UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ROSELI DA ROCHA BARBOSA

UMA ARQUITETURA ORIENTADA A SERVIÇOS COM SUPORTE AO FATIAMENTO NA BORDA DA REDE PARA A INTERNET DE VEÍCULOS

Recife
2021

ROSELI DA ROCHA BARBOSA

UMA ARQUITETURA ORIENTADA A SERVIÇOS COM SUPORTE AO FATIAMENTO NA BORDA DA REDE PARA A INTERNET DE VEÍCULOS

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Redes de Computadores

Orientador: Kelvin Lopes Dias

Recife
2021

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

B238a Barbosa, Roseli da Rocha
Uma arquitetura orientada a serviços com suporte ao fatiamento na borda da rede para a internet de veículos / Roseli da Rocha Barbosa. – 2021.
77 f.: il., fig., tab.

Orientador: Kelvin Lopes Dias.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2021.
Inclui referências.

1. Redes de computadores. 2. Internet de veículos. I. Dias, Kelvin Lopes (orientador). II. Título.

004.6 CDD (23. ed.) UFPE - CCEN 2021 – 139

ROSELI DA ROCHA BARBOSA

"UMA ARQUITETURA ORIENTADA A SERVIÇOS COM SUPORTE AO FATIAMENTO NA BORDA DA REDE PARA A INTERNET DE VEÍCULOS"

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 05/03/2021.

BANCA EXAMINADORA

Prof. Dr. José Augusto Suruagy Monteiro
Centro de Informática / UFPE

Prof. Dr. Jó Ueyama
Instituto de Ciências Matemáticas e de Computação / USP

Prof. Dr. Kelvin Lopes Dias
Centro de Informática / UFPE
(Orientador)

Dedico este trabalho a minha mãe Eliene por ser meu porto seguro.

AGRADECIMENTOS

Agradeço a Deus por ter me dado saúde e força para superar as dificuldades. Conseguir concluir este trabalho no meio de uma pandemia foi desafiador.

À minha família que sempre me apoiou em minhas decisões. Mesmo distantes fisicamente foram meu porto seguro em todos os momentos.

Ao meu orientador Kelvin Lopes Dias, que se dispôs a me orientar e me auxiliar em toda a construção deste estudo e que tenho uma eterna gratidão. Obrigada por cada contribuição para a minha formação pessoal e profissional, por toda paciência e acolhimento de sempre.

A Richardson Bruno por todo apoio e incentivo nos estudos e por sempre me ajudar com seus conhecimentos. Obrigada pela confiança que em mim depositou para a realização deste trabalho. Obrigada também por todo amor e afeto de sempre.

Aos meus amigos do grupo de pesquisa, obrigada por todos os momentos de carinho, companheirismo e dedicação nessa longa caminhada. Dedico um agradecimento especial aos meus amigos Rhodney e Chico por compartilharem comigo seus conhecimentos e experiências, que me ajudaram na construção desta dissertação.

Obrigada a todos que, mesmo não estando citados aqui, tanto contribuíram para a conclusão desta etapa na qual me fez me tornar a pessoa que sou hoje.

Mensagem de Stephen Hawking enviada ao Espaço:

Eu estou muito ciente da preciosidade do tempo. Aproveite o momento. Aja agora. Passei a vida viajando através do Universo dentro da minha mente. Por meio da Física Teórica, eu procurei responder algumas das grandes questões. Mas existem outros desafios e outras grandes questões que devem ser respondidas, e para isso será preciso também uma nova geração que seja interessada, engajada e tenha entendimento de ciência.

Como vamos alimentar uma população que cresce constantemente, fornecer água limpa, gerar energia renovável, prevenir e curar doenças e desacelerar mudança climática global?

Eu espero que a ciência e a tecnologia forneçam respostas para estas perguntas, mas precisaremos de pessoas, seres humanos com conhecimento e compreensão para implementar essas soluções.

Uma das grandes revelações da era espacial foi uma nova perspectiva para a humanidade. Quando olhamos a Terra do espaço vemos a nós mesmos como um todo; enxergamos a unidade e não as divisões. É uma imagem simples com uma mensagem inquestionável: um planeta, uma espécie humana.

Estamos juntos aqui e precisamos viver juntos com tolerância e respeito. Devemos nos tornar cidadãos globais.

Eu fui muito privilegiado, por meio do meu trabalho, de contribuir para o nosso entendimento do Universo. Mas este seria um Universo realmente vazio se não fosse pelas pessoas que amo e que me amam.

Somos todos viajantes do tempo caminhando juntos para o futuro. Mas precisamos trabalhar juntos para transformar esse futuro em um lugar que de fato desejamos visitar. Seja forte, seja determinado, supere as expectativas. Isso pode ser feito. Isso pode ser feito.

