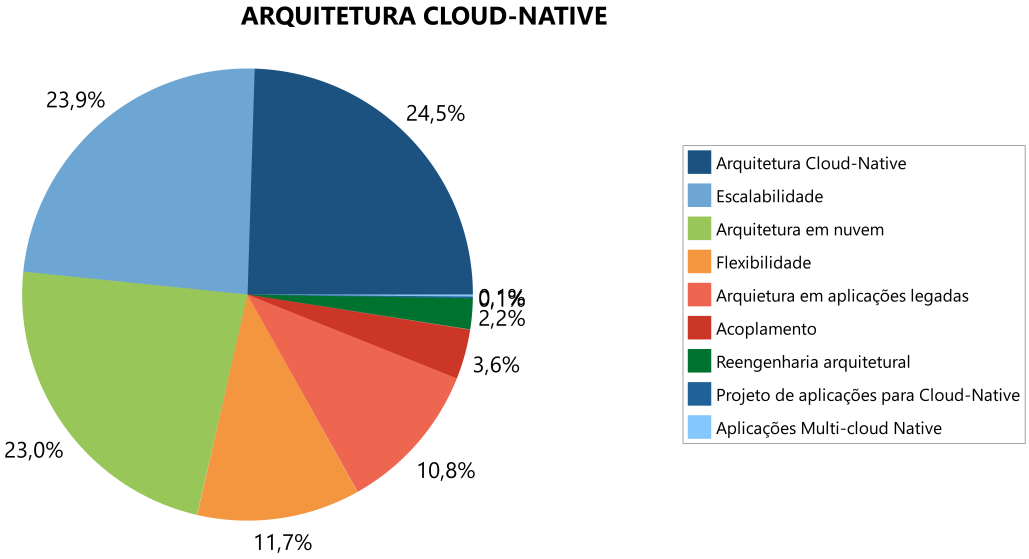


Figura 4.3: Desafios relacionados à categoria "*Arquitetura Cloud-Native*". Cada propriedade apresenta o número de citações encontradas no estudo e o percentual de citação quando considerado apenas nesta categoria.

Uma aplicação construída em uma arquitetura nativa para nuvem (*Cloud-Native Architecture*), frequentemente implementadas em arquiteturas de microsserviços, possui como características principais a portabilidade, elasticidade, flexibilidade e manutenibilidade, esses atributos contribuem para o incremento na adoção dessa arquitetura como o padrão das aplicações desenvolvidas atualmente. [31].

Porém, em determinados contextos, requisitos legais podem impor restrições geográficas à implementação de aplicações, notadamente no que concerne ao armazenamento de dados. As aplicações concebidas com base na arquitetura nativa da nuvem demonstram vantagens no atendimento a esses requisitos em comparação com a aplicações legadas. Sua portabilidade, fruto de um design modular e distribuído, faculta a implantação, parcial ou total, em regiões que melhor se adequem às necessidades e restrições impostas [5]. Os especialistas apontam que aplicações nativas da nuvem podem representar uma evolução natural de microsserviços e abordagens baseadas em contêineres [23].

Nesse contexto, entre as propriedades observadas, a preocupação com a elaboração de uma arquitetura de aplicação adequada à nativa para nuvem correspondeu a 24,5% das citações nessa categoria. A escalabilidade, com 23,9%, figurou como a segunda propriedade mais citada, conforme ilustrado na Figura 4.3.

Entretanto, os padrões de desenvolvimento não devem interferir no projeto arquitetural de um sistema. A implementação do sistema deve seguir um padrão arquitetural no

qual os componentes são construídos e implementados para serem desacoplados. Quando a necessidade de desenvolvimento determinar um acoplamento inevitável, este componente deverá ser implementado separadamente para não impactar a evolução do sistema [125]. Um aplicativo desenvolvido nativamente para a nuvem suporta alta escalabilidade e elasticidade [23]. Dessa forma, a computação em nuvem tem restrições arquiteturais na construção de software, como modularização, desacoplamento e componentização [126].

A observância ao paradigma de construção na arquitetura nativa da nuvem garante a confiabilidade, disponibilidade e escalabilidade das aplicações. Este paradigma aborda as limitações associadas em arquiteturas legadas de aplicações monolíticas [86], [5] , [127].

4.5 MULTI-CLOUD ARCHITECTURE

Ainda existem algumas divergências, entre os especialistas sobre o conceito adequado para definir Multi-Cloud, alguns consideram-no como um tema de infraestrutura, classificando nuvem como *Private Cloud*, *Public Cloud*, *Hybrid Cloud* e *Federate Cloud* [4]. Porém a abordagem ampliada conceitua Multi-Cloud além do tema de infraestrutura, abrangendo outros componentes do ecossistema incluindo a arquitetura das Aplicações com Arquitetura Nativas para Multi-Cloud (*Multi-Cloud Native Architecture*) [13].

Dessa forma, verificou-se um empenho considerável na resolução de questões relacionadas com arquiteturas de integração Multi-Cloud, como: *Hybrid Cloud, Federated Cloud, Multi-Cloud, Ad-hoc Cloud, e Micro Cloud.* Esse esforço incluiu a construção de infraestruturas de nuvem altamente resilientes, caracterizadas pela utilização de recursos redundantes estrategicamente distribuídos por múltiplos data centers em diversas zonas de disponibilidade [4].

Apesar da predominância de arquiteturas de nuvem privada e híbrida, impulsionada por restrições específicas, o aumento da oferta de serviços de nuvem pública facilitou a construção de ambientes Multi-Cloud mais diversificados, com a participação de múltiplos provedores [128].

A migração de aplicações para a nuvem permite que organizações explorem benefícios como elasticidade, tolerância a falhas, redução de custos e acessibilidade. [129]. Contudo, a busca por maximizar os benefícios de escalabilidade, flexibilidade e desacoplamento em sistemas nativos de nuvem impulsiona a adoção da estratégia multi-cloud [41]. Essa abordagem permite a distribuição de aplicações entre diferentes provedores, otimizando recursos e mitigando os riscos de *vendor lock-in* [41, 13]. A diversificação de provedores garante a continuidade dos serviços em casos de falhas ou indisponibilidade de um provedor específico, além de oferecer vantagens no gerenciamento de custos. Dessa forma, o uso de múltiplas nuvens não se limita à redundância e resiliência, mas também possibilita

uma alocação mais eficiente de recursos, permitindo que as organizações ajustem suas infraestruturas de acordo com necessidades específicas e dinâmicas de mercado [9].

A diversificação de provedores, além de garantir a continuidade dos serviços em caso de falhas, promove vantagens no gerenciamento de custos [9]. O uso de múltiplas nuvens vai além da redundância e resiliência, permitindo alocação eficiente de recursos e o ajuste da infraestrutura às necessidades dinâmicas do mercado [9].

A Figura 4.4, ilustra os principais desafios mapeados na categoria "Arquitetura Multi-Cloud": *"Multiplas nuvens", "Nuvem Hibrida", "Seleção apropriada de serviços e provedores", "Nuvem Privada", "Redução de Custo", "Benefícios da Multi-cloud", "Prevenir Vendor lock-in", "Risco com Segurança", "Modelos de decisão para adoção de Multi-Cloud", "Escolha e Flexibilidae", "Alta disponibilidade".*

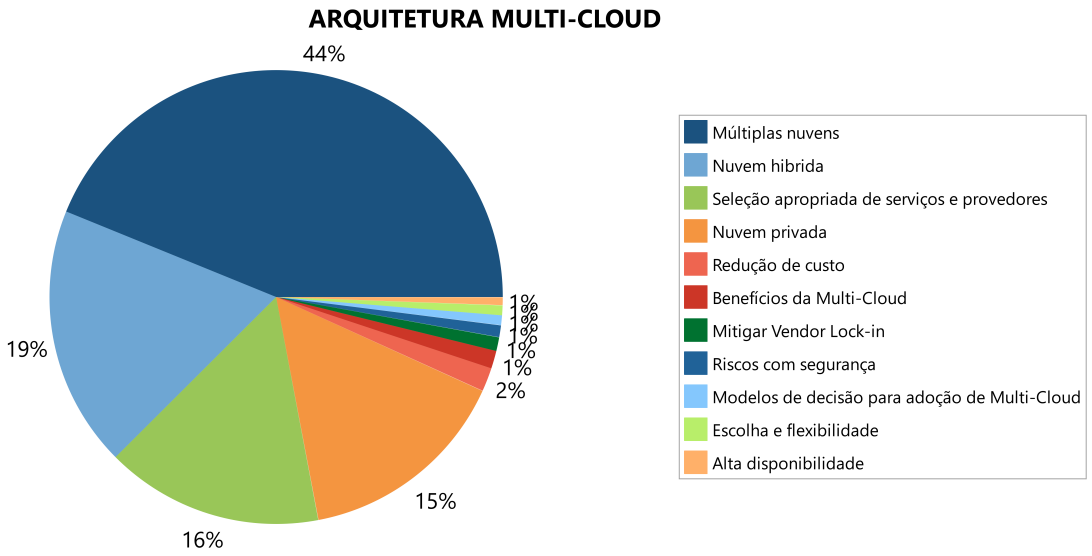


Figura 4.4: Desafios relacionados à categoria *Arquitetura Multi-Cloud*. Cada propriedade apresenta o número de citações encontradas no estudo e o percentual de cotação quando considerado apenas nesta categoria.

O uso de múltiplas nuvens, promovido por estudiosos, contribui para solucionar aspectos críticos no desenvolvimento de aplicações, como alta disponibilidade, tolerância a falhas e flexibilidade. A implementação em diversos provedores reduz o risco de indisponibilidade, favorece o desenvolvimento de aplicações com boa relação custo-benefício, além de colaborar com a redução do vendor lock-in [6].

Identificaram-se três categorias relevantes, impulsionando a adoção da arquitetura multi-cloud, descritas a seguir [130]:

- **Redução da dependência do fornecedor**: envolve a proteção dos aplicativos contra interrupções ou falhas reduzindo a concentração de fornecedor [130].

- **Otimizando a performance e os custos**: diversos provedores de nuvem apresentam vantagens exclusivas, com a integração dessas ofertas pode-se aprimorar a Qualidade de Serviço (*QoS - Quality of Service*) e a eficiência econômica. A combinação do uso de múltiplas nuvens, possibilita a redistribuição do processamento, durante períodos de alta demanda ou quando surgem melhores oportunidades financeiras [130].
- **Atender requisitos, restrições ou regulamentos:** A utilização de múltiplas nuvens mostra-se indispensável quando as necessidades de uma aplicação excedem a capacidade de um único provedor. Adicionalmente, imposições regulatórias podem demandar o armazenamento de dados sensíveis em locais específicos, como o país de origem do usuário [130].

Para viabilizar a multi-cloud, a portabilidade destaca-se como requisito fundamental, permitindo a migração de aplicações, cargas de trabalho, processos e dados entre diferentes provedores, com mínima interrupção, por meio de processos manuais ou automatizados [5]. Aplicações portáveis requerem design *Cloud-Native*, sendo genéricas e desacopladas para suportar a diversidade de serviços em DevOps [41].

4.6 PROVEDORES DE SERVIÇOS EM NUVEM

A diversidade de recursos em ambientes multi-cloud impõe desafios na escolha da configuração ideal para a implantação de aplicações [122]. A migração entre provedores apresenta obstáculos, reforçando a importância da arquitetura *Cloud-Native* e da combinação de provedores para otimizar recursos [9]. Essa transição entre provedores exige integração e ajuste de serviços, buscando flexibilidade e adaptabilidade às necessidades das aplicações, superando as limitações impostas pela diversidade em ambientes multi-cloud, conforme [6].

Os desafios relacionados à categoria "*Provedor de Serviços em Nuvem*"são: *"Provedor de Nuvem", "Intranuvem e Federação em nuvem", "Reduzida configuração de recursos", "Gestão em ambiente Multi-Cloud", "Ferramentas de gestão Multi-cloud"*

No entanto, embora provedores de nuvem ofereçam soluções para migração, o custo permanece elevado [36]. A busca por atender às necessidades dos clientes resulta em soluções com diferentes graus de exclusividade, impactando diretamente no risco de *vendor lock-in* [122].

