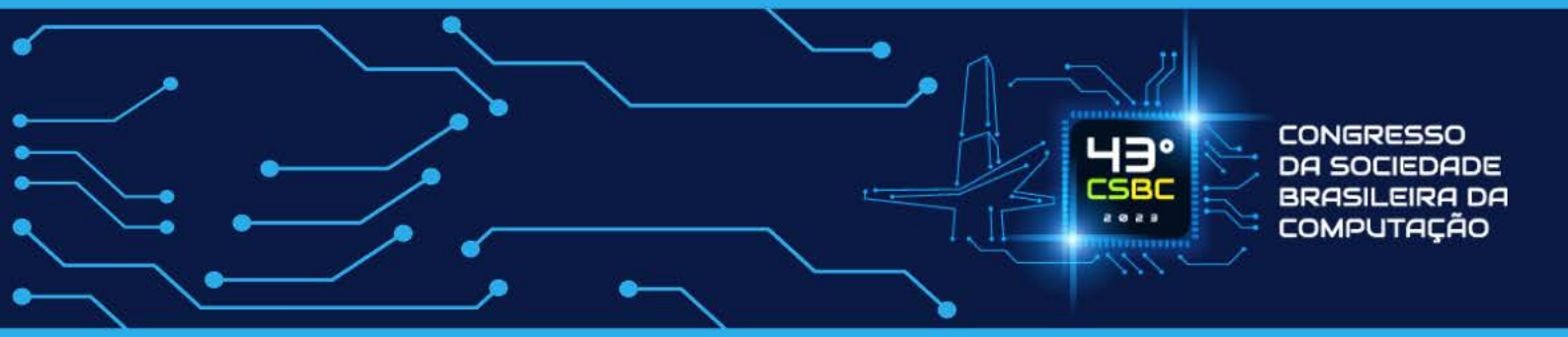
JAI 2023

42ª Jornada de Atualização em Informática

Organizadores
Alirio Santos de Sá
Weverton Luis da Costa Cordeiro



SBC
Sociedade Brasileira de Computação

Organizadores
Alírio Santos de Sá
Weverton Cordeiro

42ª Jornada de Atualização em Informática
JAI 2023

Porto Alegre
Sociedade Brasileira de Computação –SBC
2023

Dados Internacionais de Catalogação na Publicação (CIP)

C749 Congresso da Sociedade Brasileira de Computação (43.: 06 ago.-11 ago. 2023 : João Pessoa)
42ª Jornada de Atualização em Informática (JAI 2023) [recurso eletrônico] / organização: Alirio Santos de Sá; Weverton Luis da Costa Cordeiro. Dados eletrônicos. - Porto Alegre: Sociedade Brasileira de Computação, 2023.
254 p.; il.: PDF; 103 MB

Inclui bibliografia
ISBN 978-85-7669-553-0 (e-book)

1. Computação - Brasil - Congressos. 2. Informática. I. de Sá, Alirio Santos. II. Cordeiro, Weverton Luis da Costa. III. Sociedade Brasileira de Computação. IV. Título.

CDU 004(063)

Ficha catalográfica elaborada por Annie Casali - CRB-10/2339

Biblioteca Digital da SBC - SBC OpenLib

Índice para catálogo sistemático:

1. Ciência e tecnologia informáticas: Computação: Processamento de dados - Publicação de conferências, congressos, simpósios etc... 004(063)

42ª Jornada de Atualização em Informática
(JAI 2023)

Editora
Sociedade Brasileira de Computação (SBC)

Coordenação Geral do CSBC 2023
Francisco Daladier Marques Júnior (IFPB)
Paulo Ditarso Maciel Júnior (IFPB)

Coordenação da JAI 2023
Alirio Santos de Sá (UFBA)
Weverton Luis da Costa Cordeiro (UFRGS)

Comitê de Programa da JAI 2023
Aldri dos Santos (UFMG)
Aline Andrade (UFBA)
Alirio Santos de Sá (UFBA)
Cláudio Diniz (UFRGS)
Denis Rosário (UFPA)
Fabíola Guerra Nakamura (UFAM)
Flávio Assis (UFBA)
Márcio Cornélio (UFPE)
Rafael Lopes Gomes (UECE)
Rita Suzana Pitangueira Maciel (UFBA)
Rodrigo Mansilha (UNIPAMPA)
Thais Vasconcelos Batista (UFRN)
Weverton Cordeiro (UFRGS)

Prefácio

A Jornada de Atualização em Informática (JAI) é um evento que se tornou referência na apresentação de tópicos relevantes para a Pesquisa & Desenvolvimento dentro do Congresso da Sociedade Brasileira de Computação (CSBC). A JAI vem contribuindo de forma significativa com a disseminação de conhecimento de ponta para estudantes, profissionais e pesquisadores em Computação no Brasil, sempre buscando cobrir assuntos na fronteira do conhecimento da pesquisa em computação.

Em 2023, a JAI teve 14 propostas registradas e 12 submissões completas – o maior número de submissões completas nos últimos 07 anos. Considerando uma avaliação histórica das últimas edições da JAI, os coordenadores adotaram de antemão uma taxa alvo de aceitação de até 50% e solicitaram à Coordenação Geral do CSBC 2023 a reserva de 06 janelas na programação do congresso para a JAI. Para garantir imparcialidade e equidade nas avaliações das propostas submetidas pelos Coordenadores Gerais da JAI 2023, a Profa. Thais Vasconcelos Batista (UFRN) gentilmente aceitou conduzir um processo de revisão anônima, cujo resultado só ficou acessível aos organizadores após a conclusão do processo. Todas as propostas submetidas receberam ao menos três revisões.

Conforme previsto na Chamada de Propostas de Cursos da JAI, a escolha das propostas considerou não apenas uma avaliação técnica das submissões e de seus proponentes, mas também a diversidade das áreas temáticas. Assim, dentre as propostas submetidas, das seis com pontuação superior (média acima de 50), apenas cinco destas foram aceitas na primeira etapa. Duas das seis propostas, ambas muito bem avaliadas, concentravam-se em uma mesma área e possuíam grande sobreposição temática. Por isso, a de melhor pontuação dentre as duas foi selecionada para a segunda etapa. Com isso, as cinco propostas selecionadas contemplam as áreas de redes sociais, redes de computadores, engenharia de software, banco de dados e sistemas distribuídos.

Registramos aqui nossos agradecimentos pelo esforço do Comitê de Programa na realização das revisões e reservamos um agradecimento especial à Profa. Thais Vasconcelos Batista pelo apoio adicional no processo de revisão anônima. Por fim, mas não menos importante, agradecemos aos autores de propostas de cursos que prestigiaram o evento com suas submissões.

Em sua 42ª edição (JAI 2023), são tratados os temas de desinformação em plataformas digitais, confiabilidade e segurança em sistemas computacionais, sustentabilidade do software pesquisa, banco de dados em memória e softwarização de redes celulares. O Capítulo 1 (Desinformação em Plataformas Digitais: Conceitos, Abordagens Tecnológicas e Desafios) discute o cenário atual de estudos no contexto de desinformação em plataformas digitais, oferecendo uma introdução ao pesquisador que pretende explorar este tema. Para isso, inicialmente, os autores 1) apresentam e discutem os conceitos que fundamentam a área, 2) relacionam repositórios de dados que podem ser úteis para o estudo deste fenômeno, 3) sumarizam as principais estratégias exploradas para entendimento, bem como abordagens tecnológicas para detecção e monitoramento de desinformação em plataformas digitais e 4) apresentam uma visão crítica geral da área, destacando desafios e oportunidades de pesquisa neste contexto.

O Capítulo 2 (Confiabilidade e Segurança nos Sistemas Distribuídos Físico-Digitais) concentra-se no estudo de sistemas ciberfísicos distribuídos - caracterizados por componentes físicos geograficamente dispersos e interconectados por meio de redes de comunicação, criando assim uma infraestrutura descentralizada e colaborativa. Os autores cobrem os desafios na relação físico-digital nesses sistemas, entre os quais a interoperabilidade, a confiabilidade, a cibersegurança e a escalabilidade, e discutem como o gerenciamento autonômico tem sido utilizado nesses sistemas. Os autores apresentam uma abordagem teórico-prática para a construção de sistemas distribuídos autônomos ou autonômicos capazes de atender aos requisitos de desempenho, confiabilidade e segurança dos sistemas físico-digitais distribuídos.

O Capítulo 3 (Princípios e Práticas para Sustentabilidade do Software de Pesquisa) foca no Software de Pesquisa, isto é, software desenvolvido no contexto de uma pesquisa científica, destacando a preocupação na comunidade científica com sua sustentabilidade e influência na capacidade de reprodução de estudos científicos por pesquisadores independentes. O curso aborda a sustentabilidade do Software de Pesquisa sob uma perspectiva técnica, no qual os autores apresentam e discutem boas práticas de desenvolvimento de software que podem ser úteis para apoiar pesquisadores de diferentes áreas no desenvolvimento de software de pesquisa sustentável. Ao longo do curso, os autores apresentam motivações e exemplos de uso das práticas, seus benefícios e desafios relacionados por meio dos resultados de um estudo conduzido com um grupo de pesquisa da área de Física.

O Capítulo 4 (Bancos de Dados em Memória e suas Estratégias de Recuperação Após Falhas) discutem os potenciais e desafios relacionados ao uso de sistemas de bancos de dados em memória, os quais se apresentam como uma alternativa para sistemas que precisam de processamento massivo de dados em tempo real. Os autores apresentam uma visão geral da arquitetura e implementação de bancos de dados em memória e suas principais estratégias de recuperação após falhas. Para atingir essa meta, o minicurso 1) fornece uma visão geral da tecnologia de bancos de dados em memória, 2) revisa os conceitos de recuperação após falhas, 3) apresenta as principais escolhas arquiteturais para implementação de bancos de dados em memória, e 4) descreve as estratégias de recuperação implementadas por uma amostra representativa dos bancos de dados em memória modernos.

O Capítulo 5 (Desagregando e Softwarizando as Redes de Celulares e o Programa OpenRAN Brasil) tem como principal objetivo apresentar os conceitos e desafios que estão levando a academia e indústria a investir no conceito de RAN (Redes de Acesso via Rádio) aberta (OpenRAN), um framework com padrões, protocolos e componentes de softwares de código aberto que tem como objetivo democratizar segmentos da rede de telecomunicações para não depender de equipamentos de grandes indústrias e permitir a redução de custos. Os autores apresentam os fatores históricos da evolução das RANs e os conceitos de abertura e softwarização, abordando ainda a desagregação, controle inteligente da RAN (RIC), Núcleo da Rede (Core Network), virtualização, interfaces abertas e desafios. Os autores apresentam ainda as iniciativas ao redor do mundo que estão colaborando no avanço, padronização e desenvolvimento das OpenRANs, para então discutir o Programa OpenRAN Brasil, ressaltando a motivação, objetivos, resultados esperados, testbed e aplicações. Por fim, os autores apresentam as considerações finais e tendências futuras em relação à pesquisa e desenvolvimento das OpenRANs e seus componentes.

Encerramos este Prefácio agradecendo aos Coordenadores Gerais do CSBC 2023, Prof. Francisco Daladier Marques Júnior e Prof. Paulo Ditarso Maciel Júnior, pela confiança depositada nos Coordenadores Gerais da JAI 2023 e pelo suporte ao longo do processo de organização do evento.

Alirio Santos de Sá (UFBA) – aliriosa@ufba.br

Weverton Luis da Costa Cordeiro (UFRGS) - weverton.cordeiro@inf.ufrgs.br

Coordendores da JAI 2023

Sumário

Capítulo 1

Desinformação em Plataformas Digitais: Conceitos, Abordagens Tecnológicas e Desafios

Julio C. S. Reis, Philipe Melo, Márcio Silva, Fabrício Benevenuto

Abstract

Digital platforms, including online social networks and instant messaging applications, have become spaces widely abused by misinformation campaigns, directly impacting various spheres of our society such as politics, health, and others. Consequently, this has stimulated the emergence of research on this topic in several areas of knowledge, including Computer Science. Thus, the general goal of this chapter is to discuss the current scenario of studies in the context of misinformation on digital platforms, offering an introduction to the researcher to explore this theme. Thus, we first present and discuss the concepts underlying the area. Then, we list data repositories that may be useful for studying the phenomenon. Afterward, we describe the main strategies explored for understanding as well as technological approaches for detecting and monitoring disinformation on digital platforms. Last, we present a critical overview of the area, highlighting research challenges and opportunities in this context.

Resumo

Plataformas digitais, que incluem redes sociais online e aplicativos de mensagem instantânea, se tornaram espaços amplamente abusados por campanhas de desinformação com impactos diretos em diversas esferas da nossa sociedade, incluindo política, saúde, dentre outras. Consequentemente, isso tem estimulado o surgimento de pesquisas neste tema em diversas áreas do conhecimento, incluindo Ciência da Computação. Assim, o objetivo geral deste capítulo é discutir o cenário atual de estudos no contexto de desinformação em plataformas digitais, oferecendo uma introdução ao pesquisador que pretende explorar este tema. Para isso, inicialmente, são apresentados e discutidos os conceitos que fundamentam a área. Em seguida, são relacionados repositórios de dados que podem ser úteis para o estudo deste fenômeno. Depois, são sumarizadas as principais estratégias exploradas para entendimento, bem como abordagens tecnológicas para detecção e monitoramento de desinformação em plataformas digitais. Por fim, é apresentada uma visão crítica geral da área, destacando desafios e oportunidades de pesquisa neste contexto.

1.1. Introdução

As plataformas digitais, que incluem redes sociais online como Facebook, Instagram e Twitter, e aplicativos para troca de mensagens instantâneas como WhatsApp e Telegram, são usadas ativamente por mais da metade da população mundial [Gallagher, 2017]. Essas plataformas mudaram significativamente a forma como as pessoas interagem e se comunicam no ambiente online modificando vários dos ecossistemas de informação existentes. Particularmente, as plataformas digitais mudaram drasticamente a maneira como as notícias são produzidas, disseminadas e consumidas em nossa sociedade, abrindo oportunidades imprevistas e também criando desafios complexos.

Parte das razões para esta mudança é inerente à natureza das próprias plataformas digitais: *(i)* muitas vezes é mais oportuno e menos dispendioso produzir e consumir notícias nesses ambientes em comparação com meios noticiosos tradicionais, como jornais online e/ou físicos, ou ainda rádio e televisão; e *(ii)* é mais fácil compartilhar, comentar e discutir as notícias com amigos ou outros leitores em plataformas digitais, o que melhora e/ou impulsiona a comunicação e as interações entre os usuários nesses sistemas [Shu et al., 2017]. Assim, as plataformas digitais estão moldando a maneira como as pessoas consomem informações, especialmente notícias. Um estudo revelou que cerca de 62% dos usuários americanos e 66% dos usuários brasileiros recebem e consomem notícias a partir desses sistemas [Mitchell, 2016; Report, 2018]. Apesar dos inúmeros benefícios que essas plataformas trazem para nossa sociedade, elas se tornaram um local propício para realização de campanhas de desinformação que muitas vezes visam enganar as pessoas, especialmente em contextos como saúde e política.

Em relação a saúde, notícias "médicas" contendo desinformação divulgadas em plataformas digitais causaram danos irreparáveis [Dai et al., 2020]. Por exemplo, na China, um paciente com câncer confundiu um anúncio online com um tratamento experimental contra a doença, acreditando ser uma informação clinicamente confiável, o que infelizmente resultou em sua morte [Dai et al., 2020][1]. Além disso, durante a pandemia de COVID-19, houve um aumento significativo dos rumores e conspirações espalhados pelas plataformas digitais [Ferrara, 2020]. A *International Fact-Checking Network* (IFNC)[2] encontrou mais de 3.500 alegações falsas relacionadas a COVID-19 em menos de dois meses [Poynter, 2020]. Como resultado, pelo menos 800 pessoas podem ter morrido em todo o mundo por causa de desinformação relacionada ao coronavírus nos primeiros três meses de 2020[3].

Já no contexto político, eleição após eleição, podemos observar diferentes formas de desvio de conduta e estratégias complexas de manipulação de opinião por meio da disseminação de desinformação em plataformas digitais. A eleição presidencial de 2016 nos EUA ainda é lembrada pela "guerra de desinformação" que aconteceu principalmente por meio do Twitter e do Facebook. Um caso notório envolveu uma tentativa de influência da Rússia por meio de publicidade segmentada [Ribeiro et al., 2019]. Tentativas semelhantes foram observadas durante as eleições brasileiras de 2018, onde o WhatsApp foi extensivamente abusado para envio de campanhas de desinformação, com grande uso de

[1] https://www.bbc.com/news/business-36189252
[2] https://www.poynter.org/ifcn/
[3] https://www.bbc.com/news/world-53755067

imagens e memes manipulados[4] contendo todo tipo de ataque político. Por exemplo, um estudo mostrou que 88% das imagens mais populares compartilhadas no último mês antes das eleições brasileiras de 2018 eram falsas ou enganosas [Tardaguila et al., 2018]. Também por meio do WhatsApp, na Índia, boatos falsos espalhados pela plataforma foram responsáveis por vários casos de linchamento e agitação social [Arun, 2019].

Uma característica única das notícias em plataformas digitais que suportam esse fenômeno da disseminação da desinformação nesses ambientes é que qualquer pessoa pode se registrar e/ou comportar-se como um editor de notícias sem nenhum custo inicial (e.g., qualquer pessoa pode criar uma página no Facebook alegando ser um jornal ou organização de mídia de notícias, ou ainda, criar um grupo no WhatsApp ou Telegram para divulgação deste tipo de conteúdo). Consequentemente, não apenas as empresas de notícias tradicionais (e.g., jornais) estão migrando para plataformas digitais, mas também muitos veículos de notícias também estão emergindo exclusivamente nesses ambientes[5]. Por exemplo, esforços anteriores mostraram que em 2018 havia mais de 20 mil páginas nos EUA categorizadas como editores de notícias no Facebook [Ribeiro et al., 2018], e este número certamente continua crescendo.

Junto à esta transição, há uma preocupação crescente sobre como os produtores de notícias contendo desinformação elaboram e publicam este tipo de conteúdo[6], muitas vezes divulgando-as amplamente através de plataformas digitais [Lazer et al., 2018]. Por exemplo, um estudo financiado pela Avaaz[7] perguntou aos eleitores brasileiros se eles viram e acreditaram em cinco das notícias mais populares contendo desinformação e disseminadas em plataformas digitais durante as últimas semanas da eleição presidencial em 2018. De forma impressionante, os resultados revelaram que mais de 98% dos eleitores entrevistados foram expostos a uma ou mais dessas notícias contendo desinformação e que quase 90% deles acreditaram que essas histórias eram verdadeiras[8]. Potencialmente, esses números impactaram a democracia no Brasil diante das eleições presidenciais de 2018.

Desinformação, manipulação de opinião, mentiras, boatos, rumores, e enganos sempre existiram, mas a ascensão das plataformas digitais aumentou significativamente o potencial da disseminação deste tipo de conteúdo transformando este problema em um fenômeno mundial, que tem atraído a atenção de pesquisadores de diversas áreas, incluindo Ciência da Computação. Isso impulsiona a necessidade de discussões sobre o impacto das plataformas digitais frente a este fenômeno cada vez mais desafiador.

Assim, acreditamos que esse é essencial fornecer às pessoas insumos para: *(i)* suporte ao entendimento do fenômeno da desinformação considerando diferentes contextos, cenários, e ambientes; *(ii)* proposição de abordagens automatizadas que possam ser

[4]"Uma imagem, vídeo, texto, etc., tipicamente de natureza humorística, que é copiado e difundido rapidamente pelos internautas, muitas vezes com pequenas variações" [Oxford, 2020].

[5]https://www.comscore.com/Insights/Blog/Traditional-News-Publishers-Take-Non-Traditional-Path-to-Digital-Growth

[6]https://www1.folha.uol.com.br/ilustrissima/2017/02/1859808-como-funciona-a-engrenagem-das-noticias-falsas-no-brasil.shtml

[7]https://www.avaaz.org/

[8]https://www1.folha.uol.com.br/poder/2018/11/90-dos-eleitores-de-bolsonaro-acreditaram-em-fake-news-diz-estudo.shtml

efetivas na contenção e/ou detecção deste tipo de conteúdo, e por fim; *(iii)* embasamento ao processo de tomada de decisão por parte dos indivíduos até entidades e órgãos de gestão das plataformas digitais e governamentais. De forma geral, esperamos contribuir com esforços focados na minimização dos impactos ocasionados pelo problema em nossa sociedade.

Neste sentido, este capítulo visa não só uma exposição teórica dos conceitos e definições relacionadas à temática, mas também apresentar práticas introdutórias, as quais podem ser consideradas passos fundamentais no estudo e desenvolvimento de soluções no contexto da desinformação em plataformas digitais, contribuindo com a formação de recursos humanos, capacitação e compartilhamento de conhecimentos a respeito dos impactos relacionados ao fenômeno. Além disso, esperamos fornecer aos leitores uma compreensão abrangente dos conceitos relacionados à disseminação da desinformação em plataformas digitais, como redes sociais e aplicativos de mensagens.

Este capítulo está organizado da seguinte forma. Primeiramente, apresentamos brevemente o estado da arte nesta área, incluindo definições e uma descrição do ecossistema de (des)informação nessas plataformas, bem como áreas correlatas. Em seguida, relacionamos os principais repositórios de dados e estratégias atualmente explorados para entendimento e caracterização do problema, bem como abordagens tecnológicas para mitigar seus potenciais impactos na sociedade. Além disso, este capítulo também inclui um relato de experiência de um projeto real, o ELEIÇÕES SEM FAKE, desenvolvido em parceria com o Tribunal Superior Eleitoral (TSE) do Brasil e amplamente utilizado para enfrentamento à desinformação durante o processo eleitoral no país. Por fim, apresentamos considerações finais, incluindo desafios e oportunidades de pesquisa na área.

1.2. Ecossistema de Notícias em Plataformas Digitais

A mídia noticiosa tem sido objeto de estudo de diversas áreas do conhecimento, como Jornalismo, Comunicação e Ciências Políticas. No entanto, desde que ela se concentrou na Web e nas plataformas digitais, este tem sido também um tópico de interesse dos cientistas da computação. Com o surgimento da era digital, os meios de comunicação começaram a publicar no ambiente online. Assim, com o farto rastro digital de informações jornalísticas, as possibilidades de novas aplicações e o surgimento de novos desafios nesse cenário complexo, cientistas da computação têm investigado problemas relacionados ao ecossistema de notícias em plataformas digitais, mas geralmente com objetivos e finalidades diferentes.

Em suma, a base do ecossistema de notícias nas plataformas digitais pode ser dividida em três componentes principais: *(i) produção*, *(ii) consumo* e *(iii) disseminação e interação*, conforme apresentado na Figura 1.1. Primeiro, antes das plataformas digitais, os artigos de notícias eram produzidos (ou escritos) apenas por organizações tradicionais de mídia (i.e., jornais) ou por jornalistas independentes. Com o surgimento das plataformas digitais, uma das principais características da *(i) produção* de notícias nesses ambientes é que qualquer um pode ser um produtor de notícias. Por exemplo, qualquer um pode criar um usuário em uma plataforma digital para produzir e divulgar notícias sem nenhum custo inicial. Ademais, o *(ii) consumo* de notícias também mudou ao longo do tempo de papel de jornal (físico) para rádio/televisão e, depois, para notícias online

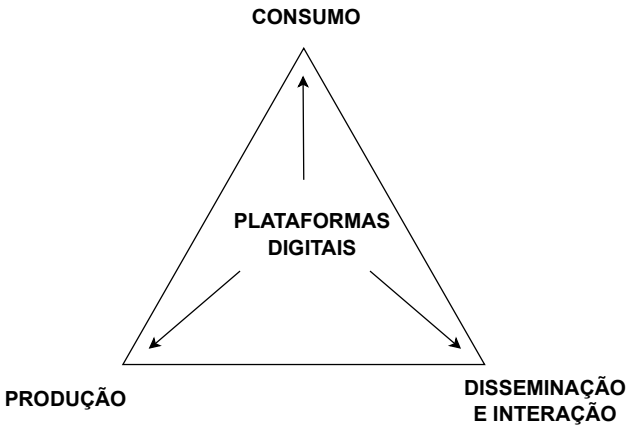


Figura 1.1. Ecossistema da informação em plataformas digitais.

e plataformas digitais, onde muitas vezes é mais oportuno e barato consumir notícias em comparação com a mídia tradicional. Por exemplo, uma pesquisa do *Pew Research Center* estima que 62% dos adultos nos EUA consomem notícias principalmente de sites de mídia social [Mitchell, 2016]. No Brasil, segundo pesquisa realizada pelo *Reuters Institute*, esse percentual chega a 66% [Report, 2018]. Por fim, as plataformas digitais introduzem novos mecanismos para *(iii) disseminação e interação*, permitindo que os usuários compartilhem e promovam notícias de acordo com sua vontade, o que pode ser comitantemente benéfico e perigoso.

Logo, em decorrência da inserção das plataformas digitais no ecossistema de notícias, existem diferentes esforços no contexto da Ciência da Computação que buscam entender melhor essas mudanças e propor soluções para dar suporte a este fenômeno em suas diversas fases. Esses esforços podem ser agrupados em três conjuntos relacionados aos principais componentes do ecossistema de notícias em plataformas digitais. O primeiro aborda a *(i) produção* de conteúdo e envolve temas como cobertura e eventos de notícias [Kwak and An, 2014; Quezada et al., 2015], credibilidade [Jin et al., 2014], aspectos da atratividade [Kim et al., 2016; Reis et al., 2015], e viés das notícias [Covert and Wasburn, 2007; Budak et al., 2016]. O segundo conjunto está relacionado ao *(ii) consumo* e envolve estudos relacionados ao entendimento de padrões de leitura das pessoas [Kourogi et al., 2015; Constantinides et al., 2015], personalização de conteúdo [Das et al., 2007; Garcin et al., 2014], sumarização e/ou resumo de notícias [Gao et al., 2012], sistemas de recomendação [Nallapati et al., 2004], visualização de conteúdo [Sheidin et al., 2017] e consumo de notícias no celular e dispositivos [Westlund, 2013; Constantinides, 2015]. Por fim, o terceiro está associado aos mecanismos de *(iii) disseminação e interação* proporcionados pelas plataformas digitais, incluindo esforços dedicadas à compreensão desses mecanismos [Chakraborty et al., 2016], motivações para os usuários compartilharem [Lee and Ma, 2012] e novos desafios que emergem desses mecanismos como bolhas de filtro e efeito de câmaras eco [Bakshy et al., 2015].

1.3. Visão Geral de Desinformação

Conforme supracitado, a mídia noticiosa adentrou as plataformas digitais e tem sido um tópico de interesse de vários pesquisadores, incluindo os cientistas da computação. No entanto, as mudanças no ecossistema da mídia jornalística ainda acontecem rapidamente, e

(a) Versão 1

(b) Versão 2

Figura 1.2. Exemplo de desinformação abordando um tratamento alternativo, milagroso e fácil para a cura do câncer. Fonte: `https://observatoriofakenews.eci.ufmg.br/2018/08/01/limonada-quente-destroi-celulas-cancerigenas/`.

algumas delas favorecem campanhas de desinformação, revelando as plataformas digitais como ambientes potenciais e/ou propícios para a disseminação deste tipo de conteúdo.

A Figura 1.2 mostra um exemplo de diferentes versões de uma mesma notícia falsa bastante popular (i.e., contendo desinformação) e propagada em plataformas digitais como Facebook, Twitter e WhatsApp sobre o potencial do suco de limão quente para a cura do câncer. A alegação foi verificada como "falsa" por várias agências de verificação de fatos em todo o mundo, incluindo Snopes[9], e Boatos.org[10] no Brasil. Especificamente sobre essa notícia contendo desinformação, a agência de verificação de fatos Snopes concluiu que "o melhor que pode ser dito é que as frutas cítricas podem potencialmente abrigar propriedades anticancerígenas que podem ajudar a prevenir o câncer. Nenhum estudo científico ou médico respeitável relatou que os limões foram definitivamente considerados um 'remédio comprovado contra cânceres de todos os tipos', nem nenhum dos (convenientemente sem nome) 'maiores fabricantes de medicamentos do mundo' relatou a descoberta de que os limões são '10.000 vezes mais forte que a quimioterapia' e que sua

[9] `https://www.snopes.com/fact-check/lemon-cancer-cure/`
[10] `https://www.boatos.org/saude/limonada-quente-mata-cancer.html`

ingestão pode 'destruir células malignas [câncer]'. Todas essas alegações são hipérboles e exageros não suportados por fatos"[11].

Esforços anteriores sugerem que existem pelo menos três tipos de notícias falsas ou desinformação [Rubin et al., 2015]. O primeiro tipo consiste em *(i)* sátira ou paródia, onde *websites* como o Onion[12] ou Daily Mash[13] publicam notícias muitas vezes contendo desinformação como tentativas humorísticas de satirizar a mídia. O segundo tipo *(ii)* contempla notícias que apresentam desinformação (i.e., informações falsas) em conjunto com informações (parcialmente) verdadeiras, mas usadas em contexto errado, incluindo boatos e notícias enganosas que não são baseadas em fatos, mas sustentam uma narrativa contínua. Por último, o terceiro grupo envolve *(iii)* notícias criadas intencionalmente com desinformação. Normalmente, elas são fabricadas e divulgadas deliberadamente em plataformas digitais para obtenção de lucros financeiros a partir, por exemplo, do número de cliques, ou ainda, para enganar, causar confusão e/ou manipular a opinião pública[14][15]. Neste capítulo, nos aprofundamos no grupo *(iii)*.

1.3.1. Definição de Desinformação

Desinformação é um tema que ainda carece de uma definição clara ou universalmente aceita. De acordo com o dicionário Collins English[16] o termo "notícias falsas" (do inglês "*fake news*") por ser definido, por exemplo, como "*informações falsas, muitas vezes sensacionalistas, disseminadas sob o disfarce de notícias*". No entanto, a definição destes termos (i.e., "desinformação", "notícias falsas", "*fake news*", etc), bem como a sua percepção, conceptualização e relação, tem sido objeto de debate recente [Shu et al., 2017]. Logo, é crucial estabelecermos a definição para o termo que será utilizada ao longo deste capítulo.

Definição 1.3.1 ***(Desinformação)*** *"Uma notícia ou mensagem publicada e propagada pela mídia, contendo informações falsas, independentemente dos meios e motivos que a embasa" [Sharma et al., 2019].*

1.3.2. Desinformação em Plataformas Digitais

De forma geral, conforme mencionado anteriormente, o ecossistema de notícias, que abrange as contendo desinformação, foi mudando ao longo do tempo do jornal impresso para o rádio/televisão e, depois, para notícias online e mais recentemente, plataformas digitais. No entanto, existem vários fundamentos sociais e teorias psicológicas e cognitivas que descrevem o impacto da desinformação tanto no nível individual quanto no ecossistema de informações sociais. Primeiro, os leitores preferem receber informações que confirmem suas opiniões existentes [Nickerson, 1998]. Em segundo lugar, os usuários fazem escolhas com base nos ganhos e perdas relativos em comparação com seu estado atual [Tversky and Kahneman, 1992]. Finalmente, os leitores tendem a acreditar que suas

[11] Tradução livre dos autores.

[12] `www.theonion.com`

[13] `www.thedailymash.co.uk`

[14] `http://www.cits.ucsb.edu/fake-news/danger-election`

[15] `https://www1.folha.uol.com.br/ilustrissima/2017/02/1859808-como-funciona-a-engrenagem-das-noticias-falsas-no-brasil.shtml`

[16] `https://www.collinsdictionary.com/woty`

percepções da realidade são as únicas visões precisas, enquanto outros que discordam são considerados desinformados, irracionais ou tendenciosos/enviesados [Ward et al., 1997]. Todos esses fatores potencializam a disseminação de desinformação pelos usuários no contexto das plataformas digitais. Além disso, existem outras características relacionadas às estes sistemas que contribuem para a disseminação da desinformação nesses ambientes, as quais serão discutidas a seguir.

Contas Maliciosas. Os usuários em plataformas digitais podem ser legítimos ou não. O baixo custo de criação de contas em plataformas digitais encorajou contas de usuários maliciosos [Shu et al., 2017], como *bots* sociais e *trolls*, que são controlados por um algoritmo de computador para interagir automaticamente com humanos (ou outros *bots*) nestes sistemas [Ferrara et al., 2016]. Nesse contexto, muitos *bots* são criados especificamente com a finalidade de causar danos, incluindo a manipulação e o espalhamento de desinformação em plataformas digitais. Existem esforços anteriores que discutem como os *bots* sociais impactaram, por exemplo, a discussão online relacionada às eleições presidenciais americanas de 2016 [Bessi and Ferrara, 2016; Badawy et al., 2019], ou ainda, como eles coordenaram campanhas de desinformação durante as eleições presidenciais francesas de 2017 [Ferrara, 2017]. Outros estudos mostram que os *bots* foram responsáveis por aumentar significativamente a disseminação de desinformação nas plataformas digitais, sugerindo que coibir os *bots* sociais pode ser uma estratégia eficaz para conter o problema neste ambiente [Shao et al., 2018; Wang et al., 2018a].

Publicidade em Mídias Digitais. Nos últimos anos, as plataformas de publicidade em mídias digitais evoluíram significativamente [Silva et al., 2020]. Com acesso a informações pessoais e atividades de milhões de pessoas ao redor do mundo, esses ambientes permitem que os anunciantes atinjam nichos muito específicos de usuários considerando informações pessoais como nome, endereço de e-mail, aspectos demográficos, comportamentos e muitos outros. No entanto, a publicidade direcionada nesses sistemas também pode ser utilizada de forma abusiva por anunciantes mal-intencionados para alcançar com eficiência pessoas suscetíveis e/ou vulneráveis a histórias falsas, alimentar queixas e incitar conflitos sociais [Ribeiro et al., 2019]. Esforços anteriores destacaram várias formas de abuso de publicidade direcionada em plataformas como Facebook, por expor informações privadas dos usuários de forma inadequada à anunciantes e por permitir publicidade discriminatória (e.g., excluir pessoas de uma determinada raça ou gênero de receber seus anúncios [Andreou et al., 2018; Venkatadri et al., 2018]). Além disso, outros estudos investigaram até que ponto os anúncios políticos da Agência de Pesquisa de Inteligência Russa (IRA) veiculados antes das eleições de 2016 nos EUA exploraram a infraestrutura de publicidade direcionada do Facebook para direcionar anúncios com eficiência em tópicos divisivos ou polarizadores (e.g., imigração, raça, etc) em subpopulações vulneráveis [Ribeiro et al., 2019]. De forma geral, os resultados sugerem que plataformas de anúncio e/ou propagandas estão sendo exploradas para uma nova forma de ataque, que é o uso de publicidade direcionada para criar discórdia social e atingir pessoas suscetíveis a informações específicas, incluindo histórias contendo desinformação.

Efeito Câmara de Eco. Finalmente, como mencionado anteriormente, as plataformas digitais deram origem a novos fenômenos disruptivos no ecossistema de notícias: as chama-

das câmaras de eco[17]. As câmaras de eco referem-se a grupos dentro de uma plataforma digital onde as pessoas (ou leitores) raramente são expostos a conteúdos que atravessam linhas ideológicas, mas são alimentados com informações que reforçam suas visões políticas ou sociais atuais pré-existentes. Embora em alguns casos a classificação algorítmica (que decide o que é apresentado no *feed* – *timeline* – ou nos resultados de pesquisa de alguém e em qual ordem) possa contribuir para esse efeito, pesquisas baseadas em dados do Facebook mostraram que as escolhas dos indivíduos são o principal fator para limitar a exposição à conteúdo transversal [Bakshy et al., 2015]. Pesquisas anteriores têm mostrado que o efeito câmara de eco facilita o acesso pelo qual as pessoas consomem e acreditam em notícias contendo desinformação devido, inclusive, a fatores como: *(i)* os relacionamentos dos usuários influenciam na confiabilidade deles em determinada fonte da informação, ou seja, os usuários tendem a perceber uma fonte como confiável se outros também perceberem-na como confiável [Shu et al., 2017]; *(ii)* os leitores podem naturalmente favorecer informações que ouvem com frequência, mesmo que sejam notícias contendo desinformação [Zajonc, 1968], e; *(iii)* nas câmaras de eco, os usuários continuam compartilhando e consumindo as mesmas informações [Shu et al., 2017]. Como resultado, esse efeito de câmara de eco cria comunidades segmentadas, homogêneas e ideologicamente polarizadas com um ecossistema de informações muito limitado, o que consequentemente favorece campanhas de desinformação [Sasahara et al., 2020].

1.3.2.1. Áreas Correlatas

Por fim, existem algumas outras áreas relacionadas ao tema de desinformação, com algumas especificidades e/ou eventuais sobreposições. Como exemplos, podemos destacar: a *(i) identificação de rumores*, que permeia história ou declaração de circulação geral, sem confirmação ou certeza dos fatos [Allport and Postman, 1947]; a *(ii) descoberta de verdade*, que visa determinar a credibilidade da fonte e a veracidade do objeto ao mesmo tempo [Li et al., 2016]; *(iii) Clickbait* que se refere a "conteúdo cujo objetivo principal é atrair a atenção e encorajar os visitantes a clicar em um link para uma determinada página da Web"[18]; ou ainda a chamada era da *(iv) pós-verdade*, em que fatos objetivos têm menos relevância na formação da opinião pública do que os apelos individuais[19], dentre outros. Logo, estas áreas e outros aspectos correlatos podem ser explorados comutantemente abrangendo, inclusive, medidas que podem ser tomadas para combater os impactos ocasionados pela disseminação da desinformação no contexto das plataformas digitais.

1.4. Repositório de Dados

Para que sejamos capazes de realizar contribuições concretas para a compreensão e detecção de desinformação em plataformas digitais, precisamos de amplos repositórios de dados contendo instâncias rotuladas por especialistas, ou seja, fatos e conteúdo verificado, abrangendo diferentes tópicos e contextos [Hui et al., 2018; Nørregaard et al., 2019].

Diante disso, realizamos um breve levantamento sobre conjuntos de dados públicos existentes comumente utilizados por trabalhos que investigam o fenômeno da desin-

[17] https://cs181journalism2015.weebly.com/the-echo-chamber-effect.html
[18] http://www.oxforddictionaries.com/definition/english/clickbait
[19] https://languages.oup.com/word-of-the-year/2016/

formação em plataformas digitais, seja para entendê-lo ou para propor soluções que visem minimizar os efeitos por ele causados. Esses conjuntos de dados rotulam o conteúdo geralmente como histórias contendo desinformação (*fake news*) ou verdadeiras. Esse conteúdo verificado pode aparecer em vários formatos diferentes, como artigos de notícias, declarações ou citações de celebridades, rumores, relatórios ou imagens e para diferentes cenários, como eleições, saúde, guerras, política. A Tabela 1.1 resume alguns dos principais conjuntos de dados públicos e suas características, incluindo uma descrição, o número total de instâncias, bem como sua distribuição por rótulo (ou seja, veredito fornecido por uma agência de checagem de fatos – verdadeiro; falso; etc.[20]) e informações sobre avaliadores (ou seja, rotuladores e/ou verificadores de fatos). Observe que colorimos em vermelho o número de instâncias verificadas e rotuladas como "desinformação" (ou falsa), em azul, as informações (ou notícias) verdadeiras, e em preto as restantes (por exemplo aquelas que são neutras).

Em alto nível, esses conjuntos de dados de conteúdo checado foram rotulados de acordo com diferentes escalas, como falso ou verdadeiro por jornalistas especializados, sites de verificação de fatos e detectores da indústria[30] fornecendo diferentes informações e contextos que nos permitem extrair características distintas [Shu et al., 2017]. Também podemos observar que, a maioria dos conjuntos de dados relacionados se concentra em notícias políticas dos EUA, notícias de entretenimento ou artigos de sátira com informações extras de sistemas tradicionais como Twitter e Facebook. Logo, ainda há uma carência na literatura de conjuntos de dados que cubram diferentes cenários de interesse, como por exemplo, eleições brasileiras, e abranjam diferentes plataformas, como aplicativos de mensagem instantânea. Isso tem motivado o nosso grupo de pesquisa no desenvolvimento de coletores de dados do WhatsApp e Telegram, conforme apresentado a seguir.

1.4.1. Arquitetura de Coleta de Dados para Aplicativos de Mensagens Instantâneas

Aplicativos de mensagens instantâneas, como o WhatsApp e o Telegram, possuem uma estrutura muito própria, bem diferente de outras plataformas. Enquanto sistemas mais tradicionais como Twitter e Facebook possuem uma esfera pública de conteúdo, onde qualquer um pode acessar e visualizar o que um grande número de usuários está fazendo na rede, suas publicações ou o perfil dos usuários, no WhatsApp e Telegram, as interações e conversas ocorrem um ambiente muito mais restrito. Neste cenário, as mensagens são descentralizadas em distintos grupos e chats de conversa e dificilmente uma pessoa tem acesso a uma parcela significativa do que os outros usuários fazem na rede.

Por isso, a arquitetura da coleta para essas plataformas possui características bem próprias que vão além de APIs, sendo necessário desde um celular com uma conta ativa na rede, até uma estratégia de localização de grupos públicos de conversa para participar, e métodos para extração, processamento e armazenamento dos dados da plataforma. Tudo isso é necessário para identificar e salvar as mensagens que circulam dentro destes ambientes. Portanto, antes de descrevermos detalhes relativos à implementação e instalação de ferramentas para auxiliar o processo de extração e processamento de dados

[20]Neste contexto é importante mencionar que é bastante vasta a nomenclatura utilizada por cada uma das agências de checagem de fatos para indicação do veredito. Logo, optamos por manter a nomenclatura original – sem tradução.

[30]BS Detector: `http://bsdetector.tech/`.

Tabela 1.1. Visão geral dos repositórios de dados rotulados disponíveis na literatura.

Repositório de Dados	Descrição	Rótulos	Rotuladores	# Instâncias
BuzzFace[21] [Potthast et al., 2018; Santia and Williams, 2018]	Notícias publicadas no Facebook por 9 agências ao longo de uma semana perto da eleição de 2016 nos EUA.	mostly false (104), mixture of true and false (245), mostly true (1669), no factual (264)	Jornalistas especialistas do BuzzFeed.	2.282
Central de Fatos [Couto et al., 2021; Marques et al., 2022]	Checagens de fatos sobre assuntos diversos de 6 agências brasileiras.	falso/fake, verdadeiro, enganoso, fato, etc	Agências de checagem de fatos.	11.647
CoAID[22] [Cui and Lee, 2020]	Notícias e reivindicações relacionadas ao COVID-19 em *websites* e plataformas sociais, juntamente com o engajamento social dos usuários sobre essas notícias.	fake (162), true (1.734)	Meios de comunicação confiáveis e sites de verificação de fatos.	1.896
Fact-Checked Images-WhatsApp [Reis et al., 2020b]	Imagens disseminadas no WhatsApp durante períodos eleitorais no Brasil e na India.	misinformation (1.032), not misinformation (780), random (1.261)	Agências de checagem de fatos.	3.073
Fact-Checked-Stat [Vlachos and Riedel, 2014]	Declarações verificadas de sites populares de verificação de fatos rotulados por jornalistas.	true (32), mostly true (34), half true (68), mostly false (37), false (49), fiction (1)	Jornalistas de agências de checagem de fatos.	221
FakeHealth [Dai et al., 2020]	Um repositório que consiste em dois conjuntos de dados, ou seja, Health-Story e HealthRelease e inclui conteúdos de notícias sobre saúde, análises de notícias, engajamentos sociais e redes de usuários.	fake (763), true (1.533)	Revisores especialistas no domínio da saúde.	2.296
Fake.Br Corpus [Monteiro et al., 2018]	Notícias verdadeiras e falsas que foram alinhadas manualmente, focando apenas no português brasileiro.	fake (3,600), true (3.600)	Pesquisadores.	7.200
Fake-News-Net[23] [Shu et al., 2017]	Um repositório para um projeto de coleta de dados em andamento para pesquisa de notícias falsas, incluindo conteúdo de notícias e recursos de contexto social com rótulos de notícias falsas de verdade de grupo confiáveis.	fake (211), real (211)	Jornalistas especialistas do BuzzFeed e checadores de fato do PolitiFact.com.	422
Fake-Real-News[24]	Artigos de notícias publicados durante 2015-2016, juntamente com seus títulos. Todo o corpus é construído a partir de notícias reais coletadas do *The New York Times*[25] e NPR[26], e notícias falsas de um conjunto de dados do Kaggle para garantir uma distribuição uniforme das amostras de ambas as classes.	fake (3.164), real (3.171)	Jornalistas para as notícias verdadeiras e anotadores humanos do BS Detector para notícias falsas.	6.335
Fake-Satire [Golbeck et al., 2018]	Conjunto de dados de notícias falsas e sátiras que são codificadas manualmente, verificadas e, no caso de notícias falsas, incluem histórias refutadas.	fake news (283), satire (203)	Pesquisadores com base em um artigo de um site de verificação de fatos ou em uma informação que refuta uma afirmação.	486
FA-KES [Salem et al., 2019]	Um conjunto de dados de notícias falsas sobre a guerra na Síria (ou seja, relatórios sobre incidentes de guerra ocorridos de 2011 a 2018).	fake (378), true (426)	Abordagem de rotulagem de verificação de fatos semi-supervisionada.	804
Fake-Twitter-Science [Vosoughi et al., 2018]	Todas as notícias verdadeiras e falsas verificadas distribuídas no Twitter de 2006 a 2017. Os dados compreendem ∼126.000 ocorrências (rumores em cascata) tuitadas por ∼3 milhões de pessoas mais de 4,5 milhões de vezes.	true (24.409), false (82.605), mixed (19.287)	Consenso entre verificadores de fatos de seis organizações independentes de checagem de fatos.	126.301
Kaggle[27]	Texto e metadados de fontes de notícias falsas e tendenciosas na Web de BS Detector[28].	bias (443), bs (11.492), conspiracy (430), fake (19), hate (246), junksci (102), satire (146), state (121)	Anotadores humanos do BS Detector.	12.997
LIAR [Wang, 2017]	Declarações curtas do PolitiFact.com rotuladas manualmente.	half-true (2638), false (2511), mostly-true (2466), barely-true (2108), true (2063), pants-fire (1050)	Checadores de fatos do PolitiFact.com.	12.836
NELA-GT-*[29] [Nørregaard et al., 2019; Gruppi et al., 2020]	Artigos de notícias de vários meios de comunicação, incluindo fontes convencionais, hiperpartidárias e de conspiração.	unreliable, mixed, reliable	Rótulos de informações básicas no nível da fonte de 7 locais de avaliação diferentes.	1,12M

oriundos destas plataformas, apresentamos uma visão geral da arquitetura por trás de um coletor do WhatsApp e de Telegram. As primeiras etapas para a coleta de dados em ambos aplicativos são a criação de um perfil (ou persona), a busca e a entrada nos grupos identificados como relevantes para a coleta. Na Figura 1.3 apresentamos um esquema de como esse processo é realizado desde a autenticação do celular, a busca em plataformas sociais até a entrada nos grupos. É importante mencionar que, para o primeiro passo, da criação da conta, é necessário a ativação do *SIM Card* com um número de celular válido e um *smartphone* conectado à Internet. Com o celular preparado, podemos instalar tanto o aplicativo do WhatsApp como do Telegram (entre outros mensageiros) e fazer a autenticação dele com o número dedicado para a coleta. Após a verificação, cria-se um perfil no aplicativo baseado no tema da coleta.

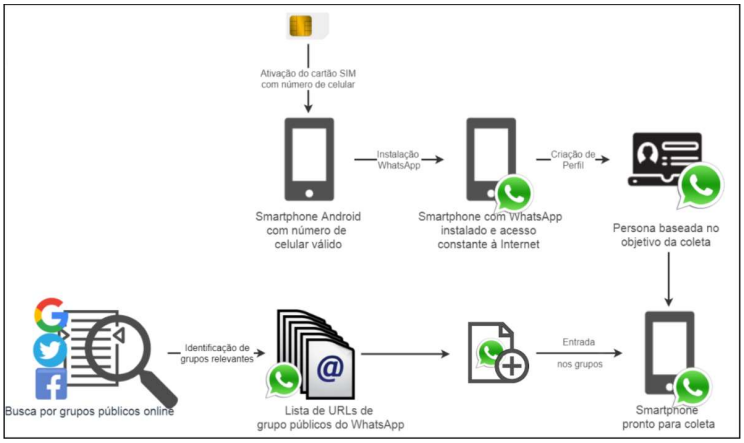


Figura 1.3. Visão geral sobre o processo de autenticação e entrada dos grupos.

Especificamente para o WhatsApp, outra etapa necessária é a preparação da conta de WhatsApp Web. Essa será a versão do WhatsApp utilizada por um dos coletores para acessar os dados através de *scripts* executados no servidor. Assim, é necessário fazer login no WhatsApp Web também por um computador na primeira vez que a ferramenta desenvolvida neste projeto for utilizada. A Figura 1.3 mostra um fluxograma de como se conectar na versão do WhatsApp Web. A partir do fluxograma, podemos observar que existe uma etapa manual que exige o uso do celular com o WhatsApp que se deseja coletar para fazer a leitura do QR Code exibido na página para conexão. Essa etapa é necessária apenas na primeira vez de configuração da coleta. Uma vez conectado, às demais instâncias de coleta já estarão com a conta devidamente logada. Entretanto, é válido destacar que eventualmente, a conta WhatsApp pode "deslogar-se" sozinha do navegador devido a atualizações do próprio WhatsApp, por exemplo, ou até mesmo caso o perfil do *browser* (e.g., Firefox) seja apagado da memória. Logo, esse processo poderá ser repetido sempre que necessário (i.e., quando a conta for desconectada do navegador). A seguir, apresentamos detalhes de cada uma das etapas.

Etapa 1 – Criação de uma Persona. Coletar dados do WhatsApp ou do Telegram requer uma interação muito maior com os outros usuários comparado a outras plataformas como Instagram, YouTube ou Twitter. É preciso efetivamente participar dos grupos no mesmo nível dos outros membros, não podendo somente observá-los a distância. Com isso, é necessária uma cautela maior para respeitar os Termos de Serviço do WhatsApp e também proceder com uma observação mais ética do projeto. Por isso, durante a criação do perfil para acompanhamento dos grupos públicos, exploramos o conceito de persona. Uma persona é uma representação fictícia e objetiva de um perfil de usuário, cuja idealização é baseada em pesquisas com usuários, observação de interesses, desejos e necessidades. Este é um mecanismo frequentemente utilizado durante o processo de desenvolvimento de um *software*. Aqui, esta abordagem foi utilizada para a coleta dos grupos de plataformas de mensagens instantâneas, uma vez que ela é capaz de sintetizar a aparência de um usuário convencional dos aplicativos alvo.

Para criação de uma persona para os grupos de WhatsApp ou Telegram, primeiro é necessário responder algumas perguntas relacionadas à sua personalidade, vida e o uso do aplicativo, tendo em mente os grupos de interesse para observação. As perguntas utilizadas são: *(a)* Qual o gênero da pessoa?; *(b)* Qual sua idade?; *(c)* A pessoa é casada,

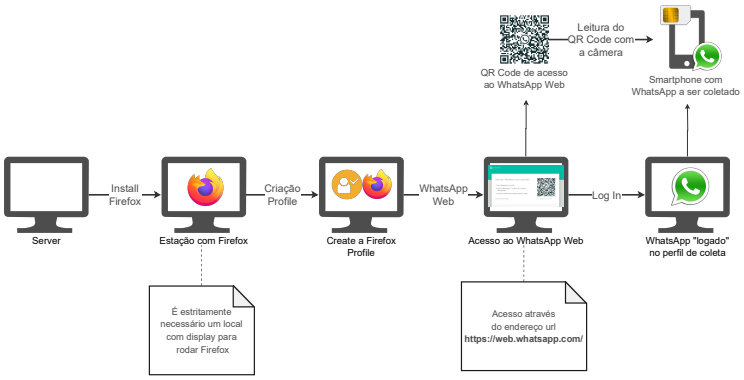


Figura 1.4. Fluxograma de conexão com o WhatsApp Web.

ou está em um relacionamento?; *(d)* A pessoa tem filhos?; *(e)* A pessoa mora sozinha, ou divide a casa? Com quem?; *(f)* Qual seu grau de escolaridade (e.g., ensino fundamental completo, ensino médio completo)?; *(g)* A pessoa trabalha? Qual o seu trabalho?; *(h)* Por que a pessoa utiliza a plataforma?; *(i)* Por quanto tempo ela utiliza o aplicativo de mensagens?; *(j)* A pessoa acessa pelo celular, computador, ou tablet?; e, *(k)* Por quais grupos ela se interessa?

A partir dessas questões é possível imaginar um usuário alvo da coleta a ser realizada, com uma personalidade que se adéque aos grupos desejados, e não seja identificada como um perfil não autentico pelos outros membros do grupo. Ademais, para a criação de personas é sugerido o uso de imagens sem restrições de uso comercial, ou imagens de pessoas que não existam (i.e., criadas com o auxílio de ferramentas de inteligência artificial). Seguindo o ideal da persona desenvolvida, além de uma metodologia já consolidada de etnografia virtual, é criado também um ponto de referência fixado na realidade dos usuários, mantendo a seleção e entrada nos grupos públicos em acordo com a proposta do projeto. Além disso, seguir a persona é uma forma de garantir que o perfil respeite os Termos de Serviço do WhatsApp, evitando, entre outras coisas, o banimento da plataforma.

Etapa 2 – Busca de Grupos Públicos Relevantes. A segunda etapa, que pode acontecer paralelamente à anterior, está relacionada a busca por grupos de interesse que serão coletados. Tanto o WhatsApp, o Telegram como algumas outras plataformas de mensagens instantâneas possuem grupos públicos de conversa nos mais variados temas. Esses grupos públicos são criados por usuários e compartilhados na Web através de URLs de convites. A ideia é que outras pessoas possam participar livremente das conversas. Como o objetivo desses usuários é divulgar seus grupos para agregar mais membros, é possível achar facilmente na Internet, como em outras plataformas sociais ou ainda em *websites* de buscas, dezenas e até centenas de grupos de tópicos variados. Existem até *websites* que funcionam como repositório próprios de grupos de WhatsApp separados por categorias[31].

Uma vez definido um foco para coleta e a persona que participará dos grupos, precisamos então levantar uma lista dessas URLs de grupos púbicos considerados relevantes para a coleta. Por exemplo, grupos públicos relacionados à política ou então com informações relacionadas à pandemia de COVID-19. De posse da lista de grupos, ainda é necessário "entrar" nesses grupos. Esse processo pode ser feito de forma manual, aces-

[31] `https://gruposdezap.com/`

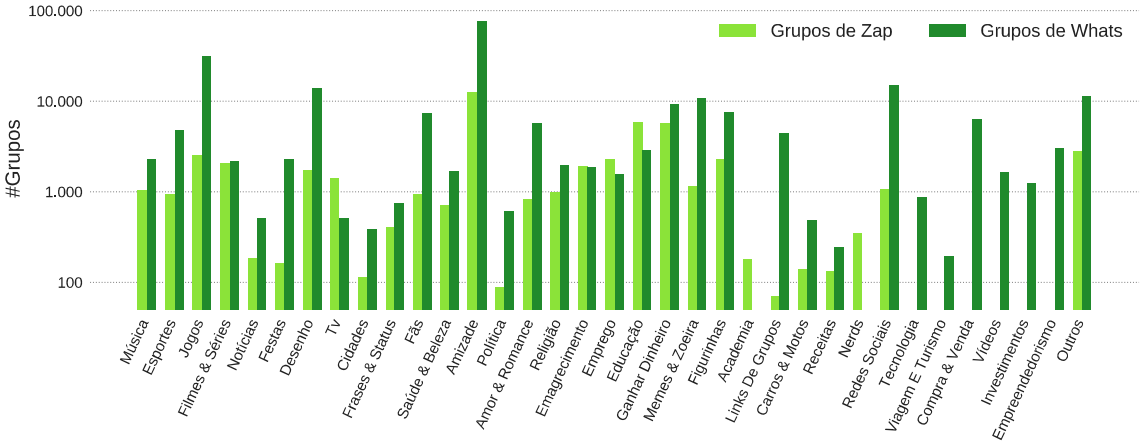


Figura 1.5. Quantidade de grupos criados por categoria nos repositórios online.

sando o aplicativo através do celular da coleta e ir clicando grupo a grupo e dando o "*Join*" naqueles grupos, ou, quando a lista é muito extensa, também é possível automatizar esse processo. No Telegram, através da API do sistema, podemos ir "entrando" em grupos de interesse, dado seu ID. Já no WhatsApp, que não dispõe de uma API, isso pode ser realizado por meio de um *script* Python com uso do *Selenium*[32], por exemplo, que faça uso do WhatsApp Web para simular o processo manual que um usuário faria indo por cada link de grupo público e "participando" dele.

Note porém, que a existência da URL de um grupo público não garante o acesso a ele. As URLs podem ser revogadas pelos administradores dos grupos, restringindo o acesso e, mesmo após a entrada em um grupo, é possível que a persona criada seja banida (expulsa) dele de acordo com a vontade dos administradores ou baseado em regras específicas definidas pelo grupo.

Em um trabalho desenvolvido recentemente pelo nosso grupo de pesquisa [Kansaon et al., 2022], nós mapeamos de forma geral as categorias de grupos de WhatsApp criados no Brasil. Para isso avaliamos dois grandes *websites* repositórios de grupos e também aqueles compartilhados no Twitter e Facebook. Com isso descobrimos as principais categorias e tópicos de grupos públicos criados no Brasil. Esta investigação nos ajuda a planejar melhor novas pesquisas que envolvem este tipo de conteúdo, uma vez que possibilita entender melhor o ecossistema do WhatsApp e que tipo de conteúdo é criado nele.

Na Figura 1.5 apresentamos as principais categorias de grupos criados no Brasil de dois dos principais repositórios de grupos no país, a saber: "Grupos de Zap" e "Grupos de Whats". É interessante observar o uso dos grupos pelos usuários, com a categoria Amizade sendo a mais popular. Além disso, outras categorias também se destacam, como a categoria de Jogos (33.954); Desenho (15.603), considerando desenhos animados e animes; Memes e Zoeira (11.846), com compartilhamento de piadas, memes e sátiras; Esportes (5.688), composto na maioria por grupos de futebol, e de Redes Sociais (16.192), com vários grupos com foco em outras plataformas como YouTube, Facebook, TikTok, Twitter e Instagram. Uma grande parte dos grupos desta categoria são sobre ganhar seguidores e curtidas em outras plataformas digitais.

Ao compartilhar grupos nesses ambientes, os usuários fornecem informações de

[32]`https://selenium-python.readthedocs.io/`

(a) Twitter (b) Facebook

Figura 1.6. Nuvem de termos das mensagens contendo grupos de WhatsApp.

contexto que descrevem o tema dos grupos. Assim, também analisamos os tópicos dos grupos compartilhados no Twitter e no Facebook. As nuvens de termos da Figura 1.6 de ambas as plataformas, revela diferenças marcantes entre elas. No Twitter, termos com conotação sexual são mais frequentes, enquanto no Facebook prevalecem palavras religiosas. As mensagens também fazem uso de termos metalinguísticos para convidar pessoas a ingressar em grupos do WhatsApp. Muitas palavras referem-se a outras plataformas sociais, como Telegram, TikTok e Instagram, indicando uma potencial relação entre elas. Além disso, foram encontrados termos relacionados à venda de produtos e serviços, sugerindo que o WhatsApp é muitas vezes usado como canal para negociações diversas.

Etapa 3 – Extração, Armazenamento e Processamento dos Dados. Finalmente, a Figura 1.7 apresenta uma visão geral de todos os passos necessários para a coleta de dados dessas plataformas, bem como os objetos envolvidos durante cada etapa do processo de execução: *(1)* O celular e o computador configurados para realizar a coleta, ambos conectados ao WhatsApp/Telegram; *(2)* Um *script* observa os dados recebidos nos grupos monitorados e extrai as mensagens; *(3)* As mensagens são estruturadas e salvas em JSON; *(4)* Os arquivos de mídia (i.e., imagem, vídeo e áudio) são baixados e salvos no servidor; *(5)* Outro *script* é responsável por analisar esses arquivos e agrupar as mensagens por similaridade; *(6)* As mensagens agrupadas são salvas em JSON contendo todas vezes que elas foram compartilhadas.

É importante mencionar que propusemos e adotamos uma estratégia mais pragmática para obtenção de metadados relacionados a cada grupo de interesse, coletando as informações imediatamente após o ingresso no grupo descoberto. Ademais, abordamos a metodologia exata para cada plataforma de mensagens da seguinte maneira. Para subconjunto dos grupos monitorados, complementamos os metadados básicos do grupo com detalhes sobre a estrutura e atividade dentro deles. Como a metodologia para esta etapa difere drasticamente para cada plataforma, abaixo, apresentamos detalhes relativos à obtenção de dados dos grupos para cada uma delas (i.e., WhatsApp e Telegram).

WhatsApp. Com a falta de suporte de API para o WhatsApp, há uma dependência da interface Web do WhatsApp para ingresso nos grupos e coleta das informações de interesse. Neste contexto propusemos e desenvolvemos uma abordagem em Python utilizando *Selenium* para salvar os dados enviados nos grupos da conta monitorada. A partir desse código implementado, temos acesso a: *(i)* mensagens dos grupos (o WhatsApp dá acesso apenas às mensagens enviadas após a data de ingresso); *(ii)* números de telefone de todos os membros do grupo; e *(iii)* data de criação do grupo.

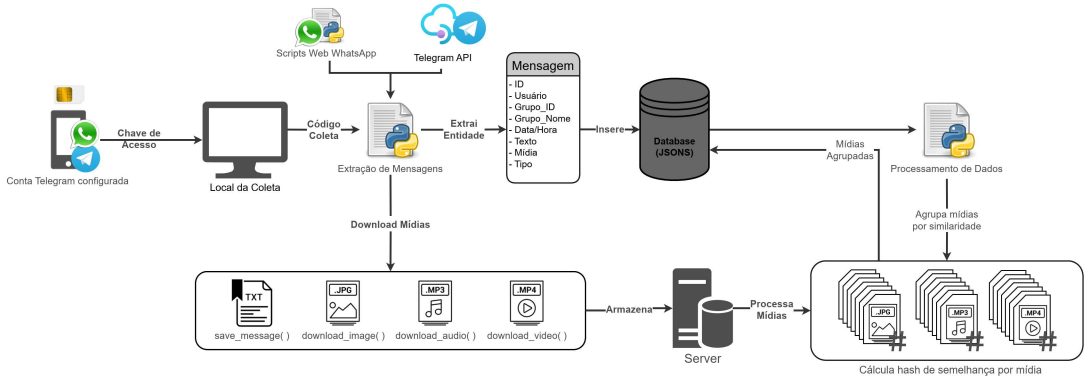


Figura 1.7. Visão geral da arquitetura da coleta de dados em aplicativos de mensagem instantânea.

Telegram. Por outro lado, obter dados dos grupos do Telegram é uma tarefa menos complexa em comparação ao WhatsApp devido sua API [33]. Logo, após ingressar nos canais e grupos, temos mais informações e também acesso a todas as mensagens, incluindo aquelas anteriores ao nosso ingresso. Para cada grupo, coletamos: *(i)* mensagens dos grupos (todas as mensagens desde a criação do grupo/canal); *(ii)* data de criação do grupo; e, *(iii)* perfis de usuário dos membros do grupo.

Aqui, vale notar uma etapa crucial para o funcionamento do sistema: o agrupamento de conteúdo semelhante. Em outras plataformas sociais, por exemplo, quando nos referimos a coleta de uma mensagem postada no Facebook ou Twitter, as informações disponibilizadas comumente contém metadados relacionados (e.g., número de curtidas ou compartilhamentos da referida mensagem dentro da plataforma). Porém, no WhatsApp, cada mensagem, cada imagem, vídeo ou áudio é postada de forma independente e isolada. Para que seja possível, por exemplo, mensurar quantas vezes um determinado conteúdo foi compartilhado na plataforma, é necessária a execução de um processo específico que envolve o processamento, rastreamento, agrupamento e contagem de um determinado conteúdo. Portanto, para saber que uma única mensagem foi compartilhada mais de uma vez, precisamos fazer esse processamento de forma manual. Este processo é apresentado em detalhes no fluxograma da Figura 1.8. Aqui vemos como as mensagens isoladas, mas que correspondem ao mesmo conteúdo, são agrupadas pela nossa metodologia. Quando dois usuários compartilham a mesma imagem, por exemplo, elas são salvas de forma isolada pelo coletor. Para traçar essa disseminação de um conteúdo de mídia através do WhatsApp realizamos um agrupamento dos conteúdos por similaridade usando um sistema de *hashes*. Para os arquivos multimídia recebidos, então, o *script* de processamento realiza os seguintes passos: *(1)* baixa os arquivos anexados a uma mensagem durante a coleta; *(2)* calcula a *hash* de cada um desses arquivos[34]; *(3)* cria-se um dicionário *hash* das mídias coletadas, de forma que duas mídias idênticas possuem a mesma hash, podendo assim identificar mensagens distintas que compartilharam o mesmo conteúdo; *(4)* agrupa, finalmente, as mensagens com todas as ocorrências em que ela foi compartilhada, armazenando dados como total de vezes que ela foi enviada, quais foram os usuários e grupos que compartilharam este conteúdo e datas de postagem.

Por fim, as mensagens coletadas especificamente no WhatsApp foram utilizadas

[33] `https://core.telegram.org/method/channels.joinChannel`

[34] Usando o *checksum* MD5 do arquivo para áudios e vídeos e uma *perceptual hash* para imagens.

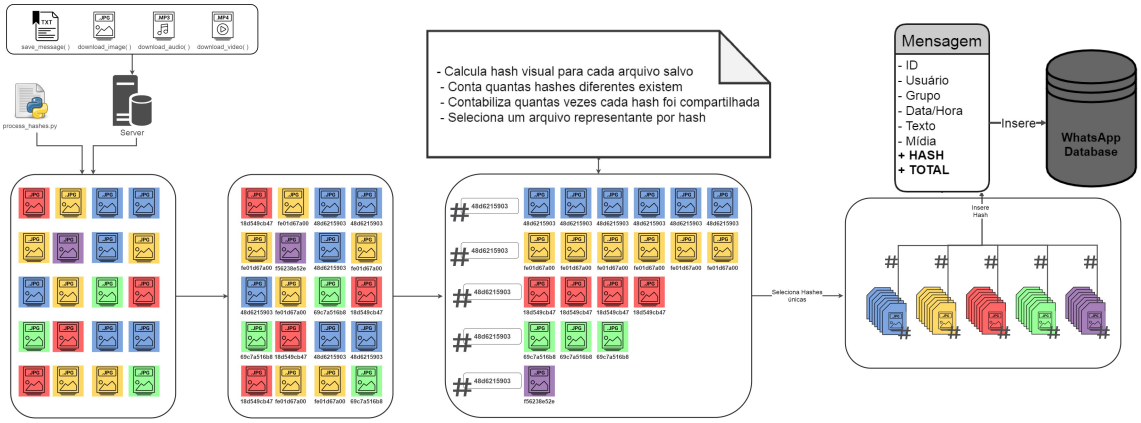


Figura 1.8. Abordagem de agrupamento do conteúdo por semelhança de *hash*.

para a construção de um conjunto de dados disponível em `https://zenodo.org/record/3779157` que tem sido amplamente utilizado na literatura para entendimento do fenômeno da desinformação bem como para proposição de abordagens que possam ser úteis para contenção do problema nesses ambientes [Reis et al., 2020a; Reis and Benevenuto, 2021; De Angeli and Reis, 2022].

1.5. Entendimento/Caracterização da Desinformação

Embora o fenômeno da desinformação em si não seja um problema novo[35], recentemente, alguns esforços estão emergindo com o objetivo de melhor compreendê-lo em diferentes plataformas digitais, como Twitter, WhatsApp, Facebook, Telegram, etc. Particularmente, Vosoughi et al. [Vosoughi et al., 2018] mostra que notícias contendo desinformação tendem a se espalhar mais rapidamente do que notícias contendo histórias verdadeiras. Lazer et al. [Lazer et al., 2018] convocam uma força-tarefa interdisciplinar para abordar esse problema complexo. No entanto, existem algumas características inerentes às próprias plataformas digitais que contribuem para a disseminação deste tipo de conteúdo nesses ambientes. Em suma, estes esforços para entendimento do fenômeno estão focados em diferentes aspectos, tais como as características do conteúdo, dinâmica de propagação (e.g., compartilhamentos), dentre outros.

Assim, nesta seção, serão discutidas as principais estratégias empregadas para entendimento do fenômeno da desinformação em plataformas digitais e meios de investigar sua disseminação através de diferentes plataformas, como, por exemplo, no WhatsApp, onde o nosso grupo de pesquisa se destaca como um dos pioneiros no Brasil [Resende et al., 2019]. Neste contexto, a natureza fechada do aplicativo, juntamente com a facilidade de compartilhamento de informações em grupos de grande escala, tornam o WhatsApp único entre outras plataformas, onde mensagens criptografadas anônimas podem se tornar virais, alcançando vários usuários em um curto intervalo de tempo. A sensação de pessoalidade e a imediatidade das mensagens foram amplamente abusadas para espalhar rumores infundados e criar campanhas de desinformação durante as eleições recentes no Brasil e na Índia [Reis et al., 2020a]. Apesar do esforço para combater o problema, não há evidências até o momento sobre a real eficácia de tais restrições. Aqui, investigamos e

[35] `http://www.politico.com/magazine/story/2016/12/fake-news-history-long-violent-214535`

propomos uma série de estratégias para, por exemplo, investigar e entender a eficácia dessas medidas na disseminação de desinformação circulando no WhatsApp. Através do uso de um modelo epidemiológico com dados reais coletados do WhatsApp no Brasil, Índia e Indonésia, avaliamos o impacto da viralidade nesse tipo de rede [Melo et al., 2019b]. Conforme mencionado anteriormente, discutiremos estas, e outras abordagens exploradas na literatura para entendimento/caracterização do problema em plataformas digitais, nesta seção.

1.5.1. Propagação de (Des)Informação no WhatsApp

Existem algumas características-chave que tornam o WhatsApp único em relação a outras plataformas sociais. Em primeiro lugar, o WhatsApp permite a conexão entre pessoas com interesses semelhantes por meio de grupos de bate-papo. Esses grupos têm um limite de 256 usuários e podem ser privados ou públicos. No caso dos grupos privados, novos membros precisam ser adicionados por um membro que assume o papel de administrador do grupo. Para os grupos públicos, o acesso é feito por meio de links de convite que podem ser compartilhados com qualquer pessoa ou disponibilizados na Web. Esses grupos públicos são frequentemente usados para discutir hobbies, paixões e também tópicos específicos, como saúde, educação e política.

Embora a maioria dos grupos seja privada e formada por pessoas com algum relacionamento social (por exemplo, família, amigos, colegas de trabalho), os grupos públicos têm sido uma característica catalisadora para a difusão de informações neste ecossistema: a maioria de seus membros são desconhecidos entre si. Isso é evidente em países como o Brasil, onde uma pesquisa relatou que 76% dos usuários do WhatsApp fazem parte de grupos, 58% participam de grupos com pessoas desconhecidas e 18% desses grupos discutem política [Newman et al., 2019]. Desta forma, grupos públicos de WhatsApp podem atuar como um atalho para que a informação percorra partes distantes da estrutura da rede social subjacente por meio de laços fracos, ampliando e acelerando a disseminação de informações [Bakshy et al., 2012]. Além disso, o aplicativo facilita ainda mais a propagação de informação uma vez que possui duas funções de compartilhamento: o *broadcast*, em que uma lista de contatos pode ser criada para enviar mensagens para até 256 contatos (usuários ou grupos) de uma só vez, e o *forward*, em que uma única mensagem recebida pode ser prontamente encaminhada para outros contatos (usuários ou grupos). Essas características permitem que a mensagem percorra rapidamente longas distâncias pela rede. Por outro lado, a criptografia ponta-a-ponta implementada pelo sistema dificulta a identificação da origem e o rastreamento da propagação destas mensagens.

Devido a essas peculiaridades, o WhatsApp tem gerado uma controvérsia relacionada às suas características de anonimato e viralidade. Esse conflito se deve ao fato de que podemos enxergar o WhatsApp de duas maneiras diferentes: como uma empresa de tecnologia que oferece um serviço de mensagens seguras e privadas e, ao mesmo tempo, como uma plataforma de mídia de comunicação em massa. Por um lado, ele garante o anonimato, privacidade e a segurança do usuário por meio da criptografia de dados. Como meio de comunicação em massa, ele transmite informações e dissemina conteúdo em grande escala em um curto espaço de tempo. Assim, mensagens enviadas anonimamente alcançam rapidamente milhares de pessoas sem qualquer regulação ética ou legal desse conteúdo disseminado, promovendo, por exemplo, campanhas de desinformação. A

disseminação massiva de (des)informação e boatos [Arun, 2019] levou, inclusive, a pedidos tanto dos governos nacionais da Índia e posteriormente do Brasil [Melo et al., 2019b] para o WhatsApp para que eles alterassem recursos tentando minimizar os danos causado pela plataforma sendo usada para espalhar desinformação em grande escala. Como consequência, o WhatsApp implementou restrições na forma como as mensagens são encaminhadas[36], reduzindo o limite de encaminhamento de conteúdo para um máximo de usuários/grupos simultâneos. No entanto, não existem estudos que investiguem o impacto dessas limitações ou se os números escolhidos pela empresa são suficientes para barrar eficientemente a propagação de conteúdo viral.

Nesta seção, avaliamos a dinâmica da propagação da (des)informação em uma rede de grupos públicos do WhatsApp. Mais especificamente, investigamos a anatomia dessa plataforma para compreendermos suas peculiaridades, bem como respondermos à pergunta de como as ferramentas de encaminhamento contribuem para a viralidade de (des)informação. Além disso, se as limitações impostas no sistema são realmente capazes de evitar a disseminação de conteúdo. Por fim, também propomos uma estratégia para medição dessa propagação em rede do WhatsApp e discutimos algumas soluções de como o problema da disseminação em larga escala pode ser contido/combatido.

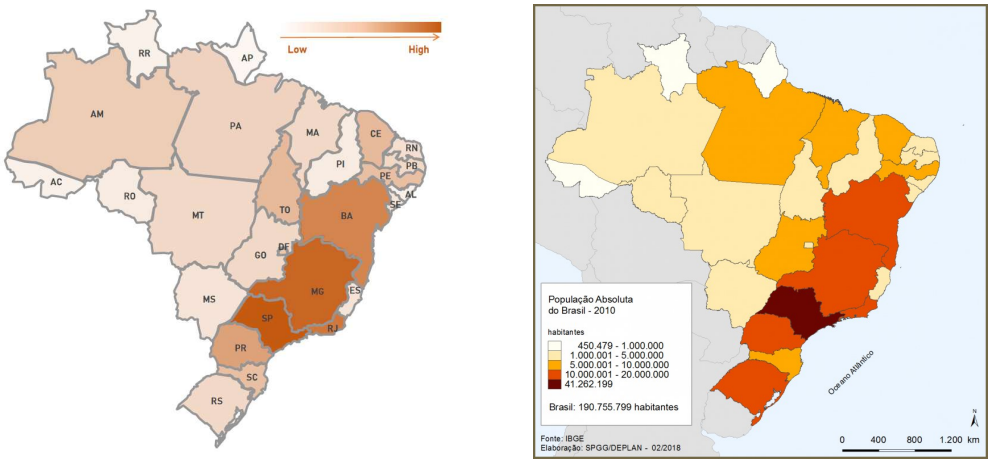
1.5.1.1. A Estrutura de Rede do WhatsApp

Iniciamos esta análise exploratória com a coleta de dados do WhatsApp para obter informações sobre a estrutura da rede e suas características. Os dados utilizados foram coletados por [Resende et al., 2019] durante o período eleitoral brasileiro entre 16 de setembro e 5 de outubro de 2018. Os dados analisados se referem especificamente às imagens compartilhadas em 364 grupos políticos públicos; adquiridos a partir de uma busca na Web utilizando de uma lista de palavras-chave relacionadas à política e notícias, juntamente com o link `chat.whatsapp.com`, para identificar e coletar grupos públicos do WhatsApp por meio de pesquisas no Google, Twitter e Facebook. Esta coleta, em particular, foi realizada utilizando a ferramenta WebWhatsAppAPI, que permite a extração de mensagens por meio da versão Web do WhatsApp. Além disso, complementamos nosso conjunto de dados com conteúdos semelhantes de grupos públicos de WhatsApp da Índia e da Indonésia em uma coleta em parceria com autores de [Garimella and Tyson, 2018].

Após a coleta dos dados, procedemos com uma caracterização inicial dos dados e a reconstrução da estrutura de rede dos grupos públicos de WhatsApp que estes grupos constituíam a fim de entender as propriedades básicas desta plataforma. O objetivo é identificar comportamentos e vieses dos dados que diferenciam a rede de grupos do WhatsApp de outras redes sociais, como Facebook e Twitter.

Além disso, exploramos a construção de um modelo generativo para criar um grafo que representasse adequadamente a rede de usuários do WhatsApp, levando em consideração suas características peculiares, como a presença de grupos de conversa. Propusemos uma divisão da rede em dois tipos de grupos: orgânicos e artificiais. Essa distinção permitiu uma análise mais aprofundada da estrutura da rede e das interações entre os grupos e seus membros.

[36] `blog.whatsapp.com/10000647/More-changes-to-forwarding`



(a) Usuários do WhatsApp por estado

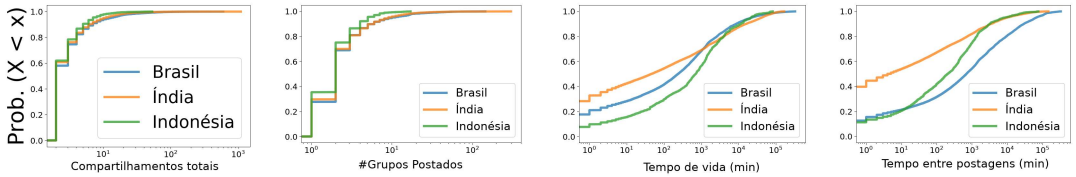
(b) População absoluta do Brasil.
Fonte: IBGE/Censo Demográfico 2010

Figura 1.9. Comparação populacional entre a geolocalização dos usuários do *WhatsApp* baseado no código DDD de telefone e população do Brasil.

Nossa primeira análise consiste em identificar os padrões geográficos contidos nos dados. Como cada usuário está associado a um número de telefone, é possível localizar geograficamente os usuários através de seu código DDD. A Figura 1.9 apresenta o mapa coroplético das localizações dos 10 mil usuários coletados distribuídos pelas unidades federativas do Brasil. Cores mais escuras indicam maior concentração de usuários naquele estado. Os estados com maior número de usuários incluem SP (1322), MG (1116), RJ (992) e BA (905). Esses valores confirmam o viés geográfico da distribuição real 1.9(b) da população sobre o território brasileiro, o que demonstra que os dados possuem uma amostra de usuários semelhante à distribuição populacional brasileira. Entretanto, podemos observar maiores discrepâncias para os estados do RS (275), que aparece subrepresentado, TO (479) e DF (380), que possuem proporcionalmente mais usuários do que a distribuição de população. Como se tratam de dados políticos, relacionamos esse aumento no DF à sua proximidade da vida política do país. Já Tocantins aparece com uma concentração também acima do normal devido a uma grande quantidade de grupos monitorados referentes àquele estado.

Uma vez que coletamos e identificamos as ocorrências dos conteúdos compartilhados nesses grupos, podemos observar a cobertura e a dinâmica de propagação dessas imagens em nossos dados. Para avaliar métricas de propagação ao longo do tempo e cobertura, consideramos apenas as imagens que foram compartilhadas pelo menos duas vezes, pois não podemos observar o efeito da propagação de imagens que são postadas apenas uma vez. Esse conjunto consiste em 2.384 imagens na Indonésia, 103.031 imagens no Brasil e 44.731 imagens na Índia, o que representa aproximadamente 20% das imagens em cada país. Embora quase 80% das imagens no WhatsApp tenham sido postadas apenas uma vez, existem algumas imagens muito populares que foram amplamente compartilhadas e atingiram vários grupos.

Primeiro, calculamos o número total de compartilhamentos de cada imagem e em quantos grupos elas apareceram. As Figuras 1.10(a) e 1.10(b) mostram a função de distribuição cumulativa (CDF) do número total de compartilhamentos e do número de grupos distintos em que cada imagem apareceu. É possível observar que existem algumas



(a) Compartilhamentos (b) Grupos (c) Tempo de vida (d) Intervalo entre postagens

Figura 1.10. CDF de cobertura de compartilhamento e métricas de dinâmica de tempo de imagens compartilhadas pelo menos duas vezes no WhatsApp.

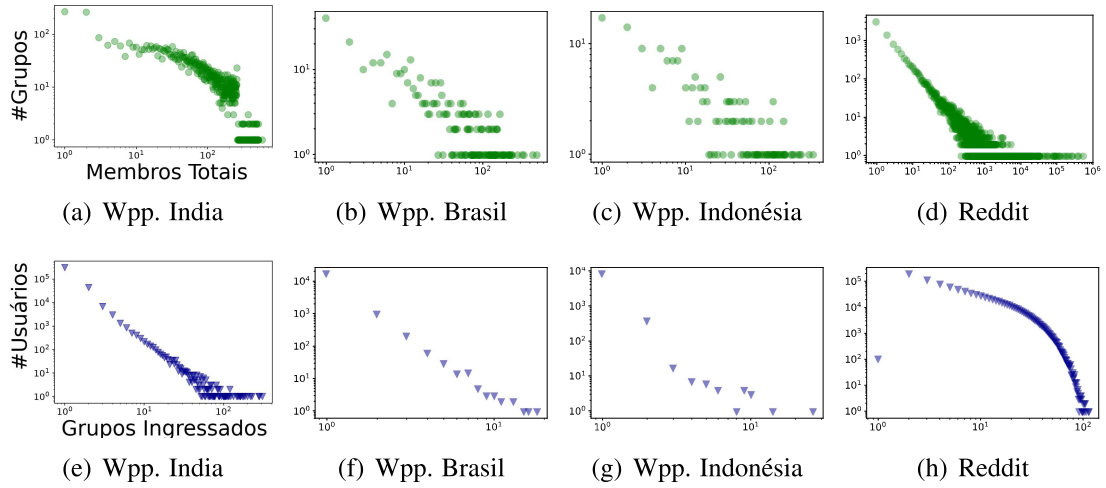
imagens muito populares que foram amplamente compartilhadas mais de 500 vezes no Brasil e mais de mil vezes na Índia, e alcançaram mais de 100 grupos em ambos os países. Embora a maior parte das imagens tenha sido compartilhada poucas vezes, isso mostra que o WhatsApp pode ser usado não apenas para conversas particulares, mas também como um meio de comunicação em massa com potencial de viralização de seu conteúdo.

Além de analisar a propagação das imagens no WhatsApp, também analisamos sua "vida útil"na Figura 1.10(c). A vida útil é determinada pela diferença entre a última e a primeira ocorrência da imagem em nosso conjunto de dados. Em resumo, embora a maioria das imagens (80%) dure no máximo 2 dias, existem imagens no Brasil e na Índia que continuaram a aparecer mesmo após 2 meses da primeira aparição (100.000 minutos). Também podemos observar que a maioria (60%) das imagens é postada antes de 1.000 minutos após sua primeira aparição. Além disso, no Brasil e na Índia, cerca de 40% dos compartilhamentos foram feitos após um dia de sua primeira aparição e 20% após uma semana. Em uma análise mais detalhada, na Figura 1.10(d), mostramos a distribuição dos "tempos entre eventos"entre postagens da mesma imagem. Observamos que o tempo entre eventos das imagens na Índia é muito mais rápido do que no Brasil e na Indonésia, ou seja, mais de 50% das postagens são feitas em intervalos de 10 minutos ou menos, enquanto apenas 20% dos compartilhamentos foram feitos nesse mesmo intervalo de tempo no Brasil e na Indonésia. Analisamos manualmente as razões por trás do curto período de tempo entre as postagens e descobrimos que nos dados da Índia há um comportamento automatizado semelhante a spam em comparação com o Brasil e a Indonésia.

Esses resultados sugerem que o WhatsApp é uma rede muito dinâmica e a maior parte do seu conteúdo de imagens é efêmera, ou seja, as imagens geralmente aparecem e desaparecem rapidamente. A arquitetura linear do sistema de chat de conversas dificulta que um conteúdo antigo seja revisitado, mas há alguns que permanecem na rede por mais tempo, disseminando-se ao longo de semanas ou mesmo meses.

Na Figura 1.11, mostramos a distribuição de grupos por usuário e usuários por grupo. Para comparar as peculiaridades do WhatsApp com outras plataformas populares, também usamos dados da rede Reddit, modelando os subreddits como grupos e os usuários como membros. Observe que, embora o Reddit tenha a mesma característica de grupos, queremos avaliar recursos específicos do WhatsApp que levam a estruturas de rede muito diferentes. O limite de 256 membros nos grupos é um elemento determinante na rede, capaz de limitar o tamanho do grupo, principalmente na Índia (Figura 1.11(a)),

onde existem mais de 300 mil usuários e mais de 5 mil grupos.[37] Por outro lado, no Reddit, onde não há limite, é possível ver que o tamanho do grupo pode ser tão grande quanto 10^5 membros, o que cria grandes concentrações de usuários. Como ambas as plataformas não têm limite para o número de grupos aos quais os usuários podem aderir, esperávamos não ver diferenças no número total de grupos dos quais os usuários participam. No entanto, observe que no Reddit, a distribuição tem um decaimento exponencial, com um limite de aproximadamente 100 grupos. Por outro lado, todas as curvas do WhatsApp são semelhantes, seguindo uma curva de lei de potência bem comportada, o que naturalmente gera uma variância maior. Observe que na Índia temos usuários que participaram de mais de 300 grupos.



(a) Wpp. India (b) Wpp. Brasil (c) Wpp. Indonésia (d) Reddit

(e) Wpp. India (f) Wpp. Brasil (g) Wpp. Indonésia (h) Reddit

Figura 1.11. Distribuições do número de membros por grupo e total de grupos por usuário no WhatsApp (Wpp) e no Reddit.

Em seguida, investigamos a estrutura de rede dos grupos públicos do WhatsApp e comparamos suas características com outras redes sociais reais e sintéticas. Para reconstruir a rede a partir dos grupos do WhatsApp coletados, criamos um grafo em que conectamos dois grupos se eles compartilham um usuário em comum. Embora o WhatsApp seja um aplicativo de chat pessoal criptografado, a possibilidade de criar grupos públicos permite que vários usuários distantes socialmente se conectem uns aos outros por meio da rede, formando uma estrutura social complexa capaz de fluir grandes volumes de informações.

Na Figura 1.12, mostramos essas redes para os três países, onde cada nó representa um grupo e as arestas conectam os nós que possuem membros em comum. O tamanho do nó é proporcional ao número de membros do grupo. Colorimos os nós de acordo com sua comunidade nesse grafo, seguindo o algoritmo de modularidade proposto por [Blondel et al., 2008]. Observe que em todos os gráficos há um componente conectado principal evidente e outros agrupamentos de grupos. Além disso, note que alguns grupos se posicionam como pontes e *hubs*, conectando diferentes comunidades na estrutura da rede.

[37]Em nossos dados, alguns grupos têm mais de 256 membros, porque nossos dados são uma captura temporal e os membros podem sair e entrar nos grupos durante esse tempo.

(a) Brasil (b) Índia (c) Indonésia

Figura 1.12. Rede de grupos públicos do WhatsApp para cada país. Cada nó é um grupo e as arestas representam membros em comum.

Esses resultados demonstram que o WhatsApp apresenta uma estrutura de rede social semelhante a outras redes sociais estudadas. A existência de um componente conectado principal indica que há uma grande interconectividade entre os grupos, permitindo a disseminação de informações de forma eficiente. Além disso, a presença de agrupamentos de grupos indica a existência de comunidades ou temas específicos dentro da rede do WhatsApp. Ao identificar grupos que atuam como "pontes" e "hubs", podemos inferir que esses grupos desempenham um papel fundamental na conectividade entre diferentes comunidades na rede. Eles podem ser responsáveis por ampliar o alcance das informações, permitindo que elas se espalhem entre grupos distintos. Essa característica do WhatsApp como uma plataforma de comunicação em grupo pode ser explorada para fins de disseminação de informações, seja de maneira positiva ou negativa.

Em suma, o estudo da estrutura de rede dos grupos do WhatsApp nos permite entender melhor como as informações são compartilhadas e propagadas dentro da plataforma. Isso pode ter implicações significativas para a compreensão do papel do WhatsApp como meio de comunicação e para o desenvolvimento de estratégias de gerenciamento de informações na era digital.

A seguir, comparamos as características destas redes de grupos do WhatsApp com outras redes: *(i)* gráficos gerados aleatoriamente usando o modelo Barabasi-Albert de rede livre de escala, o modelo Erdős–Rényi, o modelo de mundo pequeno [Watts and Strogatz, 1998] e o modelo de rede *Forest Fire* [Leskovec et al., 2005], para os quais usamos o mesmo número de nós no conjunto de dados da Índia para criar uma rede comparável; *(ii)* a rede de subreddits do Reddit [Olson and Neal, 2015]; e *(iii)* a rede do Flickr [McAuley and Leskovec, 2012], que, diferentemente das redes de grupos do WhatsApp e do Reddit, representa a rede de imagens compartilhadas pelos usuários na plataforma.

Os resultados são apresentados na Tabela 1.2. Observamos que o WhatsApp compartilha características comuns com outras redes sociais do mundo real: alto coeficiente de agrupamento, maior componente conectado gigante e pequeno comprimento médio do caminho, que são propriedades típicas de uma rede social. O WhatsApp também apresenta um coeficiente de Pearson mais alto, o que significa que os nós tendem a se conectar com outros nós que possuem valores de grau semelhantes. Em análises de epidemia, isso pode ajudar a entender a propagação de infecções na rede, pois uma campanha de desin-

Tabela 1.2. Métricas de rede para o WhatsApp em comparação com outras redes.

	#Nós	#Arestas	Grau Médio	Coeficiente Clusterização	Diâmetro	APL*	Densidade	LCC**	Coeficiente Pearson
Wapp. India	5,839	407,081	69.71	0.59	11	3.17	0.0239	92.6%	0.295
Wapp. Brazil	414	1,400	6.76	0.32	8	3.19	0.0164	65.2%	0.346
Wapp. Indonesia	217	699	6.44	0.38	9	3.09	0.0298	55.3%	0.290
Bar.-Albert	5,839	792,300	271.38	0.10	3	1.95	0.0465	100%	0.008
Erdos-Renyi	5,839	1,534,952	525.76	0.09	2	1.91	0.0901	100%	-0.001
Smallworld	5,839	604,250	206.97	0.34	3	1.98	0.0355	100%	0.007
ForestFire	5,839	12,930	4.43	0.42	17	5.25	0.0008	100%	-0.066
Reddit	15,122	4,520,054	597.81	0.82	6	2.03	0.0395	99,8%	-0.045
Flickr	105,938	2,316,948	43.74	0.09	9	4.8	0.0004	99.8%	0.247

*Comprimento Médio do Caminho (do Inglês, *Average Path Length)*
**Maior Componente Conectado (do Inglês, *Largest Connected Component)*

formação direcionada a grupos de alto grau provavelmente se espalhará para outros nós de alto grau.

Essas semelhanças indicam que o WhatsApp possui uma estrutura de rede social robusta, compartilhando características com outras redes sociais populares. Isso destaca a importância do WhatsApp como uma plataforma de comunicação e troca de informações entre os usuários. Compreender a estrutura dessa rede pode ser útil para analisar a disseminação de informações e desenvolver estratégias para lidar com a propagação de conteúdos falsos ou prejudiciais.

Ao compreender a estrutura da rede do WhatsApp e suas particularidades, podemos avançar nas análises relacionadas à propagação de (des)informação e entender como as ferramentas de encaminhamento contribuem para a viralidade de conteúdo. Essas informações são cruciais para o desenvolvimento de estratégias eficazes de combate à disseminação de desinformação em larga escala no WhatsApp e em outras plataformas de comunicação.

1.5.1.2. Propagação de Informação através do Modelo Suscetível-Exposto-Infectado

Com a rede pronta, precisamos de uma estratégia capaz de medir o poder de viralização entre os usuários dentro deste ecossistema. Baseado na topologia de redes sociais somada às descobertas a partir dos dados coletados de grupos públicos de WhatsApp, nossa metodologia busca modelar como a informação de propaga no ambiente do WhatsApp e simular o efeito de viralização de mensagens tentando emular as limitações de *forward* e *broadcast* que ocorrem dentro desta rede. Para isso, utilizamos o modelo suscetível-infectado (SI) para realização de experimentos que medem a velocidade e alcance desse espalhamento de informação dentro das redes propostas, porém modificamos o modelo às necessidades do WhatsApp, adicionando um novo estágio intermediário, os expostos, propondo assim um modo Susceptível-Exposto-Infectado (SEI) mas adequado às peculiaridades desta rede.

Embora o modelo Suscetível-Infectado (SI) seja um dos modelos epidemiológicos mais simples, ele é bastante robusto em sua aplicação, e já foi utilizado para avaliar a disseminação de informação em redes sociais [Keeling and Eames, 2005]. Para analisar a propagação de informações em grupos do WhatsApp, utilizamos sua adaptação, o Suscetível-Exposto-Infectado (SEI) [Li and Zhen, 2005], considerando a desinformação como uma infecção que se espalha entre os usuários por meio da rede de grupos.

Nesse modelo, cada usuário é representado como um nó em uma rede de grupos, e os nós infectados podem disseminar a infecção para um grupo inteiro, expondo todos os seus participantes. O SEI modela três estágios: *Suscetível* (S), *Exposto* (E) e *Infectado* (I). No estágio *Suscetível*, os usuários ainda não tiveram contato com a infecção. No estágio *Exposto*, os usuários receberam a desinformação por meio de um dos grupos dos quais participam, mas ainda não a compartilharam. No estágio *Infectado*, os usuários que foram expostos à informação compartilham essa mensagem na rede em outros grupos.

O nosso modelo SEI ainda possui três parâmetros principais: *viralidade* (α), *exposição* (β) e um *limite de encaminhamento* (φ). A *viralidade* controla a taxa de usuários infectados, representando a probabilidade de um usuário infectado compartilhar o conteúdo com seus contatos. A *exposição* é a taxa na qual usuários expostos se tornam infectados, ou seja, a probabilidade de um usuário exposto se tornar um usuário infectado. O *limite de encaminhamento* é um parâmetro que restringe a propagação da infecção, simulando as limitações de encaminhamento de mensagens no WhatsApp. Ele determina o número máximo de grupos para os quais um usuário infectado pode enviar o conteúdo.

A simulação do modelo SEI é iniciada selecionando aleatoriamente um usuário como o nó inicial infectado. A cada iteração, os usuários expostos têm uma probabilidade α de compartilhar a mensagem maliciosa. Quando um nó infectado decide encaminhar, ele está sujeito ao limite de encaminhamento φ, o qual define o número máximo de grupos para os quais o conteúdo será enviado. Após o encaminhamento, os usuários nos grupos que receberam a mensagem ficam expostos. Em seguida, cada usuário exposto tem uma probabilidade β de se tornar um nó infectado e compartilhar o conteúdo. Essa iteração continua até que todos os usuários estejam infectados.

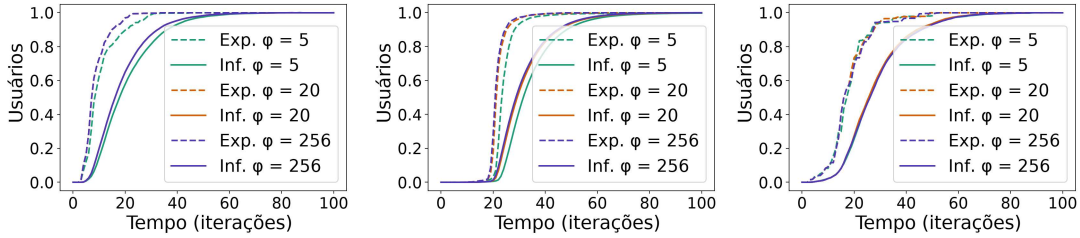
Ao adaptar o modelo SEI para o contexto do WhatsApp, consideramos os estágios de suscetibilidade, exposição e infecção, além de incorporar o limite de encaminhamento como uma restrição à propagação da informação. Essa abordagem permite avaliar a velocidade e o alcance da disseminação de informações nos grupos do WhatsApp, levando em consideração as características da plataforma, como o compartilhamento limitado e as restrições de encaminhamento. Através da manipulação dos parâmetros α, β e φ, podemos analisar os efeitos dessas restrições na viralidade das mensagens e entender como a informação se espalha dentro da rede do WhatsApp.