(HAWKING, 2018, tradução livre)

RESUMO

As redes veiculares são fundamentais para as atuais e futuras cidades inteligentes. Espera-se que o consumo de *streaming* de vídeo, bem como de mapas de alta definição atualizados em tempo real para o suporte aos carros autônomos, aumentem significativamente o tráfego de dados nos sistemas de transporte inteligentes (ITS - *Intelligent Transportation System*). Para este fim, a infraestrutura de comunicação e computação tem que lidar com a natureza dinâmica das redes veiculares. Os serviços virtualizados, já bem conhecidos de plataformas tradicionais baseadas em nuvem, devem estar alinhados aos requisitos veiculares, como suporte à baixa latência, QoS e mobilidade. Dois componentes-chave das redes 5G auxiliam no atendimento de tais requisitos: fatiamento de rede (*Network Slicing*) e MEC (*Multi-access Edge Computing*). Enquanto o fatiamento de rede permite que diferentes serviços compartilhem o mesmo substrato de rede física por meio de redes virtuais, a MEC fornece aplicações e serviços próximos ao usuário final, isto é, na borda da rede, evitando atrasos devido ao acesso a nuvens remotas. Apesar de seus benefícios, a MEC e o fatiamento de rede são, geralmente, aplicados de forma independente como soluções para requisitos de redes veiculares. Com o objetivo de fornecer uma solução conjunta que compreenda o fatiamento de rede e MEC para a Internet de veículos, esta dissertação desenvolveu uma arquitetura orientada a serviços chamada OPENS-IoV (*On-demand Provisioning of Edge-based Network Slicing for IoV*). A dissertação concebeu um middleware orientado a mensagens com base no modelo *publish/subscribe*. As camadas da arquitetura do middleware estão alinhadas com 5G SBA (*Service Based Architecture*) e incluem o componente MANO (*Management and Network Orchestration*). A proposta consiste em uma estratégia de cache parcialmente inspirada nos princípios de NDN (*Named Data Networking*) para reduzir a latência. Além disso, mecanismos de elasticidade são propostos no contexto do gerenciamento do ciclo de vida e alocação dinâmica de VNFs (*Virtualized Network Function*) e SFCs (*Service Function Chaining*) para lidar com as flutuações do tráfego veicular. Para demonstrar a eficácia da proposta, foi desenvolvido um testbed para avaliar a transmissão de vídeos em SD e 4K em um cenário de rede veicular. Os compromissos entre o provisionamento reativo e proativo de fatias de rede e os benefícios da estratégia de cache realizada na MEC foram discutidos e demonstrados por meio de métricas como utilização de CPU, uso de memória, latência de provisionamento de fatias de rede e latência total para a entrega do serviço. Com base nos resultados obtidos, foram identificados ganhos significativos com a alocação proativa e em cenários utilizando cache. Através da estratégia de cache adotada foi possível obter uma redução de até 96,56% no tempo de resposta mínimo e até 39,84% no tempo de resposta máximo. Também foi possível constatar que o uso de mecanismos de cache pode impactar diretamente no gerenciamento dos recursos computacionais.

Palavras-chave: 5G; fatiamento de rede; MEC; *caching*; internet de veículos.

ABSTRACT

Vehicular networks are critical for current and future smart cities. The consumption of video streaming, as the real time delivery of high-definition maps to support autonomous cars, it is expected to significantly increase data traffic in Intelligent Transportation System (ITS). For this purpose, the communication and computing infrastructure has to deal with the dynamic nature of vehicular networks. Virtualized services, already well-known for traditional cloud-based platforms, must be aligned with vehicle requirements, such as support for low latency, QoS and mobility. Two key components of 5G networks aid such requirements: Network Slicing and MEC (Multi-access Edge Computing). While network slicing allows different services to share the same physical network substrate through virtual networks, MEC provides applications and services close to the end user, that is, at the edge of the network, avoiding delays due to cloud-based remote access. Despite their benefits, MEC and network slicing are generally applied independently as solutions for vehicle network requirements. In order to provide a joint solution that comprises network slicing and MEC for the Internet of Vehicles, this dissertation developed a service-oriented architecture called OPENS-IoV (On-demand Provisioning of Edge-based Network Slicing for IoV). This dissertation implemented a message-oriented middleware based on the publish/subscribe model. The middleware architecture layers are aligned with 5G SBA (Service Based Architecture) and include the MANO (Management and Network Orchestration) component. The proposal consists of a cache strategy partially inspired by the principles of NDN (Named Data Networking) to reduce latency. Besides, elasticity mechanisms are proposed in the context of life cycle management and dynamic allocation of VNFs (Virtual Network Functions) and SFCs (Service Function Chains) to deal with vehicular traffic fluctuations. To demonstrate the effectiveness of the proposal, a testbed was developed to evaluate the transmission of videos in SD and 4K in a vehicular network scenario. The tradeoffs between reactive and proactive provisioning of network slices and the benefits of the cache strategy performed at MEC were discussed and demonstrated using metrics such as CPU utilization, memory usage, network slice provisioning latency, and total latency for service delivery. Based on the results obtained, we identified significative gains with proactive allocation and scenarios using cache. By means of the cache strategy it was possible to obtain a reduction of 96.56% in the minimum response time and 39.84% in the maximum response time. In addition, it was also possible to determine that the use of cache mechanisms can directly impact the management of computational resources.