Dessa forma, a diversidade de ofertas em ambientes multi-cloud intensifica as preocupações com a segurança das aplicações, notadamente no armazenamento e integração de dados. A mitigação de riscos de vazamento de informações impulsiona provedores a

investir em ferramentas de monitoramento rigorosas para proteger os dados dos clientes, como observado em [131].

4.7 PROCESSO DE MIGRAÇÃO PARA NUVEM

A modernização de aplicações legadas tem sido foco de estudo de diversos pesquisadores [8], [9], [23]. Ao analisar a categoria "*Processo de migração em nuvem*", foram identificados cinco principais desafios: "*Estratégia de migração para nuvem*", "*Migração em Multi-Cloud*", "*Benefícios de migração para nuvem*"e "*Processo de migração para nuvem*".

A migração de aplicações para a nuvem destaca-se como desafio relevante no cenário atual, conforme [8]. Assim, a complexidade na migração de aplicações legadas assume relevância na literatura científica, e estudos recentes concentram-se em otimizar a decisão de migração via técnicas de *machine learning*, visando aprimorar velocidade e precisão [9].

Ainda assim, aplicações legadas concebidas para escalabilidade vertical, não suportam a escalabilidade dinâmica dos ambientes de nuvem [8]. Por isso, a modernização impõe complexidade, especialmente pela reengenharia e ausência de documentação [9]. Seis etapas garantem o sucesso da migração: adequação do serviço, verificação da estratégia de negócios, análise do ambiente em nuvem, avaliação de riscos, análise do fornecedor e implementação. O cumprimento rigoroso mitiga riscos e assegura a transição eficiente [11].

4.8 INFRAESTRUTURA EM NUVEM

A computação em nuvem, com sua flexibilidade e capacidade de atender demandas dinâmicas, contribui para a redução de custos, conforme [23]. O ecossistema de provedores oferece diversidade de produtos e serviços, moldando a compreensão da computação em nuvem. Ambientes de nuvem permitem a alocação de serviços flexíveis e ajustáveis a qualquer aplicação, como observado em [11].

A rápida expansão da computação em nuvem impulsiona a criação de soluções e serviços inovadores, demandando, porém, enfrentamento de diversos desafios, conforme [6]. Na categoria "Infraestrutura em nuvem", destacam-se *"Arquitetura da infraestrutura em nuvem", "Definição da computação em nuvem", "Elasticidade", "Evolução da computação em nuvem", "Infraestrutura em nuvem hibrida", "Infraestrutura em nuvem pública"*

Uma infraestrutura em nuvem pode ser implantada através de Nuvem Pública, Nuvem Privada, Nuvem Hibrida ou Multi-Cloud e seus serviços podem ser oferecidos pelos modelos como IaaS, PaaS, e SaaS. Esses modelos variam no nível de controle proporcionado

sobre hardware e configuração. Cada alternativa apresenta vantagens e desvantagens significativas a serem avaliadas nas decisões empresariais, visto que escolhas inadequadas podem resultar em custos adicionais [11].

Além do mais, o monitoramento da infraestrutura e das aplicações permanece como um desafio central, especialmente no contexto de arquiteturas Multi-Cloud, que envolvem aplicações distribuídas [132]. A supervisão adequada é imprescindível para a gestão eficiente de qualquer infraestrutura de TI, uma vez que sistemas são suscetíveis a falhas e sem um monitoramento apropriado, torna-se difícil identificar as causas dos incidentes ou prever potenciais problemas futuros. A implementação de soluções eficazes de monitoramento mostra-se essencial para assegurar a resiliência e o desempenho operacional [5].

4.9 RISCO E SEGURANÇA

A segurança apresenta-se como uma preocupação importante na migração de aplicações para a nuvem, especialmente em ambientes federados e multi-cloud. A integração e gestão de múltiplos provedores impõem desafios à proteção de dados e à conformidade com normas de segurança, tornando este aspecto crítico no planejamento de migrações [133]. Outros desafios mapeados na categoria *Risco e Segurança* são: "*Privacidade dos dados*", "*Segurança*"e "*Preocupações com risco e segurança*".

Além de suas vantagens, a computação em nuvem impõe desafios à segurança e privacidade, comprometendo sua adoção. A migração de dados para serviços de nuvem gera preocupações quanto à perda de controle, confidencialidade, integridade e disponibilidade, acrescidas dos riscos de *shadow IT*. Ameaças como adulteração e vazamento de dados, falhas de segurança e ataques internos demandam atenção. [31].

Segundo os pesquisadores, a mitigação de riscos requer abordagens que abrangem soluções tecnológicas, políticas de segurança e conformidade legal, considerando as regiões de hospedagem e acesso às aplicações. A segurança e a privacidade demandam planejamento detalhado, requisitos rigorosos e métricas robustas de avaliação. A superação dos desafios, por meio de soluções técnicas, organizacionais e regulatórias, impulsiona a computação em nuvem a alcançar seu pleno potencial [127], [8].

4.10 VENDOR LOCK-IN

Diferentes provedores de nuvem oferecem serviços exclusivos e frequentemente incompatíveis, resultando na impossibilidade de migração automática de aplicativos entre prove-

dores de nuvem, causando um aprisionamento indesejado impedindo a implantação de aplicativos ***Multi-Cloud Native*** [12, 8].

A escassez de referências com propostas práticas para evitar o *vendor lock-in* configura fator de alto risco, ameaçando o sucesso da portabilidade em ambientes multi-cloud [134]. Os desafios encontrados nesta categoria são: *"Desafios do vendor lock- in", "Dificuldade de portabilidade entre nuvens", "Impactos do vendor lock-in", "Preocupações do Vendor lock-in", "Riscos do vendor lock-in".*

Além disso, a cautela na utilização de funcionalidades específicas de provedores de nuvem mostra-se relevante para preservar a portabilidade em ambientes multi-cloud, como evidenciado em [11] e [135]. A ausência de padronização dificulta a adoção da computação em nuvem, criando barreiras à portabilidade [16]. As soluções existentes para mitigar o *vendor lock-in* concentram-se em aspectos tecnológicos, com carência de estudos sobre a natureza multifacetada dessa dependência, conforme [134]. Assim, a mitigação dos desafios requer camadas de abstração e soluções de *middleware* que possibilitem a implementação e gestão de aplicações em nuvens distintas [133]. Estratégias como padrões abertos, arquiteturas multifuncionais e abordagens multi-cloud auxiliam na mitigação dos riscos de *vendor lock-in*, garantindo flexibilidade e portabilidade [133].

4.11 ARQUITETURA LEGADA

A migração de aplicações legadas para a nuvem geralmente exige reestruturação completa, abrangendo design e implementação, visando sua adequação à arquitetura Cloud-Native. Aplicações monolíticas, migradas sem reestruturação, mantêm suas limitações, especialmente quanto à escalabilidade e manutenibilidade [41].

Aplicações legadas, comumente monolíticas, integram todas as funcionalidades em uma única unidade. Apesar da ampla adoção de paradigmas tecnológicos avançados, sistemas monolíticos persistem em muitas empresas, dificultando a implementação de novos recursos, a modificação de funcionalidades e a integração de tecnologias. Limitados pela arquitetura, esses sistemas apresentam desafios de desempenho, como latência de rede e longos tempos de reinicialização, decorrentes da alta demanda por recursos computacionais [85].

4.12 Resultados da Revisão Sistemática da Literatura

A migração de sistemas legados para ambientes Multi-Cloud revelou um cenário complexo de desafios e oportunidades. A revisão sistemática identificou obstáculos multifacetados enfrentados por organizações, com destaque para o ***vendor lock-in***, que pode prejudicar a portabilidade entre nuvens heterogêneas, limitando os benefícios da Multi-Cloud. Esse fenômeno, causado pela dependência de tecnologias específicas de fornecedores, reforçou a necessidade de estratégias e modelos arquiteturais que promovam a portabilidade e a integração entre plataformas de nuvem.

O estudo evidenciou o potencial das arquiteturas Multi-Cloud para elevar a resiliência e a eficiência operacional, indicando a carência de metodologias eficazes para gerenciar implementações. A migração, além dos desafios técnicos, implica considerações estratégicas como alinhamento com objetivos de negócios, conformidade regulatória e segurança, demandando estruturas que integrem aspectos técnicos e estratégicos.

O estudo consolidou os desafios da modernização de aplicações legadas, evidenciando a importância das arquiteturas Cloud-Native para operação em ambientes Multi-Cloud. A adoção dessas arquiteturas mitiga os riscos de *vendor lock-in*, aumentando a resiliência e escalabilidade. Constatou-se, porém, lacuna na literatura quanto a diretrizes e melhores práticas para evitar o *vendor lock-in*, demandando pesquisas sobre padrões arquiteturais para aplicações multi-cloud.

A revisão sistemática da literatura reforçou a importância das arquiteturas Cloud-Native e da superação da dependência de fornecedores, abrindo perspectivas de pesquisa para o desenvolvimento de diretrizes arquiteturais, ferramentas de gestão Multi-Cloud e processos de migração abrangentes. Investigações futuras nessas áreas são essenciais para que organizações explorem o potencial da Multi-Cloud em prol da inovação e competitividade digital.

Capítulo 5

Uma proposta de migração para Multi-Cloud Native Architecture: a perspectiva a partir de um Estudo de Caso Etnográfico

Os sistemas atuais, em constante evolução, tornar-se-ão os legados do futuro. A modernização mostra-se essencial para incorporar novas tecnologias e preservar a relevância desses sistemas. Este estudo apresentou um caso prático de migração de um sistema legado mainframe COBOL para uma aplicação cloud-native em Java Quarkus [136]. Abordou-se o processo de modernização, descrevendo desafios, decisões eficazes e ineficazes. Detalhou-se o processo de modernização em uma organização do setor financeiro da América Latina, fornecendo referência referência valiosa para profissionais engajados nessa atividade.

Utilizaram-se ferramentas e técnicas consolidadas como *Domain-Driven Design (DDD)* [92], *Event Storming* [96], *SAGA Orchestration Pattern* e *Hexagonal Architecture Pattern* [29] para a modernização do sistema. O processo, ainda em andamento, mostra-se extenso devido à complexidade e ao tamanho do sistema, porém a migração do primeiro módulo já forneceu informações relevantes para profissionais envolvidos em projetos similares.

A modernização de sistemas legados mainframe para microsserviços proporciona benefícios relevantes [137]. O processo permite compreensão profunda dos domínios de negócio, simplificando a manutenção, evolução e resolução de problemas, modelando cada microsserviço para atender a um subdomínio ou funcionalidade específica [137].

Sobre os padrões arquiteturais, a adoção do *Saga Orchestration Pattern* proporcionou robustez no tratamento de transações distribuídas, elevando a disponibilidade e a resiliência do sistema. Essa estratégia mostrou-se essencial na modernização de sistemas legados, mitigando os impactos de falhas e indisponibilidades [29]. Além dos benefícios

proporcionados pelo SAGA, a Arquitetura Hexagonal facilitou a integração com os artefatos de negócio modelados pelo DDD [92], promovendo a separação entre domínios da aplicação e tecnologia [138, 29]. Dessa forma, a arquitetura Cloud-Native proporcionou escalabilidade automática e gestão eficiente de custos [139]. A escalabilidade horizontal dinâmica, permitida pela arquitetura de microsserviços, atende às demandas variáveis e garante desempenho consistente, evitando custos de infraestrutura ociosa [29]. Assim, a portabilidade mitiga os riscos de **Vendor Lock-in** e permite a exploração de múltiplos ambientes de nuvem, incluindo configurações **Multi-Cloud**. A modernização para microsserviços na nuvem assegura escalabilidade, confiabilidade e adaptabilidade [29] , [140].

5.1 O Sistema Legado

O sistema legado discutido neste artigo é um componente central no ecossistema de uma grande instituição financeira denominada ***Zetta Bank***[1]. Desenvolvido há mais de três décadas, o sistema permanece em atividade e é essencial para a oferta e gestão de operações de crédito, sendo selecionado para passar por uma modernização significativa.