Desta maneira mais formal, considerando o período de incubação relativamente pequeno e a população constante, ou seja, sem nascimentos, mortes ou migrações, de tamanho N tem-se:

$$N = S(t) + E(t) + I(t) \tag{1}$$

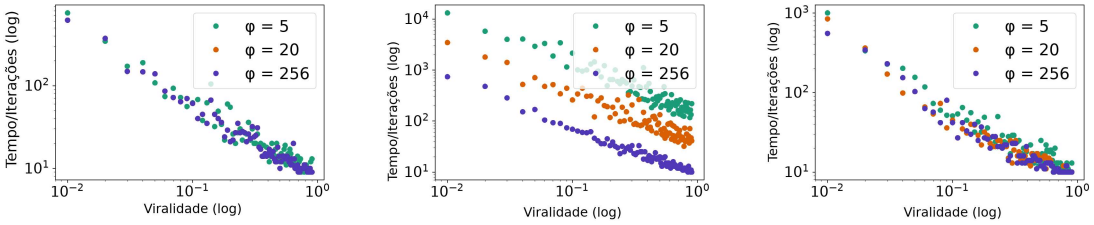
Levando-se em conta que a variação da população infectada é proporcional à população exposta, o sistema de equações diferenciais que descreve a dinâmica desta epidemia no modelo SEI é dado por:

$$\begin{cases} \dfrac{dS}{dt} = -\alpha SE & ,\alpha > 0 \\ \dfrac{dE}{dt} = \alpha SE - \beta E & ,\beta > 0 \\ \dfrac{dI}{dt} = \beta E \end{cases} \tag{2}$$



(a) Brasil (b) India (c) Indonésia

Figura 1.13. Simulações do modelo SEI variando a restrição de encaminhamento entre grupos (φ) com $\alpha = \beta = 0.1$.



(a) Brasil (b) Índia (c) Indonésia

Figura 1.14. Tempo para infectar todos os usuários da rede em simulações do modelo SEI variando a viralidade (α) de 0.001 até 1.0 .

Realizamos vários experimentos usando nosso modelo SEI para comparar a disseminação em diferentes cenários, aplicando restrições de transmissão e encaminhamento. Para essas simulações, consideramos apenas o maior componente conectado, uma vez que não seria possível alcançar nós isolados utilizando toda a estrutura da rede.

A Figura 1.13 mostra a fração de usuários infectados ao longo do tempo para todos os países quando o limite de encaminhamento (φ) é variado, ou seja, como as restrições implementadas pelo WhatsApp podem interferir na propagação. Consideramos o limite de encaminhamento para 5 grupos (cenário real), 20 grupos (limite anterior) e 256 grupos (limite atual para transmissão em massa). Observe que a taxa de usuários expostos na rede cresce muito rapidamente, independentemente dos limites de encaminhamento, sendo suficiente apenas 60 iterações para infectar toda a rede. Além disso, as limitações no encaminhamento diminuem ligeiramente a velocidade de propagação, mas não a interrompem completamente, especialmente para usuários expostos.

Também avaliamos o tempo necessário para que (des)informações com diferentes potencialidades virais infectem todos os usuários. A Figura 1.14 mostra o tempo necessário para infectar 100% dos usuários variando α de $0,001$ até $1,0$, com diferentes limites de encaminhamento. Observe que em situações de disseminação em massa (alto α), é difícil interromper a infecção devido às fortes conexões entre os grupos. No entanto, observe que os limites de encaminhamento e transmissão ajudam a retardar a propagação, principalmente em redes maiores, como na Índia. Em resumo, *os limites de encaminhamento e transmissão podem reduzir a velocidade de disseminação em uma ordem de magnitude para qualquer valor de viralidade α, entretanto não são capazes de bloquear que o conteúdo viralize pela rede*.

1.6. Abordagens para Detecção de Desinformação

Uma forma eficaz de detectar desinformação compartilhada em plataformas digitais é a checagem direta de fatos, normalmente realizada por jornalistas especializados e/ou agência de checagem. Uma tarefa de verificação de fatos (i.e., a avaliação da veracidade de uma notícia, mensagem ou afirmação) [Vlachos and Riedel, 2014] verifica a exatidão das informações comparando-as com uma ou mais fontes confiáveis. Exemplos de tais organizações incluem "Snopes.com"[38], "PolitiFact"[39], "FactCheck. org"[40], e "Aos fatos"[41], " é ou não é (G1)"[42], "Lupa"[43], "Boatos.org"[44] e "Projeto Comprova"[45], no Brasil.

No entanto, apesar da inegável importância neste contexto, a checagem de fatos é um processo demorado, pois geralmente requer uma análise detalhada para apoiar o veredicto [Vlachos and Riedel, 2014]. Consequentemente, a checagem tradicional de fatos não consegue acompanhar o enorme volume de informações que agora são geradas diariamente no ambiente online, que incluem plataformas digitais. Assim, estão emergindo vários estudos focados na verificação computacional de fatos [Atanasova et al., 2019], incluindo detecção automática de desinformação em diferentes cenários e plataformas [Conroy et al., 2015; Volkova et al., 2017]. Diante deste contexto, abordagens automáticas para a detecção de desinformação podem ser úteis para auxiliar, por exemplo, as agências de checagem de fatos na identificação de um conteúdo que necessite ser checado, sugerindo que o veredito final ainda dependa de um especialista. De forma geral, os esforços emergentes neste cenário podem ser divididos em dois grandes grupos: *(i)* estudos propõem soluções baseadas em técnicas de aprendizado de máquina; e *(ii)* esforços que exploram ferramentas ou sistemas online para suporte no monitoramento de (des)informação.

1.6.1. Soluções Baseadas em Aprendizado de Máquina

Essencialmente, notícias contendo desinformação são um viés de distorção nas informações manipuladas pelo editor/produtor do conteúdo [Shu et al., 2017]. Esforços anteriores sobre a teoria do viés de mídia [Gentzkow et al., 2015] mostram que o viés de distorção é geralmente modelado como um problema de classificação binária. Além disso, também há esforços relacionados que exploraram a detecção de desinformação (ou notícias falsas) como uma tarefa binária [Shu et al., 2017; Conroy et al., 2015; Wang, 2017; Volkova et al., 2017; Reis et al., 2019b]. Assim, com base nessas principais razões, no escopo deste capítulo, também definimos a detecção de desinformação como um problema de classificação binária em que a tarefa do classificador é distinguir notícias contendo desinformação das demais (e.g., notícias verdadeiras). Formalmente, o problema pode ser definido na seguinte forma:

[38] www.snopes.com
[39] www.politifact.com/
[40] www.factcheck.org/
[41] aosfatos.org
[42] g1.globo.com/e-ou-nao-e/
[43] piaui.folha.uol.com.br/lupa/
[44] www.boatos.org
[45] projetocomprova.com.br/

Definição 1.6.1 ***(Detecção de Desinformação.)*** *Dada uma notícia/mensagem não rotulada $a \in \mathscr{A}$, um modelo para detecção de desinformação atribui uma pontuação $S(a) \in [0,1]$ indicando até que ponto se acredita que a contenha desinformação. Por exemplo, se $S(a') > S(a)$, de acordo com o modelo é mais provável que a' contenha desinformação em comparação com a. Neste cenário, um limite τ pode ser definido de forma que a função de previsão $F : \mathscr{A} \rightarrow \{$desinformação, não contém desinformação$\}$ seja:*

$$F(a) = \begin{cases} \textit{desinformação} & \textit{if } S(a) > \tau, \\ \textit{não contém desinformação} & \textit{caso contrário.} \end{cases}$$

Assim, há vários esforços que propõem soluções baseadas em técnicas de aprendizado de máquina, como supervisionado [Conroy et al., 2015; Pratiwi et al., 2017; Wang, 2017; Volkova et al., 2017; Reis et al., 2019b], por reforço [Wang et al., 2020], ativo [Bhattacharjee et al., 2017] e profundo [Ruchansky et al., 2017; Zhang et al., 2018; Wang et al., 2018b; Kumar et al., 2020; Chen et al., 2023], e também, com base em estratégias específicas como blockchain [Paul et al., 2019]. Por exemplo, Pérez-Rosas et al. [Pérez-Rosas et al., 2017] conduzem um conjunto de experimentos de aprendizado para construir detectores precisos de notícias falsas usando conjuntos de recursos linguísticos. Da mesma forma, Volkova et al. [Volkova et al., 2017] constroem modelos linguísticos para classificar notícias suspeitas e confiáveis. De forma geral, a maioria desses esforços reduz o problema a uma tarefa de classificação, na qual as notícias são rotuladas como verdadeira/falsa e uma técnica de aprendizado de máquina é então usada para separar o conteúdo falso (ou desinformativo) dos demais com uso de um modelo "aprendido" a partir dos dados de treinamento. Especificamente, esses estudos comumente identificam padrões (ou atributos) recorrentes em notícias contendo desinformação depois que elas já foram disseminadas para propor novos recursos para treinar esses modelos a partir de dados específicos. Os principais atributos explorados na literatura para a proposição dessas abordagens são apresentados na seção a seguir.

1.6.1.1. Atributos para Detecção de Desinformação

A literatura é bastante ampla se considerarmos os esforços relacionados à credibilidade da informação, detecção de boatos, rumores e divulgação de notícias. Assim, conduzimos um levantamento sistemático desses esforços visando identificar os principais atributos propostos e explorados por trabalhos anteriores para detecção de desinformação. A Tabela 1.3 apresenta um resumo deste levantamento junto com algumas das técnicas utilizadas para extração desses atributos. Em alto nível, podemos categorizá-los da seguinte forma: *(i)* atributos extraídos do conteúdo da notícia (e.g., características do texto) [Gupta et al., 2014; Zhao et al., 2015; Wei and Wan, 2017; Volkova et al., 2017]; *(ii)* atributos da fonte da informação (e.g., confiabilidade e credibilidade) [Li et al., 2015]; e por fim *(iii)* atributos extraídas do ambiente, que geralmente envolve medidas de propagação do conteúdo dentro e fora das plataformas digitais [Ciampaglia et al., 2015]. É importante mencionar que, além de serem úteis para a proposição de abordagens tecnológicas nesta temática, esses atributos podem ser explorados para melhor entendimento da desinformação em diferentes contextos.

Tabela 1.3. Visão geral dos atributos para detecção de desinformação explorados em trabalhos anteriores.

Extraído do(a)...	Grupo de Atributos	Técnicas mais utilizadas/exemplos de atributos	Referências
Conteúdo	Atributos Sintáticos	Atributos em nível de sentença, indicadores de qualidade do texto (ex.: métricas de legibilidade), etc	[Conroy et al., 2015; Shu et al., 2017; Rubin et al., 2016; Ratkiewicz et al., 2011; Wei and Wan, 2017; Kwon et al., 2017]
	Atributos Lexicais	Atributos em nível de caracteres e palavras, incluindo número de palavras, pronomes, verbos, indicadores do uso de hashtags, pontuações, etc	[Castillo et al., 2011; Shu et al., 2017; Bhattacharjee et al., 2017; Gupta et al., 2014; Wei and Wan, 2017; Zhao et al., 2015; Kumar et al., 2016; Ribeiro et al., 2017; Ratkiewicz et al., 2011; Ahmed et al., 2017]
	Fundamentos Morais	Atributos ou medidas de fundamentos morais	[Volkova et al., 2017]
	Imagens e Vídeos	Propriedados associadas às imagens e aos vídeos (e.g., distribuições, indicadores de manipulação, etc)	[Jin et al., 2017]
	Atributos Psicolinguísticos	Sinais adicionais de linguagem persuasiva, como raiva, tristeza, etc. e indicadores de linguagem tendenciosa	[Volkova et al., 2017; Rubin et al., 2016; Vosoughi et al., 2018; Gupta et al., 2014; Kwon et al., 2017]
	Estrutura Semântica	*Word embeddings*, modelagem de tópicos (e.g., *Latent Dirichlet allocation (LDA)*), informações contextuais, medição de toxicidade do texto	[Friggeri et al., 2014; Conroy et al., 2015; Bhattacharjee et al., 2017; Rubin et al., 2016; Wei and Wan, 2017; Zhao et al., 2015; Ciampaglia et al., 2015; Wang, 2017]
	Subjetividade	Medidas de subjetividade e análise de sentimentos	[Volkova et al., 2017; Rubin et al., 2016; Ratkiewicz et al., 2011]
Fonte	Editor e Viés	Informações do produtor de conteúdo, indicadores de viés (e.g. político), polarização	[Ribeiro et al., 2017]
	Credibilidade e Confiabilidade	Estimativa da percepção do usuário sobre a credibilidade/confiabilidade da fonte	[Castillo et al., 2011; Shao et al., 2018; Shu et al., 2017]
Ambiente (Plataforma Digital e Web)	Engajamento (Interno e Externo)	Número de compartilhamentos do conteúdo, medidados (dentro e fora da plataforma digital), etc	[Castillo et al., 2011; Shu et al., 2017; Tacchini et al., 2017; Finn et al., 2014; Shao et al., 2016; Vosoughi et al., 2018; Friggeri et al., 2014; Gupta et al., 2014; Kumar et al., 2016]
	Estrutura da Rede	Conexões/redes de amizade, métricas de redes complexas	[Shao et al., 2018; Conroy et al., 2015; Shu et al., 2017; Volkova et al., 2017; Shao et al., 2016; Vosoughi et al., 2018; Castillo et al., 2011; Ratkiewicz et al., 2011; Friggeri et al., 2014; Gupta et al., 2014; Kumar et al., 2016; Tschiatschek et al., 2018]
	Padrões Temporais e Novidade	Séries temporais, medidas de propagação, métricas de novidade	[Castillo et al., 2011; Shao et al., 2018; Shu et al., 2017; Finn et al., 2014; Shao et al., 2016; Vosoughi et al., 2018; Friggeri et al., 2014; Kwon et al., 2017; Tschiatschek et al., 2018]
	Informação dos Usuários	Perfis e características dos usuários em diferentes níveis (e.g., individual, grupos, etc)	[Castillo et al., 2011; Shao et al., 2018; Shu et al., 2017; Tacchini et al., 2017; Ribeiro et al., 2017; Shao et al., 2016; Vosoughi et al., 2018; Ratkiewicz et al., 2011; Gupta et al., 2014; Kwon et al., 2017; Tschiatschek et al., 2018]

1.6.2. Sistemas para Monitoramento da (Des)Informação

Finalmente, surgiram vários sistemas com objetivo de monitorar o conteúdo disseminado em plataformas digitais, como contramedidas ao problema da desinformação nesses ambientes. Exemplos de tais sistemas incluem *(i)* "Hoaxy" [Shao et al., 2016], uma plataforma da Web para o rastreamento de notícias compartilhadas contendo desinformação; *(ii)* "Fake Tweet Buster" [Saez-Trumper, 2014], uma ferramenta da Web para identificar usuários que promovem desinformação no Twitter; e *(iii)* ELEIÇÕES SEM FAKE[46] no Brasil, nosso projeto, detalhado na seção a seguir, desenvolvido para trazer transparência na divulgação de conteúdo durante as eleições brasileiras em 2018, como um esforço para mitigar e evitar a disseminação de desinformação bem como prover transparência no espaço midiático.

1.7. Relato de Experiência: Projeto Eleições Sem Fake

Nesta seção apresentamos um relato de experiência sobre o projeto ELEIÇÕES SEM FAKE, proposto e coordenado pelo professor Fabrício Benevenuto do Departamento de Ciência da Computação (DCC) da Universidade Federal de Minas Gerais (UFMG), em colabora-

[46] www.eleicoesemfake.dcc.ufmg.br

ção com estudantes e professores de diversas instituições brasileiras, incluindo a Universidade Federal de Viçosa (UFV), a Universidade Federal de Mato Grosso do Sul (UFMS), dentre outras. Este projeto surgiu em 2018 como uma iniciativa para mitigar o problema da desinformação em plataformas digitais durante o período eleitoral daquele ano. As propriedades de rápida difusão e ampla propagação de informações nesses ambientes podem ser abusadas para fins de propaganda não solicitada, interrupção de comunicação legítima ou mesmo para manipulação de opinião. Assim, o objetivo do projeto foi trazer transparência para as campanhas políticas realizadas em plataformas digitais online, explorando soluções tecnológicas capazes de expor dados de campanhas de desinformação dentro deste contexto. Através da transparência, jornalistas, checadores de fatos, legisladores e consultores jurídicos puderam (e podem) atuar e permitir ações do poder público, caso sejam constatadas irregularidades.

Por exemplo, durante as eleições brasileiras de 2018, foram desenvolvidos sistemas que coletam, processam e analisam dados em larga escala de plataformas sociais tradicionais, como Twitter e Facebook, e especialmente do WhatsApp através do Monitor de WhatsApp [Melo et al., 2019a]. Este sistema foi extensivamente utilizado por jornalistas e equipes de checagens de fatos durante o processo eleitoral supracitado, facilitando o acesso aos dados da plataforma e permitindo uma navegação por data pelos conteúdos mais populares compartilhados por dia. Com isso, os usuários, conseguem apontar padrões e movimentos que emergem dentro do WhatsApp de uma forma que seria impossível sem o sistema, fornecendo um valioso suporte para a tarefa de checagem de fatos e reduzindo o esforço necessário para encontrar as notícias e desinformações que vão sendo viralizadas na rede.

Acreditamos que esse relato de experiência sobre o projeto ELEIÇÕES SEM FAKE seja extremamente relevante para quem deseja entender e combater a desinformação em plataformas digitais, principalmente durante períodos eleitorais. O relato mostra como a tecnologia pode ser usada para identificar e expor campanhas de desinformação, explorando de forma inteligente um grande volume de dados que é diariamente disseminado nesses ambientes. Além disso, o projeto se mostrou eficaz ao fornecer um suporte valioso para jornalistas e equipes de checagem de fatos, reduzindo o esforço necessário para encontrar notícias e desinformações viralizadas nestes ambientes e também no desenvolvimento de pesquisas acadêmicas que recorrem aos dados disponibilizados por meio dos diversos sistemas desenvolvidos. Logo, descrever experiências relacionadas ao projeto em questão pode ser útil para pessoas interessadas em desenvolver soluções tecnológicas para combater a desinformação, para jornalistas e profissionais da comunicação que desejam aprimorar sua cobertura de notícias em períodos eleitorais, bem como para pesquisadores de diferentes áreas do conhecimento interessados no estudo do fenômeno. Especificamente nas seções a seguir nos atemos a apresentar em detalhes dois dos principais sistemas que compõem o projeto, que chamamos: *(i)* MONITOR DE WHATSAPP (ou "*WhatsApp Monitor*") e o *(ii)* "Monitor de Anúncios do Facebook", referenciado como ADCOLLECTOR.

1.7.1. Monitor de WhatsApp

Infelizmente, as plataformas plataformas se tornaram ambientes propícios para a propagação de campanhas de desinformação, especialmente no contexto político. O WhatsApp,

por exemplo, com seus mais de 2 bilhões de usuários, desempenha um papel central nesse cenário, tendo sido palco de eventos de desinformação em diferentes partes do mundo. Durante as eleições presidenciais de 2018 no Brasil, por exemplo, houve uma disseminação massiva de desinformação por meio da plataforma, despertando grande preocupação sobre como o WhatsApp opera [Tardaguila et al., 2018].

Um dos principais desafios para combater a disseminação de desinformação no WhatsApp é a dificuldade em analisar o conteúdo compartilhado dentro da plataforma. Embora uma grande parte das conversas ocorram em grupos públicos com até 256 membros, o acesso restrito e a criptografia ponta-a-ponta dificultam a exploração desse conteúdo viral. Entretanto, ao contrário de outros sistemas mais tradicionais, pesquisadores e jornalistas não possuem uma ferramenta abrangente para analisar as mensagens mais populares que circulam nessa plataforma, o que torna o combate à desinformação neste cenário um desafio ainda maior. Nesse contexto, o monitor de WhatsApp, que depois foi replicado também para dados do Telegram Júnior et al. [2022], permite aos usuários explorar o conteúdo mais compartilhado em grupos públicos de aplicativos de mensageria. Com a implementação destas ferramentas, oferecemos uma forma eficaz de combater a disseminação de desinformação, auxiliando pesquisadores, jornalistas e demais interessados a analisar o conteúdo que viraliza nessa plataforma de mensagens instantâneas.

De forma resumida, nossa ferramenta monitora várias categorias de mensagens, como imagens, vídeos, áudio e texto que foram postadas em um conjunto de centenas de grupos públicos políticos do WhatsApp e exibe os conteúdos mais compartilhados por dia numa interface online. Ela já foi usada para monitorar conteúdo durante as eleições gerais brasileiras de 2018, eleições de 2019 na Índia e na Indonésia, para acompanhar notícias e outras (des)informações sobre a pandemia COVID-19 durante 2020 e 2021 e também acompanhar o conteúdo compartilhado no WhatsApp durante as eleições presidenciais de 2022. Este sistema é um dos principais esforços atualmente para estimar a disseminação de desinformação no WhatsApp e ajudar nos esforços de verificação de fatos dentro deste cenário [Melo et al., 2019a]. A arquitetura do sistema é responsável por agrupar, coletar, processar e ranquear o conteúdo explorado do WhatsApp e condensa-lo numa interface online, acessível em um navegador de Internet através de login e senha para ser possível o usuário final navegar entre os dados coletados e fazer suas próprias análises com a ajuda do sistema. O principal objetivo do MONITOR DE WHATSAPP é propiciar um sistema capaz de informar e antecipar aos comunicadores sobre o tipo de informação compartilhada no WhatsApp. A seguir, apresentamos maiores detalhes relativos ao sistema. Em seguida, discutimos os impactos da utilização da ferramenta criada e, por último, relacionamos nossas considerações finais acerca do MONITOR DE WHATSAPP.

1.7.1.1. Metodologia

O nosso sistema Web MONITOR DE WHATSAPP foi idealizado originalmente em 2018 numa versão preliminar publicada por Resende et al. [Resende et al., 2018]. Uma versão funcional da ferramenta foi desenvolvida e disponibilizada no ano seguinte [Melo et al., 2019a]. Depois, a proposta foi também replicada no Telegram [Júnior et al., 2022]. Atualmente, o MONITOR DE WHATSAPP conta com uma versão online com atualizações diárias de conteúdo com todas as funcionalidades de *ranking* e agrupamento aprimora-

das. Sua estratégia de coleta segue os mesmos passos apresentados anteriormente neste capítulo, na seção de coleta de dados de aplicativos de mensagens instantâneas. Portanto, explicaremos de forma bastante reduzida esta etapa de como ele funciona novamente aqui, enquanto focaremos mais em outros aspectos como a interface implementada e o impacto de sua utilização.

O MONITOR DE WHATSAPP usa de dados coletados de grupos públicos do WhatsApp selecionados que discutem tópicos políticos. Esses grupos públicos são operados tanto por indivíduos afiliados a partidos políticos, líderes comunitários locais ou usuários comuns com interesse no tema, e podem ser acessados livremente por qualquer pessoa com um link de convite: uma URL do WhatsApp no padrão `chat.whatsapp.com/<groupID>` compartilhada em plataformas sociais para qualquer pessoa que deseje entrar no grupo.

A primeira etapa do trabalho é, portanto, selecionar os grupos de interesse que serão monitorados pelo sistema. Enquanto os dados de grupos da Índia e Indonésia foram adquiridas em parceria nossa com pesquisadores do MIT [Garimella et al., 2018], para o Brasil seguimos a coleta de grupos públicos de política no WhatsApp. Para o levantamento dos grupos públicos, utilizamos uma lista de palavras-chaves, junto com a URL de convite do WhatsApp, numa pesquisa em plataformas sociais (e.g., Twitter e Facebook) e em ferramentas de busca (e.g., Google) para encontrar grupos relevantes para o monitoramento. Após listar os resultados, configuramos três celulares com contas válidas do WhatsApp para participar de cada grupo. Após entrar, expandimos nossa coleção com outros grupos compartilhados dentro dos chats de conversa do próprio WhatsApp, resultando em uma coleção de mais de mil grupos selecionados.

Posteriormente, com toda configuração preparada, iniciamos a extração de dados, coletando e processando todos as mensagens dos *chats* no servidor conforme a arquitetura de coleta proposta neste capítulo. O processo de coleta é feito diariamente, alimentando o banco de dados com conteúdo por dia com todos os textos e mídias relacionadas. Cada conta utilizada é membro propriamente dito de um conjunto específico de grupos que recebe, tal como qualquer outro usuário de WhatsApp, todas suas mensagens em seu celular. Para coleta propriamente dita, não existe uma API oficial do WhatsApp para este tipo de aplicação de coleta de mensagens. Desta forma, utilizamos de uma estratégia de raspagem de dados através do WhatsApp Web com auxílio de uma biblioteca WebWhatsAPI[47]. Essa ferramenta utiliza a versão de navegador do WhatsApp para fazer um *parsing* da página Web e coletar de um usuário os grupos e todo o conteúdo recebido em cada um. Esse conteúdo é processado e organizado no banco de dados de forma que, quando um usuário do MONITOR DE WHATSAPP navega no *website*, o sistema acessa esse banco e exibe um *dashboard* com as mídias e textos referentes ao período visualizado.

Uma das etapas mais importantes na elaboração e criação de um monitoramento de dados de plataformas de mensagens instantâneas é o agrupamento de conteúdo semelhante, o que permite rastrear e contar quantas vezes um determinado conteúdo, como imagens, vídeos ou áudios, foi compartilhado. Diferentemente de outras plataformas, onde as informações sobre curtidas e compartilhamentos já estão disponíveis nos metadados, no WhatsApp cada mensagem é postada de forma isolada. Portanto, é necessário

[47] `https://github.com/mukulhase/WebWhatsapp-Wrapper`

Figura 1.15. *Screenshot* da interface principal do MONITOR DE WHATSAPP com exibição do conteúdo mais popular de um determinado dia.

realizar o rastreamento, agrupamento e contagem dos dados para determinar a popularidade de cada conteúdo compartilhado.

Para rastrear os compartilhamentos de um único conteúdo e contar sua popularidade, utilizamos o *PerceptualHash* (*pHash*) para gerar uma impressão digital única para cada imagem [Du et al., 2020]. Para áudios e vídeos, calculamos um *hash* a partir do *checksum* (MD5) de cada arquivo. Com esses *hashes*, podemos agrupar as postagens que contêm o mesmo conteúdo e calcular informações sobre ele, como sua popularidade, quantos grupos o compartilharam e quantos usuários diferentes o enviaram. Além disso, usamos o índice Jaccard para comparar mensagens de texto e agrupá-las. Essa contabilização é realizada diariamente, e os usuários podem acessar o *ranking* de popularidade por meio da interface Web do MONITOR DE WHATSAPP.

Questões Éticas. O MONITOR DE WHATSAPP reúne uma quantidade considerável de dados de muitos grupos e usuários do WhatsApp. Para garantir a privacidade dos usuários, não compartilhamos ou divulgamos quaisquer informações de identificação pessoal, como números de telefone celular ou nome do usuário. Outro material sensível que pode estar presente em nosso conjunto de dados são imagens que retratam violência explícita ou conteúdo adulto. Como pode ser prejudicial para os usuários que navegam em nosso sistema, utilizamos um filtro de conteúdo adulto para detectar esse tipo de conteúdo e rotulá-lo no sistema. Para evitar o uso indevido, mesmo de informações agregadas, também limitamos o acesso do nosso sistema a um número restrito de jornalistas e pesquisadores, por meio de uma conta de login e senha. Além disso, eles também são informados sobre as limitações dos dados e o potencial viés presente em nosso sistema. Como usamos apenas grupos do WhatsApp disponíveis publicamente, assim como todos os demais membros, nossa coleta de dados não viola os termos de serviço do WhatsApp.

1.7.1.2. Interface e Uso do sistema

Nós fornecemos um sistema online no qual os usuários podem supervisionar diariamente as tendências compartilhadas nos grupos públicos do WhatsApp. Nosso sistema exibe informações sobre as mídias de texto (apenas os com mais de 140 caracteres), áudio,

Figura 1.16. Painel do MONITOR DE WHATSAPP em que os usuários podem selecionar a data ou escolher um período.

imagem e vídeo dos grupos relacionadas a tópicos políticos e de notícias monitorados diariamente. Em seguida, ranqueamos cada tipo de mídia entre as mais populares. A Figura 1.15 mostra uma captura de tela de como o conteúdo é exibido na interface do MONITOR DE WHATSAPP assim que o usuário faz login no sistema, mostrando, em ordem, os conteúdos com mais compartilhamentos. Vale mencionar que o usuário pode escolher também entre português ou inglês como idioma da interface no canto superior esquerdo da tela.

Atualmente, nosso sistema funciona para três instâncias distintas: uma indiana, uma versão indonésia e uma brasileira[48]. Depois que uma instância é escolhida e o usuário faz login, ele é levado a um painel de controle onde pode navegar entre as datas e observar o conteúdo mais compartilhado, conforme mostrado na Figura 1.16. O sistema também permite que os usuários selecionem o período que desejam, como, por exemplo, um dia, uma semana ou mês inteiro de dados. Depois de escolher uma data de início e término para a pesquisa, o sistema recupera e relata o conteúdo mais popular para toda a data selecionada. Isso permite a jornalistas e pesquisadores investigarem um período específico ou mesmo eventos que duram mais de um dia, combinando milhares de mensagens em uma interface resumida e ranqueada, na qual podem surgir alguns padrões de publicação e conteúdo que, sem o sistema, seria difícil perceber. Isso permite que os jornalistas tenham uma melhor ideia sobre o conteúdo crítico compartilhado no WhatsApp que pode valer a pena ser verificado.

No MONITOR DE WHATSAPP, quatro tipos de conteúdo são identificados e consolidados no banco: imagens, vídeos, mensagens de áudio e mensagens de texto (apenas as com mais de 140 caracteres). Nosso sistema exibe diariamente o conteúdo dividido por cada tipo de mídia e os mostra cada uma ranqueada por total de compartilhamentos. Isso permite que os jornalistas tenham diariamente uma ideia sobre o conteúdo crítico compartilhado no WhatsApp que pode valer a pena ser verificado.

Para dar mais detalhes sobre cada conteúdo compartilhado no WhatsApp, ao clicar em um objeto no painel, disponibilizamos informações compiladas de compartilhamentos entre os grupos monitorados para cada conteúdo selecionado. Um usuário tem à sua dis-

[48]A versão indiana conta apenas com dados estáticos de 2019 durante as eleições gerais indianas, enquanto o Brasil e a Indonésia continuam sendo atualizadas diariamente.

posição o número total de compartilhamento, em quantos grupos esse conteúdo apareceu e quantos usuários únicos postaram sobre aquele conteúdo. Além disso, existe uma funcionalidade de busca com botão "Na Web" para outras fontes. Clicando em uma imagem, é possível verificar o conteúdo com botão, que usa da busca reversa de imagens do Google para rastrear fontes externas onde aquela imagem foi compartilhada. Curiosamente, o próprio aplicativo WhatsApp implementou uma funcionalidade muito semelhante, chamada "Pesquisar na Web" em 2020, em que usuários podem pesquisar por conteúdo viral na Web[49].

1.7.1.3. Discussões e Impacto Científico-Social do MONITOR DE WHATSAPP

Desde as eleições brasileiras de 2018 até julho de 2021, demos acesso ao sistema a mais de 300 usuários, incluindo jornalistas, pesquisadores e agências de checagem de fatos que mencionaram explicitamente nosso sistema como fonte de dados durante as checagens. Adicionalmente, dezenas de notícias fizeram referência ao nosso sistema ou usaram nossos dados durante as eleições brasileiras e durante a pandemia de COVID-19 para entender melhor as discussões que ocorrem dentro do WhatsApp. Mais especificamente, matérias da BBC, The Guardian, El Pais, The Intercept, O Globo, Estadão, Folha, Uol, entre diversas outras que utilizam do sistema para investigar o WhatsApp e produzir a reportagem, conforme mais detalhado em [Melo et al., 2021].

Ademais, o MONITOR DE WHATSAPP foi capaz de agrupar conteúdo a partir de um grande volume de dados e ranqueá-lo diariamente até o uso de muitos jornalistas e agências. Notavelmente, nosso sistema é referenciado como parceiro do Comprova[50], um projeto jornalístico da *First Draft* com foco na verificação de conteúdo publicado na Web durante a eleição presidencial brasileira de 2018 e pela agência de checagem de fatos Lupa[51] de jornalistas do grupo Folha. Em 2018, a UFMG foi parceira do TSE através do nosso sistema para conter a desinformação nas eleições[52] e, em 2020, tivemos uma parceria com o Ministério Público de Minas Gerais (MPMG), como um dos projetos do Programa de Capacidades Analíticas, que visa prover mais transparências sobre dados públicos online[53].

Além dos impactos e colaborações mencionados, diversos estudos, feitos fora do nosso grupo de pesquisa da UFMG, também utilizaram o MONITOR DE WHATSAPP como metodologia e fonte de dados para avançarem as pesquisas nas áreas do WhatsApp e também em torno da temática de desinformação no país. O sistema auxilia esses pesquisadores a desenvolverem seus trabalhos, provendo transparência e facilidade para navegar em períodos passados do WhatsApp, como por exemplos os trabalhos de [Recuero et al., 2021] e [da Silva, 2021], entre diversos outros que tiveram o sistema como fonte de dados para sua análise. Müzell [Müzell, 2020], por exemplo, em sua dissertação de mestrado

[49] https://blog.whatsapp.com/search-the-web/?lang=en

[50] https://projetocomprova.com.br/partner/monitor-de-whatsapp-ufmg/

[51] https://piaui.folha.uol.com.br/lupa/tag/ufmg/

[52] https://www.tse.jus.br/imprensa/noticias-tse/2018/Outubro/tse-estuda-possibilidade-de-firmar-parceria-com-universidade-para-inibir-fake-news-no-whatsapp

[53] https://www.mpmg.mp.br/comunicacao/noticias/mpmg-inicia-trabalhos-de-convenio-com-ufmg-para-ampliar-capacidade-de-analise-de-dados.htm

se baseou nos dados do sistema durante as eleições de 2018, identificando estratégias e padrões sobre como a campanha política ocorreu no WhatsApp e como elas interferiram na eleição. Almeida et al. [Almeida et al., 2019], para saber quais conteúdos foram veiculados nas eleições de 2018, também utilizou o MONITOR DE WHATSAPP para explorar características próprias da circulação dessas plataformas, considerando o espaço privado de troca de informações. Soares et al. [Soares et al., 2021], com auxílio de nossa ferramenta, também estudaram a desinformação sobre COVID-19 no WhatsApp, observando como a pandemia enquadrada como debate político. Em outra direção, o estudo de Tomás et al. [Tomás et al., 2020] investigou as notícias falsas contra as universidades públicas no Brasil usando do MONITOR DE WHATSAPP.

O MONITOR DE WHATSAPP está online desde 2018, sendo atualizado diariamente com dados de mais de 900 grupos públicos sobre política na referida plataforma. Nosso sistema é utilizado como fonte de dados por vários pesquisadores e jornalistas, inclusive como fonte de três agências de checagem de fatos. A arquitetura do sistema coleta, processa, ranqueia e exibe todo conteúdo dos grupos de WhatsApp num sistema Web, hospedado no servidor do nosso grupo de pesquisa da UFMG e é acessível por navegador apenas através de usuário e senha. Nossa metodologia não só apresenta um modo inovador de coletar dados do WhatsApp [Resende et al., 2019], como também se mostrou eficaz na ajuda ao combate de desinformação. Com a transparência de acesso aos dados do WhatsApp e facilidade de uso com uma navegação por data pelos conteúdos mais populares compartilhados por dia, os usuários conseguem apontar padrões e movimentos que emergem dentro da plataforma de uma forma que seria impossível sem o sistema. Com isso, damos um valioso suporte para a tarefa de checagem de fatos, reduzindo o esforço necessário para encontrar as notícias e desinformações que vão sendo viralizadas na plataforma. Devido à natureza fechada e efêmera do WhatsApp, nosso sistema funciona como uma espécie de registro histórico dos eventos ocorridos na plataforma, uma vez que esses dados dificilmente seriam acessíveis de outra forma, dado que a empresa não armazena o conteúdo por muito tempo e mesmo usuários podem não ter mais registro das mensagens enviadas.

O sistema desenvolvido aqui pode ainda ser expandido futuramente para dar ainda mais suporte a cobertura dos eventos no WhatsApp durante próximos períodos eleitorais. Existe espaço para melhoras tanto na infraestrutura utilizada para coleta, com mais espaço no servidor da universidade e mais celulares para gerenciar contas de WhatsApp, como também o número de grupos monitorados. Pretendemos também integrar novas funcionalidades de ordenação e busca no sistema. Para fornecer novas formas de ranqueamento, estudaremos a implementação de métodos de aprendizado para detecção automática de desinformação, agregando uma pontuação (ou *score*) que indique a probabilidade do conteúdo ser conter desinformação. Esperamos que tal abordagem possa auxiliar ainda mais jornalistas encontrar conteúdos mais relevantes a serem checados. Por fim, também planejamos implementar a geração de relatórios periódicos referentes a datas específicas para facilitar a interpretação dos dados no sistema e fornecer informações mesmo para aqueles sem um cadastro no sistema.

1.7.2. Monitor de Anúncios do Facebook

Desde as eleições presidenciais dos Estados Unidos em 2016, o conteúdo patrocinado contendo Propaganda Eleitoral se tornou uma forma eficaz de campanha política. No entanto, essa eleição foi marcada pelo abuso da publicidade direcionada em Redes Sociais Online. Preocupados com o risco do mesmo tipo de abuso ocorrer nas eleições brasileiras de 2018, nós projetamos e implementamos uma abordagem computacional para monitorar anúncios políticos em plataformas de redes sociais.

Primeiramente, devido ao abuso da publicidade direcionada no Facebook (atual Meta) durante as eleições presidenciais dos Estados Unidos em 2016, escolhemos a plataforma do Facebook para validar nossa abordagem. Para isso, nós adaptamos inicialmente um *plug-in* do navegador para coletar anúncios da linha do tempo de voluntários que usam o Facebook. Nós conseguimos a adesão de mais de 2000 voluntários a ajudar em nosso projeto e instalar nossa ferramenta. Em seguida, utilizamos uma Rede Neural Convolucional (CNN) para detectar anúncios políticos no Facebook usando incorporação de palavras. Para avaliar nossa abordagem, rotulamos manualmente uma coleção de dados com 10 mil anúncios como políticos ou não políticos e, em seguida, realizamos uma avaliação detalhada da abordagem proposta para identificar anúncios políticos, comparando-a com métodos clássicos de aprendizado de máquina supervisionado.

1.7.2.1. Metodologia

Inicialmente, nosso principal objetivo era coletar anúncios diretamente da linha do tempo do usuário. Portanto, precisamos construir uma extensão de navegador cujo objetivo é coletar anúncios enquanto o usuário está navegando em seu perfil do Facebook. Além disso, nós coletamos anúncios da Biblioteca de Anúncios Políticos da Biblioteca de Anúncios do Facebook. Apesar das características históricas desses anúncios, eles contêm anúncios reais feitos por anunciantes políticos. Assim, a extensão do navegador e o coletor da Biblioteca de Anúncios pertencem ao módulo principal da arquitetura proposta chamado `AdCollector`.

A extensão de navegador consegue extrair as informações diretamente do HTML do anúncio. Nós extraímos informações como o nome do anunciante, identificador do anunciante, legendas do anúncio, imagens do anúncio, URLs e principalmente as informações fornecidas pelo Facebook sobre “Por que estou vendo isso?”. Ver Figura 1.17.

Figura 1.17. Exemplo de um “Por que estou vendo isso?”

Cada anúncio no Facebook inclui um botão e, quando os usuários clicam nele, eles recebem uma explicação sobre por que estão vendo aquele anúncio em particular. As explicações de anúncios geralmente têm duas partes: a primeira parte apresenta uma razão pela qual um usuário recebeu um anúncio. A redação geralmente indica o tipo de atributo que está sendo apresentado. A segunda parte lista os possíveis atributos que podem ter sido usados pelo anunciante para segmentar um anúncio. Geralmente inclui um conjunto limitado de informações, como idade, gênero e localização, usadas como opções de segmentação.

Página de Preferências de Anúncios. A página de preferências de anúncios é uma página personalizada que fornece aos usuários informações sobre vários atributos que influenciam os anúncios que eles veem, além de oferecer a eles um certo nível de controle sobre esses anúncios. Primeiro, os usuários podem ver os interesses que o Facebook inferiu sobre eles, como se eles estão interessados em coisas como "Jogos de Vídeo", "Pizza" ou até mesmo "Homossexualidade". Os interesses também são acompanhados por uma descrição de como eles foram inferidos. Essas descrições podem ser afirmações como "Você tem essa preferência porque clicou em um anúncio relacionado a *Interesse*" ou "Você tem essa preferência porque curtiu uma página relacionada a *Interesse*". Como mostrado em [Andreou et al., 2018], a grande maioria dessas explicações é vaga e não fornece muitos detalhes.

Nós utilizamos o ADCOLLECTOR para monitorar anúncios de 14 de março de 2018 a 28 de outubro de 2018. Esse período abrange o período eleitoral das elições, incluindo os dois turnos. No geral, mais de 2.000 usuários se voluntariaram para instalar nossa extensão de navegador e compartilhar os anúncios que receberam enquanto navegavam no Facebook com nosso plugin. Observamos, no entanto, que muitos usuários apenas instalaram a extensão do navegador, mas não a utilizaram. No entanto, um total de 715 usuários utilizaram ativamente nossa ferramenta nesse período.

A Figura 1.18 mostra o número de usuários ativos por dia. Nós consideramos um usuário ativo em um dia se ele recebeu pelo menos um anúncio do Facebook. O número de usuários diários aumentou rapidamente quando vários veículos de mídia publicaram artigos sobre nosso sistema e permaneceu relativamente estável posteriormente. A diminuição repentina de usuários ativos em meados de junho pode ser atribuída a uma mudança que o Facebook fez na forma como rotula os anúncios, resultando na perda de anúncios por alguns dias. Também observamos que a atividade dos usuários nos fins de semana diminui, o que pode indicar que alguns usuários instalaram nosso *plugin* em seus computadores no trabalho.

Dos 715 usuários, 682 são do Brasil. Nós inferimos essa informação de segmentação analisando os dados das explicações do "*Por que estou vendo isso?*" que foram coletadas pela nossa extensão. Coletamos um total de **239k anúncios únicos** enviados por **40k anunciantes**. Cada anúncio é identificado por um *id* único fornecido pelo Facebook. Dos 239k anúncios únicos, 166k foram enviados durante o período pré-eleitoral (março de 2018 a 15 de agosto de 2018) e 74k foram enviados durante o período eleitoral (16 de agosto de 2018 a 28 de outubro de 2018). Para cada anúncio, temos informações sobre o anunciante, o texto do anúncio, o texto no aviso político do anúncio, a imagem (quando disponível) e a URL de destino. Nos referimos a esse conjunto de dados como o

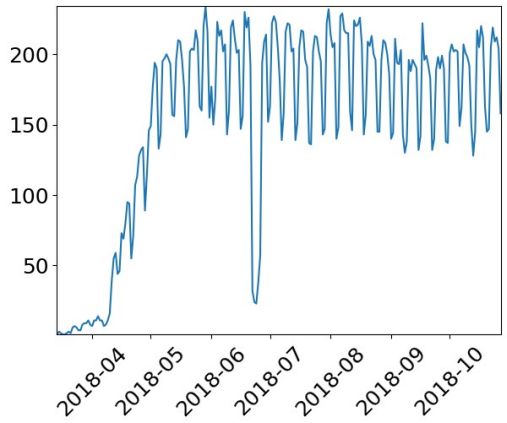


Figura 1.18. Número de usuários ativos diariamente.

ADCOLLECTORDATA.

Os anúncios políticos oficiais são diferentes dos anúncios regulares. Além da *tag* "Patrocinado", o aviso também contém a *tag* "Propaganda Eleitoral". Por causa disso, nossa extensão não coletou esses anúncios, ou seja, o conjunto de dados coletado com nossa extensão de navegador não contém nenhuma propaganda eleitoral que foi explicitamente informada ao Facebook pelos anunciantes.

Nós utilizamos o modelo CNN treinado com dados rotulados por três rotuladores independentes e atribuímos a cada anúncio uma probabilidade de ser político. Na prática, há um número significativamente maior de anúncios não políticos do que anúncios políticos (ou seja, temos um cenário de conjunto de dados desbalanceado). Para limitar o número de falsos positivos (ou seja, anúncios classificados erroneamente como políticos pelo nosso modelo, mas que na verdade são não políticos), escolhemos um limiar para declarar um anúncio como político que corresponde a uma taxa de 1% de falsos positivos (em vez de escolher o limiar típico de 0,5 para a probabilidade de ser político, que corresponde a uma taxa de falsos positivos de 8% e taxa de verdadeiros positivos de 96%). Utilizando um limiar de 0,97, nosso modelo CNN classifica 1.133 anúncios como políticos dos 38.110 anúncios testados - $2,9\%$ dos anúncios são políticos. Os 1.133 anúncios foram postados por 739 anunciantes.

Na Figura 1.19 é apresentado um anúncio identificado dentro da nossa base de dados do ADCOLLECTOR utilizando o nosso classificador. Neste anúncio, um candidato elogia o outro ressaltando suas qualidades, e no criativo do anúncio é apresentado o número do candidato. É importante salientar que as informações do CNPJ foram colocados no corpo do anúncio, porém não havia o rótulo oficial da Plataforma Meta indicando que era uma propaganda política. Desta forma, ao término desta campanha este anúncio não ficará disponível na biblioteca de anúncio da Plataforma Meta.

1.8. Desafios e Oportunidades de Pesquisa

Combater a desinformação é uma típica luta adversária. A cada eleição, por exemplo, as campanhas de desinformação exploram novas formas de manipular a opinião e novos mecanismos de defesa são criados visando ao menos mitigar as campanhas de desinformação. Aqui, apresentamos alguns desafios e oportunidades de pesquisa nesta temática.

Figura 1.19. Exemplo de anúncio de propaganda detectado pelo nosso classificador.

Compreensão do Ecossistema de (Des)Informação. As plataformas digitais estão passando por mudanças constantes que impactam diretamente a forma como as informações são disseminadas dentro desses ambientes. O WhatsApp, por exemplo, recentemente, implementou o recurso de comunidades, que potencializa o disparo massivo de mensagens dentro da plataforma. Outro desafio é identificar padrões de desinformação que permanecem inalterados em diferentes contextos e ambientes de disseminação, por exemplo, ao longo do tempo. Logo, é fundamental que sejam investigados os impactos ocasionados por essas mudanças no ecossistema de (des)informação considerando as diferentes plataformas e contextos (e.g., saúde, política, etc) a fim de que seja possível compreendê-los para criação de mecanismos de combate e/ou mitigação de seus efeitos, quando aplicável. Além disso, em vários casos, as plataformas digitais são utilizadas apenas como veículos (ou vitrine) para disseminação de conteúdos que na verdade são produzidos por *websites* externos dedicados exclusivamente à produção e disseminação de desinformação [Couto et al., 2022a,b]. Nesta direção, acreditamos que ainda há espaço de pesquisa para uma compreensão mais aprofundada desse ecossistema, potencialmente orquestrado e financiado, bem como para a proposição de abordagens efetivas neste contexto.

Propagandas Políticas. Embora as plataformas tenham criado mecanismos de transparência no contexto de propagandas políticas eles ainda são bastante limitados. Recentemente, um grupo de pesquisadores conseguiram impulsionar conteúdos 100% falsos nas eleições de 2022 na Plataforma de Anúncios da Meta, onde o anúncio afirmava que a data da eleição havia mudado[54]. Além disso, o anúncio foi pago com moeda estrangeira e criado por um usuário que estava em Londres. Portanto, ainda existe espaço para pesquisas e soluções tecnológicas com objetivo de mitigar o uso de plataformas de impulsionamentos dentro das diferentes plataformas digitais com a intenção de prejudicar ou tumultuar o processo eleitoral. Acreditamos que a regulação seja importante, no entanto isso não elimina a necessidade de proposição de soluções tecnológicas como mecanismos de apoio em diferentes contextos.

[54] https://www.globalwitness.org/en/campaigns/digital-threats/facebook-fails-tackle-election-disinformation-ads-ahead-tense-brazilian-election/

Detecção Automática de Desinformação. A desinformação não é mais disseminada exclusivamente em formato texto no ambiente online. Atualmente, a desinformação é propagada nas plataformas digitais por meio de imagens, vídeos (e.g., *deep fake*, memes, stickers, etc. Logo, é preciso que as abordagens automáticas sejam adequadas e/ou desenvolvidas para a detecção de desinformação em diferentes tipos de mídias. Além disso, acreditamos que ainda há espaço para investigação de estratégias mais sofisticadas, envolvendo aprendizado por transferência, federado, etc, ou ainda técnicas estado-da-arte de representação de conteúdo (e.g., *embeddings*) que ainda são pouco explorados neste contexto.

Explicabilidade de Abordagens Tecnológicas. Existem alguns estudos que visam investigar a explicabilidade de resultados iniciais promissores da detecção computacional de desinformação em plataformas digitais, ou seja, por que uma determinada notícia é classificada como desinformação (ou não) [Shu et al., 2019; Yang et al., 2019; Cui et al., 2019; Reis et al., 2019a; Lu and Li, 2020]. No entanto, este é um campo que ainda carece de esforços, considerando principalmente a multidisciplinaridade envolta na temática.

Investigação de Plataformas Emergentes. Por fim, existem várias ferramentas emergentes baseadas em Inteligência Artificial (IA) que, devido ao acesso massivo e a facilidade no uso sem critérios bem definidos apresentam potencial para serem exploradas como recurso para a propagação de desinformação. Como exemplos podemos citar a ChatGPT[55], e ferramentas *Text-to-Image*, como a DALL-E[56], Midjourney[57], e o *Stable Diffusion*[58] que recentemente foram exploradas para criação de informação inverídica que circulou em plataformas digitais[59]. Logo, neste contexto é fundamental entendermos como essas ferramentas emergentes estão/podem ser utilizadas para a disseminação da desinformação do ambiente online provendo uma avaliação diagnóstica que compreenda uma investigação, inclusive, de questões éticas relacionadas.

1.9. Considerações Finais

O presente capítulo apresentou discussões relacionadas ao uso das plataformas digitais para a disseminação de campanhas de desinformação em diferentes contextos. Esperamos que os leitores obtenham uma compreensão aprofundada da desinformação em plataformas digitais e de como combatê-la de forma eficaz, contribuindo assim para um ambiente digital mais saudável e confiável.

Material. Todos os material (i.e., códigos de exemplo, etc) deste capítulo estão disponíveis em um repositório criado para este fim `https://github.com/juliosoaresreis/misinformation-course-jai2023`.

Agradecimentos. Este trabalho foi parcialmente financiado por CNPQ, CAPES, MPMG, FAPEMIG e FAPESP.

[55] `https://openai.com/blog/chatgpt`
[56] `https://openai.com/product/dall-e-2`
[57] `https://midjourney.com/home/`
[58] `https://github.com/CompVis/stable-diffusion`
[59] `https://oglobo.globo.com/economia/tecnologia/noticia/2023/03/midjourney-conheca-a-ferramenta-de-inteligencia-artificial-por-tras-da-foto-do-papa-com-casacao.ghtml`

Referências

Ahmed, H., Traore, I., and Saad, S. (2017). Detection of online fake news using n-gram analysis and machine learning techniques. In *Proc. of the Int'l Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments (ISDDC)*, pages 127–138.

Allport, G. W. and Postman, L. (1947). The psychology of rumor. *Henry Holt. American Psychological Association*.

Almeida, S. L., Carvalho, P. R., Evangelo, N., and Filgueiras, R. F. D. (2019). Whatsapp: a desordem da informação na eleição presidencial brasileira de 2018. In *Proc. of the Int'l LAVITS*.

Andreou, A., Venkatadri, G., Goga, O., Gummadi, K., Loiseau, P., and Mislove, A. (2018). Investigating ad transparency mechanisms in social media: A case study of facebook's explanations. In *Proc. of the Network and Distributed System Security Symposium (NDSS)*, pages 1–15.

Arun, C. (2019). On whatsapp, rumours, and lynchings. *Econ. & Political Weekly*, 54(6):30–35.

Atanasova, P., Nakov, P., Màrquez, L., Barrón-Cedeño, A., Karadzhov, G., Mihaylova, T., Mohtarami, M., and Glass, J. (2019). Automatic fact-checking using context and discourse information. *Journal of Data and Information Quality (JDIQ)*, 11(3):1–27.

Badawy, A., Addawood, A., Lerman, K., and Ferrara, E. (2019). Characterizing the 2016 russian ira influence campaign. *Social Network Analysis and Mining*, 9(1):31.

Bakshy, E., Messing, S., and Adamic, L. A. (2015). Exposure to ideologically diverse news and opinion on facebook. *Science*, 348(6239):1130–1132.

Bakshy, E., Rosenn, I., Marlow, C., and Adamic, L. (2012). The Role of Social Networks in Information Diffusion. In *The World Wide Web Conf. (WWW'12)*, pages 519–528.

Bessi, A. and Ferrara, E. (2016). Social bots distort the 2016 us presidential election online discussion. *First Monday*, 21(11).

Bhattacharjee, S. D., Talukder, A., and Balantrapu, B. V. (2017). Active learning based news veracity detection with feature weighting and deep-shallow fusion. In *Proc. of the IEEE Int'l Conference on Big Data (Big Data)*, pages 556–565.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Budak, C., Goel, S., and Rao, J. M. (2016). Fair and balanced? quantifying media bias through crowdsourced content analysis. *Public Opinion Quarterly*, 80(S1):250–271.

Castillo, C., Mendoza, M., and Poblete, B. (2011). Information credibility on twitter. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW)*, pages 675–684.

Chakraborty, A., Ghosh, S., Ganguly, N., and Gummadi, K. P. (2016). Dissemination biases of social media channels: On the topical coverage of socially shared news. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 559–562.

Chen, M.-Y., Lai, Y.-W., and Lian, J.-W. (2023). Using deep learning models to detect fake news about covid-19. *ACM Transactions on Internet Technology*, 23(2):1–23.

Ciampaglia, G. L., Shiralkar, P., Rocha, L. M., Bollen, J., Menczer, F., and Flammini, A. (2015). Computational fact checking from knowledge networks. *PLOS ONE*, 10(6):e0128193.

Conroy, N. J., Rubin, V. L., and Chen, Y. (2015). Automatic deception detection: Methods for finding fake news. In *Proc. of the Annual Meeting of the Association for Information Science and Technology (ASIS&T)*, pages 1–4.

Constantinides, M. (2015). Apps with habits: Adaptive interfaces for news apps. In *Proc. of the Annual ACM Conf. Ext. Abstr. on Human Factors in Comput. Syst. (CHI EA)*, pages 191–194.

Constantinides, M., Dowell, J., Johnson, D., and Malacria, S. (2015). Habito news: A research tool to investigate mobile news reading. In *Proc. of the Int'l Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pages 598–598.

Couto, J. M., Pimenta, B., de Araújo, I. M., Assis, S., Reis, J. C., da Silva, A. P. C., Almeida, J. M., and Benevenuto, F. (2021). Central de fatos: Um repositório de checagens de fatos. In *Anais do III Dataset Showcase Workshop (DSW/SBBD)*, pages 128–137.

Couto, J. M., Reis, J. C., Cunha, Í., Araújo, L., and Benevenuto, F. (2022a). Caracterizando websites de baixa credibilidade no brasil. In *Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, pages 503–516.

Couto, J. M., Reis, J. C., Cunha, Í., Araújo, L., and Benevenuto, F. (2022b). Characterizing low credibility websites in brazil through computer networking attributes. In *Proc. of the IEEE/ACM Intl Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 42–46.

Covert, T. J. A. and Wasburn, P. C. (2007). Measuring media bias: A content analysis of time and newsweek coverage of domestic social issues, 1975–2000. *Social Sci. Quart.*, 88(3):690–706.

Cui, L. and Lee, D. (2020). Coaid: Covid-19 healthcare misinformation dataset.

Cui, L., Shu, K., Wang, S., Lee, D., and Liu, H. (2019). defend: A system for explainable fake news detection. In *Proc. of the Int'l ACM Conference on Information and Knowledge Management (CIKM)*, pages 2961–2964.

da Silva, G. G. (2021). Memes war:: The political use of pictures in brazil 2019. *Philósophos - Revista de Filosofia*, 25(2).

Dai, E., Sun, Y., and Wang, S. (2020). Ginger cannot cure cancer: Battling fake health news with a comprehensive data repository. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 853–862.

Das, A. S., Datar, M., Garg, A., and Rajaram, S. (2007). Google news personalization: scalable online collaborative filtering. In *Proc. of the Int'l ACM Conference on World Wide Web Conference (WWW)*, pages 271–280.

De Angeli, P. and Reis, J. C. (2022). Analyzing the potential of feature groups for misinformation detection in whatsapp. In *Anais Estendidos do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia)*, pages 45–48.

Du, L., Ho, A. T., and Cong, R. (2020). Perceptual hashing for image authentication: A survey. *Signal Processing: Image Communication*, 81:115713.

Ferrara, E. (2017). Disinformation and social bot operations in the run up to the 2017 french presidential election. *First Monday*, 22(8).

Ferrara, E. (2020). What types of covid-19 conspiracies are populated by twitter bots? *First Monday, 25(6)*.

Ferrara, E., Varol, O., Davis, C., Menczer, F., and Flammini, A. (2016). The rise of social bots. *Communications of the ACM*, 59(7):96–104.

Finn, S., Metaxas, P. T., Mustafaraj, E., O'Keefe, M., Tang, L., Tang, S., and Zeng, L. (2014). Trails: A system for monitoring the propagation of rumors on twitter. In *Computation and Journalism Symposium (C+J)*.

Friggeri, A., Adamic, L. A., Eckles, D., and Cheng, J. (2014). Rumor cascades. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 101–110.

Gallagher, K. (2017). The social media demographics report: Differences in age, gender, and income at the top platforms. http://www.businessinsider.com/the-social-media-demographics-report-2017-8.

Gao, W., Li, P., and Darwish, K. (2012). Joint topic modeling for event summarization across news and social media streams. In *Proc. of the Int'l ACM Conference on Information and Knowledge Management (CIKM)*, pages 1173–1182.

Garcin, F., Galle, F., and Faltings, B. (2014). Focal: A personalized mobile news reader. In *Proc. of the Int'l ACM Conference on Recommender Systems (RecSys)*, pages 369–370.

Garimella, K., Morales, G. D. F., Gionis, A., and Mathioudakis, M. (2018). Political discourse on social media: Echo chambers, gatekeepers, and the price of bipartisanship. In *Proc. of the Int'l ACM Conference on World Wide Web Conference (WWW)*, pages 913–922.

Garimella, K. and Tyson, G. (2018). Whatapp doc? a first look at whatsapp public group data. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 511–517.

Gentzkow, M., Shapiro, J. M., and Stone, D. F. (2015). Media bias in the marketplace: Theory. 1:623–645.

Golbeck, J., Mauriello, M., Auxier, B., Bhanushali, K. H., Bonk, C., Bouzaghrane, M. A., Buntain, C., Chanduka, R., Cheakalos, P., Everett, J. B., et al. (2018). Fake news vs satire: A dataset and analysis. In *Proc. of the Int'l ACM Conference on Web Science (WebScience)*, pages 17–21.

Gruppi, M., Horne, B. D., and Adalı, S. (2020). Nela-gt-2019: A large multi-labelled news dataset for the study of misinformation in news articles.

Gupta, A., Kumaraguru, P., Castillo, C., and Meier, P. (2014). Tweetcred: Real-time credibility assessment of content on twitter. In *Proc. of the Int'l Conf. on Social Informatics (SocInfo)*, pages 228–243.

Hui, P.-M., Shao, C., Flammini, A., Menczer, F., and Ciampaglia, G. L. (2018). The hoaxy misinformation and fact-checking diffusion network. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 528–530.

Jin, Z., Cao, J., Jiang, Y.-G., and Zhang, Y. (2014). News credibility evaluation on microblog with a hierarchical propagation model. In *Proc. of the IEEE Int'l Conference on Data Mining (ICDM)*, pages 230–239.

Jin, Z., Cao, J., Zhang, Y., Zhou, J., and Tian, Q. (2017). Novel visual and statistical image features for microblogs news verification. *IEEE Transactions on Multimedia*, 19(3):598–608.

Júnior, M., Melo, P., Kansaon, D., Mafra, V., Sa, K., and Benevenuto, F. (2022). Telegram Monitor: Monitoring Brazilian Political Groups and Channels on Telegram. In *Proc. of the ACM Conference on Hypertext and Social Media (HT)*, page 228–231.

Kansaon, D., Melo, P., and Benevenuto, F. (2022). "click here to join": A large-scale analysis of topics discussed by brazilian public groups on whatsapp. In *Proc. of the Brazilian Symposium on Multimedia and the Web (WebMedia)*.

Keeling, M. J. and Eames, K. T. (2005). Networks and Epidemic Models. *Journal of The Royal Society Interface*, 2(4):295–307.

Kim, J. H., Mantrach, A., Jaimes, A., and Oh, A. (2016). How to compete online for news audience: Modeling words that attract clicks. In *Proc. of the Int'l ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1645–1654.

Kourogi, S., Fujishiro, H., Kimura, A., and Nishikawa, H. (2015). Identifying attractive news headlines for social media. In *Proc. of the Int'l ACM Conference on Information and Knowledge Management (CIKM)*, pages 1859–1862.

Kumar, S., Asthana, R., Upadhyay, S., Upreti, N., and Akbar, M. (2020). Fake news detection using deep learning models: A novel approach. *Transactions on Emerging Telecommunications Technologies*, 31(2):e3767.

Kumar, S., West, R., and Leskovec, J. (2016). Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW)*, pages 591–602.

Kwak, H. and An, J. (2014). A first look at global news coverage of disasters by using the gdelt dataset. In *Proc. of the Int'l Conference on Social Informatics (SocInfo)*, pages 300–308.

Kwon, S., Cha, M., and Jung, K. (2017). Rumor detection over varying time windows. *PloS One*, 12(1):e0168344.

Lazer, D. M. J., Baum, M. A., Benkler, Y., Berinsky, A. J., Greenhill, K. M., Menczer, F., Metzger, M. J., Nyhan, B., Pennycook, G., Rothschild, D., Schudson, M., Sloman, S. A., Sunstein, C. R., Thorson, E. A., Watts, D. J., and Zittrain, J. L. (2018). The science of fake news. *Science*, 359(6380):1094–1096.

Lee, C. S. and Ma, L. (2012). News sharing in social media: The effect of gratifications and prior experience. *Computers in human behavior*, 28(2):331–339.

Leskovec, J., Kleinberg, J., and Faloutsos, C. (2005). Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proc. of the 11th SIGKDD Int'l Conf. on Knowledge Discovery in Data Mining*, pages 177–187.

Li, G. and Zhen, J. (2005). Global stability of an SEI epidemic model with general contact rate. *Chaos, Solitons Fractals*, 23(3):997 – 1004.

Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., Fan, W., and Han, J. (2016). A survey on truth discovery. *ACM SIGKDD Explorations Newsletter*, 17(2):1–16.

Li, Y., Li, Q., Gao, J., Su, L., Zhao, B., Fan, W., and Han, J. (2015). On the discovery of evolving truth. In *Proc. of the Int'l ACM Conf. on Knowl. Disc. and Data Mining (KDD)*, pages 675–684.

Lu, Y.-J. and Li, C.-T. (2020). Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. pages 505–514.

Marques, I., Salles, I., Couto, J. M., Pimenta, B. C., Assis, S., Reis, J. C., da Silva, A. P. C., de Almeida, J. M., and Benevenuto, F. (2022). A comprehensive dataset of brazilian fact-checking stories. *Journal of Information and Data Management*, 13(1).

McAuley, J. and Leskovec, J. (2012). Image Labeling on a Network: Using Social-Network Metadata for Image Classification. In *12th European Conf. on Computer Vision (ECCV12)*.

Melo, P., Benevenuto, F., Kansaon, D., Mafra, V., and Sá, K. (2021). Monitor de whatsapp: Um sistema para checagem de fatos no combate à desinformação. In *Anais Estendidos do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia)*, pages 79–82.

Melo, P., Messias, J., Resende, G., Garimella, K., Almeida, J., and Benevenuto, F. (2019a). Whatsapp monitor: A fact-checking system for whatsapp. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 676–677.

Melo, P., Vieira, C. C., Garimella, K., de Melo, P. O., and Benevenuto, F. (2019b). Can whatsapp counter misinformation by limiting message forwarding? In *Proc. of the Int'l Conference on Complex Networks and their Applications (Complex Networks)*, pages 372–384.

Mitchell, A. (2016). Key findings on the traits and habits of the modern news consumer. `http://www.pewresearch.org/fact-tank/2016/07/07/modern-news-consumer/`.

Monteiro, R. A., Santos, R. L., Pardo, T. A., de Almeida, T. A., Ruiz, E. E., and Vale, O. A. (2018). Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In *Proc. of the Int'l Conf. on Comput. Proces. of the Port. Lang. (PROPOR)*.

Müzell, R. B. (2020). Desinformação e Propagabilidade: uma Análise da Desordem Informacional em Grupos de Whatsapp. Master's thesis, Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Rio Grande do Sul.

Nallapati, R., Feng, A., Peng, F., and Allan, J. (2004). Event threading within news topics. In *Proc. of the Int'l ACM Conf. on Inform. and Knowledge Management (CIKM)*, pages 446–453.

Newman, N., Fletcher, R., Kalogeropoulos, A., and Nielsen, R. K. (2019). Reuters Institute Digital News Report 2019. Reuters Institute for the Study of Journalism.

Nickerson, R. S. (1998). Confirmation bias: A ubiquitous phenomenon in many guises. *Review of general psychology*, 2(2):175–220.

Nørregaard, J., Horne, B. D., and Adalı, S. (2019). Nela-gt-2018: A large multi-labelled news dataset for the study of misinformation in news articles. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 630–638.

Olson, R. S. and Neal, Z. P. (2015). Navigating the Massive World of Reddit: Using Backbone Networks to Map User Interests in Social Media. *PeerJ Computer Science*, 1:e4.

Oxford (2020). Oxford dictionaries: "memes". https://en.oxforddictionaries.com/definition/meme.

Paul, S., Joy, J. I., Sarker, S., Ahmed, S., Das, A. K., et al. (2019). Fake news detection in social media using blockchain. In *Proc. of the Int'l IEEE Conference on Smart Computing & Communications (ICSCC)*, pages 1–5.

Pérez-Rosas, V., Kleinberg, B., Lefevre, A., and Mihalcea, R. (2017). Automatic detection of fake news. pages 3391–3401.

Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., and Stein, B. (2018). A stylometric inquiry into hyperpartisan and fake news. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 231–240.

Poynter (2020). Fighting the infodemic: The coronavirusfacts alliance. https://www.poynter.org/coronavirusfactsalliance/.

Pratiwi, I. Y. R., Asmara, R. A., and Rahutomo, F. (2017). Study of hoax news detection using naïve bayes classifier in indonesian language. In *Proc. of the IEEE Int'l Conference on Information & Communication Technology and System (ICTS)*, pages 73–78.

Quezada, M., Peña-Araya, V., and Poblete, B. (2015). Location-aware model for news events in social media. In *Proc. of the Int'l ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 935–938.

Ratkiewicz, J., Conover, M., Meiss, M. R., Gonçalves, B., Flammini, A., and Menczer, F. (2011). Detecting and tracking political abuse in social media. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 297–304.

Recuero, R., Soares, F., and Vinhas, O. (2021). Discursive strategies for disinformation on whatsapp and twitter during the 2018 brazilian presidential election. *First Monday*.

Reis, J., Benevenuto, F., de Melo, P. O., Prates, R., Kwak, H., and An, J. (2015). Breaking the news: First impressions matter on online news. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 357–366.

Reis, J., Melo, P. d. F., Garimella, K., and Benevenuto, F. (2020a). Can whatsapp benefit from debunked fact-checked stories to reduce misinformation? *The Harvard Kennedy School (HKS) Misinformation Review, 1(5)*.

Reis, J. C. and Benevenuto, F. (2021). Supervised learning for misinformation detection in whatsapp. In *Proc. of the Brazilian Symposium on Multimedia and the Web (WebMedia)*, pages 245–252.

Reis, J. C., Correia, A., Murai, F., Veloso, A., and Benevenuto, F. (2019a). Explainable machine learning for fake news detection. In *Proc. of the ACM Conference on Web Science*, pages 17–26.

Reis, J. C., Correia, A., Murai, F., Veloso, A., and Benevenuto, F. (2019b). Supervised learning for fake news detection. *IEEE Intelligent Systems*, 34(2):76–81.

Reis, J. C., Melo, P., Garimella, K., Almeida, J. M., Eckles, D., and Benevenuto, F. (2020b). A dataset of fact-checked images shared on whatsapp during the brazilian and indian elections. In *Proc. of the Int'l AAAI Conference on Web and Social Media (ICWSM)*, pages 903–908.

Report, D. N. (2018). Statistic of the week: How brazilian voters get their news. https://reutersinstitute.politics.ox.ac.uk/risj-review/statistic-week-how-brazilian-voters-get-their-news.

Resende, G., Melo, P., Sousa, H., Messias, J., Vasconcelos, M., Almeida, J., and Benevenuto, F. (2019). (mis)information dissemination in whatsapp: Gathering, analyzing and countermeasures. In *Proc. of the ACM Web Conference (WWW)*, pages 818–828.

Resende, G., Messias, J., Silva, M., Almeida, J., Vasconcelos, M., and Benevenuto, F. (2018). A system for monitoring public political groups in whatsapp. In *Proc. of the Brazilian Symposium on Multimedia and the Web (WebMedia)*, page 387–390.

Ribeiro, F., Henrique, L., Benevenuto, F., Chakraborty, A., Kulshrestha, J., Babaei, M., and Gummadi, K. P. (2018). Media bias monitor: Quantifying biases of social media news outlets at large-scale. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 290–299.

Ribeiro, F. N., Saha, K., Babaei, M., Henrique, L., Messias, J., Benevenuto, F., Goga, O., Gummadi, K. P., and Redmiles, E. M. (2019). On microtargeting socially divisive ads: A case study of russia-linked ad campaigns on facebook. In *Proc. of the Conference on Fairness, Accountability, and Transparency (FAT)*, pages 140–149.

Ribeiro, M. H., Calais, P. H., Almeida, V. A., and Meira Jr, W. (2017). "everything i disagree with is# fakenews": Correlating political polarization and spread of misinformation. In *Proc. of the Workshop on Data Science + Journalism @KDD*.

Rubin, V., Conroy, N., Chen, Y., and Cornwell, S. (2016). Fake news or truth? using satirical cues to detect potentially misleading news. In *Proc. of the Workshop on Computational Approaches to Deception Detection (NAACL-HLT)*, pages 7–17.

Rubin, V. L., Chen, Y., and Conroy, N. J. (2015). Deception detection for news: three types of fakes. In *Proc. of the Annual Meeting of the Association for Information Science and Technology (ASIS&T)*, page 83. American Society for Information Science.

Ruchansky, N., Seo, S., and Liu, Y. (2017). Csi: A hybrid deep model for fake news detection. In *Proc. of the Int'l ACM Conference on Information and Knowledge Management (CIKM)*, pages 797–806.

Saez-Trumper, D. (2014). Fake tweet buster: a webtool to identify users promoting fake news on twitter. In *Proc. of the ACM Conference on Hypertext and Social Media (HT)*, pages 316–317.

Salem, F. K. A., Al Feel, R., Elbassuoni, S., Jaber, M., and Farah, M. (2019). Fa-kes: A fake news dataset around the syrian war. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 573–582.

Santia, G. and Williams, J. (2018). Buzzface: A news veracity dataset with facebook user commentary and egos. In *Proc. of the Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 531–540.

Sasahara, K., Chen, W., Peng, H., Ciampaglia, G. L., Flammini, A., and Menczer, F. (2020). Social influence and unfollowing accelerate the emergence of echo chambers. *Journal of Computational Social Science*, pages 1–22.

Shao, C., Ciampaglia, G. L., Flammini, A., and Menczer, F. (2016). Hoaxy: A platform for tracking online misinformation. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW) Companion*, pages 745–750.

Shao, C., Ciampaglia, G. L., Varol, O., Yang, K.-C., Flammini, A., and Menczer, F. (2018). The spread of low-credibility content by social bots. *Nature communications*, 9(1):4787.

Sharma, K., Qian, F., Jiang, H., Ruchansky, N., Zhang, M., and Liu, Y. (2019). Combating fake news: A survey on identification and mitigation techniques. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(3):1–42.

Sheidin, J., Lanir, J., Kuflik, T., and Bak, P. (2017). Visualizing spatial-temporal evaluation of news stories. In *Proc. of the Int'l Conference on Intelligence User Interfaces (IUI) Companion*, pages 65–68.

Shu, K., Cui, L., Wang, S., Lee, D., and Liu, H. (2019). defend: Explainable fake news detection. In *Proc. of the Int'l ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 395–405.

Shu, K., Sliva, A., Wang, S., Tang, J., and Liu, H. (2017). Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36.

Silva, M., Santos de Oliveira, L., Andreou, A., Vaz de Melo, P. O., Goga, O., and Benevenuto, F. (2020). Facebook ads monitor: An independent auditing system for political ads on facebook. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW)*, pages 224–234.

Soares, F. B., Recuero, R., Volcan, T., Fagundes, G., and Sodré, G. (2021). Desinformação sobre o covid-19 no whatsapp: a pandemia enquadrada como debate político. *Ciência da Informação em Revista*, 8(1):74–94.

Tacchini, E., Ballarin, G., Della Vedova, M. L., Moret, S., and de Alfaro, L. (2017). Some like it hoax: Automated fake news detection in social networks. In *Proc. of the Workshop on Data Science for Social Good (SoGood)*.

Tardaguila, C., Benevenuto, F., and Ortellado, P. (2018). Fake news is poisoning brazilian politics. whatsapp can stop it. `https://www.nytimes.com/2018/10/17/opinion/brazil-election-fake-news-whatsapp.html`.

Tomás, R., Tomás, L., and Andreatta, E. (2020). Da depravação ao desperdício de recursos: Estratégias de desconstrução da universidade pública em redes de fake news. *VERBUM. Cadernos de Pós-Graduação.*, 9(2):141–167.

Tschiatschek, S., Singla, A., Gomez Rodriguez, M., Merchant, A., and Krause, A. (2018). Fake news detection in social networks via crowd signals. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW) Companion*, pages 517–524.

Tversky, A. and Kahneman, D. (1992). Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and uncertainty*, 5(4):297–323.

Venkatadri, G., Andreou, A., Liu, Y., Mislove, A., Gummadi, K. P., Loiseau, P., and Goga, O. (2018). Privacy risks with facebook's pii-based targeting: Auditing a data broker's advertising interface. In *Proc. of the IEEE Symposium on Security and Privacy (SP)*, pages 89–107.

Vlachos, A. and Riedel, S. (2014). Fact checking: Task definition and dataset construction. In *Proc. of the ACL Workshop on Lang. Technol. and Computat. Social Science*, pages 18–22.

Volkova, S., Shaffer, K., Jang, J. Y., and Hodas, N. (2017). Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on twitter. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 647–653.

Vosoughi, S., Roy, D., and Aral, S. (2018). The spread of true and false news online. *Science*, 359(6380):1146–1151.

Wang, P., Angarita, R., and Renna, I. (2018a). Is this the era of misinformation yet: combining social bots and fake news to deceive the masses. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW) Companion*, pages 1557–1561.

Wang, W. Y. (2017). "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proc. of the Annual Meeting of the Assoc. for Computat. Linguistics (ACL)*, pages 422–426.

Wang, Y., Ma, F., Jin, Z., Yuan, Y., Xun, G., Jha, K., Su, L., and Gao, J. (2018b). Eann: Event adversarial neural networks for multi-modal fake news detection. In *Proc. of the Int'l ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 849–857.

Wang, Y., Yang, W., Ma, F., Xu, J., Zhong, B., Deng, Q., and Gao, J. (2020). Weak supervision for fake news detection via reinforcement learning. In *Proc. of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 516–523.

Ward, A., Ross, L., Reed, E., Turiel, E., and Brown, T. (1997). Naive realism in everyday life: Implications for social conflict and misunderstanding. *Values and knowledge*, pages 103–135.

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of 'lavits-world' networks. *Nature*, 393(6684):440.

Wei, W. and Wan, X. (2017). Learning to identify ambiguous and misleading news headlines. In *Proc. of the Int'l Joint Conference on Artificial Intelligence (IJCAI)*, pages 4172–4178.

Westlund, O. (2013). Mobile news: A review and model of journalism in an age of mobile media. *Digital journalism*, 1(1):6–26.

Yang, F., Pentyala, S. K., Mohseni, S., Du, M., Yuan, H., Linder, R., Ragan, E. D., Ji, S., and Hu, X. (2019). Xfake: explainable fake news detector with visualizations. In *Proc. of the ACM Web Conference (WWW)*, pages 3600–3604.

Zajonc, R. B. (1968). Attitudinal effects of mere exposure. *Journal of personality and social psychology*, 9(2p2):1.

Zhang, J., Cui, L., Fu, Y., and Gouza, F. B. (2018). Fake news detection with deep diffusive network model. In *arXiv preprint arXiv:1805.08751*.

Zhao, Z., Resnick, P., and Mei, Q. (2015). Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proc. of the Int'l ACM Conference on World Wide Web (WWW) Companion*, pages 1395–1405.

Capítulo 2

Confiabilidade e Segurança nos Sistemas Distribuídos Físico-Digitais

Raimundo J. de A. Macêdo[1], Alirio S. de Sá[1], Allan E. S. Freitas[2], Paulo E. Veríssimo[3], Sérgio Gorender[1], Margarete O. S. de Sá[1]

[1]Universidade Federal da Bahia, Brasil
[2]Instituto Federal de Educação, Ciência e Tecnologia da Bahia, Brasil
[3]King Abdullah University of Science and Technology, Reino da Arábia Saudita

Abstract

Distributed cyber-physical systems, also referred to as physical-digital systems, are characterized by geographically dispersed physical components interconnected through communication networks, thereby creating a decentralized and collaborative infrastructure. These systems encompass a wide range of devices such as temperature sensors, industrial robots, biosensors, drones, cameras, and other devices that interact with the physical environment. They find application across various industries including manufacturing, transportation, healthcare, environmental control, entertainment, commerce, and security. However, the physical-digital relationship in these scenarios introduces significant challenges such as interoperability, reliability, cybersecurity, and scalability. To address these challenges, the utilization of autonomic management in these systems proves beneficial. Autonomic management enables the systems to self-monitor their states and make decisions based on the collected data. This text aims to present a theoretical-practical approach for constructing autonomous or autonomic distributed systems that can fulfill the performance, reliability, and security requirements of distributed physical-digital systems. Understanding the increasing complexity of these systems and effectively managing them is crucial to ensuring their efficiency and effectiveness.

Resumo

Sistemas ciberfísicos distribuídos, também conhecidos como sistemas físico-digitais, são caracterizados por componentes físicos geograficamente dispersos e interconectados por meio de redes de comunicação, criando assim uma infraestrutura descentralizada e colaborativa. Esses sistemas abrangem uma ampla variedade de dispositivos, como sensores de

temperatura, robôs industriais, biossensores, drones, câmeras e outros dispositivos que interagem com o ambiente físico. Eles encontram aplicação em diversas indústrias, incluindo manufatura, transporte, saúde, controle ambiental, entretenimento, comércio e segurança. No entanto, a relação físico-digital nesses cenários apresenta desafios significativos, como interoperabilidade, confiabilidade, cibersegurança e escalabilidade. Para enfrentar esses desafios, a utilização do gerenciamento autonômico nesses sistemas se mostra benéfica. O gerenciamento autonômico permite que os sistemas monitorem seus próprios estados e tomem decisões com base nos dados coletados. Este texto tem como objetivo apresentar uma abordagem teórico-prática para a construção de sistemas distribuídos autônomos ou autonômicos capazes de atender aos requisitos de desempenho, confiabilidade e segurança dos sistemas físico-digitais distribuídos. Compreender a crescente complexidade desses sistemas e gerenciá-los de forma eficaz é crucial para garantir sua eficiência e eficácia.

2.1. Contexto e Motivações

2.1.1. Introdução

Sistemas físico-digitais distribuídos, também conhecidos como sistemas ciberfísicos distribuídos (do inglês, *cyber-physical systems*) são aqueles em que os componentes físicos estão geograficamente distribuídos e conectados por meio de redes de comunicação, formando uma infraestrutura descentralizada e colaborativa. Tais sistemas abrangem uma ampla variedade de dispositivos, como sensores de temperatura, robôs industriais, biossensores, drones, câmeras e outros dispositivos de interação com o ambiente físico. Essas características estão presentes em ambientes como os da indústria 4.0, da indústria 5.0, de realidade virtual e aumentada, de metaverso, entre outros, com aplicação em diversos setores estratégicos, incluindo manufatura, transporte, saúde, controle ambiental, entretenimento, comércio e segurança. Utilizaremos os termos "físico-digital" e "ciberfísico" de forma intercambiável no decorrer do texto.

Além dos requisitos dos sistemas distribuídos convencionais, como interoperabilidade, confiabilidade, segurança cibernética e escalabilidade, a relação físico-digital apresenta desafios adicionais devido à complexa interação entre o sistema computacional e os processos físicos subjacentes. Dessa forma, os sistemas físico-digitais envolvem o projeto seguro e a operação temporalmente previsível de sistemas de computação que interagem com o ambiente [Kopetz 1997; Macêdo et al. 2004], com sofisticação cada vez maior, incluindo: infraestruturas de controle de computador; monitoramento e operações de saúde; veículos autônomos em terra, ar e espaço; sistemas robóticos; monitoramento domiciliar e ambiental; etc. O projeto desses sistemas exige os fundamentos teóricos combinados de distribuição, tempo real e computação embarcada, lidando, nas mais complexas dessas infraestruturas, com um conflito frequente entre complexidade, imprevisibilidade e a necessidade de operação. Além disso, a capacidade de tornar essas propriedades sustentáveis ao longo do tempo, sob cenários de ameaças persistentes e/ou em evolução, é um grande desafio. Nesses sistemas, a operação em tempo real é fundamental, não há como desligar ou fechar, para obstar um ataque de segurança. Essa lacuna é agravada pela convergência dos Sistemas físico-digitais com o complexo Internet-Nuvem-Web.

As consequências de falhas operacionais ou ataques à segurança podem ser graves em alguns cenários, em particular porque os sistemas físico-digitais muitas vezes operam em proximidade com humanos, podendo assim causar danos físicos, incluindo risco de vida. Nessas condições, definitivamente são necessários sistemas resilientes e dinamicamente

adaptáveis a novas condições operacionais. Por exemplo, um sistema computacional encarregado de controlar uma rede ferroviária deve obedecer a restrições temporais precisas. Além disso, em tais cenários, há não apenas a interação com elementos físicos, mas também a integração de processos computacionais externos, abrangendo diversos domínios, desde a Internet das Coisas até a Computação em Nuvem e a participação humana.

Para lidar com tais desafios, paradigmas de proteção baseados em segurança clássica ou confiabilidade clássica, embora necessários, são insuficientes se trabalharem isoladamente e de forma rígida. Muitas falhas induzidas por usuários de forma maliciosa ou acidental levam à compreensão da necessidade de tratar simultaneamente falhas acidentais, erros de projeto e condições operacionais inesperadas, aumentando a robustez e resiliência desses sistemas. Nesse contexto, busca-se alcançar a resiliência dos sistemas computacionais, proporcionando uma forma autonômica de gerenciá-los de maneira resistente a ameaças diversas, sejam acidentais ou maliciosas. Nessa perspectiva, os sistemas devem possuir defesas incorporadas desde os princípios iniciais, que sejam capazes de lidar com diferentes tipos de ameaças e possam ser configurados de forma incremental. Além disso, esses sistemas devem responder automaticamente a ameaças, tolerar falhas e intrusões, adaptar-se à gravidade das situações e fornecer operação autônoma e sustentável ao longo de sua vida, com capacidade de recuperação e autocorreção.

Para lidar com esses desafios, em tempos e na qualidade desejada, uma abordagem frequentemente utilizada é a gestão autônoma ou autonômica, na qual os sistemas são configurados para monitorar seus próprios estados e tomar decisões com base em dados coletados, visando atender aos requisitos funcionais e de qualidade de serviço dinamicamente estabelecidos pelo ambiente operacional ou dos utilizadores de tais sistemas [Macêdo 2008b; Kumar et al. 2017]. Ademais, dada a grande complexidade de sistemas físico-digitais em larga escala, além da desejada autonomia, os problemas de confiabilidade e segurança são tratados em uma perspectiva unificadora e integrada [Veríssimo et al. 2009].

Para melhor compreensão da literatura sobre confiabilidade e segurança nos sistemas distribuídos físico-digitais, apresentaremos, resumidamente, um panorama da literatura clássica sobre esses conceitos, o que ajudará no percurso das demais seções.

2.1.2. Modelos de Sistemas Distribuídos

O campo dos sistemas distribuídos iniciou-se na década de 1970 com o surgimento das primeiras redes. A ideia fundamental era a de que computadores distintos poderiam colaborar e coordenar-se para atingir objetivos específicos, compartilhando recursos entre processos distribuídos. A pesquisa nesse campo pode ser dividida em dois subcampos principais: modelos de computabilidade e algoritmos básicos, por um lado, e infraestrutura e ferramentas de software, por outro. Os modelos de computabilidade tratam da resolução de problemas fundamentais e recorrentes de computação distribuída (por exemplo, consenso, exclusão mútua, etc.) em modelos específicos. A infraestrutura e as ferramentas de software abordam os problemas de como desenvolver e implantar sistemas a partir de processos distribuídos, o que envolve padrões de comunicação, como chamadas de procedimento remoto (RPC), disponibilização de serviços, protocolos padrão para comunicação e descoberta de objetos/serviços, segurança, e assim por diante.

Um sistema distribuído (SD) é composto por um conjunto Π de processos que se comunicam e sincronizam entre si, trocando mensagens por meio de canais de comunicação. Essa definição é geralmente ampliada para incluir especificações sobre o comportamento dos

processos e canais, abordando garantias temporais das operações e padrões de falha. A pesquisa no subcampo de computabilidade e algoritmos básicos concentra-se, em geral, na análise da computabilidade e complexidade dos problemas fundamentais e recorrentes encontrados em sistemas distribuídos, abrangendo uma variedade de modelos de SD. Por exemplo, podemos ter um modelo de sistema distribuído onde cada par de processos está conectado por um canal bidirecional assíncrono e confiável. Isso significa que esses canais não criam, modificam ou perdem mensagens, e as mensagens podem levar um tempo arbitrário e ilimitado para serem entregues. Esse modelo é frequentemente chamado de tempo-livre ou assíncrono, e foi comprovado que não existe uma solução determinística para o problema de consenso nesse modelo [Fisher et al. 1985]. O problema de consenso, que se refere a acordos entre processos distribuídos sobre determinados estados, é recorrente e serve de bloco de construção para outros problemas [Hurfin et al. 1999]. Por outro lado, em sistemas síncronos, onde os atrasos de transmissão de mensagens e execução de processos são limitados, vários problemas relacionados à computação distribuída, como consenso e *membership*, foram resolvidos [Lamport et al. 1982; Cristian 1991]. A má notícia é que as suposições do sistema síncrono são válidas apenas em cenários restritos. Por exemplo, essas suposições não se aplicam à Internet.

Em muitos casos, no entanto, a escolha entre um modelo síncrono ou um modelo assíncrono pode não ser a melhor opção. Sistemas reais frequentemente apresentam comportamentos síncronos em partes específicas ou em certas condições de operação. Isso tem levado ao desenvolvimento de modelos intermediários, ou híbridos, que se situam entre os modelos síncronos e assíncronos, nos quais foi provado que muitos problemas clássicos de sistemas distribuídos, como consenso, têm solução [Dwork et al. 1988; Chandra e Toueg 1996; Veríssimo e Casimiro 2002; Gorender e Macêdo 2002; Gorender, Macêdo e Raynal 2005; Macêdo, Gorender e Cunha 2005; Veríssimo 2006; Macêdo e Gorender 2008; Macêdo e Gorender 2012].

Alguns desses modelos híbridos assumem comportamentos alternados ao longo do tempo, como o modelo de sincronia temporal proposto por [Cristian e Fetzer 1999] e o modelo assíncrono com detectores de defeitos desenvolvido por [Chandra e Toueg 1996]. Outros modelos intermediários permitem composições heterogêneas, como o modelo TCB [Veríssimo e Casimiro 2002] e o modelo Síncrono Particionado [Macêdo e Gorender 2008; Macêdo e Gorender 2009; Macêdo e Gorender 2012]. Esses modelos híbridos heterogêneos, que combinam aspectos síncronos e assíncronos, têm um maior potencial de se adequarem aos sistemas físico-digitais, pois interações físicas requerem respostas em tempos pré-estabelecidos, enquanto as interações em redes de larga escala são geralmente assíncronas.

2.1.3. Confiabilidade e conceitos relacionados

Confiabilidade (*reliability*) e disponibilidade (*availability*) são propriedades complementares que qualificam a confiança que pode ser depositada no funcionamento correto de um sistema, também conhecida como *dependabilidade* (*dependability*) do sistema [Jalot 1994; Kopetz 1997; Avizienis et al. 2004]. A confiabilidade está relacionada à capacidade do sistema de se manter operacional, fornecendo continuamente um serviço correto, enquanto a disponibilidade refere-se à capacidade do sistema de estar disponível, alternando entre os estados operacional e não operacional. Para lidar com a ocorrência de falhas, se mantendo operacional, o sistema precisa dispor de mecanismos de tolerância a falhas. Como não é

possível controlar todos os fatores que levam a falhas, a confiabilidade e a disponibilidade são melhor expressas em termos probabilísticos [Cristian 1991; Veríssimo e Rodrigues 2000].

O mau funcionamento do sistema distribuído físico-digital pode implicar em prejuízos financeiros, ambientais, ou até mesmo perda de vidas humanas. Por melhores que sejam as técnicas usadas na construção destes sistemas, diversos fatores podem levar os mesmos a apresentarem defeitos: desgastes físicos de componentes, falhas humanas, fenômenos e/ou desastres naturais etc. Portanto, mecanismos adequados de tolerância a falhas devem ser empregados em tais sistemas. Os termos *falha*, *erro*, *defeito*, adotados neste texto, seguem o padrão de terminologia amplamente adotado pela comunidade científica internacional da área de tolerância a falhas [Avizienis et al. 2004]; sendo que *fault*, *error* e *failure* são traduzidos, respectivamente, para falha, erro e defeito.

A semântica de falhas [Cristian 1991] está relacionada ao comportamento de componentes ou subsistemas que apresentam falhas. Esse comportamento é influenciado pelo ambiente computacional e de comunicação subjacente. Em particular, as garantias temporais de operação desempenham um papel crucial. Como apresentado anteriormente, em modelos síncronos, há garantias temporais nas operações internas aos processos e na comunicação entre processos. Por outro lado, em modelos assíncronos, não há garantias temporais. A escolha da técnica de tolerância a falhas depende da semântica de falhas e é implementada por meio de redundância de componentes, estruturas ou procedimentos.

Conceitualmente, um defeito denota um desvio entre o comportamento de um serviço e o que foi especificado para tal serviço [Avizienis et al. 2004]. Tal desvio é originado de um estado errôneo (i.e., erro) ao qual é levado o sistema na presença de uma falha. Um erro pode permanecer latente até que algum evento no sistema promova a sua ativação. Como a falha é um evento indesejado que pode inserir um erro no sistema, pode-se afirmar que a falha é a causa indireta e primária do defeito – i.e., o evento de falha causa o estado de erro que, quando ativado, leva a um serviço defeituoso. Ou seja, um erro pode permanecer latente até que algum evento no sistema o ative.

Os conceitos de defeito e falha são relativizados pela organização hierárquica dos sistemas. De modo geral, um sistema é composto por outros sistemas (ditos subsistemas ou componentes). Da mesma forma, estes subsistemas são estruturados a partir de outros componentes e assim por diante, até se alcançar os subsistemas mais elementares, para os quais esta subdivisão não é evidente ou não tem um sentido próprio. O desvio no serviço provido por um componente ou subsistema (i.e., defeito) representa uma falha para o sistema ao qual tal componente está inserido, conforme ilustrado na Figura 2.1. Se a respectiva falha não for devidamente tratada, o erro gerado no estado do sistema pode eventualmente fazer com que o respectivo serviço esteja fora de sua especificação (defeito).

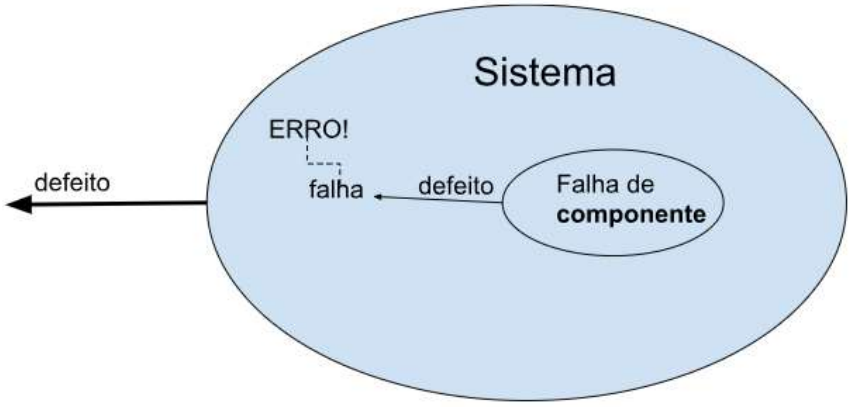


Figura 2.1: Hierarquia falha-erro-defeito

2.1.3.1. Classificação das falhas

A tipificação de falhas é feita considerando como as falhas afetam o comportamento dos componentes do sistema. Existem várias formas de classificar as falhas [Avizienis et al. 2004; Jalote 1994]. Na literatura técnica relacionada, são comumente utilizadas as seguintes classes de falhas:

- **Falha silenciosa** – ocorre quando o componente para de funcionar, deixando de responder a qualquer requisição de serviço. Esse é o tipo de falha mais comum utilizada em projetos de sistemas distribuídos. Nos sistemas distribuídos caracterizados como síncronos, essas falhas podem ser detectadas e recebem o nome de falhas do tipo *fail-stop*; nos modelos assíncronos, essas falhas são apenas estimadas e são denominadas falhas do tipo *crash*.
- **Falha por omissão** – ocorre quando o componente omite (ou deixa de produzir) o resultado esperado, conforme sua especificação. Observe que a falha por parada é um tipo particular de falha por omissão, na qual o componente omite permanentemente os resultados esperados.
- **Falha temporal ou de desempenho** – ocorre quando o componente responde fora do prazo especificado. Observe que a falha por omissão pode ser caracterizada como uma falha temporal, na qual o resultado esperado nunca será produzido dentro do prazo, caso este exista. Falhas temporais fazem somente sentido em sistemas distribuídos síncronos, onde os tempos de execução de processos e transmissão de mensagens são conhecidos.
- **Falha de valor** – ocorre quando o componente produz uma saída incorreta para um dado valor de entrada.
- **Falha arbitrária ou bizantina** – ocorre quando um componente pode manifestar qualquer tipo de comportamento na presença de falha, podendo parar de funcionar, omitir resultados, produzir resultados fora do prazo ou ainda produzir valores incorretos. A Figura 2.2 ilustra a hierarquia de classes apresentadas acima.

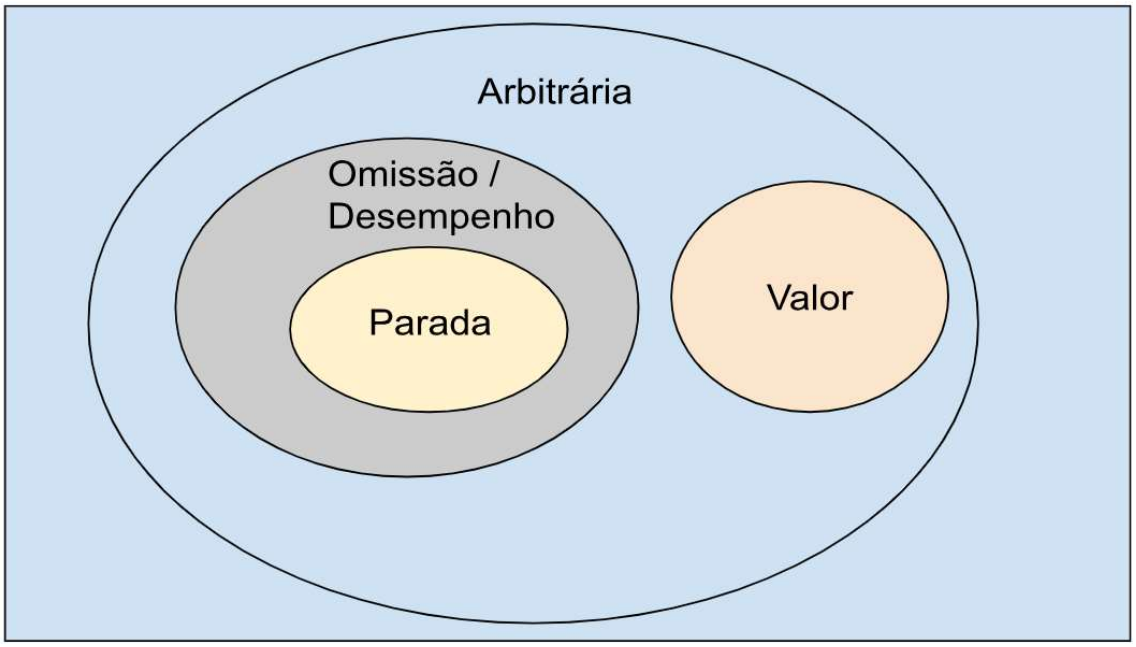


Figura 2.2. Classes de falha

2.1.4. Segurança

A segurança em sistemas distribuídos físicos-digitais é de extrema importância para garantir a integridade, confidencialidade e disponibilidade da infraestrutura crítica e dos dados sensíveis. Esses sistemas combinam elementos físicos com recursos computacionais, criando um ambiente complexo de rede onde o impacto de violações de segurança pode ter consequências graves.