Originalmente implementado em uma arquitetura de mainframe e codificado em COBOL com banco de dados DB2, o sistema é composto por **nove módulos**. Este estudo detalha a ***migração do primeiro módulo*** com aproximadamente 1,5 milhão de acionamentos mensais, e 23 programas principais, que interagem com outros programas secundários, totalizando em torno de 140 mil linhas de código, integrando com outros 13 sistemas externos para recuperar ou fornecer informações corporativas.

O sistema incorporou novas funcionalidades e atualizações tecnológicas ao longo dos anos, notadamente na camada de apresentação, com a adoção de frameworks como Struts, JSF e Angular. Apesar de refletir tecnologias anteriores, o sistema mantém-se crítico para as operações da empresa, e sua evolução funcional e a ausência de documentação atualizada acrescentaram complexidade à migração, demandando planejamento meticuloso.

O ecossistema da aplicação, abrange mais de oitenta outros sistemas, incluindo entidades e órgãos externos. O acesso, realizado por canais usuais como dispositivos móveis e internet, e por aplicações em nuvem ou mainframe, dá-se através de um componente denominado *ESB - Enterprise Service Bus*, conforme Figura 5.1.

[1]Foram apresentadas observações com base em experiências reais de uma grande instituição financeira latino americana, doravante referida como ***Zetta Bank***. As informações disponibilizadas passaram por adaptações ou ajustes sempre que necessário, com o objetivo de proteger a confidencialidade e a privacidade dos dados corporativos. Essas modificações foram realizadas de forma criteriosa, de modo a não comprometer a integridade científica da pesquisa.

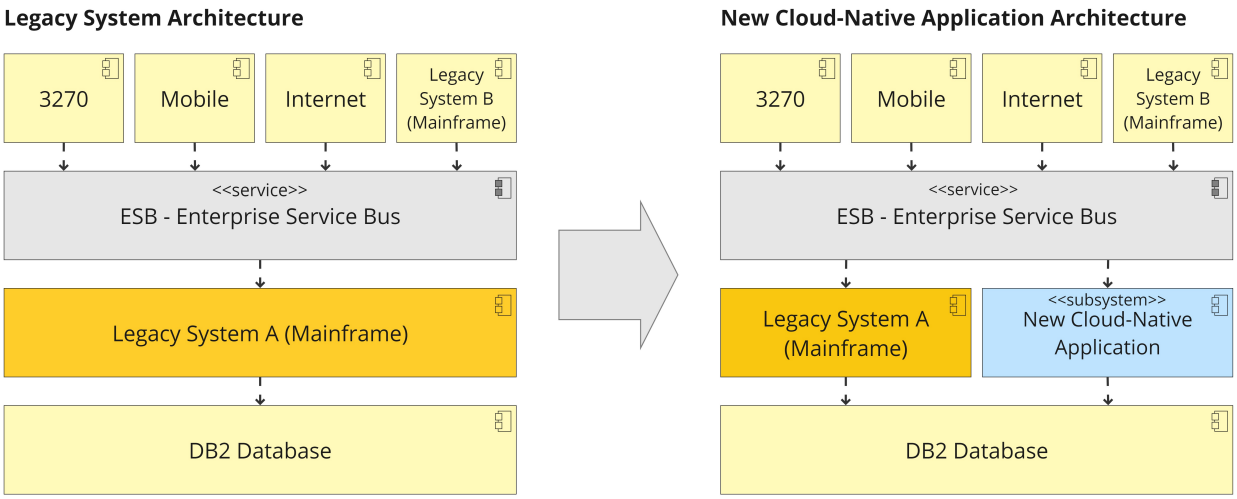


Figura 5.1: Sistema Legado.

Destaca-se a relevância de mencionar que neste estudo de caso, tanto a infraestrutura legada em mainframe quanto a nova infraestrutura em nuvem encontravam-se disponíveis durante o estudo de caso. Assim, o processo de modernização limitou-se à arquitetura da aplicação, não abrangendo a infraestrutura subjacente.

5.2 Motivações e Objetivos da Modernização

os principais motivos para a modernização das aplicações legadas, residem nos sinais de degradação de performance resultante do desgaste das aplicações e na crescente dificuldade e complexidade na sua manutenção e evolução [25]. As aplicações em questão, após um longo período em operação, acumularam novas funcionalidades e ajustes. Essa complexidade adquirida tornou qualquer modificação um processo arriscado, pois a previsão do impacto de imprevistos, mesmo com a realização de testes e validações, mostrava-se desafiadora [25].

Adicionalmente, a indisponibilidade de recursos proporcionados por tecnologias emergentes na arquitetura anterior, aliada à tendência da arquitetura legada, especialmente o Mainframe, de depender de um número restrito de fornecedores, restringia as opções e oportunidades organizacionais. Nesse contexto, essa característica não configurava uma escolha trivial,pois as aplicações legadas exibiam características de bloqueio de fornecedor (***vendor lock-in***). Mesmo quando um determinado fornecedor oferecia produtos e serviços de alta qualidade, essa concentração intensificava o risco estratégico do negócio, aspecto mitigado por várias organizações [127].

Assim foram estabelecidos objetivos pretendidos para modernização do sistema legado:

- Estabelecer o **Mapa de Domínio** da aplicação juntamente com seus respectivos subdomínios e a relação destes com os sistemas internos e externos [92].
- Propor uma reengenharia arquitetural da aplicação para uma arquitetura mais moderna e adequada aos domínios estabelecidos [29].
- Aplicar padrões arquiteturais: ***Architetural Design Pattern*** que estejam alinhados aos domínios estabelecidos, e habilitem a solução para portabilidade entre nuvens, tornando viável trabalhar com uma arquitetura ***Multi-Cloud***, mitigando um possível bloqueio de fornecedor ***vendor-lock-in*** [140] [29].
- Acelerar o desenvolvimento e entrega de novas funcionalidades através da adoção de uma cultura ***DevOps*** [137].
- Executar os trabalhos de modernização por fases de forma incremental, mitigando riscos e integrando cada entrega com o ecossistema da aplicação legada [25].

5.3 O Processo de Modernização

O processo de modernização impõe desafios na migração do processamento da aplicação e na migração dos dados históricos [22]. A integridade e a segurança dos dados são sensíveis, e falhas poderiam gerar repercussões significativas [25]. Por esse motivo, optou-se por uma migração conservadora, priorizando a integridade e a segurança durante a transição, com o objetivo de assegurar a continuidade das funções do sistema anterior. Inicialmente, migrou-se o processamento, mantendo os dados na estrutura original. Essa estratégia visou aumentar a confiança no novo sistema e reduzir riscos, preparando a futura migração dos dados [141].

A migração consistiu na fragmentação do sistema legado em blocos distintos, alocados a fases específicas do projeto de modernização. Após cada fase, a porção modernizada foi reintegrada ao ecossistema original, mantendo a harmonia com os demais sistemas e o comportamento anterior, conforme Figura 5.3.

A transição do código COBOL para Java, além de possibilitar a integração com microsserviços, permitiu a readequação arquitetural para explorar tecnologias emergentes. A colaboração entre desenvolvedores e especialistas de negócio assegurou a conformidade com regulamentações e o aumento da eficiência operacional. Já a análise do sistema legado revelou pontos de atrito devido ao alto acoplamento entre módulos COBOL, evidenciando o desgaste acumulado ao longo dos anos.

O plano de modernização para a arquitetura Cloud-Native compreendeu cinco etapas, subdivididas em fases e iterações. Cada fase, com escopo específico, contemplou tarefas de modernização (etapas 1 e 2, Figura 5.2). As iterações (etapas 3, 4 e 5), em ciclo

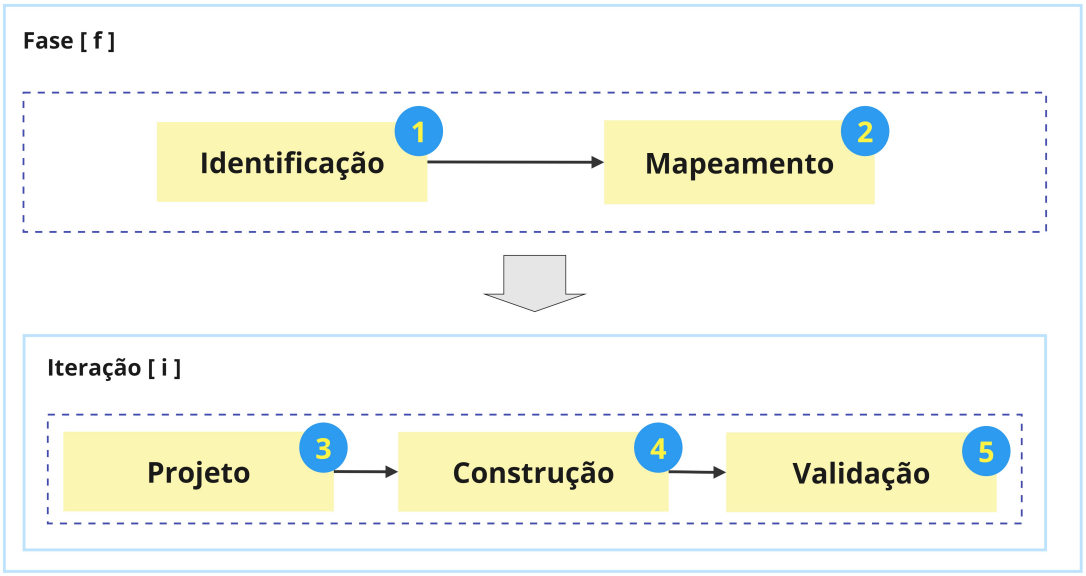


Figura 5.2: Processo de Modernização de Sistema Legado.

contínuo até a validação da modernização da fase, contemplaram a implementação do escopo definido. Concluída a fase, a equipe prosseguiu para a próxima fase, em um processo essencial para a migração para um ambiente moderno e escalável.

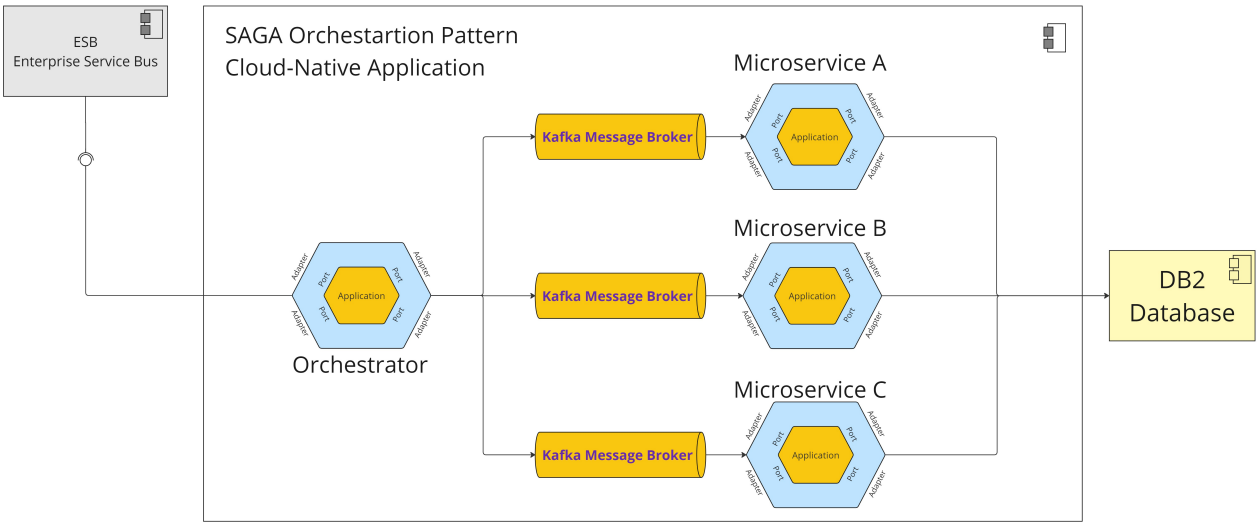


Figura 5.3: Sistema Modernizado.

Cada fase teve como propósito compreender as oportunidades e restrições da modernização e definir o escopo para tornar viável a tarefa de modernização, conduzida pelas suas iterações.

5.3.1 Fase 1

- **Primeira Etapa - *Identificação***, realizou-se um estudo do sistema legado, identificando funcionalidades e impactos no negócio, considerando a relevância do sistema e os riscos de obsolescência. O estudo incluiu reuniões com especialistas de negócio e da equipe de tecnologia, além da análise da documentação disponível.
- **Segunda Etapa - *Mapeamento***, o mapeamento das funcionalidades do sistema utilizou técnicas de Domain Driven Design (DDD) [30] e Event Storming [96]. Essa etapa viabilizou a modelagem estratégica, a identificação de domínios, funcionalidades e artefatos (eventos, comandos, entidades e agregados), e a definição do escopo das iterações subsequentes compostas pelas etapas 3, 4 e 5.

5.3.2 Iteração 1

- **Terceira Etapa - *Modelagem***, nessa atividade, a modelagem tática empregou Domain-Driven Design (DDD) e UML. A arquitetura SAGA Orchestration, selecionada com base na modelagem estratégica da etapa 2, atendeu ao requisito de fluxo de trabalho do sistema, que evidenciou o encadeamento de tarefas e atividades. A comunicação síncrona entre microsserviços via HTTP/REST, apesar da simplicidade, gerou dependências temporais. Contudo, o projeto previu a integração com o ecossistema legado, permitindo o uso de recursos modernizados e legados. A implementação dos microsserviços utilizou Java Quarkus com acesso ao DB2. AppDynamics, Prometheus e Grafana foram utilizados para monitoração.
- **Etapa Quatro - *Construção***, nesse estágio, a construção e a implantação da solução seguiram as diretrizes da terceira etapa, com atenção aos processos de governança e qualidade. A implementação utilizou o *SAGA Orchestration Pattern*, com a expectativa de que a estrutura de microsserviços permitisse o desenvolvimento, teste e escalabilidade independentes de funcionalidades, agilizando a implementação de atualizações e novos recursos [29]. Além disso a modularidade implementada com os microsserviços facilitou a manutenção e a resolução de problemas, isolando falhas e aumentando a eficiência do desenvolvimento em relação ao modelo tradicional [29].
- **Etapa Cinco - *Validação***, essa etapa teve participação central no processo. A validação do novo sistema confrontou seus resultados com o sistema legado, avaliando requisitos funcionais e não funcionais, como regras de negócio, desempenho e uso de recursos. O sucesso da validação determinou o avanço para a próxima etapa ou a realização de nova iteração. Os requisitos funcionais foram validados; porém, o desempenho mostrou-se insatisfatório, com lentidão na comunicação entre

o sistema modernizado e o legado. A comunicação síncrona impactou consideravelmente o tempo total de execução, acumulando o tempo de cada microsserviço. Além disso, o processamento dos microsserviços também apresentou oportunidades de melhoria, não atendendo à expectativa de performance esperada. ***Como resultado, a etapa de validação rejeitou a entrega da solução, com indicação de refatoração e reengenharia em uma nova iteração.***

5.3.3 Iteração 2

Diante do resultado negativo da Iteração 1, iniciou-se uma nova iteração, passando novamente pelas etapas de ***projeto***, ***construção*** e ***validação***. O foco principal desta iteração foi a correção dos problemas identificados no atendimento aos requisitos não funcionais, em especial o requisito de desempenho da aplicação. A seguir, detalham-se o processo e os resultados observados na Iteração 2.

- **Terceira etapa**, essa iteração revisou os trabalhos da Iteração 1, buscando melhorias e solucionando pontos de atenção. Confirmou-se a adequação da arquitetura *SAGA Orchestration* [29] para a gestão do fluxo de atividades. Entretanto, buscando flexibilidade e alinhamento com o *Domain-Driven Design (DDD)* [30], adotou-se o padrão arquitetural *Hexagonal* na implementação da arquitetura interna dos microsserviços. Essa decisão promoveu segregação entre o *framework* e o domínio da aplicação, resultando em implementação mais coesa. Assim, além da revisão das implementações, visando otimizar o desempenho, incorporou-se um componente de mensageria para processamento assíncrono e paralelo, favorecendo a escalabilidade da solução. Visando maior segurança na validação, a solução incorporou uma funcionalidade de simulação com dados reais, sem comprometer os dados de produção. Essa funcionalidade permitiu a revalidação da aplicação após o lançamento em produção, facilitando a análise e resolução de problemas, aumentando a confiança e mitigando riscos.
- **Etapa Quatro -** ***Construção***, iniciou-se a refatoração dos microsserviços, implementando a arquitetura Hexagonal [29] e o componente de mensageria *Kafka* [78]. Observou-se melhora no desempenho com o processamento paralelo dos microsserviços e a refatoração do código. Testes de desempenho com compilação nativa utilizando *GraalVM* [142] e *Quarkus Native* [136] foram conduzidos. A aplicação de exemplo foi submetida a 10.000 usuários simultâneos em um teste de estresse de 2 minutos utilizando o ***Apache JMeter*** [143], totalizando aproximadamente 8 milhões de requisições. Os resultados indicaram, uma melhoria no ganho de performance de quase 50% com média do tempo de resposta de 15ms medido na aplicação

em bytecode e 8ms medido na aplicação nativa, o ***throughput*** permaneceu estável com 60.185/s para o bytecode e 58.345/s para a nativa, e a quantidade de erros observados não apresentou variação significativa com 0,03% medido na aplicação em bytecode e 0,00% na aplicação nativa. No entanto, apesar dos resultados promissores da compilação nativa com *GraalVM*, a incompatibilidade de algumas bibliotecas tornou o processo complexo, levando à opção pela compilação tradicional para bytecode Java, com possibilidade de reavaliação futura.

- **Etapa Cinco - *Validação***, apesar dos requisitos funcionais terem sido validados na Iteração 1, o processo de validação foi repetido por precaução, com a confiança de que esses requisitos estariam plenamente atendidos. Ao avançar para a validação dos requisitos não funcionais de desempenho, a aplicação demonstrou o comportamento esperado. Com isso, o módulo da aplicação foi validado e autorizado para implantação em produção, permitindo o início da próxima fase de modernização de um módulo subsequente.

5.4 Lições Aprendidas

A modernização de aplicações legadas para arquitetura Cloud-Native apresenta desafios técnicos e subjetivos, com impactos significativos. As principais lições aprendidas nesse processo são descritas a seguir.

- **Mudança de Paradigmas**: A análise das tecnologias evidencia que suas particularidades as distinguem, sem implicar em julgamento de superioridade ou inferioridade. O aspecto central identificado foi a relevância de estar aberto a mudanças de paradigmas, visto que a aplicação bem-sucedida de uma arquitetura em um contexto específico pode não gerar os mesmos resultados quando adaptada para uma arquitetura alvo de modernização. Além do mais, decisões equivocadas, influenciadas por paradigmas anteriores, como a adoção de processamento sequencial e comunicação síncrona em arquitetura distribuída, demandaram revisão em iterações posteriores.
- **Mitigar o Risco com Aprendizagem Contínua**: A modernização do sistema legado, dada a sua relevância e valor para a organização, gerou a expectativa de que o novo sistema mantivesse, ou superasse, a eficiência do original. Essa expectativa resultou em ansiedade para a equipe responsável pela modernização. A mitigação dos riscos dessa transição se deu por meio da modernização em fases, com iterações sucessivas. Essa abordagem permitiu a aprendizagem contínua, a repetição de práticas eficazes e a correção de erros, minimizando impactos negativos. Por exemplo, a inclusão do componente de mensageria (Kafka) e a refatoração para o

modelo de Arquitetura Hexagonal, além de correções de código para melhorar a performance, foram implementadas nas primeiras iterações, antes que o escopo do trabalho impedisse correções eficazes.

- **Documentação e Modelo do Domínio**: As duas etapas iniciais do processo de modernização estabeleceram o escopo e a direção dos trabalhos subsequentes, definindo a direção das etapas 3, 4 e 5. Ferramentas de modelagem como Domain-Driven Design (DDD) e Event Storming mostraram-se essenciais para a definição da amplitude do trabalho e o direcionamento da equipe. A documentação de qualidade, mesmo em ambientes ágeis, consolidou-se como fator essencial para o sucesso do projeto.
- **Solucionando o Problema**: A especificidade de cada problema requer soluções únicas, demandando do especialista em tecnologia ou do arquiteto de soluções, a identificação da solução mais apropriada para cada questão. Assim, evidencia-se que o conhecimento aprofundado da necessidade, e das ferramentas disponíveis mostram-se essenciais para a proposição de soluções eficazes. Uma abordagem de pequenos passos, com correções rápidas durante o processo, mostrou-se eficaz.

5.5 Resultados

Nessa fase da pesquisa, foi relatado experiência vivenciada durante a modernização de um sistema legado em mainframe para uma solução cloud-native. Foram apresentadas as arquiteturas do sistema antes e depois da migração, assim como as etapas que compuseram o processo de modernização. Destacou-se a importância do conhecimento profundo sobre o sistema legado, essencial para identificar oportunidades e delimitar com precisão o escopo das atividades de migração. Enfatizou-se a necessidade de projetar uma arquitetura adequada para a solução do problema, sendo essa fase do desenho arquitetural determinante para o sucesso do projeto. Planejou-se consolidar essas práticas e desenvolver um conjunto de padrões reutilizáveis para a migração de aplicativos legados para uma arquitetura cloud-native. Acredita-se que essa contribuição auxiliará as equipes envolvidas nessa tarefa, pavimentando um caminho mais seguro na jornada de modernização de sistemas legados.

Capítulo 6

Conclusão

A pesquisa focou na elucidação dos desafios da migração de aplicações legadas para a Arquitetura Multi-Cloud a partir de duas iniciativas. Primeiramente, foi realizada uma Revisão Sistemática da Literatura que identificou como desafios principais: o **Gerenciamento da Infraestrutura Multi-cloud**, a **Arquitetura das Aplicações Nativas para Nuvem** e o **vendor lock-in**. Esses fatores evidenciaram o potencial limitador para a portabilidade das aplicações entre múltiplas nuvens de fornecedores distintos.

Conforme a revisão, as aplicações projetadas para arquiteturas nativas da nuvem, podem mitigar os riscos de dependência de tecnologias proprietárias, promovendo flexibilidade e resiliência. Contudo, a literatura revelou lacunas nas diretrizes arquiteturais sobre esses temas, indicando a necessidade de mais pesquisas e metodologias para gerenciar implementações complexas em ambientes Multi-Cloud. Na segunda parte da pesquisa, foi realizado um estudo de caso na migração de uma aplicação de uma empresa financeira latino-americana, para validar os fatores delineados na revisão sistemática. O estudo de caso validou esses achados, demonstrando que equívocos no desenho arquitetural podem comprometer o sucesso do projeto, como observado na necessidade de reescrever implementações na primeira iteração do desenvolvimento. A experiência evidenciou a importância do planejamento detalhado, com conhecimento dos estilos arquiteturais nativos para nuvem e do sistema legado. A integração da aplicação modernizada com o ecossistema do sistema modernizado foi outro aspecto destacado, pois se mostrou essencial para garantir uma transição tranquila e minimizar o risco de interrupções operacionais. Essa integração permitiu testar novas partes do sistema em tempo real, garantindo que todas as funcionalidades sejam validadas de forma eficaz antes da implementação completa. A abordagem não só protege, mas facilita a adaptação gradual dos processos internos à nova infraestrutura.

Durante o processo de modernização, o aprendizado para as equipes envolvidas mostrou-se valoroso. Evidenciando a experiência adquirida na superação de desafios técnicos e

operacionais, contribuiu para o desenvolvimento profissional dos membros da equipe e aumentou a resiliência organizacional. Além disso, a experiência adquirida preparou a empresa para futuros projetos de modernização, pois as lições aprendidas foram aplicadas para evitar erros do passado e otimizar abordagens futuras.