A natureza desses sistemas introduz desafios de segurança únicos. Os ataques podem visar tanto os componentes cibernéticos quanto físicos, visando interromper as operações, comprometer a integridade dos dados ou obter acesso não autorizado. Portanto, uma abordagem abrangente de segurança é necessária para lidar com essas preocupações.

Um aspecto fundamental para garantir a segurança em sistemas distribuídos físico-digitais é a implementação de mecanismos robustos de autenticação e controle de acesso. Autenticação forte de usuário, protocolos de comunicação seguros e permissões de acesso detalhadas são essenciais para evitar que indivíduos não autorizados assumam o controle de componentes críticos ou acessem informações sensíveis.

A integridade e a confidencialidade dos dados são fundamentais para proteger o sistema contra modificações não autorizadas ou violações de dados. Criptografia, armazenamento seguro e protocolos de comunicação seguros desempenham um papel vital na proteção de dados sensíveis contra interceptação, adulteração ou divulgação não autorizada.

Outra consideração crítica é a resiliência do sistema contra vários tipos de ataques. Sistemas de detecção de intrusões, algoritmos de detecção de anomalias e mecanismos de monitoramento de segurança ajudam a identificar e mitigar ameaças potenciais em tempo real. Além disso, a implementação de estratégias de redundância, tolerância a falhas e recuperação de desastres pode garantir a disponibilidade do sistema e uma resposta rápida a incidentes de segurança.

A realização de avaliações de segurança regulares, testes de vulnerabilidade e atualizações é crucial para manter a segurança dos sistemas distribuídos físicos-digitais. À medida que surgem novas vulnerabilidades e técnicas de ataque evoluem, é essencial estar atualizado com as melhores práticas de segurança e aplicar as devidas atualizações para mitigar riscos potenciais.

Em suma, a segurança em sistemas distribuídos físico-digitais requer uma abordagem holística e proativa. Ao integrar medidas de segurança robustas, monitoramento contínuo e avaliações regulares, as organizações podem melhorar a resiliência e a confiabilidade desses sistemas críticos, protegendo-os contra ameaças potenciais e garantindo seu funcionamento confiável no mundo interconectado de hoje.

2.2. Gestão autonômica de sistemas distribuídos

2.2.1. Sistemas autogerenciávies

Nesta seção, descreveremos as motivações que levaram pesquisadores de computação a proporem estruturas autogerenciáveis, ou autonômicas, para lidar com a complexidade crescente dos sistemas computacionais distribuídos, sobretudo quando se consideram sistemas de sistemas, computação em nuvem, Internet das Coisas, sistemas físicos-digitais e, mais recentemente, sistemas da Indústria 4.0 e da Indústria 5.0. O termo autonômico será precisamente definido e contextualizado para os sistemas distribuídos; todavia, para melhor entendimento da literatura sobre o tema, apresentaremos resumidamente para o leitor os esforços e terminologias utilizadas em diversas áreas da computação onde a propriedade de autonomia é tratada de forma semelhante, ainda que com enfoques variados e/ou em camadas distintas dos sistemas computacionais: engenharia de software, sistemas de tempo real, sistemas robóticos, gerência de redes, entre outras áreas.

A Iniciativa de Computação Autônoma da IBM (ACI) visa o desenvolvimento de sistemas computacionais capazes de autogerenciamento, tomando decisões autonomamente a partir de políticas de alto nível, caracterizados por propriedades de auto-*, como autoconfiguração, autorregeneração, autootimização, autoproteção entre outras [Horn 2001]. Desde o lançamento da iniciativa em 2001, grandes esforços de pesquisa têm se concentrado em como projetar tais sistemas e em mecanismos para as propriedades de auto-*. É senso comum na literatura que, para serem autônomos, os sistemas devem ser compostos por uma hierarquia de subsistemas autônomos, desde o nível mais alto (nível do usuário) até os componentes básicos mais baixos.

A ideia de autogerenciamento ou autonomia é muito antiga. De fato, até mesmo o próprio conceito de execução automática de programas de computador em arquiteturas Von Neumann traz a ideia de execução autônoma no nível da máquina. Além disso, o conceito de autonomia tem sido explorado em muitos campos, como inteligência artificial, robótica, engenharia de software, gerenciamento de redes, automação e sistemas de controle, computação biológica, confiabilidade, etc. No entanto, as principais contribuições da ACI parecem ser a perspectiva de avaliar o quão autônomo (ou autônoma) um sistema pode ser - por meio da proposição de níveis de maturidade autônoma - e também de disseminar a necessidade urgente de sistemas mais autônomos para lidar com a complexidade cada vez maior de gerenciamento dos sistemas e aplicativos de computação modernos. Paradigmas como a computação em nuvem, que permitem maior flexibilidade para os usuários finais em termos de alocação de recursos e composições de serviços, têm reforçado a necessidade de

sistemas autônomos. O fato de os usuários não precisarem planejar antecipadamente o provisionamento de recursos e os recursos computacionais (como largura de banda de rede, CPU, etc.) serem alocados sob demanda para atender aos acordos de nível de serviço definidos pelo usuário torna esses novos paradigmas muito atrativos. No entanto, se adaptar à variabilidade de recursos e às mudanças nos requisitos do usuário em tempo de execução é um grande desafio. Como consequência, esses sistemas precisam ter a capacidade de serem autogerenciáveis, se reconfigurando e se ajustando continuamente para atingir certos objetivos.

A seguir apresentaremos uma visão dos sistemas distribuídos, a partir da literatura clássica em uma perspectiva autonômica, como apresentado em [Macêdo 2013].

2.2.2. Sistemas Distribuídos Autonômicos

As suposições clássicas em modelos de sistemas distribuídos não levam em consideração a possibilidade de mudanças ao longo do tempo, o que representa uma limitação para sistemas autonômicos, nos quais as características do sistema de computação subjacente podem se alterar dinamicamente. Portanto, argumentamos que um sistema distribuído autonômico deve estar ciente da validade de suas próprias suposições, seguindo as garantias verificadas de qualidade de serviço e permitindo que os algoritmos se adaptem de acordo. Por exemplo, aproveitando comportamentos síncronos percebidos, quando possível, ou alternando para soluções síncronas quando as suposições relacionadas não forem mais válidas.

A evolução de um sistema distribuído frequentemente envolve se adaptar a novos requisitos de aplicação ou mudanças no ambiente de computação, como falhas ou adição/remoção de recursos computacionais. Portanto, ser adaptativo é um requisito fundamental para lidar com a natureza dinâmica desses sistemas. Consequentemente, sistemas adaptativos têm recebido grande atenção em áreas como confiabilidade, sistemas de tempo real e engenharia de software. Por exemplo, um sistema tolerante a falhas é inerentemente adaptativo, pois pode continuar funcionando corretamente mesmo diante de falhas de componentes, demonstrando adaptabilidade em relação a falhas de tais componentes.

É possível projetar sistemas que podem modificar seus mecanismos de tolerância a falhas durante a execução para se adequar a novas condições operacionais ou otimizar aspectos de desempenho, como sobrecarga de mensagens. Para compreender a natureza autonômica de um sistema distribuído em termos de sua adaptabilidade, classificamos os sistemas distribuídos autonômicos com base no grau de controle que podem exercer sobre os mecanismos adaptativos, se aplicável, da seguinte forma, conforme proposto por [Macêdo 2012]:

1. **Não adaptativo**: não são fornecidas capacidades de adaptação.
2. **Adaptativo offline**: a adaptação é possível, mas apenas antes da execução, sem adaptação em tempo de execução.
3. **Adaptativo online**: a adaptação ocorre em tempo de execução com base em objetivos predefinidos, mas as políticas não podem ser controladas durante a execução.
4. **Adaptativo autonômico**: as políticas ou objetivos de adaptação podem ser modificados em tempo de execução.

Para permitir a implantação de sistemas distribuídos autonômicos, os blocos fundamentais de construção dos sistemas distribuídos devem fornecer a capacidade para que

as aplicações ou serviços de camada superior possam controlar seu comportamento com base em políticas, ou objetivos específicos. Essas políticas ou objetivos podem definir requisitos de qualidade de serviço (QoS), restrições do ambiente computacional ou uma combinação de ambos. Um aspecto crucial do projeto é estabelecer objetivos ou derivá-los de acordos de nível de serviço (SLAs) de níveis superiores. Nós aplicamos essa abordagem geral para implementar detectores de defeitos [de Sá e Macêdo 2010], comunicação em grupo [Macêdo et al. 2013] e replicação bizantina [de Sá et al. 2013], que são blocos de construção essenciais em sistemas distribuídos tolerantes a falhas.

Por exemplo, no caso dos detectores de defeitos, o sistema de camada superior, seja um usuário humano ou outro sistema, pode controlar o QoS desejado para a detecção de defeitos dentro dos recursos disponíveis, como tempo de detecção e precisão. Nesse caso, temos um comportamento autonômico adaptativo, pois o objetivo autonômico pode ser alterado dinamicamente. A adaptação online representa uma forma mais fraca de sistema distribuído autonômico. É importante observar que a definição de tais objetivos autonômicos ou políticas de adaptabilidade, depende do entendimento da dinâmica dos blocos de construção de sistemas distribuídos relacionados e respectivos compromissos de desempenho.

No desenvolvimento de sistemas distribuídos físico-digitais, embora os mecanismos adaptativos sejam eficazes para garantir confiabilidade e segurança, esses mecanismos também podem precisar ser modificados para lidar com mudanças imprevistas no ambiente computacional e/ou requisitos de clientes/sistemas. Essas mudanças podem incluir novas configurações do sistema, atualizações nos componentes do sistema ou até mesmo novos acordos de nível de serviço (SLAs). Portanto, muitas vezes, a adaptação deve ocorrer em tempo real, sem conhecimento prévio completo ou antecipação de eventos potenciais.

Para lidar com esses desafios, o sistema precisa possuir comportamento autogerenciável ou autonômico em sua capacidade máxima (i.e., adaptativo autonômico). Isso significa que o sistema deve ser capaz de alterar dinamicamente suas políticas de adaptação em tempo de execução, conforme necessário. Portanto, os laços de retroalimentação são essenciais para detectar tanto o ambiente quanto os requisitos do sistema. Os sistemas devem se adaptar dinamicamente com base nessas condições percebidas e em políticas de nível superior.

Para projetar um protocolo autonômico desse tipo, é necessário abordar cuidadosamente várias questões. Entre elas, destacam-se: quais devem ser os objetivos de desempenho do protocolo e como eles podem ser expressos de maneira adequada? Como as dinâmicas do sistema e do protocolo podem ser modeladas de forma eficaz dentro dos laços de retroalimentação? Em quais momentos e de que forma o sistema deve se adaptar a diferentes modos de operação? E, por fim, com qual frequência os componentes do sistema e as variáveis do protocolo devem ser monitorados e reconfigurados?

Em geral, cada protocolo autonômico é modelado como um laço de controle realimentado, como ilustrado na Figura 2.3.

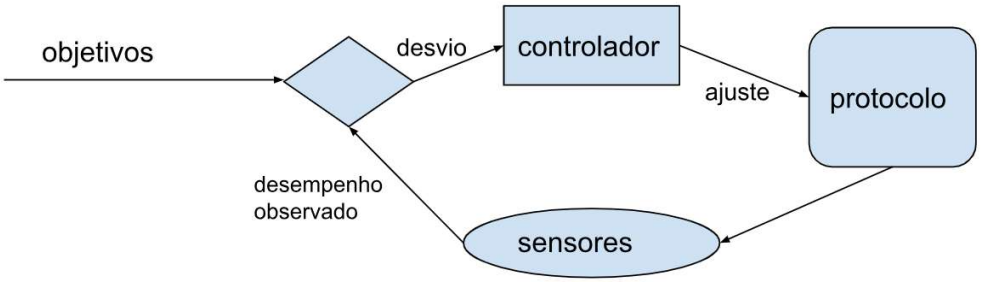


Figura 2.3. Protocolo autonômico.

O desempenho do protocolo, observado por meio de variáveis operacionais, denominadas na Figura 2.3 de sensores, é comparado com o desempenho declarado nos objetivos. Havendo desvios entre o desejado e observado, o controlador promove ajustes sobre mecanismos ou variáveis do protocolo que está sendo controlado. Quando os objetivos não mudam durante a execução, temos um protocolo **adaptativo online**; se os objetivos podem ser modificados em tempo de execução, temos um protocolo **adaptativo autonômico.**

2.3. Implementação de Sistemas distribuídos físico-digitais confiáveis e seguros

Esta seção apresenta aspectos básicos relacionados à implementação de sistemas distribuídos autonômicos, considerando a incorporação de habilidades de gestão autonômica em mecanismos e protocolos básicos usados na construção dos sistemas distribuídos. Como mencionado anteriormente, o projeto desses sistemas precisa conciliar requisitos da previsibilidade temporal, da computação distribuída e, por vezes, do sistema embarcado [Macêdo e Farines 2018].

Primeiramente, é discutida uma visão conceitual que ajuda a entender como mecanismos e protocolos básicos usados em sistemas distribuídos podem ser estendidos para incorporar facilidades de gestão autonômica sem, contudo, violar propriedades básicas de correção – esta visão conceitual, será exemplificada com uma discussão breve, usando protocolos e mecanismos básicos para sistemas distribuídos confiáveis, como protocolos de comunicação em grupo e protocolos de replicação. Em seguida, é discutido como o modelo conceitual é traduzido em desenho arquitetural de software que considera a representação de mecanismos e protocolos básicos em um loop autonômico – este desenho é usado para apresentar como políticas e requisitos definidos pelos usuários/aplicações são usados por tais mecanismos e protocolos autonômicos durante o processo de auto adaptação. O desenho arquitetural também é usado para apresentar de forma didática as atividades básicas do processo de auto adaptação, as quais envolvem o sensoriamento do ambiente distribuído e do comportamento dos próprios mecanismos e protocolos, o planejamento e a realização dos ajustes necessários para o atendimento das políticas e requisitos definidos pelos usuários e aplicações. Por fim, são discutidos de forma mais detalhada os desafios e algumas alternativas de implementação das atividades do laço autonômico envolvendo os protocolos e mecanismos básicos dos sistemas distribuídos.

2.3.1. Aspectos Básicos do Projeto de Sistemas Distribuídos Confiáveis

Para garantir o funcionamento correto e contínuo de sistemas distribuídos, mecanismos de tolerância a falhas são fundamentais [Cristian 1991]. A tolerância a falhas em sistemas distribuídos envolve o projeto de mecanismos e estratégias para garantir que o sistema

continue funcionando corretamente, mesmo na presença de falhas, o que envolve, resumidamente:

1. **Redundância:** uma das principais abordagens para alcançar a tolerância a falhas é por meio da redundância. A redundância envolve a replicação de componentes críticos, dados ou serviços em vários nós do sistema distribuído. Ao ter várias cópias do mesmo componente, se uma cópia falhar ou se tornar indisponível, o sistema pode contar com as cópias redundantes para continuar suas operações. A redundância pode ser implementada em vários níveis, como hardware, software ou dados.
2. **Detecção de defeitos de componentes:** sistemas distribuídos tolerantes a falhas utilizam mecanismos de detecção de defeitos para identificar e localizar falhas em seus componentes. Técnicas como monitoramento ou *checksums* são usadas para detectar anomalias ou desvios de comportamentos esperados. Esses mecanismos monitoram constantemente o sistema e seus componentes para identificar quaisquer falhas ou erros potenciais.
3. **Isolamento de falhas:** uma vez que uma falha ou falha é detectada, os sistemas tolerantes a falhas visam isolar o componente com defeito para evitar a propagação de erros e manter a funcionalidade geral do sistema. Técnicas de isolamento envolvem a duplicação de processos ou recursos, onde componentes redundantes assumem as responsabilidades do componente com defeito. Isolar falhas ajuda a garantir que o restante do sistema possa continuar operando sem ser afetado pelo componente com defeito.
4. **Recuperação de falhas:** após isolar uma falha, os sistemas tolerantes a falhas empregam mecanismos de recuperação para restaurar o sistema a um estado funcional. Técnicas de recuperação podem incluir reiniciar componentes falhos, restaurar dados de *backups*, reconfigurar o sistema ou migrar processos para outros nós saudáveis. O objetivo é trazer o sistema de volta a um estado consistente e confiável, minimizando o impacto da falha nas operações do sistema.

Neste texto, daremos atenção especial à detecção de defeitos ou erros (ver Figura 2.1), à comunicação em grupo e à replicação resiliente a falhas bizantinas. A comunicação em grupo, em particular, permite a entrega confiável de mensagens entre os membros de um sistema distribuído. Ao empregar protocolos como *multicast* confiável ou algoritmos de consenso, a comunicação em grupo garante que as mensagens cheguem a todos os destinatários pretendidos, mesmo na presença de falhas ou interrupções de rede. Essa confiabilidade ajuda a manter a consistência e a correção do sistema, o que é vital para a tolerância a falhas.

Entretanto, para cumprirem seus objetivos, tais mecanismos demandam custos computacionais adicionais, em termos de processamento e comunicação (Jalote 1994). Assim, para não comprometerem o funcionamento normal dos sistemas, os mecanismos de tolerância a falhas devem ser adequadamente configurados, de modo a conciliar esses custos adicionais com a expectativa de carga das aplicações, com os requisitos de desempenho especificados e com o grau desejado de confiabilidade [Dai e Wang 1992; Cristian 1991; Veríssimo e Rodrigues 2000].

As atividades de gerenciamento devem ser realizadas para assegurar a eficiência e eficácia destes mecanismos [Hegering et al. 1998]. Estas atividades consistem no monitoramento contínuo dos sistemas e dos mecanismos de tolerância a falhas e, também, na

realização de ajustes ou reconfiguração de parâmetros, quando as condições do ambiente ou os requisitos definidos pelas aplicações divergem daqueles considerados durante o projeto.

Neste contexto, um dos grandes desafios para a implementação e gestão dos mecanismos de tolerância a falhas é suportar as nuances dos ambientes distribuídos abertos (e.g., Internet de Coisas), os quais são constituídos por inúmeros dispositivos heterogêneos, conectados a partir de canais de comunicação com diferentes capacidades e sujeitos a condições de carga variadas. A consequência disto são ambientes caracterizados por incertezas em relação aos tempos de processamento e transmissão das mensagens.

Para lidar com essas incertezas, foram realizados diversos esforços de pesquisa relacionados, dentre outras coisas, à proposição de modelos de sincronia e mecanismos de tolerância a falhas adequados. As pesquisas relacionadas aos modelos de sincronia se preocupavam com a prova de correção dos algoritmos e protocolos face às incertezas temporais dos ambientes considerados -- ver, por exemplo, [Dolev et al. 1987; Lamport e Lynch 1989, Chandra e Toueg 1996; Cristian e Fetzer 1999; Veríssimo e Rodrigues 2000; Veríssimo e Casimiro 2002], entre outros. Em paralelo, muitas pesquisas relacionadas aos mecanismos de tolerância a falhas focaram na implementação de facilidades de adaptação -- ver, por exemplo, [Chockler et al. 1998; Macêdo 2000; Mills et al. 2004; Sivasubramanian et al. 2005; de Sá e Macêdo 2006; Karmakar e Gupta 2007; Macêdo 2008a], entre outros.

Para oferecer um desempenho adequado, estas facilidades de adaptação permitem aos mecanismos de tolerância a falhas ajustar dinamicamente seus parâmetros de configuração ver, por exemplo [Chen et al. 2002; Bertier et al. 2002; Falai e Bondavalli 2005; Rütti et al. 2006; Satzger et al. 2007], entre outros.

2.3.2. Aspectos Básicos do Projeto de Sistemas Físico-Digitais Distribuídos

Na medida em que novas facilidades de processamento e comunicação propiciam o surgimento de ambientes distribuídos físico-digitais com características dinâmicas, novos desafios se apresentam ao projeto e gerenciamento dos mecanismos de tolerância a falhas. Esta dinamicidade é originada de aplicações com características e requisitos dinâmicos, além de plataformas que permitem não apenas a alocação, liberação e migração dinâmica de recursos virtualizados, mas também a definição de composições dinâmicas, em que os componentes são definidos em tempo de execução [Macêdo 2008b] . Casos típicos desses ambientes distribuídos são encontrados, por exemplo, em plataformas de P2P (Peer-to-Peer), de grades computacionais e em ambientes de computação em nuvens [Milojicic et al. 2002; Baker et al. 2002; Rimal et al. 2009].

A dinamicidade acima referida dificulta o projeto e a gestão da maioria dos mecanismos de tolerância a falhas, que geralmente utilizam estimativas a respeito dos recursos e dos requisitos e características das aplicações para definirem adequadamente seus parâmetros operacionais. Os mecanismos adaptativos de tolerância a falhas existentes na literatura não estão, em geral, preparados para lidar com esta dinamicidade. Isto porque não consideram, na adaptação de seus parâmetros, a dinâmica do ambiente e dos requisitos das aplicações. Mais ainda, muitas das suposições usadas por estes mecanismos adaptativos, sobre o comportamento do ambiente, não são apropriadas. Em muitos ambientes, por exemplo, tais comportamentos são difíceis de serem caracterizados a partir de distribuições de probabilidades específicas.

Em sistemas distribuídos físico-digitais, mecanismos de tolerância a falhas devem ser adaptativos autonômicos (ver seção 2.3.3). Isto é, devem suportar a confiabilidade, adaptando

dinamicamente seus parâmetros de configuração a partir das mudanças observadas nas características dos ambientes distribuídos e nos requisitos de qualidade de serviço das aplicações [de Sá 2011]. Para isso, tais mecanismos autonômicos de tolerância a falhas devem considerar os seguintes requisitos de projeto [de Sá 2011]:

- Autogestão: atender aos desafios impostos pelos ambientes distribuídos modernos, a partir da proposição de mecanismos de tolerância a falhas com habilidade de autogerenciamento;
- Suporte aos requisitos dinâmicos de qualidade de serviço: apresentar e avaliar estratégias para implementação desses mecanismos, considerando o atendimento de requisitos dinamicamente definidos pelas aplicações ou usuários;
- Suporte ao legado de soluções existentes: permitir que os mecanismos propostos sejam implementados aproveitando as facilidades dos mecanismos de tolerância a falhas existentes, sem comprometer a correção dos mesmos;
- Reuso: conceber estratégias com baixo acoplamento em relação ao ambiente de execução, podendo atuar em diferentes modelos de sistemas distribuídos;
- Modelagem do comportamento dinâmico: explorar técnicas existentes na literatura que permitam a análise e síntese de modelos para lidar com o comportamento dinâmico do ambiente distribuído, sem comprometer o reuso da solução.

2.3.3. Visão Conceitual do Projeto de Mecanismos Autonômicos de Tolerância a Falhas

A ideia central no projeto é levar o problema da configuração dos parâmetros dos mecanismos de tolerância a falhas para um nível de abstração mais próximo das aplicações, seguindo a abordagem da computação autonômica [Horn 2001]. Isso implica no estabelecimento de novo modelo de alocação de responsabilidades no qual aplicações ou os usuários devem definir, em tempo de execução, às suas demandas ou restrições em termos de requisitos de qualidade de serviço (e.g. tempo de resposta, confiabilidade, percentual de uso de recursos etc.). Os mecanismos autonômicos de tolerância a falhas, por sua vez, devem observar as variações dinâmicas nas características do ambiente e ajustar seus parâmetros operacionais de modo a atender os requisitos dinamicamente definidos.

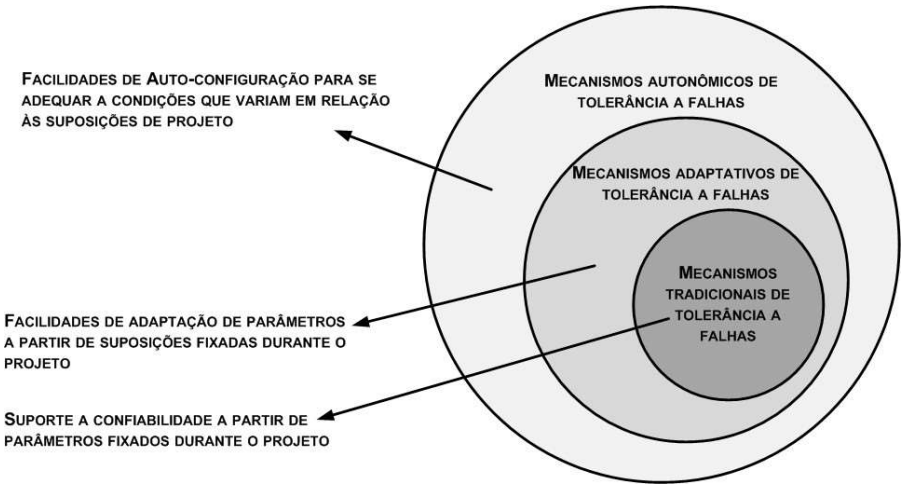


Figura 2.4. Visão conceitual dos mecanismos autonômicos de tolerância a falhas [de Sá 2011].

Conceitualmente, os mecanismos autonômicos para tolerância a falhas podem ser vistos como uma estratégia que provê o suporte à confiabilidade, encapsulando os mecanismos tradicionais ou adaptativos de tolerância a falhas (ver Figura 2.4):

- mecanismos tradicionais de tolerância a falhas se preocupam com o suporte à confiabilidade, garantindo as propriedades de correção de seu serviço, a partir de parâmetros de configuração fixados durante o projeto;
- mecanismos adaptativos de tolerância a falhas encapsulam (ou especializam) as facilidades dos mecanismos tradicionais para permitir a adaptação da configuração, considerando suposições (ou hipóteses) fixadas durante o projeto – e.g., distribuições de probabilidade, arranjo dos componentes, disponibilidade dos recursos etc;
- mecanismos autonômicos de tolerância a falhas encapsulam (ou especializam) as facilidades providas pelos mecanismos adaptativos, de modo a permitir a autoconfiguração quando as características do ambiente ou requisitos das aplicações variam em relação às suposições previamente definidas durante o projeto.

A implementação dos mecanismos autonômicos de tolerância a falhas não é uma tarefa simples, pois requer, por exemplo: modelos adequados para a representação do comportamento dinâmico do ambiente distribuído; a conciliação dinâmica de relações de compromissos associados aos requisitos de qualidade de serviços definidos etc.

2.3.4. Visão autonômica da gestão de tolerância a falhas

A concepção da habilidade de autogerenciamento (i.e., autoconfiguração) requer que os mecanismos de tolerância a falhas implementem um laço de gerenciamento autonômico (Horn, 2001), envolvendo monitoramento, planejamento e intervenções – ver Figura 2.5.

Figura 2.5. Laço de gestão implementado pelos mecanismos autonômicos de tolerância a falhas [de Sá 2011].

O monitoramento envolve a coleta de informações associadas às características dinâmicas do ambiente e à qualidade de serviço entregue pelos mecanismos de tolerância a

falhas. O planejamento envolve a definição de parâmetros operacionais adequados para levar o desempenho do mecanismo de tolerância a falhas a atender os requisitos de qualidade de serviço definidos, considerando o estado atual do ambiente distribuído. As intervenções envolvem a implantação, nos mecanismos de tolerância a falhas, dos parâmetros operacionais definidos durante o planejamento.

Este laço de gestão se assemelha aos laços de controle implementados no campo dos sistemas industriais [Ogata 1995].

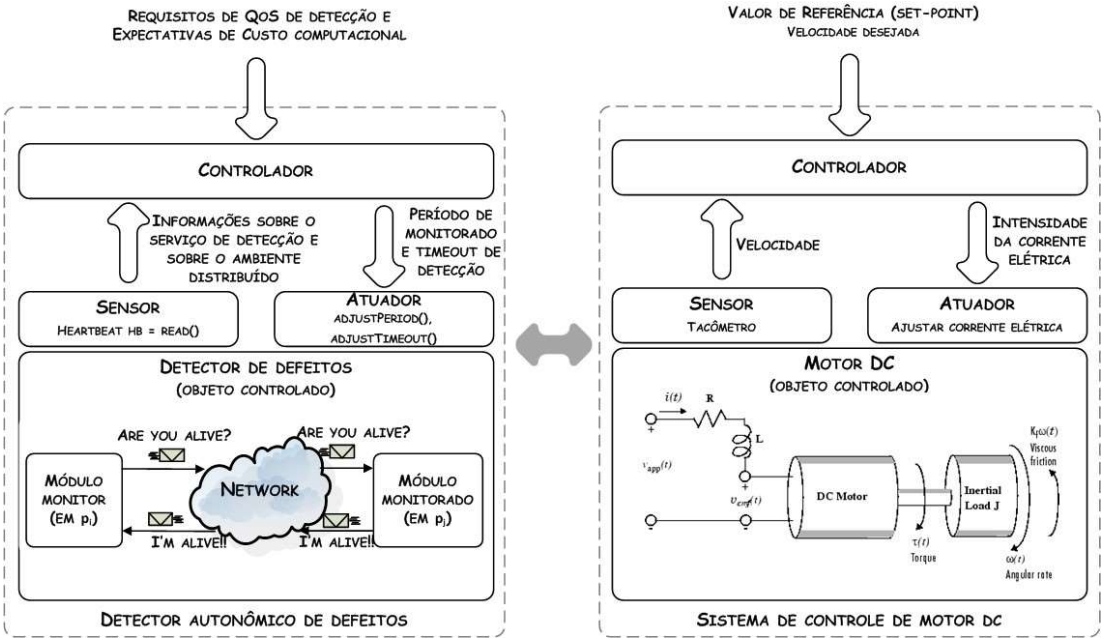


Figura 2.6. Comparativo entre um mecanismo autonômico de detecção de defeitos e um sistema de controle de motor [de Sá 2011].

Como exemplo, pode-se fazer o paralelo entre um mecanismo autonômico de detecção de defeitos e um sistema de controle de motor DC (corrente contínua), ver Figura 2.6.

Nesse exemplo, um tacômetro (sensor) monitora a velocidade do motor (objeto controlado). Em seguida, um controlador verifica se a velocidade observada está de acordo com um valor de referência (*set-point*). Existindo desvios entre as velocidades, se determina um novo valor de corrente elétrica (configuração) para o motor DC, de modo que o mesmo apresente a velocidade desejada. Então, um atuador ajusta a corrente elétrica do motor, usando o valor informado pelo controlador. No caso do detector de defeitos, métodos de leitura (sensor) coletam informações sobre comportamento do ambiente e o desempenho do serviço de detecção (objeto controlado), usando mensagens de monitoramento, por exemplo. Em seguida, o controlador verifica se o desempenho observado está consoante com os requisitos de qualidade de serviço de detecção e as expectativas de custo definidas pelo usuário (i.e., *set-points*). Existindo divergências, o controlador determina os novos parâmetros do detector (e.g. período de monitoramento de falhas e *timeout* de detecção), de modo que o detector apresente o desempenho desejado em termos de qualidade de serviço de detecção e custo computacional. Em seguida, o controlador ativa métodos de escrita (atuador) necessários para que os novos parâmetros de configuração sejam implantados.

Assim, na perspectiva do projeto, o laço de gestão dos mecanismos autonômicos de tolerância a falhas pode ser projetado seguindo a mesma abordagem usada nos sistemas de

controle, isto é: embutindo, nos mecanismos de tolerância a falhas existentes, controladores (ou gestores autonômicos) responsáveis pelo planejamento das configurações, além de métodos de sensoriamento e atuação que realizam a coleta das informações e a implantação dos novos parâmetros de configuração, respectivamente.

Por conta das técnicas de análise e síntese disponibilizadas pela teoria de controle de sistemas dinâmicos [Ogata 1995; Hellerstein et al. 2004], a implementação dos mecanismos autonômicos de tolerância a falhas pode utilizar o mesmo ferramental matemático para a modelagem de suas funções autonômicas.

2.4. Estudos de Caso

Esta seção apresenta estudos de caso nos quais a habilidade de gestão autonômica é implementada em mecanismos e protocolos básicos para a construção de sistemas distribuídos. Nestes estudos de casos, aspectos de implementação da gestão autonômica, como identificação de políticas de alto nível, variáveis a serem monitoradas e parâmetros controláveis em possíveis laços de controle, são apresentados, incluindo o mapeamento entre os requisitos de alto nível e os laços de controle. Também são apresentadas algumas estratégias para avaliação do desempenho destes mecanismos e protocolos autonômicos.

Existe uma quantidade considerável de conhecimento prático e experiência na aplicação de paradigmas e ferramentas em diversas situações e ambientes de execução para projetar e programar sistemas ciberfísicos complexos de forma segura e confiável. Essa *expertise* é ilustrada por diversos exemplos prototípicos e estudos de caso. Nesta seção, apresentaremos brevemente alguns desses estudos de caso e provas de conceito, listados abaixo, a fim de auxiliar os leitores na compreensão dos principais desafios e abordagens de soluções existentes.

2.4.1. Detectores de defeitos

Neste estudo de caso, é apresentado como detectores de defeitos para sistemas distribuídos podem ser especializados para implementar a habilidade de gestão autonômica. Por questões didáticas e de maior simplicidade em termos das propriedades e funcionamento dos detectores de defeitos, este estudo de caso será discutido sem detalhar equações utilizadas nos algoritmos. Uma descrição mais detalhada pode ser encontrada em [de Sá e Macêdo 2010; de Sá 2011]. Neste texto, discutiremos como permitir, em tempo de execução, que os detectores de defeitos autoadaptem seus parâmetros operacionais (i.e., período de monitoramento e *timeout* de detecção de defeitos), considerando métricas de qualidade de serviço de detecção e mudanças nas características do ambiente distribuído.

Detectores de defeitos são elementos básicos para garantia da confiabilidade em sistemas distribuídos, pois, na ocorrência de falhas, permitem a ativação de procedimentos de recuperação ou de reconfiguração do sistema. Em sistemas distribuídos, a detecção de defeitos é realizada a partir da troca de mensagens de monitoramento entre processos do sistema.

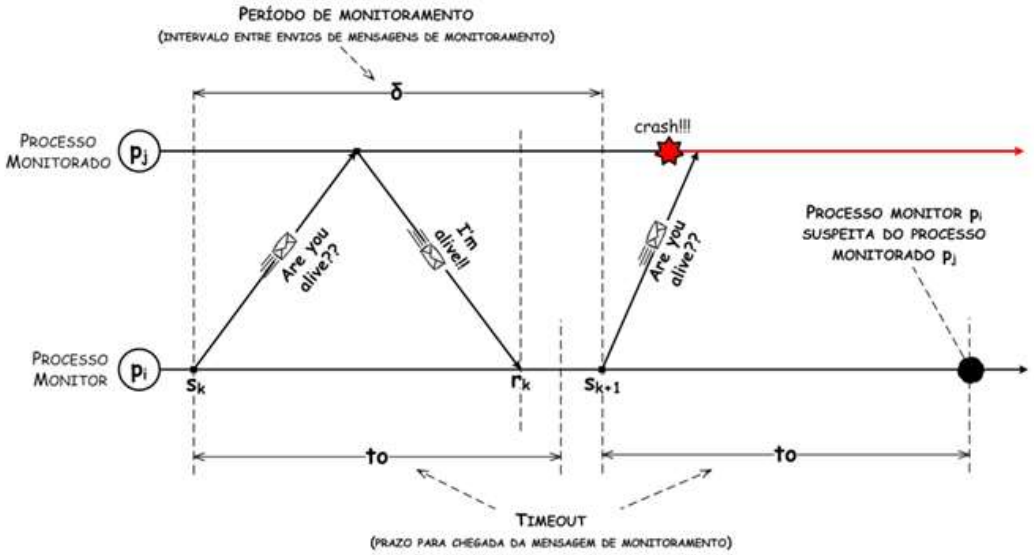


Figura 2.7. - Modelo Pull de Monitoramento de Defeitos

Dentre os modelos de monitoramento existentes, no modelo Pull [Felber 1998] (Figura 2.7), um processo monitor periodicamente envia uma mensagem de monitoramento *"are you alive?"* para o processo monitorado, que deve responder com uma mensagem *"I'm alive!"* – ver Figura 2.7. A cada *período de monitoramento*, caso o processo monitor não receba a mensagem de *"I'm alive!"* dentro de uma determinada janela de tempo (também chamada de *timeout de detecção*), então ele suspeita da falha do processo monitorado.

Para exemplificar a implementação de um detector autonômico de defeitos, este estudo de caso considera a arquitetura proposta em [de Sá e Macêdo, 2010] – ver Figura 2.8. Esse detector de defeitos oferece um nível de autonomia classificado como **Adaptativo Autonômico**.

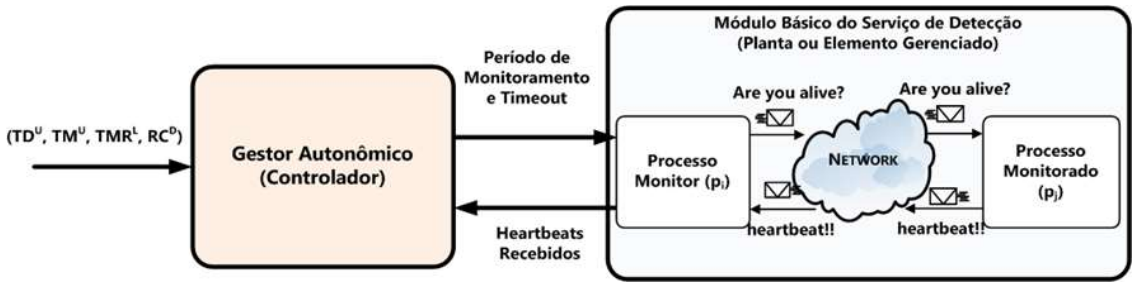


Figura 2.8. Arquitetura do Detector Autonômico de Defeitos [de Sá e Macêdo 2010]

A arquitetura do detector em [de Sá e Macêdo 2010] considera a implementação de um gestor autonômico responsável por realizar, em tempo de execução, a adaptação de parâmetros do detector de defeitos, considerando: os requisitos dinâmicos de aplicação definidos em termos de um Acordo de Nível de Serviço (SLA, Service Level Agreement) de detecção de defeitos, no qual se especifica a velocidade mínima, a confiabilidade mínima e o custo máximo da detecção de defeitos; e a qualidade de serviço (QoS) de detecção de defeitos, na qual se estima a velocidade, a confiabilidade e o custo da detecção.

A QoS de detecção de defeitos depende da definição adequada do *timeout de detecção* e do *período de monitoramento*. Timeouts de detecção muito longos podem tornar a detecção lenta. Por outro lado, timeouts muitos curtos podem levar o detector a apontar falsas suspeitas

de falhas no sistema, como nos casos em que há oscilações nos tempos de processamento ou de comunicação, por exemplo. Períodos de monitoramento muito curtos podem consumir muitos recursos computacionais e provocar ou ampliar congestionamentos nos canais de comunicação entre os processos do sistema. Períodos de monitoramento muito longos podem tornar a detecção muito lenta. Além disso, no caso de perda de mensagens, por exemplo, períodos de monitoramento longos tornam maiores os intervalos de tempos que o detector leva para corrigir falsas suspeitas de falha.

Para atender às relações de compromisso entre velocidade, confiabilidade e custo da detecção de defeitos, o detector autonômico em [de Sá e Macêdo 2010] implementa dois laços de ajuste de parâmetros. O primeiro laço ajusta dinamicamente o timeout de detecção, considerando o desvio entre a confiabilidade mínima e a observada: se a confiabilidade de detecção é superior à mínima, o timeout de detecção é decrementado para aumentar a velocidade de detecção; caso contrário, se a confiabilidade de detecção está abaixo da mínima, o timeout de detecção é incrementado para aumentar a confiabilidade de detecção. O segundo laço ajusta dinamicamente o período de monitoramento, considerando o desvio entre o custo de detecção máximo e o custo de detecção estimado: se o custo de detecção é inferior ao mínimo, então o período de monitoramento é decrementado para aumentar a velocidade de detecção; caso contrário, o período de monitoramento é incrementado para reduzir o custo de detecção. Ambos os laços de adaptação de parâmetros implementados no detector autonômico em [de Sá e Macêdo 2010] restringem os incrementos realizados no período de monitoramento e no timeout de detecção, observando a velocidade de detecção mínima esperada.

Nas abordagens do tipo **adaptativo autonômico** de detecção de defeitos, existem dois desafios significativos na implementação dos laços de adaptação de parâmetros: (a) estimar em tempo de execução a qualidade de serviço do detector de defeitos; e (b) estimar o tamanho do ajuste a ser realizado nos parâmetros do detector de defeitos de modo que o efeito desejado se reflita no desempenho do detector de defeitos.

No caso do detector autonômico em [de Sá e Macêdo 2010], a estimativa da QoS de detecção de defeitos considera a seguinte abordagem:

- **Velocidade de detecção** – obtida a partir da estimativa online do tempo de detecção, a qual é realizada, no início de um ciclo de monitoramento, baseado no atraso de interação entre os processos monitor e monitorado e em uma expectativa de pior caso para expiração do timeout de detecção.
- **Confiabilidade da detecção** – estimada a partir de uma análise do número, do intervalo entre ocorrências e da duração das falsas suspeitas de falhas.
- **Custo da detecção** – representa o percentual do uso dos recursos disponíveis e é estimado a partir do atraso na interação entre os processos do sistema, considerando que atrasos de interação maiores refletem potenciais reduções na quantidade de recursos disponíveis.

Na abordagem apresentada em [de Sá e Macêdo 2010], o atraso de interação entre os processos do sistema distribuído diz respeito ao intervalo de tempo necessário para que um processo do sistema receba uma nova mensagem (i.e., informação atualizada) de outro processo do sistema. Esse atraso é influenciado, não apenas pelos períodos de monitoramento, mas também pelos atrasos fim-a-fim e perdas de mensagens nos canais de comunicação, isto é: quanto maiores os atrasos e as perda de mensagens nos canais de comunicação, maiores são

os atrasos na interação entre os processos do sistema; da mesma forma quanto maior for o período de monitoramento, maior também será o atraso na interação entre processos monitor e monitorado.

Por fim, em [de Sá e Macêdo 2010], a estimativa do tamanho do ajuste a ser realizada nos parâmetros do detector autonômico de defeitos é realizada usando estratégias da teoria de controle realimentado [Ogata 1995]. Na abordagem o ajuste realizado é proporcional ao tamanho do desvio entre o valor desejado em termos de SLA e o valor atual em termos de QoS de detecção. O valor do peso dado ao ajuste nos parâmetros é estimado usando estratégias da teoria de controle, mas como o sistema distribuído é dinâmico e com respostas não lineares, esses pesos também são adaptados continuamente quando são percebidas mudanças no comportamento do sistema em termos dos atrasos de interação – ver maiores detalhes em [de Sá e Macêdo 2010].

2.4.2. Comunicação em Grupo

Neste estudo de caso, é apresentado como protocolos de comunicação em grupo podem ser especializados para apresentar a habilidade de gestão autonômica, permitindo que tais protocolos ajustem seus parâmetros de configuração de acordo com a carga de trabalho do ambiente computacional e de modo a se adequar ao nível de consumo de recursos desejado por uma aplicação ou pelo usuário. Este estudo de caso será apresentado principalmente para discutir como a implementação dos loops autonômicos pode ser realizada sem comprometer propriedades de correção do protocolo de comunicação em grupo.

Protocolos de comunicação em grupo se apresentam como uma poderosa abstração para o desenho de aplicações distribuídas tolerantes a falhas em uma variedade de cenários que expressam diferentes modelos de sistemas distribuídos, de sistemas síncronos a sistemas assíncronos. Embora similar em princípios, diferentes especificações e implementações foram propostas para os diferentes modelos de sistema [Birman 1993; Macêdo et al. 1993; Ezhilchelvan et al. 1995; Cristian 1996; Chandra 1996; Chockler et al. 2001; Défago et al. 2004]. De forma resumida, um protocolo de comunicação em grupo provê o serviço de entrega do mesmo conjunto de mensagens a um mesmo grupo de processos que cooperam em uma computação distribuída. Desta forma, um protocolo de comunicação em grupo abstrai, para camadas de aplicação superiores, a complexidade e incerteza do sistema de comunicação subjacente, como falhas ou entrega de mensagens fora de ordem, permitindo, por exemplo, o provimento de replicação ativa de servidores.

O protocolo de comunicação em grupo, por sua vez, utiliza um conjunto de algoritmos básicos, como difusão confiável e consenso. Por exemplo, o problema de determinar a visão deste grupo, ou seja, o conjunto de processos que o compõe, conhecido como *membership*, pode utilizar como componente da solução o consenso distribuído, ou seja, os processos concordam em um mesmo e único conjunto de processos que formam a visão atual deste grupo. Ainda, o problema da entrega ao grupo em uma mesma ordem de um mesmo conjunto de mensagens, dito difusão atômica confiável, é equivalente ao consenso distribuído, uma vez que os processos envolvidos executam uma forma de acordo [Chandra et al. 1996].

Conforme o modelo de sistema que caracteriza o cenário de atuação do protocolo de comunicação em grupo, pode-se determinar a qualidade de serviço que pode ser obtida, em termos de garantias na entrega das mensagens e na visão dos membros do grupo. O protocolo apresentado de [Macêdo e Freitas 2009] usa uma matriz de registro de relógio vetorial denominado *Timed Causal Blocks*, estrutura introduzida em [Macêdo 2007], que em ambiente

síncrono utiliza-se de limites temporais para terminação do protocolo. Por exemplo, na Figura 2.9, ilustra a matriz de blocos causais (*Causal Block Matrix*) em ambiente assíncrono, conforme introduzido em [Macêdo et al. 1993], para um grupo de 6 processos, onde o bloco 2 está completo em face dos processos p_2, p_5, e p_6 haverem enviado mensagens com número 2, e os processos p_1, p_3, e p_4, enviado mensagens com número 3. A completude do bloco 2 permite a entrega das mensagens de número 2 de forma determinística em todos os processos.

	p_1	p_2	p_3	p_4	p_5	p_6
1	+			+		
2		+			+	+
3	+		+	+		
4	+				+	
5		+				

Figura 2.9. Exemplo de Matriz de Blocos para grupo de 6 processos

Em um ambiente síncrono, usando o modelo de *Timed Causal Blocks,* é possível calcular limites temporais para a completude de blocos, onde após a criação de um bloco um processo que ainda não se manifestou em prol desta completude se manifesta em um tempo limite denominado *time silence* (*ts*). Dessa forma, um bloco causal b se torna completo tão logo uma das duas seguintes condições se verifique: (i) bloco b esteja temporalmente completo ou bloco b esteja completo logicamente (usando tempo lógico). Com o monitoramento adequado do nível de qualidade de serviço dos canais de comunicação, o mecanismo base adapta-se a esta informação e pode prover o melhor ajuste, ou seja, se houver garantias temporais, otimizar o desempenho, e, caso contrário, utilizar tais rótulos temporais apenas para suspeita de falhas por parada. Mais detalhes sobre essa abordagem podem ser encontrados em [Macêdo 2007; Macêdo e Freitas 2009; Macêdo e Freitas 2010].

Neste texto, exploraremos cenários dinâmicos de sistemas físico-digitais distribuídos, como apresentados nas seções anteriores. Neste contexto, protocolos de comunicação em grupo devem se auto ajustarem de acordo com mudanças na carga de trabalho ou nos requisitos da aplicação. Este ajuste deve ser realizado por meio da configuração dinâmica dos parâmetros de operação. Por exemplo, suponha um ambiente de computação em nuvem onde um módulo de gerenciamento de níveis de serviço denominado *SLA Manager* (de *Service Level Agreement*) define o nível de serviço negociado para a aplicação distribuída que executa na nuvem. Como uma ilustração, considere o esboço de um gerenciador de recursos da nuvem, *Resource Cloud Manager*, o qual conforme o *SLA* atual, requisita mudanças nos pontos de operação (*setpoint*) para o consumo de recursos dos protocolos subjacentes, como, por exemplo, de um protocolo de comunicação em grupo, o qual é utilizado em todas as réplicas em execução. Uma mudança neste *setpoint* pode implicar no aumento de consumo de recursos, provendo entrega de mensagens mais rápida, ou, ao contrário, reduzir o consumo de recursos para as aplicações associadas, implicando em maior retardo na entrega das mensagens. Para lidar com esses cenários, equipamos o protocolo base de comunicação em grupo para ser autogerenciável, percebendo o nível de serviço requerido para o usuário, em

termos de consumo de recursos, e ajustando os seus parâmetros operacionais de forma dinâmica.

Assumimos um modelo de sistema distribuído parcialmente síncrono, no qual o sistema torna-se síncrono em um tempo arbitrário [Dolev et al. 1987]. Sob estas premissas de modelo, construímos um protocolo de comunicação em grupo básico que assume um mecanismo subjacente que permite inferir qualidade de serviço dos processos do grupo. Desta forma, utilizamos uma matriz de blocos que registra o envio/recepção de uma mensagem e define *timeouts* de completude dos blocos e outras condições de entrega de mensagens.

A proposta autogerenciável estende o protocolo de comunicação em grupo base, de modo a tratar da escolha apropriada do parâmetro de configuração do protocolo, no caso, o tempo máximo sem transmitir mensagens em um grupo, denominado de *time-silence*. Escolher o valor apropriado do *time-silence* implica em uma relação de compromisso: a) períodos longos para *time-silence* diminuem o *overhead* do protocolo e o consumo de recursos, contudo, podem determinar um tempo de bloqueio maior na entrega de mensagens, pois caso um processo não envie mensagem de aplicação para completude dos blocos, este só contribui para a completude após o período dado pelo *time-silence*; b) do outro lado, períodos curtos de *time-silence* podem reduzir o tempo de bloqueio, porém ao custo do aumento do *overhead* e do consumo de recursos, o que pode ser problemático quando o ambiente estiver sujeito a alta carga de trabalho, implicando em aumento do atraso fim-a-fim e tempo de bloqueio na entrega de mensagens.

Utilizamos elementos da teoria de controle realimentado [Ogata 1995; Hellerstein et al. 2004] para o ajuste de parâmetros de protocolos de computação distribuída, de acordo com um dado nível de serviço. A ideia básica do controle realimentado é que um controlador utilize de sensores que observam as saídas atuais de um objeto controlado (ou planta), e executa um algoritmo de controle (ou lei de controle) para definir novos valores de entrada de modo que as saídas sigam o valor de referência (*setpoint*) para o comportamento desejado. Neste caso, a planta controlada é o protocolo de comunicação baseado no mecanismo de *timed causal blocks* [Macêdo 2007]. O sensor coleta e pré-processa dados sobre o ambiente computacional e desempenho do protocolo de comunicação em grupo, como, por exemplo, latência fim-a-fim dos canais de comunicação, *overhead* e carga de trabalho. O controlador utiliza desta informação para ajustar de forma dinâmica o *time-silence* conforme o valor de referência. Este valor de referência é definido em termos do nível de consumo de recursos contratado pela aplicação. A Figura 2.10 apresenta uma visão simplificada do laço de controle.

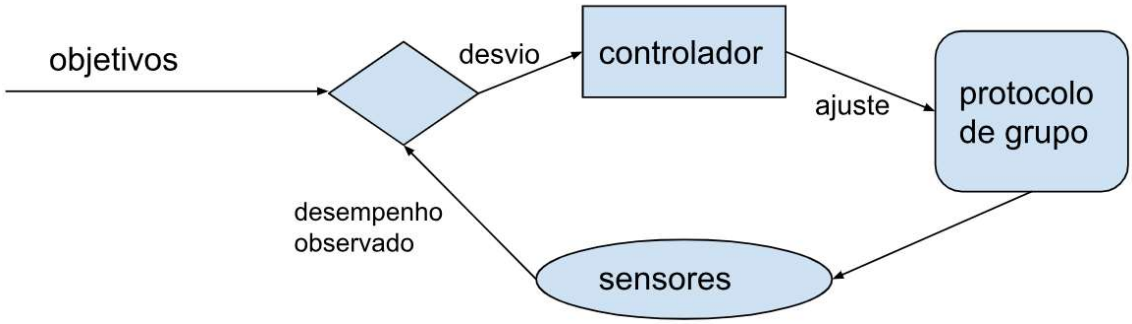


Figura 2.10. Laço de controle para auto configuração do protocolo de comunicação em grupo

Estimamos o consumo de recursos por meio de uma relação que associa as latências do ambiente aos recursos disponíveis em um dado momento: a maior disponibilidade de recursos (como largura de banda, memória, processamento etc.) produz menores latências de comunicação e processamento, e vice-versa.

A partir destes valores observados, e de uma estimativa da latência de comunicação atual do sistema (uma vez que o sistema não é síncrono), construímos a metáfora do consumo de recursos do sistema, utilizando uma relação linear entre a variação da latência observada, dita *jitter* e a quantidade de recursos consumidos do ambiente: se o consumo maior de recursos é alto, há poucos recursos de comunicação e processamento disponíveis, observando valores de latências maiores; já se há pouco consumo, esta maior disponibilidade de recursos reflete em latências observadas baixa.

É importante ressaltar que a relação entre consumo de recursos e latência possivelmente não apresenta linearidade. O escopo deste trabalho não inclui a identificação do sistema de modo a estimar a relação mais adequada, de modo que experimentamos a relação linear para avaliar a eficiência do algoritmo proposto a partir desta premissa.

As condições do ambiente dependem da carga de trabalho do sistema e do tamanho do grupo (número de nós membros). Uma vez que estas condições podem variar em tempo de execução, pode-se dinamicamente calcular referências que expressem: a) requisitos do usuário em termos de consumo de recursos desejado; b) consumo atual de recursos do ambiente; e c) número de membros ativos do grupo.

Assumimos que o *overhead* do protocolo influi no consumo de recursos, de forma proporcional: quanto maior o *overhead* do protocolo, maior o número de mensagens enviadas pelo protocolo, e assim maior o percentual de recursos consumidos. Ou seja, para o protocolo de exemplo quando todos os processos continuamente enviam mensagens ao grupo, não há atuação do mecanismo de *time-silence* e nenhuma mensagem de controle (*overhead* do protocolo); por outro lado, quando apenas um membro do grupo está ativo, os demais $n - 1$ membros contribuem sempre para a completude por meio do *time-silence* – ou seja, situação de maior *overhead* do protocolo.

Neste caso específico, quanto maior o período de *time-silence* é menor a frequência de atuação deste mecanismo e é menor o *overhead* devido às mensagens de controle do protocolo. Assumindo linearidade nesta relação e a relação linear entre *overhead* e consumo de recursos, podemos associar a um dado consumo de recursos desejado um valor de *overhead* e, por meio de uma lei de controle, o ajuste necessário no período de atuação do *time-silence*.

Para isto, temos um componente sensor que percebe as latências atuais da rede e converte estas métricas mensuradas (transdução), estimando: i) o valor atual e máximo do *overhead* do protocolo; ii) os valores máximo, mínimo e a média das latências de comunicação fim-a-fim, de forma adaptativa de modo a obter amostras mais significativas da dinâmica atual; iii) o consumo de recursos atual; e iv) carga de trabalho.

O controlador mapeia o percentual de recursos disponíveis a serem alocados (diferença entre a quantidade de recursos desejada e a quantidade de recursos consumidos) em um *setpoint* dinâmico do *overhead* do protocolo. A escolha da quantidade de recursos desejada é uma decisão da aplicação usuária, conforme o cenário de execução do protocolo. Uma lei de controle proporcional define o período do *timesilence* em função da diferença entre o *overhead* atual e o desejado: se o valor de *overhead* atual é menor que o desejado, então

reduz-se o período com mais mensagens de controle, consumindo mais recursos da rede e diminuindo o tempo de bloqueio; caso contrário, se o valor atual de *overhead* é maior que o desejado, então o período é incrementado, diminuindo as mensagens de controle e o consumo de recursos da rede e aumentando o tempo de bloqueio.

Em comparação com o protocolo base, observamos em experimentos melhores resultados do protocolo de grupo autogerenciável. Mesmo em cenários dinâmicos e sob falhas, a versão autogerenciável mantém o ponto de operação para o *overhead* desejado, inclusive na mudança de carga para diferentes tamanhos de grupo. Detalhes da implementação desta abordagem e da avaliação de desempenho comparativa com o protocolo base podem ser encontrados em [Macêdo et al. 2013].

2.4.3. Replicação Bizantina

Neste estudo de caso, discutimos a implementação da habilidade de autogestão em um protocolo clássico de replicação tolerante a falhas bizantinas, o PBFT [Castro e Liskov 2002], permitindo que o mesmo se ajuste automaticamente de acordo com o monitoramento do ambiente ao longo da execução. Este estudo de caso também será discutido de forma simplificada e tem o papel principal de exemplificar a implementação de um mecanismo semi-autonômico que realiza adaptação online, mas não permite que as políticas de adaptação sejam mudadas em tempo de execução.

Replicação é um mecanismo utilizado para elevar o nível de serviço de serviços computacionais distribuídos, baseados no paradigma cliente/servidor, face à possibilidade de falhas do servidor. Uma técnica utilizada para replicação é a máquina de estados replicada [Schneider 1990] em que cada uma das N máquinas que replicam o serviço implementam uma máquina de estados completa do serviço computacional, ou seja, o conjunto de estados internos que caracterizam o serviço provido e as transições realizadas de um estado a outro na ocorrência de uma dada operação. De modo a reproduzir o mesmo comportamento, todos os servidores replicados iniciam em um mesmo estado e as máquinas de estados são determinísticas, ou seja: em face de uma dada operação em um dado estado, sempre executam a mesma transição de estados. O conjunto de servidores deve concordar, ou chegar ao consenso, em relação à ordem de execução das requisições dos clientes, e baseado nestas premissas, todos servidores devem executar o mesmo conjunto de operações e apresentar o mesmo conjunto de resultados.

Cenários sujeitos somente a falhas por parada são menos desafiadores, mas cenários em que as máquinas podem falhar por comportamento arbitrário, devido a falhas latentes ou transientes, ou mesmo por intrusão de forma maliciosa, requerem uma abordagem mais cuidadosa. Em 1982, o clássico problema dos Generais Bizantinos [Lamport et al. 1982] apresentou as condições para o consenso distribuído diante de falhas arbitrárias, a partir de então denominadas falhas bizantinas. Neste mesmo artigo foi provado que o consenso bizantino precisa de $N = 3f + 1$ para tolerar o máximo de f processos falhos, em um sistema distribuído síncrono (limites superiores conhecidos para tempos de transmissão de mensagens e execução de passos computacionais).

A partir do trabalho de [Castro e Liskov 2002] sobre um protocolo de replicação tolerante a falhas bizantinas, dito PBFT (*Practical Byzantine Fault Tolerance*), que a replicação bizantina ganhou interesse, por propor um conjunto de otimizações, tais como técnicas de rejuvenescimento de réplicas para recuperação proativa, uso de *batching* para otimizar o processamento de conjunto de requisições, mudanças de visões periódicas para

rotacionar o papel do coordenador, uso de *checkpoint* periódico para *garbage collection*, em um cenário que é naturalmente custoso para implementação. Desta forma, há um conjunto de parâmetros relacionados às técnicas de otimização (como tamanho da janela de *batch*, período de *checkpoint*, período de ressincronização de estado), que devem ser ajustados de forma apropriada para garantir o desempenho desejado.

Conceber estratégias de adaptação de parâmetros para um protocolo de replicação bizantina não é uma tarefa simples, uma vez que a implementação destes protocolos é realizada considerando diferentes aspectos – como, por exemplo, segurança, gestão de memória, gerenciamento de réplicas, sincronização de estados. Estes diferentes aspectos, relacionados ao protocolo de replicação, tornam a análise extremamente complexa e dificultam a obtenção de estratégias adequadas de adaptação de parâmetros operacionais. Nesse sentido, a estratégia de ajuste dinâmico ora apresentada foi concebida abstraindo os detalhes internos de implementação. Para tanto, introduzimos aqui um modelo abstrato no qual os diferentes mecanismos, sub-protocolos e procedimentos do protocolo são vistos como caixas pretas organizadas de forma a compor um pipeline abstrato. Os estágios desse pipeline abstrato são selecionados considerando as etapas de interesse da adaptação de parâmetros. A partir do pipeline abstrato, propomos o aPBFT (*Adaptive PBFT*), uma extensão adaptativa do PBFT, a qual baseia a sua adaptação no ajuste dinâmico do tamanho da janela batch (*BS*, *Batch Size*) e do timeout de batching (*BT*, *Batch Timeout*). A escolha dos parâmetros de adaptação se deve ao fato do mecanismo de batching representar um ponto importante para conciliar a carga das aplicações com os custos computacionais associados ao fluxo de execução do protocolo de replicação.

O aPBFT realiza estimativas a respeito da carga imposta pelos clientes e sobre o desempenho do protocolo básico (PBFT). Para isto, a estratégia proposta implementa um laço de controle que usa informações obtidas a partir dos eventos associados ao envio, recebimento e processamento de mensagens do PBFT e determina os novos parâmetros do *batching*. As subseções a seguir apresentam em maiores detalhes os principais aspectos relacionados à implementação do aPBFT, como apresentado na Figura 2.11.

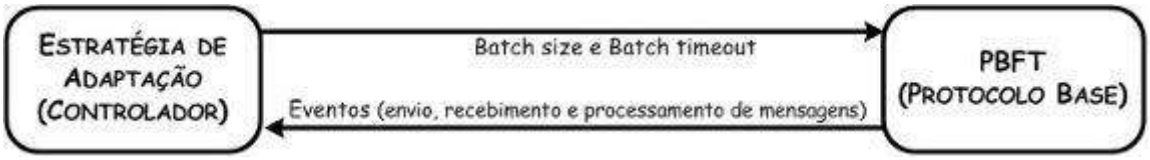


Figura 2.11. Laço de controle para a versão adaptativa do PBFT [de Sá et al. 2013]

O aPBFT abstrai o fluxo de atendimento de requisições do PBFT considerando um *pipeline* abstrato com quatro estágios, denominados: *checking*; *buffering and batching*; *ordering*; e *processing* (ver Figura 2.12).

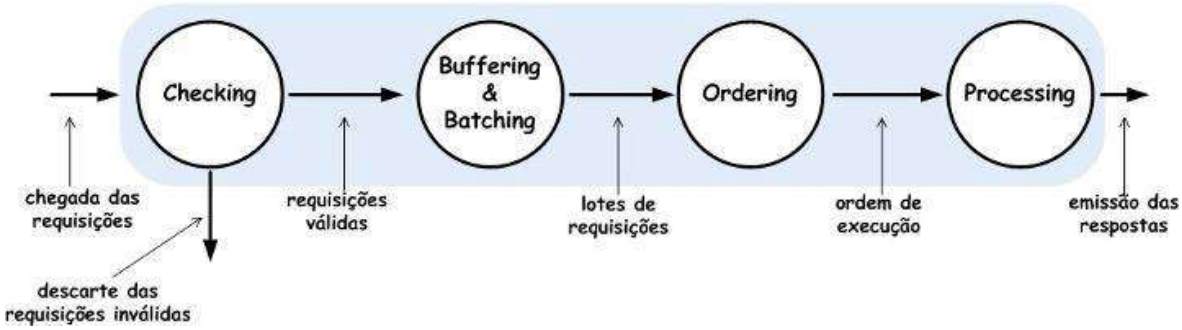


Figura 2.12. Pipeline abstrato representando o processamento de requisições no PBFT [de Sá et al. 2013]

O estágio de *checking* corresponde aos procedimentos realizados pelo PBFT para verificar a integridade, autenticidade e validade das requisições dos clientes. Requisições válidas são repassadas para o mecanismo de *batching*, enquanto que requisições inválidas são descartadas. No estágio de *buffering and batching*, as requisições aceitas são armazenadas em um *buffer*. Quando o número de requisições é suficiente para completar um lote (ou *batch*) de requisições, ou seja, o número de requisições é maior ou igual ao *batch size*, ou quando o *timeout* de *batch* expira, a réplica primária assinala este lote com um número seqüencial e, então, se inicia a fase de *ordering*. No estágio de *ordering*, as réplicas executam o acordo sobre a ordem das requisições. No estágio de *processing*, os lotes efetivados são processados e, então, cada réplica envia as respostas das requisições aos respectivos clientes solicitantes.

Evidentemente, esta descrição baseada no pipeline abstrai uma série de nuances do protocolo de replicação. Durante a fase de ordenação, por exemplo, é possível que várias instâncias do sub-protocolo de acordo sejam executadas em paralelo. Todavia, o uso do pipeline abstrato coloca o protocolo de replicação em um nível de detalhe adequado para a implementação da estratégia de adaptação. A estimativa do *timeout* de *batching* se baseia no intervalo médio de tempo para ordenação e execução das requisições, como apresentado na Figura 2.13. Neste texto, este intervalo de tempo é dito *MTE*, *Mean Time To Ordering and Execute*.

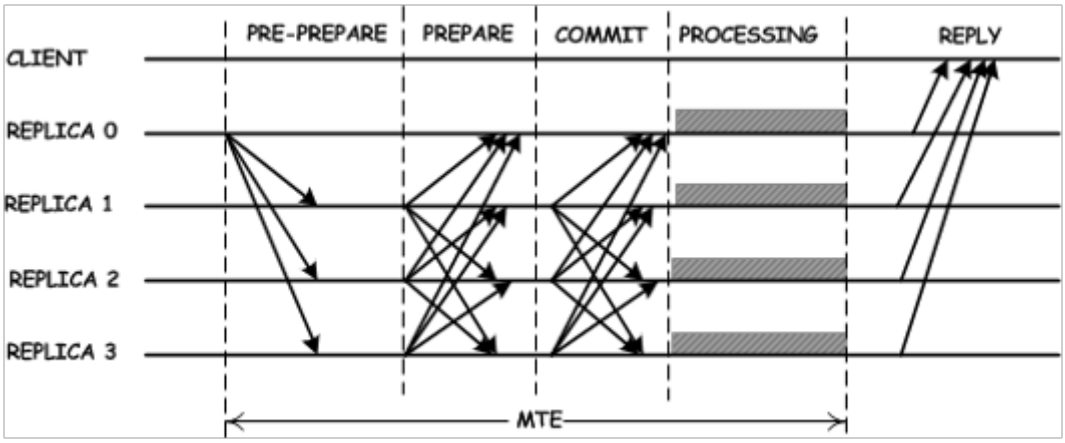


Figura 2.13. Tempo médio para ordenação e execução das requisições (MTE) no PBFT [de Sá et al. 2013].

A ideia central, usada pela estratégia de adaptação, é que o *timeout* de *batching* deve ser longo o suficiente para permitir o acúmulo de requisições para a composição do lote, mas, por outro lado, o mesmo deve ser curto o suficiente para permitir que pelo menos um lote de

requisições esteja no estágio de *ordering* – de modo a manter tal estágio sempre ocupado, na medida em que existem requisições a serem ordenadas e processadas. Entretanto, se o intervalo médio entre chegadas das requisições dos clientes (dito, *MTA – Mean Time Between Arrivals*) é maior que o *MTE*, então o *timeout* de *batching* pode ser zero, uma vez que a carga imposta pelos clientes é muito baixa e os estágios de ordering e processing do pipeline estão operando em um ritmo menor que suas capacidades de execução.

Estimamos o tamanho do *batch* a partir da relação entre o *MTE* e *MTA*. A ideia central para o ajuste é a mesma usada na adaptação do *timeout* de *batching*, isto é, *BS* deve ser definido de tal forma que permita ao menos um lote de requisições esteja nos estágios *ordering* ou *processing*. Assim, se os estágios *ordering* e *processing* levam em média *MTE* unidades de tempo para realizar em conjunto suas atividades, então o lote deve conter o número de requisições que foram possíveis de ser acumuladas neste intervalo de tempo.

O componente sensor percebe os eventos temporais relevantes no pipeline apresentado para realizar o cálculo dos intervalos de *MTE* e *MTA* ora citados. Já o componente controlador realiza os procedimentos de ajuste do *timeout* e do tamanho de *batch*. Para adaptação do *timeout* de *batching*, estima-se a carga das aplicações, e utiliza-se o valor do intervalo médio para ordenação e execução como *timeout* de *batching*. Deve-se observar que se a carga das aplicações é inferior à capacidade dos estágios de *ordering* e *processing*, logo o sistema está ocioso e o *timeout* de *batching* é definido com valor zero (não é realizado lote de requisições). Por fim, adaptamos o tamanho do *batch* usado pelo PBFT: o tamanho do *batch* reflete a relação entre a carga do sistema e o intervalo médio para ordenação e execução. Caso não possamos estimar ainda este intervalo médio por conta, por exemplo, de uma mudança de visão, o valor é definido como um (não é realizado lote de requisições).

Em comparação com o protocolo clássico PBFT, observamos em experimentos melhores resultados do aPBFT, em especial quando há mudanças na dinâmica do cenário (como carga variável), uma vez que o ajuste dinâmico de parâmetros amortiza os efeitos de alta carga de trabalho com maiores valores para o *timeout* e tamanho do *batch* e, ao mesmo tempo, o protocolo aPBFT se ajusta para rápido processamento dos lotes em condições de baixa carga – isto é, aPBFT trabalha como um mecanismo de controle de fluxo que auto-ajusta automaticamente o fluxo da máquina de estados. Detalhes da implementação desta abordagem e da avaliação de desempenho comparativa com o protocolo original podem ser encontrados em [de Sá et al. 2013].

2.4.4. Livro-razão distribuído

Um livro-razão distribuído, dito blockchain, permite o armazenamento confiável e seguro de registros de transação em um conjunto de processos conectados através de uma rede, sem requerem nenhuma entidade centralizada garantidora [Gamage et al. 2020]. Blockchains combinam computação distribuída e segura mantendo uma estrutura de dados dita cadeia de blocos que garante a persistência de transações armazenadas pelos processos que integram o sistema. As blockchains podem ser permissionadas (privadas) e não-permissionadas (públicas).

Uma blockchain permissionada requer que os processos sejam conhecidos a priori e autenticados para participarem do sistema. Um exemplo é a Hyperledger Fabric [Androulaki et al. 2018]. Por outro lado, nas blockchains não-permissionadas qualquer processo pode participar. Os processos não precisam sequer confiar uns nos outros. Exemplos incluem a Bitcoin [Nakamoto 2008] e Ethereum 1.0 [Wood 2014]. Em geral, há uma criptomoeda

associada a blockchain, sendo registrada na plataforma a movimentação destes ativos. Estas blockchains utilizam mecanismos de consenso não-determinísticos em geral, como a "Prova-de-Trabalho" (Proof-of-Work - PoW), a qual valida novos blocos com base na capacidade de processamento dos usuários, o que resulta em alto gasto energético.

Em blockchains permissionadas, há um controle prévio dos membros e o uso de algoritmos clássicos de consenso, como Raft [Ongaro e Ousterhout 2014] e PBFT [Castro e Liskov 2002]. Sistemas amplos não permissionados podem incluir uma quantidade expressiva de nós, mas apresentam uma vazão de transações bem limitada em comparação a sistemas de menor escala baseados em um grupo definido e algoritmos de consenso convencionais.

De maneira generalizada, podemos pensar em blockchain como uma forma de encapsular a máquina replicada de estados baseada em consenso, em que as transações são registradas em blocos e há uma estrutura de dados própria, a dita cadeia de blocos, a qual é resiliente à adulteração. Tal resiliência reside no fato de que cada bloco, desde o primeiro - denominado gênese – consiste em um conjunto de transações bem como do sumário criptográfico (hash) do bloco anterior e do próprio hash do bloco atual, como na Figura 2.14. Adulterar um bloco alteraria seu próprio sumário criptográfico e exigiria alteração dos blocos subsequentes. Os blocos são propostos por um dos nós participantes da blockchain, o qual pode ser denominado líder ou coordenador em blockchains permissionadas baseadas em protocolos clássicos de consenso, ou mesmo minerador, caso de blockchains não permissionadas baseadas em um desafio computacional.

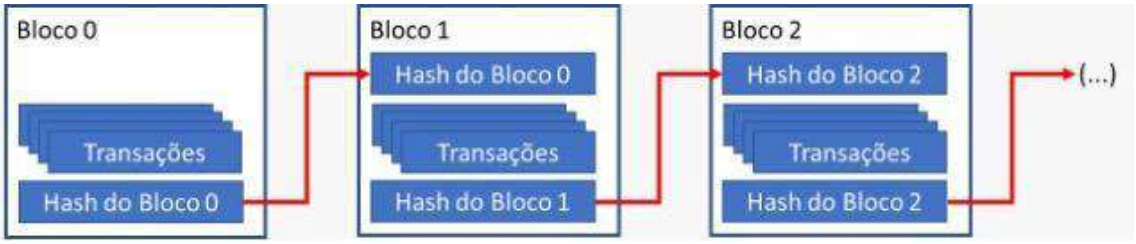


Figura 2.14. Cadeia de blocos em uma blockchain [Freitas et al. 2023].

A proposição de blocos em uma blockchain pública pode envolver mineração, ou seja, há um custo computacional para o proponente do bloco, o qual pode receber como incentivo a criptomoeda associada a blockchain, bem como uma taxa variável por transação.

Nestes cenários, é possível o uso de estruturas ditas *off-chain* que registram uma sequência de chamadas fora da blockchain pública, possibilitando submissão em lote (*batching*) de transações e a redução da taxa paga ao proponente de um novo bloco [Wang et al., 2021]. Uma estrutura *off-chain* pode ser utilizada mesmo para registrar transações até o momento que seja apropriado persistir na cadeia principal, i.e. na blockchain, redes de pagamento, como a *Lighting* [Poon e Dryja 2015], agregam canais de pagamento *off-chain* para registro de transações baseadas em criptomoeda de uma blockchain pública, possibilitando que nós empenhem valores em um canal de pagamento, realizem transações neste canal (e deste interagindo com outros canais de pagamentos pela rede de pagamento) e ao final persistam em uma transação o saldo do conjunto de operações.

Em blockchains públicas, o ponto de decisão para manutenção dos registros *off-chain* ou na blockchain pode utilizar uma política de alto nível, observando, por exemplo, o compromisso entre o nível de latência desejado e o custo associado às transações.

Uma outra abordagem é a percepção de que sendo o consenso o núcleo de operação da Blockchain, o uso de técnicas adaptativas ou autonômicas para o consenso implica na melhoria da operação da blockchain como um todo.

Por exemplo, em [Silva et al., 2023], é demonstrada a relação de compromisso entre diferentes ajustes de tamanho e *timeout* de blocos na plataforma de blockchain Hyperledger Fabric com a capacidade computacional da plataforma e o seu impacto no atraso e vazão do sistema, ou seja, a substituição do protocolo de consenso por uma versão adaptativa (como no caso do PBFT pelo aPBFT explorada na seção anterior), permitiria explorar esta relação de compromisso com ganho ao desempenho da plataforma de blockchain.

Podemos ainda incorporar mecanismos autonômicos à configuração do próprio mecanismo de consenso (e não somente em parâmetros operacionais). Em [Freitas et. al 2023], é proposta a vCubeChain, uma blockchain permissionada que tem por objetivo ser escalável. A vCubeChain é baseada na topologia distribuída hierárquica vCube [Duarte et al. 2014, Ruoso et al. 2014]. A topologia virtual é mantida através de um detector de falhas que forma um hipercubo quando todos os processos estão corretos e que de forma autonômica, na medida em que processos falham por parada, se reorganiza, mantendo diversas propriedades logarítmicas. A vCubeChain é permissionada, i.e. todos os processos são devidamente autenticados. Um líder é eleito utilizando uma estratégia autonômica de difusão confiável para disseminar blocos na rede. Em caso de falsas suspeitas, podem haver múltiplos líderes concorrentes simultaneamente, permitindo a ocorrência de *forks* temporários como em ambientes não permissionados [Nakamoto 2008], mas por meio da cadeia de blocos e de um mecanismo de difusão confiável associado a vCube, a blockchain se mantém íntegra, sempre retornando a um estado consistente de um único líder com a conciliação de divergências.

O entendimento de que a blockchain é uma solução de engenharia que associa ao consenso a estrutura da cadeia de blocos, nos permite pensar em diferentes outras possibilidades a partir da proposição ou da releitura dos algoritmos de consenso, trazendo autogerenciamento a operação destes. Por exemplo, os cenários de blockchain permissionada muitas vezes envolvem máquinas de diferentes organizações, as quais em geral estão dispostas como clusters conectados por meio da Internet. Cada cluster representa um grupo de computadores interconectados por uma rede local, onde pode-se assumir um comportamento síncrono, contudo a nuvem não é em si um sistema síncrono. Este ambiente é representado por um modelo de sistema distribuído híbrido, onde cada cluster local é um sub-sistema síncrono interconectado por canais de comunicação assíncronos com os clusters remotos.

Em um cenário como este, o uso de um consenso adaptativo apresentado em [Gorender, Macêdo e Raynal 2007] permite monitoramento da qualidade de serviço de processos e canais de comunicação, construindo o quórum do consenso adaptado à qualidade de serviço existente no ambiente distribuído e possibilitando o progresso da computação baseado nesta percepção. Ou seja, é possível o progresso mesmo na presença de uma maioria de processos falhos, se as informações do sistema permitem, por exemplo, monitoramento por canais síncronos de todos os processos.

2.4.5 Livro-razão distribuído baseado em reputação

Nesta seção apresentamos como uma blockchain pública pode apresentar maior nível de resiliência por meio de uma abordagem baseada em reputação. Muitos pontos fracos foram associados ao Bitcoin e tecnologias relacionadas, incluindo consistência fraca, baixa taxa de transferência de transações e vulnerabilidade a ataques. Nomeadamente, todas as variantes

contemporâneas baseadas em PoW da Bitcoin dependem da suposição de que um intruso não pode ter mais de 33% ou de 50% do poder de computação a qualquer momento. No entanto, com a sofisticação dos ataques montados no Bitcoin, por exemplo, ataques *flash* (também conhecidos como ataques de suborno), onde um intruso pode obter uma maioria temporária (> 50%) do poder de computação, alugando a capacidade de mineração, todos esses sistemas falham.

Esta seção resume o trabalho apresentado em [Yu et al. 2019], em que são abordadas estas deficiências e apresentadas soluções para as mesmas. Propomos o RepuCoin, o primeiro sistema que pode impedir ataques contra um intruso que pode possuir mais de 50% do poder de computação de toda a rede temporariamente (por exemplo, algumas semanas ou até meses). A prova de conceito mostra que, embora forneça melhores garantias de segurança do que os protocolos anteriores, o RepuCoin também garante uma taxa de transferência muito alta (10.000 transações por segundo - TPS). Na prática, a Visa confirma uma transação em segundos e processa 1.700 TPS em média. Isso mostra que o RepuCoin satisfaz a taxa de transferência necessária de aplicativos do mundo real.

O sistema aborda os desafios acima mencionados definindo um novo princípio de design, chamado prova de reputação. A prova de reputação é baseada na prova de trabalho, mas com duas melhorias fundamentais. Primeiro, sob prova de reputação, o poder de decisão de um minerador (ou seja, o poder de voto para chegar a um consenso no sistema) é dado por sua reputação. A reputação de um minerador não é medida pelo que chamamos de “potência instantânea” do minerador, ou seja, a potência de computação do minerador em um curto intervalo de tempo, como no clássico PoW. Em vez disso, a reputação é calculada com base na quantidade total de trabalho válido que um minerador contribuiu para o sistema e na regularidade desse trabalho, durante todo o tempo durante o qual o sistema esteve ativo. Chamamos isso de "potência integrada" do mineiro. Assim, quando um intruso entra no sistema, mesmo que ele tenha uma capacidade de mineração muito forte, ou seja, alta potência computacional (ou seja, instantânea), ele não teria potência integrada de imediato, ou mesmo logo depois, pois não contribuiu para o sistema antes deste momento de ingresso. Em segundo lugar, quando um minerador se desvia das especificações do sistema, o RepuCoin diminui a reputação do minerador e, portanto, sua potência integrada, em consequência dessa contribuição negativa. Isso evita que um poderoso minerador malicioso ataque o sistema repetidamente sem consequências significativas. Em contraste, os sistemas PoW clássicos não suportam nenhum recurso para punir mineradores que não cumprem as especificações do sistema ou punem esses mineradores simplesmente revogando suas recompensas, ou seja, isto não os impede de atacar o sistema novamente imediatamente depois.

Além disso, o RepuCoin fornece garantias determinísticas em transações, empregando um consenso de voto ponderado baseado em reputação. O consenso é realizado por um grupo formado pelos principais mineradores respeitáveis. Cada membro desse grupo tem um peso associado ao seu voto. O peso do voto de um membro é a porcentagem da reputação desse membro na reputação de todo o grupo. Esses pesos garantem que o poder de voto de alguém dependa não da potência instantânea absoluta (computacional) - que é o facilitador dos ataques de flash - mas da potência integrada, que leva tempo para ser construída e é construída com base na honestidade e no desempenho histórico do minerador.

Na Figura 2.15, apresentamos a análise da segurança fornecida pelos mecanismos usados no RepuCoin em comparativo com outras plataformas de blockchain, mostrando as características determinísticas da taxa de crescimento do poder de decisão, que o fazem resistir a todos os ataques conhecidos à data.

Attacks/Features	BitCoin	BitCoin-NG	ByzCoin	RepuCoin
Double spending attacks	☠	☠	👍	👍
Selfish mining attack	☠	☠	☠	👍
Bribery/flash attack	☠	☠	☠	👍
Eclipse attacks	☠	☠	😐	😐
Non-forkable chain	☠	☠	👍	👍
Liveness	👍	👍	☠	👍
Throughput	7 tps	?	1,000 tps	10,000 tps

Figura 2.15. Análise comparativa de segurança de RepuCoin.

Além disso, mostramos que a rede atinge robustez estocástica muito alta contra os ataques à sua vivacidade ou segurança. Por exemplo, após um único ano de operação, a RepuCoin é resiliente a todos os ataques que comprometem 26%, 33% e 51% dos recursos de computação da rede, mesmo que esse poder permaneça maliciosamente confiscado por quase 100 anos, 2 anos, e 1 ano, respectivamente. Além disso, mesmo que um intruso racional (*for profit*) possa se apoderar de um enorme poder computacional por um período específico (por exemplo, 90% por até 3 meses), ele não quebrará o RepuCoin, devido ao custo de tais ataques tornar a tentativa irracional. Finalmente, fornecemos uma análise de ataques de infiltração irracionais (ex. ciberterrorismo), com uma comparação do custo de atacar diferentes sistemas.

Modelo de Sistema e de Ameaças

RepuCoin é um sistema composto por um número não predeterminado de nós, chamados de mineradores. Cada mineiro tem uma pontuação de reputação, que determina a capacidade desse mineiro de obter recompensas. A pontuação de reputação de um minerador é baseada na correção de seu comportamento e na regularidade em adicionar blocos à cadeia existente, portanto, correlacionada com o poder de computação do minerador. A RepuCoin considera a rede não confiável, e possuindo sincronia parcial. Consideramos um adversário malicioso (também conhecido como Bizantino), que pode atrasar, descartar, reordenar, inserir ou modificar mensagens arbitrariamente. Também consideramos conluios de um número arbitrário de mineradores. No RepuCoin, verificamos e confirmamos microblocos, contidos em *keyblocks* a serem adicionados ao blockchain, usando protocolos de tolerância a faltas Bizantinas com pequenas modificações. Essa forma de acordo evita que um líder mal-intencionado gaste uma moeda em dobro e resolve possíveis bifurcações resultantes de bloqueios de teclas minerados simultaneamente.

Para lidar com as incertezas e ataques acima mencionados o RepuCoin conta com a noção de ter um grupo de consenso, denotado por X, capaz de controlar as operações da RepuCoin, ou seja, executar o protocolo de consenso sobre as transações, orquestrado por um conjunto de mineradores com reputação cumulativa esmagadora. O grupo de consenso pode ser infiltrado por adversários. Por isso, as decisões são tomadas por votação. As regras de votação no consenso não são nominais, mas baseadas em uma votação ponderada inovadora, com base na reputação. Assumimos assim que o adversário consegue controlar no máximo y membros do grupo cuja reputação coletiva é inferior a $1/3$ da reputação cumulativa dos membros do grupo de consenso X. Em consequência, refinamos a definição de quóruns da seguinte forma: chegar a um acordo não requer apenas votos de $2f + 1$ nós, mas também exige

que esses nós tenham coletivamente mais de *2/3* da reputação cumulativa do grupo de consenso. Sob essa suposição, o sistema está seguro e vivo (*safe and live*).

Sistema e função de reputação

Pretendemos definir os objetivos sociais de reputação na RepuCoin de forma precisa e parametrizável, mediante uma função que avalia o andamento da reputação de cada minerador. Esses objetivos são: (i) partida cuidadosa, por meio de um aumento inicial lento; (ii) potencial de recompensa rápida de participantes maduros, por meio de aumento rápido na meia-idade; (iii) prevenção do excesso de controle, por aumento lento próximo ao topo.

No entanto, os mineradores podem ter mau comportamento. Diz-se que um minerador se comporta mal se apresentar mensagens assinadas conflitantes para outros membros do grupo de consenso; ou, se confirmar microblocks com transações conflitantes quando o minerador for eleito líder. Após sua ocorrência, uma evidência de tal mau comportamento é incluída pelos mineradores honestos no blockchain como uma transação especial, e o minerador faltoso perde sua reputação de imediato.

Com base na forte garantia determinística derivada da votação ponderada com base na reputação, a robustez do RepuCoin cresce com o tempo de operação legítimo: quanto mais tarde o intruso entrar, mais seguro será o sistema. Por exemplo, um intruso que ingresse no sistema após um ano de operação, precisaria de pelo menos 51% do poder total de computação e precisaria se comportar corretamente no sistema por 10 meses antes de conseguir fazer com que o RepuCoin perdesse a vitalidade. Quebrar a segurança do RepuCoin é pelo menos tão difícil quanto quebrar sua vivacidade.

2.4.6. Ecossistemas de veículos autônomos (direção de carros autônomos)

Sistemas autônomos cooperativos, como veículos terrestres (carros) em sistemas rodoviários abertos, têm usado ampla tolerância a falhas, por exemplo, em funções x-by-wire, e são bastante seguros do ponto de vista de falhas acidentais. No entanto, eles apresentam desafios a serem tratados devido a falhas maliciosas. Por exemplo, os famosos modelos Tesla FSD foram recentemente sujeitos a vários ataques cibernéticos bem-sucedidos, alguns colocando a segurança em risco.

Essas ameaças incluem todo o ecossistema, desde sistemas e redes veiculares, infraestruturas rodoviárias até redes de comunicação V2V e V2I, a exemplo das vulnerabilidades listadas abaixo:

1. Hackers podem tentar obter acesso não autorizado aos sistemas de controle de veículos autônomos, comprometendo sua segurança e funcionalidade. Nesses casos, são necessárias medidas robustas de cibersegurança, como criptografia, protocolos de comunicação seguros, sistemas de detecção de intrusão e auditorias de segurança regulares para detectar e prevenir ataques cibernéticos;
2. Veículos autônomos dependem fortemente de sensores e sistemas de percepção para coletar informações sobre o ambiente ao redor. Esses sistemas podem ser suscetíveis a erros ou mau funcionamento, levando a interpretação incorreta de dados e situações potencialmente perigosas. Para esses casos, podem ser utilizados sistemas de redundância e backups para sensores críticos;
3. Veículos autônomos coletam grandes quantidades de dados, incluindo geolocalização, padrões de direção e informações pessoais de passageiros. Se esses dados caírem em

mãos erradas, podem ser usados para roubo de identidade, vigilância ou ataques direcionados. Para esses casos, são aplicadas técnicas de criptografia de dados, com consentimento explícito para coleta e uso de dados, para proteger a privacidade do usuário.

Por essas razões, a resiliência cibernética se apresenta como uma facilitadora da operação segura para os sistemas cooperativos autônomos, em particular, os automotivos.

2.4.7. Ecossistemas de veículos autônomos

Quase todos os fabricantes de automóveis experimentam veículos totalmente autônomos e começam a acumular quilômetros em estradas e rodovias para seus *safety cases*, visando aumentar progressivamente o nível de autonomia. Mas não menos importante, a conectividade também vem aumentando, por motivos como infoentretenimento, assistência de trânsito, manutenção remota ou localização de veículos. A este cenário acresce oportunidades de cooperação entre estes veículos, não apenas para melhorar a funcionalidade autônoma, mas também, e principalmente, para a segurança na condução.

Em [Lima et al. 2016], explica-se a problemática dos veículos terrestres autônomos, com ênfase num conjunto de pontos-chave: (a) a necessidade de considerar a autonomia de veículos automóveis como um todo, isto é, um ecossistema completo orientado para veículos autônomos e cooperativos, incluindo os veículos autônomos, as redes e sistemas de periferia (*road-side units, edge web and cloud*), o ambiente não habilitado, como veículos clássicos e pessoas, etc.; (b) a compreensão do aumento dramático da superfície de ameaça global que surge desta nova realidade, com ênfase para a segurança informática; (c) a introdução de um hiato cultural para a indústria, o *safety-security gap*, ameaçando os níveis de *safety* por via de uma compreensão imperfeita dos riscos de *security*.

O principal objetivo desse documento de posição é assim duplo: (i) propor um modelo holístico e genérico bastante, de ecossistema de veículos autônomos e cooperativos; e (ii) realizar uma análise do plano de ameaças, detalhando vetores de ameaças específicos aos quais os designers devem estar atentos, bem como propondo mitigações para os mesmos.

Acresce ainda que a conectividade e cooperação se tornarão a norma, e prevemos que, além dos carros, ela será estendida a, por exemplo, pedestres, bicicletas, etc., formando um universo de objetos 'sencientes', no sentido informático do termo. Nesse sentido, o ecossistema de veículos autônomos e cooperativos pode em breve emergir como uma infraestrutura de informação crítica (virtual) (CII) com grande importância social. Como consequência, a natureza e a intensidade das próprias ameaças também deve evoluir. Além de falhas acidentais e ataques de hackers casuais, devemos esperar ameaças persistentes avançadas e ataques direcionados montados por adversários altamente qualificados e bem equipados, o que reforça a atualidade deste estudo. Começamos por recordar os vários acidentes (alguns mortais) com carros autônomos da Tesla, alegadamente devido a falhas de segurança no sistema de controle da condução autônoma. Mas para além de ataques a carros bem-sucedidos e já demonstrados, comprometendo a comunicação celular e Wi-Fi para desligar o motor, desabilitar freios e trancar as portas, imagine tentativas coordenadas de intrusão em carros autônomos para causar acidentes graves, ou usá-los para bloquear estradas para ações criminosas ou terroristas. Não são infelizmente cenários de ficção os casos relatados no estudo.

A superfície de ameaças do ecossistema de veículos autônomos

No que respeita à conectividade, existem vários níveis de abertura nas redes envolvidas, que vão desde: a Internet, que é pública; as intranets RSU(*road-side unit*), que são privadas, mas expostas geograficamente, exibindo conexões sem fio; as redes V2V e V2I (*vehicle-to-vehicle, vehicle-to-infrastructure*), que são públicas e sem fio; redes veiculares sem fio; e as menos acessíveis, redes a cabo para carros, exigindo acesso físico ou um primeiro salto através da barreira entre o *gateway* de rede sem fio do carro e a infraestrutura de rede veicular restante - ver Figura 2.16. Assumimos que as ameaças podem ser efetuadas por quatro categorias de *agentes de ameaças*: Externo (X): por exemplo, computadores na Internet, RSUs comprometidos, s-Cars (veículos sencientes) hostis ou computadores hostis próximos a s-Cars; Local (L): computadores comprometidos ou hostis dentro de s-Cars, mídia conectada (por exemplo, pen-drives não autorizados). Físico (P): computadores comprometidos ou hostis via tomadas de manutenção, humanos desonestos substituindo ou inserindo hardware; subversão da cadeia de fornecimento. Ambiental (E): dispositivos que interferem nas propriedades do ambiente físico (por exemplo, bloqueadores que perturbam as transmissões sem fio ou dispositivos semelhantes que levam à criação de falsas imagens do sensor); RSUs falsos.

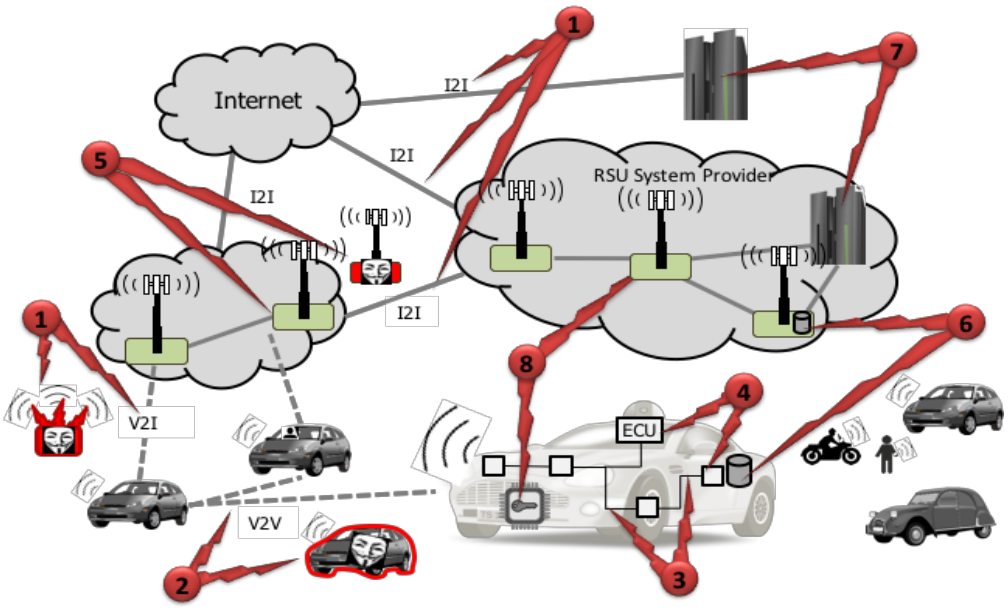


Figura 2.16. Ecossistema de Veículos Autônomos.

Essas ameaças são materializadas por meio de diversos vetores, exemplificados acima e listados abaixo - ver Figura 2.17, com menção aos possíveis agentes em cada vector.

Vector	Description	Agents
#1	Attacks on global V2I/I2I communication infrastructure	X, L, E
#2	Attacks on local V2V communication infrastructure	X, L, E
#3	Attacks on in-vehicle communication infrastructure	L, P
#4	Attacks on vehicle computing nodes' software	L,P
#5	Attacks on road-side units'software	X, P, E
#6	Attacks on sensors and control-sensitive data	X, L, P, E
#7	Attacks on authentication mechanisms	X, L, P
#8	Physical-level attacks	P

Figura 2.17. Vetores de ameaças e agentes correlacionados.

Rumo a Ecossistemas de Veículos Autônomos e Cooperativos Seguros e Protegidos

Nesta secção, resumem-se os trabalho apresentados em [Vöelp e Verissimo 2018; Shoker e Verissimo 2022], em que se tenta dar uma resposta à questão de como a *safety* e a *security* podem ser simultaneamente garantidas por métodos automáticos, fechando o hiato *safety-security*.

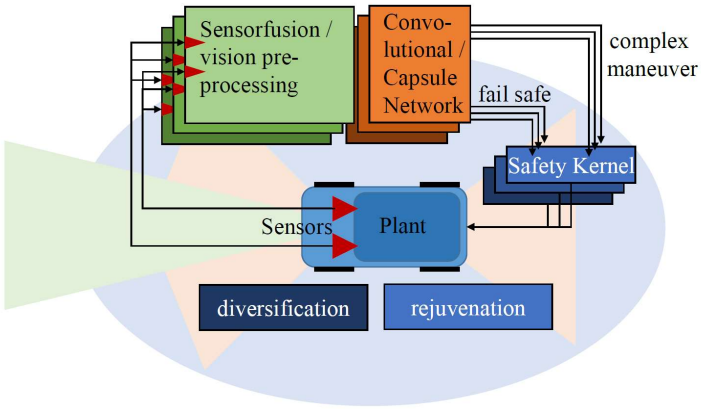


Figura 2.18. Diagrama de fluxo e componentes essenciais da arquitetura [Vöelp e Veríssimo 2018].

A Figura 2.18, do trabalho em [Vöelp e Verissimo 2018], esboça o diagrama de fluxo e os componentes essenciais de nossa arquitetura tolerante a intrusões para condução autônoma. A plataforma é construída em torno do conceito de hibridização arquitetônica. Instanciações prévias do conceito no contexto CPS são, por exemplo, a arquitetura *Simplex* ou a arquitetura *Timely Computing Base*. A arquitetura segue portanto um modelo de falha híbrido, que neste trabalho atual é estendido da *safety* para a *security*. Baseamo-nos na extensão para CPS de trabalhos anteriores em tolerância a intrusões, como a arquitetura *Trusted Timely Computing Base*. Na figura, um controlador simples mas altamente confiável— o kernel de segurança — decide em última análise, caso necessário, acerca da execução correcta das manobras propostas normalmente pelos subsistemas complexos de visão e de IA, de maneira atempada e segura, dependendo das condições de funcionamento em cada momento. A simplicidade exigida pelo kernel de segurança é um artefacto da solução: deriva, de facto, da necessidade óbvia de verificabilidade, de modo a atingir um design ultra correcto e assim justificar a hipótese de falha híbrida — o kernel deve ter uma probabilidade de falha infinitamente menor que a do sistema principal.