Em última análise, a migração do mainframe para a nuvem proporciona diversas oportunidades, tanto para os avanços tecnológicos quanto para a inovação nos modelos de negócios e na prestação de serviços. À medida que as empresas tornam-se mais ágeis e adaptáveis, estão melhor preparadas para enfrentar os desafios futuros e explorar novas oportunidades no cenário digital em transformação.

Concluiu-se que o sucesso na migração para arquiteturas Multi-Cloud ou Cloud-Native depende da combinação de estratégias técnicas e de negócios. Assim, a mitigação do bloqueio de fornecedor, o desenvolvimento de diretrizes arquiteturais e a adaptação ao contexto dos sistemas legados são elementos fundamentais para a superação dos desafios identificados.

Tendências Futuras em Arquitetura Multi-Cloud

No contexto da arquitetura de aplicações, as tendências futuras para multi-cloud apontam para uma **maior abstração e portabilidade**. Com a possibilidade de observar um aumento na adoção de tecnologias **cloud-native** com aplicações aderentes a padrões arquiteturais Cloud-Native, implementadas em contêineres (*Docker, Kubernetes*) e microsserviços, **que facilitam a implantação e a movimentação de aplicações entre diferentes provedores de nuvem**. Além disso, a integração de inteligência artificial (IA) e machine learning (ML) diretamente nas plataformas de nuvem poderá impulsionar a criação de aplicações mais inteligentes e adaptáveis, sendo capazes de aproveitar os serviços especializados de cada provedor. O grande desafio será garantir a interoperabilidade e a consistência entre esses diferentes ambientes, exigindo ferramentas e práticas de desenvolvimento que considerem a natureza distribuída e heterogênea da multi-cloud.

Quanto à arquitetura no contexto de infraestrutura multi-cloud, a tendência é a busca por uma gestão unificada e simplificada. Ferramentas de orquestração e automação multi-cloud (*Ansible, Terraform*) serão essenciais para permitir o provisionamento, a configuração e o monitoramento de recursos em diversas nuvens a partir de um único ponto de controle. Já a segurança e a governança em um ambiente multi-cloud complexo continuarão sendo prioridades, com o desenvolvimento de soluções que ofereçam visibilidade e controle consistentes sobre a identidade, o acesso e a conformidade em todas as plataformas. A adoção de redes multi-cloud e a interconexão direta entre provedores também tendem a se tornar mais eficientes, visando otimizar a latência e o desempenho das aplicações distribuídas. Os principais benefícios futuros incluem maior resiliência, flexibilidade

na escolha dos melhores serviços de cada provedor e otimização de custos, enquanto os desafios residirão na complexidade da gestão, na garantia da segurança e na necessidade de desenvolver habilidades especializadas em múltiplas plataformas.

Nesta pesquisa foi possível observar a necessidade das empresas acelerarem sua preparação para entrar no mundo multi-cloud por diversos motivos estratégicos. Primeiramente, a multi-cloud oferece maior resiliência e evita a dependência de um único fornecedor (**vendor lock-in**), garantindo a continuidade dos negócios em caso de falhas ou interrupções em um provedor específico. Em segundo lugar, permite que a empresa escolha os melhores serviços de cada plataforma para atender às suas necessidades específicas, otimizando custos e desempenho. Por exemplo, uma empresa pode usar um provedor para armazenamento de dados com preços competitivos e outro para serviços de IA/ML avançados. Além disso, a multi-cloud facilita a conformidade com regulamentações regionais, a expansão geográfica e a proteção de instabilidades geopolíticas, permitindo que os dados e as aplicações sejam hospedados em locais que atendam às exigências legais e estratégicas. Preparar-se para a multi-cloud envolve investir em treinamento de equipes, adotar ferramentas de gestão e orquestração multi-cloud e desenvolver uma estratégia clara para a distribuição de cargas de trabalho entre os diferentes provedores, garantindo assim uma transição suave e a maximização dos benefícios dessa arquitetura.

Referências

[1] Ross, Jeanne W, Cynthia M Beath e Martin Mocker: *Designed for digital: How to architect your business for sustained success*. Mit Press, 2019. 1

[2] Hosseini Shirvani, Mirsaeid, Gholam R Amin e Sara Babaeikiadehi: *A decision framework for cloud migration: A hybrid approach*. IET software, 16(6):603–629, 2022. 1

[3] Fard, Mostafa Vakili, Amir Sahafi, Amir Masoud Rahmani e Peyman Sheikholharam Mashhadi: *Resource allocation mechanisms in cloud computing: a systematic literature review*. IET Software, 14(6):638–653, 2020. 1

[4] Caceres, A. e L. Globa: *State-of-the-art architectures for interoperability of heterogeneous clouds*. Em *Proceedings - 16th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2022*, páginas 704–709, 2022, ISBN 9781665468619. 1, 9, 42, 74, 75, 76, 77, 78, 79

[5] Bainomugisha, E. e A. Mwotil: *Crane cloud: A resilient multi-cloud service abstraction layer for resource-constrained settings*. Development Engineering, 7, 2022. 1, 2, 20, 39, 40, 41, 42, 44, 46, 74, 75, 76, 77, 78, 79

[6] Ardagna, D: *Cloud and multi-cloud computing: Current challenges and future applications*. páginas 1–2, 2015, ISBN 2156-793X. `https://ieeexplore-ieee-org.ez54.periodicos.capes.gov.br/document/7172841`. 2, 43, 44, 45, 75, 76, 77, 78

[7] Shirvani, M. Hosseini, A.M. Rahmani e A. Sahafi: *An iterative mathematical decision model for cloud migration: A cost and security risk approach*. Software - Practice and Experience, 48:449–485, 2018. 2, 74, 75, 76, 77, 78, 79

[8] Gholami, M.F., F. Daneshgar, G. Low e G. Beydoun: *Cloud migration process—a survey, evaluation framework, and open challenges*. Journal of Systems and Software, 120:31–69, 2016. 2, 19, 20, 39, 45, 46, 47, 74, 75, 76, 77, 78, 79

[9] Asthana, S., A. Megahed e I. Iyoob: *Multi-cloud Solution Design for Migrating a Portfolio of Applications to the Cloud*, volume 12632 LNCS. 2021, ISBN 9783030763510. 2, 8, 37, 38, 43, 44, 45, 75, 76, 77, 78

[10] Shastry, A.L., D.S. Nair, B. Prathima, C.P. Ramya e P. Hallymysore: *Approaches for migrating non cloud-native applications to the cloud*. páginas 632–638, 2022, ISBN 9781665483032. 2, 74, 75, 76, 77, 78, 79

[11] Mateen, A., S.Y. Nam, M.A. Haider e A. Hanan: *A dynamic decision support system for selection of cloud storage provider*. Applied Sciences (Switzerland), 11, 2021. `https://www.mdpi.com/1381328`. 2, 9, 11, 39, 45, 46, 47, 74, 75, 76, 77, 78, 79

[12] Carvalho, J.O. De, F. Trinta e D. Vieira: *Pacificclouds: A flexible microservices based architecture for interoperability in multi-cloud environments*. Volume 2018-Janua, páginas 448–455, 2018, ISBN 9789897582950. 2, 38, 47, 74, 75, 76, 77, 78, 79

[13] Alonso, Juncal, Leire Orue-Echevarria, Valentina Casola, Ana Isabel Torre, Maider Huarte, Eneko Osaba e Jesus L Lobo: *Understanding the challenges and novel architectural models of multi-cloud native applications–a systematic literature review*. Journal of Cloud Computing, 12(1):1–34, 2023. 2, 3, 4, 37, 42

[14] Maniah, B. Soewito, F. Lumban Gaol e E. Abdurachman: *A systematic literature review: Risk analysis in cloud migration*. Journal of King Saud University - Computer and Information Sciences, 34:3111–3120, 2022. 2

[15] Paula, A.C.M. de e G.F. de Carneiro: *A systematic literature review on cloud computing adoption and migration*, volume 703. 2016, ISBN 9783319563893. 2

[16] Jamshidi, P., A. Ahmad e C. Pahl: *Cloud migration research: A systematic review*. IEEE Transactions on Cloud Computing, 1:142–157, 2013. 2, 12, 30, 31, 47

[17] *Cloud computing adoption, cost-benefit relationship and strategies for selecting providers: A systematic review*. páginas 27–39, 2016. 2, 29, 74

[18] Marcotte, P., F. Gregoire e F. Petrillo: *Multiple fault-tolerance mechanisms in cloud systems: A systematic review*. páginas 414–421, 2019, ISBN 9781728151380. 2

[19] Patel, M., A. Mehta e S. Patel: *Container migration and placement in hybrid cloud-fog environment: Systematic review*. International Journal on Technical and Physical Problems of Engineering, 14:130–135, 2022. 2

[20] Mulder, Jeroen: *Multi-Cloud Architecture and Governance*. Packt Publishing, 2020, ISBN 9781800203198. 4, 14

[21] Ross, Jeanne W., Cynthia M. Beath e Martin Mocker: *Designed for Digital How to Architect your Business for Susteained Sucess*. Massachusetts Institute Technology, 2019, ISBN 9780262042888. 4

[22] Abgaz, Yalemisew, Andrew McCarren, Peter Elger, David Solan, Neil Lapuz, Marin Bivol, Glenn Jackson, Murat Yilmaz, Jim Buckley e Paul Clarke: *Decomposition of monolith applications into microservices architectures: A systematic review*. IEEE Transactions on Software Engineering, 49(8):4213–4242, 2023. 4, 52

[23] Kratzke, Nane e Peter Christian Quint: *Understanding cloud-native applications after 10 years of cloud computing - a systematic mapping study*. The Journal of systems and software, 126:1–16, 2017, ISSN 0164-1212. 4, 20, 30, 39, 41, 42, 45, 74, 75, 76, 77, 78, 79

[24] Belafia, R., P. Jeanjean, O. Barais, G.L. Guernic e B. Combemale: *From monolithic to microservice architecture: The case of extensible and domain-specific ides.* páginas 454–463, 2021, ISBN 9781665424844. 4, 20, 74, 75, 76, 77, 78

[25] Knoche, Holger e Wilhelm Hasselbring: *Using microservices for legacy software modernization.* IEEE software, 35(3):44–49, 2018. 4, 51, 52

[26] Rai, R., G. Sahoo e S. Mehfuz: *Exploring the factors influencing the cloud computing adoption: a systematic study on cloud migration.* SpringerPlus, 4:1–12, 2015. 4, 74, 75, 76, 77, 78

[27] vmware: *O que é multi-cloud?* `https://www.vmware.com/br/topics/glossary/content/multi-cloud.html`, April 2022. 5

[28] Asthana, S., A. Megahed e I. Iyoob: *A cloud-migration feasibility advisor.* Volume 2020-Novem, páginas 1171–1172, 2020, ISBN 9781728170022. `https://ieeexplore.ieee.org/document/9355695`. 8

[29] Richardson, Chris: *Microservices patterns: with examples in Java.* Simon and Schuster, 2018. 8, 26, 27, 49, 50, 52, 54, 55

[30] Khononov, Vlad: *Learning Domain-Driven Design.* " O'Reilly Media, Inc.", 2021. 8, 24, 25, 26, 54, 55

[31] Ahmad, N., S. Qamar, N. Khan, A. Naim, M.R. Hussain, Q.N. Naveed e M.R. Mahmood: *Cloud Computing Trends and Cloud Migration Tuple*, volume 107. 2020. 9, 38, 40, 41, 46, 74, 75, 76, 77, 78

[32] Baby, Kiran e Anupriya Vysala: *Multicloud architecture for augmenting security in clouds.* Em *2015 global conference on communication technologies (GCCT)*, páginas 474–478. IEEE, 2015. 9, 75, 76, 77, 78

[33] Brogi, A., J. Carrasco, J. Cubo, F. D Andria, A. Ibrahim, E. Pimentel e J. Soldani: *Eu project seaclouds adaptive management of service-based applications across multiple clouds.* páginas 758–763, 2014. 9, 74, 75, 77

[34] Microsoft: *What is a private cloud?* https://azure.microsoft.com/en-in/resources/cloud-computing-dictionary/what-is-a-private-cloud/, 2023. Accessed: 2023-03-27 at 07:35 PM. 9