Em [Shoker e Verissimo 2022], apresentamos uma evolução e generalização destas aproximações, através do conceito de *Intrusion Resilience Systems* (IRS) para veículos autônomos modernos. O IRS visa contribuir para uma revolução disruptiva nas arquiteturas atuais de computadores e redes de veículos, estendendo as propriedades de segurança e proteção de arquiteturas baseadas em componentes (por exemplo, AUTOSAR). Propomos aproveitar a corrente maturidade do mercado de ECUs, que oferece hoje em dia uma diversidade e independência de oferta, a qual irá permitir propor arquiteturas tolerantes a faltas e intrusões implementadas por SW, de forma semelhante aos princípios estabelecidos pelas arquiteturas pioneiras de TI dos anos 80 em tolerância a faltas distribuída, modular e incremental. Os novos conceitos avançados pelo middleware automotivo IRS, permitem a execução de múltiplas e possivelmente diversas réplicas de um processo de aplicação com estado completo em diferentes ECUs, formando uma Máquina de Estado Replicada

determinística resiliente. Aplicativos distribuídos como fechaduras de portas, controle de janelas, atualizações de software Over-the-Air (OTA) são alguns exemplos de aplicativos viáveis no topo do IRS.

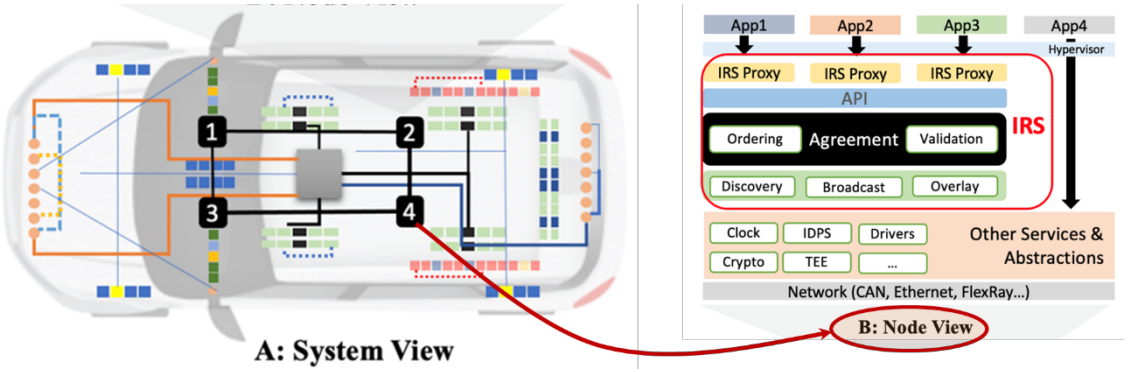


Figura 2.19. Arquitetura do sistema IRS.

Modelo e Arquitetura do Sistema

Apresentamos uma vista da arquitetura do sistema IRS na Figura 2.19. No lado esquerdo a vista global mostra um sistema veicular com um número N de nós IRS ($N = 4$, neste caso) replicados sobre N ECUs. Um nó é composto por um dispositivo de computação, ou seja, uma ECU, uma pilha de software correspondente e um aplicativo veicular de tempo real (crítico) para simplificar. Um nó pode se comunicar com suas contrapartes por meio de mensagens por meio de uma rede veicular, por meio de um link direto, um comutador ou um gateway. Para maior clareza, usamos Unidades de Controle Zonal (ZCU) como ECUs para hospedar diferentes aplicações (por exemplo, fechaduras de portas e controle de janelas) na mesma ECU. Por outro lado, o lado direito da figura apresenta a vista interna de um dos nós (ou seja, o nó 4) descrevendo seus componentes e relação dentro da pilha de *hardware/software* (HW/SW).

Um aplicativo pode falhar ou se comportar de forma arbitrária ou maliciosa quando sujeito a uma intrusão. Assumimos que no máximo uma fração F de N nós pode falhar por vez, e que existe independência de falhas entre os nós. Isso pode ser alcançado empregando ECUs de diversos fornecedores, diferentes bibliotecas, pilha de software e implementação, etc., o que não é incomum no cenário automotivo. O conceito do IRS é baseado na ideia de mascaramento de erro de intrusão em vez de detecção e prevenção como no IPS/IDS. Ao executar várias (N) réplicas/versões de um aplicativo e comparar suas saídas em diferentes nós (ECUs), é possível mascarar qualquer erro causado por falhas acidentais ou maliciosas que ocorrem em nós F com falha, adotando o estado de saída de um não infectado maioria ($N - F$). Nesta abordagem, o estado de uma aplicação crítica só pode ser modificado com a concordância de pelo menos N-F contrapartes.

2.4.8. Redes veiculares autônomas sensíveis ao contexto

As Redes Veiculares Ad-Hoc (VANETs) apresentam como grande motivador a possibilidade de viabilizar o desenvolvimento e execução de aplicações veiculares orientadas para a segurança no trânsito e para o tráfego inteligente. Estas aplicações poderão, por exemplo, gerenciar cruzamentos e semáforos evitando acidentes, assim como controlar o tráfego para

que este tenha um fluxo mais eficiente. Estas possibilidades fazem das VANETs um relevante tópico de pesquisa, com novos resultados sendo continuamente apresentados.

Estas redes apresentam uma topologia dinâmica, sendo construída e modificada a todo momento, devido à alta mobilidade dos veículos. Manter a comunicação nestes casos, e descobrir rotas de comunicação que possam se manter estáveis por mais tempo, é um desafio. Protocolos para VANETs precisam de informações atualizadas sobre o ambiente de comunicação e sobre as próprias aplicações, para ser capaz de prover um serviço de comunicação com o máximo de eficiência possível. Uma maneira de se formalizar estas informações é através do conceito de sistemas cientes de contexto.

Em sistemas cientes do contexto (Context-Aware Systems -- CAS) informações coletadas do ambiente são utilizadas pelo sistema para tomar decisões relativas à sua execução, adaptando seu comportamento a alterações em diferentes aspectos deste ambiente. Desta forma, o sistema estará sempre monitorando e coletando novas informações de contexto para serem utilizadas nas suas decisões [Schilit e Theimer 1994]. O conceito de sistemas cientes do contexto foi inicialmente proposto para sistemas pervasivos e sistemas baseados em sensores, nos anos 90 descrevendo diferentes aspectos e tipos de contexto e seu uso por sistemas.

Pode-se considerar que Contexto é qualquer informação que pode ser usada para caracterizar a situação de entidades ditas relevantes para os sistemas e seus ambientes de execução, incluindo informações dos próprios usuários e das aplicações, assim como informações sobre o estado do meio físico nos quais estes sistemas executam.

O conceito de contexto tem sido utilizado em redes veiculares ad hoc (VANETs) para fornecer diferentes tipos de informações aos sistemas.. Nestas redes, o contexto pode ser relativo a aspectos da mobilidade dos veículos, da comunicação, das mensagens sendo trocadas, das aplicações executadas ou dos próprios usuários e motoristas [Vahdat-Nejad et al. 2016].

Em [Sá e Gorender 2019] apresentamos um contexto de comunicação para VANETs. Este contexto é construído a partir da monitoração de informações de mobilidade e comunicação da rede, tais como a última posição conhecida dos veículos, sua velocidade e direção de sua movimentação. Estas informações são coletadas localmente por cada veículo, através de sensores tais como GPS, velocímetro e acelerômetro, ou são calculadas a partir das informações coletadas. Os veículos trocam suas informações de contexto, utilizando um sistema de monitoração de contexto. Cada veículo mantém o seu contexto sobre o ambiente de comunicação, relacionando suas informações locais com informações de outros veículos.

A partir das informações sendo mantidas, são definidos o contexto de vizinhança e o contexto de comunicação.

O contexto de vizinhança pode assumir um entre dois valores:

- Vizinho: um veículo é considerado ser vizinho do veículo local quando ambos estão dentro dos seus alcances de comunicação sem fio; e
- Não-Vizinho: os veículos estão fora dos seus alcances de comunicação.

Este contexto é definido a partir da distância entre os veículos, calculada a partir da última posição disponível dos veículos, e do alcance das antenas de comunicação.

O contexto de comunicação pode assumir os seguintes valores:

- Comunicável: um veículo remoto é considerável comunicável com relação ao veículo local se são vizinhos, ou se existe uma rota de comunicação entre ambos; e
- Não-comunicável: os veículos não se comunicam.

Assumimos que veículos vizinhos são comunicáveis. Veículos que não são vizinhos podem ser comunicáveis, se for possível construir uma rota de comunicação entre ambos, por meio de nós intermediários, que entre si, em sequência, sejam vizinhos. Para a construção e manutenção destas rotas de comunicação é utilizado um protocolo de roteamento ciente do contexto. Este protocolo atualiza as rotas de comunicação, sob demanda, ao tempo em que o contexto vai sendo atualizado, provendo uma comunicação mais estável e confiável.

O conceito de Qualidade de Contexto (QoC) foi proposto como uma forma de fornecer métricas de qualidade sobre as informações do contexto, para serem também utilizadas pelos sistemas ao tomarem decisões baseadas no contexto [Buchholz et al. 2003]. Este conceito foi também proposto inicialmente para sistemas baseados em sensores, propondo métricas associadas às aplicações e métricas associadas a características dos sensores. Estas métricas podem indicar o quanto uma informação de contexto é acurada, recente e confiável.

Em [Sá e Gorender 2019] e [Sá e Gorender 2021] foram apresentadas métricas de QoC aplicadas ao contexto de comunicação, que podem ser utilizadas pelas aplicações e pelo sistema de comunicação em suas decisões.

A métrica Validity Time (Tempo de Validade) fornece uma janela estimada de comunicação para dois veículos comunicáveis.

A métrica Confidence (Confiança) define um grau estimado de confiança na informação do contexto, considerando o momento em que esta informação foi atualizada e sua idade (Age).

Estas e outras métricas são úteis em definir o quanto o contexto pode ser considerado útil e válido para ser utilizado. Este é um trabalho em andamento no Laboratório de Sistemas Distribuídos (LaSiD) e no Programa de Pós-graduação em Mecatrônica (PPGM), ambos da Universidade Federal da Bahia (UFBA).

2.4.9. Sistemas de Controle de Plantas Elétricas baseados em IoT e Sistemas Ciberfísicos

Nas últimas décadas, as infraestruturas de serviços elétricos tornaram-se amplamente informatizadas, controladas remotamente/automaticamente e interconectadas. Ainda existe a crença de que os sistemas SCADA [Supervisão, Controle e Aquisição de Dados] que controlam essas infraestruturas são legados, fechados e inatacáveis, ou que basta apenas usar um firewall e um detector de intrusão. Esta percepção é inadequada, pelo que é necessário investir em configurações onde a abordagem de prevenção seja complementada com dispositivos de *middleware* que alcancem a segurança automática, através da tolerância a falhas e intrusões. As redes elétricas, se arquitetadas e gerenciadas tendo em mente os mesmos objetivos de segurança e confiabilidade dos sistemas clássicos de tecnologias da informação, podem apresentar níveis muito altos de resiliência.

A resiliência à severidade esperada das ameaças aos sistemas de energia requer mecanismos adicionais que busquem uma operação sustentável e autônoma. Mecanismos de recuperação proativos e reativos para autocorreção são discutidos, bem como mecanismos de monitoramento de confiabilidade que permitem adaptação confiável às situações não previstas ou além das suposições.

O problema de segurança e confiabilidade, ou genericamente falando, resiliência de infraestruturas de *utilities*, isto é, infraestruturas baseadas em sistemas ciberfísicos, não é totalmente compreendido, principalmente devido à composição híbrida destas infraestruturas. O controle do processo das infra-estruturas das concessionárias assenta nos sistemas SCADA (*Supervisory Control and Data Acquisition*) que conferem a capacidade operacional de aquisição de dados, supervisão e controle seja qual for o negócio em causa (eletricidade, água, gás, telecomunicações). No entanto, eles também têm interconexões com as intranets corporativas padrão e, portanto, indiretamente com a Internet. Os sistemas SCADA mencionados acima não foram projetados para serem amplamente distribuídos e acessados remotamente, muito menos para serem abertos. Eles cresceram fechados, sem pensar na *security*. Focando na rede elétrica, as redes de produção, transmissão e distribuição estão hoje em dia largamente automatizadas nos aspectos funcionais, e abertas à rede e à utilização de HW/SW *standard* de mercado (*COTS*), mas carecem de uma aproximação robusta à resiliência, isto é, a garantia combinada de *safety* e *security*.

As perspectivas de danos que podem resultar desta exposição são esmagadoras. Eles vão desde manobras erradas, até ações maliciosas vindas de terminais localizados fora, em algum lugar da Internet. Os alvos dessas ações são unidades de controle de computador, componentes e sistemas embarcados, ou seja, dispositivos conectados ao hardware operacional (por exemplo, bombas e filtros de água, geradores de energia elétrica e proteções de energia, comportas de barragens, etc.). Esta situação foi, por exemplo, analisada sistemicamente no projeto CRUTIAL [Dondossola et al. 2006], onde foram dados vários exemplos de falhas (acidentais ou maliciosas) em sistemas de energia. Essas falhas têm-se repetido ao longo dos anos até recentemente.

As infraestruturas críticas enfrentam um risco visivelmente alto. Possíveis ameaças incluem extorsão, terrorismo e ataques patrocinados pelo governo. Infelizmente, os níveis de vulnerabilidade dessas infraestruturas são altos. Estas infraestruturas críticas combinam computação com processos físicos ou mecânicos controlados eletronicamente por sistemas (SCADA, PCS) que usualmente têm fraca *security*, sendo muitos sistemas legados proprietários com alguma antiguidade. Correntemente, estes encontram-se combinados com sistemas recentes standard de mercado. Em suma, a desculpa comum, “os hackers não conhecem nossos sistemas”, não é mais verdadeira.

Resumimos nesta seção o trabalho apresentado em [Bessani et al., 2008], onde se descrevem os resultados do projeto Europeu CRUTIAL, Critical Utility Infrastructural Resilience, com particular destaque para uma nova arquitetura e protocolos que preservam os sistemas legados e para um novo dispositivo que fornece proteção incremental, garantindo diferentes níveis de resiliência aos diferentes partes da infraestrutura, de acordo com sua criticidade.

A aborgagem CRUTIAL à proteção de infraestruturas críticas ciberfisicas

Um dos pontos cruciais do problema de proteção de infraestrutura crítica é a segurança das interconexões entre provedores de infraestrutura, reguladores, operadores e outros. Para enfrentar esse problema, precisamos de uma arquitetura que nos permita modelar e raciocinar sobre essa realidade. No Crutial, toda a arquitetura da infraestrutura é representada como uma rede WAN-of-LANs, isto é, uma rede pública (WAN) de redes locais (LANs). Este modelo representa bem uma típica infraestrutura de informação crítica, como ilustrado na Figura 2.20.

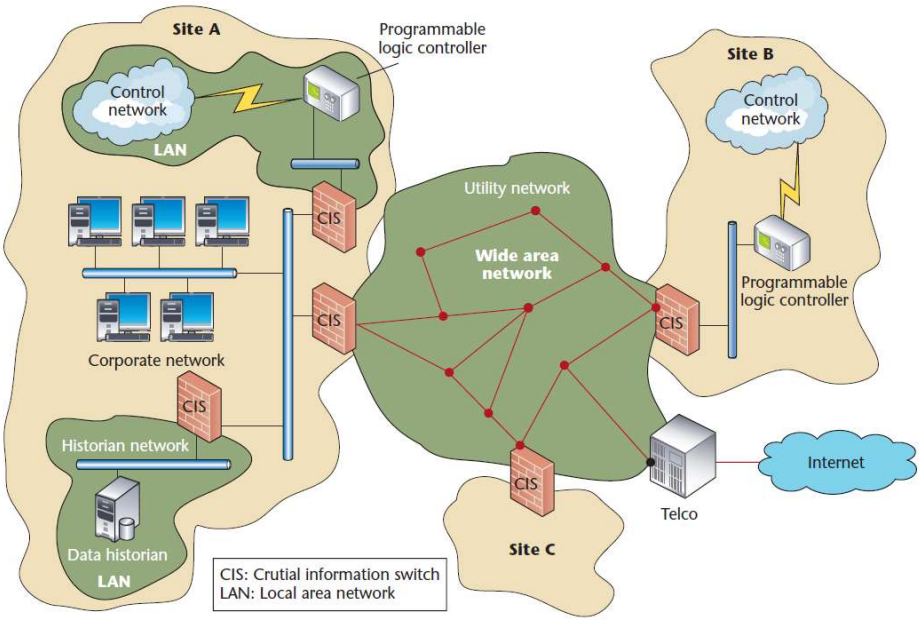


Figura 2.20. Infraestrutura típica de informação crítica.

Essa arquitetura nos permite definir domínios com diferentes níveis de confiabilidade, permitindo defender os domínios uns dos outros — ou seja, uma LAN de outra LAN ou da WAN. Um comutador de informações (CIS) protege cada LAN (ver acima) e fornece dois serviços básicos: (1) o serviço de proteção garante que o tráfego de entrada e saída da LAN satisfaça a política de segurança da organização que gerencia a LAN; (2) o serviço de comunicação que suporta comunicação segura entre os CIS das várias LANs e, em última análise, entre os dispositivos locais e o exterior.

Um CIS é um dispositivo de proteção distribuído com as seguintes características principais: (i) assemelha-se a um *firewall* distribuído porque as organizações podem implantá-lo não apenas na borda da rede, mas também dentro da rede para proteger equipamentos críticos; (ii) interpreta um modelo elaborado de controle de acesso; (iii) é tolerante a intrusões, ou seja, funciona corretamente mesmo que um intruso invada algum de seus componentes, sendo seu objetivo resistir a um alto grau de hostilidade do ambiente. Comparado a outros trabalhos tolerantes a intrusões, o CIS assegura a integridade de funcionamento dos dispositivos legados eventualmente a seu jusante (dentro das LANS) contra aos ataques que possam ser perpetrados por réplicas comprometidas do CIS que não satisfaçam a política de segurança (ver mais adiante). Do ponto de vista funcional, o serviço de proteção CIS funciona como um firewall, interposto entre a WAN e os dispositivos vulneráveis de uma LAN, como ilustra a Figura 2.22.

Designs incrementais da resiliência do CIS

Dados os vários níveis de criticidade de equipamentos de infraestrutura crítica e o custo de usar um dispositivo replicado, vale a pena definir uma hierarquia de projetos de CIS incrementalmente mais resilientes.

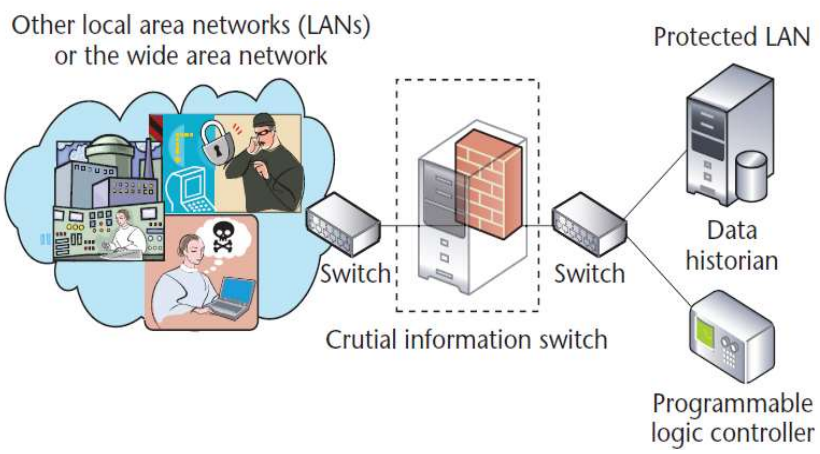


Figura 2.21. Serviço de proteção CIS.

CIS não tolerante a intrusões: a Figura 2.21 sugere um CIS não tolerante a intrusões. É o projeto mais barato porque requer apenas uma máquina, assim como um firewall clássico. Oferece melhor proteção do que os firewalls normais contra ataques de *bypass*, usando a aplicação de políticas no nível do aplicativo e um modelo de controle de acesso avançado. No entanto, é vulnerável a ataques diretos ao próprio CIS.

CIS tolerante a intrusões: o projeto de um CIS tolerante a intrusões consiste em utilizar *2f + 1* máquinas processando os pedidos de tráfego em paralelo, para tolerar intrusões em f réplicas do CIS - ver Figura 2.22. Um pedido chega a todas pela utilização, por exemplo, de um *hub Ethernet*. Os resultados de todas as réplicas são votados, e passa o resultado majoritário, isto é, se e somente se pelo menos f + 1 réplicas votarem a favor: deixar passar a mensagem original (e não uma modificada), ou o bloqueio da mesma. O CIS então transmite uma mensagem aprovada para o destino usando uma réplica distinta, o líder, para que nenhuma multiplicação de tráfego desnecessária ocorra dentro da LAN.

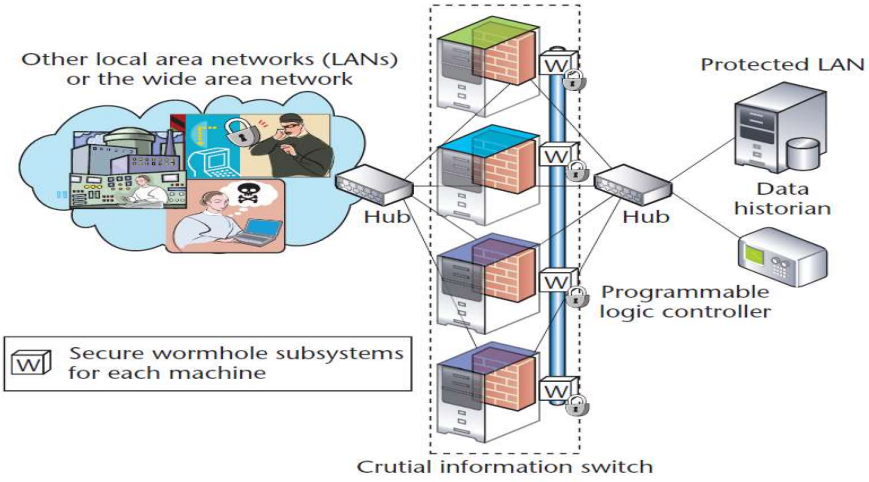


Figura 2.22. CIS tolerante a intrusões.

CIS tolerante a intrusões e autorreparável (self-healing): O mais resiliente projeto de CIS, ilustrado na figura acima, combina tolerância a intrusões com mecanismos de autorreparação para lidar com algumas limitações já descritas. Mecanismos de autorreparação usam um

serviço de recuperação proativo-reativo que combina rejuvenescimentos periódicos acionados por tempo com rejuvenescimentos acionados por eventos quando detectam ou suspeitam de um ataque. Este último mecanismo resolve o problema de ataques de réplicas contaminadas, a dispositivos das LANs. Para isso, a arquitetura híbrida apresentada acima possui uma rede simples de comunicação de controle entre os híbridos de todas as réplicas (nós W, *security kernels*), por onde eles podem trocar alarmes e chegar a consensos simples. Por exemplo, se um intruso comprometer uma réplica do CIS e fizer uma ação detectável (por exemplo, enviar uma mensagem de ataque a um dispositivo frágil da LAN), as outras réplicas (correctas) vão detectar essa acção, trocam mensagens, chegam a consenso sobre a réplica comprometida, e a recuperação reativa da mesma é imediatamente disparada pelo seu híbrido W, que rejuvenesce a réplica. Por outro lado, recuperações proativas são acionadas periodicamente em todas as réplicas, mesmo que não estejam comprometidas. O objetivo é remover os efeitos de ataques maliciosos ou falhas, mesmo que o intruso permaneça inativo (furtivo). Este projeto CIS, como se vê na figura, em vez das clássicas $2f + 1$ réplicas, requer $2f + k + 1$ máquinas para tolerar f intrusões por período de recuperação (ditado pela periodicidade das recuperações proativas). O novo parâmetro k representa o número de réplicas que se recuperam ao mesmo tempo. Seu valor é tipicamente um. Se não existisse k, o CIS poderia ficar indisponível durante as recuperações, ou vulnerável a ataques por exaustão de recursos de redundância.

2.5. Conclusões

Os sistemas distribuídos físico-digitais, também conhecidos como sistemas ciberfísicos, surgiram como resultado da distribuição geográfica e interconexão de componentes físicos por meio de redes de comunicação. Esses sistemas abrangem uma ampla variedade de dispositivos que interagem com o ambiente físico, incluindo sensores, robôs, drones e câmeras, e encontram aplicações em setores diversos, como manufatura, transporte, saúde e segurança. No entanto, a relação físico-digital nessas aplicações apresenta desafios significativos, sobretudo em relação à confiabilidade e segurança dos sistemas envolvidos.

Para enfrentar esses desafios, a gestão autônoma desses sistemas oferece uma solução promissora. Ao configurar os sistemas para monitorar seus próprios estados e tomar decisões com base nos dados coletados, eles podem se adaptar e responder efetivamente a condições em constante mudança. O objetivo deste curso é fornecer uma abordagem teórico-prática abrangente para a construção de sistemas distribuídos autônomos ou autonômicos capazes de atender aos requisitos de desempenho, confiabilidade e segurança dos sistemas físico-digitais distribuídos. Compreender a crescente complexidade desses sistemas e adotar estratégias de gestão adequadas é crucial para garantir sua eficiência e eficácia.

Em conclusão, este curso visa capacitar os estudantes com o conhecimento e as habilidades necessárias para enfrentar as complexidades dos sistemas distribuídos físico-digitais, em especial nos aspectos de confiabilidade e segurança. Por meio da exploração de conceitos teóricos e técnicas de implementação práticas, os estudantes terão a oportunidade de adquirir *insights* valiosos nesses campos. Espera-se que, ao final do curso, os estudantes possam contribuir para a implementação bem-sucedida de sistemas físico-digitais distribuídos em diversos domínios.

Referências

[Androulaki et al. 2018] Androulaki, E. et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proc. 13th EuroSys Conference, EuroSys ’18. ACM. 2018.

[Avizienis et al. 2004] Avizienis, Algirdas et al. Basic concepts and taxonomy of dependable and secure computing. IEEE transactions on dependable and secure computing, v. 1, n. 1, p. 11-33, 2004.

[Baker et al. 2002] Baker, Mark, Rajkumar Buyya, and Domenico Laforenza. Grids and Grid technologies for wide‐area distributed computing. Software: Practice and Experience, v. 32, n. 15, p. 1437-1466, 2002.

[Bertier et al. 2002] Bertier, Marin; Marin, Olivier; Sens, Pierre. Implementation and performance evaluation of an adaptable failure detector. In: Proceedings International Conference on Dependable Systems and Networks. IEEE, 2002. p. 354-363.

[Bessani et al., 2008] Bessani, A; Sousa, P; Correia, M.; Neves, N.; Verissimo, P. The CRUTIAL Way of Critical Infrastructure Protection. IEEE Security and Privacy, vol. 6, no. 6, pp. 44-51, Nov/Dec 2008., Dec. 2008.

[Birman 1993] Birman, K. P. The process group approach to reliable distributed computing. Communications ofthe ACM, ACM, New York, NY, USA, v. 36, n. 12, p. 37–53, December 1993.

[Buchholz et al. 2003] Buchholz, T., Küpper, A. and Schiffers, M. “Quality of context: What it is and why we need it,” in Proceedings of the workshop of the HP OpenView University Association, vol. 2003, 2003.

[Castro e Liskov 2002] Castro, Miguel; Liskov, Barbara. Practical Byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems (TOCS), v. 20, n. 4, p. 398-461, 2002.

[Chandra e Toueg 1996] Chandra, Tushar Deepak; Toueg, Sam. Unreliable failure detectors for reliable distributed systems. Journal of the ACM (JACM), v. 43, n. 2, p. 225-267, 1996.

[Chandra et al. 1996] Chandra, T. D. et al. On the impossibility of group membership. In: Proc. of the 15th annual ACM Symposium on Principles of Distributed Computing. New York, NY, USA: ACM, 1996.

[Chen et al. 2002] Chen, W., Toueg, S., and Aguilera, M. K. On the quality of service of failure detectores. IEEE Trans. On Computer, 51(2):561–580. 2002.

[Chockler et al. 1998] Chockler, Gregory V., Nabil Huleihel, and Danny Dolev. An adaptive totally ordered multicast protocol that tolerates partitions. In: Proceedings of the seventeenth annual ACM symposium on Principles of distributed computing. 1998. p. 237-246.

[Chockler et at. 2001] Chockler, G. V.; Keidar, I.; Vitenberg, R. Group communication specifications: a comprehensive study. ACM Computing Surveys, ACM, New York, NY, USA, v. 33, n. 4, p. 427–469, December 2001.

[Cristian 1991] Cristian, Flavin. Understanding fault-tolerant distributed systems. Communications of the ACM, v. 34, n. 2, p. 56-78, 1991.

[Cristian 1996] Cristian, F. Synchronous and asynchronous group communication. Communications of the ACM, ACM, New York, NY, USA, v. 39, n. 4, p. 88–97, April 1996.

[Cristian e Fetzer 1999] Cristian, Flaviu; Fetzer, Christof. The timed asynchronous distributed system model. IEEE Transactions on Parallel and Distributed systems, v. 10, n. 6, p. 642-657, 1999.

[Dai e Wang 1992] Dai, Shu-Ho; Wang, Ming-O. Reliability analysis in engineering applications. Van Nostrand Reinhold Company, 1992.

[de Sá 2011] de Sá, Alirio Santos. Mecanismos Autonômicos de Tolerância a Falhas para Sistemas Distribuídos. Tese de Doutorado, Universidade Federal da Bahia. 2011.

[de Sá e Macêdo 2006] de Sá, Alirio Santos; Macêdo, R. J. de A. . Avaliando o Impacto de Detectores de Defeitos na Estabilidade de Sistemas de Controle de Tempo Real sobre Redes Convencionais. In: WTF 2006 - VII Workshop de Testes e Tolerância a Falhas, 2006, Curitiba. Anais do WTF 2006. Curitiba: SBC, 2006. v. 1. p. 55-66.

[de Sá e Macêdo 2010] de SÁ, Alirio Santos; Macêdo, R. J. de A.. QoS Self-configuring Failure Detectors for Distributed Systems. In: DAIS. 2010. p. 126-140.

[de Sá et al. 2013] de Sá, Alirio Santos; Freitas, Allan Edgard Silva; Macêdo, Raimundo José de Araújo. Adaptive request batching for byzantine replication. ACM SIGOPS Operating Systems Review, v. 47, n. 1, p. 35-42, 2013.

[Défago et al. 2004] Défago, X.; Schiper, A.; Urbán, P. Total order broadcast and multicast algorithms: Taxo- nomyand survey. ACM Computing Surveys, ACM, New York, NY, USA, v. 36, n. 4, p. 372–421, December 2004.

[Dolev et al. 1987] Dolev, Danny; Dwork, Cynthia; Stockmeyer, Larry. On the minimal synchronism needed for distributed consensus. Journal of the ACM (JACM), v. 34, n. 1, p. 77-97, 1987.

[Dondossola et al. 2006] Dondossola, G; Deconinck, G; Giandomenico, F; Donatelli, S; Kaaniche, M; Verissimo, P. Critical Utility Infrastructure Resilience. Workshop on Security and Networking in Critical Real-Time and Embedded Systems (CRTES'06), @ RTAS'06, April 2006, San Jose, California, USA.

[Duarte et al. 2014] Duarte Jr, E., Bona, L., and Ruoso, V. VCube: A provably scalable distributed diagnosis algorithm. In 5th ScalA Workshop, 2014.

[Dwork et al. 1988] Dwork, Cynthia; Lynch, Nancy; Stockmeyer, Larry. Consensus in the presence of partial synchrony. Journal of the ACM (JACM), v. 35, n. 2, p. 288-323, 1988.

[Ezhilchelvan et al. 1995] Ezhilchelvan, Paul; Macêdo, Raimundo and Shrivastava, Santosh. Newtop: a fault-tolerant group communication protocol. Proceedings of 15th International Conference on Distributed Computing Systems (ICDCS96), Vancouver, BC, Canada, May 1995.

[Falai e Bondavalli 2005] Falai, Lorenzo; Bondavalli, Andrea. Experimental evaluation of the QoS of failure detectors on wide area network. In: 2005 International Conference on Dependable Systems and Networks (DSN'05). IEEE, 2005. p. 624-633.

[Felber 1998] Felber, P. The corba object group service : A service approach to object groups in CORBA. 179 p. Tese (Doutorado), Département D'Informatique, École Polytechnique Fédérale De Lausanne, 1998.

[Fisher et al. 1985] Fischer, Michael J.; Lynch, Nancy A.; Paterson, Michael S. Impossibility of distributed consensus with one faulty process. Journal of the ACM (JACM), v. 32, n. 2, p. 374-382, 1985.

[Freitas et al. 2023] Freitas, Allan Edgard Silva; Rodrigues, Luiz Antonio; Duarte Jr, Elias Procópio. vCubeChain: Uma Blockchain Permissionada Escalável. In: Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. SBC, 2023.

[Gamage et al. 2020] Gamage, H. T. M., Weerasinghe, H., and Dias, N. G. J. A survey on blockchain technology concepts, applications, and issues. SN Computer Science, 1:1–15, 2020.

[Gorender e Macêdo 2002]. Gorender, Sérgio; Macêdo, Raimundo. A dynamically qos adaptable consensus and failure detector. Proceedings of The IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2002 - Fast Abstract Track, pp. B80–B8, Maryland, USA, June 2002. An extended version has been published in the Proceedings of Brazilian Symposium on Computer Networks and Distributed Systems (SBRC2002), Pages 277–292.

[Gorender, Macêdo e Raynal 2005] Gorender, S.; Macêdo, R. J. A.; Raynal, M. A Hybrid and Adaptive Model for Fault-Tolerant Distributed Computing. *Proceedings of IEEE/IFIP Int. Conference on Computer Systemas and Networks (DNS05).p. 412-42.* Yokohama, Japan, June 2005

[Gorender, Macêdo e Raynal 2007] Gorender, S.; Macêdo, R. J. A.; Raynal, M. An adaptive programming model for fault- tolerant distributed computing. IEEE Transactions on Dependable and Secure Computing, v. 4, n. 1, p. 18–31, January 2007.

[Hegering et al. 1998] Hegering, H. G.; Abeck, S.; Neumair, B. Integrated management of networked systems: concepts, architectures, and their operational application. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998.

[Hellerstein et al. 2004] Hellerstein, Joseph L. et al. Feedback control of computing systems. John Wiley & Sons, 2004.

[Horn 2001] Horn, Paul. Autonomic computing: IBM's perspective on the state of information technology. 2001.

[Hurfin et al. 1999] Hurfin, Michel; Macêdo, R; Raynal, M,; Tronel, F. A general framework to solve agreement problems. In: Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems. IEEE, 1999. p. 56-65.

[Jalote 1994] Jalote, Pankaj. Fault tolerance in distributed systems. New Jersey: Prentice Hall, 1994.

[Karmakar e Gupta 2007] Karmakar, Sushanta; Gupta, Arobinda. Adaptive broadcast by distributed protocol switching. In: Proceedings of the 2007 ACM symposium on Applied computing. 2007.

[Kopetz 1997] Kopetz, Hermann. Real-Time Systems: Design Principles for Distributed Embedded Applications. Springer, 1997.

[Kumar et al. 2017] Kumar, S; Bhargava, B; Macêdo, R. J. de A.; Mani, G. Securing IoT-Based Cyber-Physical Human Systems against Collaborative Attacks. 2017 IEEE International Congress on Internet of Things (ICIOT), Honolulu, HI, USA, 2017, pp. 9-16,

[Lamport e Lynch 1989] Lamport, Leslie; Lynch, Nancy. Distributed computing: Models and methods. In: Formal models and semantics. Elsevier, 1990. p. 1157-1199.

[Lamport et al. 1982] Lamport, L.; Shostak, R.; Pease, M. The byzantine generals problem. ACM Transactions on Programming Languages and Systems, v. 4, n. 3, p. 382–401, July 1982.

[Lima et al., 2016] Caldeira Lima, A; Rocha, F; Volp, M; Verissimo, P. Towards Safe and Secure Autonomous and Cooperative Vehicle Ecosystems. In Proceedings of the Second

ACM Workshop on Cyber-Physical Systems Security and PrivaCy, CPS-SPC@CCS (2016, October).

[Macêdo 2000] Macêdo, Raimundo José de Araújo. Failure detection in asynchronous distributed systems. In: Proceedings of the 2nd Workshop on Tests and Fault-Tolerance. pages 76-81. Curitiba, Paraná, Brazil, July, 2000. Available on: <https://sol.sbc.org.br/index.php/wtf/article/view/23478/23305>.

[Macêdo 2007] Macêdo, Raimundo J. de A. An Integrated Group Communication Infrastructure for Hybrid Real-Time Distributed Systems. Proceedings of the 9th Workshop on Real-Time Systems (WTR 2007), p.81-88. Belém, Brazil, 2007.

[Macêdo 2008a] Macêdo, Raimundo J. de A. Adaptive and Dependable Group Communication. Technical Report 001/2008. Distributed Systems Laboratory, UFBA. January 2008. Available on http://www.lasid.ufba.br.

[Macêdo 2008b] Macêdo, R. J. A. Approaches for adaptive and dependable distributed systems. IFIP Workshop on Dependability of Large-Scale and Dynamic Systems (IFIP'2008). Natal, RN, Brazil, February 2008.

[Macêdo 2012] Macêdo, Raimundo J. de A. A Vision on Autonomic Distributed Systems. Proceedings of II Workshop de Sistemas Distribuídos Autonômicos (WoSiDA 2012), pp 31-35, 2012.

[Macêdo 2013] Macêdo, Raimundo J. de A. Designing self-manageable protocols for dependable distributed systems: an experience report. Workshop on Dependability and Fault Tolerance. 64rd Meeting of IFIP Working Group 10.4 - Dependable Computing and Fault Tolerance. Visegrád, Hungary, 2013.

[Macêdo e Farines 2018] Macêdo, Raimundo; Farines, Jean-Marie. Projeto de Sistemas Distribuídos e de Tempo Real para Automação. EDUFBA, 2018. 250 p. ISBN 978-85-232-1675-7.

[Macêdo e Freitas 2009] Macêdo, Raimundo José de Araújo; Freitas, Allan Edgard Silva. A generic group communication approach for hybrid distributed systems. In: Distributed Applications and Interoperable Systems: 9th IFIP WG 6.1 International Conference, DAIS 2009, Lisbon, Portugal, June 9-11, 2009. Proceedings 9. Springer Berlin Heidelberg, 2009.

[Macêdo e Freitas 2010] Macêdo, Raimundo José de Araújo; Freitas, Allan Edgard Silva. Group Communication for Self-Aware Distributed Systems. In: Simposio Brasileiro de Redes de Computadores e Sistemas Distribuidos, 2010, Gramado. Anais do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuidos. Porto Alegre-RS: SBC, 2010.

[Macêdo e Gorender 2008] Macêdo, R. J. de A.; Gorender, Sérgio. Detectores Perfeitos em Sistemas Distribuídos Não Síncronos. In: IX Workshop de Teste e Tolerância a Falhas (WTF 2007), 2008, Rio de Janeiro. Anais do IX Workshop de Teste e Tolerância a Falhas (WTF 2007). Rio de Janeiro: SBC - Sociedade Brasileira de Computação, 2008. v. 1. p. 127-140.

[Macêdo e Gorender 2009] Macêdo, R. J. de A.; Gorender, Sérgio. Perfect Failure Detection in the Partitioned Synchronous Distributed System Model. In: Fourth International Conference on Availability, Reliability and Security (ARES 2009 The International Dependability Conference), 2009, Fukuoka. ARES/CISIS 2009 proceedings. Japan: IEEE Computer Society, 2009.

[Macêdo e Gorender 2012] Macêdo, R. J. de A.; Gorender, Sérgio. Exploiting Partitioned Synchrony to Implement Accurate Failure Detectors. International Journal of Critical Computer-Based Systems (IJCCBS), v. 3, p. 168-186, 2012.

[Macêdo et al. 1993] Macêdo, R. J. de A.; Ezhilchelvan, P; Shrivastava, P., K. Modelling Group Communication using Causal Blocks", 5th European Workshop on Dependable Computing, Lisbon, Feb., 1993.

[Macêdo et al. 2004] Macêdo, R.; Lima, G; Barreto, L; Andrade, A.; Sá, A; Barboza, F; Albuquerque, R.; Andrade, S. Tratando a previsibilidade em sistemas de tempo-real distribuidos: especificação, linguagens, middleware e mecanismos básicos. Capítulo do Livro de mini-cursos do 22o. Simpósio Brasileiro de Redes de Computadores, SBRC'2004.pp. 105-163. Gramado, RS, maio 2004.

[Macêdo et al. 2013] Macêdo, Raimundo José de Araújo; Freitas, Allan Edgard Silva; de Sá, Alírio Santos. Enhancing group communication with self-manageable behavior. Journal of Parallel and Distributed Computing, v. 73, n. 4, p. 420-433, 2013.

[Macêdo, Gorender e Cunha, 2005] Macêdo, Raimundo; Gorender, Sérgio.; Cunha, Paulo. The Implementation of a Distributed System Model for Fault Tolerance with QoS. *In Anais do Simposio Brasileiro de Redes de Computadores (SBRC 2005). p. 827-840,* Fortaleza, May 2005.

[Mills et al. 2004] Mills, K.; Rose, S.; Quirolgico, S.; Britton, M.; Tan, C. An autonomic failure-detection algorithm. In: Proceedings of the 4th International Workshop on Software and Performance. New York, NY, USA: ACM, 2004.

[Milojicic et al. 2002] Milojicic, Dejan S. et al. Peer-to-peer computing. 2002.

[Nakamoto 2008] Nakamoto, S. . Bitcoin: A peer-to-peer electronic cash system. 2008.

[Ogata 1995] Ogata, Katsuhiko et al. Modern control engineering. Upper Saddle River, NJ: Prentice hall, 1995.

[Ongaro e Ousterhout 2014] Ongaro, D. e Ousterhout, J. In search of an understandable consensus algo- rithm. In Proc. of the 2014 USENIX Conference, USENIX ATC'14, USA. USENIX. 2014.

[Poon e Dryja, 2015] Poon, Joseph; Dryja, Thaddeus. The bitcoin lightning network. Scalable o-chain instant payments, 2015.

[Rimal et al. 2009] Rimal, Bhaskar Prasad; Choi, Eunmi; Lumb, Ian. A taxonomy and survey of cloud computing systems. In: 2009 Fifth international joint conference on INC, IMS and IDC. Ieee, 2009. p. 44-51.

[Ruoso et al. 2014] Ruoso, V., Bona, L., and Duarte Jr, E. P. Uma estratégia de testes logarítmica para o algoritmo hi-adsd. InWorkshop de Testes e Tolerância a Falhas, pages 31–44. SBC. 2014.

[Rütti et al. 2006] Rütti, Olivier; Wojciechowski, Paweł T.; Schiper, André. Service interface: a new abstraction for implementing and composing protocols. In: Proceedings of the 2006 ACM symposium on Applied computing. 2006. p. 691-696.

[Sá e Gorender 2019] Sá, Margarete e Gorender, Sérgio. Quality of Context for VANETs: QoC Metrics for Connectivity in VANETs. Proceedings of the 18th International Conference on Ad Hoc Networks and Wireless (AdHoc-Now 2019), Springer, pp 420-431, 2019.

[Sá e Gorender 2021] Sá, Margarete e Gorender, Sérgio. Uma análise da Confiança na Qualidade do Contexto em VANETs. Anais do XI Simpósio Brasileiro de Engenharia de Sistemas Computacionais (SBESC 2021), 2021.

[Satzger et al. 2007] Satzger, Benjamin et al. A new adaptive accrual failure detector for dependable distributed systems. In: Proceedings of the 2007 ACM symposium on Applied computing. 2007. p. 551-555.

[Schilit e Theimer 1994] Schilit, B. N. and Theimer, M. M. Disseminating active map information to mobile hosts. IEEE Network, vol 8 no. 5, pp 22-32, 1994.

[Schneider 1990] Schneider, F. B. Implementing fault-tolerant services using the state machine approach: a tu- torial. ACM Computing Surveys, ACM, New York, NY, USA, v. 22, n. 4, p. 299–319, December 1990. ISSN 0360-0300.

[Shoker e Verissimo, 2022] Shoker, A; Verissimo, P. "Intrusion Resilience Systems for Modern Vehicles - Position Paper". In the 7th *Critical Automotive applications: Robustness & Safety workshop*, CARS@EDCC, September 2022. (See significantly extended version in: Shoker, Rahli, Decouchant, and Esteves-Verissimo. "Intrusion Resilience Systems for Modern Vehicles". In *the 97th IEEE Vehicular Technology Conference: VTC2023,*Florence, Italy, June 2023.)

[Silva et al. 2023] Silva, Francisco A. et al. Avaliação de Desempenho de Blockchains Permissionadas Hyperledger Orientada ao Planejamento de Capacidade de Recursos Computacionais. In: Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. SBC, 2023.

[Sivasubramanian et al. 2005] Sivasubramanian, Swaminathan et al. GlobeDB: Autonomic data replication for web applications. In: Proceedings of the 14th international conference on World Wide Web. 2005. p. 33-42.

[Vahdat-Nejad et al. 2016] Vahdat-Nejad, H., Ramazani, A., Mohammadi, T. and Mansoor, W. “A survey on context-aware vehicular network applications”, Vehicular Communications, vol. 3, pp. 43–57, 2016.

[Veríssimo 2006] Veríssimo, Paulo E. Travelling through wormholes: a new look at distributed systems models. ACM SIGACT News, v. 37, n. 1, p. 66-81, 2006.

[Veríssimo e Casimiro 2002] Veríssimo, Paulo; Casimiro, António. The timely computing base model and architecture. IEEE Transactions on Computers, v. 51, n. 8, p. 916-930, 2002.

[Veríssimo e Rodrigues 2000] Veríssimo, P.; Rodrigues, L. Distributed systems for systems architects. USA: Kluwer Academic Publishers, 2000.

[Verissimo et al. 2009] - Verissimo, P.; Correia, M.; Neves, N.; Sousa, P. “Intrusion-Resilient Middleware Design and Validation”, in *Information Assurance, Security and Privacy Services*, ser. Handbooks in Information Systems. Emerald Group Publishing Limited, May 2009, vol. 4, pp. 615–678.

[Vöelp e Veríssimo 2018] Vöelp, Marcus; Veríssimo, Paulo. Intrusion-tolerant autonomous driving. In: 2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC). IEEE, 2018. p. 130-133.

[Wang et al. 2021] Wang, Yibo et al. iBatch: saving Ethereum fees via secure and cost-effective batching of smart-contract invocations. In: Proceedings of the 29th ACM

Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. 2021.

[Wood 2014] Wood, D. D. Ethereum: A secure decentralised generalised transaction ledger. 2014.

[Yu et al. 2019] - Yu, Jiangshan; Kozhaya, David; Decouchant, Jérémie; Verissimo, Paulo. RepuCoin: Your Reputation is Your Power. In IEEE Transactions on Computers (2019).

Capítulo 3

Princípios e Práticas para Sustentabilidade do Software de Pesquisa

Christina von Flach, Joenio M. Costa e Daniela Feitosa

Abstract

The growing use of Research Software, that is, software developed in the context of scientific research, has awakened in the scientific community a concern with its sustainability and influence on the ability to reproduce scientific studies by independent researchers. Software sustainability relates to the long-term consequences of designing, building and delivering a software project and concerns the software's ability to last and continue to be supported over time. However, researchers are not familiar with good practices for developing sustainable software. This course addresses the sustainability of research software from a technical perspective. We present best practices in software development that can be useful to support researchers from different areas in developing sustainable research software. We motivate and illustrate the use of practices, their benefits and related challenges using the results of a study conducted with a research group in the field of Physics.

Resumo

O uso crescente de Software de Pesquisa, isto é, software desenvolvido no contexto de uma pesquisa científica, tem despertado na comunidade científica uma preocupação com sua sustentabilidade e influência na capacidade de reprodução de estudos científicos por pesquisadores independentes. O conceito de Sustentabilidade de Software está relacionado às consequências de longo prazo de projetar, construir e entregar um projeto de software e diz respeito à capacidade do software de perdurar e ter suporte ao longo do tempo. Entretanto, pesquisadores não estão familiarizados com boas práticas de desenvolvimento de software sustentável. Este curso aborda a sustentabilidade do Software de Pesquisa sob uma perspectiva técnica. Apresentamos boas práticas de desenvolvimento de software que podem ser úteis para apoiar pesquisadores de diferentes áreas no desenvolvimento de software de pesquisa sustentável. Nós motivamos e ilustramos o uso das práticas, seus benefícios e desafios relacionados por meio dos resultados de um estudo conduzido com um grupo de pesquisa da área de Física.

3.1. Introdução

Nas últimas décadas, produtos de *software* têm assumido um papel fundamental no âmbito da pesquisa científica. Na Ciência Aberta [UNESCO 2021], ao lado dos dados de pesquisa, o software desempenha um papel central, seja como parte do método científico ou como um de seus resultados [Bezjak et al. 2018].

Coletar e analisar dados, construir e testar modelos tornaram-se atividades complexas em quase todas as áreas de pesquisa, das ciências exatas às ciências humanas. [Hettrick et al. 2014] relataram que 92% dos cientistas do Reino Unido usavam software em suas pesquisas, 69% declararam que a pesquisa não poderia ser realizada sem software, e 56% desenvolviam seu próprio software, sendo que 21% destes não possuíam treinamento em desenvolvimento de software. A capacidade de lidar com tal complexidade depende de software especializado, chamado de *software científico* [Hannay et al. 2009], *software acadêmico* [Howison and Herbsleb 2011] ou *software de pesquisa*.

O termo *Software de Pesquisa*[1] foi criado para designar, de modo abrangente, o software utilizado durante a pesquisa científica, incluindo o software de terceiros usado para coleta, processamento e análise de dados [Allen et al. 2017]. Recentemente, o uso do termo foi restrito para designar apenas *o software desenvolvido especificamente no contexto da pesquisa* [Nieuwpoort and Katz 2023], com recomendação para que, ao lado dos conjuntos de dados da pesquisa, o código-fonte ou o ambiente de execução do *software de pesquisa* também sejam disponibilizados. Em tal contexto, há uma preocupação legítima por parte da comunidade científica com a qualidade do *software de pesquisa* , em especial, a sua sustentabilidade [Carver et al. 2022]. A falta de sustentabilidade do *software de pesquisa* pode ferir a reprodutibilidade na pesquisa, ou a capacidade de reprodução de estudos científicos por pesquisadores independentes, podendo ocasionar graves erros em conclusões centrais da Ciência [Merali 2010].

O conceito de *sustentabilidade de software* está relacionado às consequências de longo prazo de projetar, construir e entregar um projeto de software [Venters et al. 2014, Venters et al. 2021]. Para o *software de pesquisa*, sustentabilidade diz respeito à capacidade do software de perdurar e de continuar sendo suportado ao longo do tempo, o que implica em qualidades de longevidade e manutenibilidade. O *software de pesquisa* sustentável deve permanecer utilizável por um longo período de tempo e retornar resultados consistentes (mesmo diante de software e hardware em evolução), com benefícios para a comunidade de pesquisa. O *software de pesquisa* sustentável precisa ser atualizado, adaptado para novos ambientes e plataformas e testado.

[Carver et al. 2022] realizaram uma pesquisa com 1.149 pesquisadores para identificar os desafios relacionados a sustentabilidade enfrentados no desenvolvimento e uso de software de pesquisa. Dentre os resultados, a pesquisa identificou que (i) o *software de pesquisa* é desenvolvido e gerenciado sem considerar sua sustentabilidade no longo prazo e, (ii) há necessidade de treinamento para os desenvolvedores de *software de pesquisa* sobre o ciclo de vida do software e ferramentas para seu desenvolvimento e manutenção. De fato, grande parte dos cientistas que escrevem *software de pesquisa* ainda carecem de uma

[1]Tradução nossa para *Research Software*. O termo *Software de Pesquisa* será utilizado ao longo do texto, sempre em itálico.

formação em boas práticas de engenharia de software voltadas para o desenvolvimento de software sustentável, por exemplo, uso de sistemas de controle de versão, documentação adequada e testes automatizados. Neste cenário, a sustentabilidade do *software de pesquisa* pode ficar comprometida, levar a erros em conclusões científicas e à falta de reprodutibilidade na pesquisa.

Além das preocupações com sustentabilidade, e considerando que software é um artefato de pesquisa digital, o princípio da reprodutibilidade requer que, assim como os dados de pesquisa [Wilkinson et al. 2016], o *software de pesquisa* seja *FAIR*, isto é, facilmente localizável, acessível, interoperável e reutilizável [Chue Hong et al. 2022]. Finalmente, é importante considerar que a escolha de um software ou de uma versão do mesmo software pode ter influência nos resultados da pesquisa: as versões do software podem ter funcionalidades parecidas mas com diferenças não documentadas que podem levar as resultados diferentes.

Este capítulo apresenta uma introdução ao *software de pesquisa*, contextualizado no universo da Ciência Aberta, e caracterizado com base em princípios científicos e práticas da engenharia de software, buscando destacar o seu papel na promoção da pesquisa sustentável e reprodutível.

As definições apresentadas sobre Ciência Aberta têm como base a Recomendação da UNESCO sobre Ciência Aberta [UNESCO 2021] e o Manual de Treinamento em Ciência Aberta [Bezjak et al. 2018], disponibilizado sob a licença Creative Commons CC0 1.0 Universal (CC0 1.0) Dedicação ao Domínio Público[2]. As práticas para o desenvolvimento de *software de pesquisa* sustentável apresentadas têm como base parte do material público disponibilizado por *Software Carpentry* [Munk et al. 2019, Nenadic et al. 2022], *Library Carpentry* [Munk et al. 2019], *Netherlands eScience Center* [Drost et al. 2020] e *The Turing Way* [Community 2022]. Finalmente, os exemplos e o material de um estudo realizado com um grupo de pesquisa na área de Física foram criados pelos autores especialmente para este curso e estão disponíveis sob a licença licença Creative Commons CC0 1.0 Universal (CC0 1.0) Dedicação ao Domínio Público.

3.2. O que é Ciência Aberta?

O *software de pesquisa* desempenha um papel central na Ciência Aberta, seja como parte do método científico ou como um de seus resultados [Bezjak et al. 2018]. Nesta seção, apresentamos uma definição para Ciência Aberta e os conceitos mais relevantes para contextualizar o *software de pesquisa*, dentre eles, *Dados Abertos*, *Acesso Aberto*, *Revisão por Pares Aberta*, *Pesquisa Reprodutível Aberta* e *Código Aberto*.

3.2.1. Definição

A Recomendação da UNESCO sobre Ciência Aberta [UNESCO 2021] a define como "um construto inclusivo que combina vários movimentos e práticas que têm o objetivo de disponibilizar abertamente conhecimento científico multilíngue, torná-lo acessível e reutilizável para todos, aumentar as colaborações científicas e o compartilhamento de informações para o benefício da ciência e da sociedade, e abrir os processos de criação,

[2] https://creativecommons.org/publicdomain/zero/1.0/

avaliação e comunicação do conhecimento científico a atores da sociedade, além da comunidade científica tradicional".

Na definição da UNESCO, a Ciência Aberta abrange todas as disciplinas científicas e todos os aspectos das práticas acadêmicas, incluindo ciências básicas e aplicadas, ciências naturais, sociais e humanas, e está estruturada sobre quatro pilares (Figura 3.1): conhecimento científico aberto, infraestrutura científica aberta, envolvimento aberto dos atores sociais e diálogo aberto com outros sistemas de conhecimento.

Figura 3.1. Definição de Ciência Aberta [UNESCO 2021].

No contexto deste capítulo, que trata especificamente sobre princípios e práticas do *software de pesquisa*, colocamos ênfase em conceitos do *conhecimento científico aberto*, que versa sobre o acesso aberto a publicações científicas, dados de pesquisa, software de pesquisa, que estejam disponíveis em domínio público ou sob direitos autorais e licenciados sob uma licença aberta. Conceitos relacionados aos demais pilares estão descritos na Recomendação da UNESCO sobre Ciência Aberta [UNESCO 2021].

Consideramos que o termo *Ciência Aberta* designa a *prática da Ciência* de forma que outros possam *colaborar e contribuir*, onde publicações, dados, software e outros artefatos de pesquisa estão *disponíveis online e gratuitamente*, com base em termos que permitem a reutilização, redistribuição e *reprodução* da pesquisa, seus dados e métodos subjacentes.

3.2.2. Dados Abertos

Dados abertos estão acessíveis e disponíveis online, gratuitamente e podem ser usados, reutilizados e distribuídos desde que a fonte de dados seja atribuída. Pesquisadores podem

depositar os dados de sua pesquisa em um *repositório de dados aberto*, para que outros possam encontrar, utilizar e construir sobre o seu trabalho [Bezjak et al. 2018]. *Metadados* ou informações sobre os dados, tais como, criador, palavras-chave, unidades, são essenciais para facilitar a descoberta de dados e *identificadores permanentes* (PIDs) são necessários para rastrear os dados.

Um *identificador de objeto digital* (DOI) é um identificador exclusivo atribuído a um conjunto de dados para garantir que os dados não sejam perdidos ou identificados incorretamente. O DOI facilita a citação e o rastreamento do impacto dos conjuntos de dados, assim como os artigos de periódicos citados. A *citação de dados* deve ser considerada tão importante quanto a citação de artigos de periódicos e outros documentos científicos. Em geral, uma citação de dados deve incluir nome do autor/criador, data de publicação e título do conjunto de dados, editora/organizador e DOI.

Agências de fomento à pesquisa passaram a exigir que dados coletados ou criados em projetos financiados sejam compartilhados com a comunidade em geral. No Brasil, a Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) lançou as bases para a abertura de dados com a exigência, a partir do final de 2017, de Planos de Gestão de Dados quando da submissão de projetos[3].

Em geral, a organização de dados de pesquisa abertos é temática. O World Ocean Database (WOD)[4] é a maior coleção do mundo de dados abertos de perfis oceânicos uniformemente formatados, com controle de qualidade e disponíveis ao público. Os conjuntos de dados (*datasets*) submetidos recebem um número de acesso (*Accession Number*). Usuários podem utilizar o número de acesso para consultar o Geoportal do *National Centers for Environmental Information (NCEI)* e obter uma cópia exata dos dados originais e seus metadados.

3.2.3. Acesso Aberto

Uma das formas mais comuns de divulgar os resultados de uma pesquisa científica é escrever um artigo e publicá-lo em periódicos, anais de conferências ou livros. Em geral, tais publicações são disponibilizadas ao público mediante pagamento, através de uma assinatura institucional ou de forma individual. Entretanto, nos últimos anos, as taxas praticadas pelas editoras têm sido altíssimas, inviabilizando o amplo acesso ao conhecimento científico e criando desigualdades regionais.

O Movimento de Acesso Aberto[5] surgiu com o propósito de tornar todos os resultados de pesquisa disponíveis para o público sem quaisquer restrições. As duas estratégias mais conhecidas para publicação de artigos científicos promovem modelos alternativos, visando a ampla divulgação e redução de custos.

Publicação em repositórios institucionais ou temáticos

Esta estratégia recomenda o auto-arquivamento por pesquisadores, que depositam e divulgam os seus artigos em repositórios abertos – institucionais ou temáticos.

[3] https://fapesp.br/gestaodedados
[4] https://www.ncei.noaa.gov/products/world-ocean-database
[5] https://en.unesco.org/open-access/open-access-movement

No Brasil, várias universidades oferecem repositórios institucionais para que pesquisadores depositem a sua produção acadêmica. Por exemplo, o Repositório Institucional da Universidade Federal da Bahia (RI-UFBA)[6] é um serviço de informação científica que utiliza o DSpace[7], um pacote de software livre e de código aberto, para gerenciar e a disseminar a produção acadêmica da UFBA em consonância com as recomendações da Ciência Aberta.

Nos últimos anos, pesquisadores têm depositado uma versão de seus artigos pré-impressão (*preprints*) em repositórios abertos temáticos, por exemplo, o arXiv[8]. O *Computing Research Repository (CoRR) - arXiv*[9] hospeda artigos da área de Computação.

Publicação em periódicos e revistas de acesso aberto

Esta estratégia recomenda o desenvolvimento de uma nova geração de periódicos que usam os direitos de autor e outros instrumentos para garantir o acesso aberto permanente a todos os artigos que publicam. Os periódicos de acesso aberto permitem o acesso gratuito aos leitores e autorizam a reutilização da maioria dos seus conteúdos quase sem restrições. Vários periódicos têm sido criados com acesso aberto, por exemplo, a série *PeerJ*. O periódico *PeerJ Computer Science*[10] é o veículo voltado para a área de Computação.

O *Journal of Open Source Software* (JOSS)[11] é um periódico de acesso aberto voltado para a publicação de *pacotes de software de pesquisa*, sem taxas de processamento de artigos ou taxas de assinatura [Smith et al. 2017]. O JOSS possui um processo formal de revisão por pares para melhorar a qualidade do software submetido. Após a aceitação no JOSS, o artigo recebe um DOI e é listado no site do JOSS.

Publicação de Artefatos de Pesquisa

Além dos artigos científicos, repositórios de acesso aberto também são úteis para preservar e compartilhar *outros produtos e artefatos* gerados durante a pesquisa científica, permitindo que outros pesquisadores possam estudá-los e trabalhar com eles. Há diversos repositórios abertos internacionais, dentre eles, Digital Commons, DSpace e Wikimedia Commons [Abdo 2015]. Exemplos de repositórios abertos conhecidos pela comunidade brasileira são Zenodo, Figshare e OSF.

Zenodo[12] é um repositório aberto de uso geral, criado para apoiar os movimentos de acesso aberto e dados abertos na Europa. Pesquisadores podem depositar artigos, conjuntos de dados, *software de pesquisa*, relatórios e quaisquer outros artefatos digitais relacionados à pesquisa. Para cada envio, um identificador de objeto digital persistente (DOI) é criado, o que torna os itens armazenados facilmente citáveis. *Figshare*[13] permite a hospedagem de grandes quantidades de dados que aparecem em artigos online. O *Open*

[6] https://repositorio.ufba.br
[7] https://github.com/DSpace
[8] https://arxiv.org
[9] https://arxiv.org/archive/cs
[10] https://peerj.com/computer-science/
[11] https://joss.theoj.org
[12] https://zenodo.org
[13] https://figshare.com

Science Framework (OSF)[14] é uma ferramenta de código aberto para o gerenciamento de projetos que oferece suporte aos pesquisadores durante todo o ciclo de vida do projeto de pesquisa. É também uma ferramenta de colaboração, que dá suporte ao trabalho em equipe em projetos de forma privada ou torna todo o projeto e seus artefatos publicamente acessíveis para ampla divulgação.

3.2.4. Pesquisa Reprodutível Aberta

A *Pesquisa Reprodutível Aberta* é definida como "o ato de praticar Ciência Aberta e a provisão de oferecer aos usuários acesso gratuito a elementos experimentais para reprodução de pesquisas" [Bezjak et al. 2018]. A Reprodutibilidade é definida como a capacidade de investigadores independentes de tirar as mesmas conclusões de um experimento seguindo a documentação compartilhada pelos pesquisadores que originalmente realizaram a pesquisa. A reprodutibilidade na pesquisa científica aumenta a credibilidade da pesquisa.

Na pesquisa reprodutível aberta, os dados abertos fornecem os insumos necessários para que os pesquisadores validem e reproduzam os resultados uns dos outros. Entretanto, o acesso aos dados de pesquisa e artigos científicos não é mais suficiente para assegurar a reprodutibilidade na pesquisa. É importante definir e documentar os fluxos de trabalho (*workflows*) da pesquisa.

O pesquisador deve pensar em reprodutibilidade desde o início de sua pesquisa, documentando e compartilhando cada etapa do fluxo de trabalho – desde a coleta ou acesso aos dados, filtragem e limpeza, até a análise de dados – de modo a criar um roteiro ou guia para que ele e outros pesquisadores possam seguir. A documentação deve incluir e justificar decisões importantes, por exemplo, excluir certos valores ou ajustar certos parâmetros do modelo. O uso de software de *workflow* permite que cientistas automatizem e repliquem facilmente os fluxos de trabalho em cada etapa de coleta e análise de dados.

A iniciativa *Workflows Community*[15] compartilha recursos fornecidos pela comunidade de plataformas de software de coleta e análise de dados em pesquisa científica. Ferramentas como o Nextflow[16] orquestram, de maneira simplificada e aberta, *pipelines* de dados de maneira portável e escalável, com suporte a instalação em uma variedade de plataformas, incluindo computadores locais, infrastruturas como HPC e Kubernetes, e de Computação em Nuvem como AWS, Azure e Google Cloud. Outras soluções, por exemplo, o *Guix Workflow Language (GWL)*[17] oferecem extensões de apoio a computação científica para a linguagem declarativa de pacotes do GNU Guix, permitindo combinar pacotes Guix com workflows científicos.

Na Ciência Aberta, além dos dados abertos e fluxos de trabalho documentados, o software também é reconhecido como elemento experimental e parte fundamental do método científico.

[14] `https://osf.io`
[15] `https://workflows.community/`
[16] `https://nextflow.io`
[17] `https://guixwl.org`

3.2.5. Código Aberto

Na Ciência Aberta, *Código Aberto* é o termo usado para caracterizar software amplamente acessível para uso, seja como software livre, gratuito ou comercial [Bezjak et al. 2018]. "Aberto" pode ter dois significados diferentes: apenas o código-fonte está disponível em código aberto ou o software é desenvolvido de forma aberta [Hermann and Fehr 2022].

Projetos de Software Livre disponibilizam o código-fonte e são desenvolvidos de forma aberta, com licenças de software atribuídas que permitem ao usuário leitura, execução, adaptação e distribuição. Alguns pesquisadores consideram que o software livre é um pré-requisito para a Ciência Aberta [Flach and Kon 2021, Anzt et al. 2021].

Uma outra forma de compartilhar código aberto são os *notebooks*, fundamentados na Programação Letrada (*Literate Programming*) [Knuth 1984]). Um programa letrado é uma combinação de documentação e programa-fonte organizada de modo que possa ser lida por pessoas. Ferramentas de programação letrada extraem do arquivo uma documentação legível e um programa-fonte compilável. *Notebooks* Jupyter[18] são usados para análise exploratória e preparação de dados, treinamento e teste dos modelos. O código escrito em um *notebook* pode ser extraído e empacotado como um módulo ou biblioteca Python para posterior reutilização.

3.3. Software na Ciência

Na Ciência moderna, os dados de pesquisa (*research data*) dependem, de modo crescente, do ambiente de software em que foram criados, gerenciados, analisados e apresentados. Resultados digitais da pesquisa, como publicações científicas e conjuntos de dados abertos, não podem ser visualizados ou usados sem o software apropriado.

Muitas instituições, grupos e pesquisadores reconhecem que todo o software desenvolvido no contexto de uma pesquisa científica deve ser considerado um resultado de pesquisa [Jay et al. 2021] e que a boa prática científica exige que o software mencionado em publicações científicas seja mantido para reprodutibilidade e verificação de resultados científicos. O Manifesto do Código da Ciência (*Science Code Manifesto*)[19] destaca a importância do software usado ou desenvolvido *durante* a pesquisa científica:

> "Software é um produto de pesquisa essencial e o esforço para produzir, manter, adaptar, e fazer a curadoria do código deve ser reconhecido. O Software é parte de outras contribuições científicas vitais além de artigos publicados."

3.3.1. Software como Instrumento

Software é usado como parte integral de muitos instrumentos científicos, por exemplo, telescópios e microscópios. Outras vezes, o próprio software é o instrumento, gerando dados de pesquisa, validando dados de pesquisa, ou testando hipóteses. Isto inclui métodos computacionais ou modelos e simulações, por exemplo, modelos climáticos, modelos baseados em agentes em ciências sociais, simulação de hardware, dentre outros [Nieuwpoort and Katz 2023].

[18] `https://jupyter.org`
[19] `http://sciencecodemanifesto.org/`

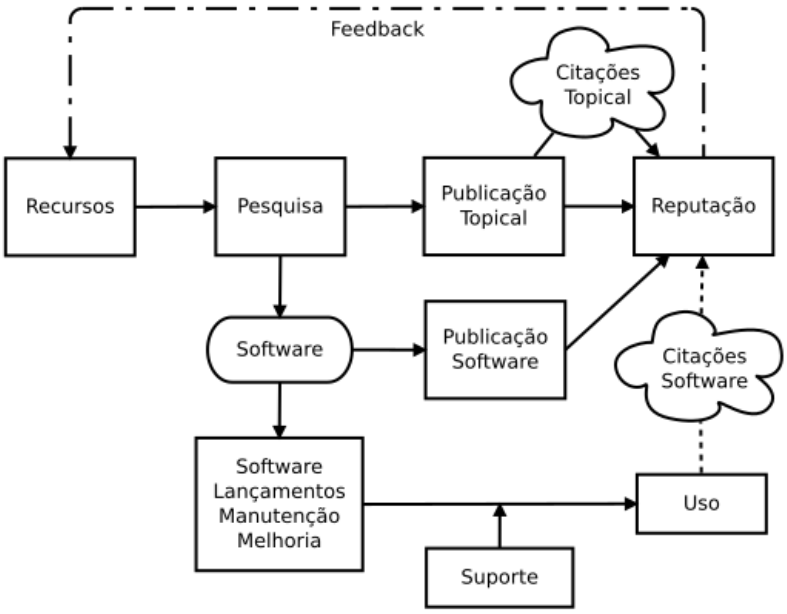


Figura 3.2. Uma visão dos incentivos de reputação num contexto misto entre Ciência e práticas de software de pesquisa. [Howison and Herbsleb 2011]

3.3.2. Software como Contribuição Científica

Software tornou-se parte fundamental de um ecossistema científico que engloba recursos, publicações [Howison and Herbsleb 2011] e possui a particularidade de se relacionar com o sistema econômico de reputação científica, especialmente com o seu modelo de publicações, influenciando e sendo influenciado diretamente pelo impacto de suas publicações [Howison et al. 2015].

A Figura 3.2 mostra que a *Pesquisa* conduzida por um ou mais pesquisadores pode gerar *Publicações científicas* com resultados para a área do conhecimento, e também pode produzir *Software* e *Publicação do software* [Howison et al. 2015]. As publicações científicas podem receber *Citações* dos pares, o que contribui para aumentar a *Reputação* acadêmica dos pesquisadores e, eventualmente, retroalimentar o sistema na forma de novos *Recursos* para a pesquisa.

A Figura 3.2 também relaciona *Software* a *Lançamentos, Manutenção e Melhoria* que exigem *Suporte* e levam à disponibilização do software para *Uso*. Entretanto, não há uma ligação direta e explícita entre a publicação do software e eventuais citações ao software. As setas tracejadas sugerem que o *Uso* do software pode gerar citações ao software e que estas podem influenciar de modo diferente (ou mesmo não influenciar) a reputação dos pesquisadores. Este problema pode ser explicado pela falta de padronização da citação de software e prática frequente de "citar" apenas uma URL que leva a um local onde o software pode ser encontrado.

A citação de software permite aos desenvolvedores receber reconhecimento e crédito pelo seu trabalho, além de facilitar a rastreabilidade e replicação dos resultados. Também contribui para a visibilidade e a colaboração na comunidade científica, fortalecendo a sustentabilidade técnica do software. No entanto, a importância do software na pesquisa ainda é subestimada, e o software é tratado como um recurso de apoio ou ferramenta, e não

como uma contribuição científica em si. No ecossistema da Ciência Aberta, o software de pesquisa deve ser citável e reconhecido como uma valiosa contribuição da pesquisa científica, tão importante quanto os artigos publicados, dados e metadados. No entanto, esforços e incentivos são necessários para que os créditos para o software de pesquisa se tornem mais específicos e rastreáveis na Ciência.

3.3.3. *Better Software, Better Research*

Pesquisadores sofrem uma intensa e constante pressão para publicar novos resultados rapidamente sem a devida cobrança sobre a necessidade de desenvolver código de qualidade. Como consequência, o software desenvolvido durante a pesquisa tende a não ser sustentável ou usável além da duração da pesquisa ou dos recursos que a financiam, e impede a comunidade científica de aproveitar plenamente o seu potencial impacto em pesquisas futuras.

Em 2016, inspirada pelo artigo *Better Software, Better Research* [Goble 2014], o workshop *Dagstuhl Perspectives* [Goble et al. 2016] reuniu ativistas, especialistas e interessados em discutir sobre o software produzido no contexto acadêmico e sua qualidade, na perspectiva de grupos de pesquisa que (i) consomem ou produzem software como saída do processo científico, ou (ii) consomem ou produzem software como um dos componentes dos métodos de pesquisa.

Dentre os resultados principais e com base em dados sobre o crescimento no uso de software em pesquisas científicas, o workshop identificou seis áreas no ecossistema de software de pesquisa que merecem atenção: *crédito ao software*, tornando-o rastreável nas pesquisas científicas; *herança cultural*, ou preocupação com as implicações das escolhas tecnológicas; *princípios FAIR para software* (discutidos na Seção 3.4.3); *financiamento* de atividades de desenvolvimento e manutenção de software; *políticas e e práticas* para incentivar a colaboração, a transparência e o compartilhamento de conhecimento entre os pesquisadores e instituições; e necessidade de *treinamentos* para pesquisadores e outros atores envolvidos com o desenvolvimento de software para uso na pesquisa.

Sobre treinamentos, o workshop recomendou que os mesmos devem cobrir diversos tópicos relacionados ao desenvolvimento de software e níveis de prática, e que cursos oferecidos por alguns institutos, por exemplo, *Carpentries*, devem ser estimulados e apoiados. Práticas centrais para manutenibilidade do software devem ser apresentadas, incluindo controle de versão, padrões de codificação, revisão de código, programação em par, testes, cobertura de código, integração contínua, refatoração e documentação. Outros tópicos também devem ser considerados, por exemplo, quão aberto é um software, se é desenvolvido por uma comunidade ou por um indivíduo, quão ativa é a manutenção do software (correção de *bugs* e criação de novas funcionalidades) ou disponibilidade de recursos diversos [Sufi et al. 2020].

3.4. Software de Pesquisa

3.4.1. Definição

O termo *software de pesquisa* foi criado para designar, de modo abrangente, o software *utilizado* durante a pesquisa científica, incluindo o software de terceiros usado para co-

leta, processamento e análise de dados [Allen et al. 2017]. No contexto de avaliação de qualidade do software de pesquisa, [Gruenpeter et al. 2021] adotam uma visão mais restrita, e consideram que sistemas operacionais, bibliotecas, dependências, pacotes e *scripts* utilizados na pesquisa científica, mas que não foram criados durante ou com uma intenção de pesquisa clara, são *software usado na pesquisa* mas *não são software de pesquisa*. No restante deste capítulo, adotaremos a visão de software de pesquisa de [Gruenpeter et al. 2021]:

Software de Pesquisa é o software desenvolvido durante o processo de pesquisa e inclui (mas não está limitado a) código-fonte, algoritmos, scripts, fluxos de trabalho computacionais e executáveis.

O *software de pesquisa* é parte integrante do ecossistema de pesquisa moderno, sendo amplamente utilizado nas Ciências e Engenharias para gerar resultados que servem como evidência em publicações científicas. Em geral, devido à complexidade em desenvolver software e ao conhecimento de domínio especializado necessário, os próprios pesquisadores desenvolvem o *software de pesquisa* ou estão intimamente envolvidos com o seu desenvolvimento [Carver et al. 2007].

Há várias preocupações relacionadas à *natureza* e à *qualidade* do *software de pesquisa*. Como artefato digital, ele pode assumir muitas formas, por exemplo, um *script* shell bash, com apenas 50 linhas para manipular e filtrar arquivos, um conjunto de *scripts* R de 100 linhas para análise de dados, 10.000 linhas de código Java para software de análise de dívida técnica ou 50.000 linhas de C++ para análise de imagens médicas.

Como produto de software, o *software de pesquisa* deve considerar alguns atributos de qualidade desejáveis. Por exemplo, é desejável que o *software de pesquisa* seja confiável, eficiente, fácil de usar, fácil compreender, testar, reparar, estender ou adaptar. Em especial, há uma preocupação com a sustentabilidade do *software de pesquisa* e sua influência na reprodutibilidade da pesquisa científica.

Neste capítulo, apresentamos aspectos relacionados à sustentabilidade do *software de pesquisa* e boas práticas para o desenvolvimento de software sustentável, a saber: registro e identificação do software, licenças de software e grau de abertura (*openness*), reconhecimento e citação do software, comunidade e usuários, longevidade e manutenibilidade, aderência aos princípios FAIR para software, avaliação do *software de pesquisa* e sua influência na reprodutibilidade científica.

3.4.2. Sustentabilidade

Sustentabilidade é um dos princípios norteadores da Ciência Aberta [UNESCO 2021] e define que a mesma deve se basear em práticas, serviços, infraestruturas e modelos de financiamento de longo prazo que garantam a participação igualitária dos indivíduos que produzem ciência originários de instituições e países menos privilegiados. As infraestruturas científicas abertas devem ser organizadas e financiadas com base em uma visão essencialmente sem fins lucrativos e de longo prazo, que aprimorem as práticas de Ciência Aberta e garantam o acesso permanente e irrestrito a todos, na medida do possível. No contexto da Ciência Aberta e sua dependência no código aberto (Seção 3.2.5), observa-se uma preocupação crescente com a sustentabilidade do software usado ou desenvolvido

durante a pesquisa, em especial, com a longevidade e disponibilidade do software, e seu impacto na reprodutibilidade científica.

O tema Sustentabilidade, inicialmente associado ao campo da Ecologia, agora faz parte dos interesses de outras áreas do conhecimento, ainda que adaptado às especificidades de cada uma. Na Ciência da Computação, a preocupação com a sustentabilidade emerge como um tópico importante em diversas subáreas, incluindo Inteligência Artificial, Computação de Alto Desempenho, Interação Humano-Computador, Computação Científica e Engenharia de Software [Venters et al. 2021]. Sustentabilidade é um desafio a ser enfrentado, não um problema a ser resolvido [Becker et al. 2015].

O *Manifesto de Karlskrona* para o Design Sustentável [Becker et al. 2015] define que sustentabilidade é, em sua essência, um conceito sistêmico, multi-facetado e deve ser entendida em um conjunto de cinco dimensões: recursos ambientais, social, bem-estar individual, prosperidade econômica e viabilidade técnica de longo prazo.

Neste capítulo, abordamos aspectos da dimensão técnica da sustentabilidade do software [Kehrer and Penzenstadler 2018] e, de forma complementar, elementos de sua dimensão social [de Souza 2023]. Não trataremos de aspectos da Engenharia de Software Verde [Mourão et al. 2018][20].

Sustentabilidade de Software

Na Engenharia de Software, há duas linhas de pesquisa voltadas para Sustentabilidade que se destacam: (i) Sustentabilidade de Software, e (ii) Engenharia de Software para Sustentabilidade (SE4S). A pesquisa sobre *Sustentabilidade de Software* tem como preocupação central a capacidade do software de perdurar ao longo do tempo, enquanto que a pesquisa relacionada a SE4S preocupa-se com sistemas intensivos em software, como integrar a sustentabilidade em seus processos de desenvolvimento de software e apoiar a sustentabilidade ambiental na ampla variedade de domínios em que o software é implantado [Venters et al. 2021].

No que se refere à sustentabilidade de software, a longevidade como expressão de tempo (“longo prazo”) e a capacidade de manutenção são fatores-chave para sua compreensão [Venters et al. 2017]. A preocupação com a longevidade do software estende-se ao atributo de manutenibilidade, e ao modelo e processo de desenvolvimento de software adotados, que podem influenciar atributos relacionados à sustentabilidade. A manutenibilidade é reconhecidamente uma qualidade interna fundamental de sistemas de software [IEC 2014], enquanto que a sua relação com a sustentabilidade de software ainda é objeto de pesquisa na área. O termo sustentabilidade propriamente dito ainda não está bem definido neste contexto, deixando espaço para diferentes interpretações, e ainda há pouca evidência ou orientação sobre processos ou modelos de desenvolvimento voltados para a sustentabilidade de software [Venters et al. 2021].

De um ponto de vista puramente técnico, [Venters et al. 2021] define sustentabilidade de software como um requisito composto, não-funcional, de primeira classe, abran-

[20]A Engenharia de Software Verde leva em consideração práticas e arquitetura de software, design de hardware e *data center*, o mercado de eletricidade e mudanças climáticas, visando gerar menos emissões de gases de efeito estufa e reduzir produção de carbono de uma empresa [Mourão et al. 2018].

gendo as medidas de alguns conceitos centrais de atributos de qualidade de software, incluindo, no mínimo, manutenibilidade, extensibilidade e usabilidade. Na dimensão técnica, a sustentabilidade de software está relacionada às consequências de longo prazo de projetar, construir e entregar um projeto de software e se espalha por diversas áreas, dentre elas, qualidade de software e métricas, requisitos de software e arquitetura de software [Kehrer and Penzenstadler 2018, Venters et al. 2021]. Por fim, sustentabilidade de software diz respeito a assegurar que o software continue funcional para seus usuários ao longo do tempo, considerando também sua manutenção, inclusão de novos recursos, reparo de *bugs*, e adaptações a novos ambientes de software e hardware.

Software de Pesquisa Sustentável

O *Software de Pesquisa Sustentável* deve permanecer *disponível e funcional* para a comunidade científica durante períodos de tempo significativos. Não há uma resposta geral para a questão de quanto tempo o software precisa ser sustentado ou mantido. Este período pode depender da área de pesquisa, finalidade, função, frequência de uso, e da comunidade que o desenvolveu.

Sustentabilidade de *software de pesquisa* inclui o processo de desenvolvimento e manutenção de software para que o mesmo continue a cumprir seu propósito ao longo do tempo. Certamente, o *software de pesquisa* sustentável precisará ser atualizado com novas funcionalidades e correções de *bugs*, adaptado a novos ambientes computacionais, manter-se amigável aos seus usuários, tornar-se multiplataforma, ser testado e certificado. Garantir que o *software de pesquisa* continue executável é desafiador, sendo mais trabalhoso e caro do que um arquivamento simples de uma versão do software. As alterações necessárias para que o software permaneça executável devem garantir a confiabilidade nos resultados gerados pelas versões mais antigas do software. Se os resultados forem diferentes, justificativas objetivas devem ser fornecidas.

Há diversas maneiras de promover e investir na sustentabilidade do *software de pesquisa*, incluindo atração de desenvolvedores, suporte à comunidade de usuários, busca por financiamento ou até comercialização – todas válidas, desde que resultem na disponibilidade de longo prazo do software para a comunidade científica. Espera-se que, por meio da publicação de instruções, diretrizes e outras formas de ajuda e suporte, pesquisadores sejam capazes de decidir a forma de manter o seu software sustentável.

3.4.3. *FAIRness*

Os princípios FAIR [Wilkinson et al. 2016] foram especificados para melhorar o reuso de dados de pesquisa digitais, tornando-os mais fáceis de encontrar, acessíveis, interoperáveis e reutilizáveis (**F**indable, **A**ccessible, **I**nteroperable, **R**eusable). Tais princípios abordam a *forma* de fornecer artefatos para a comunidade científica, mas não tratam do conteúdo funcional ou da qualidade dos artefatos [Lamprecht et al. 2020]. Os *Princípios FAIR para Dados de Pesquisa* [Wilkinson et al. 2016], incluindo quatro princípios fundamentais e 15 princípios norteadores, estabelecem que os dados de pesquisa devem ser facilmente localizáveis, acessíveis, interoperáveis e reutilizáveis.

Segundo [Chue Hong et al. 2022], o *software de pesquisa* deve seguir os princípios FAIR usados para dados abertos, considerando-se que o software também é um

artefato de pesquisa digital e, como tal, deve ser facilmente localizável, acessível, interoperável e reutilizável. Os *Princípios FAIR para Software de Pesquisa* (FAIR4RS) foram definidos a partir de uma reformulação dos princípios FAIR originais para dados abertos [Lamprecht et al. 2020, Chue Hong et al. 2022, Barker et al. 2022]. É importante destacar que, diferentemente dos dados, o software não é um artefato estático e só pode ser (re)utilizado se for sustentável [Lamprecht et al. 2020]. A Tabela 3.1 apresenta a versão mais recente dos princípios FAIR para *software de pesquisa*.

Projetos de Software Livre seguem os princípios FAIR. Software livre pode ser localizado em repositórios com base em identificadores e descritores, utilizando diversos critérios como palavras-chave, linguagem de programação, versão do software, entre outros. A acessibilidade é encorajada em software disponível em repositórios abertos, com licenças de compartilhamento explícitas e bem definidas e documentação associada. A definição de interfaces de programação, formatos de entrada/saída e uso de padrões promovem a interoperabilidade e o reúso por vários grupos de pesquisa. Nessa perspectiva, as práticas usadas no modelo de desenvolvimento de software livre podem ser adotadas no desenvolvimento e evolução de *software de pesquisa* [Flach and Kon 2021].

Tabela 3.1. Princípios FAIR para Software [Barker et al. 2022].

Princ.	**Descrição**
F:	Software, and its associated metadata, is easy for both humans and machines to find.
F1	Software is assigned a globally unique and persistent identifier.
F1.1	Software components representing granularity levels are assigned distinct identifiers.
F1.2	Different versions of the software are assigned distinct identifiers.
F2	Software is described with rich metadata.
F3	Metadata clearly and explicitly include the identifier of the software they describe.
F4	Metadata are FAIR, searchable and indexable.
A:	Software, and its metadata, is retrievable via standardised protocols.
A1	Software is retrievable by its identifier using a standardised communications protocol.
A1.1	The protocol is open, free, and universally implementable.
A1.2	The protocol allows for authentication and authorization procedure, where necessary.
A2	Metadata are accessible, even when the software is no longer available.
I:	Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1	Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2	Software includes qualified references to other objects.
R:	Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).
R1	Software is described with a plurality of accurate and relevant attributes.
R1.1	Software is given a clear and accessible license.
R1.2	Software is associated with detailed provenance.
R2	Software includes qualified references to other software.
R3	Software meets domain-relevant community standards.

3.4.4. Reprodutibilidade

A *Reprodutibilidade* é um dos fundamentos do método científico e um princípio norteador da Ciência Aberta [UNESCO 2021]. A boa prática científica exige que os artefatos de pesquisa mencionados em publicações científicas sejam mantidos e fiquem disponíveis para escrutínio dos pares, reprodução independente e verificação de resultados.

Para o *software de pesquisa*, a submissão ou publicação de um artigo é um dos momentos em que a versão do software usada no estudo precisa ser identificada, documentada e lançada. Se houver modificações no *software de pesquisa*, elas precisam ser registradas, usando algum esquema de nomenclatura que identifique o <*software*>, a <*versão*> e o <*lançamento*>. Em geral, a versão refere-se a mudança estratégica durante a evolução do software e o lançamento refere-se a mudanças simples de serviço.

A recomendação para que pesquisadores compartilhem e permitam o acesso aos dados descritos em suas publicações científicas, não garante a reprodutibilidade da pesquisa. Na prática, pesquisadores devem compartilhar o código de todo o *software de pesquisa* desenvolvido e fluxos de trabalho de suas pesquisas para assegurar a reprodutibilidade. Na maioria dos casos, o arquivamento de baixo custo do software, acompanhado de documentação clara deve ser suficiente.

Diretrizes para o desenvolvimento sustentável de *software de pesquisa* também podem contribuir para a condução de uma pesquisa científica reprodutível, apresentando princípios e boas práticas da engenharia de software. As implementações das diretrizes podem variar entre os diferentes domínios científicos, mas as implementações devem ser públicas, abertas para discussão entre os pesquisadores e continuamente adaptadas em reposta às mudanças tecnológicas e nos domínios.

3.4.5. Avaliação de *software de pesquisa*

Consideramos a avaliação do *software de pesquisa* sob duas perspectivas: (1) sustentabilidade ou direcionada a sua longevidade, e (2) aderência aos princípios FAIR (*FAIRness*) ou direcionada ao grau de abertura (*openness*) do *software de pesquisa*.

Sustentabilidade

A avaliação da sustentabilidade do *software de pesquisa* busca determinar se o mesmo é sustentável com base em critérios relevantes para o ecossistema científico. Em geral, a avaliação não gera um resultado *é sustentável/não é sustentável*, mas tende a valorizar pontos fortes do software e identificar pontos para sua melhoria. A avaliação pode considerar atributos ou práticas usadas no desenvolvimento do software para prover uma visão geral ou detalhada da sustentabilidade do *software de pesquisa*.

O Instituto de Sustentabilidade de Software (SSI) fornece um serviço de avaliação quantitativa do software baseada em critérios relacionados a sustentabilidade, manutenibilidade e usabilidade[21]. A avaliação é feita com base em respostas dadas a um conjunto de perguntas simples, seguidas por explicações sobre a importância de algumas práticas e recomendações. O conjunto de perguntas é aplicável para *software de pesquisa*.

[21]O questionário do SSI para avaliação de sustentabilidade está disponível em `http://www.software.ac.uk/online-sustainability-evaluation`.

Tabela 3.2. Avaliação de sustentabilidade do *software de pesquisa* baseada em práticas.

P	Descrição	Atende?	Comentário
P1	O software está hospedado em um repositório público		
P2	O software utiliza controle de versão		
P3	O software adota explicitamente uma licença		
P4	O software está registrado e apresenta um DOI		
P5	A estrutura de arquivos do projeto de software comunica a finalidade de seus elementos		
P6	O software usa formato de dados e interfaces padronizadas		
P7	A documentação apresenta uma visão geral do software		
P8	O software possui testes		
P9	O código é revisado antes de ser publicado		
P10	O projeto de software utiliza rastreador de tarefas e *bugs*		
P11	Tarefas repetitivas são automatizadas		
P12	Há integração e implantação contínuas		
P13	Há lançamento de versões do software		
P14	Há evidência de uma comunidade (presente ou futuro)		
P15	O software é divulgado para a comunidade acadêmica		
P16	Há uma forma recomendada para citação do software		

Por exemplo, para uma resposta negativa para a pergunta "*Seu software de pesquisa está disponível como um pacote que pode ser instalado sem compilar?*", a avaliação oferece a recomendação geral "Construir software pode ser complicado e demorado. Fornecer seu *software de pesquisa* como um pacote que pode ser implantado sem compilar pode economizar tempo e esforço dos usuários, especialmente se não forem desenvolvedores de software. Idealmente, deve-se observar e testar se o *software de pesquisa* é compilado e executado em diversas plataformas para as quais ele oferece suporte, para então prover pacotes para sua distribuição."

Em nossa pesquisa, a avaliação de sustentabilidade é simples, baseada em práticas, e busca determinar se um *software de pesquisa* foi desenvolvido seguindo boas práticas que potencialmente o qualificariam como software sustentável. As práticas usadas em nossas avaliações estão relacionadas a identidade e disponibilidade do software, adoção de licenças, controle de versão, documentação, estrutura do código-fonte, testes, política de contribuidores e suporte, entre outras. O resultado de cada avaliação é sintetizado em uma tabela e apresentado em um relatório simples. A Tabela 3.2 mostra práticas e aspectos que consideramos na avaliação da sustentabilidade de *software de pesquisa*.

FAIRness

A avaliação de *FAIRness* do *software de pesquisa* pode ser definida como um processo de avaliação do seu grau de aderência aos princípios FAIR adaptados para software. É importante lembrar que os princípios FAIR não são prescritivos, mas buscam oferecer uma visão para melhorar o compartilhamento e a reutilização de dados ou software por pessoas e máquinas [Wilkinson et al. 2016, Chue Hong et al. 2022]. Ainda assim, a avaliação de *FAIRness* para software pode servir para apoiar a decisão de usuários e desenvolvedores sobre uma eventual adoção do *software de pesquisa* em seu projeto de pesquisa.

No estudo apresentado neste capítulo, consideramos a versão mais recente dos princípios FAIR para software [Barker et al. 2022].

3.5. Desenvolvimento de Software de Pesquisa

Em geral, o desenvolvimento de software de pesquisa exige conhecimento específico sobre o domínio do estudo sendo realizado, por exemplo, entender como o DNA genômico se transforma em cristais de proteína, ou estar familiarizado com os meandros da dinâmica dos fluidos, ou saber como resolver 20 equações diferenciais parciais simultâneas [Segal and Morris 2008]. Isto explica a grande participação dos cientistas no desenvolvimento de software de pesquisa.

No Reino Unido, um estudo pioneiro de [Hettrick et al. 2014] envolvendo todas as áreas da Ciência, mostrou que 56% dos cientistas estavam envolvidos no desenvolvimento de software acadêmico [Hettrick et al. 2014]. Outros estudos em grupos específicos mostraram números ainda mais expressivos. Na área de Astronomia, por exemplo, 90% dos cientistas desenvolvem software de pesquisa [Momcheva and Tollerud 2015]. No entanto, a maior parte dos cientistas não havia recebido treinamento sobre conceitos e boas práticas de desenvolvimento de software.

Em geral, os pesquisadores que desenvolvem o seu *software de pesquisa* não testam ou documentam os seus projetos de software, e não seguem práticas básicas de desenvolvimento, como escrever código legível, revisar código, usar controle de versão ou testes unitários [Wilson et al. 2017]. A falta de conhecimento sobre a natureza do software, conceitos e práticas de desenvolvimento de software pode ocasionar sérios erros computacionais em conclusões centrais da literatura acadêmica, gerando retrabalho para retratar tais erros nas mais diversas áreas da Ciência [Merali 2010]. Além disso, dados podem ser perdidos, análises podem consumir mais tempo que o necessário e a pesquisa pode ter sua eficiência comprometida se pesquisadores não desenvolverem e trabalharem com *software de pesquisa* de qualidade [Wilson et al. 2017]. Por fim, o desconhecimento sobre práticas de desenvolvimento colaborativo e aberto pode causar um impacto negativo na visibilidade do *software de pesquisa*, na capacidade de ser encontrado e compartilhado [Howison and Herbsleb 2013, Katz 2014] e em sua sustentabilidade.

3.5.1. Ciclo de Vida do Software

O modelo de ciclo de vida do software proposto por Rajlich [Rajlich and Bennett 2000] descreve o software como um produto em constante evolução e serve para contextualizar os estágios de um *software de pesquisa* sustentável. A Figura 3.3 apresenta as etapas ou estágios em que o software pode se encontrar ao longo de sua existência.

Na etapa de Desenvolvimento inicial (*Initial development*), desenvolvedores implementam a primeira versão funcional do software. Se o desenvolvimento inicial for bem-sucedido, o software entra no estágio de Evolução (*Evolution*), quando ocorrem mudanças iterativas, modificações e exclusões de funcionalidade. O modelo de Rajlich destaca que, em determinados intervalos, *uma versão do software é liberada para a comunidade*. Na etapa de Serviço (*Servicing*), desenvolvedores fazem pequenos reparos de defeitos e mudanças funcionais simples. Na etapa de *Phaseout*, o software não recebe manutenção, mas continua disponível. No estágio de Fechamento (*Closedown*), o software

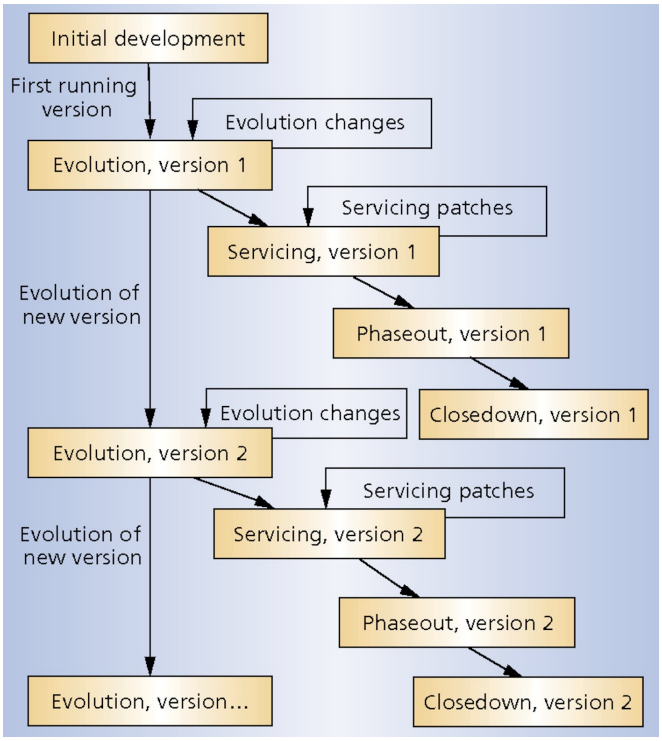


Figura 3.3. ***O modelo de Rajlich com versões enfatiza a natureza evolutiva do desenvolvimento de software. Fonte: [Rajlich and Bennett 2000].***

deixa de existir e os usuários podem ser redirecionados para outro software.

No contexto de desenvolvimento de *software de pesquisa*, deve-se considerar que ele se insere em um projeto de pesquisa que inclui *dados e software*. No início do desenvolvimento do *software de pesquisa* deve-se decidir sobre a licença do software, onde o software será armazenado (por exemplo, GitHub, GitLab, etc.), tipo de sistema de versionamento e padrões de entrada e saída. O lançamento de uma versão do *software de pesquisa* e sua liberação para a sua comunidade podem ser motivados pela publicação de um artigo científico. O texto do artigo deve mencionar claramente a versão do *software de pesquisa* utilizada.

Também é essencial considerar a continuidade do *software de pesquisa* ao longo do tempo e o suporte à reprodutibilidade, especialmente quando o software for usado na pesquisa publicada em artigos científicos. Ao contrário de outros tipos de software, a etapa de Fechamento (*Closedown*) não deveria ser considerada em *software de pesquisa* para permitir a reprodutibilidade dos resultados divulgados nos artigos publicados. Manter o *software de pesquisa* vivo, disponível e acessível é fundamental para que outros cientistas possam reproduzir e verificar os resultados da pesquisa.

3.5.2. Pessoas

Considerando que muitos problemas de manutenção de software não são problemas técnicos, mas problemas relacionados às pessoas [Versen 2020], percebemos que a dimensão social desempenha um papel importante na evolução do software, influenciando tanto o processo quanto o resultado final. As pessoas interessadas no desenvolvimento, utilização

e manutenção de software têm um impacto significativo em sua evolução. A interação e colaboração entre os membros da equipe de desenvolvimento e os usuários contribuem com suas perspectivas e experiências, influenciando as decisões sobre novas funcionalidades, correções de *bugs* e melhorias em geral. Além disso, as demandas e expectativas dos usuários, assim como mudanças em suas necessidades e preferências podem exigir adaptações e atualizações do software para atender a essas demandas.

O caso específico da evolução de um *software de pesquisa* apresenta características distintas. O *software de pesquisa* está inserido em um ecossistema científico, onde os pesquisadores buscam avançar o conhecimento em suas áreas. Nesse contexto, sua evolução está intimamente ligada à evolução da própria pesquisa e ao surgimento de novas abordagens e técnicas, exigindo atualizações e modificações no software. Além disso, a estrutura do grupo de pesquisa também influencia a evolução do software. À medida em que a pesquisa avança, novos membros ingressam no grupo, pesquisadores saem e a equipe se reconfigura. Tais mudanças na composição do grupo demandam mecanismos para transmitir o conhecimento legado e práticas estabelecidas.

Tornar um *software de pesquisa* disponível publicamente e aceitar contribuição de pessoas externas ao grupo de pesquisa pode resultar em uma comunidade maior de desenvolvedores envolvidos, facilitando o desenvolvimento mais rápido e eficiente, incluindo correções de *bugs* e implementação de novos recursos de forma colaborativa. No entanto, a coordenação de um *software de pesquisa* aberto pode exigir esforço adicional para estabelecer processos de governança e revisão de contribuições, para garantir que estas sejam de alta qualidade e estejam alinhadas com os objetivos do projeto. Além disso, a abertura precoce do *software de pesquisa* pode expor pesquisas em andamento antes que estejam prontas para a publicação, o que pode comprometer a confidencialidade e ineditismo dos resultados.

3.5.3. Práticas

Há alguns anos a comunidade científica já reconhece ser essencial que cientistas, pesquisadores e estudantes sejam capazes de aprender e adotar um novo conjunto de habilidades e metodologias relacionadas a software [Katz et al. 2016]. Alguns pesquisadores já abraçaram boas práticas de desenvolvimento de software e, em particular, uma classe especializada de desenvolvedores de software, chamados de *Engenheiros de Software de Pesquisa* (*Research Software Engineers*), está surgindo em ambientes acadêmicos como parte integrante de grupos de pesquisa bem-sucedidos [Katz et al. 2016]. O uso consistente de boas práticas é apoiado por tecnologias e ferramentas conhecidas e tradicionalmente usadas por engenheiros de software.

3.6. Práticas para o Desenvolvimento de Software de Pesquisa Sustentável

Projetos de software bem organizados adotam um modelo de desenvolvimento que resulta em código de alta qualidade que se adapta rapidamente a diferentes situações. Várias *práticas* estabelecidas e usadas pelas comunidades de software livre são reconhecidas como contribuições importantes para a manutenção de *software de pesquisa* de alta qualidade. Nesta seção, apresentamos dezesseis boas práticas para uso em projetos de *software de pesquisa*, descritas e ilustradas no contexto de projetos hospedados no GitHub. Parte do

jargão técnico usado pelo GitHub não foi traduzido e alguns termos não foram definidos. Termos e suas definições podem ser encontrados no *Glossário do GitHub*[22].

3.6.1. Práticas Básicas

P1. Hospedagem do projeto

É recomendável hospedar e compartilhar o *software de pesquisa* em um repositório público, exceto se houver questões de confidencialidade ou estratégicas relacionadas ao projeto de pesquisa (por exemplo, software implementa um algoritmo original, ainda não compartilhado amplamente com a comunidade científica). A hospedagem em repositório público pode ser estendida para todos os ativos da pesquisa, por exemplo, dados, fluxos de trabalho e relatórios.

A *hospedagem pública do software* é fundamental para promover a reprodutibilidade, evitar a redundância (por exemplo, cópias do software em diversos repositórios locais), e estimular o compartilhamento entre pares da comunidade acadêmica. A hospedagem pública do software e o acesso aos dados e fluxos de trabalho, permitem que os resultados da pesquisa possam ser verificados, replicados, e reproduzidos, fortalecendo a confiabilidade e a transparência dos estudos. Além disso, o *software de pesquisa* hospedado torna-se localizável e acessível, facilitando seu uso ou reuso. Se o código for aberto, outros pesquisadores podem se beneficiar dele, evitando o retrabalho e construindo sobre uma base já estabelecida. Isso promove a colaboração, a troca de conhecimentos e a aceleração do progresso científico.

Exemplo. O *software de pesquisa* `flosssearch` estava hospedado em uma pasta local no computador de um pesquisador. Por segurança, o pesquisador realizava backup manual periodicamente. Ao finalizar a implementação da primeira funcionalidade do software, o pesquisador decidiu que o projeto precisava ser hospedado adequadamente e compartilhado com outros membros do grupo de pesquisa. O software `flosssearch` foi colocado em um repositório público no GitHub, uma plataforma baseada na web onde os usuários podem hospedar repositórios, compartilhar e promover a colaboração em projetos de software. O repositório foi compartilhado com dois pesquisadores do grupo.

Ao ser hospedado no GitHub, o software recebeu uma URL, um nome oficial e a sua identidade foi definida, de forma clara e única. O nome oficial público atribuído ao *software de pesquisa* é `flosssearch`. Além disso, o software `flosssearch` passou a contar com um serviço de backup, considerando que os arquivos do projeto e todo o seu histórico são armazenados no servidor remoto e nos repositórios locais dos pesquisadores.

P2. Controle de versão

Controle de versão é a prática de rastrear e gerenciar alterações no código de um software. Os sistemas de controle de versão são ferramentas que permitem que várias versões do mesmo *software de pesquisa* sejam mantidas e, possivelmente, referenciadas por experimentos de pesquisa e artigos científicos. Um sistema de controle de versão também permite o acompanhamento das atividades realizadas, fornece um registro das alterações e permite que desenvolvedores e usuários do *software de pesquisa* acompanhem o seu

[22] `https://docs.github.com/en/get-started/quickstart/github-glossary`

desenvolvimento e evolução, tornando a pesquisa mais aberta e reprodutível. Ao usar o controle de versão, o pesquisador nunca perde as versões anteriores do *software de pesquisa*, e erros podem ser revertidos. Adicionalmente, o versionamento do software também promove a colaboração entre pesquisadores.

Exemplo. A plataforma GitHub utiliza o Git[23], um sistema de controle de versão distribuído gratuito e de código aberto. Ao ser colocado no GitHub, com um nome e repositório associado, o *software de pesquisa* `flosssearch` recebeu suporte para controle de versão distribuído.

P3. Licenças de Software

A licença de software escolhida é essencial para o *software de pesquisa* disponível publicamente. O financiamento do projeto pode terminar após um certo período, e os mantenedores podem terminar suas pesquisas ou mudar de área de interesse. Assim, para garantir a disponibilidade contínua do projeto, os desenvolvedores precisam chegar a um acordo formal, ou seja, uma licença de software em que os termos de uso sejam explícitos.

Recomenda-se que projetos de *software de pesquisa* de código aberto sejam lançados sob uma licença compatível com a GNU General Public License - GPL[24], a licença OSS mais popular e amplamente utilizada. Outras licenças de software de código aberto e suas descrições podem ser encontradas no site da Open Source Initiative (OSI)[25].

O GitHub permite que o usuário escolha uma licença de software para o seu repositório e cria automaticamente o arquivo *LICENSE* com uma cópia da licença escolhida. Entretanto, apenas colocar uma cópia da licença em um arquivo em seu repositório não declara explicitamente que o código no repositório pode ser usado sob tal licença. Sem uma declaração explícita, não fica claro se as permissões da licença se aplicam a qualquer arquivo fonte no repositório.

A Figura 3.4 mostra um aviso que o desenvolvedor deve incluir no início de cada arquivo fonte para declarar a licença e a exclusão da garantia. Cada arquivo deve ter pelo menos a linha “copyright” e indicação sobre local onde o aviso completo pode ser encontrado. A falta de uma licença claramente definida pode gerar incerteza legal e dificultar a contribuição e o compartilhamento do software, já que os colaboradores podem ficar inseguros sobre seus direitos e obrigações, o que pode limitar a adoção e colaboração.

Exemplo. Ao ser colocado em um repositório público no GitHub, o *software de pesquisa* `flosssearch` tornou-se visível, acessível e utilizável por terceiros, sendo necessária a escolha de uma licença de código aberto. Os pesquisadores escolheram a licença GNU GPL 3.0. Além do arquivo LICENSE, os arquivos com código-fonte do projeto possuem uma breve descrição sobre o seu propósito, a linha “copyright” e indicação sobre a licença de software escolhida e onde o texto completo pode ser encontrado (Figura 3.5).

[23] `https://git-scm.com`
[24] `https://www.gnu.org/licenses/gpl-3.0.html`
[25] `https://opensource.org/licenses/`

> *<one line to give the program's name and a brief idea of what it does.>*
>
> Copyright (C) *<year>* *<name of author>*
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <https://www.gnu.org/licenses/>.

Figura 3.4. Declaração explícita sobre uso de licença GPL e exclusão da garantia.

> `flossearch` is a web application designed with the goal of supporting instructors and students in the selection of OSS projects for software engineering education.
>
> Copyright (C) *2021 Moara Brito*
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License 3.0 <https://www.gnu.org/licenses/>.

Figura 3.5. Declaração sobre uso de licença GPL para inclusão nos arquivos do `flosssearch`.

P4. Registro de Software

O *software de pesquisa* deve ser publicado formalmente para satisfazer os princípios FAIR, promover a sustentabilidade de software e permitir a citação de software. Os repositórios de publicação dão suporte ao registro do software e fornecem versões de software publicadas com identificadores únicos e persistentes, como um DOI.

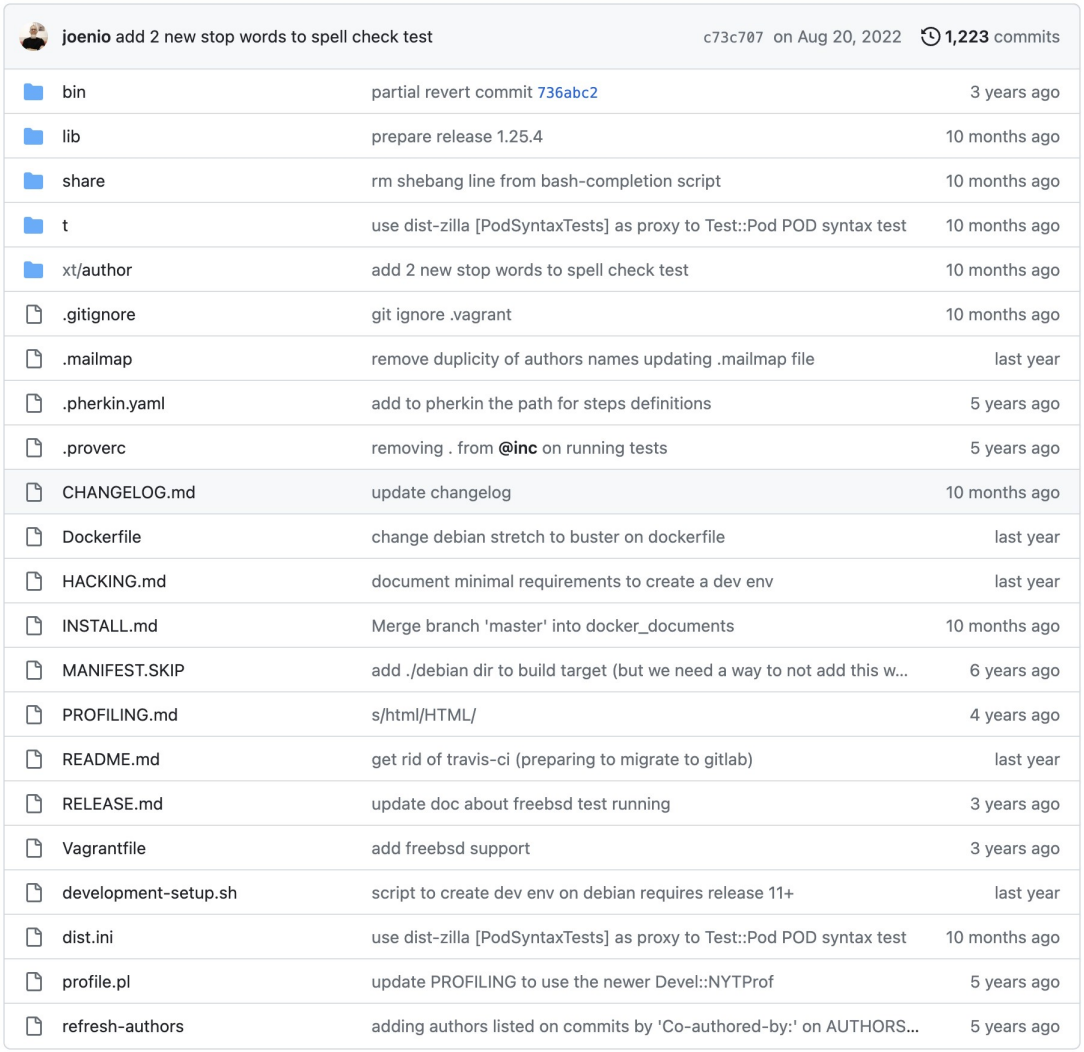
Há várias formas de associar um DOI a um repositório GitHub. Zenodo[26] e Figshare[27] podem ser utilizados para fazer o registro do DOI e arquivar versões especificas do software. Há também o projeto *The Research Software Directory*[28] desenhado para centralizar metadados de software de pesquisa e mostrar os seus impactos na pesquisa e na sociedade, estimulando o reuso de software e encorajando a citação de maneira apropriada para garantir que pesquisadores e engenheiros de *software de pesquisa* recebam créditos por seus trabalhos.

Exemplo. O *software de pesquisa* `Parcels`[29] (**P**robably **A** **R**eally **C**omputationally **E**fficient **L**agrangian **S**imulator) é um conjunto de classes e métodos em Python para criar simulações de rastreamento de partículas, como plâncton, plástico e peixes. O `Parcels` está registrado no Research Software Directory[30].

[26] `https://zenodo.org`
[27] `https://figshare.com`
[28] `https://research-software-directory.org`
[29] `https://github.com/OceanParcels/parcels`
[30] `https://research-software-directory.org/software/parcels`

3.6.2. Organização do Projeto

O suporte à reprodutibilidade na pesquisa que utiliza um *software de pesquisa* requer que seus artefatos estejam bem organizados em um repositório aberto.



joenio add 2 new stop words to spell check test	c73c707 on Aug 20, 2022	**1,223** commits
bin	partial revert commit 736abc2	3 years ago
lib	prepare release 1.25.4	10 months ago
share	rm shebang line from bash-completion script	10 months ago
t	use dist-zilla [PodSyntaxTests] as proxy to Test::Pod POD syntax test	10 months ago
xt/author	add 2 new stop words to spell check test	10 months ago
.gitignore	git ignore .vagrant	10 months ago
.mailmap	remove duplicity of authors names updating .mailmap file	last year
.pherkin.yaml	add to pherkin the path for steps definitions	5 years ago
.proverc	removing . from **@inc** on running tests	5 years ago
CHANGELOG.md	update changelog	10 months ago
Dockerfile	change debian stretch to buster on dockerfile	last year
HACKING.md	document minimal requirements to create a dev env	last year
INSTALL.md	Merge branch 'master' into docker_documents	10 months ago
MANIFEST.SKIP	add ./debian dir to build target (but we need a way to not add this w...	6 years ago
PROFILING.md	s/html/HTML/	4 years ago
README.md	get rid of travis-ci (preparing to migrate to gitlab)	last year
RELEASE.md	update doc about freebsd test running	3 years ago
Vagrantfile	add freebsd support	3 years ago
development-setup.sh	script to create dev env on debian requires release 11+	last year
dist.ini	use dist-zilla [PodSyntaxTests] as proxy to Test::Pod POD syntax test	10 months ago
profile.pl	update PROFILING to use the newer Devel::NYTProf	5 years ago
refresh-authors	adding authors listed on commits by 'Co-authored-by:' on AUTHORS...	5 years ago

Figura 3.6. Estrutura de arquivos da ferramenta `Analizo`.

P5. Estrutura

É importante ter uma estrutura de arquivos que comunica a finalidade dos elementos dentro de um *software de pesquisa*, separando os interesses em uma hierarquia de pastas e usando nomes auto-explicativos, facilitando a compreensão de todos que tenham interesse revisitar, revisar e desenvolver o software.

Exemplo. A Figura 3.6 apresenta a estrutura de arquivos da ferramenta de análise estática `Analizo`[31][Terceiro et al. 2010], desenvolvida como *software de pesquisa* no contexto de uma pesquisa de doutorado. Pode-se observar que há pastas e arquivos com nomes auto-explicativos, por exemplo, *bin*, *lib* e *share*, INSTALL.md, README.md.

[31] `https://github.com/analizo/analizo`

P6. Padronização

A padronização e o uso de protocolos de comunicação entre sistemas tem uma grande importância no software de pesquisa. A partir da adoção de padrões e protocolos bem definidos, os diferentes sistemas podem interagir de forma consistente, facilitando a integração e desencoraja a dependência de fornecedores específicos. Além disso, a padronização simplifica a interoperabilidade e a reutilização de componentes de software, permitindo que os pesquisadores desenvolvam seus trabalhos sobre trabalhos existentes e tornem mais rápido o desenvolvimento de novas soluções.

Exemplo. O software `MoSyn` é um *software de pesquisa* que se preocupa com padronização pois depende de um software proprietário, o MATLAB[32]. A Seção 3.7 apresenta uma avaliação do `MoSyn` com base nas práticas para o desenvolvimento adotadas.

P7. Documentação

A documentação frequente e contínua é uma prática recomendada para manter guias do usuário, manuais e outros documentos relevantes atualizados em relação à versão mais recente do software, facilitando o entendimento e colaboração. Para o *software de pesquisa*, a documentação é um pré-requisito fundamental para reuso e reprodução de estudos [Chue Hong et al. 2022]. [Hermann and Fehr 2022] analisaram o estado-da-prática sobre documentação de *software de pesquisa*. Eles reportaram que, ainda que alguns guias com boas práticas científicas valorizem e recomendem a prática de documentar explicitamente [Forschungsgemeinschaft 2022], a documentação do *software de pesquisa* ainda é considerada inadequada [Wilson et al. 2017, Chue Hong et al. 2022].

A documentação de um projeto só é útil se está atualizada. Por isso, uma questão importante em um projeto de software é a sincronização entre o desenvolvimento do software e a documentação dele, garantindo que a documentação representa a versão mais atualizada do software. Para conseguir essa sincronização é essencial integrar o processo de atualização da documentação no ciclo de desenvolvimento do software. Além disso, é possível automatizar a geração de documentação a partir de comentários no código-fonte, reduzindo o esforço necessário para manter a documentação atualizada.

Em geral, geradores de documentação são usado para interfaces de programação de aplicativos (APIs), destinadas a um público de desenvolvedores. *Doxygen*[33] é uma ferramenta que extrai documentação a partir do código-fonte e de outros arquivos e gera uma versão em formato estruturado, como HTML, PDF ou LaTeX. Doxygen é a ferramenta padrão de fato para gerar documentação para C++, mas também suporta outras linguagens de programação populares como C, Objective-C, C#, PHP, Java e Python.

Uma documentação precisa ser clara e considerar os diferentes interessados no *software de pesquisa*. A documentação precisa atender às necessidades específicas de cada grupo, usando linguagem e terminologia que ajudem cada público a compreender e utilizar as informações de maneira eficaz.

A *documentação para desenvolvedores* deve abordar aspectos técnicos, como arquitetura, APIs, dependências e configurações, incluindo descrições detalhadas de como

[32] `https://www.mathworks.com/products/matlab.html`
[33] `https://www.doxygen.nl`

utilizar as funcionalidades do software. É importante que sejam fornecidas orientações passo-a-passo sobre instalação, configuração, execução e dependências necessárias para o ambiente de desenvolvimento do software. Para facilitar a contribuição, é essencial descrever conceitos fundamentais, padrões de codificação e boas práticas de desenvolvimento. Testes de software também oferecem uma forma de documentar a funcionalidade do software por meio da qual desenvolvedores podem ter uma visão mais detalhada sobre o funcionamento interno do software e integração com outros sistemas.

Na *documentação para pesquisadores* que não são desenvolvedores, deve-se utilizar uma linguagem acessível, evitando jargões técnicos e explicando conceitos de forma simples e compreensível. A descrição da utilização deve destacar as funcionalidades e benefícios do software. Para facilitar o uso, podem ser apresentados exemplos práticos de como o *software de pesquisa* pode ser aplicado em diferentes cenários e incluir uma seção de perguntas frequentes para responder às dúvidas mais comuns dos usuários. Dessa forma, a documentação para os pesquisadores que não são desenvolvedores pode ajudá-los a entender como utilizar o software de forma prática para atingir seus objetivos de pesquisa, sem detalhes técnicos desnecessários.

Para todos os tipos de usuários, recomenda-se que, ao publicar o *software de pesquisa*, o desenvolvedor disponibilize uma página de boas-vindas e descrição do software. Isso pode ser feito facilmente com um arquivo *README* colocado no diretório raiz do projeto. Uma boa descrição sintetiza as funcionalidades do *software de pesquisa* e permite que outros pesquisadores possam decidir se querem usá-lo ou não. O desenvolvedor também deve incluir informações de contato e canais de suporte, como e-mail, fórum de discussão e indicar ferramentas que podem ser usadas para que os usuários solicitem ajuda adicional.

Exemplo. O software `flosssearch` disponibiliza para os usuários informações sobre funcionalidades, tecnologias e licença utilizada, em um arquivo README, sob controle de versão, que pode ser atualizado regularmente. Porém, não há informações de contato ou sobre canais de comunicação. A Figura 3.7 mostra uma parte da excelente documentação para usuários disponibilizada no website da ferramenta `Analizo`[34].

3.6.3. Qualidade

O processo de garantia da qualidade do *software de pesquisa* é tão necessário quanto o código que foi escrito. *Confiabilidade* é um dos atributos de qualidade do software, em geral definido como a probabilidade do mesmo funcionar sem a ocorrência de falhas em um período específico. Testes e revisão de código são duas técnicas populares que podem ser usadas no desenvolvimento de *software de pesquisa* confiável.

P8. Teste de software

Teste de software é a atividade de executar um produto de software para verificar se ele está em conformidade com suas especificações. Testes fornecem evidências sobre a confiabilidade do software, além de promover agilidade e, possivelmente, redução nos custos de desenvolvimento, depuração e manutenção do software [Delamaro et al. 2013].

[34] `https://www.analizo.org`

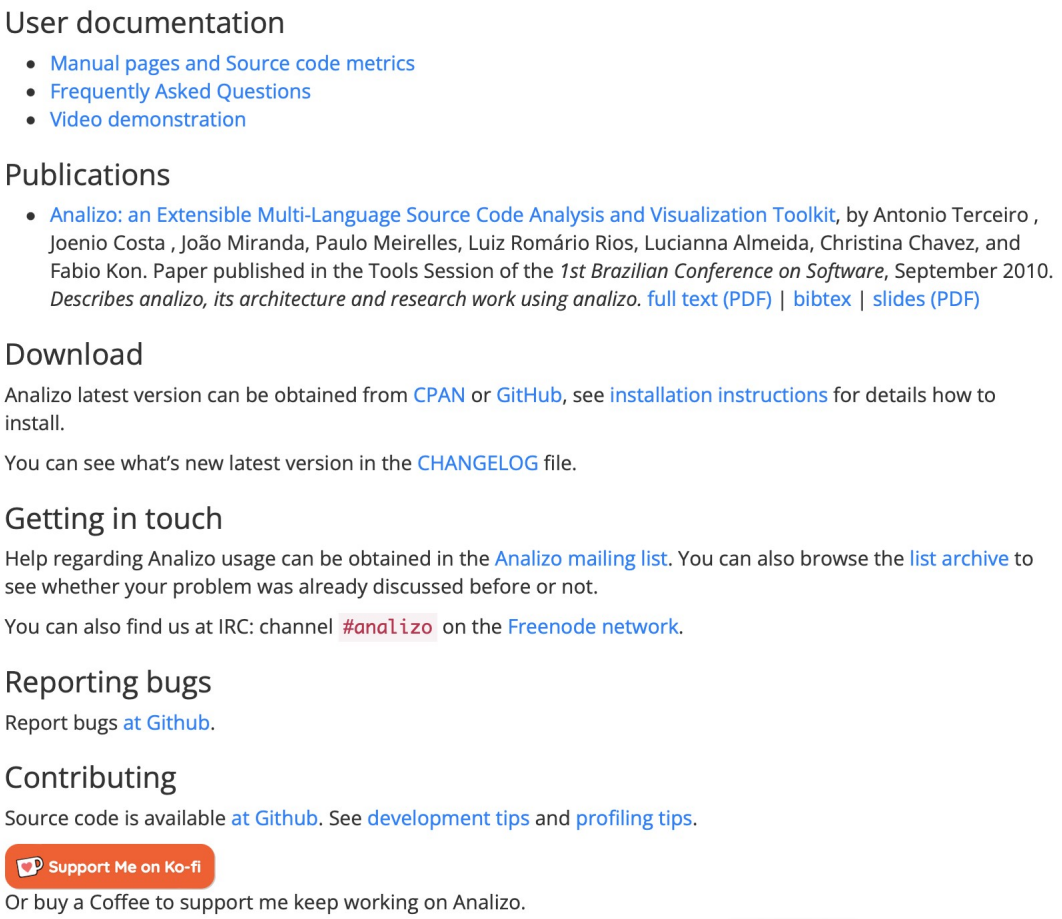


Figura 3.7. Documentação para usuários da ferramenta `Analizo`

O uso contínuo desta prática protege o *software de pesquisa* em relação à introdução de *bugs* em correções ou modificações futuras. *Testes de unidade* são usados para testar métodos, funções, *scripts* e outros tipos de unidades de programa, concentrando-se nas saídas geradas em resposta às entradas e às condições de execução [Delamaro et al. 2013]. Em geral, os *testes funcionais* tratam o sistema como uma "caixa preta", pelo fato de não haver acesso aos detalhes de código para a criação dos casos de teste.

Há diversos *frameworks* disponíveis para criação e *automação de testes* funcionais e de unidade para linguagens de programação popularmente usadas no desenvolvimento de *software de pesquisa*, por exemplo, *JUnit* (http://junit.org/) para Java, *CUnit* (http://cunit.sourceforge.net/) para C, *py.test* (http://pytest.org/) para Python e *PHPUnit* (https://phpunit.de) para PHP.

Exemplo. A ferramenta de análise estática `Analizo` possui um conjunto de testes automatizados. A Figura 3.8 mostra informações sobre como executar seus testes.

Apesar da linguagem PHP possuir um framework para automação de testes, o *PHPUnit*, o software `flosssearch` não possui testes automatizados. A falta de testes pode desestimular os desenvolvedores para consertar, estender ou melhorar o *software de pesquisa*, e desencorajar outros pesquisadores a usá-lo.

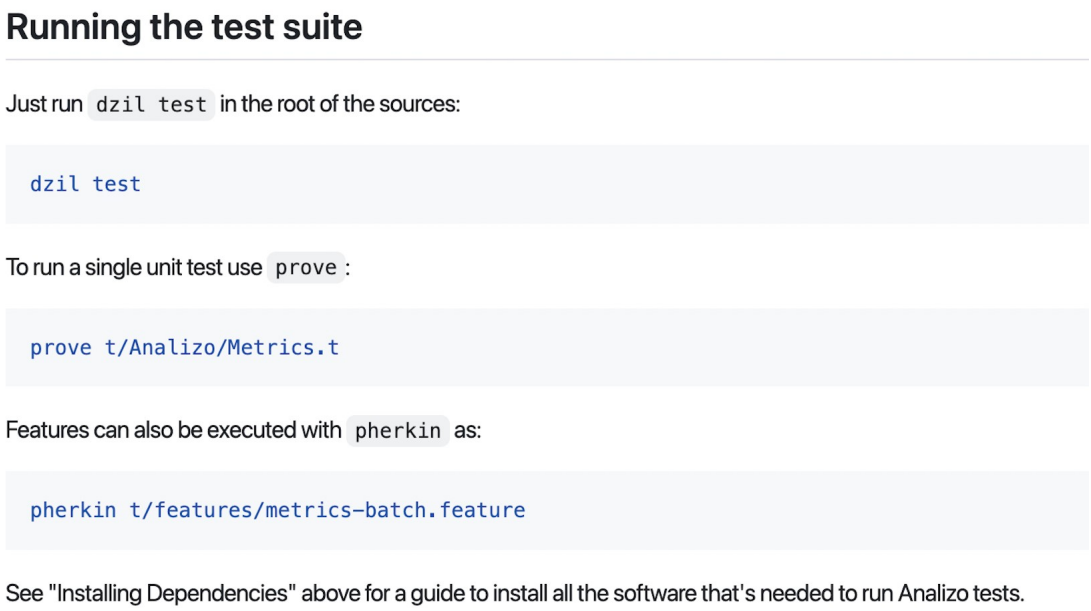


Figura 3.8. Trecho da documentação no GitHub mostrando como executar os testes da ferramenta `Analizo`.

P9. Revisão de código

Revisão de código, ou revisão por pares, é uma prática consolidada para a melhoria da qualidade do software. A colaboração entre pares durante a revisão é facilitada por meio do acesso compartilhado ao código-fonte e vários canais de comunicação. Outros desenvolvedores podem perceber problemas na qualidade do código do *software de pesquisa* que testes automatizados podem não detectar. Os revisores podem identificar alguns tipos de *bugs*, problemas lógicos, falta de padronização e de conformidade com as práticas e diretrizes do projeto.

No contexto do *software de pesquisa*, a revisão de código pode, além de promover a melhoria da qualidade do software, ser aplicada a outros ativos da pesquisa. A revisão de código também permite que outros desenvolvedores e pesquisadores avaliem se os métodos aplicados e implementados no *software de pesquisa* estão em conformidade com as teorias subjacentes da pesquisa. A falta de conformidade entre teoria e implementação em um *software de pesquisa* pode levar a conclusões errôneas em todo um estudo e seus resultados.

Exemplo. Ao ser colocado em um repositório público, o software `flosssearch` foi compartilhado e pôde ser revisado por outros pesquisadores. O texto de apresentação mostrado na tela principal do `flosssearch` em execução, foi revisado por outro pesquisador do grupo e algumas inconsistências foram reportadas e corrigidas.

3.6.4. Gerência

As práticas de gerência apresentadas estão relacionadas aos processos e ferramentas usados para acompanhar e aumentar a eficiência de tarefas de desenvolvimento do *software de pesquisa*.

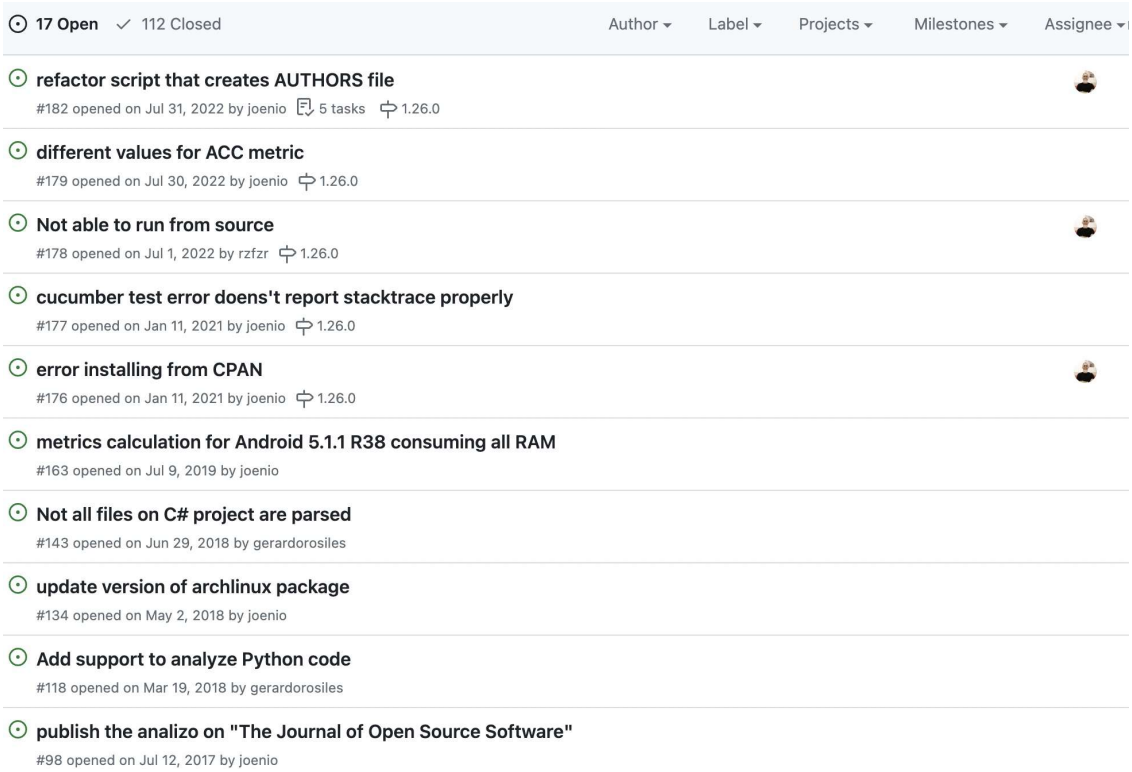


Figura 3.9. *Issue tracker* da ferramenta `Analizo` no GitHub.

P10. *Issue Trackers*

Issue trackers[35] são ferramentas para rastrear e gerenciar problemas, em geral associados a tarefas como resolução de *bugs* e solicitações de melhoria para um software. *Issue trackers* facilitam a colaboração e permitem que os colaboradores acompanhem o progresso das atividades e trabalhem em conjunto para resolvê-las. Para indicar que o trabalho está em andamento, um desenvolvedor pode vincular uma tarefa (*issue*) ao envio de uma solicitação de melhoria no projeto (*pull request*).

Em geral, plataformas de hospedagem de repositórios, como Github, Gitlab[36], Codeberg[37], e Launchpad[38] oferecem um *issue tracker* nativo e integrado. Alternativas estão disponíveis em ferramentas externas, específicas para gestão de tarefas, dentre elas, Jira, Bugzilla, Trello, e Trac.

No contexto de um *software de pesquisa*, o *issue tracker* também pode ser utilizado para acompanhar a escrita colaborativa de artigos científicos em formato texto simples, usando linguagens de marcação (*Markdown* ou para *LaTeX*), para posterior processamento.

Exemplo. A Figura 3.9 apresenta a página do *issue tracker* da ferramenta *Analizo* no

[35]Usamos o termo em inglês, *issue tracker*, ao invés de traduzí-lo para *rastreador de problemas*, dado que o primeiro é amplamente utilizado pela comunidade internacional de desenvolvedores de software.

[36]`https://gitlab.com`

[37]`https://codeberg.org`

[38]`https://launchpad.net`

GitHub. O projeto recebe solicitações de pessoas externas ao projeto e seus desenvolvedores interagem com os *tickets* criados na tentativa de entender o problema ou ajudar na solução. O software `flosssearch` não usa o recurso de *issue tracker* do GitHub, nem ferramentas externas com essa finalidade.

P11. Automatização de tarefas

Durante o desenvolvimento de software muitas tarefas são realizadas de forma repetitiva. A automatização permite que essas tarefas sejam executadas de forma rápida e consistente por meio de *scripts*, ferramentas ou sistemas automatizados. Além dos testes, outras tarefas podem ser automatizadas para ajudar a encontrar e investigar *bugs* mais rapidamente, melhorar a qualidade do software e reduzir o tempo para validação e lançamento de versões do software. A automatização pode ser configurada para executar diferentes tipos de testes (unitários, integração), para analisar o código-fonte em busca de problemas de estilo, convenções de codificação ou más práticas, entre outras tarefas.

Exemplo. A Figura 3.10 mostra um exemplo de automatização de tarefas e apresenta um arquivo com a definição de passos para a execução dos testes unitários do *software de pesquisa* `Parcels`.

P12. Integração e implantação contínuas

Integração Contínua ("Continuous Integration") e *Implantação Contínua* ("Continuous Deployment") são práticas de automação no desenvolvimento de software que possibilitam a entrega rápida e confiável de um software. Na integração contínua, as mudanças são automaticamente verificadas e integradas, permitindo que os problemas sejam detectados e corrigidos mais cedo no processo de desenvolvimento. A implantação contínua permite que as mudanças sejam disponibilizadas automaticamente para os usuários.

Para configurar um sistema de integração contínua é necessária a configuração de uma ferramenta, como *Jenkins*, *CircleCI*, *Travis CI* ou *GitHub Actions*, para monitorar o repositório do software e executar automaticamente testes e tarefas de verificação. A Figura 3.11 mostra um *pull request* no repositório do *software de pesquisa* `Parcels`. O projeto implementa integração contínua e, a cada *pull request* nesse repositório, tarefas automatizadas são executadas, como testes unitários e de integração e uso de uma ferramenta de análise de código-fonte (*linter*) para identificar possíveis problemas no código. Como as tarefas são obrigatórias, o código não poderá ser incorporado ao *branch* principal se alguma das atividades falhar. Na Figura 3.11, podemos observar que os testes unitários falharam.

Os *pipelines* de implantação contínua são fluxos de trabalho automatizados que ajudam a *implantar novas versões do software em diferentes ambientes*. Eles podem ser criados usando ferramentas como *Docker*, *Kubernetes*, *GitLab CI/CD* e *CircleCI*. Para exemplificar a prática, a Figura 3.12 mostra as tarefas automatizadas realizadas quando um *pull request* é incorporado ao projeto na *branch* principal. O projeto é o *software de*

```yaml
name: unit-tests
on:
  push:
    branches:
      - "master"
      - "test-me/*"
  pull_request:
    branches:
      - "*"
  schedule:
    - cron: "0 7 * * 1" # Run every Monday at 7:00 UTC

defaults:
  run:
    shell: bash -el {0}

jobs:
  unit-test:
    name: Unittesting on ${{ matrix.os }} with python latest
    runs-on: ${{ matrix.os }}-latest
    strategy:
      fail-fast: false
      matrix:
        os: [macos, ubuntu, windows]
        include:
          - os: macos
            os-short: osx
          - os: ubuntu
            os-short: linux
          - os: windows
            os-short: win
    steps:
      - name: Checkout
        uses: actions/checkout@v3
      - name: Setup Conda and parcels
        uses: ./.github/actions/install-parcels
        with:
          environment-file: environment_py3_${{ matrix.os-short }}.yml
          environment-name: py3_parcels
      - name: Unit test
        run: |
          coverage run -m pytest -v -s --html=${{ matrix.os }}_unit_test_report.html --self-contained-html tests
          coverage xml
      - name: Codecov
        uses: codecov/codecov-action@v3.1.1
        with:
          flags: unit-tests
      - name: Upload test results
        if: ${{ always() }} # Always run this step, even if tests fail
        uses: actions/upload-artifact@v3.1.2
        with:
          name: Unittest report
          path: ${{ matrix.os }}_unit_test_report.html
```

Figura 3.10. Fluxo de trabalho automatizados no repositório do *software de pesquisa* `Parcels`

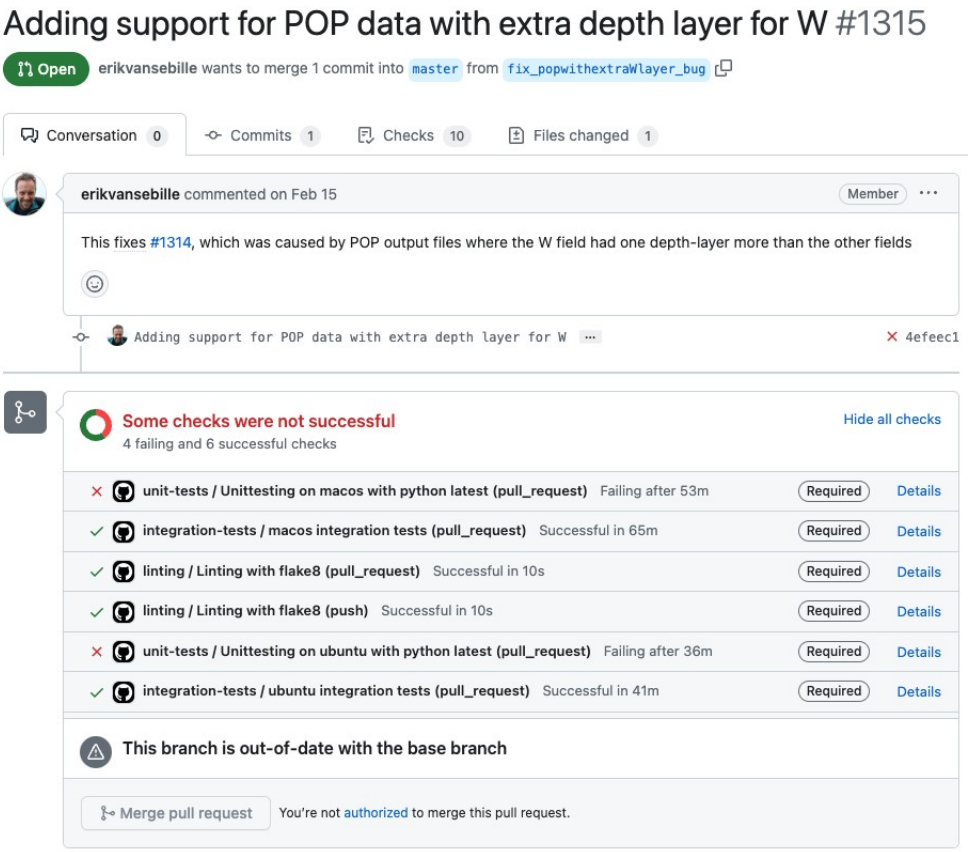


Figura 3.11. *Pull request* com testes unitários falhando no repositório do projeto Parcels

pesquisa GGIR[39], que converte dados brutos de dispositivos vestíveis em relatórios para pesquisadores que investigam a atividade física diária e o sono humano. Após a aprovação do *pull request*, as tarefas automatizadas são executadas e o lançamento da versão.

Ao adotar essas práticas de integração e implantação contínuas no *software de pesquisa*, a confiabilidade do software será maior e a entrega de novas funcionalidades será mais rápida e segura.

P13. Lançamento de versões

O *lançamento de versões* é importante durante o desenvolvimento de *software de pesquisa* sustentável para permitir a rastreabilidade das mudanças e permitir que a pesquisa seja reproduzida a partir de uma versão específica do software. O lançamento de versões envolve a marcação de pontos específicos no histórico do código-fonte para representar marcos no desenvolvimento do software. Os repositórios de código fonte fornecem vários recursos que apoiam o lançamento de versões.

Uma estratégia comum é o uso de *tags* (etiquetas), ou marcadores atribuídos a um *commit* para identificar uma versão específica do software. As *tags* podem ser usadas para

[39] `https://github.com/wadpac/GGIR`

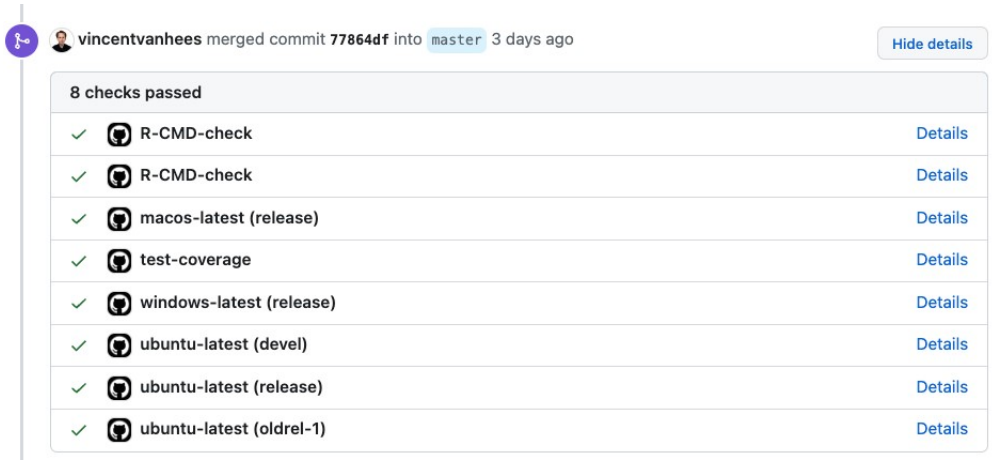


Figura 3.12. Tarefas automatizadas executadas após a incorporação de um *pull request* no repositório do *software de pesquisa*.

identificar versões estáveis, versões de correção de *bugs*, pontos de lançamento importantes ou algum *milestone* (marco) no desenvolvimento do projeto. Por exemplo, uma *tag* pode ser criada para indicar a primeira versão funcional do software ("versão 1.0").

Outro recurso importante é o uso de *releases* (lançamentos), com software empacotado e disponibilizado para os usuários. *Pacote* (*package*) é o termo utilizado para representar um conjunto organizado de arquivos e recursos relacionados que são distribuídos juntos como uma unidade coesa. O pacote pode incluir o código-fonte do software, dependências, arquivos de configuração, documentação e qualquer outro componente necessário para executar ou instalar o software. A disponibilização do *software de pesquisa* como pacote facilita a distribuição, instalação e utilização do software, entregando um conjunto funcional de recursos e funcionalidades para os usuários. Cada *release* pode incluir *release notes* (notas de lançamento) detalhando as alterações, correções de *bugs* e outras informações relevantes.

O nome da versão geralmente segue uma convenção numérica ou alfanumérica, como "1.0", "2.3" ou também pode incluir um identificador descritivo, como "versão beta" ou "versão alpha". O versionamento semântico[40] é uma prática comum fortemente recomendável ao adotar e usar versionamento para gerenciar as *releases* do *software de pesquisa*. O *escopo de uma versão* engloba as alterações e atualizações contidas na versão específica, inseridas pelos *commits* incluídos desde a versão imediatamente anterior até a versão sendo lançada. Recomenda-se incluir na descrição da *release* informações sobre o seu escopo, por exemplo, funcionalidades adicionadas, problemas resolvidos, melhorias de desempenho, atualizações de segurança e outros detalhes relevantes. Essas informações ajudam a comunicar claramente o que uma determinada versão oferece aos usuários.

Exemplo. O *software de pesquisa* `Parcels` está em um repositório público e utiliza as funcionalidades de *tags* que descrevem o escopo da versão nos lançamentos de versões. A Figura 3.13 apresenta a lista de *tags* no repositório e a Figura 3.14 mostra as notas do lançamento de uma versão do projeto.

[40] `https://semver.org/lang/pt-BR`

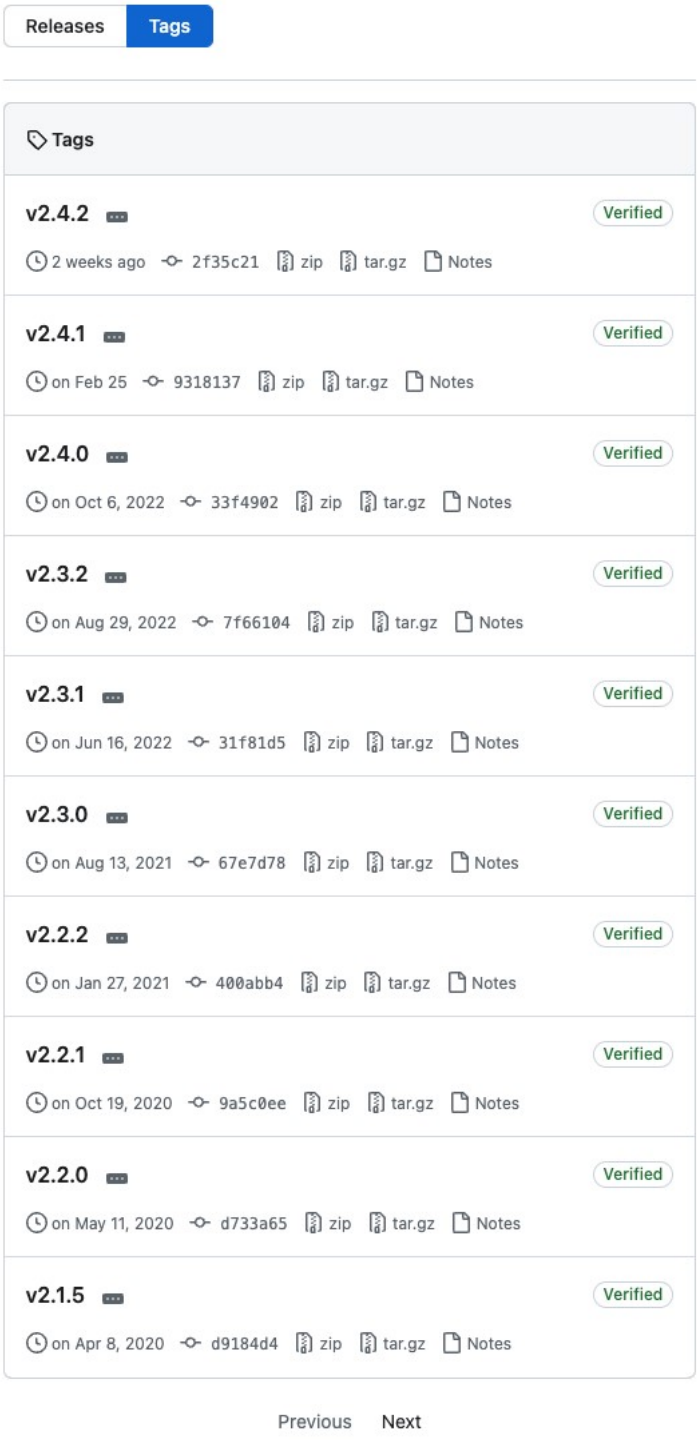


Figura 3.13. Lista de *tags* no repositório do software de pesquisa `Parcels`.

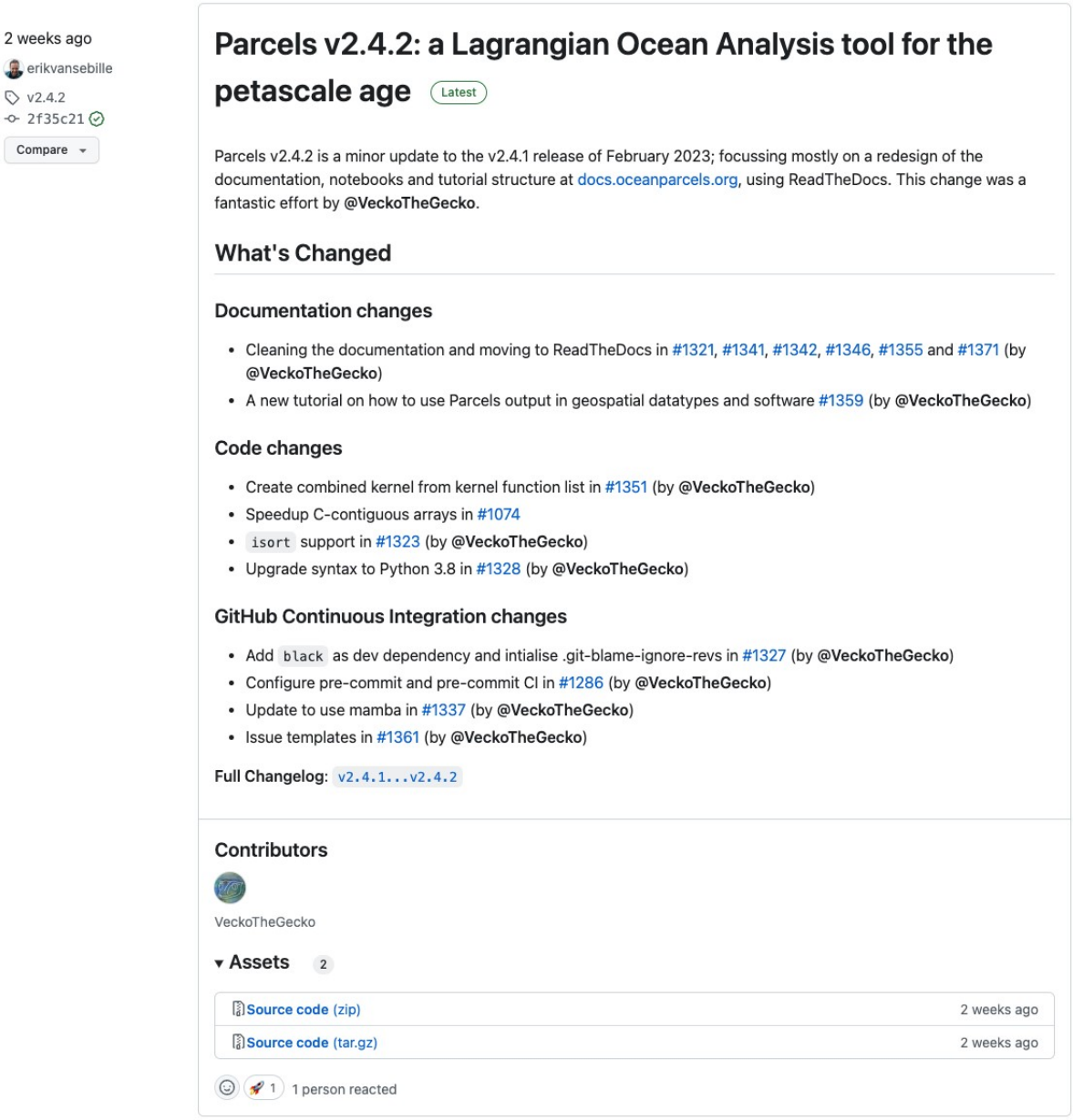


Figura 3.14. Notas de lançamento de uma versão no repositório do software de pesquisa Parcels.

3.6.5. Reconhecimento

As boas práticas que promovem a sustentabilidade técnica do *software de pesquisa* podem atrair usuários e contribuidores para o software. As práticas apresentadas a seguir estão relacionadas com o reconhecimento da relevância do *software de pesquisa* para outros pesquisadores e da necessidade de sua divulgação e citação apropriadas.

P14. Comunidade do projeto

Em geral, o *software de pesquisa* é desenvolvido por indivíduos ou em pequenos grupos de pesquisa dentro de uma única instituição, resultando em protótipos para atender às suas próprias necessidades, por exemplo, validar uma hipótese ou apoiar atividades da pesquisa. Criar uma comunidade em torno de um *software de pesquisa* é essencial para aumentar a adoção e colaboração, ampliando seu impacto na comunidade científica. Além disso, a comunidade ajuda na identificação e correção de problemas e traz novas sugestões para o desenvolvimento do software.

Uma comunidade ativa e engajada promove a colaboração entre pesquisadores e desenvolvedores de diferentes instituições e áreas de pesquisa. Tal engajamento facilita a troca de ideias, a discussão de problemas, a identificação de *bugs*, a proposição de novas funcionalidades, a implementação de melhorias e novas funcionalidades para o software. Outra vantagem da comunidade é realizar revisão por pares do código-fonte, documentação e resultados produzidos pelo software. Essas revisões ajudam a garantir a qualidade e a confiabilidade do software. A diversidade de perspectivas e experiências na comunidade contribui para um processo de desenvolvimento mais completo. Além disso, a existência de uma comunidade ativa em torno do *software de pesquisa* pode aumentar sua sustentabilidade no longo prazo. Com uma base de usuários e desenvolvedores engajados, o software tem maior probabilidade de receber suporte financeiro, recursos e atualizações contínuas, facilitando sua disponibilidade e relevância para outros pesquisadores.

Para envolver usuários e colaboradores em um *software de pesquisa*, é importante estabelecer diretrizes claras, reconhecer e valorizar as contribuições, disponibilizar canais de comunicação, organizar eventos e atividades, fornecer documentação detalhada e acessível e incentivar a colaboração ativa.

As diretrizes que orientam a participação na comunidade abordam aspectos sobre como contribuir com código, como reportar problemas e participar em discussões. Idealmente, o projeto deve disponibilizar um código de conduta e um guia sobre como contribuir. Quando definidas e seguidas, diretrizes ajudam a criar um ambiente colaborativo e respeitoso. Outra estratégia essencial é reconhecer e valorizar as contribuições dos usuários e colaboradores. Com agradecimentos públicos, inclusão de nomes nos créditos do projeto, menções em publicações relacionadas ao *software de pesquisa* e outros gestos de reconhecimento, incentivam-se o engajamento contínuo e a motivação para continuar colaborando com o projeto.

Para facilitar a interação entre os membros da comunidade é fundamental disponibilizar *canais de comunicação*, como fóruns de discussão online, listas de e-mails, salas de bate-papo e plataformas como GitHub e GitLab. Esses canais permitem que

os colaboradores obtenham suporte técnico, compartilhem ideias, troquem experiências e contribuam com melhorias no *software de pesquisa*. Também é possível promover a interação direta entre os membros da comunidade com a organização ou participação em conferências, workshops, treinamentos e eventos.

Para incentivar a colaboração ativa dos usuários é importante a identificação de tarefas específicas que precisam de contribuições, a criação de programas de mentoria para novos colaboradores, orientações claras sobre como contribuir para o projeto. Por fim, é essencial fornecer uma documentação detalhada e acessível para que os colaboradores possam compreender e utilizar o software de forma eficaz.

Exemplo. A ferramenta `flosssearch` foi implementada em PHP, por conveniência e experiência do desenvolvedor, pensando na construção de uma comunidade que, no futuro, poderia transformar `flosssearch` em um produto. Porém não há políticas ou recomendações definidas para os membros da comunidade.

A plataforma GitHub apresenta suas políticas e recomendações para os membros da comunidade[41], incentivando-os a moderar seus projetos sempre que possível e denunciar qualquer conteúdo que possa violar tais políticas.

P15. Divulgação de Software

A divulgação do *software de pesquisa* em eventos científicos e estudos com outros pesquisadores é de extrema importância. Essas oportunidades permitem que os pesquisadores compartilhem e apresentem suas soluções de software, que avaliem a usabilidade da ferramenta e potenciais interesses em comum.

Ao apresentar o *software de pesquisa* para a comunidade acadêmica e colaborar com outros pesquisadores, é possível receber opiniões, estabelecer parcerias, trocar conhecimento e, talvez, aumentar o impacto e a adoção do software de pesquisa, impulsionando o alcance da ciência.

Exemplo. O software `flosssearch` foi utilizado e publicado em artigos científicos, tanto como instrumento da pesquisa, como produto da pesquisa. Entretanto, ainda não há informações documentadas no repositório sobre a correspondência entre artigos publicados e versões do software.

O arquivo README do `Parcels` apresenta uma seção sobre “manuscrito e código”, onde cita artigos publicados e a versão software usada:

> The manuscript detailing the performance of `Parcels v2.4` is available at Computers & Geosciences and can be cited as:
>
> Kehl, C, PD Nooteboom, MLA Kaandorp and E van Sebille (2023). *Efficiently simulating Lagrangian particles in large-scale ocean flows – Data structures and their impact on geophysical applications*, Computers and Geosciences, 175, 105322. https://doi.org/10.1016/j.cageo.2023.105322

[41] `https://docs.github.com/en/site-policy/github-terms/github-community-guidelines`

P16. Citação de Software

A boa prática científica exige que o *software de pesquisa* mencionado em publicações científicas seja mantido para reprodutibilidade e verificação de resultados científicos. Citar um software de pesquisa é uma forma de reconhecer e dar visibilidade aos autores e ao próprio software, possibilitando que outros pesquisadores o localizem e eventualmente reusem. Entretanto, a citação de software em publicações científicas ainda não é uma prática comum entre cientistas, mas padrões de citação têm sido propostos e adotados.

Em seu artigo "Recognizing the value of software: a software citation guide", [Katz et al. 2020] apresentam instruções simples sobre como o software pode se tornar citável e destacam que o uso de identificadores persistentes (PIDs) e metadados descritivos são elementos essenciais da citação de software. Informações sobre autoria, nome, local de publicação, data de publicação e identificador do software são consideradas obrigatórias. Adicionalmente, alguns estilos de citação pedem uma descrição da citação entre colchetes. Para software, pode-se usar *[Computer software]*. A Tabela 3.3 apresenta o formato de citação de software proposto por [Katz et al. 2020].

Tabela 3.3. Formato de Citação de Software [Katz et al. 2020].

Informação	Descrição
Criador(es)	Autor(es) ou projeto que desenvolveram o software
Título	Nome do software
Local	Local de publicação do software, por exemplo, um arquivo ou repositório que fornece identificadores persistentes.
Data	Data em que o software foi publicado, por exemplo, a data associada a um lançamento ou versão do software.
Identificador	Um apontador resolvível para o software, de preferência um PID que leve para uma página contendo metadados descritivos sobre o software, semelhante a um DOI.

O formato *Citation File Format* (CFF)[42] tem se popularizado como formato e padrão para documentar como citar um projeto de software [Druskat 2021]. Adicionar um arquivo *CITATION.cff* ao repositório do *software de pesquisa* pode ser uma boa forma documentar e atualizar as recomendações definidas para citação do software. Recentemente, Zenodo e GitHub facilitaram a adição de metadados aos repositórios e a geração de citações para esses repositórios [Smith 2021].

Além disso, pode-se definir um ID persistente para o *software de pesquisa*, por exemplo, um DOI obtido para um artigo que apresenta o software para a comunidade (*software paper*) ou ainda usar formatos mais recentes como o SWHID[43] promovido pelo projeto Software Heritage[44], uma iniciativa dedicada a coletar e preservar software em formato de código-fonte para preservação e acesso público. Entretanto, [Katz et al. 2020] recomendam que, se existir um artigo que apresenta o *software de pesquisa* para a comu-

[42] https://citation-file-format.github.io
[43] https://www.swhid.org
[44] https://www.softwareheritage.org

nidade científica, ele deve ser citado como uma referência bibliográfica *adicional*, mas não deve substituir a citação do software propriamente dita.

Exemplo. O *software de pesquisa* `flosssearch` não definiu como deve ser citado. A ferramenta `Analizo` indica uma publicação que pode ser considerada como citação para o software [Terceiro et al. 2010] mas não apresenta uma citação de software.

3.7. Avaliação do *software de pesquisa* `MoSyn`

Nesta seção, apresentamos os resultados de um estudo exploratório conduzido com um grupo de pesquisa da área de Física com o objetivo de identificar problemas relacionados ao *software de pesquisa* desenvolvido pelo grupo, fazer uma caracterização inicial da sustentabilidade e aderência aos princípios FAIR de um *software de pesquisa* e recomendar boas práticas para torná-lo mais sustentável e aberto.

3.7.1. Entrevista com Pesquisador

A primeira atividade do estudo foi a realização de uma entrevista com um pesquisador sênior e líder de grupo de pesquisa da área de Física Aplicada, com mais de 30 anos de experiência. Seu interesse no desenvolvimento de *software de pesquisa* é motivado pelo desejo de utilizar a computação como ferramenta para avançar o conhecimento em sua área de pesquisa. As respostas forneceram importantes considerações sobre o uso de práticas de sustentabilidade no *software de pesquisa* desenvolvido por seu grupo.

De forma geral, as opiniões do pesquisador endossam fortemente o uso de *software de pesquisa* sustentável para o grupo de pesquisa. Ele reconhece o valor desse trabalho e o desejo de contribuir para a comunidade de pesquisa em geral. Sua crença nos benefícios da sustentabilidade, aliada ao entusiasmo em tornar o software acessível e ao reconhecimento da importância da replicação, reforça ainda mais o apoio do pesquisador às práticas de *software de pesquisa* sustentável. Ele destaca a necessidade de mais conhecimento por parte dos desenvolvedores do grupo de pesquisa. Os integrantes do grupo valorizariam ter suporte para entender e implementar essas práticas de forma eficaz.

Os resultados também destacam as considerações e dilemas relacionados ao desejo de abertura, à proteção de suas contribuições intelectuais e ao receio de comprometer a integridade e a credibilidade de sua pesquisa ao compartilhá-la prematuramente. A falta de familiaridade dos desenvolvedores de *software de pesquisa* com as melhores práticas de desenvolvimento de código também é considerada uma barreira. O grupo de pesquisa inclui pesquisadores com formação acadêmica em diferentes áreas, sendo que sua *expertise* principal está nas respectivas áreas de pesquisa, e não na engenharia de software.

Durante a entrevista, o pesquisador inicialmente tinha pouco conhecimento sobre as práticas de sustentabilidade em *software de pesquisa* , mas demonstrou entusiasmo e aceitou as práticas apresentadas assim que foram introduzidas, reconhecendo os benefícios e a importância da adoção de práticas de sustentabilidade no desenvolvimento de *software de pesquisa*. Outro fator de apoio identificado é o compromisso com a promoção da reprodutibilidade em sua pesquisa. Com os dados e o código publicamente disponíveis, a pesquisa pode ser replicada e potencialmente expor quaisquer erros ou inconsistências, contribuindo para a confiabilidade e robustez do conhecimento científico como um todo.

A atitude positiva do pesquisador em relação ao *software de pesquisa* sustentável reflete uma apreciação genuína pelos benefícios que ele oferece ao seu trabalho. O pesquisador deseja que seu software esteja disponível para todos, reconhecendo seu potencial para auxiliar inúmeros pesquisadores. Ele enfatiza as aplicações práticas dos resultados da pesquisa em ambientes clínicos, onde o software poderia ser usado para avaliar e tratar pessoas de forma eficaz. Além disso, o pesquisador destaca a importância da replicação na pesquisa e o papel do software sustentável em garantir resultados precisos e confiáveis, reconhecendo que erros e *bugs* são inerentes ao software. Ele e seu grupo de pesquisa estão interessados em identificar e corrigir esses problemas. O pesquisador também reconhece os benefícios dos testes automatizados na garantia de que as novas versões do software sejam confiáveis e na correção de *bugs* anteriores, economizando assim tempo e esforço valiosos. O pesquisador compreende os benefícios de tornar o *software de pesquisa* sustentável e de colaborar para avançar o conhecimento científico.

No final da entrevista, solicitamos que o pesquisador indicasse alguns projetos de *software de pesquisa* para avaliação de sustentabilidade. Os resultados da avaliação, reportados em um relatório técnico, com sugestão de práticas e melhorias, seriam enviados para o pesquisador. Para a primeira avaliação de sustentabilidade, escolhemos o software `MoSyn`[45], um aplicativo para análise de grafos variáveis no tempo.

3.7.2. Avaliação da Sustentabilidade

O *software de pesquisa* `MoSyn` é um aplicativo baseado em MATLAB, projetado para a análise de grafos variantes no tempo (TVGs) e suas medidas associadas. A ferramenta fornece uma estrutura modular com várias classes e funções para lidar com diferentes aspectos da análise, como configuração, recursos gráficos e gerenciamento de projetos.

A Tabela 3.4 apresenta um resumo da avaliação da sustentabilidade do software `MoSyn`. A seguir, apresentamos trechos do *Relatório de Avaliação* do software `MoSyn` com referências às práticas apresentadas na Tabela.

 # Relatório de Avaliação da Sustentabilidade do Software `MoSyn`

Práticas Básicas

O software `MoSyn` esteve hospedado em um repositório público desde o início de seu desenvolvimento (**P1**). Apesar da hospedagem do *software de pesquisa* no GitHub e uso das facilidades da plataforma, um backup periódico tem sido realizado em um dispositivo de armazenamento do grupo de pesquisa, mantido nas instalações do grupo. No GitHub a identidade do software é clara e única: o nome público do software é `MoSyn`. Apesar de ter um arquivo README.md, não há uma descrição do projeto que facilite sua indexação nos mecanismos de busca.

Por estar hospedado no GitHub, `MoSyn` possui suporte para controle de versão (**P2**). Inicialmente, o repositório do projeto não apresentava a licença de software (**P3**). Após quatro meses da realização da entrevista com o pesquisador sênior do grupo, houve um *commit* no repositório do software para a inclusão do arquivo *LICENSE*, descrevendo a licença escolhida, porém sem deixar claro se as permissões se aplicam a qualquer ar-

[45] `https://github.com/mpnetto/MoSyn`

Tabela 3.4. Sustentabilidade do software `MoSyn`.

P	Descrição	Atende?	Comentário
P1	O projeto está hospedado em um repositório público	Sim	O software está disponibilizado em um repositório público no GitHub
P2	O software implementa controle de versão	Sim	O controle de verão é implementado por utilizar o GitHub como plataforma de hospedagem
P3	Uma licença de software foi adotada	Parcialmente	Um arquivo declarando a licença MIT. Porém não está claro se as permissões se aplicam a qualquer arquivo fonte no repositório
P4	O software está publicado formalmente e apresentam um DOI	Não	O repositório não menciona um DOI associado a ele mas pode ser encontrado no GitHub pelo nome
P5	A estrutura de arquivos comunica a finalidade dos elementos do projeto	Sim	A estrutura de pastas está organizada de forma descritiva e permite inferir o conteúdo
P6	Adota formatos de dados e interfaces comuns	Sim	Apesar de não haver documentação explícita, o software utiliza o formato de entrada e saída que facilita a integração com o MATLAB
P7	A documentação apresenta uma visão geral sobre o software	Parcialmente	O projeto utiliza o *GitHub pages* mas as informações estão incompletas e algumas URLs direcionam para um destino não válido.
P8	O software implementa testes	Não	Não há testes automatizados para o software
P9	O código é revisado antes de ser incorporado ao código	Parcialmente	Todos os *pull requests* listados no repositórios foram aprovados pelo próprio autor, sugerindo que não houve revisão de código por outra pessoa
P10	Disponibiliza e usa *issue tracker*	Sim	O projeto aproveita a funcionalidade de rastreamento de *bugs* e tarefas disponível no GitHub
P11	As tarefas repetitivas são automatizadas	Não	Não encontramos tarefas automatizadas no projeto
P12	Há integração e implantação contínua	Não	Não há integração contínua. Por ser um plugin instalado manualmente no MATLAB, a implantação contínua não é viável
P13	O software faz lançamento de versões	Não	O projeto não utiliza a funcionalidade de lançamento de versões no repositório do GitHub
P14	Há evidência de uma comunidade (presente ou futuro)	Não	Há apenas um desenvolvedor como autor
P15	O software é divulgado em eventos científicos	Não	Não encontramos divulgação em eventos científicos
P16	O software é citado em publicações científicas	Não	Não encontramos citação do software em publicações.

quivo fonte do repositório. A licença atribuída ao software `MoSyn` é a MIT[46], um licença usada em projetos de software livre e em projeto de software proprietário. Quanto ao registro do software (**P4**), o repositório não menciona se há um DOI associado a ele. Além disso, não encontramos uma referência para o software `MoSyn` no Zenodo.

Organização do Projeto

O repositório do `MoSyn` possui uma estrutura de arquivos (**P5**) bem definida, e usa nomes auto-explicativos para pastas e arquivos que facilitam a compreensão do propósito e utilidade do *software de pesquisa*. O software `MoSyn` também se preocupa com padronização (**P6**) visto que é uma aplicação que depende do software MATLAB[47], uma plataforma paga para programação e computação numérica usada por engenheiros e cientistas para analisar dados, desenvolver algoritmos e criar modelos. Assim, o `MoSyn` utiliza formatos de entrada e saída que facilitam a integração com o MATLAB.

No `MoSyn`, a preocupação com documentação do software (**P7**) ainda é incipiente e voltada para usuários do software. Inicialmente, o repositório não tinha qualquer documentação. Quatro meses após a entrevista realizada com o pesquisador, um arquivo *README* foi adicionado ao repositório com descrição da ferramenta, lista de funcionalidades, informações sobre utilização e contribuição e licença utilizada pelo software (**P7**). Os autores do software não estão listados em um arquivo, mas é possível identificar as pessoas que contribuíram com o software a partir de informações sobre autores das mudanças registradas pelo sistema de controle de versão (*commits*). Finalmente, os desenvolvedores deram início à construção de um site para publicação de informações sobre o projeto de pesquisa e o software usando o *GitHub Pages*.

Qualidade

O *MATLAB Test*[48] é um conjunto de ferramentas para o desenvolvimento, gerenciamento, análise e teste de aplicações MATLAB. Entretanto, o *software de pesquisa* `MoSyn` ainda não implementa testes de software automatizados (**P8**). Vale destacar que as ferramentas do *MATLAB Test*, assim como o MATLAB, são produtos de software fechados e que requerem assinaturas pagas.

Considerando que o software `MoSyn` está publicado no GitHub, é possível realizar a revisão de código (**P9**) colaborativamente, utilizando a infraestrutura oferecida pela plataforma. Porém, até o dia da avaliação do *software de pesquisa*, todos os pedidos de mudança no código (*pull requests*) listados no repositório foram aprovados pelo próprio autor, sugerindo que não houve revisão de código por outra pessoa.

Gerência

O software `MoSyn` está hospedado no GitHub e conta com o seu rastreador de tarefas e *bugs* nativo (**P10**) nativo. O *GitHub Docs*[49], apresenta uma breve introdução sobre como reportar um problema usando o rastreador de tarefas. Entretanto, o rastreador nativo do GitHub ainda não foi utilizado no projeto `MoSyn`. Não foram encontradas tare-

[46] `https://opensource.org/license/mit/`
[47] `https://www.mathworks.com/products/matlab.html`
[48] `https://www.mathworks.com/products/matlab-test.html`
[49] `https://docs.github.com/pt`

fas automatizadas ou indício de automatização de tarefas (**P11**). Por fim, não há suporte para integração e implantação contínuas (**P12**). Por ser um *plugin* instalado manualmente pelo usuário no MATLAB, tais práticas não são viáveis.

O projeto não segue a prática de lançamento de versões (**P13**) no repositório do GitHub. As funcionalidades de 'Releases' e 'Tags' que facilitariam a referência e a descrição das funcionalidades e bugs incluídos naquela versão específicas não são utilizadas. Apesar de não realizarem lançamentos oficiais de versões, se fosse necessário citar o *software de pesquisa* em um artigo, os autores poderiam fazer referência a um *commit* específico no repositório do projeto. No histórico do projeto vimos que um *commit* com a mensagem "mosyn 2.0" foi incorporado ao projeto quatro meses após a entrevista. Essa mensagem sugere uma intenção de indicar uma alteração significativa no projeto e associar um número de versão.

Reconhecimento

O projeto ainda não possui uma comunidade (**P14**). Apenas o usuário dono do repositório submeteu *commits* e *pull requests* no projeto. A página do projeto no GitHub não apresenta outros usuários que adicionaram o projeto como favorito e não mostra usuários que fizeram uma cópia independente (*fork*) do projeto.

Não encontramos artigos ou outras formas de divulgação do *software de pesquisa* em eventos científicos (**P15**). O software `MoSyn` / MATLAB é mencionado e foi usado em uma dissertação de mestrado na área de Saúde para extrair índices de redes funcionais cerebrais (RFC) para análise estatística [Toutain 2019]. Entretanto, não encontramos no texto da dissertação um identificador ou referência para o software ou para a versão usada. Também não encontramos citação do software em publicações e na dissertação mencionada (**P16**), nem informações sobre como o software deveria ser citado.

3.7.3. Avaliação de *FAIRness*

As Tabelas 3.5, 3.6 e 3.7 apresentam uma avaliação preliminar de *FAIRness* do software `MoSyn`. Cada linha da tabela apresenta um princípio, sua descrição, indicação se o software atende ou não ao princípio, e uma justificativa se procedente. As tabelas mostram apenas as linhas em que o software não atendeu ou atendeu parcialmente ao princípio. A seguir, discutimos como *software de pesquisa* incorporou os princípios *FAIR*.

 # Relatório de Avaliação de *FAIRness* do Software `MoSyn`

F: O software e seus metadados associados são facilmente encontrados tanto por humanos quanto por máquinas

O software `MoSyn` está hospedado com uma identidade clara e única no GitHub, não globalmente e o repositório não garante a persistência do identificador se o software for movido, por exemplo (**F1**). Os componentes do software, como classes e bibliotecas, apresentam identificadores distintos (**F1.1**). Cada versão do software pode ser unicamente identificada por um *hash de commit* e, a partir dele, é possível recuperar os metadados da versão específica (**F1.2**). O software apresenta metadados, mas não descreve as dependências, informações detalhadas sobre utilização e configuração e como deve ser a entrada e saída de arquivos (**F2**). Ao analisar os metadados a partir de um commit, é possível

verificar qual versão específica do software o metadado está se referindo (**F3**). Apesar de incluir um arquivo README com informações do projeto, não há uma descrição na configuração do projeto que facilite a indexação nos mecanismos de busca (**F4**).

A: O software e seus metadados podem ser obtidos através de protocolos padronizados

O software `MoSyn` pode ser obtido a partir do repositório do projeto no GitHub (**A1**). Não há restrições para baixar o código-fonte, como taxas os custos relacionados à licença (**A1.1**). É possível configurar o repositório para ser acessado apenas por pessoas autorizadas (**A1.2**). Os metadados estão descritos em arquivos no repositório, então ele não seriam acessíveis caso o repositório do software não esteja mais disponíveis (**A2**).

I: O software interopera com outros software trocando dados e/ou metadados através da interação via interface de programação de aplicativos (APIs), descrito por meio de padrões

O software `MoSyn` é uma aplicação para MATLAB e, portanto, interopera seguindo seu padrão, mas a forma como a interação ocorre não está explicitamente descrito (**I1**). O repositório menciona o MATLAB como pré-requisito para executar a aplicação e inclui agradecimentos pela utilização de bibliotecas e recursos externos e recursos, mas não inclui referências qualificadas, como websites ou os nomes da bibliotecas e recursos externos utilizados (**I2**).

R: O software é tanto utilizável (pode ser executado) quanto reutilizável (pode ser compreendido, modificado, aprimorado ou incorporado a outros softwares)

O software `MoSyn` não disponibiliza muitas informações descrevendo como reutilizar o software (**R1**). O software está licenciado sob a licença de código aberto MIT, que permite a reutilização, mas não deixa claro se todas as partes do software seguem a mesma licença (**R1.1**). A partir dos commits e da lista de contribuidores do projeto no GitHub é possível saber quais pessoas contribuíram com o software, mas não há informações explícitas sobre como o software foi desenvolvido ou quais foram as intenções originais (**R1.2**). Os nomes e informações sobre as bibliotecas utilizadas não são mencionadas na documentação (**R2**). A comunidade MATLAB recomenda que sejam seguidas as melhores práticas de desenvolvimento de software em relação a correção, clareza e generalização e o software, em geral, atende a esses padrões (**R3**).

3.8. Considerações Finais

O reconhecimento da necessidade de reprodutibilidade na pesquisa científica, bem como o advento recente da criação de comitês de avaliação de artefatos de pesquisa em conferências e periódicos levaram à recomendação para que os autores disponibilizem dados de pesquisa, *software de pesquisa*, documentação e materiais usados nas publicações científicas. Nesse contexto, o *software de pesquisa* desenvolvido deve ser sustentável para que cientistas possam entender, replicar e reproduzir pesquisas anteriores ou conduzir novas pesquisas de forma eficaz. O suporte à reprodutibilidade ainda demanda que o *software de pesquisa* seja *FAIR* – localizável, acessível, interoperável e reutilizável, para responder às necessidades de pesquisa, presentes e futuras.

Tabela 3.5. *FAIRness* no Software MoSyn – *Findable*.

Princ.	Descrição	Atende?	Comentário
F:	O software e seus metadados associados são facilmente encontrados tanto por humanos quanto por máquinas.	Parcialmente	
F1	O software recebe um identificador globalmente único e persistente.	Parcialmente	O software possui uma identidade única apenas no GitHub e não é garantida a persistência.
F1.1	Aos componentes do software que representam diferentes níveis de granularidade são atribuídos identificadores distintos.	Sim	
F1.2	As diferentes versões do software recebem identificadores distintos.	Sim	
F2	O software é descrito com metadados detalhados.	Parcialmente	O software menciona alguns metadados, mas sem muitos detalhes que permitam encontrar o software facilmente
F3	Os metadados contêm de forma clara e explícita o identificador do software que descrevem.	Parcialmente	Os metadados não apresentam de forma clara, mas é possível obter a partir do *commit*
F4	Os metadados seguem os princípios FAIR, são pesquisáveis e indexáveis.	Parcialmente	Há informações nos arquivos do repositório, mas não há descrição na configuração do projeto para facilitar indexação

Entretanto, grande parte dos cientistas que escrevem *software de pesquisa* ainda carecem de uma formação em boas práticas de engenharia de software, por exemplo, uso de sistemas de controle de versão, documentação adequada e testes automatizados. Neste cenário, a sustentabilidade e outros atributos de qualidade do *software de pesquisa* podem ficar comprometidos e levar a erros em conclusões científicas e falta de reprodutibilidade na pesquisa que depende do software.

Portanto, ainda que demande tempo e esforço, uma mudança na forma como o desenvolvimento e a manutenção de *software de pesquisa* são realizados é necessária, bem como a definição de um modelo de sustentabilidade de software que reconheça a importância de incorporar práticas para promover a sustentabilidade no desenvolvimento de *software de pesquisa*. Um caminho natural é investir em treinamento para cientistas de diversas áreas do conhecimento sobre conceitos, boas práticas e ferramentas para o desenvolvimento de *software de pesquisa* sustentável.

Ao adotar práticas sustentáveis, os pesquisadores podem criar oportunidades para que outros especialistas na área colaborem no desenvolvimento de seu *software de pesquisa*, forneçam *feedback* e sugestões de melhorias. Além disso, a disseminação dos resultados da pesquisa e a divulgação do software, podem ampliar o impacto além da comunidade acadêmica, beneficiando o público em geral.

Tabela 3.6. FAIRness no Software MoSyn – *Accessible*.

Princ.	Descrição	Atende?	Comentário
A:	O software e seus metadados podem ser obtidos através de protocolos padronizados.	Parcialmente	
A1	O software é pode ser obtido por meio de seu identificador utilizando um protocolo de comunicação padronizado.	Sim	
A1.1	O protocolo é aberto, gratuito e universalmente implementável.	Sim	
A1.2	O protocolo permite procedimentos de autenticação e autorização, quando necessário.	Sim	
A2	Os metadados são acessíveis, mesmo quando o software não está mais disponível.	Não	Caso o software não esteja mais disponível, os metadados também ficarão inacessíveis

A discussão sobre *software de pesquisa* sustentável é incipiente e o tema tem sido pouco explorado pela comunidade acadêmica brasileira de Computação. O Simpósio Brasileiro de Engenharia de Software (SBES 2022) foi inovador ao apresentar um documento delineando Políticas de Ciência Aberta para o evento [Flach 2022], buscando encorajar os autores a disponibilizar os artefatos usados na pesquisa – dados abertos anonimizados e código do *software de pesquisa*, e despertar o interesse na reprodutibilidade dos estudos quantitativos publicados. O tema do SBES 2022 foi *Sustentabilidade de Software*, e o título de uma das palestras convidadas, proferida pelo Prof. Daniel Katz, foi *Towards Sustainable Research Software* [Katz 2022].

3.8.1. Institutos e Associações

Há várias iniciativas que contemplam a questão da *Sustentabilidade do Software de Pesquisa* e o apoio a cientistas e engenheiros de software, com o objetivo de promover a adoção de boas práticas no desenvolvimento de *software de pesquisa* e torná-lo mais sustentável. Alguns países da Europa, América do Norte, Ásia e África têm inaugurado institutos nacionais para apoiar cientistas que desenvolvem *software de pesquisa* e preparar uma nova categoria de profissionais: os engenheiros de *software de pesquisa* [Jiménez et al. 2017]. Entretanto, o Brasil ainda não investiu na criação de organizações ou institutos dedicados a este tema.

Dentre os institutos e associações pioneiros, destacam-se (1) *The Carpentries*[50], (2) *Software Sustainability Institute* (SSI)[51], (3) *Research Software Engineers* (RSE)[52], (4) *The Society of Research Software Engineering*[53] e (5) *Netherlands eScience Center* [Drost et al. 2020].

[50] https://carpentries.org
[51] https://www.software.ac.uk
[52] https://researchsoftware.org/
[53] https://society-rse.org

Tabela 3.7. FAIRness no Software MoSyn – *Interoperable, Reusable*.

Princ.	Descrição	Atende?	Comentário
I:	O software interopera com outros software trocando dados e/ou metadados através da interação via interface de programação de aplicativos (APIs), descrito por meio de padrões.		
I1	O software lê, escreve e troca dados de acordo com padrões da comunidade relacionados ao domínio.	Parcialmente	Não está explícito como a troca de dados ocorre com o MATLAB
I2	O software inclui referências qualificadas a outros objetos.	Parcialmente	As referências se resumem a menção sobre utilização, sem citar nomes, websites ou detalhes dos outros objetos
R:	O software é utilizável (pode ser executado) e reutilizável (pode ser compreendido, modificado, aprimorado ou incorporado a outros softwares).	Parcialmente	
R1	O software é descrito com uma variedade de atributos precisos e relevantes.	Parcialmente	Não há muitas informações sobre atributos e como reutilizar o software
R1.1	É atribuída uma licença clara e acessível ao software.	Sim	*MIT License.*
R1.2	O software possui informações detalhadas de procedência.	Parcialmente	Não há informações sobre como o software foi desenvolvido ou quais foram as intenções originais
R2	O software inclui referências qualificadas a outros softwares.	Parcialmente	Não há informações relevantes sobre dependências.
R3	O software atende aos padrões relevantes da comunidade do domínio.	Sim	

3.8.2. Cursos e Treinamentos

Institutos, associações e outras organizações oferecem cursos e treinamentos gratuitos sobre diversos temas ligados ao *software de pesquisa* e sua sustentabilidade. Recomendamos que visitem os websites e consultem o material disponibilizado.

- *The Carpentries* é um instituto que oferece treinamento em habilidades computacionais e ciência de dados para cientistas em todo o mundo.
- *Software Carpentry* é um projeto voluntário dedicado a ensinar habilidades básicas de computação para pesquisadores [Munk et al. 2019, Nenadic et al. 2022].
- *Library Carpentry* é uma comunidade global que ensina habilidades de software e dados para pessoas que trabalham em funções relacionadas a bibliotecas e informações [Munk et al. 2019].

- *Netherlands eScience Center* é uma fundação independente baseada na Holanda para desenvolvimento e aplicação de software de pesquisa [Drost et al. 2020].
- *The Turing Way* oferece um manual para ciência de dados reprodutível, ética e colaborativa [Community 2022].

Referências

[Abdo 2015] Abdo, A. H. (2015). Direções para uma academia contemporânea e aberta. In *Ciência Aberta, Questões Abertas*, pages 287–306. IBCT-Instituto Brasileiro de Informação em Ciência e Tecnologia.

[Allen et al. 2017] Allen, A. et al. (2017). Engineering Academic Software (Dagstuhl Perspectives Workshop 16252). *Dagstuhl Manifestos*, 6(1):1–20.

[Anzt et al. 2021] Anzt, H. et al. (2021). An environment for sustainable research software in germany and beyond: current state, open challenges, and call for action [version 2; peer review: 2 approved]. *F1000Research*, 9(295).

[Barker et al. 2022] Barker, M., Chue Hong, N. P., Katz, D. S., and Lamprecht, A.-L. (2022). Introducing the FAIR Principles for Research Software. *Scientific Data*, 9(622).

[Becker et al. 2015] Becker, C., Chitchyan, R., Duboc, L., Easterbrook, S., Mahaux, M., Penzenstadler, B., Rodriguez-Navas, G., Salinesi, C., Seyff, N., Venters, C., Calero, C., Kocak, S. A., and Betz, S. (2015). The Karlskrona Manifesto for Sustainability Design.

[Bezjak et al. 2018] Bezjak, S. et al. (2018). *Open Science Training Handbook, doi: 10.5281/zenodo.1212496*. Zenodo.

[Carver et al. 2022] Carver, J., Weber, N., Ram, K., Gesing, S., and Katz, D. (2022). A Survey of the state of the practice for Research Software in the United States. *PeerJ Comput. Science*.

[Carver et al. 2007] Carver, J. C., Kendall, R. P., Squires, S. E., and Post, D. E. (2007). Software development environments for scientific and engineering software: A series of case studies. In *Proceedings of the 29th International Conference on Software Engineering*, ICSE '07, page 550–559, USA. IEEE Computer Society.

[Chue Hong et al. 2022] Chue Hong, N. P. et al. (2022). FAIR Principles for Research Software (FAIR4RS Principles), doi: 10.15497/RDA00068.

[Community 2022] Community, T. T. W. (2022). The Turing Way: A handbook for reproducible, ethical and collaborative research.

[de Souza 2023] de Souza, A. C. M. (2023). Social sustainability approaches for a sustainable software product. *ACM SIGSOFT Softw. Eng. Notes*, 48(1):38–43.

[Delamaro et al. 2013] Delamaro, M., Jino, M., and Maldonado, J. (2013). *Introdução ao Teste de Software*. Elsevier Brasil.

[Drost et al. 2020] Drost, N. et al. (2020). *Netherlands eScience Center - Software Development Guide*. Zenodo.

[Druskat 2021] Druskat, S. (2021). Making software citation easi(er) - The Citation File Format and its integrations.

[Flach 2022] Flach, C. (2022). Políticas de Ciência Aberta do Simpósio Brasileiro de Engenharia de Software (SBES).

[Flach and Kon 2021] Flach, C. and Kon, F. (2021). Software livre: Pré-requisito para a ciência aberta (pt-br). *Computação Brasil*, 1(46):12–15.

[Forschungsgemeinschaft 2022] Forschungsgemeinschaft, D. (2022). Guidelines for Safeguarding Good Research Practice. Code of Conduct. Available in German and in English.

[Goble 2014] Goble, C. (2014). Better software, Better Research. *IEEE Internet Computing*, 18(5):4–8.

[Goble et al. 2016] Goble, C., Howison, J., Kirchner, C., Nierstrasz, O., and Vinju, J. J. (2016). Engineering Academic Software (Dagstuhl Perspectives Workshop 16252). *Dagstuhl Reports*, 6(6):62–87.

[Gruenpeter et al. 2021] Gruenpeter, M. et al. (2021). Defining Research Software: a Controversial Discussion.

[Hannay et al. 2009] Hannay, J., MacLeod, C., Singer, J., Langtangen, H., Pfahl, D., and Wilson, G. (2009). How do scientists develop and use scientific software? In *ICSE Workshop on Software Engineering for Computational Science and Engineering*, pages 1–8.

[Hermann and Fehr 2022] Hermann, S. and Fehr, J. (2022). Documenting Research Software in Engineering Science. *Scientific Reports*, 12(6567).

[Hettrick et al. 2014] Hettrick, S. et al. (2014). UK Research Software Survey 2014, doi: 10.5281/zenodo.14809.

[Howison et al. 2015] Howison, J., Deelman, E., McLennan, M. J., Ferreira da Silva, R., and Herbsleb, J. D. (2015). Understanding the scientific software ecosystem and its impact: Current and future measures. *Research Evaluation*, 24(4):454–470.

[Howison and Herbsleb 2011] Howison, J. and Herbsleb, J. D. (2011). Scientific software production: incentives and collaboration. In *Proc. of the ACM 2011 conference on Computer supported cooperative work*, pages 513–522.

[Howison and Herbsleb 2013] Howison, J. and Herbsleb, J. D. (2013). *Incentives and integration in scientific software production*, pages 459–470. ACM.

[IEC 2014] IEC, I. (2014). ISO/IEC 25010: Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE. *Switzerland: ISO*.

[Jay et al. 2021] Jay, C., Haines, R., and Katz, D. S. (2021). Software Must be Recognised as an Important Output of Scholarly Research. *Int. Journal of Digital Curation*, 16(1):6. Number: 1.

[Jiménez et al. 2017] Jiménez, R. C., Kuzak, M., Alhamdoosh, M., Barker, M., Batut, B., Borg, M., Capella-Gutierrez, S., Chue Hong, N., et al. (2017). Four simple recommendations to encourage best practices in research software. *F1000Research*, 6:876.

[Katz 2014] Katz, D. (2014). Transitive credit as a means to address social and technological concerns stemming from citation and attribution of digital products. *Journal of Open Research Software*, 2(1).

[Katz et al. 2020] Katz, D., Chue Hong, N., et al. (2020). Recognizing the value of software: a software citation guide.

[Katz 2022] Katz, D. S. (2022). Towards sustainable research software.

[Katz et al. 2016] Katz, D. S. et al. (2016). Report on the third workshop on sustainable software for science: Practice and experiences (WSSSPE3). *CoRR*, abs/1602.02296.

[Kehrer and Penzenstadler 2018] Kehrer, T. and Penzenstadler, B. (2018). An exploration of sustainability thinking in research software engineering. In *Proc. of the 7th Int. Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy 2018)*, volume 2223 of *CEUR Workshop Proc.*, pages 34–43. CEUR-WS.org.

[Knuth 1984] Knuth, D. E. (1984). Literate Programming. *The Computer Journal*, 27(2).

[Lamprecht et al. 2020] Lamprecht, A.-L. et al. (2020). Towards FAIR Principles for Research Software. *Data Science*, 3(1):37–59.

[Merali 2010] Merali, Z. (2010). Computational science: Error, why scientific programming does not compute. *Nature*, 467(7317):775–777.

[Momcheva and Tollerud 2015] Momcheva, I. and Tollerud, E. (2015). Software use in astronomy: an informal survey. *arXiv preprint arXiv:1507.03989*.

[Mourão et al. 2018] Mourão, B. C., Karita, L., and do Carmo Machado, I. (2018). Green and Sustainable Software Engineering – a Systematic Mapping Study. In *Proc. of the XVII Brazilian Symposium on Software Quality*, SBQS '18, page 121–130, New York, NY, USA. ACM.

[Munk et al. 2019] Munk, M., Koziar, K., Leinweber, K., Silva, R., Michonneau, F., et al. (2019). swcarpentry/git-novice: Software Carpentry: Version Control with Git, June 2019.

[Nenadic et al. 2022] Nenadic, A. et al. (2022). carpentries-incubator/python-intermediate- development: beta.

[Nieuwpoort and Katz 2023] Nieuwpoort, R. v. and Katz, D. S. (2023). Defining the roles of research software. *Upstream*.

[Rajlich and Bennett 2000] Rajlich, V. and Bennett, K. (2000). A staged model for the software life cycle. *Computer*, 33(7):66–71.

[Segal and Morris 2008] Segal, J. and Morris, C. (2008). Developing scientific software. *IEEE Software*, 25:18–20.

[Smith 2021] Smith, A. (2021). Enhanced support for citations on github.

[Smith et al. 2017] Smith, A. M. et al. (2017). Journal of open source software (JOSS): design and first-year review. *CoRR*, abs/1707.02264.

[Sufi et al. 2020] Sufi, S. et al. (2020). Report on the Workshop on Sustainable Software Sustainability 2019 (WOSSS19). Technical report, Zenodo.

[Terceiro et al. 2010] Terceiro, A., Costa, J., Miranda, J., Meirelles, P., Rios, L. R., Almeida, L., Flach, C., and Kon, F. (2010). Analizo: an extensible multi-language source code analysis and visualization toolkit. In *Brazilian Conference on Software: Theory and Practice (CBSoft) – Tools*, Salvador-Brazil.

[Toutain 2019] Toutain, T. G. L. d. O. (2019). Avaliação da estabilidade cerebral e conexões intra e inter-hemisféricas na modulação afetiva da dor. Master's thesis, Universidade Federal da Bahia, (PPGPIOS), Brasil.

[UNESCO 2021] UNESCO (2021). UNESCO Recommendation on Open Science.

[Venters et al. 2017] Venters, C. et al. (2017). Software sustainability: Research and practice from a software architecture viewpoint. *Journal of Systems and Software*, 138.

[Venters et al. 2021] Venters, C. et al. (2021). Software Sustainability: Beyond the Tower of Babel. In *BoKSS 2021*. Publisher: figshare.

[Venters et al. 2014] Venters, C., Jay, C., et al. (2014). Software sustainability: The Modern Tower of Babel. *CEUR Workshop Proceedings*, 1216:7–12.

[Versen 2020] Versen (2020). Manifesto on Software Research and Education in the Netherlands. versen.nl. `https://www.versen.nl/assets/manifesto/manifesto_software_onderzoekers_06-2.pdf`. [Accessed 10-Jun-23].

[Wilkinson et al. 2016] Wilkinson, M. D. et al. (2016). The FAIR Guiding Principles for Scientific Data Management and Stewardship. *Scientific Data*, 3.

[Wilson et al. 2017] Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., and Teal, T. K. (2017). Good enough practices in scientific computing. *PLOS Computational Biology*, 13(6):1–20.