[35] Mulder, Jeroen: *Multi-Cloud Strategy for Cloud Architects - Second Edition.* Packt Publishing Ltd, 2023. 9, 11

[36] Google: *What is a public cloud?* https://cloud.google.com/learn/what-is-public-cloud/, 2023. Accessed: 2023-03-27 at 07:57 PM. 9, 10, 44

[37] Hirway, Manoj: *Hybrid Cloud for Developers: Develop and deploy cost-effective applications on the AWS and OpenStack platforms with ease.* Packt Publishing Ltd, 2018. 10

[38] Habiba, Mansura: *Hybrid Cloud Infrastructure and Operations Explained*. Packt Publishing, 2022. 10, 38

[39] IBM: *What is a hybrid cloud?* https://www.ibm.com/topics/hybrid-cloud, 2023. Accessed: 2023-03-27 at 08:18 PM. 10

[40] IBM: *What is multicloud?* https://www.ibm.com/topics/ multicloud, 2023. Accessed: 2023-03-27 at 08:22 PM. 11

[41] Jamshidi, P., C. Pahl e N.C. Mendonça: *Pattern-based multi-cloud architecture migration*. Software - Practice and Experience, 47:1159–1184, 2017. 11, 19, 30, 42, 44, 47, 74, 75, 76, 77, 78, 79

[42] Mulder, Jeroen: *Multi-Cloud Strategy for Cloud Architects - Second Edition*. Packt Publishing Ltd, 2023. 11, 12, 13, 14, 20, 21, 22, 23

[43] Brian, Joe: *Hands-On Multi-Cloud Kuberbetes: Multi-cluster kubernetes deployment and scaling with FluxCD, Virtual Kubelet, Submariner and KubeFed*. Leanpub, 2023. 11, 12, 13, 14, 15

[44] Raj, Jennifer S.: *Efficient information maintenance using computational intelligence in the multi-cloud architecture*. Journal of Soft Computing Paradigm, 2019. 11

[45] Perumal, Arun Pandiyan: *Improving operational efficiency and productivity through the fusion of devops and sre practices in multi-cloud operations*. International Journal of Cloud Computing and Database Management, 2022. 11

[46] Mamidi, Sundeep Reddy: *Securing multi-cloud architectures: A machine learning perspective*. Jaigs, 2024. 11, 12

[47] Sharma, Akrati: *Developing a novel load balancing approach for mitigating resource contention in multi-cloud environments*. Jes, 2024. 11

[48] Voruganti, Kiran Kumar: *Orchestrating multi-cloud environments for enhanced flexibility and resilience*. Journal of Technology and Systems, 2024. 11, 12

[49] Tomarchio, Orazio, Domenico Calcaterra e Giuseppe Modica: *Cloud resource orchestration in the multi-cloud landscape: A systematic review of existing frameworks*. Journal of Cloud Computing Advances Systems and Applications, 2020. 12

[50] Gundu, Srinivasa Rao, Charan Arur Panem e Anuradha Thimmapuram: *Hybrid it and multi cloud an emerging trend and improved performance in cloud computing*. Sn Computer Science, 2020. 12

[51] Paladi, Nicolae, Antonis Michalas e HaiVan Dang: *Towards secure cloud orchestration for multi-cloud deployments*. 2018. 12

[52] Tricomi, Giuseppe, Giovanni Merlino, Alfonso Panarello e Antonio Puliafito: *Optimal selection techniques for cloud service providers*. Ieee Access, 2020. 12

[53] CNCF: *Cloud native computing foundation*, 2024. `https://www.cncf.io/`, Accessed: 2024-12-28 at 16:14 PM. 12

[54] Zeng, Sen, Guo Qi Ni, Miao Fan, Zheshuai Lin e Yuan He: *A service selection algorithm based on heterogeneous qos model and synthetic weights*. Applied Mechanics and Materials, 2015. 13

[55] Singh, D. e R. Venkata Rao: *A hybrid multiple attribute decision making method for solving problems of industrial environment*. International Journal of Industrial Engineering Computations, 2011. 13

[56] Lanjewar, Pramod B., R. Venkata Rao, A. V. Kale e Paweł Ocłoń: *Evaluation and selection of energy technologies using an integrated graph theory and analytic hierarchy process methods*. Decision Science Letters, 2016. 13

[57] Rancher: *What is rancher?* https://ranchermanager.docs.rancher.com/, 2024. Accessed: 2024-06-24 at 14:03 PM. 14, 16

[58] kubeadm: *Kubeadm*, 2024. `https://kubernetes.io/pt-br/docs/reference/setup-tools/kubeadm/`, Accessed: 2024-12-28 at 17:19 PM. 14

[59] kOps: *kops - kubernetes operations*, 2024. `https://kops.sigs.k8s.io/`, Accessed: 2024-12-28 at 17:19 PM. 14

[60] istio: *Istio service mesh. simplified. easily build cloud native workloads securely and reliably with istio, with or without sidecars.*, 2024. `https://istio.io/`, Accessed: 2024-12-28 at 17:19 PM. 14

[61] kubedirector: *Kubedirector*, 2024. `https://github.com/bluek8s/kubedirector/tree/master`, Accessed: 2024-12-28 at 17:19 PM. 14

[62] redhat: *Red hat ansible automation platform*, 2024. `https://www.redhat.com/en/technologies/management/ansible`, Accessed: 2024-12-28 at 17:19 PM. 14

[63] HashiCorp: *Terraform community: Automate infrastructure on any cloud with terraform*, 2024. `https://www.redhat.com/en/technologies/management/ansible`, Accessed: 2024-12-28 at 17:19 PM. 14

[64] Helm: *Helm - the package manager for kubernetes*, 2024. `https://helm.sh/`, Accessed: 2024-12-28 at 17:19 PM. 15

[65] Lingamallu, Phani Kumar e Fabio Oliveira: *AWS Observability Handbook: Monitor, trace, and alert your cloud applications with AWS'myriad observability tools*. Packt Publishing Ltd, 2023. 15

[66] Boten, Alex: *Cloud-Native Observability with OpenTelemetry: Learn to gain visibility into systems by combining tracing, metrics, and logging with OpenTelemetry*. PACKT PUBLISHING LIMITED, 2022. 15

[67] Mulder, Jeroen: *Multi-Cloud Architecture and Governance: Leverage Azure, AWS, GCP, and VMware vSphere to build effective multi-cloud solutions*. Packt Publishing Ltd, 2020. 16

[68] RKE2: *Rke2*, 2024. https://docs.rke2.io/, Accessed: 2024-12-28 at 16:14 PM. 16

[69] Harvester: *Harvester - the open-source hyperconverged infrastructure solution for a cloud-native world*, 2025. https://harvesterhci.io/, Accessed: 2025-01-04 at 17:19 PM. 16

[70] Longhorn: *Longhorn - longhorn cloud native distributed block storage for kubernetes easy to use, 100% open source, run anywhere*, 2025. https://longhorn.io/, Accessed: 2025-01-04 at 18:46 PM. 16

[71] Prometheus: *Prometheus - from metrics to insight power your metrics and alerting with the leading open-source monitoring solution.*, 2025. https://prometheus.io/, Accessed: 2025-01-04 at 18:46 PM. 16

[72] Pai, Karthik e B.K Srinivas: *Enhanced visibility for real-time monitoring and alerting in kubernetes by integrating prometheus,prometheus, grafana, loki, and alerta.* Interantional Journal of Scientific Research in Engineering and Management, 08:1–5, 2024. 16, 17

[73] GrafanaLabs: *Grafana - compose and scale your observability with one, some, or all of the grafana stack pieces.*, 2025. https://grafana.com/, Accessed: 2025-01-04 at 18:46 PM. 17

[74] Nginx: *Nginx project.*, 2025. https://nginx.org/, Accessed: 2025-01-04 at 19:24 PM. 17

[75] Kubernetes: *Kubernetes - production-grade container orchestration.* https://kubernetes.io/, 2024. Accessed: 2024-06-24 at 14:00 PM. 17

[76] Foundation, Linux: *The linux foundation - decentralized innovation. built on trust.*, 2025. https://www.linuxfoundation.org/, Accessed: 2025-01-04 at 19:48 PM. 17

[77] wso2: *Wso2 api managemer - the 1 open source api management platform.*, 2025. https://wso2.com/api-manager/, Accessed: 2025-01-04 at 19:48 PM. 17

[78] Kafka: *Apache kafka.* https://kafka.apache.org, 2024. Accessed: 2024-08-15 at 17:09 PM. 17, 55

[79] ArgoCD: *Argo cd declarative continuous delivery with a fully-loaded ui.*, 2025. https://argoproj.github.io/cd/, Accessed: 2025-01-04 at 19:48 PM. 17

[80] ArgoCD: *Jenkins build great things at any scale the leading open source automation server, jenkins provides hundreds of plugins to support building, deploying and automating any project.*, 2025. https://www.jenkins.io/, Accessed: 2025-01-04 at 19:48 PM. 18

[81] Kubernetes: *Opa gatekeeper: Política e governança para kubernetes*, 2025. https://kubernetes.io/blog/2019/08/06/opa-gatekeeper-policy-and-governance-for-kubernetes/, Accessed: 2025-01-04 at 19:48 PM. 18

[82] OpaGatekeeper: *Opa gatekeeper: A customizable cloud native policy controller that helps enforce policies and strengthen governance*, 2025. `https://open-policy-agent.github.io/gatekeeper/website/`, Accessed: 2025-01-04 at 19:48 PM. 18

[83] Harbor: *Harbor: Our mission is to be the trusted cloud native repository for kubernetes*, 2025. `https://goharbor.io/`, Accessed: 2025-01-04 at 19:48 PM. 18

[84] Cassandra: *Cassandra: Open source nosql database: Manage massive amounts of data, fast, without losing sleep*, 2025. `https://cassandra.apache.org/_/index.html`, Accessed: 2025-01-05 at 11:34 PM. 18

[85] Haugeland, S.G., P.H. Nguyen, H. Song e F. Chauvel: *Migrating monoliths to microservices-based customizable multi-tenant cloud-native apps*. páginas 170–177, 2021, ISBN 9781665427050. 19, 20, 39, 47, 74, 75, 76, 77, 78, 79

[86] Weerasinghe, S. e I. Perera: *Taxonomical classification and systematic review on microservices*. International Journal of Engineering Trends and Technology, 70:222–233, 2022. 19, 20, 42

[87] Jambunathan, B. e K. Yoganathan: *Architecture decision on using microservices or serverless functions with containers*. 2018, ISBN 9781538637012. 20, 39, 40, 74, 75, 76, 77, 78, 79

[88] Bulgu, Aykut, Eduardo Ramírez Ronco, Jaime Ramírez Castillo, Marek Czernek e Pablo Solar Vilariño: *Red Hat Build of Quarkus 2.13 DO378 - Red Hat Cloud-native Microservices Development with Quarkus*. Red Hat, 2023. 20

[89] 12factorApp: *The twelve-factor app*, 2025. `https://12factor.net/`, Accessed: 2025-01-23 at 21:00 PM. 23, 24

[90] Fowler, Martin: *Patterns of enterprise application architecture*. Addison-Wesley, 2012. 23

[91] Fowler, Martin: *Refactoring: improving the design of existing code*. Addison-Wesley Professional, 2018. 23

[92] Evans, Eric: *Domain-driven design: tackling complexity in the heart of software*. Addison-Wesley Professional, 2004. 24, 25, 26, 49, 50, 52

[93] Jagtap, Sandeep: *Domain-Driven Design And Microservices Explained with Examples*. Leanpub, 2023. 24

[94] Vernon, Vaughn: *Implementing domain-driven design*. Addison-Wesley, 2013. 24

[95] Vernon, Vaughn: *Domain-driven design distilled*. Addison-Wesley Professional, 2016. 24, 25, 26

[96] Brandolini, Alberto: *Introducing EventStorming: An act of Deliberate Collective Learning*. Leanpub, 2021. 24, 49, 54

[97] Rogério, Anderson: *Event Storming em Ação, Modelando Sistemas Complexos*. Safe-Courses, 2023. 24, 25