Capítulo 4

Bancos de dados em memória e suas estratégias de recuperação após falhas

Arlino Magalhães, Ângelo Brayner and José Maria Monteiro

Abstract

In-memory database systems have proven to be an alternative for systems that need massive real-time data processing. In-memory systems keep the database in main memory to provide low latency and high throughput. However, due to memory volatility, these systems are more sensitive to failures than traditional disk databases. Although the components of disk and in-memory databases look similar, these two systems differ greatly in the way they implement their components. This short course provides an overview of the architecture and implementations of in-memory databases and their main recovery strategies after failures. To achieve this goal, this short course provides an overview of in-memory database technology, reviews the concepts of recovery after failures, presents the main architectural choices for implementing in-memory databases and, finally, describes the recovery strategies implemented by a representative sample of modern in-memory databases.

Resumo

Os sistemas de bancos de dados em memória têm se mostrado como uma alternativa para sistemas que precisam de processamento massivo de dados em tempo real. Os sistemas em memória mantêm o banco de dados em memória principal para prover baixa latência e altas taxas de vazão. Contudo, devido à volatilidade da memória, esses sistemas são mais sensíveis a falhas do que os tradicionais bancos de dados em disco. Embora os componentes dos bancos de dados em disco e em memória pareçam similares, esses dois sistemas diferem muito na maneira como implementam os seus componentes. Esse minicurso provê uma visão geral da arquitetura e implementação de bancos de dados em memória e suas principais estratégias de recuperação após falhas. Para atingir essa meta, esse minicurso fornece uma visão geral da tecnologia de bancos de dados em memória, revisa os conceitos de recuperação após falhas, apresenta as principais escolhas

arquiteturais para implementação de bancos de dados em memória e, por fim, descrever as estratégias de recuperação implementadas por uma amostra representativa dos bancos de dados em memória modernos.

4.1. Introdução

Vários cenários de aplicações atuais têm necessitado de um processamento massivo de dados e/ou em tempo real. Os bancos de dados em memória principal têm se mostrado como uma alternativa para tais sistemas. Os sistemas em memória mantêm os dados em memória principal e, por isso, possuem baixa latência e altas taxas de vazão de dados. Essa abordagem fornece muitas implicações importantes em como os componentes desses sistemas devem ser implementados. Por exemplo, os bancos de dados em disco tentam otimizar o acesso ao disco, enquanto os bancos de dados em memória tentam otimizar o acesso à memória [Malviya et al. 2014, Gruenwald et al. 1996, Mohan and Levine 1992].

Bancos de dados em memória são utilizados tipicamente em sistemas estilo OLTP (*Online Transaction Processing* ou Processamento de Transações *Online*). Sistemas OLTP comumente fazem muitas modificações em poucas partes do banco de dados por meio de operações de curta duração. Em contraste, sistemas OLAP (*Online Analytical Processing* ou Processamento Analítico *Online*) tipicamente possuem operações de mais longa duração que manipulam e analisam um grande volume de dados. Eles são usados para dar suporte a *business intelligence*, por exemplo. Comumente, cargas de trabalho OLTP e OLAP são executadas em bancos de dados diferentes: transacional e *data warehouse*, respectivamente. Existe ainda os sistemas HTAP (*Transaction/Analytical Processing* ou Processamento Analítico e de Transações) que precisam de percepções analíticas dos dados mais atuais do banco de dados. Para suportar HTPA, alguns sistemas de bancos de dados em memória permitem cargas de trabalho OLTP e OLAP no mesmo banco de dados [Arulraj et al. 2016a, Copeland and Khoshafian 1985]. A Seção 4.3.1 discute como os sistemas em bancos de dados em memória que suportam HTPA.

A principal vantagem dos sistemas em memória (armazenar os dados na memória principal) também é a principal desvantagem desses sistemas. A volatilidade da memória principal torna os bancos de dados em memória mais sensíveis a falhas que não acontecem nos tradicionais bancos de dados em disco. Por exemplo, todo o banco de dados é perdido em caso de uma falta de energia. Tanto os bancos de dados em disco como os em memória implementam técnicas de *logging*, *checkpoint* e recuperação para prover tolerância a falhas. Contudo, esses dois sistemas diferem muito na maneira como implementam essas técnicas [Faerber et al. 2017, Mohan et al. 1992, Härder and Reuter 1983].

Bancos de dados em memória são pouco abordados em livros e cursos de bancos de dados. Apesar das várias pesquisas e publicações, a recuperação após falhas de bancos de dados ainda é pouco compreendida pela comunidade. Esse minicurso visa elucidar os principais aspectos relacionado a bancos de dados em memória, principalmente os relacionados a recuperação após falhas. Para atingir essa meta, esse minicurso discute o restante dos tópicos como descrito a seguir.

A Seção 4.2 faz uma breve introdução aos sistemas de bancos de dados em memória, um pequeno histórico do desenvolvimento desses sistemas e motivações para o uso deles. Além disso, são discutidas algumas tecnologias emergentes que impulsionaram o

desenvolvimento dos sistemas em memória.

A Seção 4.3 descreve as principais escolhas arquiteturais utilizadas pelos bancos de dados em memória, tais como: armazenamento e organização dos dados, indexação, controle de concorrência e durabilidade e recuperação após falhas. Os tópicos dessa seção são discutidos comparando os bancos de dados em memória com os em disco.

A Seção 4.4 apresenta com detalhes os principais conceitos de recuperação após falhas, focando em bancos de dados em disco. Ela discute o tradicional algoritmo de recuperação ARIES e suas variantes. Além disso, é apresentada a técnica de recuperação instantânea de bancos de dados através de *log* indexado. O objetivo da seção é fornecer os conceitos teóricos básicos necessários para as seções seguintes.

A Seção 4.5 descreve com detalhes as técnicas de *logging*, *checkpoint* e recuperação utilizadas pelos bancos de dados em memória para prover tolerância a falhas. A Seção 4.6 apresenta as principais estratégias implementadas pelas técnicas de tolerância a falhas utilizadas por uma amostra significativa de bancos de dados em memória modernos. Por fim, a seção 4.7 apresenta os principais desafios e direções futuras da pesquisa e desenvolvimento em bancos de dados em memória cujo objetivo é orientar outros pesquisadores.

4.2. Bancos de dados em memória

Os Sistemas de Gerenciamento de Bancos Dados - SGBDs (*Database Management Systems* - DBMSs) tradicionais em disco armazenam o banco de dados na memória secundária (*secondary memory*), tais como o disco rígido (*Hard Disk Drive* - HDD) e a unidade em estado sólido (*Solid State Drive* - SSD). Em contraste, os bancos de dados em memória (*Main Memory Databases* - MMDBs ou *In-Memory Databases* - IMDBs) são sistemas cujos dados residem na memória principal (main memory), como a memória RAM (*Random Access Memory* ou Memória de Acesso Aleatório) [Tan et al. 2015, Zhang et al. 2015a, Garcia-Molina and Salem 1992].

A Figura 4.1 (a) ilustra de forma simplificada a arquitetura de um banco de dados em disco. Esse sistema utilizam um mecanismo de *buffering* que copia os dados da memória secundária para a memória principal para que, em seguida, esses dados sejam processados pela CPU (*Central Processing Unit* ou Unidade Central de Processamento). Se necessário, atualizações nos dados devem ser copiadas de volta para a memória secundária. Adicionalmente, esses sistemas mantêm uma cópia do banco de dados em arquivo de *log* para fins de tolerância a falhas [Faerber et al. 2017, Härder and Reuter 1983].

Os bancos de dados em memória (Figura 4.1 (b)) mantêm os dados primariamente e permanentemente na memória principal, eliminando o gargalo de Entrada/Saída - E/S (*Input/Output* - I/O) da memória secundária. Essa abordagem faz esses sistemas muito mais rápidos do que os sistemas em disco. Porém, devido à volatilidade da memória RAM, esses sistemas devem manter uma cópia do banco de dados nos arquivos de *log* e *checkpoint* para prover durabilidade e tolerância a falhas [Faerber et al. 2017, Zhang et al. 2015a, Härder and Reuter 1983].

O cenário dos sistemas de gerenciamento de dados está cada vez mais fragmentado com base em domínios de aplicações. Existem SGBDs em memória relacionais comerciais, como: SAP HANA [Färber et al. 2012], VoltDB [Malviya et al. 2014], Ti-

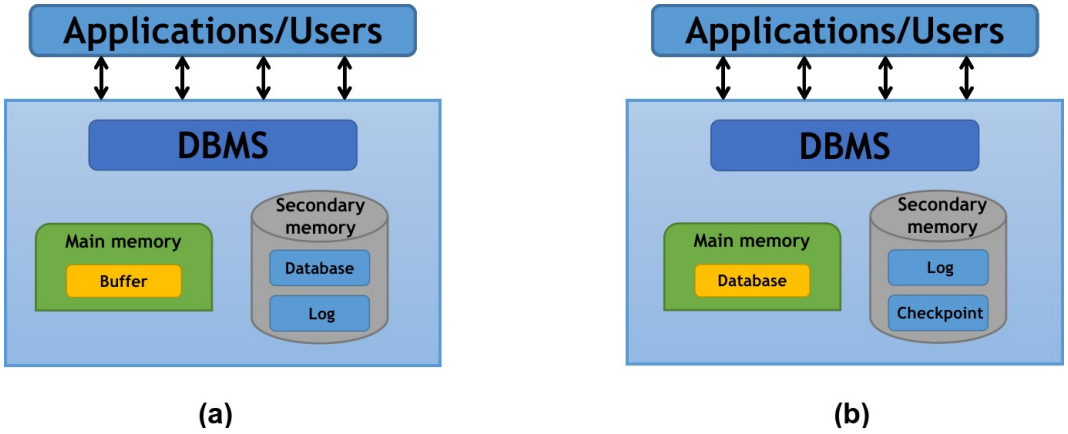


Figura 4.1. Arquiteturas dos bancos de dados em disco (a) e em memória (b).

mesTen [Lahiri et al. 2013], SolidDB [Lindström et al. 2013], Hekaton [Larson et al. 2013], Calvin [Thomson et al. 2012] e MemSQL [SingleStore 2022]. Existem também alguns bancos de dados em memória relacionais acadêmicos/*open-source*, tais como: H-Store [Kallman et al. 2008], HyPer [Funke et al. 2014], Silo [Tu et al. 2013], MySQL Cluster NDB [MySQL 2022] e Peloton [Pavlo et al. 2017].

Bancos de dados NoSQL têm se mostrado como uma alternativa mais viável para escalabilidade de bancos de dados, principalmente quando distribuídos em *clusters* [Sadalage and Fowler 2019, Magalhães et al. 2017]. Assim, bancos de dados em memória têm sido desenvolvidos para as categorias NoSQL, como os bancos chave-valor (por exemplo, Redis [Redis 2020], RAMCloud [Ousterhout et al. 2009], MemepiC [Zhang et al. 2015b] e Pilaf [Mitchell et al. 2013]), os orientados a documento (por exemplo, MongoDB [MongoDB Inc. 2022] e Couchbase [Lim et al. 2014]) e os orientados a grafo (por exemplo, Neo4J [Neo4J 2023], Infinite Graph [Infinite Graph 2023] e OrientDB [OrientDB 2023] .

Com o surgimento da era *Big Data*, houve a necessidade de processar quantidades massivas de dados em pequenos intervalos de tempo, com suporte a serviços com latência muito baixa e análise de dados em tempo real. Nesse contexto, foram desenvolvidos bancos de dados em memória para análise de dados que focam no processamento em lote, tais como: Spark [Bishop et al. 2011], SINGA [Ooi et al. 2015], Pregel [Malewicz et al. 2010], GraphLab [Low et al. 2012], Mammoth [Shi et al. 2015], Phoenix [Yoo et al. 2009] e GridGain [GridGain Team 2022]. Além disso, foram desenvolvidos bancos de dados em memória para processamento de dados em tempo real, como: Storm [Apache Software Foundation 2022], Yahoo! S4 [Neumeyer et al. 2010], Spark Streaming [Zaharia et al. 2013] e MapReduce Online [Condie et al. 2010].

4.2.1. Pequeno histórico

A pesquisa e o desenvolvimento de bancos de dados em memória surgiu no início da década de 80, quando foram publicados os primeiros artigos na área, como DeWitt et al. 1984, Hagmann 1986, Eich 1986, Eich 1987a, Lehman and Carey 1987 e Eich 1987b. Em geral, a maioria desses trabalhos focou em melhorar o desempenho dos bancos de dados em disco, assumindo que todo (ou quase todo) o banco de dados cabia na memória

principal. São exemplos de bancos de dados em memória desenvolvidos nessa época: IMS/Fast Path [Strickland et al. 1982], MARS MMDB [Eich 1987b], System M [Salem and Garcia-Molina 1990], TPK [Li and Naughton 1988], OBE [Bitton et al. 1987] e HALO [Garcia-Molina and Salem 1992]. Entretanto, o preço elevado e a capacidade limitada da memória RAM, nessa época, tornou a aplicação de sistemas em memória uma solução inviável para a grande maioria dos problemas.

Nos anos noventa, os avanços na tecnologia geraram novamente interesse na pesquisa em bancos de dados em memória. Dois fatores impulsionaram a volta do interesse em sistemas em memória: preço/capacidade da memória RAM e paralelismo *multicore*. Atualmente, os servidores modernos possuem CPUs que proveem estratégias de processamento em paralelo e podem armazenar um banco de dados inteiro (ou uma parte significativa) em memória a um preço razoável. Além disso, muitos dos servidores contemporâneos possuem vários *sockets*, onde podem ser conectados muitos *terabytes* de DRAM e processadores com centenas de núcleos. Além disso, outras tecnologias recentes têm sido utilizadas para explorar melhor o potencial dos sistemas em memória, tais como: NUMA, SIMD, RDM, HTM e NVRAM [Magalhães et al. 2021b, Zhang et al. 2015a, Garcia-Molina and Salem 1992].

Muitos dos trabalhos desenvolvidos na década de 80 sobre sistemas em memória foram invalidados com as melhorias tecnológicas da década de 90. Os bancos de dados em memória da década de 90 começaram a refletir as tendências e arquiteturas dos sistemas em memória atuais. Alguns produtos comerciais emergiram nesse período sendo empregados em aplicações de desempenho crítico, como telecomunicações, por exemplo. São exemplos desses sistemas de bancos de dados: Dali/DataBlitz [Jagadish et al. 1994], ClustRa [Hvasshovd et al. 1995], TimesTen [Lahiri et al. 2013] e P*Time [Cha and Song 2004].

A volta do interesse em sistemas em memória trouxe uma nova onda de pesquisa e desenvolvimento em bancos de dados em memória. A maioria das empresas desenvolvedoras/vendedoras de bancos de dados atuais possui uma solução de banco de dados em memória, como o TimesTen da Oracle e o Hekaton da Microsoft. Surgiram também *startups*, como VoltDB [VolDB 2022] e MemSQL [SingleStore 2022] (chamado atualmente de SingleStore). O resultado dessas pesquisas e desenvolvimento é um novo tipo de sistema de banco de dados cujo projeto difere radicalmente quando comparado com os tradicionais sistemas em disco. A sessão 4.3 discute a arquitetura e as principais escolhas de implementação de bancos em memória.

4.2.2. Motivação para o uso de bancos de dados em memória

A tecnologia de SGBDs em memória tem se mostrado como uma eficiente alternativa para situações que exigem alto desempenho e/ou visão em tempo-real devido à baixa latência e alta vazão de dados desses sistemas. São exemplos de aplicações contemporâneas com essas exigências: *trading*, publicidade, jogos, previsão do tempo, análise de *big data*, etc. [Magalhães et al. 2018b, Zhang et al. 2015a, Garcia-Molina and Salem 1992].

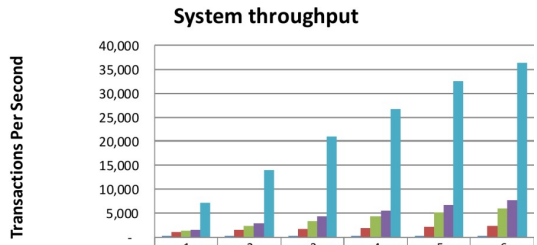
Os sistemas em disco podem não ser capazes de oferecer tempos de resposta aceitáveis para sistemas de desempenho crítico devido à alta latência de acesso à memória secundária. Esse cenário foi inicialmente percebido por empresas de *internet* como Ama-

zon, Google, Facebook e Twitter. Contudo, atualmente outras empresas/organizações têm encontrado esse mesmo obstáculo para fornecer serviços com tempos de resposta aceitáveis. Por exemplo, muitas empresas comerciais precisam detectar uma mudança súbita nos preços de negociação para reagir instantaneamente (em poucos milissegundos) [Magalhães et al. 2018a, Zhang et al. 2015a, Hazenberg and Hemminga 2011].

Melhorias em *hardware* têm provido memórias com alta capacidade de armazenamento a baixo custo. Nas últimas décadas, os *chips* de memória têm dobrado em capacidade de armazenamento a cada 3 anos, enquanto seus preços têm caído em um fator de 10 a cada 5 anos. Como consequência, usar bancos de dados em memória principal para aplicações que tradicionalmente usam bancos de dados em disco tornou-se tecnicamente possível e economicamente viável [Faerber et al. 2017, Zhang et al. 2015a].

O desenvolvimento recente de novas tecnologias têm fornecido ganhos de desempenho com baixa sobrecarga para os sistemas em memória. São exemplos dessas tecnologias: arquitetura NUMA, instruções SIMD, redes RDMA, memória transacional em *hardware* e memória RAM não volátil. Essas tecnologias alavancaram o desenvolvimento de SGBDs em memória [Magalhães et al. 2021b, Faerber et al. 2017, Zhang et al. 2015a]. A seção 4.2.3 detalha o uso dessas tecnologias em sistemas em memória.

Para ilustrar o potencial do uso de bancos de dados em memória, a Figura 4.2 apresenta um experimento de escalabilidade entre o banco em memória Microsoft Hekaton e o tradicional banco em disco Microsoft SQL Server. Em ambos os sistemas, a base de dados estava armazenada inteiramente na memória principal. Os resultados mostram que, por exemplo, com 12 *cores*, Hekaton teve uma vazão de 36.375 transações por segundo, o que correspondeu a um ganho de desempenho quase 16 vezes maior do que o SQL Server, que executou apenas 2.312 transações por segundo. Esse resultado é devido ao fato dos componentes arquiteturais do SQL Server serem projetados para tentar otimizar o acesso à memória secundária, mesmo com a base de dados armazenada inteiramente na memória principal. Por outro lado, os componentes de Hekaton tentam otimizar o acesso à memória principal [Magalhães et al. 2021b, Diaconu et al. 2013]. A seção 4.3 discute os componentes arquiteturais de bancos de dados.



	1	2	3	4	5	6
Number of cores	2	4	6	8	10	12
SQL with contention	984	1,363	1,645	1,876	2,118	2,312
SQL without contention	1,153	2,157	3,161	4,211	5,093	5,834
Interop	1,518	2,936	4,273	5,459	6,701	7,709
Native	7,078	13,892	20,919	26,721	32,507	36,375

Figura 4.2. Experimentos de escalabilidade entre o banco de dados em memória Hekaton e o banco de dados em disco SQL Server [Diaconu et al. 2013].

4.2.3. Tecnologias emergentes para bancos de dados em memória

Embora os sistemas em memória tenham surgido nos anos 80, esses sistemas não se consolidaram como uma solução viável devido ao alto preço e limitações de *hardware* da

época. Soluções de *hardware*/arquitetura de sistemas recentes têm sido exploradas para melhoras de ganho de desempenho. Essa seção discute brevemente tecnologias emergentes que alavancaram o desenvolvimento dos bancos de dados em memória [Magalhães et al. 2021b, Magalhaes et al. 2022, Magalhaes et al. 2023, Faerber et al. 2017].

4.2.3.1. Arquitetura NUMA

NUMA (*Non-Uniform Memory Access* ou Acesso não Uniforme a Memória) é uma arquitetura cujo acesso do processador a memória não é uniforme. Nessa arquitetura, o processador pode acessar uma memória local com latência mínima e memórias remotas com latência maior. A Figura 4.3 ilustra uma arquitetura NUMA com 4 nós. Nesse exemplo, no fluxo (1), o processador do *socket* 0 faz um acesso à memória local, que possui latência mínima. No fluxo (2), esse mesmo processador faz um acesso à memória remota no *socket* 3, que possui uma latência maior. Através dessa abordagem, NUMA permite aumentar a largura de banda e o tamanho total de memória que podem ser implementados em um servidor [Li et al. 2013, Zhang et al. 2015a].

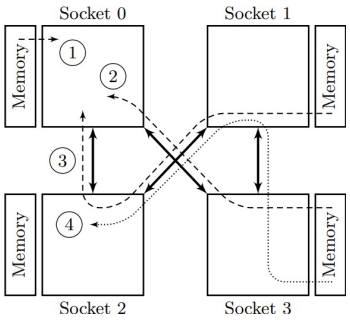


Figura 4.3. Arquitetura NUMA com 4 *sockets* [Li et al. 2013].

Os sistemas de bancos de dados modernos devem ser projetados considerando o uso de NUMA. As pesquisas de bancos de dados em NUMA tentam, por exemplo, minimizar o acesso remoto de dados [Maas et al. 2013, Leis et al. 2014a] e gerenciar os efeitos do acesso NUMA em cargas de trabalho sensíveis à latência (OLTP, por exemplo) [Porobic et al. 2012, Porobic et al. 2014].

4.2.3.2. Instruções SIMD

SIMD (*Single Instruction Multiple Data* ou Única Instrução Múltiplos Dados) é um conjunto de instruções que está presente nos processadores atuais e provê uma fácil alternativa para execução paralela ao nível de dados. Nessa abordagem, uma mesma instrução pode aplicar várias operações simultaneamente a diversos dados para produzir vários resultados. A Figura 4.4 (a) mostra a execução de uma instrução com uma única operação sobre os dados X_0 e Y_0 para produzir o resultado Z_0. Em contraste, a Figura 4.4 (b) mostra a execução de uma instrução SIMD sobre n dados X e n dados Y para produzir n resultados Z. As n operações são executadas em paralelo no processador. Como desvantagens, uma

instrução SIMD possui um limite de paralelismo permitido e restrições na estrutura de dados em que pode operar [Willhalm et al. 2009, Neumann 2011].

Um projeto eficiente de bancos de dados deve considerar paralelismo ao nível de dados. Instruções SIMD podem acelerar operações caras de bancos de dados, como junções e ordenações [Chhugani et al. 2008,Neumann 2011]. O banco de dados SAP HANA, por exemplo, possui um esquema de vetor que utiliza instruções SIMD para acelerar operações de *scan* (varredura) [Willhalm et al. 2013].

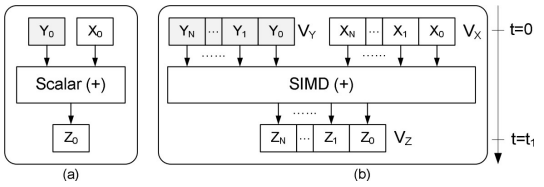


Figura 4.4. Modo escalar (a): uma operação produz um resultado. Modo SIMD (b): uma operação produz vários resultados [Willhalm et al. 2009].

4.2.3.3. Redes RDMA

RDMA (*Remote Direct Memory Access* ou Acesso Remoto Direto à Memória) permite a transmissão de dados entre as memórias de dois computadores sem envolver os sistemas operacionais das duas máquinas. Diferentemente da tradicional rede *ethernet*, um cliente pode acessar a memória de um servidor diretamente, sem a coordenação desse último. Essa transmissão de dados não requer o uso de CPU, *cache* e troca de contexto. Como o servidor utiliza menos recursos, ele pode direcioná-los para outros fins [Mitchell et al. 2013].

RDMA permite um grande tráfego de dados com baixa latência. Porém, essa estratégia é ineficiente na coordenação e sincronização de múltiplos acessos à memória remota [Mitchell et al. 2013]. O bando de dados FaRM implementa leituras livres de bloqueio através de RDMA [Dragojevic et al. 2014]. HyPer faz *backups* do banco de dados via RDMA para livrar o servidor do trabalho de transmissão de dados [Kemper and Neumann 2011].

4.2.3.4. Memória transacional em *hardware*

HTM (*Hardware Transactional Memory* ou Memória Transacional em *Hardware*) provê um mecanismo de controle de concorrência, semelhante ao mecanismo utilizado pelas transações de banco de dados, que visa aumentar o paralelismo e minimizar conflitos. A Figura 4.5 esboça os benefícios de HTM em relação a outros mecanismos de controle de concorrência implementados em SGBDs em memória. Nesse exemplo, HTM (em vermelho) teve uma vazão de transações superior à execução serial (em amarelo) e ao bloqueio de duas fases (em verde). HTM apresenta um desempenho próximo ao do esquema de execução serial em particionamento estático otimizado (em azul), que frequentemente é muito difícil de implementar [Herlihy and Moss 1993, Karnagel et al. 2014, Leis et al. 2014b, Makreshanski et al. 2015]. A Seção 4.3.3 discute os protocolos de controle de concorrência com mais detalhes.

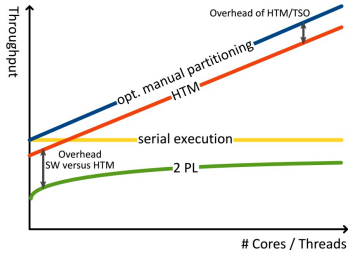


Figura 4.5. HTM *versus* bloqueio em duas fases, execução serial e particionamento estático [Leis et al. 2014b].

A ideia básica de HTM é usar a *cache* como *buffer* local de transação para prover isolamento e detectar conflitos. Para ilustrar essa abordagem, suponha que a Transação 1 (Figura 4.6 (a)) e a Transação 2 (Figura 4.6 (b)) são executadas simultaneamente. Essas duas transações possuem operações em conflito, uma vez que elas atualizam o mesmo objeto a. Durante a execução, as duas transações registram os valores a serem armazenados na memória principal em *cache* temporariamente. Os conflitos são verificados e tratados na hora do *commit* (finalização) da transação. Caso ocorra algum conflito, apenas uma das transações é finalizada. A outra transação deve ser forçada a abortar e reiniciar [Leis et al. 2014b].

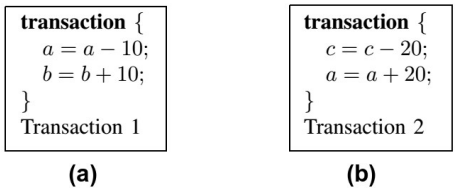


Figura 4.6. Duas transações com operações em conflito [Leis et al. 2014b].

HTM é semelhante ao mecanismo de controle de concorrência implementado pelos SGBDs. Contudo, HTM permite que transações executem com menos sobrecarga, uma vez que a concorrência é gerenciada pelo *hardware*. HTM possui algumas desvantagens: as transações devem ter o tamanho limitado ao da *cache* e transações podem ser abortadas devido a falsos conflitos, troca de contexto, interrupções ou *page faults* [Hammond et al. 2004, Harris et al. 2007]. Os bancos de dados HyPer e DBX, por exemplo, suportam HTM para controle de concorrência [Leis et al. 2014b, Wang et al. 2014].

4.2.3.5. Memória RAM não volátil

NVRAM (*Non-Volatile Random Access Memory* ou Memória RAM não volátil) é uma tecnologia que possui a alta velocidade e endereçamento ao nível de *bytes* da RAM com a persistência e grande capacidade de armazenamento do disco. Contudo, as arquiteturas existentes para bancos de dados em disco e em memória são inapropriados para um banco de dados em NVRAM. Essas duas primeiras utilizam uma estratégia de propagação de dados entre as memórias principal e secundária para fins de durabilidade. Além disso, os dados são mantidos em duas cópias (banco de dados e *log*) com o objetivo de prover tolerância a falhas. Essas estratégias de propagação e duplicação de dados são desnecessárias

em um banco projetado em NVRAM e, ainda, podem causar degradação no desempenho do sistema. As características de NVRAM podem remover os custos causados por essas duas estratégias [Arulraj et al. 2015, Arulraj and Pavlo 2017].

Embora existam muitas pesquisas que descrevem a arquitetura de um SGBD em NVRAM (por exemplo, van Renen et al. 2018, Oukid et al. 2017, Arulraj and Pavlo 2017, Harris 2016, Arulraj et al. 2016b, Garg et al. 2015, Arulraj et al. 2015, [Schwalb et al. 2015], Zhang et al. 2015c, Chen et al. 2011a), ainda não está claro qual o melhor projeto para um SGBD que utiliza NVRAM. Bancos de dados em NVRAM não são o foco desse minicurso e, por isso, não serão mais discutidos.

4.3. Projeto de bancos de dados em memória

Uma suposição simples de como implementar um banco de dados em memória seria ter um SGBD em disco com memória principal suficiente para armazenar a base de dados inteira. Essa estratégia melhora o desempenho do sistema, pois diminui o acesso à memória secundária. Porém, apenas trocar a camada de armazenamento do disco para a memória não torna um SGBD em disco tão eficiente quanto um SGBD em memória. Os SGBDs em disco focam em otimizar o acesso à memória secundária. Por outro lado, os SGBDs em memória são projetados para tentar otimizar o acesso à memória principal. Essa abordagem fornece muitas implicações importantes em como os componentes arquiteturais dos SGBDs em memória devem ser implementados [Magalhães et al. 2021b, Magalhaes et al. 2022, Magalhaes et al. 2022]. As abordagens de projeto utilizadas pelos SGBDs em memória são diversas. Essa seção discute as principais escolhas de implementação e componentes arquiteturais dos SGBDs em memória.

4.3.1. Armazenamento de dados

Os bancos de dados em disco utilizam um mecanismo de *buffer* para acessar registros na memória secundária. Quando é necessário acessar um registro no banco de dados, o gerenciador de *buffer* verifica se a página do banco de dados que contém esse registro está na memória. Caso contrário, ele deve copiar o bloco (ou blocos) do disco, onde a página está armazenada, para a memória principal. A página é a unidade de leitura/gravação de informação no banco de dados e pode conter várias linhas de dados. Uma página possui uma representação semelhante ao bloco do disco para evitar traduções entre as duas representações. Com a página na memória principal, o gerenciador de *buffer* ainda deve fazer um acesso indireto ao registro através dos seguintes passos: (1) encontrar a página no *buffer* e (2) calcular o deslocamento necessário para acessar o registro na página [Härder and Reuter 1983, Ramakrishnan and Gehrke 2003].

A Figura 4.7 esquematiza o funcionamento do gerenciador de *buffer*. O *buffer* funciona como um quadro de páginas na memória principal. Ele ainda implementa algum mecanismo para acessar as páginas mais rapidamente na memória principal, como uma tabela *hash*. O gerenciador de *buffer* é uma solução simples e elegante. Através desse mecanismo, os outros componentes do SGBD abstraem como os dados são acessados no disco. Contudo, o acesso indireto de registros no *buffer* pode afetar o desempenho dos sistemas em memória. Os SGBDs em memória não são dependentes do formato dos blocos de disco. Assim, as suas representações de dados são implementadas de forma a

levar ao melhor desempenho no sistema [Magalhães et al. 2021b, Faerber et al. 2017, Hazenberg and Hemminga 2011].

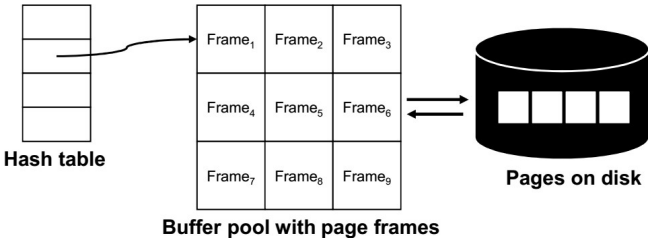


Figura 4.7. Gerenciador de *buffer* em um SGBD em disco [Faerber et al. 2017].

Os bancos de dados em memória comumente implementam ponteiros para acesso direto a memória. Como os ponteiros evitam a sobrecarga do acesso indireto a registros, o sistema utiliza menos ciclos de CPU [Larson and Levandoski 2016,Faerber et al. 2017]. O uso de ponteiros pode melhorar o acesso à memória em ordens de magnitude. Experimentos realizados no IBM Starburst evidenciam essa melhoria. Os experimentos compararam os componentes de banco de dados em memória e em disco desse sistema. Os resultados mostraram que o gerenciador de *buffer* foi responsável por até 40% do tempo de execução de uma consulta, mesmo com as bases de dados dos dois componentes armazenadas inteiramente na memória principal [Lehman et al. 1992].

4.3.1.1. Particionamento de dados

Alguns SGBDs, como H-Store, VoltDB e Calvin, usam um esquema de particionamento de dados. Uma partição divide um banco de dados em partes disjuntas e independentes. As transações são executadas serialmente em partições de dados. Essa estratégia tem a vantagem de uma transação poder executar com acesso exclusivo aos recursos do sistema sem a sobrecarga do controle de concorrência, uma vez que a transação executa sozinha nas partições que precisa. Transações diferentes ainda podem executar em paralelo em partições diferentes usando núcleos de CPU diferentes. Além disso, partições podem ser armazenadas em máquinas diferentes, permitindo ao sistema possuir mais memória do que seria disponível em apenas uma máquina [Faerber et al. 2017, Taft et al. 2014, Thomson et al. 2012].

A desvantagem do esquema de particionamento ocorre quando uma transação precisa acessar mais de uma partição e alguma delas não está disponível. Assim, a transação deve esperar até que todas as partições que precisa estejam disponíveis. Além disso, essa estratégia precisa de balanceamento de carga em partições muito frequentemente acessadas. Em contraste, as transações em sistemas não participados (por exemplo, Hekaton, SAP HANA, MemSQL e TimesTen) podem acessar qualquer parte do banco de dados [Funke et al. 2014, Malviya et al. 2014, Thomson et al. 2012].

A Figura 4.8 ilustra um esquema de tabelas particionadas. As tabelas *A*, *B* e *C* estão divididas nas partições *X*, *Y* e *Z*. Por exemplo, a tabela *A* está dividida em partes *A'*, *A"* e *A"'* armazenadas nas partições *X*, *Y* e *Z*, respectivamente. Consultas podem executar em paralelo em partições diferentes. Por exemplo, enquanto uma consulta está percorrendo *A'* em *X*, outra consulta pode percorrer *A"* em *Y* ao mesmo tempo. Porém,

uma consulta que precise percorrer a tabela *A* inteira deve esperar até que as três partições estejam disponíveis.

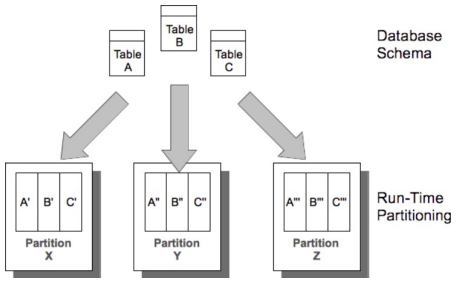


Figura 4.8. Esquema de particionamento de tabelas de bancos de dados [VoltDB Documentation 2022].

4.3.1.2. Versionamento de dados

Muitos SGBDs em memória implementam o multi versionamento de dados, como Hekaton, HyPer e SAP HANA. Nessa abordagem, uma modificação em um dado não sobrescreve o seu conteúdo. A modificação é escrita para outra versão do dado, chamada de versão sombra (*shadow version*), em um endereço de memória diferente do original. Quando a modificação é finalizada, a versão sombra se torna a versão atual do banco de dados. A versão anterior permanece no banco de dados, sendo removida apenas posteriormente por algum sistema de coleta de lixo (*garbage collection*) [Chen et al. 2011b, Malviya et al. 2014, Neumann et al. 2015].

O versionamento de dados facilita a implementação de protocolos de controle de concorrência sem bloqueios. Os protocolos de bloqueio podem ser uma fonte de sobrecarga para sistemas em memória. A Seção 4.3.3 explica os mecanismos de controle de concorrência em mais detalhes. O versionamento permite ainda uma maneira fácil de criar *backups* do banco de dados para fins de tolerância a falhas [Chen et al. 2011b, Malviya et al. 2014, Neumann et al. 2015]. A Seção 4.5 explica melhor como os sistemas em memória proveem tolerância a falhas. O versionamento de dados também facilita a criação de sistemas HTAP [Arulraj et al. 2016a, Copeland and Khoshafian 1985].

A Figura 4.9 esboça um esquema de armazenamento multi versionado. Esse exemplo representa uma tabela simples de contas bancárias. As versões dos dados podem ser acessadas por meio de uma tabela *hash*. Por exemplo, o *bucket J* da *hash* aponta para uma lista que contém quatro versões, mas apenas duas delas são válidas. As colunas *Begin* (início) e *End* (fim) da tabela indicam o intervalo de tempo de validade de cada versão de um registro. Assim, a versão (Jonh, Londom, 100) começou a ser válida no tempo 10 e se tornou obsoleta no tempo 20. O registro (Jonh, Londom, 130) é uma versão válida atual, pois seu tempo final de validade possui o valor *inf*, de *infinite* (infinito) [Diaconu et al. 2013].

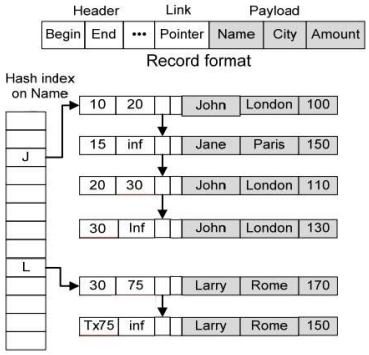


Figura 4.9. Esquema de versionamento de dados [Diaconu et al. 2013].

4.3.1.3. Organização de dados

Os SGBDs em memória podem ser orientados a linhas (*N-ary Storage Model* - NSM ou Modelo de Armazenamento N-ário). Nesse modelo, os atributos dos registros do banco de dados são armazenados contiguamente. Esse *layout* é mais adequado para sistemas OLPT cujas transações tendem a manusear os atributos de determinados registros por vez. A Figura 4.10 (a) ilustra uma tabela orientado a linhas. Por exemplo, todos os atributos do registro de *ID* 101 estão armazenados um após o outro, seguidos pelos atributos do registro de *ID* 102 [Arulraj et al. 2016a, Copeland and Khoshafian 1985].

Os SGBDs em memória podem ser orientados a colunas (*Decomposition Storage Model* - DMS ou Modelo de Armazenamento de Decomposição). O modelo de organização colunar armazena os valores de um atributo de todos os registros contiguamente. Esse modelo é comumente empregado em sistemas OLAP. Geralmente, uma transação nesse tipo de sistema acessa um atributo (ou um subconjunto de atributos) de muitos registros ao mesmo tempo. A Figura 4.10 (b) exemplifica uma tabela orientado a colunas. Por exemplo, todos os valores da coluna *ID* são armazenados um após o outro, seguidos pelos valores da coluna *IMAGE-ID* [Copeland and Khoshafian 1985, Arulraj et al. 2016a].

ID	IMAGE-ID	NAME	PRICE	DATA
101	201	ITEM-101	10	DATA-101
102	202	ITEM-102	20	DATA-102
103	203	ITEM-103	30	DATA-103
104	204	ITEM-104	40	DATA-104

(a)

ID	IMAGE-ID	NAME	PRICE	DATA
101	201	ITEM-101	10	DATA-101
102	202	ITEM-102	20	DATA-102
103	203	ITEM-103	30	DATA-103
104	204	ITEM-104	40	DATA-104

(b)

Figura 4.10. Modelos de tabelas orientadas a linhas (a) e a colunas (b) [Arulraj et al. 2016a]

Os SGBDs em memória são comumente empregados em sistemas OLTP pelo fato de possibilitarem um acesso rápido a registros individuais. Porém, alguns sistemas possuem características HTAP, ou seja, precisam de percepções analíticas dos dados mais atuais do banco de dados. Para esses sistemas, o modelo FSM (*Flexible Storage Model* ou Modelo de Armazenamento Flexível) é o mais adequado. O FSM é uma generalização dos modelos NSM e DSM, ou seja, ele suporta tanto o modelo orientado a linhas como o colunar no mesmo banco de dados. Geralmente, FSM armazena os dados primeiramente

no formato orientado a linhas. Quando um dado se torna pouco (ou nunca) acessado, ele é reorganizado para o formato colunar [Copeland and Khoshafian 1985, Arulraj et al. 2016a].

HyPer usa um esquema de memória virtual para separar os dados entre os mais imutáveis (para transações OLAP) e os mais recentemente atualizados (para transações OLTP), ver Seção 4.6.1.3 para mais detalhes [Funke et al. 2014]. SAP HANA implementa uma estrutura de tabela unificada para suportar HTAP, ver Seção 4.6.1.4 para mais detalhes [Sikka et al. 2012].

4.3.2. Indexação

Um índice é uma estrutura de dados que organiza os registros do banco de dados para otimizar determinados tipos de operações de busca. Eles permitem rápido acesso aos dados sem a necessidade de percorrer uma tabela inteira. Os índices nos bancos de dados em disco são mapeados para páginas gerenciadas pelo *buffer* visando minimizar o acesso à memória secundária. A maioria dos bancos de dados em disco suportam índices em árvores B^+. Sistemas em disco também utilizam índices clusterizados, que mantêm os registros em uma ordem específica no disco. Esses dois tipos de índice são eficientes em buscas usando faixas de valores. Uma alternativa é o índice com tabela *hash*, muito eficiente em buscas com um valor específico [Ramakrishnan and Gehrke 2003, Elmasri and Navathe 2000].

Embora os sistemas em memória tenham altas taxas de vazão de dados, eles ainda precisam de índices para acelerar o acesso a dados. Algumas técnicas de indexação tentam otimizar o acesso a *cache*, como: HOT [Binna et al. 2018], CSS-Trees [Rao and Ross 2000], CSB-Trees [Rao and Ross 2000], pB-Trees [Chen et al. 2001], FAST [Kim et al. 2010] e ART [Leis et al. 2013]. Algumas estratégias de indexação consideram o paralelismo *multi-core*, como: MRBTree [Pandis et al. 2011], Bw-tree [Levandoski et al. 2013] e Mass-Tree [Mao et al. 2012]. Os sistemas em memória também usam índices de tabela *hash* [Fan et al. 2013].

Como os bancos de dados em memória não implementam um *buffer*, os seus índices armazenam ponteiros direto para registros, ao invés de *IDs* ou chaves primárias (como nos bancos de dados em disco). A Figura 4.11 (a) representa uma indexação de SGBDs em disco cujos índices apontam para páginas. Em contraste, a Figure 4.11 (b) representa uma estrutura de índice de SGBDs em memória que possui ponteiros direto para registros [Faerber et al. 2017, Ailamaki et al. 1999].

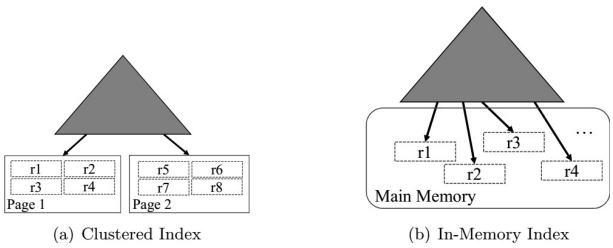


Figura 4.11. Estruturas de índices nos bancos de dados em disco (a) e em memória (b) [Faerber et al. 2017].

4.3.3. Controle de concorrência

Controle de concorrência é um método usado para impedir que transações acessem o mesmo dado simultaneamente e levem o banco de dados a uma inconsistência. Comumente, os bancos de dados em disco utilizam algum controle de concorrência baseado em bloqueio, como o bloqueio em duas fazes (*Two-Phase Locking* - 2PL) para garantir o controle de concorrência. Nesse protocolo, por exemplo, uma transação T_1 deve adquirir bloqueios nos dados que necessita antes de começar a modificá-los. Uma transação T_2 que necessite alterar dados bloqueados por T_1 deve esperar até que T_1 não precise mais desses dados e desbloqueie-os [Weikum and Vossen 2002, Ramakrishnan and Gehrke 2003].

O gerenciador de bloqueio realiza as operações de bloqueio e desbloqueio atomicamente. Além disso, ele ainda deve verificar se um dado está bloqueado sempre que uma transação solicita acesso a esse dado. Essas operações são toleradas em bancos de dados em disco, visto que o acesso ao disco é o maior gargalo. Porém, bloqueios são muito custosos em sistemas em memória, pois acrescentam ciclos de CPU. Por esse motivo, os sistemas em memória evitam implementar um gerenciador de bloqueio [Weikum and Vossen 2002, Hazenberg and Hemminga 2011].

Alguns SGBDs em memória usam metadados embutidos em registros para controlar o acesso a eles. Por exemplo, um *bit* de bloqueio é embutido no registro para informar se ele está bloqueado ou não [Garcia-Molina and Salem 1992]. Uma alternativa é usar um número contador para representar a quantidade de requisições de bloqueio feitas por transações a um registro [Ren et al. 2012].

Muitos SGBDs em memória adotam um modelo de controle de concorrência multi-versionado (*Multi-Version Concurrency Control* - MVCC). Nesse modelo, transações de apenas leitura não precisam bloquear dados, visto que transações de escrita fazem suas modificações em versões diferentes dos dados lidos. Como desvantagem, MVCC possui a sobrecarga de criar novas versões de dados e de remover versões obsoletas [Neumann et al. 2015, Lomet et al. 2012].

Hekaton and HyPer implementam um MVCC otimista. Essa abordagem não utiliza bloqueios e conflito são verificados apenas na finalização da transação. Se algum conflito for detectado, apenas uma das transações envolvidas pode finalizar, as demais devem ser abortadas [Neumann et al. 2015, Lomet et al. 2012]. SAP HANA implementa um MVCC pessimista. Nesse método, MVCC é utilizado apenas em operações de leitura. Em caso de operações de escrita, bloqueios ao nível de registro são usados [Lee et al. 2013].

Sistema que usam particionamento de dados (H-Store, VoltDB e Calvin, por exemplo) não precisam implementar um mecanismo de controle de concorrência, pois as transações executam serialmente, como descrito na Seção 4.3.1. Porém, como desvantagem, transações devem esperar até que todas as partições que precisam estejam disponíveis [Kallman et al. 2008, Thomson et al. 2012, Stonebraker and Weisberg 2013].

4.3.4. Processamento de consultas

O processamento de consultas visa extrair os dados do banco de dados da maneira mais eficiente possível. Para isso, o processador de consultas realiza as etapas de análise, otimi-

zação e execução de consulta. Os SGBDs em disco e memória realizam as atividades de análise e otimização de forma semelhante. Porém, eles diferem muito na implementação de execução de consultas [Silberschatz et al. 2020, Ramakrishnan and Gehrke 2003, Hazenberg and Hemminga 2011].

O processamento de consultas nos bancos de dados em disco foca em minimizar as E/Ss no disco. Comumente, esses sistemas utilizam algum modelo de interação estilo Volcano [Graefe and McKenna 1993]. Esse modelo é bem simples e utiliza uma combinação variada de operadores implementados de forma genérica para executar uma consulta [Hazenberg and Hemminga 2011, Zhang et al. 2015a, Faerber et al. 2017].

O modelo de interação é ineficiente em SGBDs em memória. Devido à natureza genérica, um operador deve ser interpretado em tempo de execução e ainda pode ser chamado várias vezes. Por esse motivo, o modelo de interação leva a sobrecargas desnecessárias nos bancos de dados em memória. Assim, esses sistemas preferem compilar transações em código de máquina para evitar a interpretação em tempo de execução e, consequentemente, fazer melhor uso de memória e CPU. VoltDB implementa transações compiladas em código de máquina através de procedimentos armazenados (*stored procedures*). Como desvantagem dessa abordagem, o sistema precisa conhecer as transações com antecedência, o que pode ser difícil de implementar em alguns sistemas. Por exemplo, as consultas de sistemas *ad hoc* são criadas a partir de um requisito específico do momento em que são criadas [Stonebraker and Weisberg 2013, Diaconu et al. 2013, Kemper and Neumann 2011, Menon et al. 2017].

4.3.5. Durabilidade e recuperação após falhas

A maioria dos bancos de dados em disco utilizam alguma forma de recuperação após falhas baseada no protocolo ARIES. ARIES escreve informações de atualizações em páginas para um arquivo de *log* na memória secundária. O *log* possui informação suficiente para desfazer ou refazer uma atualização inconsistente. Além disso, *checkpoints* são realizados para identificar atualizações de transações que já foram persistidas em disco e, consequentemente, diminuir a quantidade de informação no *log* a ser processada em uma recuperação após falha [Mohan and Levine 1992]. A Seção 4.4 detalha a recuperação de SGBDs em disco.

Os bancos de dados em memória também realizam atividades de *logging* e *checkpoint* para fins de tolerância a falhas. Além disso, essas técnicas também proveem durabilidade, uma vez que a memória principal é volátil. Embora os componentes desses dois sistemas pareçam semelhantes, eles diferem muito na maneira como são implementados [Magalhães et al. 2021b, Magalhaes et al. 2022, Magalhaes et al. 2022]. A Seção 4.5 discute a recuperação de SGBDs em memória com mais detalhes.

4.4. Recuperação em bancos de dados em disco

Os gerenciadores de *buffer*, comumente, o utilizam os protocolos *No-force* e *Steal* em suas políticas de substituição de páginas. No protocolo *No-force*, uma página "suja" não precisa ser enviada para a memória secundária imediatamente à finalização de sua transação. Uma página é considerada "suja"quando ela possui modificações na memória principal que ainda não foram copiadas para a memória secundária. *No-force* permite que pági-

nas muito atualizadas permaneçam no *buffer*, evitando múltiplas E/Ss para a memória secundária, as quais são operações caras. Em *Steal*, páginas "sujas" de transações ativas (transações em execução) podem ser movidas para a memória secundária antes da finalização da transação. Com *Steal*, o sistema pode ter um banco de dados maior do que a memória disponível, pois páginas de transações ativas podem ser movidas para o disco visando dar espaço na memória para novas páginas [Härder and Reuter 1983, Mohan et al. 1992].

As falhas (*crashes*) de bancos de dados são eventos que afetam o processamento do sistema, tais como: *bugs* de *software*, entradas de dados erradas, defeitos de *hardware*, falta de energia, erros humanos, dentre outros. Após um *crash*, o gerenciador de recuperação deve assegurar que o banco de dados não fique em um estado inconsistente. Uma transação pode ser considerada a unidade de recuperação em um banco de dados. Assim, o gerenciador de recuperação deve assegurar que as transações finalizadas tenham suas páginas escritas na memória secundária, mesmo que essas páginas não tenham sido movidas para o disco antes da falha. Além disso, deve ser assegurado que transações não finalizadas não tenham suas modificações refletidas na memória secundária. Esses dois cenários de inconsistência são possíveis devido às políticas de substituição de páginas utilizadas pelo gerenciador de *buffer* [Härder and Reuter 1983, Mohan et al. 1992].

Basicamente, existem as falhas de transação, sistema e disco. A falha de transação ocorre quando uma transação não pode atingir suas metas ou viola alguma integridade do banco de dados (duplicação de chave primária, por exemplo) então deve ser cancelada e desfeita. Esse tipo de falha não interrompe a operação do sistema e nem atrapalha o processamento das outras transações. A falha de sistema acontece quando o conteúdo da memória principal é perdido (falta de energia, por exemplo), impossibilitando que o sistema continue operando. A falha de disco acontece quando a memória secundária é perdida (defeito de disco, por exemplo), impedindo o funcionamento do sistema. Nessa falha, diferentemente das outras duas, é necessário reparar ou trocar o dispositivo de armazenamento defeituoso antes de começar o processo de recuperação [Mohan et al. 1992].

A Figura 4.12 exemplifica uma falha de sistema. Nesse cenário, cinco transações executam até acontecer a falha de sistema. Após esse tipo de falha, o sistema não pode continuar operando e deve ser reiniciado. Assim que o sistema reinicia, o gerenciador de *buffer* começa o processo de recuperação do banco de dados. As modificações das transações T1, T2 e T3 devem ser persistidas, uma vez que elas finalizaram. Assim, essas transações precisam ser refeitas, se necessário. Como as transações T4 e T5 não finalizaram, as suas modificações devem ser desfeitas [Härder and Reuter 1983].

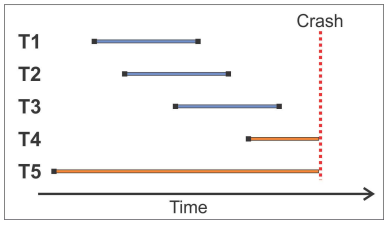


Figura 4.12. Um cenário de falha de sistema [Härder and Reuter 1983].

4.4.1. O algoritmo de recuperação ARIES

ARIES (*Algorithms for Recovery and Isolation Exploiting Semantics*) é um método de recuperação após falhas bastante conhecido e utilizado pelos bancos de dados. A maioria dos SGBDs em disco usa algum método de recuperação estilo ARIES. Esse método de recuperação suporta *Steal*, *No-force*, WAL, *checkpoint fuzzy*, registro de compensação de *log* e granularidade fina [Mohan et al. 1992].

4.4.1.1. *Write-ahead logging*

O protocolo WAL (*Write-Ahead Logging*) assegura que, antes de uma modificação no bando de dados, seja registrada uma informação que descreve essa modificação em um arquivo de *log* na memória secundária. O *log* é um arquivo cujos registros são armazenados sequencialmente. Cada registro possui um LSN (*Log Sequence Number*) que é um número em ordem crescente que funciona como uma identificação única. Geralmente, os bancos de dados em disco utilizam *log* físico cujos registros armazenam informações físicas dos dados atualizados (páginas, por exemplo). Um registro de *log* armazena a imagem de antes (*before image*) e a imagem de depois (*after image*) da modificação de um dado [Mohan et al. 1992].

SGBDs comerciais, como MySQL [MySQL 2020] e Oracle [Oracle 2020], tipicamente implementam um *log* fisiológico por questões de desempenho. No *log* fisiológico, um registro referencia uma página e uma operação lógica na página. Por exemplo, um registro pode conter a imagem da página de antes da atualização e a operação de atualização na página [Mohan et al. 1992, Tucker 2004, Gray and Reuter 1993].

Os registros de *log* mapeiam todas as atualizações feitas no banco de dados. Assim, após um *crash*, os registros de *log* são utilizados para refazer (REDO) ou desfazer (UNDO) modificações de transações que tenham gerado alguma inconsistência no banco de dados. Os registros de *log* de uma determinada transação são ligados, como em uma lista encadeada, permitindo percorrê-los para refazer ou desfazer a transação individualmente. O arquivo de *log* deve ser copiado para discos diferentes para sobreviver a falhas de disco e, se possível, armazenado em locais diferentes para sobrevier a catástrofes [Mohan et al. 1992].

4.4.1.2. *Checkpoint fuzzy*

Como o *log* é um arquivo em crescimento, ele deve ser truncado periodicamente. Essa ação é necessária, pois o disco possui espaço limitado. Alguns sistemas tendem a ter um arquivo de *log* muito maior do que o banco de dados, como os sistemas OLTP que fazem muitas modificações em poucas partes do banco de dados. Além disso, quanto mais registros existirem no *log*, mais informação deverá ser processada durante uma recuperação e, consequentemente, a recuperação será mais lenta. Assim, *checkpoints* devem ser executados periodicamente para encontrar um novo ponto no *log* de onde a recuperação pode começar. Esse ponto é um registro no *log* onde se tem a certeza de que todas as modificações de transações finalizadas anteriormente ao ponto foram persistidas na memória secundária. Assim, os registros anteriores ao *checkpoint* podem ser descartados do

log [Mohan et al. 1992].

ARIES usa o *checkpoint fuzzy*. Essa técnica copia informações de ocupação do *buffer* durante o processamento de transações, como páginas "sujas" e transações ativas. Essas informações são utilizadas para encontrar o registro no *log* de onde a recuperação deve iniciar. *Fuzzy* possui a vantagem de não interferir no processamento das transações. Contudo, ele torna o processo de recuperação mais lento, pois é necessário analisar o *log* para encontrar o registro de início da recuperação [Mohan et al. 1992].

A Figura 4.13 ilustra um cenário de execução do *checkpoint fuzzy*. As páginas P1, P2, P3 e P4 possuem escritas de registros de atualização no arquivo de *log*, representadas pelas linhas pontilhadas. Por exemplo, a página P1 armazenou registros de *log* com LSNs 30, 60 e 100. Durante a execução do *checkpoint*, informações sobre as páginas "sujas" P1, P3 e P4 são armazenadas no arquivo de *checkpoint*. Informações sobre a página P2 não são armazenadas, uma vez que essa página foi salva na memória secundária antes do início do *checkpoint*. Adicionalmente, o *checkpoint* também armazena informações sobre as transações ativas, que não estão representadas nesse exemplo. Após uma falha, as páginas P1, P3 e P4 são identificadas como "sujas". Com essa informação o sistema pode identificar que a recuperação deve iniciar a partir do registro de LSN 20, pois ele é o mais antigo entre os registros das páginas "sujas". Além disso, as transações que estavam ativas também são identificadas [Magalhães 2022, Mohan et al. 1992, Härder and Reuter 1983].

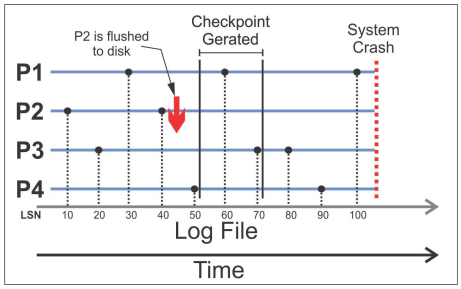


Figura 4.13. Um cenário de execução de *checkpoint fuzzy* [Magalhães 2022].

4.4.1.3. Recuperação após falha de sistema

Após uma falha de sistema, ARIES recupera um banco de dados por meio de três fases: Análise, Refazer e Desfazer [Mohan et al. 1992].

Assim que o sistema reinicia após uma falha de sistema, o gerenciador de recuperação examina o último *checkpoint* para obter as informações gravadas de transações ativas e páginas "sujas". Em seguida, a fase de análise percorre o arquivo de *log* começando do último *checkpoint* até o final do arquivo para encontrar todas as transações ativas e páginas "sujas" até o momento do *crash*. Com essas informações, o sistema calcula de onde a fase Refazer deve começar [Mohan et al. 1992].

A fase Refazer percorre o arquivo de *log* começando do primeiro registro de atualização das páginas "sujas" até o final do arquivo. O objetivo dessa fazer é refazer todas as transações ativas com páginas "sujas"de antes da falha. Esse método é chamado de pa-

radigma de repetição da história, pois reconstrói o banco de dados para o mesmo estado em que estava no momento da falha [Mohan et al. 1992, Mohan 1999].

A fase Desfazer percorre o arquivo de *log* em ordem inversa começando do último registro até desfazer todas as transações que não finalizaram até o *crash*. Cada transação é desfeita em ordem inversa de execução de seus registros de *log*. Assim que todas as transações não finalizadas forem desfeitas, o processo de recuperação termina e o banco de dados pode processar novas transações [Mohan et al. 1992].

4.4.1.4. Registro de *log* de compensação

ARIES introduz o conceito de registro de *log* de compensação (*Compensation Log Record* - CLR). Para cada ação de registro de *log* desfeita, um CLR é armazenado no arquivo de *log*. Um CLR descreve as ações executadas durante uma operação de desfazer (*rollback*). Esse tipo registro nunca é desfeito, ou seja, nunca acontece um desfazer de um registro de *rollback*. O CLR possui um campo que aponta para o registro de sua transação que é anterior ao registro desfeito. Assim, durante o desfazer de uma transação, não é necessário desfazer o CLR e nem o registro cujo *rollback* é descrito por esse CLR. Isso é possível porque esses dois registros possuem ações opostas e desfazê-las um após a outra não traz efeito algum ao estado final do banco de dados [Mohan et al. 1992, Mohan 1999].

A Figura 4.14 mostra um exemplo de CLR gerado por um *rollback*. A Figura 4.14 (a) representa as atualizações feitas por uma transação *T* e a Figura 4.14 (b) os registros armazenados no *log* através dessas atualizações. As ações *A1*, *A2*, *A3* e *A4* são atualizações normais e *L1*, *L2*, *L3* e *L4* são os respectivos registros de *log* dessas atualizações. *A3'* é uma ação de desfazer de *A3*. Assim, *L3'* é um registro de *log* de compensação que descreve *A3'*. Por exemplo, se *L3* representar uma operação de inserção de uma linha *R* em uma página *P* então *L3'* deve descrever a exclusão de *R* em *P*. Durante a recuperação, assumindo que a transação *T* não foi finalizada, as ações de *T* devem ser desfeitas, começando do registro *L4*. Quando o gerenciador de recuperação atingir o registro de *log* de compensação L3', ao invés de desfazer L3' e L3, ele pula para o registro L2 [Mohan et al. 1992, Mohan 1999].

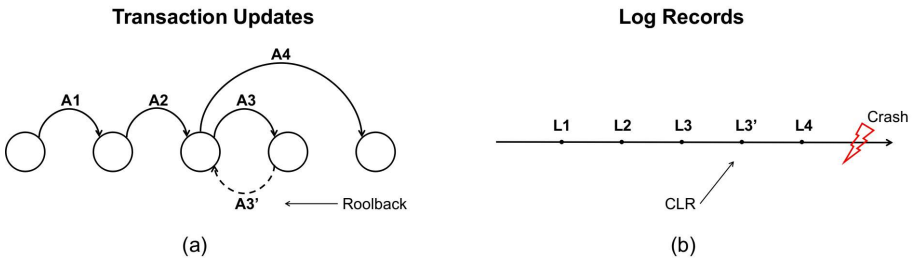


Figura 4.14. Registro de *log* de compensação gerado ao desfazer uma ação de transação [Mohan et al. 1992].

Os CLRs diminuem a quantidade de registros processados durante a recuperação após falhas, até em face a falhas sucessivas. Na ocorrência de tais eventos, os CLRs evitam o aninhamento de ações de desfazer, que podem causar um grande atraso no processo de recuperação. Em contraste, alguns sistemas, como DB2/MVS V1 [Cheng et al. 1984], NonStop SQL [Tandem Database Group 1987] e AS/400 [Clark and Corrigan 1989], não

usavam o conceito de CLRs e desfaziam a mesma ação de desfazer uma ou mais vezes em caso de falhas sucessivas [Mohan et al. 1992].

4.4.1.5. Granularidade fina

Sistemas antes de ARIES, como System R [Gray et al. 1981] e DB2/MVS V1 [Crus 1984], comumente implementavam o algoritmo UNDO/REDO para fins de tolerância a falhas [Bernstein et al. 1987]. Esse algoritmo executa a fase UNDO antes da REDO. Essa abordagem é chamada de paradigma de refazer seletivo (*paradigm of selective redo*). Nessa abordagem, a granularidade dos dados armazenados no *log* não pode ser maior que a dos dados manipulados no mecanismo de bloqueio (granularidade grossa). Por exemplo, se o SGBD armazena páginas no *log*, ele não deve manusear granularidade menor do que páginas nos bloqueios. Caso contrário, o banco de dados pode ficar inconsistente após uma recuperação. Isso acontece porque uma ação de desfazer durante o UNDO pode fazer o *log* perder o rastreamento para refazer uma transação durante o REDO [Mohan et al. 1992, Bernstein et al. 1987, Härder and Reuter 1983].

O algoritmo ARIES possui granularidade fina, ou seja, ele permite bloqueios de dados com granularidade menor que os dados do *log*. ARIES resolve o problema da granularidade grossa através do paradigma de repetição de história (*paradigm of repeating history*), ou seja, executando o passo REDO antes do UNDO [Mohan 1999, Mohan et al. 1992].

4.4.2. A família de algoritmos ARIES

Alguns algoritmos estendem ARIES acrescentando funcionalidades que o algoritmo original não implementa. São exemplos desses algoritmos: ARIES/NT (ARIES *for Nested Transactions*), ARIES-RRH (ARIES *with Restricted Repeating of History*), ARIES/IM (ARIES *for Index Management*), ARIES/KVL (ARIES *using Key-Value Locking*), ARIES/LHS (ARIES *for Linear Hashing with Separators*) e ARIES/CSA (ARIES *for the Client-Server Architecture*) [Mohan and Narang 1994, Mohan 1999].

ARIES/NT [Rothermel and Mohan 1989] acrescenta o uso de transações aninhadas. Uma transação aninhada (transação filha) é uma transação que inicia sua execução no escopo de execução de outra transação (transação pai). Esse algoritmo usa uma árvore para fazer o encadeamento dos registros de *log* das transações pai e filhas. A árvore é utilizada para percorrer os registros na ordem correta durante a recuperação do banco de dados.

ARIES/IM [Mohan and Levine 1992] e ARIES/KVL [Mohan 1990] tentam controlar a concorrência e recuperação de índices em árvores. Técnicas anteriores não consideravam a recuperação de índices com a granularidade fina de bloqueios. O principal objetivo desses algoritmos é resolver os problemas dos algoritmos anteriores de controle de concorrência de índices, como o algoritmo original de System R [Gray et al. 1981]. Por exemplo, ARIES/IM e KVL tentam evitar *deadlocks* entre transações durante o acesso aos índices. Um *deadlock* acontece quando duas ou mais transações entram em conflito por algum recurso e bloqueiam uma à outra permanentemente. Como consequência, a execução de nenhuma delas é possível.

ARIES/LHS [Mohan 1993] manuseia o controle de concorrência e recuperação de

índices em *hashes* dinâmicas. ARIES/LHS tenta evitar *deadlocks* de transações durante o acesso aos índices. Porém, esse algoritmo não foi implementado.

ARIES/CSA [Mohan and Narang 1994] foi projetado para trabalhar numa arquitetura cliente-servidor. Nesse algoritmo, as máquinas clientes fazem atualizações no banco de dados, produzem os registros de *log* dessas atualizações e enviam esses registros para o servidor armazená-los. O servidor é responsável por organizar os registros em seu arquivo de *log* local e gerenciar a recuperação do banco de dados. Esse algoritmo não foi implementado.

4.4.3. Recuperação através de *log* indexado

Existem técnicas modernas de recuperação de bancos de dados após falhas que utilizam *log* estruturado em forma de árvore, tais como: *Single-page repair* [Graefe and Kuno 2012], *Single-pass restore* [Sauer et al. 2015], *Instant restart* [Graefe et al. 2015, Graefe et al. 2016] e *Instant restore* [Sauer et al. 2017, Sauer 2017, Sauer 2019]. Ao invés do tradicional arquivo de *log* sequencial utilizado por ARIES, essas técnicas usam uma árvore B particionada [Graefe 2003] como arquivo de *log*. Existem também outras abordagens que utilizam outras estruturas de indexação, como árvore B^+, tabela *Hash* e árvore ART [Magalhães 2022, Magalhães et al. 2021a, Lee et al. 2022].

A árvore B particionada (Figura 4.15 (a)) armazena os registros em partições (Figura 4.15 (b)). Em cada partição, os registros de *log* são ordenados primariamente pelo ID da página que eles alteraram e secundariamente pela ordem das alterações. Cada partição é um arquivo que contém um índice que permite se deslocar aos registros de *log* de uma determinada página. No exemplo da Figura 4.15 (b), a partição possui registros de *log* que alteraram as páginas *A* e *B*. Cada chave de busca da árvore aponta para as partições que contêm registros de *log* que alteraram uma determinada página. Assim, é possível recuperar todos os registros de *log* de uma determinada página mediante uma busca na árvore [Graefe et al. 2015, Sauer et al. 2017, Sauer 2017].

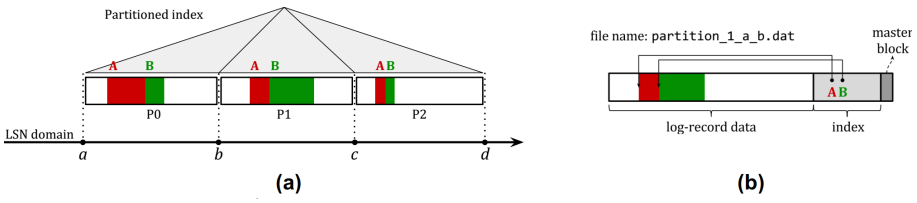


Figura 4.15. Árvore B particionada (a) e arquivo de partição (b) [Sauer 2017].

A recuperação do banco de dados via *log* indexado é realizada em paralelo à execução de transações, ou seja, o sistema não precisa esperar a recuperação completa para, só depois, começar a operação de novas transações. Essa abordagem é chamada de recuperação instantânea, pois os usuários/aplicações do SGBD têm a impressão de que o banco de dados foi recuperado imediatamente após uma falha [Graefe et al. 2015, Sauer et al. 2017, Sauer 2017, Magalhães et al. 2021a, Magalhães 2022].

A Figura 4.16 ilustra o esquema de recuperação instantânea de bancos de dados após uma falha de disco. Durante o processamento normal das transações, registros são armazenados no *log* indexado (árvore B particionada). Após uma falha, o processo de re-

cuperação copia páginas incrementalmente do *backup*. Enquanto um conjunto de páginas (seguimento) é copiado, os registros dessas páginas são buscados no *log* indexado. Em seguida, as ações dos registros são aplicadas em suas respectivas páginas. Assim que uma página é restaurada completamente, ela pode ser usada por novas transações. Caso uma nova transação precise de uma página que ainda não foi restaurada, essa página pode ser recuperada sob demanda por meio de buscas nos índices do *backup* e do *log* [Sauer et al. 2017, Sauer 2017, Sauer 2019].

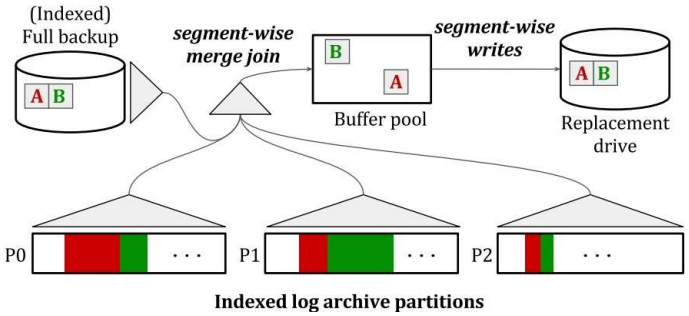


Figura 4.16. Esquema de recuperação instantânea após uma falha de disco [Sauer 2017].

4.5. Durabilidade e recuperação em bancos de dados em memória

Os bancos de dados em memória implementam técnicas de *logging*, *checkpoint* e recuperação para prover tolerância a falhas, como nos bancos em disco. Porém, quando examinadas em detalhes, a implementação dessas técnicas nesses dois sistemas se diferem muito. Por exemplo, a maioria dos bancos em memória evita protocolos estilo ARIES por questões de desempenho. Durabilidade e recuperação após falhas são as únicas razões para um SGBD em memória acessar o disco. Assim, além de prover tolerância a falhas, o mecanismo de recuperação também é responsável pela durabilidade dos SGBDs em memória [Jagadish et al. 1993, Gruenwald et al. 1996].

A Seção 4.4 forneceu os principais conceitos de recuperação de bancos de dados após falhas, focando em SGBDs em disco. Essa seção detalha como os SGBDs em memória implementam tolerância a falhas.

4.5.1. *Logging*

Registros de atualizações devem ser escritos para um arquivo de *log* em armazenamento estável devido à volatilidade da memória RAM. Um armazenamento estável é uma memória não volátil, onde dados podem persistir. Escrever registros em um arquivo de *log* é a maior fonte de sobrecarga para o processamento das transações nos SGBDs em memória devido às operações de E/S para a memória secundária. Assim, o mecanismo de *logging* é otimizado para interferir o mínimo possível no processamento das transações [Garcia-Molina and Salem 1992, Yao et al. 2016].

Os bancos de dados em memória comumente utilizam um *log* lógico que armazena registros com descrições das operações de atualização no banco de dados em alto nível, como a inserção de uma linha em uma tabela. Geralmente, registros lógicos são mais leves, pois armazenam menos itens de dados do que registros físicos utilizados pelos SGBDs em disco [Malviya et al. 2014, Wu et al. 2017, Zheng et al. 2014].

Para reduzir a quantidade de dados enviados para a memória secundária, o *log*

armazena registros REDO-*only* (apenas de refazer), ou seja, não são gravados registros de desfazer. Como consequência, transações devem gravar atualizações no *log* apenas durante o *commit*. Assim, caso o sistema falhe, não existirão atualizações de transações não finalizadas a serem desfeitas. Gravar registros apenas ao final de uma transação não é um problema para SGBDs em memória, pois eles são utilizados tipicamente em sistemas OLTP, que possuem transações de curta duração [Malviya et al. 2014, Wu et al. 2017].

Os SGBDs em memória ainda podem manter registros de UNDO temporariamente na memória principal para desfazer uma transação abortada. Após a transação ser finalizada ou desfeita (devido a um *abort*), os seus registros de desfazer são excluídos. Porém, muitos bancos de dados em memória não precisam desfazer transações, como nos sistemas multi versionados. Nesses sistemas, por exemplo, uma transação T_1 faz suas modificações em novas versões dos dados atuais do banco. Essas novas versões são vistas apenas pela transação T_1. Outras transações podem ver as versões de T_1 apenas após a finalização de T_1, ou seja, quando as versões de T_1 se tornam as versões atuais do banco de dados. Caso T_1 seja abortada, as suas versões são apenas descartas e, posteriormente, removidas por um coletor de lixo [Malviya et al. 2014, Wu et al. 2017, Zheng et al. 2014].

Para reduzir ainda mais o tráfego de dados para a memória secundária, alguns sistemas gravam registros de *log* ao nível de transação. Essa técnica é chamada de *logging* de transação ou de comando. Nessa técnica, cada transação deve ser um procedimento armazenado cadastrado previamente. Para cada transação executada, apenas um registro de *log* é gravado com o identificador do procedimento armazenado e seus parâmetros. Essa técnica leva a pouca sobrecarga no processamento das transações, pois apenas um registro é escrito no *log* para cada execução de transação, mesmo que essa transação possua muitas operações de escrita [Malviya et al. 2014, Wu et al. 2017].

Adicionalmente, muitos sistemas evitam escrever registros de atualização em índices no *log*. Assim, após uma falha, esses sistemas preferem reconstruir as estruturas de índice em paralelo ao processo de recuperação do banco de dados [Malviya et al. 2014, Wu et al. 2017, Zheng et al. 2014]. Esse minicurso não cobre recuperação de índices.

Os bancos de dados utilizam algumas técnicas para melhorar o desempenho de escrita de registros de *log* para a memória secundária, como *group commit* e *pre-commit*.

O *group commit* acumula registros de *log* produzidos por várias transações que executam no mesmo período de tempo para escrevê-los juntos em uma única operação de E/S. Essa técnica diminui o número de E/Ss para a memória secundária, uma vez que uma única E/S pode ser utilizada por várias transações [Hagmann 1987, DeWitt et al. 1984].

No *pre-commit*, uma transação libera os seus bloqueios assim que seus registros de *log* são enviados para a memória secundária, sem esperar a confirmação de escrita desses registros. Assim, a transação evita a sobrecarga de escrita para o *log*. Entretanto, o sistema pode perder as garantias de durabilidade em caso de falhas [DeWitt et al. 1984, Garcia-Molina and Salem 1992].

4.5.2. Checkpoint

Embora o *checkpoint fuzzy* (discutido na Seção 4.4.1) seja uma técnica que impõe pouca sobrecarga ao processamento de transações, ele não é adequado para SGBDs em memória.

Fuzzy depende de registros de *log* para refazer e desfazer transações. Porém, bancos de dados em memória não armazenam registros de UNDO. As técnicas de *checkpoint* em bancos em memória são muito diferentes das técnicas dos bancos em disco [Salem and Garcia-Molina 1989, Liedes and Wolski 2006, Ren et al. 2016].

Alguns SGBDs em memória materializam as operações lógicas do *log* em dados físicos armazenados em um arquivo de *checkpoint* na memória secundária. Assim, os registros de *log* anteriores a um *checkpoint* podem ser descartados, pois o arquivo de *checkpoint* funciona como um *backup* do banco de dados [Eich 1986, Garcia-Molina and Salem 1992, Diaconu et al. 2013, Stonebraker and Weisberg 2013].

A maioria dos bancos de dados em memória utiliza uma técnica de *checkpoint* consistente que produz um arquivo comumente chamando de *snapshot*. O *snapshot* é o *backup* do banco de dados em um dado instante de tempo. Porém, o sistema não precisa parar o processamento das transações para produzir um *snapshot* [Eich 1986, Garcia-Molina and Salem 1992, Diaconu et al. 2013, Stonebraker and Weisberg 2013].

Após um *checkpoint*, os registros de *log* anteriores ao *checkpoint* podem ser descartados. Como consequência, *checkpoints* reduzem o tempo de recuperação, pois menos registros de *log* precisam ser processados durante a recuperação. Além disso, carregar dados físicos para a memória é menos custoso do que executar operações lógicas, que requerem mais processamento do sistema. Além disso, geralmente um arquivo de *checkpoint* contém menos informação do que um arquivo de *log*. Por exemplo, sistemas OLTP fazem muitas modificações em poucas partes do banco de dados. Assim, um sistema OLTP tende a possuir muito mais registros no *log* do que dados no banco de dados [Eich 1986, Diaconu et al. 2013, Stonebraker and Weisberg 2013].

Foram propostos alguns algoritmos para produzir *snapshots* em SGBDs em memória, como Naive [Bronevetsky et al. 2006, Schroeder and Gibson 2007], Zigzag [Chen et al. 2011b], PingPong [Chen et al. 2011b], Hourglass [Li et al. 2018a, Li et al. 2018b] e Piggyback [Li et al. 2018a, Li et al. 2018b]. Porém, a maioria dos sistemas utiliza um algoritmo mais simples chamado de *Copy-on-Update* (COU).

O algoritmo de *snapshot* COU é executado durante a execução normal do sistema, ou seja, ele não para o processamento das transações. Esse algoritmo percorre todos os registros do banco de dados para copiá-los para um arquivo de *snapshot* na memória primária. Ele usa um *array* de *bits* para identificar registros inseridos, atualizados ou excluídos desde o início do processo de *checkpoint*. Assim, o algoritmo pode pular registros inseridos. Além disso, antes de uma atualização ou exclusão de registro, o conteúdo original desse registro é copiado para uma tabela sombra para que o *checkpoint* possa ler a versão antiga do registro. Durante o *checkpoint*, um processo serializa o *snapshot* da memória principal para um arquivo na memória secundária. Como desvantagem, essa técnica pode produzir uma sobrecarga de memória, pois ela pode potencialmente precisar de um espaço duas ou três vezes maior que o banco de dados [Chen et al. 2011b, Malviya et al. 2014].

4.5.3. Recuperação

O mecanismo de *logging* adotado pelos SGBDs em memória não produz inconsistências em face a falhas. Isso acontece porque os registros de REDO são gravados apenas na hora da confirmação das transações. Como consequência, não há a possibilidade de alguma transação não finalizada, devido a um *chash*, persistir dados inconsistentes. Além disso, registros de UNDO não são gravados no *log*. Assim, o processo de recuperação após uma falha de sistema em um SGBD em memória é, na verdade, um processo de carregar os dados da memória secundária para a memória principal [Magalhães et al. 2021b].

Sempre que acontece uma falha de sistema em um banco de dados em memória, o sistema para de funcionar e o conteúdo da memória principal é perdido. Como consequência, o banco de dados também é perdido. Assim que o sistema reinicia, o gerenciador de recuperação executa duas tarefas: (1) carregar o último *snapshot* da memória secundária para o banco de dados na memória principal e, em seguida, (2) reaplicar todas as operações dos registros de *log*. Após esse processo terminar, o sistema é recuperado para o seu último estado consistente de antes da falha e novas transações podem ser executadas [Eich 1986, Li and Eich 1993, Tang Yanjun and Luo Wen-hua 2010].

Não existe uma falha de disco em SGBDs em memória, visto que o banco de dados reside na memória principal. O dispositivo de armazenamento estável para onde são escritos os arquivos de *log* e *snapshot* pode falhar. Entretanto, isso não faz o sistema parar de funcionar, necessariamente. Porém, caso o sistema pare nesse evento, supondo que existem cópias do *log* e *checkpoint* em outros dispositivos, após a troca do dispositivo defeituoso, o *crash* pode ser tratado como uma falha de sistema [Eich 1986, Li and Eich 1993, Tang Yanjun and Luo Wen-hua 2010].

A Figura 4.17 ilustra a arquitetura básica de um banco de dados em memória. Durante o processamento normal das transações, o componente Logger copia registros de atualização no banco de dados para o arquivo de *log* na memória secundária. Além disso, periodicamente, o componente Checkpointer produz *snapshots* do banco de dados e os armazena na memória secundária. Após uma falha de sistema, todo o conteúdo do banco de dados é perdido. Assim que o SGBD reinicia, o componente Restorer copia o último *snapshot* para a memória principal e, em seguida, executa todas as operações dos registros de *log*. Novas transações podem ser executadas apenas após a recuperação completa do banco de dados [Magalhães et al. 2021b].

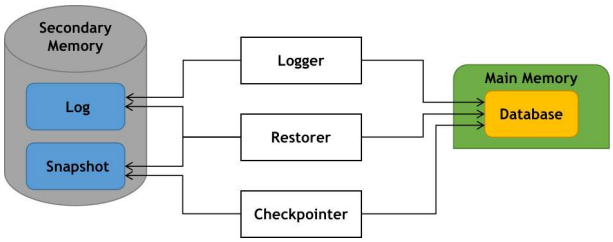


Figura 4.17. Arquitetura de um banco de dados em memória [Magalhães et al. 2021b].

4.6. Estratégias de recuperação de bancos de dados em memória

Existem várias abordagens para implementar um SGBD em memória, como discutido na Seção 4.3. Como consequência, as estratégias de recuperação adotadas por esses sistemas

são diversas. A Tabela 4.1 exibe as principais características das técnicas de tolerância a falhas implementadas por uma amostra representativa dos SGBDs em memória modernos. Os demais SGBDs em memória utilizam alguma combinação dessas técnicas.

Tabela 4.1. Estratégia de recuperação de SGBDs em memória e suas principais características.

SGBD	*Logging*	*Checkpoint*	Recuperação
Hekaton	operação	versões de dados	padrão
VoltDB	transação	*snapshot*	padrão
HyPer	transação	*snapshot*	padrão
SAP HANA	operação	*snapshot*	padrão
SiloR	valor	*snapshot "fuzzy"*	padrão
PACMAN	transação	-	paralela
Adaptive	transação/ estilo-ARIES	*snapshot*	paralela
FineLine	fisiológico	-	instantânea
HiEngine	valor	*snapshot*	"instantânea"
MM-Direct	operação	-	instantânea

Embora os SGBDs em memória utilizem várias técnicas diferentes para prover tolerância a falhas, as estratégias de recuperação são semelhantes. Assim, nesse trabalho, categorizamos essas estratégias em padrão, paralela e instantânea. As estratégias de recuperação foram categorizadas para simplificar a discussão delas. As seções seguintes detalham cada uma dessas estratégias.

4.6.1. Recuperação padrão

Hekaton, VoltDB, HyPer, SAP HANA e SiloR são exemplos de SGBDs em memória modernos que realizam tolerância a falhas como descrito na Seção 4.5, que é o método de recuperação utilizado pela maioria do SGBDs em memória. Contudo, se observados em detalhes, esse SGBDs diferem em como implementam suas estratégias de recuperação.

4.6.1.1. Hekaton

Hekaton é um banco de dados em memória incorporado ao Microsoft SQL Server. Assim, um banco de dados no SQL Server pode ter tabelas em disco e em memória. Uma transação pode atualizar os dois tipos de tabela. Contudo, uma transação que atualiza apenas tabelas em memória pode ser otimizada para acesso à memória. Hekaton utiliza armazenamento multi visionado e um MVCC otimista [Diaconu et al. 2013, Freedman et al. 2014, Larson et al. 2013]

Hekaton armazena versões de dados inseridos e atualizados em registros lógicos de REDO no arquivo de *log*. Para dados excluídos, apenas os IDs das versões desses dados são enviados para o *log*. *Checkpoints* são realizados periodicamente para fazer um mapeamento das versões no *log*. O *checkpoint* é armazenado em dois tipos de arquivo:

(*1*) *data* que contém as novas versões e (*2*) *delta* que contém informações das versões excluídas. Após uma falha, o arquivo *delta* é utilizado para filtrar quais versões no arquivo *data* devem ser carregadas na memória. Após a verificação completa do *checkpoint*, os registros de *log* após o *checkpoint* são reaplicados [Diaconu et al. 2013].

4.6.1.2. VoltDB

VoltDB é um SGBD em memória projetado a partir do banco de dados acadêmico H-Store [Kallman et al. 2008]. VoltDB armazena seus dados em partições com o objetivo de executar as transações serialmente. Além disso, o banco de dados pode ser distribuído e replicado em vários nós. Suas transações podem ser compiladas em procedimentos armazenados [Malviya et al. 2014, Stonebraker and Weisberg 2013].

VoltDB implementa *logging* de transações e *snapshots* para prover tolerância a falhas. Uma transação que executa em um único nó armazena seus registros de *log* apenas para o seu nó. Transações distribuídas executam em partições de mais de um nó. Nesse caso, um desses nós é escolhido para coordenar os demais. Apenas o coordenador armazena os registros de *log* gerados pela transação, como também as mensagens trocadas entre os nós [Malviya et al. 2014, Stonebraker and Weisberg 2013].

Após uma falha de sistema, o SGBD carrega o último *snapshot* na memória. Em seguida, cada um dos nós reaplica as ações de seus registros de *log*. Se necessário, o nó envia registros de *log* para outros nós os reaplicarem [Malviya et al. 2014, Stonebraker and Weisberg 2013].

4.6.1.3. HyPer

HyPer é um SGBD em memória capaz de suportar cargas de trabalho HTPA. Esse SGBD utiliza um esquema de memória virtual (Figura 4.18) para separar os dados entre os mais recentemente atualizados (para transações OLTP) e os mais imutáveis (para transações OLAP) [Faerber et al. 2017, Funke et al. 2014].

Inicialmente, transações OLTP e OLAP compartilham a mesma memória virtual. Quando um dado é atualizado, a nova versão desse dado é copiada para uma nova memória virtual acessada apenas por transações OLTP. As transações OLAP continuam acessando a versão antiga do dado na sessão de memória virtual em que foram criadas. Versões cujos dados não foram atualizados continuam sendo acessadas por transações OLTP e OLAP. As transações em HyPer executam suas operações sereialmente em partições [Faerber et al. 2017, Funke et al. 2014].

No exemplo da Figura 4.18, os objetos *a*, *a2* e *a3* representam versões antigas do objeto *a4* que é a versão atual de um dado. Apenas transações OLTP podem acessar a página do dado *a4*. As páginas de cada versão de dado são acessadas por suas correspondentes sessões OLAP [Funke et al. 2014].

HyPer implementa *logging* de transações e produz *snapshots*. Um *snapshot* é produzido como um *backup* do banco de dados através de uma sessão de memória virtual (ver Figura 4.18). Após um *crash*, o último *snapshot* é carregado para a memória e as

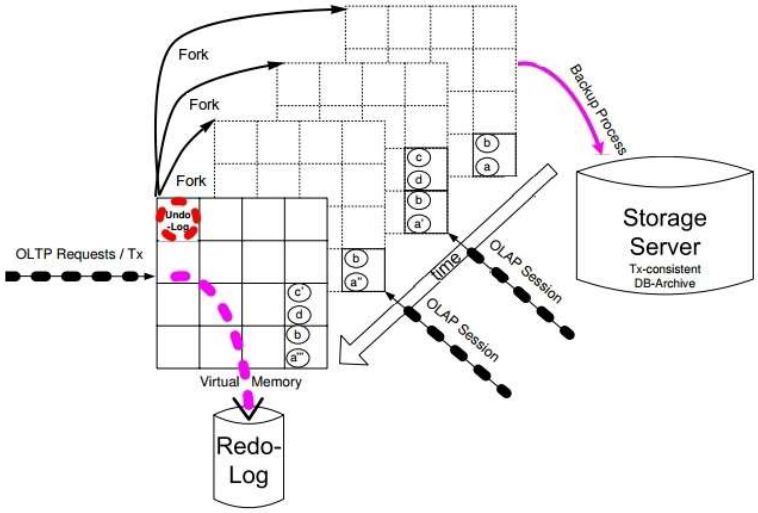


Figura 4.18. Esquema de memória virtual implementado por HyPer para suportar cargas de trabalho HTPA [Funke et al. 2014].

ações dos registros de *log* após o *chekcpoint* são executados novamente [Funke et al. 2014, Kemper and Neumann 2011, Mühe et al. 2011].

4.6.1.4. SAP HANA

O banco de dados SAP HANA é capaz de suportar cargas de trabalho HTPA através de uma estrutura de tabela unificada (Figura 4.19). A tabela unificada propaga tuplas para três representações de armazenamento: *L1-delta*, *L2-delta* e *Main store*. Em *L1-delta*, a tabela é orientada a linhas. *L2-delta* é uma estrutura de organização intermediária cuja tabela é orientada a colunas. *Main store* implementa tabelas no formato colunar com *dictionary encoding* (codificador de dicionário) altamente comprimido. O armazenamento de dados é multi versionado e o controle de concorrência é MVCC. [Färber et al. 2012, Färber et al. 2011, Faerber et al. 2017, Sikka et al. 2012].

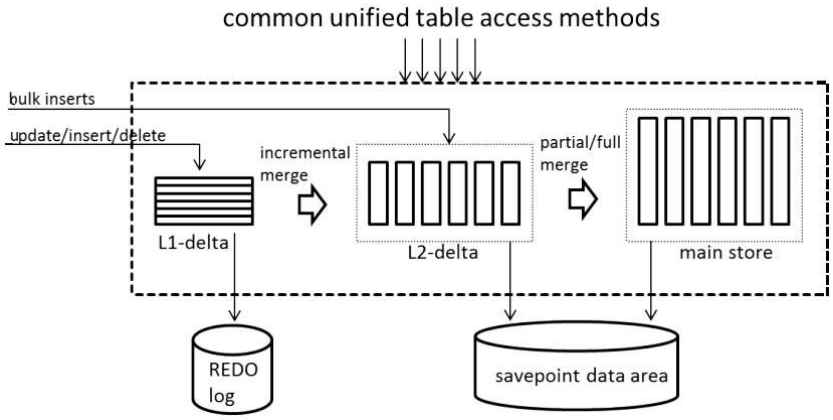


Figura 4.19. Esquema de tabela unificada implementada por SAP HANA [Sikka et al. 2012].

Apenas a parte *delta* do banco de dados manuseia operações OLTP: *L1-delta* para inserção, atualização e exclusão ou *L2-delta* para operações em lote. *Main store* é utilizado para operações OLAP. A medida que um dado é pouco atualizado, ele passa do armazenamento *L1-delta* para *L2-delta* e depois de *L2-delta* para *Main store* [Färber et al. 2012, Sikka et al. 2012].

SAP HANA implementa *logging* de operações e *savepoints* (*snapshots*). Transações que operam em *L1 e L2-delta* enviam registros lógicos de REDO para um arquivo de *log*. *Savepoints* são gerados apenas para as partes *L2-delta* e *Main store*. Após um *crash*, o sistema carrega o último *savepoint* na memória e refaz as ações do arquivo de *log* [Färber et al. 2012, Sikka et al. 2012].

4.6.1.5. SiloR

SiloR utiliza um esquema de épocas que implementa um número global *E* embutido no ID das transações. Uma *thread* é responsável por gerar o número *E* que é uma espécie de contador incrementado a cada 40 milissegundos. A abordagem de épocas impacta na maneira como SiloR implementa tolerância a falhas [Zheng et al. 2014].

SiloR armazena chaves e valores no *log* ao invés dos registros lógicos tipicamente implementados em sistemas em memória. O *logging* de valores permite uma recuperação em paralelo. Contudo, essa estratégia envia mais dados para o *log* aumentando o tempo de execução das transações. Os registros podem ser armazenados em vários arquivos de *log* e em qualquer ordem, uma vez que o número da época controla a ordem de execução das transações [Zheng et al. 2014].

O mecanismo de *checkpoint* divide o banco de dados em diferentes partes e atribui cada uma dessas partes a uma *thread* diferente. Essas *threads* copiam suas partes do banco de dados para arquivos de *checkpoint* diferentes. Como transações executam durante o processo de *checkpoint*, as *threads* podem não ver todas as modificações. Assim, SiloR implementa um *checkpoint* "*fuzzy*" [Zheng et al. 2014].

Após uma falha, *threads* carregam os arquivos de *checkpoint* na memória em paralelo. Em seguida, os arquivos de *log* são processados em paralelo. Ao final do processamento, cada dado é associado ao valor da sua última modificação. Então o *log* é reaplicado em ordem reversa. Essa abordagem permite que valores não sejam sobrescritos e, consequentemente, a CPU seja usada mais eficientemente [Zheng et al. 2014].

4.6.2. Recuperação em paralelo

Alguns sistemas em memória implementam uma estratégia de recuperação em paralelo com o objetivo de acelerá-la. Esses sistemas requerem que o arquivo de *log* grave registros ao nível de transação. Nessa caso, cada transação deve ser um procedimento armazenado previamente cadastrado, como descrito na Seção 4.5.1.

4.6.2.1. PACMAN

PACMAN é uma estratégia de recuperação implementada no banco de dados Peloton [Pavlo et al. 2017]. Sempre que um procedimento armazenado é compilado, PACMAN

divide-o em fatias (*slices*). Por exemplo, o procedimento *Deposit* da Figura 4.20 (a) é divido 3 fatias. As fatias são grupos de operações do procedimento que podem ser potencialmente executadas em paralelo. Em seguida, é construído um grafo de dependência local que identifica restrições de execução entre as fatias do procedimento armazenado. As restrições verificam se as operações conflitam em escrita ou devem obedecer uma determinada ordem. Assim, o grafo pode identificar possíveis oportunidades de execução em paralelo entre as fatias. A Figura 4.20 (b) apresenta o grafo de dependência construido para o procedimento *Deposit* [Wu et al. 2017]. Nesse grafo, as fatias D_2 e D_3 pode executar em paralelo, mas só podem iniciar após o final da execução de D_1.

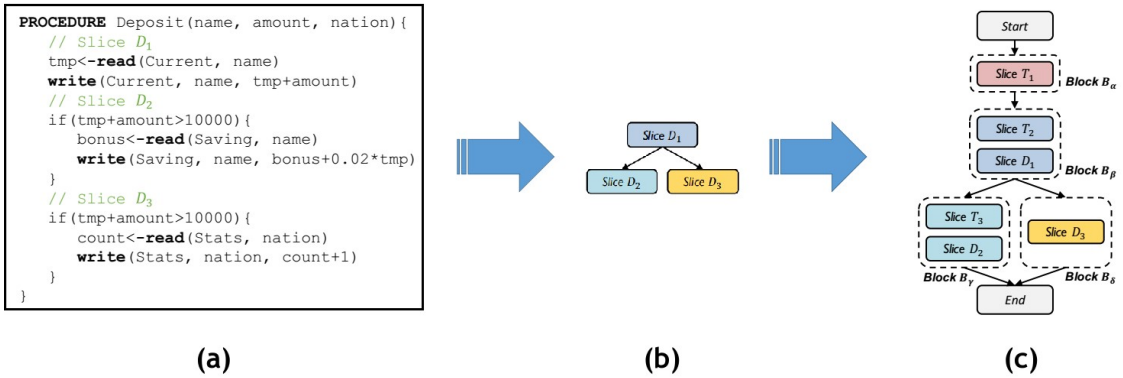


Figura 4.20. Procedimento armazenado (a), grafo de dependência local (b) e grafo de dependência global (c) [Wu et al. 2017].

O grafo de dependência local de cada procedimento armazenado é integrado a um grafo de dependência global (Figura 4.20 (c)) que representa a ordem de execução de todas as fatias de todos os procedimentos armazenados do sistema. Através do grafo de dependência global, PACMAN é capaz produzir escalonamentos de execução das fatias dos procedimentos. Cada escalonamento pode executar em paralelo. Durante uma recuperação, dinamicamente, PACMAN tenta otimizar ainda mais o processo de recuperação. PACMAN permite mais execuções paralelas explorando a disponibilidade dos valores de parâmetros de procedimentos em tempo de execução e aplicando execução em *pipeline* [Wu et al. 2017].

4.6.2.2. Adaptive Logging

Adaptive Logging é uma estratégia de recuperação paralela implementada no SGBD H-Store. Essa estratégia é capaz de adaptar-se para melhorar o desempenho do processo de recuperação escolhendo quando é melhor utilizar *logging* de transações ou ARIES [Yao et al. 2016].

Após uma falha, antes de começar o processo de recuperação, o sistema percorre o arquivo de *log* para gerar um grafo de dependência entre as transações. Esse grafo relaciona transações que possuem escrita nos mesmos registros. Quando os registros de *log* estão sendo refeitos, transações sem relacionamento pode executar em paralelo. Caso contrário, as transações devem esperar até que suas dependências terminem de executar [Yao et al. 2016].

A dependência entre transações pode fazer poucas transações bloquearem muitas outras. Adaptive Logging pode identificar gargalos durante uma recuperação utilizando um modelo de custo que usa um valor dado pelo usuário. Quando uma transação é identificada como gargalo, ela é materializada em *log* ARIES. Assim, transações dependentes do gargalo não precisam esperar muito [Yao et al. 2016].

4.6.3. Recuperação instantânea

Existem SGBDs em memória que usam estratégias de recuperação instantânea, ou seja, o sistema não precisa esperar por um longo período de tempo, após uma falha, para voltar a processar novas transações. Esses sistemas utilizam uma estrutura de índice no *log* ao invés do tradicional arquivo de sequencial [Magalhães et al. 2021a, Sauer 2019].

4.6.3.1. FineLine

FineLine implementa uma estratégia de recuperação instantânea cujo arquivo de *log* é uma árvore B particionada, descrita na Seção 4.4.3. Durante o processamento normal das transações, os registros de cada *group commit* são gravados em uma partição da árvore. Os registros de *log* armazenam páginas, ou seja, trata-se de um *log* físico [Sauer 2017, Sauer et al. 2018, Sauer 2019]. Essa abordagem prejudica o desempenho dos SGBDs em memória, uma vez que manusear um arquivo de índice é mais custoso do que um arquivo sequencial. Além disso, os registros de *log* físico são mais pesados do que os de *log* lógico.

Após uma falha, o sistema percorre o *log* indexado para recuperar páginas do banco de dados incrementalmente. Além disso, páginas podem ser buscadas na árvore para atender transações sob demanda. Para recuperar uma determinada página, o sistema deve percorrer múltiplas partições da árvore [Sauer 2017, Sauer et al. 2018, Sauer 2019].

FineLine não implementa *checkpoints*. Porém, o sistema mescla partições periodicamente para diminuir o tempo de recuperação de registros no *log*. Contudo, à medida que o arquivo de *log* aumenta, mais registros de *log* devem ser analisados durante uma recuperação após uma falha. Como consequência, quanto maior for *log*, mais lenta a recuperação será [Sauer 2017, Sauer et al. 2018, Sauer 2019].

4.6.3.2. HiEngine

HiEngine é um banco de dados desenvolvido pela Huawei [Huawei 2022]. Esse sistema armazena versões das tuplas do banco de dados em registros de um arquivo de *log* sequencial. Ele utiliza uma Árvore Radix Adaptativa (*Adaptive Radix Tree* - ART) como estrutura de indexação do *log*. A árvore funciona como um *array* indireto que aponta para os IDs das tuplas na memória ou para o endereço do último registro no *log* que modificou uma tupla. Essa abordagem permite a HiEngine carregar as tuplas do banco de dados na memória sob demanda. Contudo, manusear índices de *log* no processamento de transações pode degradar o desempenho do sistema. A Figura 4.21 ilustra a arquitetura de HiEngine [Lee et al. 2022, Ma et al. 2022].

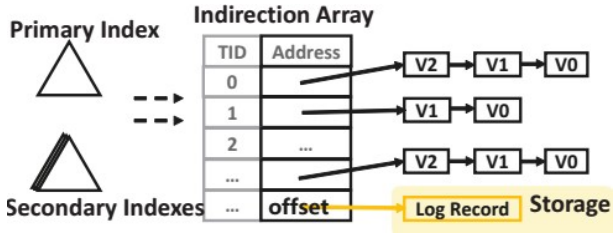


Figura 4.21. Arquitetura de HiEngine [Lee et al. 2022].

Após uma falha, HiEngine reconstrói o *array* indireto na memória e as tuplas são carregadas na memória sob demanda. Periodicamente, HiEngine produz *snapshots* da árvore. O *snapshot* armazena apenas o deslocamento para os registros no *log* ao invés dos registros. Dessa maneira, o tamanho do *snapshot* tende a ser pequeno e pode ser carregado na memória mais rapidamente. Segundo os autores, HiEngine pode recuperar o sistema após uma falha "instantaneamente"em aproximadamente 10 segundos e possui uma perda de desempenho aproximada de 5% a 11%. Adicionalmente, os arquivos de *log* e *snapshot* de HiEngine devem ser armazenados em um dispositivo de NVRAM, o que representa uma dependência dessa tecnologia [Lee et al. 2022].

4.6.3.3. MM-DIRECT

MM-DIRECT (*Main Memory Database Instant RECovery with Tuple consistent checkpoint* ou Recuperação instantânea de bancos de dados em memória com *checkpoint* consistente de tupla) é uma abordagem de recuperação implementada no banco de dados Redis [Redis 2020]. Essa abordagem permite a execução de novas transações imediatamente ao reinício do sistema após uma falha. Segundo os autores, o sistema possui um tempo de espera de aproximadamente 0,007 segundos após o seu reinício. MM-DIRECT requer um sistema simples contendo uma hierarquia de memória em dois níveis: (1) memória principal para o banco de dados e (2) memória persistente para os arquivos de *log* [Magalhães 2022, Magalhães et al. 2021a, Magalhães 2021].

Em MM-DIRECT, registros lógicos de REDO são enviados para um arquivo de *log* sequencial (Figura 4.22 (a)) durante o processamento normal das transações. Os registros são copiados do *log* sequencial para um arquivo de *log* indexado (Figura 4.22 (b)), que é uma árvore B^+. Esse processo é assíncrono ao processamento das transações, ou seja, as transações não precisam esperar pela escrita no *log* indexado para poder finalizar. Essa estratégia evita a degradação do desempenho do processamento de transações, uma vez que escrever registros para uma árvore B^+ é muito mais custoso do que escrevê-los para um arquivo sequencial. MM-DIRECT também suporta tabela *hash* como estrutura de *log* indexado [Magalhães 2022, Magalhães et al. 2021a, Magalhães 2021].

As chaves de busca da árvore B^+ são os IDs das tuplas do banco de dados. Cada nó da árvore contém uma lista dos registros de *log* que atualizaram uma determinada tupla. Os registros são ainda ordenados na lista pelo LSN. Dessa maneira, uma única busca na árvore pode encontrar todos os registros para recuperar uma tupla completamente na memória [Magalhães 2022, Magalhães et al. 2021a, Magalhães 2021].

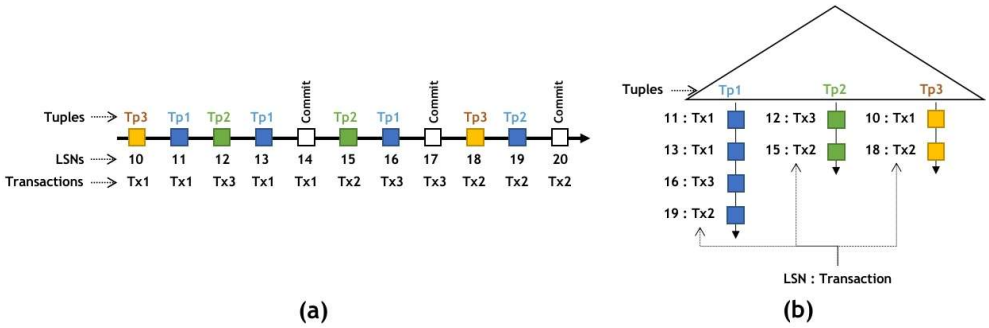


Figura 4.22. Log sequencial (a) e log indexado (b) [Magalhães et al. 2021a].

Após uma falha, MM-DIRECT percorre o *log* indexado para recuperar o banco de dados. Cada visita a um nó da árvore recupera um tupla completamente na memória. Assim que uma tupla é recuperada, ela pode ser acessada por novas transações. Contudo, caso uma transação precise de uma tupla que ainda não foi recuperada, a tupla pode ser recuperada sob demanda através de uma busca na árvore [Magalhães 2022, Magalhães et al. 2021a, Magalhães 2021].

MM-DIRECT implementa um mecanismo de *checkpoint* para reduzir o número de registros na árvore e, consequentemente, reduzir o tempo de recuperação após falha. Para cada nó da árvore, essa técnica substitui a lista de registros de *log* por um único registro cuja ação é equivalente à ação de todos os registros da lista. Assim, menos registros serão processados em caso de uma falha. Essa técnica ainda pode processar apenas os nós das tuplas mais frequentemente modificadas, diminuindo a sobrecarga do processo de *checkpoint*. Essa técnica de *checkpoint* é consistente, uma vez que suas modificações persistem caso o processo de *checkpoint* seja interrompido, por uma falha de sistema, por exemplo. [Magalhães 2022, Magalhães et al. 2021a, Magalhães 2021].

4.7. Principais desafios e direções futuras

Os bancos de dados em memória modernos tipicamente utilizam registros lógicos REDO-*only* com o objetivo de diminuir a quantidade de informação enviada para o *log* na memória secundária e, consequentemente, aumentar o desempenho no processamento das transações. Por outro lado, executar operações lógicas requer mais ciclos de CPU do que simplesmente copiar dados físicos para a memória. Como consequência, a recuperação em SGBDs em memória tende a ser mais lenta. Para acelerar a recuperação, alguns sistemas em memória têm utilizado alguma técnica de recuperação paralela [Magalhães et al. 2021b].

Algumas estratégias modernas de tolerância a falhas têm adotado técnicas radicalmente diferentes, como uso de estruturas de índice nos arquivos de *log* e *checkpoint*. Porém, manusear índices pode degradar o desempenho do sistema [Lee et al. 2022, Magalhães et al. 2021a, Sauer et al. 2018]. Além disso, criar *snapshots* de estruturas de índices não é tão simples quanto gerar *snapshots* do banco de dados ou do arquivo de *log* sequencial. HiEngine propôs alguns algoritmos de *snapshots* de *log* indexado, como ChainIndex, MirrorIndex e IACoW. Contudo, esses algoritmos ainda impõem alguma sobrecarga significativa ao processamento de transações [Lee et al. 2022].

Os sistemas de bancos de dados em memória têm tentado utilizar cada vez mais o *hardware* moderno. Por exemplo, HiEngine usa NVRAM para acessar os arquivos de *log* e *snapshot* mais rapidamente [Lee et al. 2022]. HyPer usa RDMA para fazer *backups* e livrar o sistema dessa tarefa [Kemper and Neumann 2011]. Além de prover ganhos de desempenho aos sistemas em memória, as novas tecnologias têm fornecido oportunidades que não eram possíveis com *hardware* antigo. Porém, os sistemas se tornam dependentes dessas tecnologias.

4.8. Conclusões

Esse trabalho elucida as principais questões relacionadas a recuperação de bancos de dados, principalmente as relacionadas a bancos de dados em memória. Para isso, o trabalho provê uma visão geral das escolhas arquiteturais, de tecnologia e implementação de bancos de dados em memória e suas principais estratégias de recuperação após falhas de uma amostra representativa dos SGBDs em memória.

Referências

[Ailamaki et al. 1999] Ailamaki, A., DeWitt, D. J., Hill, M. D., and Wood, D. A. (1999). Dbmss on a modern processor: Where does time go? In Atkinson, M. P., Orlowska, M. E., Valduriez, P., Zdonik, S. B., and Brodie, M. L., editors, *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*, pages 266–277. Morgan Kaufmann.

[Apache Software Foundation 2022] Apache Software Foundation (2022). Apache storm. Disponível em: "http://gridgain.com/". Acessado em: 19/12/2022.

[Arulraj and Pavlo 2017] Arulraj, J. and Pavlo, A. (2017). How to build a non-volatile memory database management system. In Salihoglu, S., Zhou, W., Chirkova, R., Yang, J., and Suciu, D., editors, *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 1753–1758. ACM.

[Arulraj et al. 2015] Arulraj, J., Pavlo, A., and Dulloor, S. (2015). Let's talk about storage & recovery methods for non-volatile memory database systems. In Sellis, T. K., Davidson, S. B., and Ives, Z. G., editors, *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 707–722. ACM.

[Arulraj et al. 2016a] Arulraj, J., Pavlo, A., and Menon, P. (2016a). Bridging the archipelago between row-stores and column-stores for hybrid workloads. In Özcan, F., Koutrika, G., and Madden, S., editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 583–598. ACM.

[Arulraj et al. 2016b] Arulraj, J., Perron, M., and Pavlo, A. (2016b). Write-behind logging. *Proceedings of the VLDB Endowment*, 10(4):337–348.

[Bernstein et al. 1987] Bernstein, P. A., Hadzilacos, V., and Goodman, N. (1987). *Concurrency Control and Recovery in Database Systems*. Addison-Wesley.

[Binna et al. 2018] Binna, R., Zangerle, E., Pichl, M., Specht, G., and Leis, V. (2018). HOT: A height optimized trie index for main-memory database systems. In Das, G., Jermaine, C. M., and Bernstein, P. A., editors, *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 521–534. ACM.

[Bishop et al. 2011] Bishop, B., Kiryakov, A., Ognyanoff, D., Peikov, I., Tashev, Z., and Velkov, R. (2011). OWLIM: A family of scalable semantic repositories. *Semantic Web*, 2(1):33–42.

[Bitton et al. 1987] Bitton, D., Hanrahan, M., and Turbyfill, C. (1987). Performance of complex queries in main memory database systems. In *Proceedings of the Third International Conference on Data Engineering, February 3-5, 1987, Los Angeles, California, USA*, pages 72–81. IEEE Computer Society.

[Bronevetsky et al. 2006] Bronevetsky, G., Fernandes, R., Marques, D., Pingali, K., and Stodghill, P. (2006). Recent advances in checkpoint/recovery systems. In *20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25-29 April 2006, Rhodes Island, Greece*. IEEE.

[Cha and Song 2004] Cha, S. K. and Song, C. (2004). P*time: Highly scalable OLTP DBMS for managing update-intensive stream workload. In Nascimento, M. A., Özsu, M. T., Kossmann, D., Miller, R. J., Blakeley, J. A., and Schiefer, K. B., editors, *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004*, pages 1033–1044. Morgan Kaufmann.

[Chen et al. 2001] Chen, S., Gibbons, P. B., and Mowry, T. C. (2001). Improving index performance through prefetching. In Mehrotra, S. and Sellis, T. K., editors, *Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Santa Barbara, CA, USA, May 21-24, 2001*, pages 235–246. ACM.

[Chen et al. 2011a] Chen, S., Gibbons, P. B., and Nath, S. (2011a). Rethinking database algorithms for phase change memory. In *CIDR 2011, Fifth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 9-12, 2011, Online Proceedings*, pages 21–31. www.cidrdb.org.

[Chen et al. 2011b] Chen, S., Gibbons, P. B., and Nath, S. (2011b). Rethinking database algorithms for phase change memory. In *CIDR 2011, Fifth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 9-12, 2011, Online Proceedings*, pages 21–31. www.cidrdb.org.

[Cheng et al. 1984] Cheng, J. M., Looseley, C. R., Shibamiya, A., and Worthington, P. S. (1984). IBM database 2 performance: Design, implementation, and tuning. *IBM Systems Journal*, 23(2):189–210.

[Chhugani et al. 2008] Chhugani, J., Nguyen, A. D., Lee, V. W., Macy, W., Hagog, M., Chen, Y., Baransi, A., Kumar, S., and Dubey, P. (2008). Efficient implementation of sorting on multi-core SIMD CPU architecture. *Proceedings of the VLDB Endowment*, 1(2):1313–1324.

[Clark and Corrigan 1989] Clark, B. E. and Corrigan, M. J. (1989). Application system/400 performance characteristics. *IBM Systems Journal*, 28(3):407–423.

[Condie et al. 2010] Condie, T., Conway, N., Alvaro, P., Hellerstein, J. M., Elmeleegy, K., and Sears, R. (2010). Mapreduce online. In *Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2010, April 28-30, 2010, San Jose, CA, USA*, pages 313–328. USENIX Association.

[Copeland and Khoshafian 1985] Copeland, G. P. and Khoshafian, S. (1985). A decomposition storage model. In Navathe, S. B., editor, *Proceedings of the 1985 ACM SIGMOD International Conference on Management of Data, Austin, Texas, USA, May 28-31, 1985*, pages 268–279. ACM Press.

[Crus 1984] Crus, R. A. (1984). Data recovery in IBM database 2. *IBM Systems Journal*, 23(2):178–188.

[DeWitt et al. 1984] DeWitt, D. J., Katz, R. H., Olken, F., Shapiro, L. D., Stonebraker, M., and Wood, D. A. (1984). Implementation techniques for main memory database systems. In Yormark, B., editor, *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, USA, June 18-21, 1984*, pages 1–8. ACM Press.

[Diaconu et al. 2013] Diaconu, C., Freedman, C., Ismert, E., Larson, P., Mittal, P., Stonecipher, R., Verma, N., and Zwilling, M. (2013). Hekaton: SQL server's memory-optimized OLTP engine. In Ross, K. A., Srivastava, D., and Papadias, D., editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 1243–1254. ACM.

[Dragojevic et al. 2014] Dragojevic, A., Narayanan, D., Castro, M., and Hodson, O. (2014). Farm: Fast remote memory. In Mahajan, R. and Stoica, I., editors, *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2014, Seattle, WA, USA, April 2-4, 2014*, pages 401–414. USENIX Association.

[Eich 1986] Eich, M. H. (1986). Main memory database recovery. In *Proceedings of the Fall Joint Computer Conference, November 2-6, 1986, Dallas, Texas, USA*, pages 1226–1232. IEEE Computer Society.

[Eich 1987a] Eich, M. H. (1987a). A classification and comparison of main memory database recovery techniques. In *Proceedings of the Third International Conference on Data Engineering, February 3-5, 1987, Los Angeles, California, USA*, pages 332–339. IEEE Computer Society.

[Eich 1987b] Eich, M. H. (1987b). MARS: the design of a main memory database machine. In Kitsuregawa, M. and Tanaka, H., editors, *Database Machines and Knowledge Base Machines, 5th International Workshop on Database Machines, Tokyo, Japan, 1987, Proceedings*, volume 43 of *The Kluwer International Series in Engineering and Computer Science*, pages 325–338. Kluwer.

[Elmasri and Navathe 2000] Elmasri, R. and Navathe, S. B. (2000). *Fundamentals of Database Systems, 3rd Edition*. Addison-Wesley-Longman.

[Faerber et al. 2017] Faerber, F., Kemper, A., Larson, P., Levandoski, J. J., Neumann, T., and Pavlo, A. (2017). Main memory database systems. *Foundations and Trends in Databases*, 8(1-2):1–130.

[Fan et al. 2013] Fan, B., Andersen, D. G., and Kaminsky, M. (2013). Memc3: Compact and concurrent memcache with dumber caching and smarter hashing. In Feamster, N. and Mogul, J. C., editors, *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2013, Lombard, IL, USA, April 2-5, 2013*, pages 371–384. USENIX Association.

[Färber et al. 2011] Färber, F., Cha, S. K., Primsch, J., Bornhövd, C., Sigg, S., and Lehner, W. (2011). SAP HANA database: data management for modern business applications. *ACM Sigmod Record*, 40(4):45–51.

[Färber et al. 2012] Färber, F., May, N., Lehner, W., Große, P., Müller, I., Rauhe, H., and Dees, J. (2012). The SAP HANA database – an architecture overview. *IEEE Database Engineering Bulletin*, 35(1):28–33.

[Freedman et al. 2014] Freedman, C., Ismert, E., and Larson, P. (2014). Compilation in the microsoft SQL server hekaton engine. *IEEE Database Engineering Bulletin*, 37(1):22–30.

[Funke et al. 2014] Funke, F., Kemper, A., Mühlbauer, T., Neumann, T., and Leis, V. (2014). Hyper beyond software: Exploiting modern hardware for main-memory database systems. *Datenbank-Spektrum*, 14(3):173–181.

[Garcia-Molina and Salem 1992] Garcia-Molina, H. and Salem, K. (1992). Main memory database systems: An overview. *IEEE Transactions on Knowledge and Data*, 4(6):509–516.

[Garg et al. 2015] Garg, V., Singh, A., and Haritsa, J. R. (2015). On improving write performance in pcm databases. Technical report, Technical report, TR-2015-01, IISc.

[Graefe 2003] Graefe, G. (2003). Sorting and indexing with partitioned b-trees. In *First Biennial Conference on Innovative Data Systems Research, CIDR 2003, Asilomar, CA, USA, January 5-8, 2003, Online Proceedings*. www.cidrdb.org.

[Graefe et al. 2016] Graefe, G., Guy, W., and Sauer, C. (2016). *Instant Recovery with Write-Ahead Logging: Page Repair, System Restart, Media Restore, and System Failover, Second Edition*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

[Graefe and Kuno 2012] Graefe, G. and Kuno, H. A. (2012). Definition, detection, and recovery of single-page failures, a fourth class of database failures. *Proceedings of the VLDB Endowment*, 5(7):646–655.

[Graefe and McKenna 1993] Graefe, G. and McKenna, W. J. (1993). The volcano optimizer generator: Extensibility and efficient search. In *Proceedings of the Ninth International Conference on Data Engineering, April 19-23, 1993, Vienna, Austria*, pages 209–218. IEEE Computer Society.

[Graefe et al. 2015] Graefe, G., Sauer, C., Guy, W., and Härder, T. (2015). Instant recovery with write-ahead logging. *Datenbank-Spektrum*, 15(3):235–239.

[Gray et al. 1981] Gray, J., McJones, P. R., Blasgen, M. W., Lindsay, B. G., Lorie, R. A., Price, T. G., Putzolu, G. R., and Traiger, I. L. (1981). The recovery manager of the system R database manager. *ACM Computing Surveys (CSUR)*, 13(2):223–243.

[Gray and Reuter 1993] Gray, J. and Reuter, A. (1993). *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann.

[GridGain Team 2022] GridGain Team (2022). Gridgain — extreme speed and scale for data-intensive apps | gridgain systems. Disponível em: "http://gridgain.com/". Acessado em: 19/12/2022.

[Gruenwald et al. 1996] Gruenwald, L., Huang, J., Dunham, Margaret H anA model of crash recovery in main memory databased Lin, J.-L., and Peltier, A. C. (1996). Recovery in main memory databases.

[Hagmann 1986] Hagmann, R. B. (1986). Crash recovery scheme for a memory-resident database system. *IEEE Computer Architecture Letters*, 35(9):839–843.

[Hagmann 1987] Hagmann, R. B. (1987). Reimplementing the cedar file system using logging and group commit. In Belady, L., editor, *Proceedings of the Eleventh ACM Symposium on Operating System Principles, SOSP 1987, Stouffer Austin Hotel, Austin, Texas, USA, November 8-11, 1987*, pages 155–162. ACM.

[Hammond et al. 2004] Hammond, L., Wong, V., Chen, M. K., Carlstrom, B. D., Davis, J. D., Hertzberg, B., Prabhu, M. K., Wijaya, H., Kozyrakis, C., and Olukotun, K. (2004). Transactional memory coherence and consistency. In *31st International Symposium on Computer Architecture (ISCA 2004), 19-23 June 2004, Munich, Germany*, pages 102–113. IEEE Computer Society.

[Härder and Reuter 1983] Härder, T. and Reuter, A. (1983). Principles of transaction-oriented database recovery. *ACM Computing Surveys (CSUR)*, 15(4):287–317.

[Harris 2016] Harris, R. (2016). Windows leaps into the nvm revolution. Disponível em: "https://www.zdnet.com/article/windows-leaps-into-the-nvm-revolution/". Acessado em: 22/07/2020.

[Harris et al. 2007] Harris, T., Cristal, A., Unsal, O. S., Ayguadé, E., Gagliardi, F., Smith, B., and Valero, M. (2007). Transactional memory: An overview. *IEEE Micro*, 27(3):8–29.

[Hazenberg and Hemminga 2011] Hazenberg, W. and Hemminga, S. (2011). Main memory database systems: Opportunities and pitfalls. *SC@ RUG 2011 proceedings*, page 113.

[Herlihy and Moss 1993] Herlihy, M. and Moss, J. E. B. (1993). Transactional memory: Architectural support for lock-free data structures. In Smith, A. J., editor, *Proceedings of the 20th Annual International Symposium on Computer Architecture, San Diego, CA, USA, May 1993*, pages 289–300. ACM.

[Huawei 2022] Huawei (2022). Huawei - building a fully connected, intelligent world. Disponível em: "https://www.huawei.com". Acessado em: 22/04/2022.

[Hvasshovd et al. 1995] Hvasshovd, S., Torbjørnsen, Ø., Bratsberg, S. E., and Holager, P. (1995). The clustra telecom database: High availability, high throughput, and real-time response. In Dayal, U., Gray, P. M. D., and Nishio, S., editors, *VLDB'95, Proceedings of 21th International Conference on Very Large Data Bases, September 11-15, 1995, Zurich, Switzerland*, pages 469–477. Morgan Kaufmann.

[Infinite Graph 2023] Infinite Graph (2023). Infinitegraph – limitless data possibilities. Disponível em: "https://infinitegraph.com". Acessado em: 12/05/2023.

[Jagadish et al. 1994] Jagadish, H. V., Lieuwen, D. F., Rastogi, R., Silberschatz, A., and Sudarshan, S. (1994). Dalí: A high performance main memory storage manager. In Bocca, J. B., Jarke, M., and Zaniolo, C., editors, *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, pages 48–59. Morgan Kaufmann.

[Jagadish et al. 1993] Jagadish, H. V., Silberschatz, A., and Sudarshan, S. (1993). Recovering from main-memory lapses. In Agrawal, R., Baker, S., and Bell, D. A., editors, *19th International Conference on Very Large Data Bases, August 24-27, 1993, Dublin, Ireland, Proceedings*, pages 391–404. Morgan Kaufmann.

[Kallman et al. 2008] Kallman, R., Kimura, H., Natkins, J., Pavlo, A., Rasin, A., Zdonik, S. B., Jones, E. P. C., Madden, S., Stonebraker, M., Zhang, Y., Hugg, J., and Abadi, D. J. (2008). H-store: a high-performance, distributed main memory transaction processing system. *Proceedings of the VLDB Endowment*, 1(2):1496–1499.

[Karnagel et al. 2014] Karnagel, T., Dementiev, R., Rajwar, R., Lai, K., Legler, T., Schlegel, B., and Lehner, W. (2014). Improving in-memory database index performance with intel® transactional synchronization extensions. In *20th IEEE International Symposium on High Performance Computer Architecture, HPCA 2014, Orlando, FL, USA, February 15-19, 2014*, pages 476–487. IEEE Computer Society.

[Kemper and Neumann 2011] Kemper, A. and Neumann, T. (2011). Hyper: A hybrid oltp&olap main memory database system based on virtual memory snapshots. In Abiteboul, S., Böhm, K., Koch, C., and Tan, K., editors, *Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany*, pages 195–206. IEEE Computer Society.

[Kim et al. 2010] Kim, C., Chhugani, J., Satish, N., Sedlar, E., Nguyen, A. D., Kaldewey, T., Lee, V. W., Brandt, S. A., and Dubey, P. (2010). FAST: fast architecture sensitive tree search on modern cpus and gpus. In Elmagarmid, A. K. and Agrawal, D., editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*, pages 339–350. ACM.

[Lahiri et al. 2013] Lahiri, T., Neimat, M., and Folkman, S. (2013). Oracle timesten: An in-memory database for enterprise applications. *IEEE Database Engineering Bulletin*, 36(2):6–13.

[Larson and Levandoski 2016] Larson, P. and Levandoski, J. J. (2016). Modern main-memory database systems. *Proceedings of the VLDB Endowment*, 9(13):1609–1610.

[Larson et al. 2013] Larson, P., Zwilling, M., and Farlee, K. (2013). The hekaton memory-optimized OLTP engine. *IEEE Database Engineering Bulletin*, 36(2):34–40.

[Lee et al. 2013] Lee, J., Kwon, Y. S., Färber, F., Muehle, M., Lee, C., Bensberg, C., Lee, J., Lee, A. H., and Lehner, W. (2013). SAP HANA distributed in-memory database system: Transaction, session, and metadata management. In Jensen, C. S., Jermaine, C. M., and Zhou, X., editors, *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, pages 1165–1173. IEEE Computer Society.

[Lee et al. 2022] Lee, L., Xie, S., Ma, Y., and Chen, S. (2022). Index checkpoints for instant recovery in in-memory database systems. *Proc. VLDB Endow.*, 15(8):1671–1683.

[Lehman and Carey 1987] Lehman, T. J. and Carey, M. J. (1987). A recovery algorithm for A high-performance memory-resident database system. In Dayal, U. and Traiger, I. L., editors, *Proceedings of the Association for Computing Machinery Special Interest Group on Management of Data 1987 Annual Conference, San Francisco, CA, USA, May 27-29, 1987*, pages 104–117. ACM Press.

[Lehman et al. 1992] Lehman, T. J., Shekita, E. J., and Cabrera, L. (1992). An evaluation of starburst's memory resident storage component. *IEEE Transactions on Knowledge and Data*, 4(6):555–566.

[Leis et al. 2014a] Leis, V., Boncz, P. A., Kemper, A., and Neumann, T. (2014a). Morsel-driven parallelism: a numa-aware query evaluation framework for the many-core age. In Dyreson, C. E., Li, F., and Özsu, M. T., editors, *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 743–754. ACM.

[Leis et al. 2013] Leis, V., Kemper, A., and Neumann, T. (2013). The adaptive radix tree: Artful indexing for main-memory databases. In Jensen, C. S., Jermaine, C. M., and Zhou, X., editors, *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, pages 38–49. IEEE Computer Society.

[Leis et al. 2014b] Leis, V., Kemper, A., and Neumann, T. (2014b). Exploiting hardware transactional memory in main-memory databases. In Cruz, I. F., Ferrari, E., Tao, Y., Bertino, E., and Trajcevski, G., editors, *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*, pages 580–591. IEEE Computer Society.

[Levandoski et al. 2013] Levandoski, J. J., Lomet, D. B., and Sengupta, S. (2013). The bw-tree: A b-tree for new hardware platforms. In Jensen, C. S., Jermaine, C. M., and Zhou, X., editors, *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, pages 302–313. IEEE Computer Society.

[Li and Naughton 1988] Li, K. and Naughton, J. F. (1988). Multiprocessor main memory transaction processing. In Jajodia, S., Kim, W., and Silberschatz, A., editors, *Proceedings of the International Symposium on Databases in Parallel and Distributed Systems, Austin, Texas, USA, December 5-7, 1988*, pages 177–187. IEEE Computer Society.

[Li et al. 2018a] Li, L., Wang, G., Wu, G., and Yuan, Y. (2018a). Consistent snapshot algorithms for in-memory database systems: Experiments and analysis. In *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*, pages 1284–1287. IEEE Computer Society.

[Li et al. 2018b] Li, L., Wang, G., Wu, G., Yuan, Y., Chen, L., and Lian, X. (2018b). A comparative study of consistent snapshot algorithms for main-memory database systems. *CoRR*, abs/1810.04915.

[Li and Eich 1993] Li, X. and Eich, M. H. (1993). Post-crash log processing for fuzzy checkpointing main memory databases. In *Proceedings of the Ninth International Conference on Data Engineering, April 19-23, 1993, Vienna, Austria*, pages 117–124. IEEE Computer Society.

[Li et al. 2013] Li, Y., Pandis, I., Müller, R., Raman, V., and Lohman, G. M. (2013). Numa-aware algorithms: the case of data shuffling. In *CIDR 2013, Sixth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 6-9, 2013, Online Proceedings*. www.cidrdb.org.

[Liedes and Wolski 2006] Liedes, A. and Wolski, A. (2006). SIREN: A memory-conserving, snapshot-consistent checkpoint algorithm for in-memory databases. In Liu, L., Reuter, A., Whang, K., and Zhang, J., editors, *Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, 3-8 April 2006, Atlanta, GA, USA*, page 99. IEEE Computer Society.

[Lim et al. 2014] Lim, H., Han, D., Andersen, D. G., and Kaminsky, M. (2014). MICA: A holistic approach to fast in-memory key-value storage. In Mahajan, R. and Stoica, I., editors, *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2014, Seattle, WA, USA, April 2-4, 2014*, pages 429–444. USENIX Association.

[Lindström et al. 2013] Lindström, J., Raatikka, V., Ruuth, J., Soini, P., and Vakkila, K. (2013). IBM soliddb: In-memory database optimized for extreme speed and availability. *IEEE Database Engineering Bulletin*, 36(2):14–20.

[Lomet et al. 2012] Lomet, D. B., Fekete, A. D., Wang, R., and Ward, P. (2012). Multi-version concurrency via timestamp range conflict management. In Kementsietsidis, A. and Salles, M. A. V., editors, *IEEE 28th International Conference on Data Engineering (ICDE 2012), Washington, DC, USA (Arlington, Virginia), 1-5 April, 2012*, pages 714–725. IEEE Computer Society.

[Low et al. 2012] Low, Y., Gonzalez, J., Kyrola, A., Bickson, D., Guestrin, C., and Hellerstein, J. M. (2012). Distributed graphlab: A framework for machine learning in the cloud. *arXiv preprint arXiv:1204.6078*.

[Ma et al. 2022] Ma, Y., Xie, S., Zhong, H., Lee, L., and Lv, K. (2022). Hiengine: How to architect a cloud-native memory-optimized database engine. In Ives, Z., Bonifati, A., and Abbadi, A. E., editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 2177–2190. ACM.

[Maas et al. 2013] Maas, L. M., Kissinger, T., Habich, D., and Lehner, W. (2013). BUZZARD: a numa-aware in-memory indexing system. In Ross, K. A., Srivastava, D., and Papadias, D., editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 1285–1286. ACM.

[Magalhães 2021] Magalhães, A. (2021). Main memory databases instant recovery. In Bernstein, P. A. and Rabl, T., editors, *Proceedings of the VLDB 2021 PhD Workshop co-located with the 47th International Conference on Very Large Databases (VLDB 2021), Copenhagen, Denmark, August 16, 2021*, volume 2971 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Magalhães 2022] Magalhães, A. (2022). *Main memory database instant recovery*. PhD thesis, Federal University of Ceara, Brazil.

[Magalhaes et al. 2022] Magalhaes, A., Brayner, A., and Monteiro, J. M. (2022). Main memory database recovery strategies. In *Anais Estendidos do XXXVII Simpósio Brasileiro de Bancos de Dados*, pages 175–180. SBC.

[Magalhaes et al. 2023] Magalhaes, A., Brayner, A., and Monteiro, J. M. (2023). Main memory database recovery strategies. In *SIGMOD/PODS '23: Companion of the 2023 International Conference on Management of Data*, pages 31–35.

[Magalhães et al. 2021a] Magalhães, A., Brayner, A., Monteiro, J. M., and Moraes, G. (2021a). Indexed log file: Towards main memory database instant recovery. In Velegrakis, Y., Zeinalipour-Yazti, D., Chrysanthis, P. K., and Guerra, F., editors, *Proceedings of the 24th International Conference on Extending Database Technology, EDBT 2021, Nicosia, Cyprus, March 23 - 26, 2021*, pages 355–360. OpenProceedings.org.

[Magalhães et al. 2017] Magalhães, A., Monteiro, J. M., and Brayner, A. (2017). Ajuste de performance em bancos de dados nosql. In *III Escola Reginal de Informática do Piauí , ERIPI 2017, Picos, PI, Brazil, 2017*.

[Magalhães et al. 2018a] Magalhães, A., Monteiro, J. M., and Brayner, A. (2018a). Gerenciamento e processamento de big data com bancos de dados em memória. In *I Jornada latino-americana de atualização em informática , JOLAI 2018, São Paulo, SP, Brazil, 2018*.

[Magalhães et al. 2018b] Magalhães, A., Monteiro, J. M., and Brayner, A. (2018b). Sistemas de gerenciamento de banco de dados em memória. In *XIV Simpósio Brasileiro de Sistemas de Informação , SBSI 2018, Caxias do Sul, RS, Brazil, 2018*.

[Magalhães et al. 2021b] Magalhães, A., Monteiro, J. M., and Brayner, A. (2021b). Main memory database recovery: A survey. *ACM Comput. Surv.*, 54(2):46:1–46:36.

[Makreshanski et al. 2015] Makreshanski, D., Levandoski, J. J., and Stutsman, R. (2015). To lock, swap, or elide: On the interplay of hardware transactional memory and lock-free indexing. *Proceedings of the VLDB Endowment*, 8(11):1298–1309.

[Malewicz et al. 2010] Malewicz, G., Austern, M. H., Bik, A. J. C., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G. (2010). Pregel: a system for large-scale graph processing. In Elmagarmid, A. K. and Agrawal, D., editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*, pages 135–146. ACM.

[Malviya et al. 2014] Malviya, N., Weisberg, A., Madden, S., and Stonebraker, M. (2014). Rethinking main memory OLTP recovery. In Cruz, I. F., Ferrari, E., Tao, Y., Bertino, E., and Trajcevski, G., editors, *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*, pages 604–615. IEEE Computer Society.

[Mao et al. 2012] Mao, Y., Kohler, E., and Morris, R. T. (2012). Cache craftiness for fast multicore key-value storage. In Felber, P., Bellosa, F., and Bos, H., editors, *European Conference on Computer Systems, Proceedings of the Seventh EuroSys Conference 2012, EuroSys '12, Bern, Switzerland, April 10-13, 2012*, pages 183–196. ACM.

[Menon et al. 2017] Menon, P., Pavlo, A., and Mowry, T. C. (2017). Relaxed operator fusion for in-memory databases: Making compilation, vectorization, and prefetching work together at last. *Proceedings of the VLDB Endowment*, 11(1):1–13.

[Mitchell et al. 2013] Mitchell, C., Geng, Y., and Li, J. (2013). Using one-sided RDMA reads to build a fast, cpu-efficient key-value store. In Birrell, A. and Sirer, E. G., editors, *2013 USENIX Annual Technical Conference, San Jose, CA, USA, June 26-28, 2013*, pages 103–114. USENIX Association.

[Mohan 1990] Mohan, C. (1990). ARIES/KVL: A key-value locking method for concurrency control of multiaction transactions operating on b-tree indexes. In McLeod, D., Sacks-Davis, R., and Schek, H., editors, *16th International Conference on Very Large Data Bases, August 13-16, 1990, Brisbane, Queensland, Australia, Proceedings*, pages 392–405. Morgan Kaufmann.

[Mohan 1993] Mohan, C. (1993). ARIES/LHS: A concurrency control and recovery method using write-ahead logging for linear hashing with separators. In *Proceedings of the Ninth International Conference on Data Engineering, April 19-23, 1993, Vienna, Austria*, pages 243–252. IEEE Computer Society.

[Mohan 1999] Mohan, C. (1999). Repeating history beyond ARIES. In Atkinson, M. P., Orlowska, M. E., Valduriez, P., Zdonik, S. B., and Brodie, M. L., editors, *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*, pages 1–17. Morgan Kaufmann.

[Mohan et al. 1992] Mohan, C., Haderle, D., Lindsay, B. G., Pirahesh, H., and Schwarz, P. M. (1992). ARIES: A transaction recovery method supporting fine-granularity locking and partial rollbacks using write-ahead logging. *ACM Transactions on Database Systems (TODS)*, 17(1):94–162.

[Mohan and Levine 1992] Mohan, C. and Levine, F. E. (1992). ARIES/IM: an efficient and high concurrency index management method using write-ahead logging. In Stonebraker, M., editor, *Proceedings of the 1992 ACM SIGMOD International Conference on Management of Data, San Diego, California, USA, June 2-5, 1992*, pages 371–380. ACM Press.

[Mohan and Narang 1994] Mohan, C. and Narang, I. (1994). ARIES/CSA: A method for database recovery in client-server architectures. In Snodgrass, R. T. and Winslett, M., editors, *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data, Minneapolis, Minnesota, USA, May 24-27, 1994*, pages 55–66. ACM Press.

[MongoDB Inc. 2022] MongoDB Inc. (2022). Mongodb: The developer data platform | mongodb. Disponível em: "http://www.mongodb.org". Acessado em: 19/12/2022.

[Mühe et al. 2011] Mühe, H., Kemper, A., and Neumann, T. (2011). How to efficiently snapshot transactional data: hardware or software controlled? In Harizopoulos, S. and Luo, Q., editors, *Proceedings of the Seventh International Workshop on Data Management on New Hardware, DaMoN 2011, Athens, Greece, June 13, 2011*, pages 17–26. ACM.

[MySQL 2020] MySQL (2020). Mysql. Disponível em: "http://www.mysql.com". Acessado em: 03/10/2020.

[MySQL 2022] MySQL (2022). Mysql. Disponível em: "http://www.mysql.com/". Acessado em: 20/12/2022.

[Neo4J 2023] Neo4J (2023). Neo4j graph data platform | graph database management system. Disponível em: "https://neo4j.com". Acessado em: 12/05/2023.

[Neumann 2011] Neumann, T. (2011). Efficiently compiling efficient query plans for modern hardware. *Proceedings of the VLDB Endowment*, 4(9):539–550.

[Neumann et al. 2015] Neumann, T., Mühlbauer, T., and Kemper, A. (2015). Fast serializable multi-version concurrency control for main-memory database systems. In Sellis, T. K., Davidson, S. B., and Ives, Z. G., editors, *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 677–689. ACM.

[Neumeyer et al. 2010] Neumeyer, L., Robbins, B., Nair, A., and Kesari, A. (2010). S4: distributed stream computing platform. In Fan, W., Hsu, W., Webb, G. I., Liu, B., Zhang, C., Gunopulos, D., and Wu, X., editors, *ICDMW 2010, The 10th IEEE International Conference on Data Mining Workshops, Sydney, Australia, 13 December 2010*, pages 170–177. IEEE Computer Society.

[Ooi et al. 2015] Ooi, B. C., Tan, K., Wang, S., Wang, W., Cai, Q., Chen, G., Gao, J., Luo, Z., Tung, A. K. H., Wang, Y., Xie, Z., Zhang, M., and Zheng, K. (2015). SINGA: A distributed deep learning platform. In Zhou, X., Smeaton, A. F., Tian, Q., Bulterman, D. C. A., Shen, H. T., Mayer-Patel, K., and Yan, S., editors, *Proceedings of the 23rd*

Annual ACM Conference on Multimedia Conference, MM '15, Brisbane, Australia, October 26 - 30, 2015, pages 685–688. ACM.

[Oracle 2020] Oracle (2020). Oracle | integrated cloud applications and platform services. Disponível em: "http://www.oracle.com". Acessado em: 04/12/2020.

[OrientDB 2023] OrientDB (2023). Home | orientdb community edition. Disponível em: "http://orientdb.org". Acessado em: 12/05/2023.

[Oukid et al. 2017] Oukid, I., Booss, D., Lespinasse, A., Lehner, W., Willhalm, T., and Gomes, G. (2017). Memory management techniques for large-scale persistent-main-memory systems. *Proceedings of the VLDB Endowment*, 10(11):1166–1177.

[Ousterhout et al. 2009] Ousterhout, J. K., Agrawal, P., Erickson, D., Kozyrakis, C., Leverich, J., Mazières, D., Mitra, S., Narayanan, A., Parulkar, G. M., Rosenblum, M., Rumble, S. M., Stratmann, E., and Stutsman, R. (2009). The case for ramclouds: scalable high-performance storage entirely in DRAM. *ACM SIGOPS Oper. Syst. Rev.*, 43(4):92–105.

[Pandis et al. 2011] Pandis, I., Tözün, P., Johnson, R., and Ailamaki, A. (2011). PLP: page latch-free shared-everything OLTP. *Proceedings of the VLDB Endowment*, 4(10):610–621.

[Pavlo et al. 2017] Pavlo, A., Angulo, G., Arulraj, J., Lin, H., Lin, J., Ma, L., Menon, P., Mowry, T. C., Perron, M., Quah, I., Santurkar, S., Tomasic, A., Toor, S., Aken, D. V., Wang, Z., Wu, Y., Xian, R., and Zhang, T. (2017). Self-driving database management systems. In *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research, Chaminade, CA, USA, January 8-11, 2017, Online Proceedings*. www.cidrdb.org.

[Porobic et al. 2014] Porobic, D., Liarou, E., Tözün, P., and Ailamaki, A. (2014). Atrapos: Adaptive transaction processing on hardware islands. In Cruz, I. F., Ferrari, E., Tao, Y., Bertino, E., and Trajcevski, G., editors, *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*, pages 688–699. IEEE Computer Society.

[Porobic et al. 2012] Porobic, D., Pandis, I., Branco, M., Tözün, P., and Ailamaki, A. (2012). OLTP on hardware islands. *Proceedings of the VLDB Endowment*, 5(11):1447–1458.

[Ramakrishnan and Gehrke 2003] Ramakrishnan, R. and Gehrke, J. (2003). *Database management systems (3. ed.)*. McGraw-Hill.

[Rao and Ross 2000] Rao, J. and Ross, K. A. (2000). Making b^+-trees cache conscious in main memory. In Chen, W., Naughton, J. F., and Bernstein, P. A., editors, *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, pages 475–486. ACM.

[Redis 2020] Redis (2020). Redis. Disponível em: "https://redis.io". Acessado em: 26/08/2020.

[Ren et al. 2016] Ren, K., Diamond, T., Abadi, D. J., and Thomson, A. (2016). Low-overhead asynchronous checkpointing in main-memory database systems. In Özcan, F., Koutrika, G., and Madden, S., editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 1539–1551. ACM.

[Ren et al. 2012] Ren, K., Thomson, A., and Abadi, D. J. (2012). Lightweight locking for main memory database systems. *Proceedings of the VLDB Endowment*, 6(2):145–156.

[Rothermel and Mohan 1989] Rothermel, K. and Mohan, C. (1989). ARIES/NT: A recovery method based on write-ahead logging for nested transactions. In Apers, P. M. G. and Wiederhold, G., editors, *Proceedings of the Fifteenth International Conference on Very Large Data Bases, August 22-25, 1989, Amsterdam, The Netherlands*, pages 337–346. Morgan Kaufmann.

[Sadalage and Fowler 2019] Sadalage, P. J. and Fowler, M. (2019). *NoSQL essencial: um guia conciso para o mundo emergente da persistência poliglota*. Novatec Editora.

[Salem and Garcia-Molina 1989] Salem, K. and Garcia-Molina, H. (1989). Checkpointing memory-resident databases. In *Proceedings of the Fifth International Conference on Data Engineering, February 6-10, 1989, Los Angeles, California, USA*, pages 452–462. IEEE Computer Society.

[Salem and Garcia-Molina 1990] Salem, K. and Garcia-Molina, H. (1990). System M: A transaction processing testbed for memory resident data. *IEEE Transactions on Knowledge and Data*, 2(1):161–172.

[Sauer 2017] Sauer, C. (2017). *Modern techniques for transaction-oriented database recovery*. PhD thesis, Kaiserslautern University of Technology, Germany.

[Sauer 2019] Sauer, C. (2019). Modern techniques for transaction-oriented database recovery. In Grust, T., Naumann, F., Böhm, A., Lehner, W., Härder, T., Rahm, E., Heuer, A., Klettke, M., and Meyer, H., editors, *Datenbanksysteme für Business, Technologie und Web (BTW 2019), 18. Fachtagung des GI-Fachbereichs „Datenbanken und Informationssysteme"(DBIS), 4.-8. März 2019, Rostock, Germany, Proceedings*, volume P-289 of *LNI*, pages 487–496. Gesellschaft für Informatik, Bonn.

[Sauer et al. 2015] Sauer, C., Graefe, G., and Härder, T. (2015). Single-pass restore after a media failure. In Seidl, T., Ritter, N., Schöning, H., Sattler, K., Härder, T., Friedrich, S., and Wingerath, W., editors, *Datenbanksysteme für Business, Technologie und Web (BTW), 16. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme"(DBIS), 4.-6.3.2015 in Hamburg, Germany. Proceedings*, volume P-241 of *LNI*, pages 217–236. GI.

[Sauer et al. 2017] Sauer, C., Graefe, G., and Härder, T. (2017). Instant restore after a media failure. In Kirikova, M., Nørvåg, K., and Papadopoulos, G. A., editors, *Advances in Databases and Information Systems - 21st European Conference, ADBIS 2017, Nicosia, Cyprus, September 24-27, 2017, Proceedings*, volume 10509 of *Lecture Notes in Computer Science*, pages 311–325. Springer.

[Sauer et al. 2018] Sauer, C., Graefe, G., and Härder, T. (2018). Fineline: log-structured transactional storage and recovery. *Proceedings of the VLDB Endowment*, 11(13):2249–2262.

[Schroeder and Gibson 2007] Schroeder, B. and Gibson, G. A. (2007). Understanding failures in petascale computers. In *Journal of Physics: Conference Series*, volume 78, page 012022. IOP Publishing.

[Schwalb et al. 2015] Schwalb, D., Berning, T., Faust, M., Dreseler, M., and Plattner, H. (2015). nvm malloc: Memory allocation for NVRAM. In Bordawekar, R., Lahiri, T., Gedik, B., and Lang, C. A., editors, *International Workshop on Accelerating Data Management Systems Using Modern Processor and Storage Architectures - ADMS 2015, Kohala Coast, Hawaii, USA, August 31, 2015*, pages 61–72.

[Shi et al. 2015] Shi, X., Chen, M., He, L., Xie, X., Lu, L., Jin, H., Chen, Y., and Wu, S. (2015). Mammoth: Gearing hadoop towards memory-intensive mapreduce applications. *IEEE Trans. Parallel Distributed Syst.*, 26(8):2300–2315.

[Sikka et al. 2012] Sikka, V., Färber, F., Lehner, W., Cha, S. K., Peh, T., and Bornhövd, C. (2012). Efficient transaction processing in SAP HANA database: the end of a column store myth. In Candan, K. S., Chen, Y., Snodgrass, R. T., Gravano, L., and Fuxman, A., editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 731–742. ACM.

[Silberschatz et al. 2020] Silberschatz, A., Korth, H. F., and Sudarshan, S. (2020). *Database System Concepts, Seventh Edition*. McGraw-Hill Book Company.

[SingleStore 2022] SingleStore (2022). Singlestoredb is the real-time distributed sql database, designed for data-intensive applications. Disponível em: "https://www.singlestore.com". Acessado em: 20/12/2022.

[Stonebraker and Weisberg 2013] Stonebraker, M. and Weisberg, A. (2013). The voltdb main memory DBMS. *IEEE Database Engineering Bulletin*, 36(2):21–27.

[Strickland et al. 1982] Strickland, J. P., Uhrowczik, P. P., and Watts, V. L. (1982). IMS/VS: an evolving system. *IBM Systems Journal*, 21(3):490–510.

[Taft et al. 2014] Taft, R., Mansour, E., Serafini, M., Duggan, J., Elmore, A. J., Aboulnaga, A., Pavlo, A., and Stonebraker, M. (2014). E-store: Fine-grained elastic partitioning for distributed transaction processing. *Proceedings of the VLDB Endowment*, 8(3):245–256.

[Tan et al. 2015] Tan, K., Cai, Q., Ooi, B. C., Wong, W., Yao, C., and Zhang, H. (2015). In-memory databases: Challenges and opportunities from software and hardware perspectives. *ACM Sigmod Record*, 44(2):35–40.

[Tandem Database Group 1987] Tandem Database Group (1987). Nonstop SQL: A distributed, high-performance, high-availability implementation of SQL. In Gawlick, D., Haynie, M. N., and Reuter, A., editors, *High Performance Transaction Systems, 2nd*

International Workshop, Asilomar Conference Center, Pacific Grove, California, USA, September 28-30, 1987, Proceedings, volume 359 of *Lecture Notes in Computer Science*, pages 60–104. Springer.

[Tang Yanjun and Luo Wen-hua 2010] Tang Yanjun and Luo Wen-hua (2010). A model of crash recovery in main memory database. In *2010 International Conference On Computer Design and Applications*, volume 5, pages V5–206–V5–207.

[Thomson et al. 2012] Thomson, A., Diamond, T., Weng, S., Ren, K., Shao, P., and Abadi, D. J. (2012). Calvin: fast distributed transactions for partitioned database systems. In Candan, K. S., Chen, Y., Snodgrass, R. T., Gravano, L., and Fuxman, A., editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 1–12. ACM.

[Tu et al. 2013] Tu, S., Zheng, W., Kohler, E., Liskov, B., and Madden, S. (2013). Speedy transactions in multicore in-memory databases. In Kaminsky, M. and Dahlin, M., editors, *ACM SIGOPS 24th Symposium on Operating Systems Principles, SOSP '13, Farmington, PA, USA, November 3-6, 2013*, pages 18–32. ACM.

[Tucker 2004] Tucker, A. B. (2004). *Computer science handbook*. CRC press.

[van Renen et al. 2018] van Renen, A., Leis, V., Kemper, A., Neumann, T., Hashida, T., Oe, K., Doi, Y., Harada, L., and Sato, M. (2018). Managing non-volatile memory in database systems. In Das, G., Jermaine, C. M., and Bernstein, P. A., editors, *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1541–1555. ACM.

[VolDB 2022] VolDB (2022). Voldb active data. Disponível em: "http://www.voltdb.com". Acessado em: 20/12/2022.

[VoltDB Documentation 2022] VoltDB Documentation (2022). Volt active data documentation. Disponível em: "https://docs.voltdb.com". Acessado em: 27/12/2022.

[Wang et al. 2014] Wang, Z., Qian, H., Li, J., and Chen, H. (2014). Using restricted transactional memory to build a scalable in-memory database. In Bulterman, D. C. A., Bos, H., Rowstron, A. I. T., and Druschel, P., editors, *Ninth Eurosys Conference 2014, EuroSys 2014, Amsterdam, The Netherlands, April 13-16, 2014*, pages 26:1–26:15. ACM.

[Weikum and Vossen 2002] Weikum, G. and Vossen, G. (2002). *Transactional Information Systems: Theory, Algorithms, and the Practice of Concurrency Control and Recovery*. Morgan Kaufmann.

[Willhalm et al. 2013] Willhalm, T., Oukid, I., Müller, I., and Faerber, F. (2013). Vectorizing database column scans with complex predicates. In Bordawekar, R., Lang, C. A., and Gedik, B., editors, *International Workshop on Accelerating Data Management Systems Using Modern Processor and Storage Architectures - ADMS 2013, Riva del Garda, Trento, Italy, August 26, 2013*, pages 1–12.

[Willhalm et al. 2009] Willhalm, T., Popovici, N., Boshmaf, Y., Plattner, H., Zeier, A., and Schaffner, J. (2009). Simd-scan: Ultra fast in-memory table scan using on-chip vector processing units. *Proceedings of the VLDB Endowment*, 2(1):385–394.

[Wu et al. 2017] Wu, Y., Guo, W., Chan, C., and Tan, K. (2017). Fast failure recovery for main-memory dbmss on multicores. In Salihoglu, S., Zhou, W., Chirkova, R., Yang, J., and Suciu, D., editors, *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 267–281. ACM.

[Yao et al. 2016] Yao, C., Agrawal, D., Chen, G., Ooi, B. C., and Wu, S. (2016). Adaptive logging: Optimizing logging and recovery costs in distributed in-memory databases. In Özcan, F., Koutrika, G., and Madden, S., editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 1119–1134. ACM.

[Yoo et al. 2009] Yoo, R. M., Romano, A., and Kozyrakis, C. (2009). Phoenix rebirth: Scalable mapreduce on a large-scale shared-memory system. In *Proceedings of the 2009 IEEE International Symposium on Workload Characterization, IISWC 2009, October 4-6, 2009, Austin, TX, USA*, pages 198–207. IEEE Computer Society.

[Zaharia et al. 2013] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., and Stoica, I. (2013). Discretized streams: fault-tolerant streaming computation at scale. In Kaminsky, M. and Dahlin, M., editors, *ACM SIGOPS 24th Symposium on Operating Systems Principles, SOSP '13, Farmington, PA, USA, November 3-6, 2013*, pages 423–438. ACM.

[Zhang et al. 2015a] Zhang, H., Chen, G., Ooi, B. C., Tan, K., and Zhang, M. (2015a). In-memory big data management and processing: A survey. *IEEE Transactions on Knowledge and Data*, 27(7):1920–1948.

[Zhang et al. 2015b] Zhang, H., Chen, G., Ooi, B. C., Wong, W., Wu, S., and Xia, Y. (2015b). "anti-caching-based elastic memory management for big data. In Gehrke, J., Lehner, W., Shim, K., Cha, S. K., and Lohman, G. M., editors, *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*, pages 1268–1279. IEEE Computer Society.

[Zhang et al. 2015c] Zhang, Y., Yang, J., Memaripour, A., and Swanson, S. (2015c). Mojim: A reliable and highly-available non-volatile memory system. In Özturk, Ö., Ebcioglu, K., and Dwarkadas, S., editors, *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS '15, Istanbul, Turkey, March 14-18, 2015*, pages 3–18. ACM.

[Zheng et al. 2014] Zheng, W., Tu, S., Kohler, E., and Liskov, B. (2014). Fast databases with fast durability and recovery through multicore parallelism. In Flinn, J. and Levy, H., editors, *11th USENIX Symposium on Operating Systems Design and Implementation, OSDI '14, Broomfield, CO, USA, October 6-8, 2014*, pages 465–477. USENIX Association.

Capítulo 5

Desagregando e Softwarizando as Redes de Celulares e o Programa OpenRAN Brasil

Daniel A. L. Marques, Fernando N. N. Farias, Fuad M. A. Junior, Daniel L. Feferman, Christian R. E. Rothenberg, Antônio J. G, Abelém, José F. Rezende

Abstract

Advances in network function virtualization associated with market and regulatory demand for more openness and interoperability on Radio Access Networks (RAN) for next-generation cellular networks motivated the introduction of the framework with standards, protocols, and open-source software components known as OpenRAN. This openness aims to democratize parts of the telecommunications network and thus not depend on large telecommunications equipment manufacturers, allowing costs and power reduction for large manufacturers. The OpenRAN architecture combines modular base station software with off-the-shelf hardware, assigning baseband and radio unit components from single vendors for seamless interoperability, whether there are virtualized/disaggregated RAN elements or not. In addition, the introduction of open interfaces between different OpenRAN components allows the gathering of information from the RAN and the update of control policies, which enables the utilization of Artificial Intelligence (AI) and Machine Learning (ML) techniques to perform optimizations and smart control of the RAN. Therefore, this minicourse has the main objective to present the concept and challenges which are leading academia and industry to invest in the concept of the open RAN. This short course will be essentially theoretical and will begin by introducing the historical factors of the evolution of the RANs, concepts of openness and softwarization, moreover, it will be addressed disaggregation, RAN intelligent controller (RIC), Core Networks, virtualization, open interfaces, and challenges. Next, the initiatives around the world that are collaborating in the advance, standardization and development of OpenRAN will be presented. After that, the OpenRAN Brazil Program are detailed, where it is presented the motivation, objectives, expected results, testbed, and applications. Finally, the final considerations and future trends regarding the research and development of OpenRANs and their components are presented.

Resumo

Avanços na função de virtualização de rede associados com o mercado e a demanda regulatória por mais abertura e interoperabilidade nas Redes de Acesso via Rádio (RAN) da próxima geração de redes celulares motivaram a criação de um framework, com padrões, protocolos e componentes de softwares de código aberto, denominado como OpenRAN. Essa abertura tem como objetivo democratizar segmentos da rede de telecomunicações para não depender de equipamentos de grandes indústrias, assim, permitindo a redução de custos e o poderio desses conglomerados. A arquitetura OpenRAN combina o software estação base modular com hardware pronto para uso, onde são atribuídos componentes de banda base e unidades de rádio de fornecedores únicos, assim viabilizando uma interoperabilidade transparente, não importando se os elementos de RAN são ou não virtualizados/desagregados. Além disso, a introdução de interfaces abertas entre os diferentes componentes OpenRAN permite que se obtenha informações da RAN e se efetue atualizações de políticas de controle, o que viabiliza a utilização de técnicas de Inteligência Artificial (IA) e Aprendizagem de Máquina (ML) para efetuar otimizações e controle inteligente da RAN. Logo, este minicurso tem como principal objetivo apresentar os conceito e desafios que estão levando a academia e indústria a investir no conceito de RAN aberta. O minicurso será essencialmente teórico e iniciará apresentando os fatores históricos da evolução das RANs e os conceitos de abertura e softwarização, além disso, serão abordados a desagregação, controle inteligente da RAN (RIC), Núcleo da Rede (Core Network), virtualização, interfaces abertas e desafios. Em seguida, as iniciativas ao redor do mundo que estão colaborando no avanço, padronização e desenvolvimento das OpenRANs serão apresentadas. Após isso, será detalhado o Programa OpenRAN Brasil, onde será discorrido a motivação, objetivos, resultados esperados, testbed e aplicações. Por fim, são apresentadas as considerações finais e tendências futuras em relação à pesquisa e desenvolvimento das OpenRANs e seus componentes.

5.1. Introdução

Nos últimos dez anos, houve um desenvolvimento significativo nas infraestruturas de rede, com uma forte tendência em direção ao software em ambiente de nuvem. Essa evolução trouxe consigo tanto benefícios quanto desafios. A softwarização das redes facilitou a programação dos elementos de rede e a virtualização de seus recursos, permitindo a alocação dinâmica e o particionamento da rede em fatias logicamente isoladas. Essas características impulsionaram o desenvolvimento de componentes de software, especialmente controladores e orquestradores, que possibilitam o gerenciamento programático do ciclo de vida das fatias de rede, bem como das aplicações e serviços associados a elas.

Essa softwarização foi impulsionada pelo surgimento do paradigma SDN (Software-Defined Networking ou Redes Definidas por Software). O conceito de SDN consiste na separação dos planos de controle e de dados, que antes eram implementados de forma monolítica e proprietária nos equipamentos de rede. Com essa separação, o plano de controle passa a ser implementado de forma centralizada e externa à rede, por meio de controladores SDN, enquanto o plano de dados se torna programável através de uma interface aberta fornecida pelos equipamentos e utilizada pelos controladores.

A exigência de programabilidade dos equipamentos imposta pelo paradigma SDN

impulsionou o desenvolvimento de diferentes interfaces de programação e controladores SDN. A inteligência da rede é centralizada no controlador SDN, que utiliza estatísticas de tráfego e informações de topologia constantemente coletadas para definir as regras de encaminhamento a serem utilizadas pelos equipamentos, de acordo com as políticas estabelecidas pelos diversos serviços de rede.

Recentemente, o conceito de SDN expandiu-se para além do domínio de rede e dados, sendo aplicado também aos domínios óptico e sem fio nas redes de comunicações das provedoras de serviços. Isso possibilita que um controlador SDN orquestre elementos da rede óptica, como transponders, comutadores ópticos, amplificadores, e elementos de redes sem fio (como as redes de acesso) .

Para que isso seja viável, os equipamentos devem ser programáveis, permitindo que suas configurações sejam alteradas dinamicamente por meio de determinadas interfaces. A padronização destas interfaces é o que permite a construção de redes totalmente abertas, onde é possível a participação de diferentes atores de mercado tanto no desenvolvimento dos equipamentos quanto no desenvolvimento de controladores inteligentes capazes de otimizar o funcionamento destas redes.

Desta forma, para minimizar os custos, os operadores de rede precisam atualizar a infraestrutura de comunicações móveis, acompanhando os novos casos de uso com as respectivas tecnologias e requisitos do mercado. Uma das formas de gerenciar e otimizar tais sistemas perpassa pela abertura das redes de acesso de rádio (do inglês, Radio Access Networks - RAN), que permitirá uma otimização e automação baseadas no acesso aos dados.

Originalmente os componentes de rede eram unidades monolíticas vistas como caixas pretas pelos operadores e fornecidas por um número limitado de fabricantes. Isso resultou em diversos problemas de configuração da RAN, tais como: os equipamentos não poderem ser ajustados para suportar diversas implementações e perfis de tráfego; a coordenação limitada entre nós de rede, o que impede a otimização conjunta e controle de componentes RAN; e a dependência em relação a poucos fornecedores, com opções restritas para as operadoras implantarem e fazerem interface com equipamentos RAN de vários fornecedores.

Nos últimos anos, as redes de acesso de rádio vêm passando por um processo de desagregação dos elementos da rede, buscando reduzir os custos de operação e manutenção. Nessa evolução, surgiu o conceito de CRAN (Centralized RAN ou Rede de Acesso de Rádio Centralizada), onde o processamento de sinal foi softwarizado, ou seja, as funcionalidades foram transformardas em software, e centralizado em servidores, dispostos em data centers. Esse servidores são capazes de processar o sinal proveniente de múltiplas estações rádio-base, denominadas de unidades de rádio remotas (RRUs).

A centralização proporcionada pelo conceito de CRAN resultou em uma grande redução de custos, devido aos ganhos na multiplexação estatística no uso dos recursos de processamento, o que leva a uma diminuição na capacidade necessária para atender a um certo número de unidades de rádio. A virtualização destes componentes de software (vRAN) permite um ganho ainda maior com relação à energia consumida por tais servidores, além de permitir uma maior flexibilidade na localização de tais servidores.

Posteriormente, o processamento realizado por esses servidores foi dividido em duas partes: uma que deve ficar mais próxima das estações-base e outra que pode ficar em locais mais distantes. Essa divisão ou desagregação da RAN resulta em novas reduções de custos, uma vez que diminui ainda mais a necessidade de equipamentos para atender um determinado número de RRUs.

A arquitetura OpenRAN segue esta abordagem de desagregação da RAN em dois componentes, denominados DU (Distributed Unit) e CU (Centralized Unit), e define interfaces abertas e padronizadas para o controle e gerenciamento destes componentes e também da RU (Radio Unit). Isto permite a construção de controladores SDN que fazem uso destas interfaces para otimizar e automatizar a RAN. Este controlador é chamado, na arquitetura da O-RAN Alliance, de RIC (Radio Intelligent Controller), o qual estabelece laços de controle da RAN em diferentes escalas de tempo.

A Aliança O-RAN é uma iniciativa colaborativa, criada em 2018, formada por mais de vinte empresas e organizações da indústria de telecomunicações de diversos países. Seu principal objetivo é remodelar a indústria relacionada às redes de acesso por rádio, promovendo de forma mais sólida conceitos como interoperabilidade, inteligência, virtualização, bem como a padronização e a adoção de soluções abertas. Essa abordagem inovadora busca separar as funções de hardware e software nas redes de acesso rádio, permitindo que os operadores de rede selecionem e combinem componentes de diferentes fornecedores, promovendo a concorrência e a flexibilidade. As especificações desenvolvidas pela O-RAN complementam os padrões estabelecidos pelo 3rd Generation Partnership Project (3GPP), uma colaboração entre organizações de telecomunicações que define os padrões para redes móveis de próxima geração. Essas especificações abrangem desde a desagregação e automação até a virtualização da RAN, assim a aliança busca criar um mercado mais competitivo, com um número crescente de participantes em ascensão.

A desagregação e softwarização da RAN, seguida pela definição de interfaces abertas e padronizadas entre os seus componentes, alavanca um enorme e vibrante ecossistema de inovação centrado nas redes de telecomunicações do futuro. Pela primeira vez, a academia pode participar ativamente no desenvolvimento e experimentação de novas funcionalidades nestas redes, envolvendo o uso de aprendizado de máquina para a completa automação e otimização destas redes.

Por outro lado, esta nova arquitetura da RAN possibilita a construção de redes de telecomunicações mais econômicas, quebrando a dependência de adquirir equipamentos de grandes fabricantes deste setor.

No entanto, o controle separado dos domínios de diferentes tecnologias não possibilita a exploração otimizada e totalmente automatizada dos recursos desses domínios na implementação de serviços fim a fim.

Diante desta perspectiva, este capítulo tem o objetivo de apresentar os principais conceitos, motivações e desafios que estão levando, tanto a indústria quanto a academia, a investir em pesquisa e desenvolvimento na abertura da Rede de Acesso de Rádio. Serão descritos os desafios e soluções que estão tratando da evolução do OpenRAN, tais como: interfaces, arquiteturas, protocolos, serviços e soluções, que servirão de suporte para consolidação do conceito. Além disso, será apresentada a iniciativa brasileira de pesquisa e

desenvolvimento em OpenRAN, denominada de “Programa OpenRAN Brasil”.

Além desta seção introdutória, este capítulo é composto por mais quatro seções. A Seção 5.2 detalhará a junção dos conceitos de abertura e softwarização das RANs, descrevendo a arquitetura OpenRAN, com seus principais componentes e interfaces, além dos seus princípios arquiteturais. Na Seção 5.3 serão discutidas as iniciativas OpenRAN pelo mundo, apresentando as principais plataformas open-source para a implementação dos diferentes componentes, e os principais consórcios que estão direcionando a padronização do OpenRAN, tanto no contexto de software quanto de hardware. A Seção 5.4 apresentará, em detalhes, o Programa OpenRAN@Brasil, que atualmente é o maior programa de pesquisa, desenvolvimento e inovação na área de redes aberta e desagregada para acesso via rádio no Brasil. Por fim, a Seção 5.5 apresentará as considerações finais, destacando as vantagens e oportunidades de se investir em pesquisa e desenvolvimento em OpenRAN, e discutirá qual será o futuro da abordagem no Brasil.

5.2. Abertura e a Softwarização na RAN: OpenRAN

A abertura e a softwarização na RAN (Rede de Acesso por Rádio) estão revolucionando a forma como as redes de comunicação são projetadas, implementadas e operadas. A tradicional arquitetura de RAN, que historicamente era baseada em sistemas proprietários e fechados, está sendo transformada por uma abordagem inovadora chamada OpenRAN.

OpenRAN refere-se a um ecossistema de tecnologias e padrões abertos que permitem a separação dos componentes da RAN, como hardware e software. Essa abordagem permite que diferentes fornecedores forneçam componentes interoperáveis, estimulando a concorrência e a escolha de soluções mais eficientes e econômicas.

Na Seção 5.2, serão apresentados a junção dos conceitos de abertura e softwarização às RANs, abordando a origem da ideia, a motivação e a necessidade por trás desses conceitos. Além disso, será apresentada a arquitetura OpenRAN, seus principais componentes e interfaces, bem como seus princípios arquiteturais. Neste tópico, serão abordados os seguintes temas: desagregação, controle inteligente da RAN (RIC), Núcleo da Rede (Core Network), virtualização, interfaces abertas e desafios associados a essa nova abordagem.

5.2.1. Padronização do 5G NR

Iniciado em 2016, a *Release 15* (Rel-15) dos padrões 3GPP introduziu oficialmente a tecnologia *5G New Radio (5G NR)* [Lin and et al. 2019]. Os objetivos da Rel-15 foram aprovados em março de 2017 e as especificações finalizadas em junho de 2018, com os primeiros terminais móveis compatíveis com 5G NR Rel-15 lançados em 2019. A Rel-15 aprimorou significativamente a comunicação móvel para cenários de banda larga móvel, chamados de *enhanced Mobile BroadBand* (eMBB), introduzindo taxas de dados mais rápidas, maior capacidade e menor latência. Para isso, na Rel-15 foi introduzida na Rel-15 uma nova tecnologia de rede de acesso (RAN), denominada 5G New Radio (5G NR), e também um novo núcleo de rede conhecido como 5G Core (5GC). Além disso, novas faixas do espectro em ondas milimétricas (e.g. mmWave) e maior suporte às funcionalidades, como formação de feixes (e.g. beamforming) e MIMO (do inglês multiple input, multiple output) massivo, foram suportadas para atender à maior demanda

por largura de banda das redes móveis. Por fim, a Rel-15 aprimorou novas funcionalidades de Internet das Coisas (IoT), suportando tecnologias de baixo consumo de energia de área ampla (LWPAN) e segurança melhorada.

O Release 16 (Rel-16) do 3GPP começou a ser desenvolvido após a conclusão do Rel-15, expandindo ainda mais as capacidades do 5G com novos recursos e aprimoramentos visando principalmente a garantia de evolução e suporte maior à aplicações industriais [Parkvall et al. 2020, Flynn 2020]. A Rel-16 foi concluída em julho de 2020, e teve os primeiros terminais móveis compatíveis lançados em 2021. Com a Rel-16, houve a introdução da comunicação ultra-confiável e de baixa latência avançada (eURLLC), que possibilita a execução de aplicações que demandem baixa latência extrema, como aplicações industriais e cirurgia remota, além de viabilizar a implementação de redes sensíveis ao tempo (TSN) que necessitam de comunicação em tempo real. Relacionado a isso, a Rel-16 introduziu melhorias na massificação de dispositivos do tipo *Internet of Things (IoT)*, muito usados nas tecnologias de Narrowband IoT (NB-IoT) e no Enhanced Machine-Type Communication (eMTC) para oferecer maior capacidade, melhor cobertura e menor consumo de energia, suportando assim um grande número de dispositivos IoT em uma ampla gama de aplicações como monitoramento ambiental, agricultura e rastreamento de ativos.

A Rel-16 melhorou ainda o suporte a redes não-públicas (NPN) e redes privadas, para atender às necessidades de segurança, desempenho e controle das aplicações de missão crítica e IoT massivo, incluindo funcionalidades como o fatiamento de recursos de rede (network slicing), que permite que operadoras criem redes virtuais otimizadas para aplicações ou serviços específicos. A Rel-16 introduziu o uso integrado para acesso e redes de transporte, também chamadas de *backhaul* (IAB), viabilizando operadoras usarem uma combinação de redes cabeadas e sem fio para melhorar o desempenho de suas redes, bem como funcionalidades para economia de energia que ampliam a vida útil da bateria dos terminais. A Rel-16 também introduziu a comunicação celular 5G de veículos para quaisquer outras coisas (C-V2X), pela qual os veículos se comunicam entre si e com a infraestrutura da estrada, aumentando a segurança e a eficiência. Por fim, a Rel-16 também aprimorou o suporte para multicast e broadcast, permitindo a entrega eficiente de conteúdo em tempo real, como vídeo e áudio, para um grande número de usuários simultaneamente.

O Release 17 (Rel-17) iniciou após a conclusão do Rel-16, e ampliou ainda mais o ecossistema 5G NR com novos recursos, melhorias de desempenho, cobertura e capacidade, suportando também novos casos de uso como realidade aumentada (e.g. extended reality - XR), realidade virtual (e.g. virtual reality - VR), carros autônomos e aplicações industriais [Samdanis and Taleb 2020, Rahman et al. 2021]. A Rel-17 foi concluída em Março de 2022, e os primeiros terminais suportando funcionalidades da Rel-17 são esperados para 2023. A Rel-17 introduziu novos recursos para suportar serviços de missão crítica e IoT massivo. Também melhorou o suporte para redes privadas e redes de acesso sem fio fixas (FWA – Fixed Wireless Access). Além disso, introduziu novos recursos para melhorar o desempenho da rede em áreas rurais. Por fim, trouxe melhorias de segurança para operação integrada com redes 4G, funcionalidades de segurança para serviços baseados em proximidade (e.g. Proximity Services - ProSe) e melhorias na segurança para proteção contra estações rádio-base falsas.

Recentemente, a Release 18 (Rel-18) iniciou com a conclusão da Rel-17 em Março de 2022, e marcou o início do desenvolvimento do chamado 5G Advanced [Lin 2022, Zhang and Long 2022, Chen et al. 2022, Chen et al. 2023]. Embora a data de congelamento da Rel-18 tenha sido deslocada recentemente para Março de 2024, hoje já se tem uma boa visão de quais funcionalidades serão contempladas. Por exemplo, se antevê a introdução de novos recursos para melhorar o desempenho da rede 5G NR em áreas urbanas densas, tanto em termos de taxas de pico de vazão como em termos de taxas médias de vazão, e também a diminuição da latência média. Além disso, serão adicionados novos recursos para suportar serviços de missão crítica e IoT massivo. Por fim, a Rel-18 introduz melhorias no suporte para redes privadas e redes de acesso sem fio fixas (Fixed Wireless Access - FWA), e introduz novos recursos para melhorar a eficiência energética da rede. A Release 18 também marca o início de muitos itens de estudo (Study Items - SIs) para a evolução do 5G Advanced, que se tornarão itens de trabalho (Working Items - WIs) em releases futuras.

5.2.2. Desagregação da RAN dos padrões 5G NR

O advento das redes 5G NR levou à uma mudança de paradigma na RAN quando se compara com as redes 4G, com a desagregação da estação rádio-base 5G (conhecida como gNodeB, ou simplesmente gNB) tendo um papel crucial para se atingir os objetivos de melhorias de desempenho, flexibilidade e escalabilidade. Além disso, o movimento pela abertura e desagregação das redes de acesso conhecido como Open RAN não só aprofundou a desagregação na própria RAN, como introduziu novos componentes que permitem que se extraia informações da RAN para que algoritmos externos à gNB possam implementar decisões para gerenciamento de recursos de rádio (Radio Resource Management - RRM) que se traduzem em políticas de RRM que são devolvidas à gNBs, com o uso massivo de inteligência artificial e aprendizado de máquina e também visando a diversidade de fornecedores e diminuição de custos de capital e custos operacionais. A Figura 5.1 apresenta os componentes de uma RAN aberta desagregada e suas interfaces.

A desagregação do 5G NR RAN visa otimizar o desempenho da rede e a utilização de recursos ao desacoplar suas funções principais em componentes distintos. A desagregação vertical, que aborda principalmente a separação de funcionalidades dentro da RAN, envolve a divisão da gNB em dois componentes essenciais: a unidade centralizada (Central Unit - CU) e a unidade distribuída (Distributed Unit - DU), no que é conhecido como Split 2. A CU é responsável por lidar com as funções da camadas superiores, como gerenciamento de mobilidade e controle de recursos de rádio (Radio Resource Control - RRC), enquanto a DU gerencia as funções da camadas inferiores, incluindo processamento da camada de controle de acesso (Medium Access Control - MAC), camada física (Physical interface - PHY) e a própria transmissão de sinal de radiofrequência (Radio Frequency - RF). Essa divisão de trabalho aumenta a flexibilidade e a escalabilidade, permitindo que as operadoras de rede atualizem e dimensionem independentemente cada componente com base nas necessidades específicas de diferentes áreas dentro da cobertura da rede. A interface que conecta a CU com a DU se chama F1, e será explorada em maiores detalhes na próxima seção.

A desagregação horizontal, por outro lado, concentra-se na separação de funcionalidades dentro da própria CU. Isso implica dividir o CU em dois sub-componentes

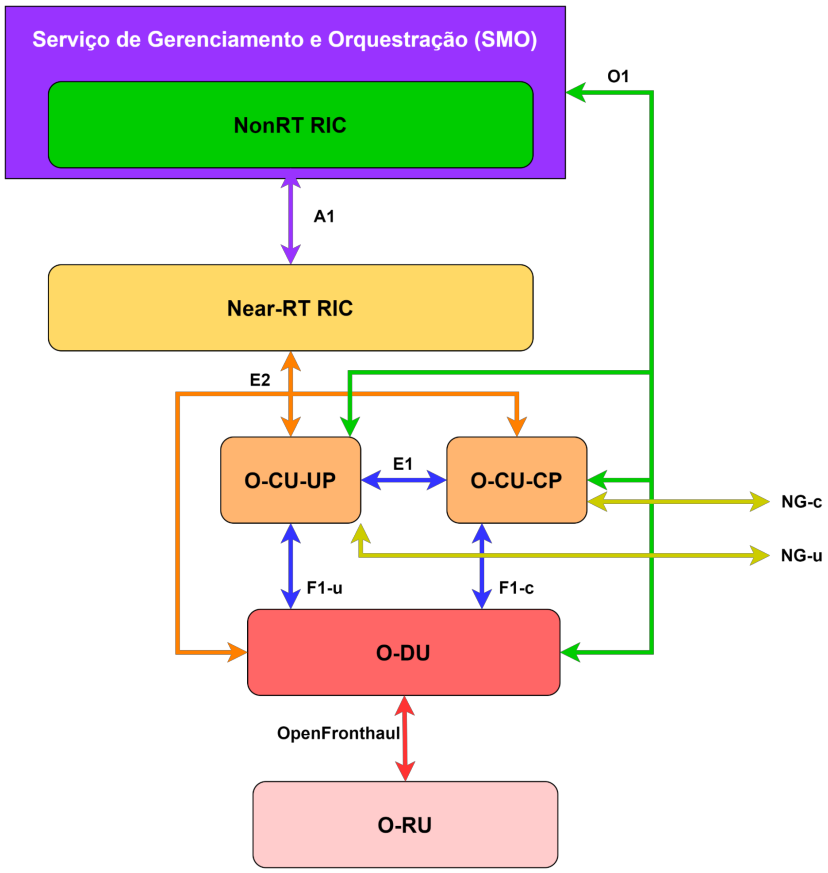


Figura 5.1: Arquitetura Open RAN

distintos: o plano de usuário da CU (CU-UP) e o plano de controle da CU (CU-CP). O CU-UP é responsável por gerenciar o tráfego do usuário, como dados e voz, enquanto o CU-CP lida com sinalização de controle e tarefas de gerenciamento de rede. Dessa forma, ao isolar as funções do plano do usuário e do plano de controle, a desagregação horizontal facilita a alocação de recursos aprimorada, gerenciamento de rede simplificado e otimização mais granular do desempenho da rede. A combinação de abordagens de desagregação vertical e horizontal em RANs 5G NR contribui para aumentar a adaptabilidade, melhorar a eficiência da rede e reduzir os custos operacionais. A interface que conecta a CU-UP ao CU-CP se chama E1, e será melhor explorada na próxima seção.

Avanços na função de virtualização de rede associadas com o mercado e a demanda regulatória por mais abertura e interoperabilidade nas RAN 5G NR motivaram a criação de um arcabouço composto por padrões, protocolos e componentes de softwares de código aberto denominado como Open RAN. Essa abertura tem como objetivo democratizar segmentos da rede de telecomunicações para não depender de equipamentos de grandes indústrias, e assim permitir a redução de custos e o poderio desses conglomerados. A arquitetura Open RAN combina o software da gNB modular com hardware dito "pronto para uso", onde são atribuídos componentes de banda-base e unidades de rádio de fornecedores distintos, assim viabilizando uma interoperabilidade transparente, não importando se os elementos de RAN são ou não virtualizados e/ou desagregados. Além disso, a introdução de interfaces abertas entre os diferentes componentes Open RAN permite que se obtenha informações da RAN e se efetue atualizações de políticas de controle, o que viabiliza a utilização de Aprendizagem de Máquina e entre outras técnicas de Inteligência Artificial para efetuar otimizações e controle inteligente da RAN.

Uma primeira desagregação introduzida pelo padrão Open RAN foi a desagregação vertical da DU em dois outros componentes, a saber a unidade distribuída aberta (O-DU) e a unidade de rádio aberta (O-RU). Essa abordagem inovadora, conhecida também como Split 7.2x, visa aumentar a interoperabilidade e a flexibilidade dentro da rede ao desacoplar elementos de hardware e software, permitindo o uso de componentes de vários fornecedores via interfaces abertas. O O-DU é responsável por tarefas de processamento em tempo real, como processamento de banda-base, enquanto o O-RU lida com as funções na linha de frente da interface de radiofrequência (RF), incluindo conversão de sinal, amplificação e transmissão pela interface aérea. Ao segregar essas funcionalidades, o padrão Open RAN promove uma arquitetura RAN mais modular e adaptável, permitindo que as operadoras atualizem, dimensionem e personalizem cada subcomponente de forma independente para atender a requisitos de rede específicos e otimizar o desempenho. A adoção do padrão O-RAN (Open RAN), portanto, promove um ecossistema mais competitivo, impulsiona a inovação e, por fim, leva a implantações de redes mais econômicas e eficientes. A interface que conecta a O-DU com a O-RU implementa uma interface conhecida como OpenFronthaul, e será melhor explorada na seção seguinte.

A arquitetura Open RAN (O-RAN) também apresenta o conceito do controlador inteligente da RAN (RAN Intelligent Controller - RIC) como um componente chave desagregado, projetado para aprimorar a inteligência e a adaptabilidade da RAN. O RIC é uma plataforma baseada em software que permite a introdução de funcionalidades avançadas como gerenciamento de recursos de rádio em tempo real, otimização de rede e aplicação de políticas de gerenciamento de recursos de rádio. Ao empregar algoritmos

de aprendizado de máquina e inteligência artificial, o RIC pode realizar tomadas de decisão dinâmicas e inteligentes, otimizando o desempenho da rede com base em padrões de tráfego, comportamento do usuário e outras informações contextuais. Essa camada adicional de inteligência permite que as operadoras de rede forneçam serviços mais confiáveis, consistentes e eficientes, melhorando a qualidade geral da experiência dos usuários finais. O RIC se conecta aos outros elementos da arquitetura Open RAN por meio das interfaces E2, A1 e O1 que serão melhor exploradas na próxima seção.

A arquitetura O-RAN também abrange a plataforma de orquestração e gerenciamento de serviços (Service Management and Orchestration - SMO). O SMO é responsável por coordenar e gerenciar o ciclo de vida de ponta a ponta dos serviços RAN, facilitando a coordenação entre os elementos da rede, maior eficiência da rede e uso mais eficaz dos recursos. Ao incorporar o SMO na arquitetura O-RAN desagregada, as operadoras podem se beneficiar de uma infraestrutura de rede ágil, automatizada e inteligente que pode se adaptar rapidamente às demandas em constante mudança e oferecer desempenho aprimorado em uma ampla variedade de casos de uso. O SMO se conecta à todos os componentes da gNB Open RAN desagregada (e.g. O-CU, O-DU e O-RU) por meio da interface O1, explorada em maiores detalhes na próxima seção.

5.2.3. Controlador Inteligente de Rádio (RAN Intelligent Controller - RIC)

Na RAN monolítica com fabricantes tradicionais (Huawei, Nokia, etc), a otimização de rede normalmente consiste em coletar as métricas de desempenho por períodos estendidos de tempo (e.g. dia ou semana), selecionar algumas células com piores indicadores e realizar ajustes físicos ou lógicos buscando melhorar os resultados dessas métricas. A otimização também pode ser provocada por reclamações dos clientes (e.g. falta de sinal ou má qualidade de serviço), onde é comum o engenheiro responsável não receber todas as informações necessárias para identificar e endereçar o problema. Os ajustes na maioria das vezes são feitos manualmente, alterando a cobertura das antenas através de modificações físicas ou os parâmetros de configuração da estação rádio base ou células envolvidas (e.g. potência de irradiação, agregação de canais, etc). As alterações visam melhorar a experiência da maioria dos usuários na região e normalmente são estáticos (não alteram até a próxima intervenção). Com o tempo foram desenvolvidas algumas implementações mais sofisticadas e tecnologias de automação conhecidas como SON (Self-Organizing Network) que permitem alterar alguns parâmetros de configuração da rede dinamicamente, inclusive considerando determinados indicadores de desempenho. Tais funcionalidades são projetadas individualmente para cada fabricante (frequentemente pelos próprios fabricantes), possuem código fechado e normalmente entram em ação depois que ocorre algum problema de desempenho na rede.

Sendo assim, o RIC atua como próximo passo na evolução das tecnologias de otimização das redes móveis, buscando complementar as soluções já existentes, trazendo mais eficiência e automação. Embora opcional, o RIC é considerado o elemento central da arquitetura Open RAN e atualmente é o componente onde reside a inteligência das redes da nova geração.

Existem dois tipos de controladores RIC com características e propósitos distintos, onde a principal diferença é o tempo de atuação. Como os próprios nomes sugerem,

o Near-RT RIC é projetado para executar ações em tempo quase real e o Non-RT RIC age mais lentamente, podendo realizar os controles com intervalos de alguns minutos ou horas. Dessa forma, estando localizados em partes diferentes da arquitetura Open RAN, os dois se complementam, agindo em conjunto para realizar as tarefas de RRM (Radio Resource Management).

Near-RT RIC: de acordo com definição oficial, é uma função lógica que permite controle e otimização em tempo quase real das funções e recursos dos nós E2 por meio de coleta de dados de baixa granularidade e ações na interface E2 com loops de controle na ordem de 10 ms - 1 segundo [ORAN 2023f].

Para atuar em tempo quase real, o Near-RT RIC precisa estar mais próximo à estação rádio base (DU - Distributed Unit e à CU - Control Unit). Tipicamente está localizado na borda ou em nuvem regional de telecomunicações. Através da interface E2, ele interage com DUs e CUs na RAN, bem como com gNBs (5G) e eNBs (4G) compatíveis com O-RAN, operando as malhas de controle de recursos de rádio. O Near-RT RIC consiste em vários aplicativos com lógica personalizada chamados xApps, e serviços que são necessários para suportar a execução dos mesmos (como banco de dados, estrutura para transporte das mensagens entre componentes, gerenciadores de aplicativos e inscrições, etc). Uma xApp é uma aplicação de software que pode ser desenvolvida por terceiros e que implementa uma funcionalidade específica dentro do near-RT RIC (como o controle de recursos rádio). Há um esforço crescente para construir uma loja desses aplicativos para operadoras de redes, que tem o potencial de estimular a inovação ao trazer novas perspectivas, soluções personalizadas e uma competição saudável entre os desenvolvedores.

Non-RT RIC: é um componente do SMO (Service Management and Orchestration) e complementa o Near-RT RIC para operação e otimização de RAN. Tipicamente instalado num ponto mais central, ele pode trabalhar com vários Near-RT RICs localizados nas bordas, tendo uma visão mais ampla da rede. De acordo com a definição oficial, o principal objetivo do Non-RT RIC é oferecer suporte à otimização inteligente da RAN, fornecendo orientação baseada em políticas, gerenciamento de modelos ML e informações de enriquecimento para a função Near-RT RIC, para que a RAN possa otimizar, por exemplo, o RRM sob certas condições. Ele também pode executar a função de gerenciamento de recursos de rádio inteligente em intervalo de tempo não real (maior que 1 segundo). De forma análoga, a plataforma Non-RT RIC também possui alguns serviços embarcados para o gerenciamento de seu funcionamento (como banco de dados, interfaces de comunicação, entre outros), além das suas próprias aplicações de otimização e fornecimento de políticas, as quais são nesse caso denominadas rApps. Devido ao seu ciclo de operação poder ser muito mais extenso (podendo realizar análises na escala de horas ou dias, inclusive), esse elemento da arquitetura também pode executar o treinamento e reconfiguração dos parâmetros dos modelos de otimização empregados, sejam nas rApps ou nas xApps [ORAN 2022]. Esses dois elementos se comunicam através de uma interface específica, chamada A1. Por ela, as políticas de execução de otimizações de quase tempo real são transmitidas, além de modelos treinados previamente [ORAN 2023a].

5.2.4. O ciclo de vida e operação de um modelo de IA/ML no RIC

A pesquisa e experimentação de ferramentas de otimização baseadas em aprendizado de máquina (ML) aplicadas em contextos de telecomunicações estão cada vez mais intensas. Até agora, as redes celulares implantadas têm utilizado técnicas de otimização rudimentares, como mencionado anteriormente. No entanto, existe um grande potencial para o uso de técnicas mais sofisticadas, considerando que o poder de processamento para esses algoritmos está se tornando menos limitante, graças ao uso de servidores mais robustos e arquiteturas mais elaboradas, incluindo nuvens e implementações nas bordas de rede para redução da latência da comunicação.

Nesse contexto, a plataforma RIC (Inteligência de Controle da RAN) possibilita a aplicação desses modelos diretamente nos elementos da RAN, utilizando dados coletados por meio da interface E2. Isso permite que um ciclo completo de vida de um modelo de ML empregado no RIC seja realizado, abrangendo todos os passos envolvidos no processo, desde os primeiros estudos até a operação.

Ao compreender o ciclo de vida de um modelo de ML no RIC, é possível explorar todo o potencial dessa abordagem avançada de otimização. Os estudos iniciais envolvem a seleção e preparação dos dados relevantes, a definição do objetivo da otimização e a escolha do algoritmo de ML adequado. Em seguida, são realizadas etapas de treinamento e validação do modelo, utilizando conjuntos de dados históricos e métricas de desempenho.

Após o treinamento, o modelo é implantado no ambiente de produção e começa a receber dados em tempo real da interface E2, permitindo que ele faça previsões e tome decisões de otimização. Esse processo é contínuo, com o monitoramento contínuo do desempenho do modelo e possíveis atualizações para melhorar sua precisão e eficácia.

Portanto, entender o ciclo de vida de um modelo de ML no contexto do RIC é fundamental para aproveitar todo o potencial dessa tecnologia, permitindo uma otimização mais sofisticada e eficiente das redes de telecomunicações.

A representação do ciclo de vida mencionado é ilustrada na Figura 5.2. Inicialmente, o modelo é testado em um ambiente experimental ou simulado, utilizando dados previamente obtidos e ferramentas de design típicas, como as bibliotecas Keras, PyTorch, Tensorflow, entre outras. O modelo pode ser treinado e avaliado em um ambiente chamado Training Host, que pode estar localizado dentro do Non-RT RIC ou ser tratado de forma offline. O Training Host solicita dados específicos da RAN e/ou do Near-RT RIC para o treinamento do modelo. Nesse estágio, o modelo pode passar por modificações adicionais, visando melhorias e conformidade, até estar pronto para implantação.

Após o treinamento, o modelo é disponibilizado no catálogo de modelos de otimização do SMO (Service Management and Orchestration) e, em seguida, publicado para o Non-RT RIC, juntamente com as licenças de uso adequadas e metadados associados. O Non-RT RIC é responsável por empacotar o modelo em um contêiner, de acordo com o gerenciador de contêineres utilizado na rede, e definir as políticas para sua operação. Em seguida, o Non-RT RIC implanta o aplicativo já conteinerizado em seu local de atuação, por exemplo, no Near-RT RIC. Nesse ponto, o modelo está pronto para operar, recebendo os dados da RAN por meio da interface E2 e executando otimizações propostas em tempo quase real, interferindo na rede. Assim, novos dados podem ser coletados e enviados para

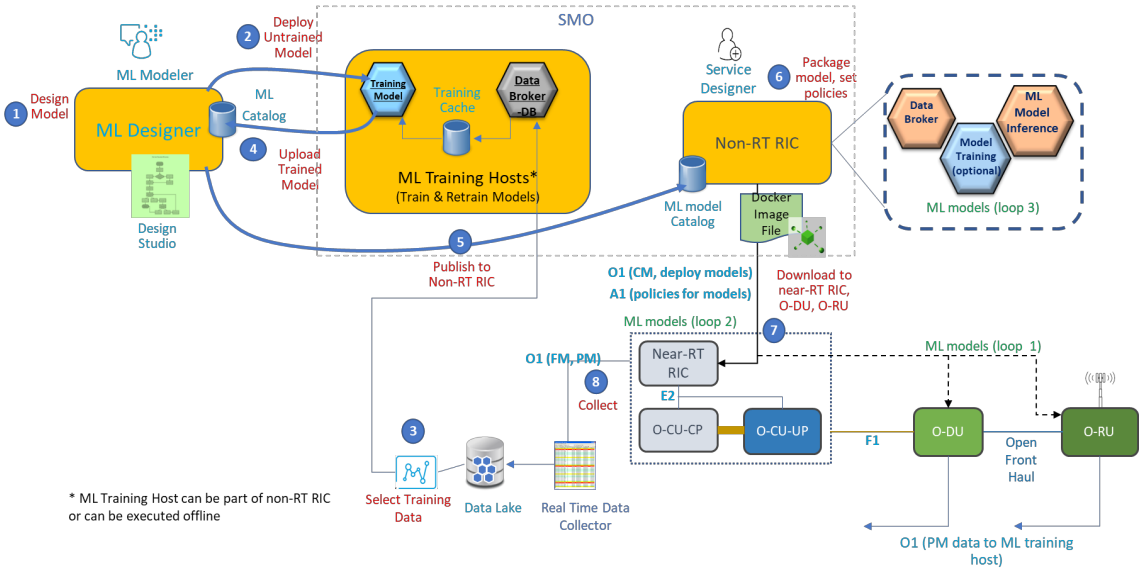


Figura 5.2: Ciclo de Vida do Modelo de Aprendizado de Máquina no contexto do RIC. [ORAN 2023d]

camadas superiores para avaliação e retreinamento, quando necessário [ORAN 2023d].

5.2.5. As técnicas de IA/ML e casos de uso

As métricas observáveis da rede são, em geral, Key Performance indicators (KPIs) relacionados ao desempenho dos elementos da rede (as células, sua ocupação, qualidade de serviço média, latências, parâmetros relacionados ao uso massivo de antenas, potência de sinais, entre diversas outras). Dessa forma, são esperados dados numéricos, séries temporais ou em formato de tabela. Alguns outros poderão ser categóricos, indicando condições atingidas, disponibilidade de certo recurso, determinação de prioridades em tarefas (roteamento, por exemplo), alocação de usuário em feixes, entre outros. De toda forma, são pouco esperados dados não estruturados, imagens, fragmentos de textos (strings), etc.

Além disso, em relação aos tipos de tarefas que terão maior ocorrência, são esperadas as denominadas tarefas de aprendizado supervisionado, considerando a natureza dos dados e as otimizações clássicas da rede. Ou seja, são tarefas que possuem um rótulo para entradas e saídas, como a classificação e a regressão. Ademais, as técnicas de aprendizado profundo (Deep Learning) mais características no contexto de redes móveis desagregadas são as redes neurais recorrentes (que lidam bem com problemas de análise temporal), o aprendizado por reforço (com contínua interação com o ambiente) e o aprendizado federado (pelas características dessa estratégia, ao não compartilhar todos os dados dos usuários com a rede, por alguma questão de privacidade de dados, por exemplo).

Alguns casos de uso clássicos em redes móveis podem ser endereçados com certas técnicas citadas. A previsão do uso de carga de uma célula ou um cluster de células, a estimação de condições de canal ou a otimização na transição da conexão em uma certa localidade. Porém, novos casos de uso também são previstos em especificações técnicas, mais complexas, porém facilitadas pela arquitetura Open RAN, como o compartilhamento de RAN entre diferentes operadoras, otimização para ambiente IoT industrial, previsão de

ataques orquestrados à rede, entre outros [ORAN 2023c].