[98] Newman, Sam: *Building microservices*. " O'Reilly Media, Inc.", 2021. 26, 27

[99] Bellemare, Adam: *Building event-driven microservices*. " O'Reilly Media, Inc.", 2020. 27

[100] Garcia-Molina, Hector e Kenneth Salem: *Sagas*. ACM Sigmod Record, 16(3):249–259, 1987. 27

[101] Luciano, Bruno Theodoro: *Implementing cross-platform business logic in mobile applications using hexagonal architecture*, 2023. 27, 28

[102] Achimugu, P., B. Afolabi, O. Adeniran, I. Gambo e O. Oluwagbemi: *Software architecture and methodology as a tool for efficient software engineering process: a critical appraisal*. Journal of Software Engineering and Applications, 03:933–938, 2010. 28

[103] Dasanayake, S., S. Aaramaa, J. Markkula e M. Oivo: *Impact of requirements volatility on software architecture: how do software teams keep up with ever-changing requirements?* Journal of Software: Evolution and Process, 31, 2019. 28

[104] Power, K. e K. Conboy: *A metric-based approach to managing architecture-related impediments in product development flow: an industry case study from cisco*. 2015 IEEE/ACM 2nd International Workshop on Software Architecture and Metrics, páginas 15–21, 2015. 28

[105] Ataei, P. e A. T. Litchfield: *Neomycelia: a software reference architecturefor big data systems*. 2021 28th Asia-Pacific Software Engineering Conference (APSEC), 2:452–462, 2021. 28

[106] Nunkesser, Robin: *Using hexagonal architecture for mobile applications.* Em *ICSOFT*, páginas 113–120, 2022. 28

[107] Chen, X., W. Zhang, P. Liang e K. He: *A replicated experiment on architecture pattern recommendation based on quality requirements*. 2014 IEEE 5th International Conference on Software Engineering and Service Science, 2014. 28

[108] Cockburn, Alistair e Juan Manuel Garrido de Paz: *Hexagonal Architecture Explained*. Humans and Technology Inc (May 8, 2024), 2018. 28

[109] Jamshidi, P., C. Pahl, S. Chinenyeze e X. Liu: *Cloud migration patterns: A multi-cloud service architecture perspective*, volume 8954. 2015, ISBN 9783319228846. 30, 74, 75, 76, 77, 78, 79

[110] Brereton, Pearl, Barbara A Kitchenham, David Budgen, Mark Turner e Mohamed Khalil: *Lessons from applying the systematic literature review process within the software engineering domain*. Journal of systems and software, 80(4):571–583, 2007. 31

[111] Tranfield, David, David Denyer e Palminder Smart: *Towards a methodology for developing evidence-informed management knowledge by means of systematic review.* British journal of management, 14(3):207–222, 2003. 31, 34

[112] Wohlin, Claes: *Guidelines for snowballing in systematic literature studies and a replication in software engineering.* Em *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, páginas 1–10, 2014. 32, 33, 37

[113] Liberati, Alessandro, Douglas G Altman, Jennifer Tetzlaff, Cynthia Mulrow, Peter C Gøtzsche, John PA Ioannidis, Mike Clarke, Philip J Devereaux, Jos Kleijnen e David Moher: *The prisma statement for reporting systematic reviews and meta-analyses of studies that evaluate health care interventions: explanation and elaboration.* Annals of internal medicine, 151(4):W–65, 2009. 34

[114] Wiesche, Manuel, Marlen C. Jurisch, Philip W. Yetton e Helmut Krcmar: *Grounded theory methodology in information systems research.* MIS Quarterly, 41(3):pp. 685–702, A1–A9, 2017, ISSN 02767783, 21629730. `https://www.jstor.org/stable/26635009`, acesso em 2023-07-24. 34

[115] Charmaz, Kathy: *Constructing grounded theory: A practical guide through qualitative analysis.* sage, 2006. 34

[116] Neergaard, Helle e John P Ulhøi: *Handbook of qualitative research methods in entrepreneurship.* Edward Elgar Publishing, 2007. 35

[117] Sarmento, Manuel Jacinto: *O estudo de caso etnográfico em educação.* 2011. 35

[118] Aniche, Maurício, Joseph Yoder e Fabio Kon: *Current challenges in practical object-oriented software design.* Em *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, páginas 113–116. IEEE, 2019. 35

[119] André, Marli: *Etnografia da prática escolar.* Papirus editora, 2013. 36

[120] Wolcott, Harry F: *Ethnography: A way of seeing.* Rowman Altamira, 1999. 36

[121] Hammersley, Martyn e Paul Atkinson: *Etnografia: Princípios em prática.* Editora Vozes, 2022. 36

[122] Brogi, Antonio, Michela Fazzolari, Ahmad Ibrahim, Jacopo Soldani, Jose Carrasco, Javier Cubo, Francisco Duran, Ernesto Pimentel, Elisabetta Di Nitto e Francesco D Andria: *Adaptive management of applications across multiple clouds: The seaclouds approach.* CLEI electronic journal, 18(1):2–2, 2015. 38, 44, 74, 75, 76, 77, 78, 79

[123] Aydin, H.: *A study of cloud computing adoption in universities as a guideline to cloud migration.* SAGE Open, 11, 2021. 39, 74, 75, 76, 77, 78

[124] Ranchal, R., P. Bastide, X. Wang, A. Gkoulalas-Divanis, M. Mehra, S. Bakthavachalam, H. Lei e A. Mohindra: *Disrupting healthcare silos: Addressing data volume, velocity and variety with a cloud-native healthcare data ingestion service.* IEEE Journal of Biomedical and Health Informatics, 24:3182–3188, 2020. 39, 75, 76, 77, 78

[125] Petcu, D. e A.V. Vasilakos: *Portability in clouds: Approaches and research opportunities.* Scalable Computing, 15:251–271, 2014. 42, 74, 75, 76, 77, 78, 79

[126] Stavru, S., I. Krasteva e S. Ilieva: *Challenges for migrating to the service cloud paradigm: An agile perspective*, volume 7652 LNCS. 2013, ISBN 9783642383328. 42, 75, 77, 78, 79

[127] Alouffi, Bader, Muhammad Hasnain, Abdullah Alharbi, Wael Alosaimi, Hashem Alyami e Muhammad Ayaz: *A systematic literature review on cloud computing security: Threats and mitigation strategies.* IEEE Access, 9:57792–57807, 2021, ISSN 2169-3536. 42, 46, 51

[128] Brogi, Antonio, José Carrasco, Javier Cubo, Francesco D'Andria, Ahmad Ibrahim, Ernesto Pimentel, Jacopo Soldani *et al.*: *Seaclouds: Seamless adaptive multi-cloud management of service-based applications.* Em *CIbSE*, páginas 95–108, 2014. 42, 75, 76, 77, 78, 79

[129] Perrons, R.K. e A. Hems: *Cloud computing in the upstream oil; gas industry: A proposed way forward.* Energy Policy, 56:732–737, 2013. 42, 74, 75, 76, 77, 78

[130] Sousa, Gustavo, Walter Rudametkin e Laurence Duchien: *Automated setup of multi-cloud environments for microservices applications.* Em *2016 IEEE 9th International Conference on Cloud Computing (CLOUD)*, páginas 327–334. IEEE, 2016. 43, 44, 74, 75, 76, 77, 78, 79

[131] Gourisaria, M.K., A. Samanta, A. Saha, S.S. Patra e P.M. Khilar: *An Extensive Review on Cloud Computing*, volume 1079. 2020, ISBN 9789811510960. 45, 74, 75, 76, 77, 78

[132] Silva, M.A.A. Da, A. Abhervé e A. Sadovykh: *From the desktop to the multi-clouds: The case of modeliosaas.* páginas 462–469, 2013, ISBN 9781479930357. 46, 74, 75, 76, 77, 78, 79

[133] Lahmar, F. e H. Mezni: *Multicloud service composition: A survey of current approaches and issues.* Journal of Software: Evolution and Process, 30, 2018. 46, 47, 74, 75, 76, 77, 78, 79

[134] Opara-Martins, Justice, Reza Sahandi e Feng Tian: *Critical analysis of vendor lock-in and its impact on cloud computing migration: a business perspective.* Journal of Cloud Computing, 5:1–18, 2016. 47, 74, 75, 76, 77, 78, 79

[135] Stavru, S., I. Krasteva e S. Ilieva: *Challenges of model-driven modernization: An agile perspective.* páginas 219–230, 2013, ISBN 9789898565426. 47

[136] Quarkus: *Quarkus: Supersonic subatomic java*. https://quarkus.io, 2024. Accessed: 2024-08-16 at 20:26 PM. 49, 55

[137] Balalaie, Armin, Abbas Heydarnoori e Pooyan Jamshidi: *Microservices architecture enables devops: Migration to a cloud-native architecture*. IEEE Software, 33(3):42–52, May 2016, ISSN 1937-4194. 49, 52

[138] Sharma, Sourabh: *Mastering Microservices with Java*. Packt Publishing Ltd, 2019. 50

[139] Balalaie, Armin, Abbas Heydarnoori e Pooyan Jamshidi: *Migrating to cloud-native architectures using microservices: an experience report*. Em *Advances in Service-Oriented and Cloud Computing: Workshops of ESOCC 2015, Taormina, Italy, September 15-17, 2015, Revised Selected Papers 4*, páginas 201–215. Springer, 2016. 50

[140] Chandrasekaran, Premanand e Karthik Krishnan: *Domain-Driven Design with Java - A Practitioner's Guide: Create simple, elegant, and valuable software solutions for complex business problems*. Packt Publishing, 2022, ISBN 1800560737. 50, 52

[141] Mateus, Bruno Góis, Matias Martinez e Christophe Kolski: *Learning migration models for supporting incremental language migrations of software applications*. Information and Software Technology, 153:107082, 2023. 52

[142] GraanVM: *Graanvm: Build faster, smaller, leaner applications - an advanced jdk with ahead-of-time native image compilation*. https://www.graalvm.org/, 2024. Accessed: 2024-08-19 at 18:06 PM. 55

[143] JMeter: *Apache jmeter*. https://jmeter.apache.org, 2024. Accessed: 2024-08-19 at 18:28 PM. 55

[144] Elgedawy, I: *Sultan: A composite data consistency approach for saas multi-cloud deployment*. páginas 122–131, 2015. `https://ieeexplore-ieee-org.ez54.periodicos.capes.gov.br/document/7431403`. 74, 75, 76, 77, 79

[145] Elmroth, E., J. Tordsson, F. Hernández, A. Ali-Eldin, P. Svärd, M. Sedaghat e W. Li: *Self-management challenges for multi-cloud architectures (invited paper)*, volume 6994 LNCS. 2011, ISBN 9783642247545. 74, 75, 76, 77, 78

[146] Fowley, F. e C. Pahl: *Cloud migration architecture and pricing – Mapping a licensing business model for software vendors to a SaaS business model*, volume 707. 2018, ISBN 9783319721248. 74, 75, 76, 77, 78, 79

[147] Hajjat, M., X. Sun, Y. W.E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai e M. Tawarmalani: *Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud*. páginas 243–254, 2010, ISBN 9781450302012. 74, 75, 76, 77, 78, 79

[148] Hwang, J.: *Computing resource transformation, consolidation and decomposition in hybrid clouds*. páginas 144–152, 2015, ISBN 9783901882777. 74, 75, 76, 77, 78, 79

[149] Mohamed, Amany M e Hisham M Abdelsalam: *A multicriteria optimization model for cloud service provider selection in multicloud environments*. Software: Practice and Experience, 50(6):925–947, 2020. 74, 75, 76, 77, 78, 79

[150] Ravi, N. e M. Thangarathinam: *Emergence of middleware to mitigate the challenges of multi-cloud solutions onto mobile devices*. International Journal of Cooperative Information Systems, 28, 2019. 74, 75, 76, 77, 78, 79

[151] Raza, M., A. Imtiaz e U. Shoaib: *A review on security issues and their impact on hybrid cloud computing environment*. International Journal of Advanced Computer Science and Applications, 10:353–357, 2019. 74, 75, 76, 77, 78

[152] Repschlaeger, J., R. Zarnekow, S. Wind e K. Turowski: *Cloud requirement framework: Requirements and evaluation criteria to adopt cloud solutions*. 2012, ISBN 9788488971548. 74, 75, 76, 77, 78

[153] Rosati, P., F. Fowley, C. Pahl, D. Taibi e T. Lynn: *Making the cloud work for software producers: Linking architecture, operating cost and revenue*. Volume 2018-Janua, páginas 364–375, 2018, ISBN 9789897582950. 74, 75, 76, 77, 78

[154] Saif, M.A.N., S.K. Niranjan, B.A.H. Murshed, F.A. Ghanem e A.A.Q. Ahmed: *Cso-ilb: chicken swarm optimized inter-cloud load balancer for elastic containerized multi-cloud environment*. Journal of Supercomputing, 2022. 74, 75, 76, 77, 78

[155] Shyamasundar, R.K., N.V.N. Kumar e M. Rajarajan: *Information-flow control for building security and privacy preserving hybrid clouds*. páginas 1410–1417, 2017, ISBN 9781509042968. 74, 75, 76, 77, 78

[156] Sorheller, V.U., E.J. Hovik, E. Hustad e P. Vassilakopoulou: *Implementing cloud erp solutions: A review of sociotechnical concerns*. Volume 138, páginas 470–477, 2018. 74, 76, 77, 78

[157] Tona, A.A. e D.P. Sharma: *Dps-aa: Intranet migration strategy model for clouds*. International Journal of Modern Education and Computer Science, 12:55–63, 2020. 74, 75, 76, 77, 78

[158] Wright, Peter, Terence Harmer, John Hawkins e Yih Leong Sun: *A commodity-focused multi-cloud marketplace exemplar application*. Em *2011 IEEE 4th International Conference on Cloud Computing*, páginas 590–597, July 2011. 74, 75, 76, 77

[159] Zhou, B., F. Zhang, J. Wu e Z. Liu: *Cost reduction in hybrid clouds for enterprise computing*. páginas 270–274, 2017, ISBN 9781538632925. 74, 75, 76, 77, 78

[160] Balalaie, Armin, Abbas Heydarnoori e Pooyan Jamshidi: *Microservices architecture enables devops: Migration to a cloud-native architecture*. IEEE Software, 33(3):42–52, May 2016, ISSN 1937-4194. 74, 75, 76

[161] Sailer, A., B. Yang, S. Jain, A.E. Tomala-Reyes, M. Singh e A. Ramnath: *Healthcare application migration in compliant hybrid clouds*, volume 11236 LNCS. 2018, ISBN 9783030035952. 74, 75, 76, 77, 78

[162] Lichtenthäler, R., M. Prechtl, C. Schwille, T. Schwartz, P. Cezanne e G. Wirtz: *Requirements for a model-driven cloud-native migration of monolithic web-based applications.* Software-Intensive Cyber-Physical Systems, 35:89–100, 2020. 75, 76, 77, 78, 79

[163] Mahmood, F., A.Z. Khan e R.H. Bokhari: *Erp issues and challenges: a research synthesis.* Kybernetes, 49:629–659, 2020. 75, 77, 78, 79

[164] Monrat, Ahmed Afif, Olov Schelen e Karl Andersson: *A survey of blockchain from the perspectives of applications, challenges, and opportunities.* IEEE Access, 7:117134–117151, 2019. 75, 76, 78

[165] Naik, N.: *Performance evaluation of distributed systems in multiple clouds using docker swarm.* 2021, ISBN 9781665444392. 75, 76, 77, 79

[166] Wolfswinkel, Joost F, Elfi Furtmueller e Celeste PM Wilderom: *Using grounded theory as a method for rigorously reviewing literature.* European journal of information systems, 22(1):45–55, 2013. 75

[167] Brogi, A., J. Carrasco, J. Cubo, F. D'Andria, A. Ibrahim, E. Pimentel e J. Soldani: *SeaClouds: Seamless adaptive multi-cloud management of service-based applications.* Em *CIBSE 2014: Proceedings of the 17th Ibero-American Conference Software Engineering*, páginas 95–108, 2014, ISBN 9789562362474. 76

Apêndice A

Categories Table

Appendix Table 1
Categories, Challenges and Benefits in Citations Articles

Category	Challenge / Benefit	Citation
CLOUD MANAGEMENT	Cost	[123], [5], [33], [4], [132], [12], [17], [144], [145], [146], [8], [147], [148], [109], [41], [23], [133], [11], [149], [129], [125], [26], [150], [151], [152], [153], [154], [10], [7], [155], [156], [130], [157], [158], [159]
	DevOps	[31], [5], [160], [24], [12], [85], [87], [41], [23], [149], [134], [125], [150], [161]
	Monitoring	[5], [33], [122], [132], [12], [144], [146], [8], [133], [11], [149], [134], [125], [151], [154], [155]
	Multi-tenant	[31], [145], [146], [8], [131], [85], [87], [41], [151], [152], [153], [7]

Continued on next page...

Category	Challenge / Benefit	Citation
	Performance	[31], [6], [9], [123], [32], [5], [160], [24], [33], [122], [4], [132], [12], [144], [145], [146], [8], [131], [147], [148], [87], [109], [41], [133], [162], [163], [11], [149], [164], [165], [26], [124], [150], [151], [152], [153], [154], [161], [10], [7], [130], [126], [157], [166], [158], [159]
	Portability	[31], [5], [160], [33], [122], [12], [144], [8], [131], [87], [109], [41], [23], [133], [134], [129], [125], [150], [151], [152], [161], [7]
CLOUD-NATIVE ARCHITECTURE	Architecture refactoring	[9], [160], [132], [8], [85], [41]
	Cloud architecture	[31], [9], [123], [5], [128], [122], [4], [132], [12], [145], [146], [8], [147], [87], [109], [41], [23], [133], [162], [11], [129], [125], [150], [151], [153], [10], [155], [126], [159]
	Cloud-native architecture	[31], [9], [5], [24], [12], [146], [8], [85], [87], [109], [41], [23], [162], [125], [153], [10], [130]
	Coupling	[5], [12], [8], [85], [162], [125], [124], [154], [126]
	Flexibility	[9], [123], [32], [5], [160], [128], [122], [4], [132], [12], [144], [145], [146], [8], [131], [147], [109], [41], [133], [162], [164], [134], [129], [125], [124], [151], [152], [153], [10], [7], [155], [126]
	Legacy application architecture	[9], [128], [122], [132], [12], [8], [147], [85], [87], [109], [41], [133], [162], [125], [10], [130]

Continued on next page...

Category	Challenge / Benefit	Citation
	Scalability	[31], [9], [5], [160], [24], [167], [122], [132], [12], [144], [146], [8], [131], [147], [85], [148], [87], [109], [41], [23], [133], [162], [11], [164], [165], [134], [129], [125], [26], [124], [150], [152], [153], [154], [10], [7], [156], [130], [157], [158], [159]
MULTI-CLOUD ARCHITEC-TURE	Benefits of multi-cloud	[12], [41], [133], [165], [7]
	Cost reduction	[6], [146], [133], [149], [7], [130]
	Decision models for multi-cloud adoption	[41], [11], [7]
	Flexibility and choice	[122, 12], [133], [7]
	Hight availability	[5], [145], [11]
	Hybrid cloud	[31], [9], [123], [32], [5], [4], [12], [8], [131], [147], [148], [109], [41], [23], [133], [11], [134], [129], [125], [150], [151], [154], [161], [7], [155], [157], [159]
	Low security risks	[7]
	Multi-cloud	[31], [6], [9], [5], [128], [122], [4], [132], [12], [144], [145], [109], [41], [23], [149], [165], [150], [151], [154], [7], [130]
	Multiple clouds	[6], [123], [32], [5], [24], [128], [122], [4], [132], [12], [145], [8], [131], [147], [148], [109], [41], [133], [11], [149], [165], [134], [125], [26], [124], [150], [154], [161], [7], [155], [130], [159]

Continued on next page...

Category	Challenge / Benefit	Citation
	Private cloud	[31], [123], [32], [5], [33], [122], [4], [132], [12], [145], [131], [148], [87], [109], [41], [23], [133], [162], [163], [11], [134], [129], [125], [150], [151], [152], [7], [155], [130], [158], [159]
	Selecting appropriate providers and services	[9], [5], [4], [144], [8], [133], [11], [149], [152], [7], [130], [157], [158]
	Vendor lock-in avoidance	[12], [165], [7]
CLOUD SERVICE PROVIDERS	Cloud provider	[31], [6], [9], [123], [32], [5], [128], [122], [4], [132], [12], [145], [146], [8], [131], [147], [148], [87], [109], [41], [23], [133], [162], [163], [11], [149], [165], [134], [129], [125], [26], [124], [150], [151], [152], [153], [154], [161], [7], [155], [156], [130], [126], [157], [158], [159]
	Intracloud and federation	[122], [4], [12], [109], [41], [133], [125], [150], [158]
	Low resource settings	[5], [133]
	Multi-cloud Environment and Management	[9], [5], [122], [12], [8], [148], [109], [41], [133], [149], [150], [154], [7], [130]
	Multi-cloud management tools	[33], [133], [149], [130]
CLOUD MIGRATION PROCESS	Benefits of cloud migration	[8], [147], [109], [41], [11], [134], [151]
	Cloud migration	[31], [9], [5], [128], [122], [4], [132], [144], [146], [8], [131], [147], [85], [148], [87], [109], [41], [23], [133], [162], [11], [134], [129], [125], [26], [124], [151], [152], [153], [161], [10], [7], [155], [156], [126], [157], [159]
	Cloud migration strategies	[31], [9], [24], [8], [149], [26], [154], [10], [157]

Continued on next page...

Category	Challenge / Benefit	Citation
	Multi-cloud migration	[9], [5], [122], [12], [8], [85], [109], [41], [133], [7]
	Process	[31], [24], [122], [145], [146], [8], [131], [147], [85], [148], [109], [41], [162], [11], [125], [26], [153], [161], [10], [126], [157]
CLOUD INFRASTRUCTURE	Cloud computing architecture	[123], [41], [133], [162], [126]
	Definition of cloud computing	[31], [123], [4], [87], [23], [150], [152], [7]
	Elasticity	[31], [9], [123], [5], [24], [12], [145], [8], [131], [85], [41], [23], [162], [134], [125], [124], [151], [153], [154], [7]
	Evolution of cloud computing	[6], [123], [8], [41], [23], [163], [11], [152]
	Hybrid cloud	[31], [9], [123], [32], [5], [4], [12], [8], [131], [147], [148], [109], [41], [23], [133], [11], [134], [129], [125], [150], [151], [154], [161], [7], [155], [157], [159]
	Public cloud	[31], [6], [123], [32], [5], [128], [122], [4], [132], [12], [8], [131], [148], [87], [109], [41], [23], [133], [163], [11], [149], [134], [129], [125], [150], [151], [152], [153], [7], [155], [130], [159]
RISK and SECURITY	Data privacy	[5], [8], [131], [147], [85], [133], [11], [134], [125], [152], [155], [126]
	Security	[31], [9], [123], [5], [8], [131], [147], [85], [41], [23], [133], [162], [163], [11], [134], [125], [26], [124], [151], [152], [161], [7], [155], [156], [126]
	Security risk concerns	[31], [9], [8], [131], [133], [163], [11], [164], [134], [129], [151], [7]

Continued on next page...

Category	Challenge / Benefit	Citation
VENDOR LOCK-IN	Challenge of vendor lock-in	[5], [12], [150]
	Difficulty portability between clouds	[5], [128], [122], [132], [12], [144], [146], [8], [147], [148], [134], [125]
	Impact of vendor lock-in	[23]
	Vendor lock-in	[5], [122], [4], [132], [12], [144], [8], [133], [163], [11], [149], [165], [134], [125], [150], [10], [7], [130], [126]
	Vendor lock-in concern	[5], [12], [8], [11], [134], [150], [7]
	Vendor lock-in risk	[122], [144], [8], [11], [134], [150]
LEGACY ARCHITECTURE	Monolithic application	[5], [12], [8], [85], [87], [109], [41], [162]