5.2.6. Interfaces

Diversas interfaces abertas promovem a comunicação entre os elementos da arquitetura Open RAN. Elas são padronizadas pelos grupos de trabalho da O-RAN Alliance e, de forma objetiva, são tratadas a seguir:

- **A1**: Conecta os dois RICs, habilitando o Non-RT RIC para fornecer controles de ação para o Near-RT RIC que pode afetar um UE ou um grupo deles. Isso ajuda a estabelecer as políticas de otimização de alto nível e gerenciamento de modelos de ML em xApps [ORAN 2023a].
- **O1**: É uma ligação direta entre os elementos da arquitetura, como entre o Near-RT RIC e RAN e entre o SMO e o Non-RT RIC. Ele oferece suporte a serviços de gerenciamento, que incluem o gerenciamento do ciclo de vida dos componentes O-RAN, desde a inicialização e configuração até a tolerância a falhas, garantias de desempenho, rastreamento, coleta de KPIs e gerenciamento de software e arquivos [ORAN 2023b].
- **E2**: Conecta o Near-RT RIC aos nós E2 (estações base compatíveis com DU, CU e O-RAN). O Near-RT RIC usa essa interface para controlar procedimentos e funções dos nós E2. Além disso, a interface E2 garante a coleta e transmissão de métricas RAN periódicas ou acionadas por eventos para o Near-RT RIC [ORAN 2023e].
- **F1**: É responsável por conectar os elementos CU e DU da RAN. Por definição, é dividida entre o plano de controle e o plano de dados, de forma a separar esse tipo de tráfego adequadamente. Originalmente foi definida tecnicamente pelo 3GPP, mas segue complementada pelas especificações da O-RAN Alliance (atribuída ao grupo WG5, conforme tratado em seção específica) [ORAN 2023b].
- **E1**: É responsável por transmitir os sinais de rádio entre a DU e o RU, permitindo a comunicação e o controle da estação base. Essa interface inclui funções como transmissão de dados, controle de potência, modulação e demodulação do sinal de rádio, além de outros recursos relacionados à camada física da rede [ORAN 2023b].

5.2.7. FrontHaul O-RAN

A indústria de telecomunicações tem enfrentado sérios desafios para acomodar com eficiência o crescimento exponencial do tráfego de redes. Novos casos de uso e recursos escassos de espectro colocaram uma enorme pressão sobre os provedores de serviços de comunicação para fazer o uso mais eficiente de seu espectro de rádio alocado. Para atender às crescentes demandas, as operadoras estão buscando novas maneiras de projetar redes flexíveis, dimensionar dinamicamente a capacidade da rede, expandir a cobertura de serviços, implantar rapidamente novos serviços e melhorar a experiência geral do usuário, reduzindo o custo total de implantação e o tempo de comercialização.

Como já foi dito, a O-RAN Alliance está padronizando o próximo nível de especificações das Redes de Acesso por Rádio (RAN – Radio Access Networks). A Open

RAN é uma colaboração de fabricantes de equipamentos e empresas de telecomunicações em vários grupos de trabalho para resolver esse problema de interoperabilidade por meio da criação de padrões. Desde que o equipamento atenda aos padrões RAN abertos, o mesmo deve ser compatível com equipamentos fabricados por qualquer outro fornecedor cujo equipamento também atenda aos padrões especificados pela O-RAN Alliance.

Para tanto, a interface Fronthaul é a chave para a implantação flexível de uma rede Open RAN, pois antecipa os diversos cenários sobre como uma estação base irá interagir com o rádio, quais serão as demandas subjacentes das redes 5G, além de possíveis problemas de sincronização. O Fronthaul O-RAN é uma arquitetura que divide o pipeline de processamento do modem sem fio 5G em estágios padronizados, separados e conectados por determinadas interfaces de rede padrão, podendo conectar várias Unidades de Rádio (RU – Radio Unit) a uma única Unidade Distribuída (DU – Distributed Unit), de modo que haja uma bifurcação em várias interfaces aéreas 5G.

As configurações da arquitetura Fronthaul tem a capacidade de equilibrar as demandas de confiabilidade, taxa de transferência e latência de aplicativos avançados em redes 5G. Dependendo do tipo de tratamento de dados necessário, a arquitetura em camadas Fronthaul pode ser distribuída pela rede entre datacenters de ponta e datacenters centrais. Essa estrutura permitirá que operadoras sem fio e operadoras móveis atualizem e aprimorem suas redes sem fio sem estarem vinculadas a equipamentos legados e acelerar a implantação de Fronthaul 5G em larga escala de maneira escalável.

O protocolo mais comumente usado para o Fronthaul é o Common Public Radio Interface (CPRI), mas isso tem algumas limitações e desafios. O CPRI foi projetado para que a Unidade de Rádio Remoto (RRU – Remote Radio Unit) e a Unidade de Estação Base (BaseBand Unit – BBU) sejam co-localizadas, ou seja, há um limite de distância física que pode existir entre elas. Com a RAN Centralizada (C-RAN) se tornando cada vez mais popular, essa limitação de distância física pode se tornar um desafio para locais fora de áreas construídas. Como o CPRI é especificado para topologia ponto a ponto, esse protocolo torna-se uma barreira para implantações de vários fornecedores. O CPRI também pode suportar apenas taxas de dados Fronthaul limitadas, tornando-o inadequado em cenários de alta taxa de dados para redes 5G. O projeto enhanced CPRI (eCPRI) foi desenvolvido para superar essas e outras limitações.

O eCPRI substitui a transferência síncrona de dados no CPRI por um protocolo baseado em Ethernet, removendo a dependência específica do fornecedor. O protocolo eCPRI suporta comunicação ponto-a-multiponto e multiponto-a-multiponto, além de fornecer uma topologia ponto-a-ponto, permitindo que as operadoras misturem e combinem vários fornecedores. O eCPRI possui três planos de operações como parte do protocolo: planos de usuário, sincronização, controle e gerenciamento. As definições de transporte específicas do plano de usuário são padronizadas pelo protocolo eCPRI em relação aos formatos do quadro de dados, pacote e cabeçalho; enquanto os planos de sincronização, controle e gerenciamento não são explicitamente limitados pelo protocolo eCPRI.

A interface Fronthaul O-RAN inclui o Plano de Controle (C-Plane – Control Plane), Plano de Usuário (U-Plane – User Plane), Plano de Sincronização (S-Plane – Synchronization Plane) e Plano de Gerenciamento (M-Plane – Management Plane) sobre as interfaces Lower-Layer Split (LLS). A O-RAN define esses planos de operação da

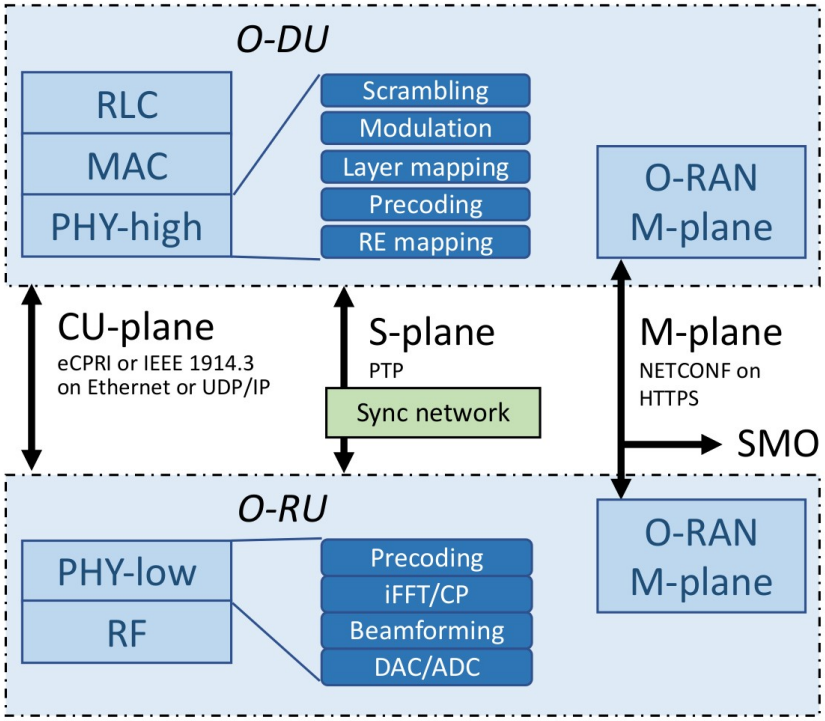


Figura 5.3: Fronthaul O-RAN, planos de operações e protocolos. [Polese et al. 2023]

seguinte forma:

- O plano de controle é responsável por definir o agendamento, a coordenação necessária para a transferência de dados e Beamforming, por exemplo;
- O plano de usuário é responsável pela transferência eficiente de dados dentro dos limites de tempo definidos de acordo com as numerologias NR;
- O plano de sincronização é responsável pelos aspectos de temporização e sincronização entre a O-DU e a O-RU;
- O plano de gerenciamento é responsável por gerenciar a unidade de rádio. O plano de gerenciamento fornece uma variedade de funções de gerenciamento para O-RU com o objetivo de definir parâmetros no lado da O-RU conforme exigido pelos planos de controle, usuário e sincronização, como por exemplo: gerenciar o software da O-RU.

Os protocolos de especificações da interface Fronthaul O-RAN para cada plano mencionado acima é mostrado na Figura 5.3:

Os planos de controle e usuário suportam uma pilha de protocolos que transmitem dados através dos protocolos eCPRI e/ou RoE. Vale ressaltar que, os protocolos User Datagram Protocol (UDP) e Internet Protocol (IP) podem ser utilizados de forma opcional para a transmissão de mensagens dos planos de controle e usuário. Em relação ao plano de sincronização, as especificações para a interface Fronthaul O-RAN definem

os protocolos PTP (Precision Time Protocol) e SyncE como responsáveis pela obtenção da sincronização precisa entre O-RU e O-DU, para tanto, utiliza-se o relógio disponível (como referência de tempo para sincronização) localizado na O-DU. Por fim, o plano de gerenciamento utiliza protocolo NETCONF/YANG e Ethernet para a transmissão de mensagens de gerenciamento de rede.

Embora o desenvolvimento dos padrões Open RAN venha permitindo instalações de redes celulares móveis tanto a nível de pesquisa acadêmica quanto a nível de instalações comerciais, diversos desafios em termos de pesquisa e desenvolvimento se apresentam.

Um dos primeiros desafios encontra-se no mapeamento dos principais casos de uso para instalações de redes Open RAN. Embora os padrões Open RAN forneçam as diretrizes para a criação de uma infraestrutura para implementar uma RAN completa com controle fechado, a identificação de um conjunto de casos de uso "chave" que alavancam esses recursos estendidos ainda está em desenvolvimento. A O-RAN Alliance fornece uma lista de casos de uso relevantes para o controle habilitado para RIC, que inclui o controle "clássico" de gerência de recursos de rádio (RRM) relacionados a transição de conexão de forma otimizada, alocação de recursos, otimização de qualidade de experiência (Quality of Experience - QoE), entre outros. Apesar disso, com o avanço dos padrões 3GPP para novos cenários como redes não-terrestriais (Non-Terrestrial Networks - NTN) e suporte a Realidade Aumentada (Augmented Reality - AR)/ Realidade Virtual (Virtual Reality - VR) no metaverso, torna-se necessário refinar ainda mais e avaliar os futuros casos de uso Open RAN em suporte à estes novos cenários de uso desafiadores.

Um destes cenários desafiadores é o de redes privativas implementadas com componentes Open RAN. As redes celulares privadas estão emergindo rapidamente como um cenário chave de implantação de 5G, com aplicações em automação industrial, armazéns, saúde, educação e entretenimento. As implantações 5G privadas *greenfield* (i.e. com novas instalações) podem incorporar facilmente soluções Open RAN para controle e otimização de rede, como também reduzir os custos de propriedade e operação graças a nós desagregados e virtualizados. O projeto de redes privadas habilitadas para O-RAN apresenta desafios em termos de otimização específica de domínio, integração com sistemas de ponta e interrupções locais e suporte para conectividade em ambientes restritos.

Outro cenário desafiador é o da implementação de soluções de compartilhamento de espectro baseadas em componentes Open RAN. Controladores de rede e nós RAN programáveis abertos provêem novas oportunidades para o desenvolvimento de sistemas de compartilhamento de espectro. As especificações O-RAN já incluem recursos para compartilhamento de espectro dinâmico 4G e/ou 5G, mas o design de algoritmos para permitir isso ainda é um assunto em aberto. Pesquisas futuras podem investigar como habilitar de forma prática a detecção baseada em Open RAN, soluções de adaptação de espectro reativa e proativa, considerando sistemas 3GPP e não 3GPP, bem como extensões relacionadas ao compartilhamento da arquitetura O-RAN.

É importante observar que o desenvolvimento das tecnologias Open RAN é impulsionado pelas necessidades específicas de grandes players internacionais, como operadoras, provedores de serviços de nuvem e integradores, expressas nas diretrizes estabelecidas nesses fóruns. Geralmente, essas diretrizes têm como objetivo atender às demandas

dos principais mercados do Hemisfério Norte. No entanto, os desafios enfrentados na implantação das tecnologias Open RAN para casos de uso no Brasil e na América Latina podem não ser os mesmos encontrados nesses grandes centros.

Por exemplo, a região enfrenta deficiências de infraestrutura que não são encontradas nos principais centros da América do Norte, Europa e Ásia. Isso pode inviabilizar a desagregação dos componentes da RAN ou a implementação em larga escala de recursos de Mobile Edge Cloud (MEC) devido à falta de infraestrutura de fibra óptica. Além disso, podem existir casos de uso específicos da região que não são abordados pelas diretrizes existentes, bem como um ambiente regulatório e de negócios diversificado. É necessário considerar também as necessidades dos players de pequeno e médio porte na região.

Portanto, é essencial garantir que esses interesses únicos no Brasil e na América Latina sejam adequadamente atendidos no desenvolvimento das tecnologias Open RAN. É necessário um esforço para adaptar as soluções e diretrizes existentes às realidades regionais, considerando as deficiências de infraestrutura, casos de uso específicos e o ambiente regulatório e de negócios local. Isso garantirá que as tecnologias Open RAN sejam relevantes e eficazes para os desafios e necessidades específicas do Brasil e da América Latina.

A automação e orquestração avançadas em ambientes MEC com múltiplos "locatários"da infraestrutura de computação de borda (i.e. *multi-tenant*) e/ou com uso de recursos de múltiplas "nuvens"(i.e. *multi-cloud*) apresentam desafios significativos. O compartilhamento de recursos MEC entre múltiplos "locatários"permite uma utilização mais eficiente da infraestrutura disponível, reduzindo os custos gerais e promovendo uma instalação e operação da MEC mais eficientes e econômicas do que com recursos MEC de um único proprietário/locatário.

Além disso, o compartilhamento de recursos MEC em cenários multilocatários oferece flexibilidade e escalabilidade adicionais. Como os recursos da MEC são compartilhados entre vários "locatários", é mais fácil ajustar a alocação desses recursos de acordo com a demanda variável, adicionando ou removendo "locatários"conforme necessário. Isso facilita o suporte a uma ampla variedade de usuários e aplicativos, permitindo escalar a alocação de recursos MEC de um "locatário"para cima ou para baixo conforme necessário.

Além dos benefícios mencionados, o compartilhamento de recursos MEC entre múltiplos "locatários"oferece oportunidades adicionais de colaboração e inovação. Ao permitir que os "locatários"compartilhem recursos da MEC e se conectem entre si, cria-se um ambiente dinâmico e interativo, no qual eles podem aprender uns com os outros e desenvolver novas ideias e soluções.

No entanto, a implementação efetiva dessa automação e orquestração avançadas em ambientes MEC multi-tenant/multi-cloud apresenta desafios técnicos e operacionais. É necessário desenvolver mecanismos robustos para gerenciar a alocação dinâmica de recursos, garantir a segregação adequada entre os "locatários"e garantir a qualidade de serviço e a segurança da rede. Além disso, a interoperabilidade entre diferentes provedores de MEC e a padronização de interfaces e protocolos são essenciais para facilitar o compartilhamento de recursos e a colaboração entre os "locatários".

Dessa forma, a automação e orquestração avançadas em ambientes MEC multi-tenant/multi-cloud são áreas que exigem pesquisa e desenvolvimento contínuos, buscando soluções eficientes e seguras para permitir o compartilhamento de recursos, a flexibilidade, a escalabilidade e a colaboração entre os "locatários"na MEC.

Um dos principais desafios para avançar nesse tema é gerenciar e coordenar a alocação de recursos entre vários locatários e provedores. Em um ambiente multilocatário ou multinuvem, pode haver muitos usuários e aplicativos diferentes competindo pelo acesso aos mesmos recursos, o que dificulta garantir que todos tenham acesso aos recursos necessários. Portanto, é necessário implementar o isolamento entre workloads para permitir o atendimento multilocatário, permitindo que várias operadoras compartilhem o mesmo datacenter e ambiente multinuvem com diferentes elementos de cada fornecedor. Isso requer um planejamento e coordenação cuidadosos para evitar conflitos ou gargalos, bem como modificações em software, protocolos existentes e integração entre diferentes sistemas e clouds.

Outro desafio é aproveitar ao máximo o uso de algoritmos de Inteligência Artificial (IA) para lidar com a enorme quantidade de parâmetros e ações possíveis para controlar um sistema complexo, com múltiplos componentes, provedores e clientes. Além disso, há o objetivo de reduzir o consumo de energia das redes 5G, o que pode ser potencializado pelo emprego de algoritmos de inteligência artificial na gestão da rede fim-a-fim ou pela introdução de hardware especializado que acelere o processamento de algoritmos específicos enquanto diminui o consumo de energia.

Outro desafio é garantir compatibilidade e interoperabilidade entre diferentes locatários e provedores. Em um ambiente multilocatário ou multinuvem, pode haver muitas tecnologias e fornecedores diferentes envolvidos, o que dificulta garantir que todos os componentes da infraestrutura de rede possam trabalhar juntos perfeitamente. Isso requer o uso de protocolos e interfaces abertos e padronizados para garantir que diferentes locatários e provedores possam se conectar e se comunicar entre si.

A adaptação das tecnologias MEC multilocatário e multinuvem às particularidades do ecossistema do país e da América Latina também é um desafio. Isso inclui as grandes distâncias entre os centros principais e as localidades remotas, bem como deficiências na infraestrutura de Telecomunicações. Além disso, é necessário alinhar os desenvolvimentos em termos de software, hardware, padrões e protocolos com o que está sendo discutido para as redes de 6ª geração (6G).

Além disso, no desenvolvimento das tecnologias Open RAN em si, existem desafios importantes a serem enfrentados. Um deles é a evolução da arquitetura Open RAN. Ainda há questões em aberto sobre como essa arquitetura pode ser implantada de forma efetiva, como a distribuição dos elementos de rede em toda a rede de borda e nuvem, ou a proporção adequada entre os nós RAN e os elementos RIC. É necessário realizar mais pesquisas para projetar extensões adicionais da arquitetura O-RAN, que permitam, por exemplo, o controle em tempo real na RAN, juntamente com xApps, para aproveitar dados que não podem ser transferidos para análise na RAN ou no RIC, como amostras de sinal modulado (i.e. sinal I/Q) ou informações de estimativa em granulação fina do canal.

Além disso, é preciso lidar com o envolvimento do controle em múltiplas escalas

temporais. Considerando a arquitetura Open RAN completa, diferentes malhas de controle operam e têm visibilidade no sistema em diferentes escalas de tempo, o que gera desafios em termos de controle multiescala. Portanto, são necessárias pesquisas abrangentes no design de algoritmos multiescala temporais, levando em consideração a identificação de instabilidades no sistema, bem como possíveis conflitos entre diferentes loops de controle. Também é importante explorar a seleção automatizada de malhas de controle ideais que possam ser usadas para atingir níveis específicos de desempenho.

Em resumo, o desenvolvimento das tecnologias Open RAN enfrenta desafios como a evolução da arquitetura Open RAN, a criação de extensões que permitam o controle em tempo real na RAN em conjunto com xApps, e o envolvimento do controle em múltiplas escalas temporais, exigindo pesquisas adicionais e a implementação de algoritmos apropriados para superar esses desafios.

Outros dois grandes desafios tecnológicos na área de Open RAN são a eficiência energética e a segurança das redes. A virtualização e o controle de malha fechada fornecem recursos úteis para a alocação dinâmica de funções de rede e, assim, para maximizar a eficiência energética. É importante avançar nas pesquisas para desenvolver rotinas de orquestração que incorporem a eficiência energética como uma meta de otimização, além de xApps e rApps que adotem ações de controle ou políticas que incluam a eficiência energética.

No que diz respeito à segurança, a abertura da RAN aumenta a superfície de ataque, mas também possibilita novas abordagens para a segurança da rede. Por exemplo, a maior visibilidade do desempenho da RAN, a telemetria e a capacidade de implantar plug-ins xApps e rApps para análise de segurança e detecção de ameaças abrem caminho para explorar novas abordagens para proteger as redes sem fio, tornando-as mais robustas e resilientes. É essencial investir em pesquisa e desenvolvimento de abordagens de segurança que aproveitem os recursos do Open RAN e aprimorem a integridade, resiliência e disponibilidade de suas implantações. Isso é fundamental para tornar as abordagens Open RAN uma alternativa viável e preparada para o futuro em comparação às implantações de RAN tradicionais.

Em resumo, os desafios tecnológicos relacionados à eficiência energética e segurança nas redes Open RAN exigem avanços em pesquisas para incorporar a eficiência energética na otimização da orquestração, bem como o desenvolvimento de abordagens de segurança que sejam adaptadas às características do Open RAN, visando proteger e fortalecer as redes sem fio.

5.3. Iniciativas OpenRAN pelo Mundo

Uma das principais plataformas open source para a implementação dos diferentes componentes Open Ran é a Open Air Interface (OAI). A OAI é uma plataforma open source para o desenvolvimento e implantação de sistemas de comunicação sem fio via Rádio Definido por Software (Software Defined Radio - SDR) e Virtualização das Funções de Rede (Network Function Virtualization - NFV). Ela permite que engenheiros, pesquisadores e desenvolvedores realizem pesquisa e desenvolvimento utilizando um ecossistema de desenvolvimento para o core da rede e para a camada de acesso das redes celulares baseadas no 3GPP.

A plataforma OAI foi fundada pela Aliança de Software do OpenAirInterface (OpenAirInterface Software Alliance - OSA), um consórcio sem fins lucrativos para o desenvolvimento de um ecossistema aberto de redes celulares 3GPP. Tal consórcio foi fundado pelo instituto de pesquisa EURECOM na França e hoje está altamente alinhado com a filosofia Open RAN, tendo sido recentemente assinado um termo de alinhamento (i.e. *Memorandum of Understanding* - MoU) entre O-RAN Alliance e OSA, com o objetivo de cooperação em RANs abertas [OAI 2023c].

No sentido de promover um ecossistema Open RAN, a OSA conta, além de uma comunidade ativa no desenvolvimento de componentes 3GPP e interfaces abertas Open RAN, com parceiros estratégicos que possibilitam uma gama de desenvolvimentos e prototipagem de testes com o objetivo de tornar a RAN cada vez mais versátil e acessível para diferentes casos de uso.

Ao longo dos anos, a OAI evoluiu e ganhou força significativa nas comunidades de pesquisa e indústria. Tornou-se uma plataforma amplamente reconhecida para prototipagem, teste e validação de sistemas de comunicação sem fio. O projeto OAI recebeu contribuições de várias organizações, incluindo instituições acadêmicas, participantes da indústria e desenvolvedores individuais, tais como Qualcomm, Vodafone, Nokia Bell Labs, NVIDIA, entre outros.

O foco inicial da OAI estava na tecnologia 4G Long Term Evolution (LTE). O projeto teve como objetivo fornecer uma implementação de código aberto da pilha LTE, incluindo os componentes da rede de acesso por rádio (RAN), como a estação base e o User Equipment (UE). Dessa forma, permitiu-se que pesquisadores e desenvolvedores experimentassem diferentes recursos LTE, otimizassem o desempenho e desenvolvessem novos aplicativos e serviços.

Com o surgimento da tecnologia 5G, a Open Air Interface expandiu seu escopo para incluir suporte para redes 5G New Radio (NR). O projeto incorporou as especificações do 3rd Generation Partnership Project (3GPP) para 5G e forneceu uma implementação de código aberto do 5G RAN. Isso permitiu que os desenvolvedores explorassem os recursos do 5G, como taxas de dados mais altas (Enhanced Mobile Broadband - eMBB), baixa latência (Ultra-Reliable Low-Latency Communication - URLLC) e comunicações massivas do tipo máquina (massive Machine-Type Communications - mMTC).

Desta forma, a plataforma OAI está permitindo que os desenvolvedores e pesquisadores não sejam mais limitados à abordagem tradicional para desenvolvimento das comunicações sem fio utilizando hardware e software proprietários, o que limitava a inovação e tornava desafiador os experimentos com novos protocolos e algoritmos. Atualmente, a OAI está desenvolvendo 3 projetos: 5G RAN, 5G Core Network e o MOSAIC5G. Abaixo discutiremos cada um desses projetos.

Essa discussão proporcionará uma visão abrangente das iniciativas e projetos relevantes que estão moldando o cenário do OpenRAN globalmente, destacando o esforço colaborativo da comunidade para impulsionar a adoção e o avanço dessa nova abordagem na arquitetura de redes de acesso sem fio.

5.3.1. 5G RAN

O projeto OAI 5G RAN se concentra na criação de uma solução RAN personalizável e interoperável. Ela fornece uma implementação definida por software da pilha de protocolo 5G RAN que pode ser implantada em hardware de uso geral, permitindo a avaliação e implementação de novos algoritmos, protocolos e arquiteturas [OAI 2023b].

Principais recursos do projeto 5G RAN:

- **Código aberto**: O projeto fornece acesso ao código-fonte completo, permitindo que os usuários entendam e modifiquem a implementação de acordo com suas necessidades e requisitos.
- **Implantação flexível**: O projeto pode ser implantado em hardware comercial pronto para uso de prateleira (i.e. *Commercial Off-The-Shelf* - COTS), permitindo soluções econômicas, facilitando o acesso e incentivando a inovação.
- **Pilha de protocolos**: O projeto abrange vários aspectos da pilha de protocolos 5G RAN, incluindo camada física, MAC (Medium Access Control), RLC (Radio Link Control), PDCP (Packet Data Convergence Protocol), RRC (Radio Resource Control), entre outros.
- **Interoperabilidade**: O projeto visa garantir compatibilidade e interoperabilidade com outros componentes e interfaces da rede móvel, permitindo uma integração com a infraestrutura de rede móvel existente.

5.3.2. 5G Core Network

O projeto OAI 5G Core está focado no desenvolvimento de uma implementação 5G Core Network (5GC) flexível e personalizável. O objetivo é fornecer uma plataforma aberta para pesquisadores, desenvolvedores e operadores de rede explorarem, experimentarem e contribuírem para a evolução das principais tecnologias de rede 5G [OAI 2023a].

Um fato importante é que o projeto 5G core foi base para a plataforma Magma. O Magma é uma plataforma de software de código aberto que oferece às operadoras de rede uma solução de núcleo de rede móvel aberta, flexível e extensível. Os recursos incluem um núcleo de pacote móvel distribuído centrado em software, automação de rede avançada e ferramentas de gerenciamento e capacidade de coexistir com redes LTE existentes [MAGMA 2023].

Importante ressaltar também que provedores regionais do Brasil que adquiriram recentemente frequências no leilão da ANATEL estão com planos de utilizar o Magma como core da rede. Isto está acontecendo devido aos altos custos de implantação do Core da rede utilizando empresas tradicionais como Ericsson e Nokia [TELETIME 2023b]. Portanto, o Magma surge como uma alternativa de baixo custo para a sua implantação. Atualmente, podemos citar a operadora Brisanet, uma operadora regional que adquiriu recentemente frequências no leilão da ANATEL, que implantou o sistema e está utilizando-o comercialmente [TELETIME 2023a].

Principais recursos do projeto 5G Core Network:

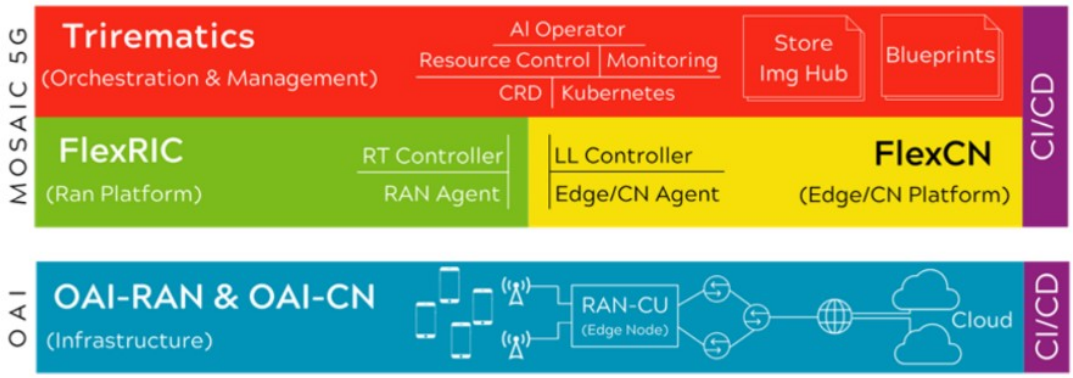


Figura 5.4: Recursos do projeto MOSAIC5G. [OAI 2023d]

- **Código aberto**: O projeto fornece acesso ao código-fonte completo, permitindo que os usuários entendam e modifiquem a implementação de acordo com suas necessidades e requisitos.
- **Conformidade com 3GPP**: O projeto visa alinhar-se às especificações definidas pelo 3rd Generation Partnership Project (3GPP), responsável pela padronização das tecnologias de comunicação celular. Ele se esforça para garantir a conformidade com os padrões 3GPP para promover a interoperabilidade e compatibilidade com outros componentes 5G.
- **Network Function Virtualization (NFV)**: O projeto fornece flexibilidade e escalabilidade na implantação de funções de rede através do uso de NFV. Isso permite que as funções de rede sejam virtualizadas em hardwares de uso geral, facilitando a economia e gerenciamento de recursos.

5.3.3. MOSAIC5G

O projeto MOSAIC5G visa transformar a RAN e o CN em plataformas de entrega de serviços de rede ágeis e abertas. Tal plataforma permite explorar novos casos de uso de interesse para diferentes indústrias verticais [OAI 2023d].

Principais recursos do projeto MOSAIC5G:

- **Código aberto**: O projeto fornece acesso ao código-fonte completo, permitindo que os usuários entendam e modifiquem a implementação de acordo com suas necessidades e requisitos.
- **Protocolo E2**: O projeto fornece a implementação do protocolo E2. O protocolo E2 é dividido em E2 Application Protocol (E2AP) e E2 Service Model (E2SM). O E2AP é um protocolo básico que coordena como o near-Real Time RIC e os nós E2 se comunicam entre si e fornece um conjunto básico de serviços. E2SM define um modelo de dados padrão para a troca de informações e controle entre os componentes RAN

- **FLEXRIC** - FlexRIC é a abreviação de "Flexible RAN Intelligent Controller". Ele faz interface com a pilha de rádio OAI através da interface E2 para monitorar e controlar a RAN em tempo real. O FlexRIC possui um modelo de serviço (SM) integrado para monitoramento e fatiamento, que pode ser facilmente personalizado e estendido para atender aos diversos casos de uso de 5G.

Também podemos citar algumas contribuições alinhadas com a filosofia Open RAN que surgiram através das parcerias citadas anteriormente:

- **Integração do split 7.2 O-RAN**: em parceria com a VVDN Technologies, a integração desse split é uma melhoria importante para a pilha de protocolos 5G OAI, abrindo a oportunidade de integração de Radio Units (RUs) 5G em conformidade com o padrão O-RAN e facilitando a comercialização de uma solução 5G Open RAN com software rodando em hardware de uso geral e RUs comerciais (COTS);
- **Interoperabilidade com L1 NVIDIA Aerial SDK**: parceria com a NVIDIA, sendo outra oportunidade de utilização de RUs comerciais Open RAN, mas dessa vez em uma integração que explora o split NFAPI da pilha OAI e a utilização de parte da camada física rodando em componentes COTS da NVIDIA (GPUs e NICs) com implementação aberta rodando em containers;
- **Implementação do split 2 (interface F1)**: implementação do split já vem sendo explorada no OAI há algum tempo, mas recentemente testes de interoperabilidade com CU do parceiro Accelleran vem sendo feitos com objetivo de aumentar a robustez da implementação da interface e possibilitar interoperabilidade também com outros fornecedores;
- **Implementação da interface E2**: iniciada dentro do projeto MOSAIC5G visando integração com o FlexRIC, hoje a implementação visa ser compatível com as especificações O-RAN para a interface E2 e busca integração também com o μONOS RIC do projeto ONOS da ONF.

5.3.4. O-RAN Alliance

A própria O-RAN Alliance disponibiliza diversas implementações open source dentro da arquitetura Open RAN, são elas: SMO, nonRT-RIC, nearRT-RIC, O-CU e O-RU. Dessa forma, a O-RAN Alliance permite a disseminação do conhecimento e testes de acordo com a sua norma. Abaixo detalhamos um pouco de cada um dos projetos da O-RAN Alliance:

- **SMO**: o SMO de referência da O-RAN Alliance é chamado de Orchestration and Management (OAM), seguindo as especificações dos documentos do grupo de trabalho da O-RAN Alliance (WG1). O mesmo é baseado em partes da implementação do projeto ONAP tendo como destaque a implementação e suporte de modelos YANGs que são usados para a configuração e monitoramento através do uso de NETCONF (O1). Através da interface O1, o projeto provê também emuladores de O-CU e O-RU para integração via O1 com o SMO.

- **Near-RT RIC**: implementação de referência seguindo as especificações da O-RAN Alliance habilitando terminações A1, E2 e O1. É baseado nos componentes executados como micro serviços dentro de um cluster Kubernetes, com possibilidade de embarcar os xApps usando imagens Docker.
- **nonRT-RIC**: A implementação do nonRT-RIC foca na interface A1, responsável pela comunicação entre o nonRT-RIC e o nearRT-RIC. Dessa forma, o projeto visa a testes usando xApps que utilizem a interface para comunicação de ambas soluções da O-RAN Alliance: o SMO (OAM) e o nearRT-RIC (OSC).

5.3.5. Consórcios e projetos Open RAN

Historicamente, a infraestrutura da Rede de Acesso Rádio (RAN) em redes celulares era limitada a um pequeno número de fornecedores de grande porte, devido à complexidade e aos altos custos de pesquisa e desenvolvimento envolvidos. Essas soluções geralmente consistiam em sistemas proprietários de um único fornecedor, nos quais o hardware e o software eram integrados, resultando em sistemas RAN fechados e com pouca competição e flexibilidade no mercado.

No entanto, esse cenário começou a mudar com o avanço da capacidade computacional disponível na nuvem (computação em nuvem) e a redução dos custos de desenvolvimento e operação de plataformas de hardware de aceleração, como GPUs, ASICs e FPGAs. Esses avanços, combinados com o desenvolvimento de padrões abertos para as interfaces entre os componentes da RAN, possibilitaram a realização do conceito de RAN aberta e desagregada.

A abordagem de RAN aberta e desagregada visa desagregar a RAN tradicional, abrindo e padronizando os protocolos e interfaces entre os vários componentes da RAN, como rádios, hardware e software. Isso permite a criação de um ecossistema mais competitivo e vibrante de provedores de soluções RAN. Com a desagregação, é possível combinar diferentes componentes de diferentes fornecedores, escolhendo aqueles que melhor atendem às necessidades específicas de uma operadora.

A RAN aberta e desagregada traz benefícios significativos para a indústria de telecomunicações. Ela promove a inovação e a competição no mercado, pois permite que fornecedores de diferentes especialidades contribuam para o desenvolvimento da RAN. Além disso, possibilita a adoção de soluções mais flexíveis e escaláveis, pois os componentes podem ser atualizados e substituídos independentemente uns dos outros. Isso reduz a dependência de um único fornecedor e oferece às operadoras mais opções para personalizar e otimizar sua infraestrutura de rede.

Em suma, a evolução da capacidade computacional, a redução dos custos de hardware de aceleração e o desenvolvimento de padrões abertos permitiram a concretização do conceito de RAN aberta e desagregada. Essa abordagem traz mais competição, flexibilidade e inovação para o mercado de RAN, rompendo com o modelo tradicional de fornecedores de grande porte e sistemas fechados.

Dentre as primeiras iniciativas da indústria nesse sentido encontram-se o xRAN Forum, fundado em outubro de 2016 por empresas como a AT&T, Deutsche Telekom, SK Telecom, Intel, entre outras. O xRAN Forum visava desenvolver, padronizar e promo-

ver uma alternativa aberta à RAN tradicional. Da mesma forma, o conceito de C-RAN (Centralized/Cloud-RAN), introduzida pela China Mobile em 2010, visava centralizar as funções da unidade de banda base (BBU) para alcançar eficiências de custo e melhorias de desempenho. Finalmente, a O-RAN Alliance foi o resultado de uma fusão entre o xRAN Forum e o C-RAN em 2018, visando alinhar e focar esforços na abertura dos protocolos e interfaces entre os subcomponentes da RAN. Ela se constitui como uma comunidade global de operadoras de redes móveis e fornecedores de tecnologia com a missão de reestruturar a maneira como as RAN são construídas e operadas.

Desde sua criação, a O-RAN Alliance tem trabalhado na criação de interfaces abertas e padronizadas e na promoção de uma RAN definida por software e virtualizada para incentivar implantações de vários fornecedores e um ecossistema aberto de soluções RAN. A O-RAN Alliance também promove o uso de tecnologias de inteligência artificial e aprendizado de máquina para melhorar a eficiência e o desempenho das RANs. Em termos técnicos, a O-RAN Alliance define padrões para interfaces abertas entre vários elementos de uma RAN aberta por meio de vários grupos de trabalho (Working Groups - WGs), cada um focando em um aspecto da rede. Cada grupo de trabalho é composto por representantes de várias empresas, de uma maneira muito similar à como o próprio 3GPP também é constituído. Os principais grupos de trabalho são:

- **WG1 - Use Cases and Overall Architecture Workgroup**: Ele tem responsabilidade geral pela arquitetura O-RAN e casos de uso. O WG 1 identifica as tarefas a serem concluídas no escopo da arquitetura e dos casos de uso e atribui aos líderes de WG a condução dessas tarefas enquanto trabalham com outros WGs da O-RAN Alliance.

- **WG2 - Non-real-time RAN Intelligent Controller and A1 Interface Workgroup**: O principal objetivo do Non-RT RIC é oferecer suporte ao gerenciamento de recursos de rádio inteligente em tempo não-real, otimização de procedimento de camada superior, otimização de política em RAN e fornecimento de modelos AI/ML para Near-RT RIC.

- **WG3 - Near-real-time RIC and E2 Interface Workgroup**: O foco deste WG é definir uma arquitetura baseada em controlador inteligente de rádio em tempo quase real (Near-RT RIC), que permite controle e otimização quase em tempo real de elementos e recursos RAN por meio de ações e coleta de dados refinadas pela interface E2.

- **WG4 - Open Fronthaul Interfaces Workgroup**: O objetivo deste WG é fornecer interfaces front-haul verdadeiramente abertas, nas quais a interoperabilidade DU-RU de vários fornecedores pode ser realizada.

- **WG5 - Open F1/W1/E1/X2/Xn Interface Workgroup**: O objetivo deste WG é fornecer especificações de perfil de vários fornecedores totalmente operáveis (que devem ser compatíveis com a especificação 3GPP) para interfaces F1/W1/E1/X2/Xn e, em alguns casos, propor aprimoramentos de especificação 3GPP.

- **WG6 - Cloudification and Orchestration Workgroup**: O WG de orquestração e migração para a cloud busca conduzir a dissociação do software RAN das plataformas de hardware subjacentes e produzir tecnologias e projetos de referência que permitiriam que as plataformas de hardware de commodities fossem aproveitadas para todas as partes de uma implantação de RAN, incluindo a CU e a DU.
- **WG7 - White-box Hardware Workgroup**: A promoção de hardware de design de referência aberta é uma maneira potencial de reduzir o custo da implantação do 5G que beneficiará tanto as operadoras quanto os fornecedores. O objetivo deste Grupo de Trabalho é especificar e liberar um design de referência completo para promover uma plataforma de software e hardware desacoplada.
- **WG8 - Stack Reference Design Workgroup**: O objetivo deste WG é desenvolver a arquitetura de software, design e plano de liberação para a CU O-RAN (O-CU) e DU O-RAN (O-DU) com base nas especificações O-RAN e 3GPP para o Pilha de protocolo NR.
- **WG9 - Open X-haul Transport Workgroup**: Esse WG centra-se no domínio dos transportes, constituído pelos equipamentos de transporte, meios físicos e protocolos de controle/gestão associados à rede de transporte.
- **WG10 - OAM for O-RAN**: Esse Grupo Técnico (GT) é responsável pelos requisitos do OAM, arquitetura e interface O1.
- **WG11 - Security Work Group**: Esse WG concentra-se nos aspectos de segurança do ecossistema RAN aberto.

Além desses WGs, existe ainda o Test & Integration Focus Group (TIFG), que define a abordagem geral do O-RAN para teste e integração, incluindo a coordenação de especificações de teste em vários WGs. Isso pode incluir a criação de especificações de teste e integração de ponta a ponta; perfis para facilitar a produtização, operacionalização e comercialização de O-RAN; abordagens para atender aos requisitos gerais; e especificações de processos para realização de integração e verificação de soluções. O TIFG planeja e coordena os O-RAN ALLIANCE PlugFests e define diretrizes para os Centros de Integração e Teste Aberto (Open Testing and Integration Centres - OTIC) terceirizados.

Em muitas medidas complementares a estas atividades, o Telecom Infra Project (TIP) foi formado em 2016 com o objetivo de possibilitar acesso global para todos através de uma metodologia colaborativa e focada em engenharia para a construção e implantação de infraestrutura de rede de telecomunicações global. O projeto é dirigido conjuntamente por seu grupo de empresas de tecnologia e telecomunicações fundadoras, que formam seu conselho de administração. As empresas membros hospedam laboratórios e aceleradoras de tecnologia incubadora e o TIP hospeda uma conferência anual de infraestrutura. A organização promove transparência de processo e colaboração no desenvolvimento de novas tecnologias, com mais de 500 organizações membros participantes, incluindo operadoras, fornecedores, desenvolvedores, integradores, startups e outras entidades. Essas organizações participam em vários grupos de projetos TIP, que empregam estudos de caso atuais para evoluir equipamentos e softwares de telecomunicações em formas mais flexíveis, ágeis e interoperáveis.

O TIP opera como uma entidade sem fins lucrativos que se dedica a avançar a conectividade global. A organização é liderada por um conselho de administração representando empresas líderes. O TIP se dedica a entender as necessidades de seus membros prestadores de serviços e outros membros do ecossistema e identificar lacunas na indústria que se alinham com a missão do TIP. A comunidade propõe ou define casos de uso específicos e recomenda a formação de grupos de projetos onde os membros colaboram em soluções. O comitê técnico do TIP apoia esses grupos, identificando um caminho a seguir e removendo obstáculos para garantir um caminho claro para o sucesso.

O TIP é uma organização que visa impulsionar a inovação e a colaboração na indústria de telecomunicações. Ele é composto por vários elementos interligados, começando pelos grupos de projetos, que abrangem todas as áreas da rede de telecomunicações, desde o acesso até o núcleo e os serviços.

Esses grupos de projetos trabalham de forma colaborativa para identificar problemas-chave enfrentados pela indústria e propor soluções inovadoras. Eles se concentram em construir a tecnologia que aborda esses problemas, desenvolvendo soluções práticas e viáveis.

As soluções desenvolvidas pelos grupos de projetos são testadas e validadas em ambientes de laboratório. O TIP possui Laboratórios Comunitários que são espaços físicos que facilitam a colaboração entre os membros do grupo de projetos, permitindo a construção e teste das soluções propostas.

Além dos laboratórios, o TIP também realiza eventos chamados de PlugFests, nos quais as soluções são testadas em um ambiente de ponta a ponta, composto por vários elementos de rede interoperáveis. Esses PlugFests verificam a prontidão técnica das soluções desenvolvidas.

Após a validação em laboratório e PlugFests, as soluções são testadas em cenários do mundo real. O TIP realiza Testes de Campo em ambientes controlados de pequena escala e testes de mercado em ambientes comerciais maiores. Essas etapas permitem avaliar o desempenho e a eficácia das soluções em diferentes situações reais.

Quando uma tecnologia desenvolvida pelo TIP se torna madura e está pronta para ser implantada, ela é listada na TIP Exchange. A TIP Exchange é uma plataforma que fornece informações sobre as soluções validadas e permite que os operadores identifiquem as tecnologias mais adequadas para suas necessidades. As soluções listadas na TIP Exchange recebem emblemas que representam os níveis de validação realizados pela comunidade TIP.

O TIP promove um ciclo constante de feedback entre os grupos de projetos e os membros que estão produzindo cada componente da pilha de tecnologia. Esse feedback contínuo permite ciclos de desenvolvimento mais rápidos e eficientes, garantindo que as soluções atendam às necessidades específicas dos operadores e sejam relevantes para o mercado.

Em resumo, o TIP impulsiona a colaboração e a inovação na indústria de telecomunicações por meio de grupos de projetos, laboratórios, PlugFests e testes em campo. Ele permite o desenvolvimento e a validação de soluções práticas e viáveis, que são lista-

das na TIP Exchange quando estão prontas para implantação.

Um dos grupos de projeto de destaque dentro do Telecom Infra Project (TIP) é o Open RAN Project Group, cujo objetivo é estabelecer e construir soluções de Rede de Acesso Rádio (RAN) para redes de comunicação móvel de segunda, terceira, quarta e quinta geração (2G, 3G, 4G e 5G), baseadas em hardware de propósito geral "neutro- isto é, sem depender de um fornecedor específico - interfaces abertas e software aberto.

Esse grupo de projeto trabalha em estreita colaboração com organizações como a O-RAN Alliance e o 3GPP na definição de padrões, buscando coletar requisitos comuns, desenvolver componentes de software aberto em parceria com organizações como a Open Network Foundation (ONF), criar plataformas de teste e validação, promover testes de campo e implementações de novas redes, e, por fim, fomentar a adoção de redes Open RAN por meio da plataforma TIP Exchange.

Dentro do Open RAN Project Group, existe um subgrupo específico chamado subgrupo de RU (Remote Radio Unit), que se concentra no desenvolvimento de hardware de RU construído com uma arquitetura aberta e desagregada. O objetivo é obter reduções significativas nos custos operacionais de instalação e operação em comparação com as soluções proprietárias existentes.

Essas iniciativas visam impulsionar a inovação, a interoperabilidade e a adoção de soluções Open RAN, permitindo a construção de redes mais flexíveis, escaláveis e econômicas, além de promover a competição e a diversidade de fornecedores no mercado de infraestrutura de telecomunicações.

Conforme mencionado anteriormente, a Open Networking Foundation (ONF) é uma organização sem fins lucrativos cujo objetivo é acelerar a adoção de soluções de rede abertas e desagregadas. Foi fundada em 2011 por grandes empresas como Deutsche Telekom, Facebook, Google, Microsoft, Verizon e Yahoo, entre outras. O principal propósito da ONF é reinventar a infraestrutura de rede e criar novas arquiteturas de rede que sejam abertas e programáveis. A ONF tem uma estrutura única na qual combina as forças de projetos de código aberto tradicionais, operadores de rede, fornecedores e pesquisa acadêmica. A organização tem várias áreas de trabalho, referidas como "projetos", cada uma focando em um aspecto específico de rede. Estes incluem Aether, ONOS (Open Network Operating System), CORD (Central Office Re-architected as a Datacenter), Stratum e outros. Esses projetos são supervisionados pela equipe de liderança técnica da ONF, que é responsável pela direção técnica da fundação.

Os produtos da ONF incluem software, padrões, melhores práticas, planos de teste e até mesmo designs de hardware para redes. O mais conhecido deles é provavelmente o projeto ONOS, que é um sistema operacional de rede definido por software (Software Defined Networks - SDN) projetado para alta disponibilidade, desempenho e escalabilidade. Outro projeto proeminente é o CORD, que visa transformar a infraestrutura de telecomunicações em uma plataforma mais flexível, econômica e escalável. No escopo de software para RAN aberta e desagregada, a ONF desenvolve alguns projetos relevantes como a plataforma de 5G e 4G chamada Aether e softwares baseado no ONOS que inclui soluções completas para a implementação de diversos componentes Open RAN. A plataforma Aether inclui projetos como o SD-RAN, um Near-RT-RIC e um conjunto de xApps

para o controle da RAN conforme os padrões Open RAN, e o SD-CORE, um conjunto de software que implementa as funcionalidades de núcleo de rede 4G / 5G.

Outra organização relevante no cenário é a Linux Foundation (LF), um consórcio tecnológico sem fins lucrativos criado em 2000 com o objetivo de padronizar o sistema operacional Linux, apoiar o seu desenvolvimento e promover a sua adoção comercial, além de hospedar e promover o desenvolvimento colaborativo de projetos de software de código aberto. Em relação à conectividade em geral e Open RAN em específico, a Linux Foundation tem várias iniciativas:

- **LF Networking (LFN)**: Uma iniciativa abrangente que visa aumentar a colaboração e a excelência operacional em projetos de rede. O LFN suporta o desenvolvimento de vários projetos de rede de código aberto, como ONAP (Open Network Automation Platform) e OPNFV (Open Platform for NFV).

- **LF Edge**: Esta iniciativa visa estabelecer uma estrutura aberta e interoperável para computação de borda independente de hardware, silício, nuvem ou sistema operacional. É uma organização guarda-chuva que hospeda vários projetos, incluindo Akraino Edge Stack, EdgeX Foundry e Open Horizon, que são relevantes para a computação de borda, uma tecnologia chave para futuras redes 5G e RAN.

- **O-RAN Software Community (OSC)**: Embora não seja um projeto exclusivo da Linux Foundation, esta é uma comunidade de código aberto com um objetivo semelhante de criar software para redes de acesso por rádio (RAN) de sistemas sem fio de próxima geração que são mais abertos e flexíveis do que os padrões atuais. Essa comunidade está associada à O-RAN Alliance, que está desenvolvendo um padrão para Open RAN.

Recentemente, eles anunciaram um projeto chamado “FL Connectivity”, criado em colaboração com a Meta (Facebook), e voltado para avançar tecnologias que irão acelerar as redes e aplicações de conectividade. Como parte deste projeto, a Meta irá introduzir três sub-projetos iniciais, a saber o Terragraph (tecnologia sem fio para viabilizar acesso em altas taxas para usuários residenciais), o Open M-Plane (software e design de hardware Evenstar da Meta para configuração e gerenciamento da RAN) e o Maveric (software para planejamento e avaliação de desempenho de instalações de rede).

Uma iniciativa notável no contexto do Open RAN é o projeto Open RAN Gym, que oferece ferramentas para aquisição de dados RAN e experimentação com Inteligência Artificial (IA), incluindo seu testbed de pesquisa em comunicações sem fio. O testbed conhecido como Colosseum é reconhecido como um dos maiores e mais avançados ambientes de teste em redes sem fio já construídos. Ele fornece uma infraestrutura flexível e escalável que permite que pesquisadores e desenvolvedores testem e avaliem algoritmos, protocolos, técnicas de transmissão, modulações, gerenciamento de espectro e outras tecnologias relacionadas à comunicação sem fio.

O Colosseum combina 128 nós de rádio padrão com um emulador de canal digital suportado por uma rede de roteamento composta por Field Programmable Gate Arrays (FPGAs). Cada um dos nós de rádio oferece uma plataforma para comunicação

por radiofrequência e aplicações de aprendizado de máquina, utilizando um servidor Dell R730 equipado com uma GPU NVIDIA K40M e um Universal Software Radio Peripheral (USRP) Ettus X310 equipado com uma FPGA XILINX Kintex 7.

O Colosseum é capaz de criar ambientes virtuais que simulam rádios operando em áreas abertas, centros urbanos, shoppings ou desertos, gerando mais de 52 terabytes de dados por segundo. Cada nó pode hospedar imagens de pilhas de rádio 4G/5G, Core de rede, RIC ou UE, possibilitando a criação de cenários sofisticados e inovadores, como otimização de recursos durante a segmentação de redes (network slicing) por meio de xApps que fazem uso de técnicas de aprendizado de máquina.

Essa infraestrutura avançada oferecida pelo Colosseum permite que pesquisadores e desenvolvedores explorem diversas possibilidades e conduzam experimentos de ponta, impulsionando o desenvolvimento e aprimoramento contínuo das tecnologias Open RAN, bem como a aplicação de IA para otimizar o desempenho e a eficiência das redes de comunicação sem fio.

5.4. Programa OpenRAN Brasil

O Programa OpenRAN@Brasil é uma iniciativa voltada para impulsionar a adoção e desenvolvimento da tecnologia Open RAN no Brasil. Em parceria com empresas, instituições acadêmicas e governamentais, o programa tem como objetivo promover a inovação, a colaboração e o avanço das soluções de redes de acesso sem fio. Nesta seção, serão apresentadas as motivações por trás do programa, uma descrição detalhada do programa em si, bem como informações sobre o seu testbed.

5.4.1. Motivação do Programa

O acesso à Internet está se tornando cada vez mais um serviço essencial, assim como o fornecimento de energia elétrica e água. A pandemia de COVID-19 evidenciou o papel fundamental que a Internet exerce para a continuidade das atividades econômicas, sociais e acadêmicas, seja pela viabilização do teletrabalho, seja pela realização de aulas online para escolas e universidades. Além disso, os usuários passaram a ter uma necessidade de uso constante da Internet, com capacidade de banda para suportar diversas atividades, como: videoconferências, aplicações de tempo real e streaming de vídeos, por exemplo.

Com este cenário, o investimento em infraestrutura para conectividade passou a ser uma prioridade em diversos países. No entanto, muitas nações têm dificuldades em manter e atualizar sua infraestrutura de rede para atender estas demandas crescentes, sendo o próprio Brasil um exemplo. Além do cenário costumeiramente desafiador, os fornecedores de tecnologia impõe soluções proprietárias e fechadas, o que acaba gerando lock-ins, desta forma inviabilizando o uso de outras soluções, assim encarecendo os equipamentos e limitando a inovação e oportunidades para a indústria nacional.

Uma forma de atenuar este problema é por meio do desenvolvimento e adoção de soluções abertas para tecnologias de rede. Iniciativas como o TIP (Telecom Infra Project), projeto liderado pela Meta, e ONF (Open Networking Foundation), da Linux Foundation, bem como O-RAN Alliance, buscam oferecer uma pilha de tecnologia aberta. Essas iniciativas visam impulsionar o desenvolvimento e adoção de soluções abertas, estimulando

a desagregação, na qual o software é embarcado em hardware de tipo white box, que pode ser desenvolvido por diferentes fornecedores.

O modelo atual de elementos de rede verticalizados e fechados com hardware, software e interfaces proprietárias não garante a flexibilidade e o custo-benefício necessários para os provedores. Assim, o movimento de desagregação causa dois impactos. Primeiro, o hardware produzido passa a ser totalmente aberto, permitindo a fabricação por diferentes fornecedores e o desenvolvimento por terceiros do software a serem embarcados nesses equipamentos. Em segundo, o software desenvolvido tende a adotar o conceito de software de código aberto, liderado por comunidades que envolvem operadores, indústria e academia.

Estas mudanças levam a uma redução nos custos dos equipamentos e estimula uma maior inovação, assim, permitindo que operadoras e os provedores possam implementar e testar novas funcionalidades de forma mais ágil, resultando em serviços de melhor qualidade. Além disso, os provedores de rede podem reduzir os seus gastos operacionais e em infraestrutura, uma vez que por conta da desagregação, só é necessário trocar uma parte dos equipamentos.

Diante dessas vantagens, é de suma importância que o Brasil participe desse movimento. A médio prazo, trará benefícios tanto para os pequenos provedores de rede quanto para os usuários, além de impulsionar a criação de um ecossistema de inovação ao redor dessa iniciativa. Esse ecossistema envolverá fabricantes nacionais de equipamentos de rede, integradores, desenvolvedores de software e pesquisadores.

5.4.2. Programa OpenRAN@Brasil

O OpenRAN@Brasil [ORAN-Brasil 2023] é um programa do Ministério da Ciência, Tecnologia e Inovação (MCTI), executado em conjunto pela RNP (Rede Nacional de Ensino e Pesquisa), CPQD (Centro de Pesquisa e Desenvolvimento em Telecomunicações), Inatel e Eldorado. O programa visa acelerar o desenvolvimento do ecossistema de redes abertas, desagregadas e programáveis a partir de pesquisa, desenvolvimento, inovação e capacitação em tecnologias e aplicações, em 5G e além.

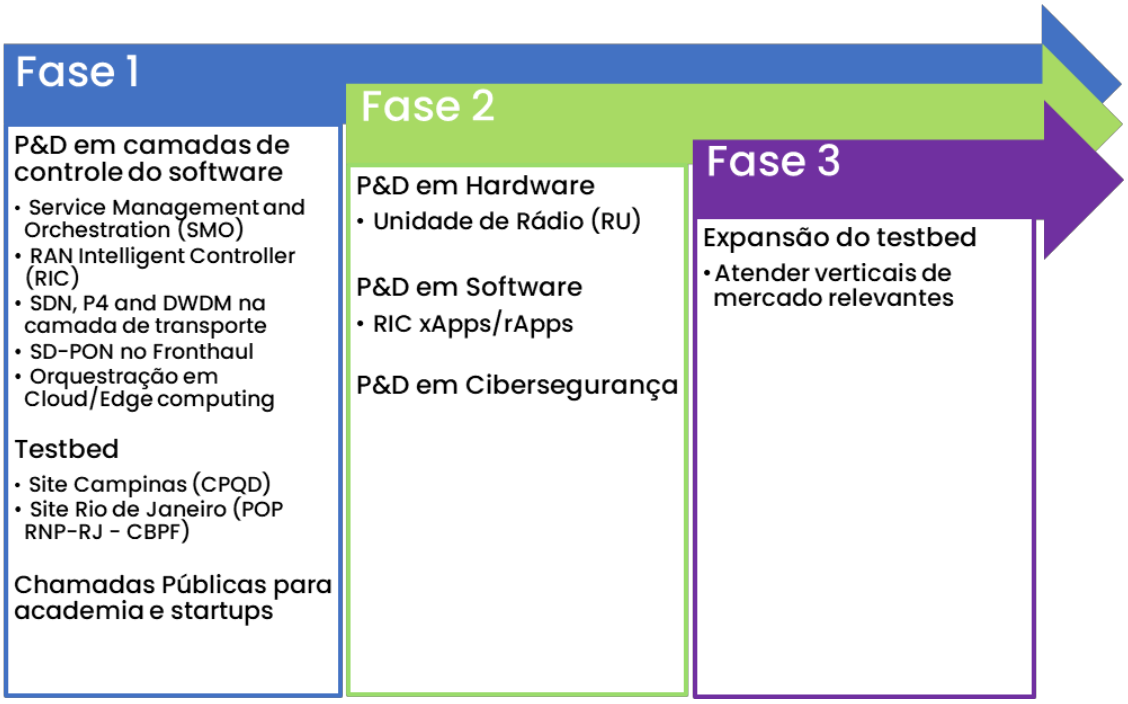


Figura 5.5: Visão macro das fases do Programa OpenRAN@Brasil.

O Programa OpenRAN@Brasil para alcançar a visão estabelecida, elenca três pilares:

- Pesquisar, desenvolver, implantar e validar soluções inovadoras de gerenciamento e controle inteligente de redes abertas e desagregadas em diferentes domínios tecnológicos (redes de pacotes, acesso rádio e óptico);
- Construir e disponibilizar infraestruturas de rede softwarizadas (testbed), nesses diferentes domínios tecnológicos, que adotam as tendências de abertura e desagregação, com a finalidade de integrar, demonstrar e validar aplicações inovadoras;
- Capacitar profissionais de TI, desenvolvedores e engenheiros de rede no paradigma de softwarização das infraestruturas de rede e engajar outros atores (universidades, empresas) e outras iniciativas no processo de inovação

O Programa OpenRAN@Brasil é composto por três fases, estas apresentadas de forma resumida na imagem 5.5. Abaixo, cada um dos estágios é detalhado.

Fase I: A primeira fase do Programa teve seu início em dezembro de 2021 e a possui uma duração de 36 meses. Para este estágio do projeto foram elencados os seguintes objetivos:

1. A pesquisa e desenvolvimento de tecnologias que habilitam a softwarização de infraestruturas de rede multi-domínio com o uso de soluções abertas e desagregadas;
2. A construção e operação de uma plataforma de experimentos (testbed) composto por equipamentos de pacotes, ópticos e sem fio abertos e desagregados;

3. A fomentação de um ecossistema de inovação na área de desenvolvimento de softwares inteligentes para o controle e gerenciamento de redes abertas e desagregadas, através da disseminação de conhecimento e capacitação de estudantes, pesquisadores e desenvolvedores;

A pesquisa e o desenvolvimento realizados nesta parte do projeto, tem como objetivo focar nos componentes de software para gerenciamento, controle e automação de infraestruturas. No caso, são estudados os seguintes elementos da arquitetura: Service Management and Orchestration (SMO); RIC (RAN Intelligent Controller); na camada de transporte são estudados o DWDM, FFTX e P4; SD-PON no Fronthaul; e Orquestração em Cloud e Edge Computing.

O ambiente de experimentação que está sendo construído neste estágio do OpenRAN@Brasil, será composto de dois sites, um ficará em Campinas, implementando no ambiente do CPQD e administrado pelo mesmo, e outro site ficará na cidade do Rio de Janeiro, nas instalações do CBPF (Centro Brasileiro de Pesquisas Físicas), sendo que este último site será administrado pela RNP. Nesta seção haverá um maior detalhamento deste ambiente de testes.

Para a fomentação do ecossistema de inovação, destacamos a realização das seguintes atividades: o lançamento da Chamada Pública do Programa OpenRAN@Brasil, para aumentar o engajamento da comunidade acadêmica com o projeto; a realização de workshops para as comunidades da academia, indústria e operação com o objetivo disseminar o Programa; a participação do projeto em eventos, como o WRNP; e em futuro próximo ações de capacitação e uma Chamada destinada para o meio de startups.

A execução desta etapa é feita em conjunto pela RNP e pelo CPQD. Além das instituições previamente mencionadas, também há a colaboração de pesquisadores das universidades UNICAMP, UFF, UFRGS e UFRJ.

Fase II: Esta etapa do Programa começou a sua execução no final do ano de 2022 e tem duração de 30 meses. O seu objetivo é o desenvolvimento de uma unidade de rádio nacional 5G (O-RU 5G) aderente aos requisitos definidos pela O-RAN Alliance para uso em macrocélulas na banda de sub-6GHz. A escolha por focar o P&D neste componente se dá por conta que a maior parte do custo de uma solução Open RAN está concentrada na unidade de rádio e, também, por existirem poucos fabricantes no mundo para este componente, assim diminuindo a competitividade.

O equipamento que está em desenvolvimento tem como pontos norteadores: o baixo custo, a alta programabilidade e o atendimento de nichos de mercado relevantes para o desenvolvimento do País. Ao final do projeto, a O-RU 5G desenvolvida será integrada e testada em conjunto com os demais componentes da solução Open RAN, fornecidos por terceiros, o que permitirá exercitar as diferentes interfaces abertas previstas na arquitetur

Além do objetivo de desenvolver um hardware nacional, nesta fase também será abordada a pesquisa e desenvolvimento de aplicações SDN inteligentes na camada de inteligência na rede de acesso rádio, os xApps/rApps, e em aspectos de cibersegurança da arquitetura OpenRAN.

O grupo atuante é composto pela RNP, como instituição coordenadora, CPQD,

Inatel e Eldorado atuando como instituições executoras. O time de execução também contará com pesquisadores de universidades especialistas nas áreas de 5G, redes de acesso de rádio (RAN), inteligência artificial, segurança cibernética e no paradigma Open RAN. Este último time participa da equipe de execução da RNP, liderada internamente pela DPDI (Diretoria de Pesquisa, Desenvolvimento e Inovação).

Fase III: Nesta fase, o Programa expandirá o testbed proposto na Fase I para as demais áreas do País, no caso, as áreas Norte, Nordeste, Sul e Sudeste devem ser contempladas com novos sites. Essas novas infraestruturas serão focadas em verticais temáticas de interesse, como saúde e indústria. Esta etapa ainda encontra-se em elaboração.

5.4.3. Testbed do Programa OpenRAN@Brasil

Como apresentado na seção anterior, a fase 1 do programa OpenRAN@Brasil a tem o objetivo de pesquisar e desenvolver softwares para a construção de uma plataforma de código aberto para o controle e gerenciamento de infraestruturas de rede programáveis compostas por equipamentos abertos e desagregados. Isso significa que esses equipamentos são construídos integrando vários componentes fornecidos por diferentes fabricantes de hardware e software. Portanto, é essencial construir um testbed baseado nessas tecnologias para fornecer um ambiente de recursos para experimentadores e executar os casos de uso planejados no projeto.

As soluções mencionadas envolvem o uso de equipamentos abertos, que são gerenciados e orquestrados por sistemas abertos desenvolvidos por comunidades, fóruns e consórcios internacionais. Essas soluções aproveitam os novos paradigmas de softwarização, virtualização e desagregação. Além disso, essas soluções abertas abrangem o controle de infraestruturas de redes avançadas que englobam múltiplos domínios tecnológicos, como os domínios de pacotes (*IP/Ethernet*), óptico (transporte e acesso) e de acesso sem fio (rede celular 5G e *WiFi*).

Para este testbed, além das tecnologias de redes de acesso sem fio (5G) baseadas em Open RAN, ele também é composto por mais duas outras tecnologias: redes de acesso óptico através da tecnologia PON (*Passive Optical Network*) e redes de transporte com tecnologias de pacotes (IP/Ethernet) e ópticas WDM (*Wavelength Division Multiplexing*) - Redes de Multiplexação por Divisão de Comprimento de Onda. Além disso, o testbed contará com uma camada de computação em nuvem/borda para oferecer suporte aos softwares de aplicações, controle e orquestração da infraestrutura, bem como para processamento dos dados provenientes das antenas 5G Open RAN (RIC, CU e DU).

Na fase 1 do programa, foram estabelecidos dois locais para a construção do testbed inicial. O primeiro será localizado na RNP, junto ao PoP (Ponto de Presença da RNP) do Rio de Janeiro, no prédio do CBPF (Centro Brasileiro de Pesquisas Físicas). O segundo local está situado no prédio do CPQD (Centro de Pesquisa e Desenvolvimento em Telecomunicações). Para permitir a integração entre os locais, será estabelecido um enlace de 10 Gbps. Essa conexão possibilitará a realização de experimentos conjuntos, aproveitando os recursos disponíveis em ambos os sites, conforme ilustrado na Figura 5.6.

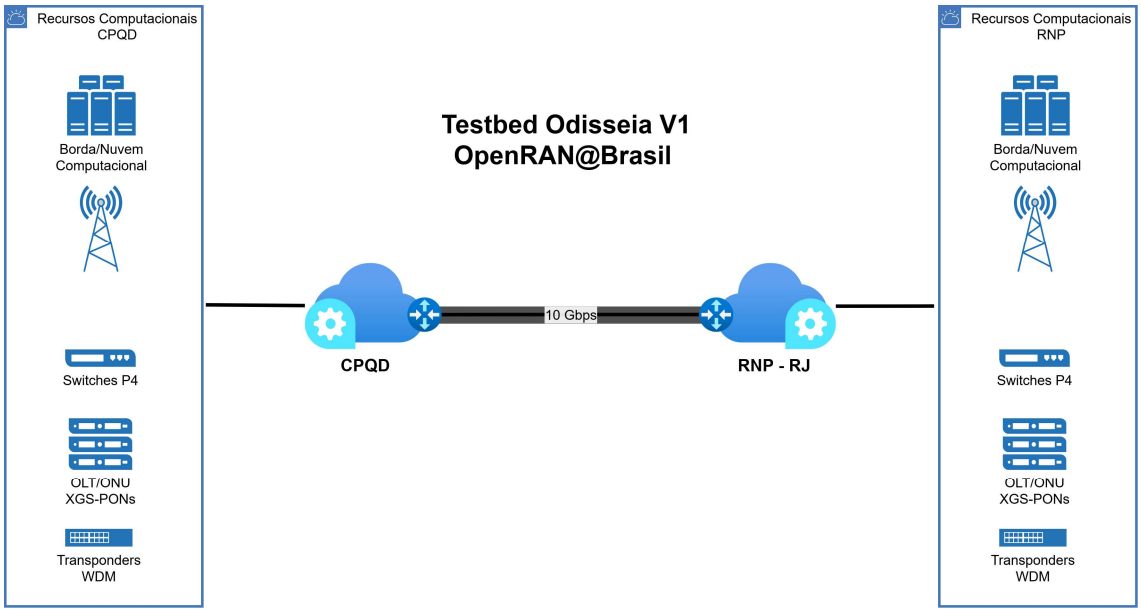


Figura 5.6: Visão global do testbed do OpenRAN@Brasil fase 1.

Basicamente, cada um dos sites possui um conjunto de 6 a 7 servidores, que estão categorizados em 5 configurações de hardware específicas de acordo com seus objetivos dentro do testbed. Além disso, cada site inicialmente contará com um conjunto de 3 antenas indoor 5G baseadas em Open RAN, seguindo a arquitetura split 7.2X da O-RAN Alliance.

No domínio de pacotes, será aplicada uma topologia leaf-spine, na qual apenas a camada de leafs será composta por switches P4. Além disso, essa topologia permitirá a utilização de um projeto interno da RNP chamado de P7 [Rodriguez et al. 2022], que possibilitará a manipulação e criação de topologias virtuais, assim como a criação de enlaces com características personalizadas, como atraso, largura de banda e variação do atraso.

No domínio óptico, serão disponibilizadas duas tecnologias. A primeira é baseada em PON, com as tecnologias XGS-PON e GPON. A segunda é baseada em WDM, utilizando transponders ópticos.

Nas Figuras 5.7 e 5.8, apresentam-se as topologias físicas de ambos os sites, RNP e CPQD, respectivamente. Esses diagramas representam as conexões físicas implementadas dentro de cada instituição e servirão como modelos para a fase de implantação do testbed.

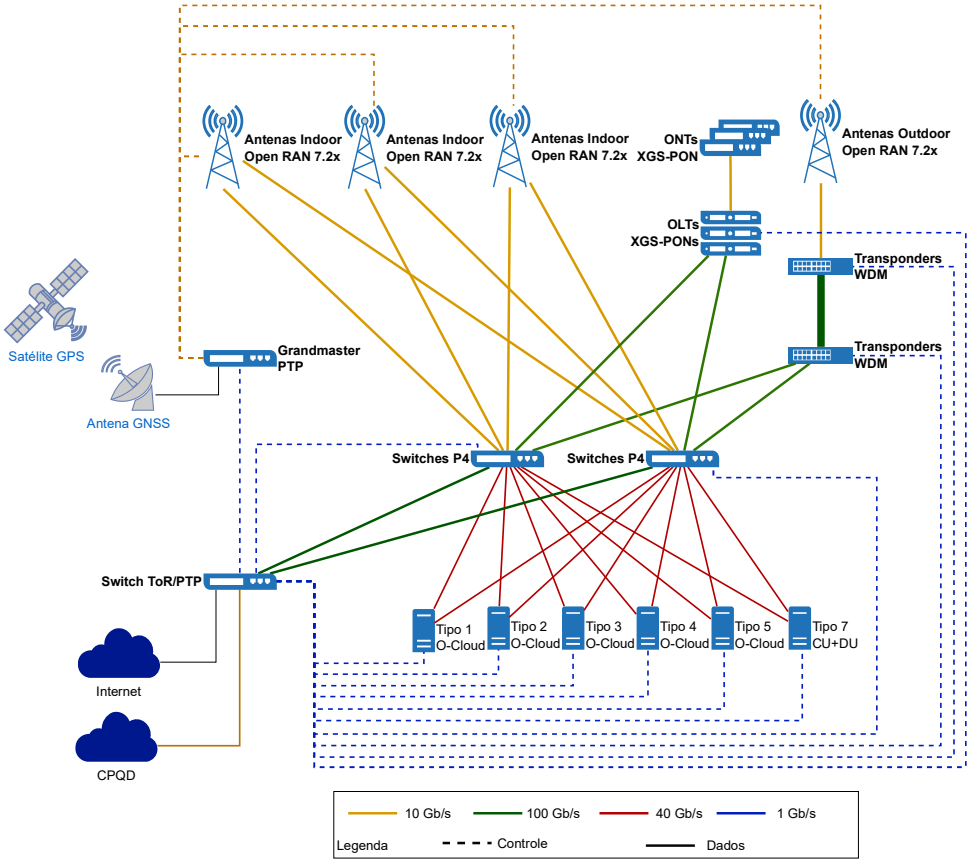


Figura 5.7: Arquiteura do física dos testbeds na RNP

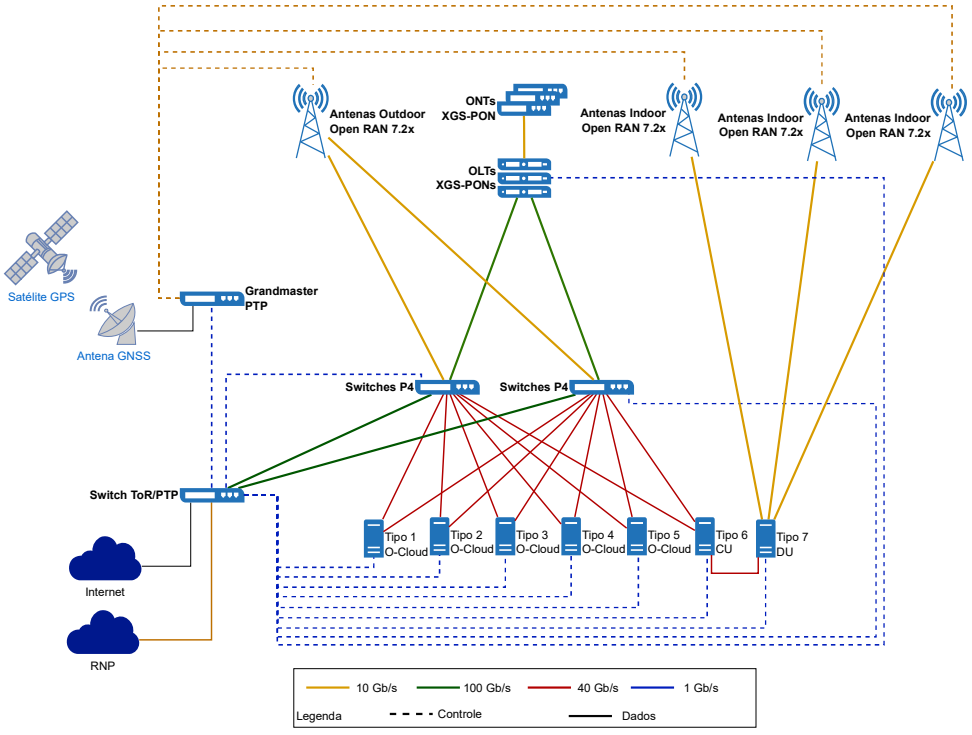


Figura 5.8: Arquiteura do física dos testbeds na CPQD

5.4.4. Hardwares utilizados no testbed

A especificação de hardware para os servidores do domínio cloud foi baseada nos requisitos das redes 5G privadas, nos requisitos de implantação do testbed e nas recomendações de funcionalidades e capacidades indicadas pelos projetos e aplicações previstas para utilização no projeto do testbed. Os servidores foram agrupados em tipos e conjuntos, apresentando características e recursos intercambiáveis entre si. Isso permite o uso das aplicações e projetos em diferentes topologias, além de possibilitar a análise da melhor arquitetura tanto para a infraestrutura de cloud do testbed quanto para as aplicações do testbed, visando o fornecimento do serviço final.

Inicialmente, foi criada uma classificação e nomenclatura com base na função geral dos servidores dentro do testbed. A tabela abaixo resume os tipos de servidores que compõem o domínio da cloud em cada localidade selecionada (RNP: RJ - Rio de Janeiro, e CPqD: CPS - Campinas), utilizando esse critério:

Tipo de Servidor Função Geral Localidade Cloud Controller Controlar e orquestrar a infraestrutura da cloud RNP: RJ Application Server Hospedar e executar aplicações do testbed RNP: RJ Storage Server Armazenar e gerenciar dados e recursos do testbed RNP: RJ Edge Server Prover serviços de borda e processamento local RNP: RJ, CPqD: CPS Management Server Gerenciar e monitorar a infraestrutura do testbed CPqD: CPS

Tipo de Servidor	Função Geral	Localidade
Cloud Controller - Tipo I	Controlar e orquestrar a infraestrutura da cloud	RNP, CPQD
Application Server - Tipo III e IV	Hospedar e executar aplicações do testbed	RNP, CPQD
Storage Server - Tipo VI	Armazenar e gerenciar dados e recursos do testbed	RNP, CPQD
Edge Server - Tipo II e VII	Prover serviços de borda e processamento local	RNP, CPQD
Management Server - Tipo IV e V	Gerenciar e monitorar a infraestrutura do testbed	RNP, CPQD

Tabela 5.1: Arquiteura do física dos testbeds na CPQD

Essa classificação e nomenclatura foram estabelecidas levando em consideração as funções específicas desempenhadas por cada tipo de servidor dentro do domínio da cloud em cada local escolhido. O fabricante fornecedor dos servidores selecionado foi a SUPERMICRO, devido ao preço oferecido e à facilidade de personalização dos servidores para atender às necessidades específicas, principalmente em relação a processadores e placas de aceleração. A lista completa dos tipos de servidores e suas configurações está disponível no Anexo A.

Em relação aos demais equipamentos especificados neste testbed, segue a classificação abaixo, descrevendo as características de cada equipamento adquirido, de acordo com seu domínio, fabricante, modelo, entre outros:

- **Domínio de Acesso Sem Fio (*5G-Open RAN - Indoor*):**

- Fabricante: Foxconn
- Modelo: HW-MOBE-RPQN-7801E
- Características:
  - ORAN option 7.2 over 10Gb/s RJ45/SFP+
  - 5G NR x 4 (4TR)

- **Domínio de Acesso Óptico (PON):**
  - Fabricante:
    - Radisys
    - Edgecore
  - Modelo:
    - RLT-1600X OLT (RLT-1600X) Any PON (XGS-PON e GPON)
    - ASXvOLT16 (ASXvOLT16-O-AC-F-US) XGS-PON
- **Domínio de Transporte Óptico (WDM):**
  - Fabricante:
    - Edgecore
  - Modelo:
    - AS7716-24SC (Cassini)
- **Domínio de Transporte de Pacotes (IP/Ethernet):**
  - Fabricante:
    - Edgecore
  - Modelo:
    - DSC 801 (Wedge100BF-32QS)
    - DSC 810 (AS9516-32D)
- **Domínio de Sincronização (PTP Grandmaster):**
  - Fabricante:
    - FIBROLAN
  - Modelo:
    - Falcon RX (5G xHaul Timing Aware O-RAN Switch & PTP Grandmaster)
- **Domínio de Gerenciamento/Controle (Switch de Gerenciamento - ToR):**
  - Fabricante:
    - UFISpace
  - Modelo:

- S9600-72XC (Open Aggregation Router)

Essa classificação fornecerá uma visão geral dos equipamentos adquiridos para cada domínio específico do testbed, incluindo detalhes sobre fabricante, modelo e outras características relevantes.

5.4.5. Escolha da pilha Open RAN

Em relação à pilha de processamento da RAN, conforme definida pela O-RAN Alliance, foi observado que as iniciativas das comunidades que desenvolvem código aberto ainda não possuem implementações completas de todas as interfaces do O-RAN.

Levando em consideração os critérios e justificativas mencionados acima, foram realizadas pesquisas sobre as opções disponíveis no mercado de fornecedores, como Mavenir, Baicells, Nokia, Radisys, entre outros, que estão surgindo e/ou adaptando-se ao conceito de Open RAN. Devido à natureza relativamente nova dessa tecnologia, muitos fornecedores ainda não adotaram todas as funcionalidades do O-RAN, especialmente em relação às interfaces E2, A1 e O1. Alguns fornecedores ainda utilizam soluções híbridas, combinando interfaces proprietárias com as interfaces padronizadas pela O-RAN Alliance. Essa migração para uma solução totalmente baseada em interfaces abertas é um processo mais lento e está planejada no roadmap desses fornecedores. Esse cenário já era esperado, uma vez que a tecnologia/arquitetura ainda está em processo de amadurecimento e sua aplicação em ambientes de produção requer uma maturidade mínima.

Atualmente, 90% das implantações mundiais de Open RAN se baseiam em soluções da Intel, que desenvolveu uma solução de referência de camada física, chamada FlexRAN, otimizada para seus processadores. Quando membros do projeto consultaram representantes da Intel para a disponibilização da solução FlexRAN ao projeto, os mesmos recomendaram consulta à empresa *Radisys*, para que houvesse um suporte adequado, uma vez que a Intel não pode atender no momento um projeto de pequena escala como o do Testbed proposto. Ademais, foi informado também que a Radisys dispõe de implementação própria das camadas de mais alto nível (L2/L3), e que esta implementação é utilizada como base de produtização por boa parte dos fornecedores comerciais. Somado ao fato de que a *Radisys* vem apoiando trials de demonstração da tecnologia Open RAN em comunidades como a *Open Networking Foundation*, há uma série de fatores que levam a esta recomendação.

Tal projeto foi destacado e usado como referência na justificativa para uso do Open RAN. Nos documentos oficiais do projeto *Intel Smart Edge*, existe a referência ao uso da pilha RAN da Radisys junto com camadas mais baixas da pilha RAN da Intel FlexRAN.

Devido à estreita colaboração da *Radisys* com o *Intel Smart Edge* e o suporte negociado que eles fornecerão em nossa implementação do TestBed, bem como a disponibilidade das interfaces O-RAN E2, A1 e O1, e considerando que a *Radisys* foi o fornecedor que mais atendeu aos nossos critérios de seleção, foi decidido selecionar a pilha Open RAN da *Radisys* para o nosso projeto. Essa escolha baseia-se na confiança na capacidade da Radisys de fornecer as soluções e o suporte necessários para o sucesso do nosso TestBed.

5.4.6. Licenciamento do Espectro 5G

Como uma instituição de pesquisa e desenvolvimento, temos o objetivo de impulsionar a inovação tecnológica e contribuir para o avanço científico no campo das telecomunicações. O 5G representa uma nova geração de tecnologia de comunicação sem fio, com potencial para transformar setores como saúde, indústria, transporte e muito mais. É uma oportunidade única para explorar e compreender a aplicabilidade do 5G em cenários específicos e seu impacto nas mais diversas áreas.

Por isso, buscamos obter autorização para conduzir experimentos práticos, testes de viabilidade e pesquisas científicas que nos permitirão avaliar o desempenho e as capacidades do 5G em diferentes cenários e aplicações. Esses estudos nos ajudarão a compreender melhor os benefícios e os desafios associados ao uso do 5G e Open RAN, além de fornecer insights valiosos para o desenvolvimento de soluções tecnológicas avançadas.

É importante ressaltar que o objetivo dos nossos experimentos e pesquisas é estritamente científico e experimental. Não temos a intenção de fornecer serviços comerciais ou competir com as operadoras de telecomunicações estabelecidas. Nossas atividades serão realizadas em ambiente controlado, em conformidade com as regulamentações e diretrizes estabelecidas pela Anatel, a fim de evitar interferências prejudiciais a outros serviços e usuários do espectro.

Acreditamos que o pedido de licenciamento de espectro 5G para fins científicos e experimentais é fundamental para impulsionar a pesquisa e o desenvolvimento de tecnologias avançadas no Brasil. Além disso, contribuiremos para o fortalecimento do ecossistema de inovação, promovendo a colaboração entre instituições acadêmicas, empresas e outras entidades envolvidas no campo das telecomunicações.

5.5. Considerações Finais e o Futuro do OpenRAN

A evolução das redes de comunicação tem impulsionado uma demanda crescente por conectividade, inovação e flexibilidade. Nesse contexto, surgiram iniciativas como o OpenRAN, que visa transformar a infraestrutura de redes de acesso por rádio (RAN) através de uma abordagem aberta, desagregada e interoperável. O OpenRAN promete revolucionar a maneira como as redes são projetadas, implantadas e gerenciadas, oferecendo uma série de benefícios para operadoras, fornecedores e usuários finais. O projeto representa uma oportunidade única para impulsionar a inovação e transformar o cenário das redes de comunicação no país. Com base nas informações fornecidas sobre as organizações ONF (Open Networking Foundation), Linux Foundation e a plataforma OAI (Open Air Interface), podemos destacar várias conclusões motivadoras que mostram o potencial e os benefícios do projeto OpenRAN Brasil em uma escala ainda maior:

- **Desenvolvimento de uma infraestrutura de comunicação moderna e flexível**: O OpenRAN oferece uma abordagem inovadora para a construção de redes de comunicação, baseada em componentes desagregados e interoperáveis. Ao adotar essa abordagem, o projeto OpenRAN Brasil tem o potencial de desenvolver uma infraestrutura de comunicação moderna e flexível, capaz de se adaptar rapidamente às demandas em constante evolução. Isso significa que o país poderá aproveitar as novas oportunidades proporcionadas pelas tecnologias emergentes, como Internet

das Coisas (IoT), inteligência artificial e veículos autônomos.

- **Estímulo à concorrência e redução dos custos**: A adoção do OpenRAN Brasil pode abrir espaço para a entrada de novos fornecedores de hardware e software, aumentando a concorrência no mercado de telecomunicações. Isso pode levar a uma redução dos custos para os operadores de rede, bem como para os usuários finais, resultando em serviços mais acessíveis e de melhor qualidade. Além disso, a interoperabilidade entre diferentes componentes e fornecedores no OpenRAN permite que as operadoras escolham as soluções mais adequadas para suas necessidades, sem ficarem presas a um único fornecedor.

- **Fortalecimento da indústria nacional de tecnologia**: O projeto OpenRAN Brasil pode impulsionar o desenvolvimento da indústria nacional de tecnologia, incentivando a colaboração entre empresas locais, universidades e instituições de pesquisa. Ao fornecer uma plataforma aberta e acessível para experimentação e inovação, o projeto permite que os desenvolvedores brasileiros contribuam com soluções personalizadas e desenvolvam habilidades técnicas avançadas. Isso fortalece a capacidade de inovação do país e cria oportunidades de negócios, exportação de tecnologia e geração de empregos qualificados.

- **Expansão da conectividade em áreas remotas e rurais**: O OpenRAN Brasil tem o potencial de levar conectividade de alta qualidade para áreas remotas e rurais, onde a infraestrutura de telecomunicações tradicional pode ser limitada. Com a flexibilidade e a escalabilidade oferecidas pelo OpenRAN, o projeto OpenRAN Brasil pode promover a inclusão digital nessas regiões, permitindo que mais pessoas tenham acesso a serviços de comunicação e internet. Isso contribui para reduzir a desigualdade digital, proporcionando igualdade de oportunidades em termos de educação, saúde, desenvolvimento econômico e participação cívica.

- **Colaboração global e intercâmbio de conhecimentos**: O projeto OpenRAN Brasil se beneficia da colaboração global com organizações como a ONF, a Linux Foundation e a comunidade OAI. Essas parcerias permitem o intercâmbio de conhecimentos, melhores práticas e experiências com outros países que estão desenvolvendo iniciativas semelhantes. Isso amplia o alcance do projeto, aumenta sua relevância internacional e facilita a adoção de padrões e tecnologias globalmente aceitos. A colaboração global também pode atrair investimentos e apoio técnico de organizações internacionais, fortalecendo ainda mais o projeto OpenRAN Brasil.

Como pode ser visto, o projeto OpenRAN Brasil tem o potencial de revolucionar as redes de comunicação no país, estimulando a inovação, reduzindo os custos, fortalecendo a indústria nacional, expandindo a conectividade e promovendo a colaboração global. Com um compromisso contínuo das partes interessadas, investimentos adequados e uma estratégia bem definida, o projeto pode impulsionar o Brasil para a vanguarda da próxima geração de redes de comunicação. O OpenRAN oferece uma visão promissora de um futuro mais aberto, inclusivo e eficiente em termos de custos para as telecomunicações brasileiras.

Trabalhando em colaboração com operadoras, fornecedores e outros participantes da indústria, organizações como a O-RAN Alliance, Open Network Foundation e Linux Foundation visam democratizar segmentos da rede de telecomunicações, reduzir custos e diminuir a dependência de grandes indústrias de equipamentos. Através do desenvolvimento de um framework com padrões, protocolos e componentes de software de código aberto, como o OpenRAN, eles estão impulsionando a interoperabilidade, permitindo que diferentes componentes de RAN sejam combinados de forma transparente, independentemente de serem virtualizados ou desagregados.

No futuro, as tecnologias Open RAN têm o potencial de impulsionar uma série de avanços e aplicações inovadoras nas redes de comunicação. Com a flexibilidade e a interoperabilidade oferecidas pelo Open RAN, espera-se que surjam novos serviços e modelos de negócios. Por exemplo, a introdução de interfaces abertas e a capacidade de integrar componentes de diferentes fornecedores podem permitir a criação de redes híbridas, combinando infraestruturas de comunicação tradicionais com tecnologias emergentes, como redes 5G, Internet das Coisas (IoT) e computação em nuvem. Isso abrirá caminho para uma ampla gama de aplicativos, desde cidades inteligentes e automação industrial até saúde digital e veículos autônomos.

Outra utilidade para a virtualização de redes é a adoção do Open RAN para facilitar a implementação de redes privadas, permitindo que setores como manufatura, energia, transporte e agricultura tenham maior controle e segurança sobre suas infraestruturas de comunicação. Redes privadas baseadas em OpenRAN podem fornecer conectividade confiável e personalizada para atender às necessidades específicas de cada setor, permitindo a implementação de soluções avançadas, como monitoramento remoto, automação de processos e análise de dados em tempo real. Isso impulsiona a transformação digital em vários setores e promoverá a eficiência operacional, a produtividade e a inovação.

Além disso, a introdução de interfaces abertas entre os diferentes componentes do OpenRAN possibilita a obtenção de informações da RAN e a atualização de políticas de controle, permitindo a utilização de técnicas avançadas, como Inteligência Artificial (IA) e Aprendizado de Máquina (ML), para otimização e controle inteligente da RAN.

Em conclusão, o projeto OpenRAN Brasil tem o potencial de revolucionar as redes de comunicação no país, estimulando a inovação, reduzindo os custos, fortalecendo a indústria nacional, expandindo a conectividade e promovendo a colaboração global. Com um compromisso contínuo das partes interessadas, investimentos adequados e uma estratégia bem definida, o projeto pode impulsionar o Brasil para a vanguarda da próxima geração de redes de comunicação. O OpenRAN oferece uma visão promissora de um futuro mais aberto, inclusivo e eficiente em termos de custos para as telecomunicações brasileiras à medida que mais organizações e países adotam essa abordagem, espera-se que surjam novas aplicações e benefícios que impulsionam a transformação digital e contribuam para um mundo mais conectado e inteligente.

Agradecimentos

Gostaríamos de expressar nosso especial agradecimento aos seguintes colaboradores pelo seu inestimável apoio na elaboração deste material: Ariel Goes de Castro (Unipampa), Eduardo Melão (CPQD), João Paulo Sales Henrique Lima (CPQD), Maykon Renan Pe-

reira Da Silva (CPQD), Vitalii Afanasiev (CPQD) e Weskley Vinicius Fernandes Maurício (CPQD).

Também gostaríamos de estender nossos agradecimentos a todos os colaboradores do Programa OpenRAN@Brasil. Este projeto conta com o apoio do Ministério da Ciência, Tecnologia e Inovação por meio dos recursos da Lei nº 8.248, de 23 de outubro de 1991, conforme orientação da Secretaria de Empreendedorismo e Inovação do MCTI. Além disso, expressamos nossa gratidão aos contribuidores da RNP (Rede Nacional de Ensino e Pesquisa).

Referências

[Chen et al. 2023] Chen, W., Lin, X., Lee, J., Toskala, A., Sun, S., Chiasserini, C. F., and Liu, L. (2023). 5g-advanced towards 6g: Past, present, and future.

[Chen et al. 2022] Chen, W., Montojo, J., Lee, J., Shafi, M., and Kim, Y. (2022). The standardization of 5g-advanced in 3gpp. *IEEE Communications Magazine*, 60(11):98–104.

[Flynn 2020] Flynn, K. (2020). Progress on 5g releases 16/17 in 3gpp. *IEEE Communications Standards Magazine*, 4(2):4–5.

[Lin 2022] Lin, X. (2022). An overview of 5g advanced evolution in 3gpp release 18. *IEEE Communications Standards Magazine*, 6(3):77–83.

[Lin and et al. 2019] Lin, X. and et al. (2019). 5G New Radio: Unveiling the Essentials of the Next Generation Wireless Access Technology. *IEEE Communications Standards Magazine*, 3(3):30–37.

[MAGMA 2023] MAGMA (2023). Disponível: `https://magmacore.org/about-magma//`. online: acessado em 30/06/2023.

[OAI 2023a] OAI (2023a). O-ran alliance 5g core project. Disponível: `https://openairinterface.org/oai-5g-core-network-project/`. online: acessado em 07/06/2023.

[OAI 2023b] OAI (2023b). O-ran alliance 5g ran project. Disponível: `https://openairinterface.org/oai-5g-ran-project/`. online: acessado em 07/06/2023.

[OAI 2023c] OAI (2023c). O-ran alliance announces mou with oai. Disponível: `https://openairinterface.org/news/o-ran-alliance-announces-mou-with-oai-new-otics-o-ran/`. online: acessado em 06/06/2023.

[OAI 2023d] OAI (2023d). O-ran alliance mosaic5gproject. Disponível: `https://openairinterface.org/mosaic5g/`. online: acessado em 07/06/2023.

[ORAN 2022] ORAN (2022). O-ran working group 2 non-rt ric architecture. Disponível: `https://orandownloadsweb.azurewebsites.net/specifications`. online: acessado em 05/06/2023.

[ORAN 2023a] ORAN (2023a). O-ran working group 1 a1 interface: General aspects and principles. Disponível: https://orandownloadsweb.azurewebsites.net/specifications. online: acessado em 05/06/2023.

[ORAN 2023b] ORAN (2023b). O-ran working group 1 architecture description. Disponível: https://orandownloadsweb.azurewebsites.net/specifications. online: acessado em 05/06/2023.

[ORAN 2023c] ORAN (2023c). O-ran working group 1 use cases detailed specification. Disponível: https://orandownloadsweb.azurewebsites.net/specifications. online: acessado em 05/06/2023.

[ORAN 2023d] ORAN (2023d). O-ran working group 2 ai/ml workflow description and requirements. Disponível: https://orandownloadsweb.azurewebsites.net/specifications. online: acessado em 06/06/2023.

[ORAN 2023e] ORAN (2023e). O-ran working group 3 e2 general aspects and principles (e2gap). Disponível: https://orandownloadsweb.azurewebsites.net/specifications. online: acessado em 05/06/2023.

[ORAN 2023f] ORAN (2023f). O-ran working group 3 near-rt ric architecture. Disponível: https://orandownloadsweb.azurewebsites.net/specifications. online: acessado em 05/06/2023.

[ORAN-Brasil 2023] ORAN-Brasil (2023). Openran@brasil - desenvolvimento do ecossistema de redes abertas a partir de pesquisa, inovação e capacitação em tecnologias e aplicações, em 5g e além. Disponível: https://www.openranbrasil.org.br/. online: acessado em 11/06/2023.

[Parkvall et al. 2020] Parkvall, S. et al. (2020). 5g nr release 16: Start of the 5g evolution. *IEEE Communications Standards Magazine*, 4(4):56–63.

[Polese et al. 2023] Polese, M., Bonati, L., D'Oro, S., Basagni, S., and Melodia, T. (2023). Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges. *IEEE Communications Surveys Tutorials*, 25(2):1376–1411.

[Rahman et al. 2021] Rahman, I. et al. (2021). 5g evolution toward 5g advanced: An overview of 3gpp releases 17 and 18. *Ericsson Technology Review*, 2021(14):2–12.

[Rodriguez et al. 2022] Rodriguez, F., Vogt, F. G., De Castro, A. G., Schwarz, M. F., and Rothenberg, C. (2022). P4 programmable patch panel (p7): An instant 100g emulated network on your tofino-based pizza box. In *Proceedings of the SIGCOMM '22 Poster and Demo Sessions*, SIGCOMM '22, page 4–6, New York, NY, USA. Association for Computing Machinery.

[Samdanis and Taleb 2020] Samdanis, K. and Taleb, T. (2020). The road beyond 5g: A vision and insight of the key technologies. *IEEE Network*, 34(2):135–141.

[TELETIME 2023a] TELETIME (2023a). Brisanet usa tecnologia do facebook para levar conexão a locais sem infraestrutura. Disponível: https://www.opovo.com.br/noticias/economia/2021/04/14/brisanet-usa-tecnologia-do-facebook-para-levar-conexao-a-locais-sem-i html. online: acessado em 07/06/2023.

[TELETIME 2023b] TELETIME (2023b). Provedores regionais utilizam solucao de core do facebook para facilitar servico lte. Disponível: https://teletime.com.br/21/05/2019/provedores-regionais-utilizam-solucao-de-core-do-facebook-para-facili online: acessado em 07/06/2023.

[Zhang and Long 2022] Zhang, Y. and Long, B. (2022). A review of 5g-advanced service and system aspects standardization in 3gpp. In *2022 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, pages 94–99.