Keywords: 5G; network slicing; MEC; caching; internet of vehicles.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3GPP	*3rd Generation Partnership Project*
5G	*Fifth Generation*
API	*Application Programming Interface*
CSMF	*Communication Service Management Function*
C-V2X	*Cellular Vehicle to Everything*
CAPEX	*Capital Expenditure*
CECC	Com Elasticidade e com Cache
CESC	Com Elasticidade e sem Cache
DSRC	*Dedicated Short Range Communications*
EPC	*Evolved Packet Core*
ETSI	*European Telecommunications Standards Institute*
HSS	*Home Subscriber Server*
IaaS	*Infrastructure as a Service*
IoV	*Internet of Vehicles*
ITS	*Intelligent Transportation System*
LTE	*Long Term Evolution*
MANETs	*Mobile Ad-hoc Networks*
MANO	*Management and Network Orchestration*
MEAO	*MEC Application Orchestrator*
MEC	*Multi-access Edge Computing*
MEO	*MEC Orchestrator*
MEPM	*MEC Platform Manager*

MEPM-V *MEC Platform Manager - NFV*
MME *Mobility Management Entity*

NDN *Named Data Networking*
NF *Network Function*
NFV *Network Functions Virtualization*
NFVI *NFV Infrastructure*
NFVO NFV *Orchestrator*
NGMN *Next Generation Mobile Network*
NS *Network Slice*

OPENS-IoV *On-demand Provisioning of Edge-based Network Slicing for IoV*
OPEX *Operational Expenditure*

PCRF *Policy and Charging Rules Function*
PGW *Packet Data Network Gateway*
PIT *Pending Interest Table*
PoC *Proof-of-Concept*

QoS *Quality of Service*
QUIC *Quick UDP Internet Protocol*

RAN *Radio Access Network*
REST *Representational State Transfer*
RSU *Roadside Unit*

SD *Standard Definition*
SDN *Software Defined Networking*
SECC Sem Elasticidade e com Cache
SESC Sem Elasticidade e sem Cache
SFC *Service Function Chaining*
SGW *Serving Gateway*
SLA *Service Level Agreement*

UE	*User Equipment*
UMTS	*Universal Mobile Telecommunication System*
V2I	*Vehicle to Infrastructure*
V2V	*Vehicle to Vehicle*
V2X	*Vehicle to Everything*
VANETs	*Vehicular Ad-hoc Networks*
VIM	*Virtualized Infrastructure Manager*
VNF	*Virtualized Network Function*
VNFM	*VNF Managers*
WAVE	*Wireless Access in Vehicular Environments*
WiMax	*Worldwide Interoperability for Microwave Access*

SUMÁRIO

1 INTRODUÇÃO

Este Capítulo apresenta as principais motivações para a concepção desta dissertação, com base nos problemas de pesquisa identificados na literatura. Os conceitos essenciais deste trabalho também são introduzidos e elencados de acordo com os objetivos definidos. Por último, este Capítulo descreve o objetivo geral e os objetivos específicos, bem como a estrutura desta dissertação.

1.1 MOTIVAÇÃO

Um relatório publicado pelo Departamento de Pesquisa Statista, em 2018, estimou que até o final de 2020 83 milhões de carros no mundo estariam conectados à Internet (STATISTA, 2018b). Recentemente, em outubro de 2020, o mesmo grupo de pesquisa lançou uma nova estimativa com a projeção de mais de 330 milhões até 2035, apenas para o mercado americano, ultrapassando as previsões da China e da Europa (STATISTA, 2020a). Os autores também estimaram que, ainda em 2020, o mercado global de veículos conectados corresponderia a cerca de 54 bilhões de dólares americanos (STATISTA, 2020b).

Os usuários se beneficiarão da conectividade veicular para consumir e compartilhar conteúdos e informações na rede, como entretenimento multimídia, notícias diversas e informações fornecidas pelo Sistema de Transporte Inteligente (ITS). Tais perspectivas vêm ganhando atenção de grandes *players* do mercado de entretenimento, gerenciamento de frotas e fabricantes de automóveis *premium*, tais como: Panasonic e Pioneer, Trimble Navigation e TomTom, BMW e VW, respectivamente. Espera-se que as chamadas cidades inteligentes possam fornecer a infraestrutura necessária para a adesão aos veículos conectados (STATISTA, 2018a).

A nova geração das redes móveis, conhecida como redes de quinta geração (5G), trouxe grandes oportunidades para a inovação de serviços em setores verticais, como a indústria automobilística. Particularmente, as Redes Ad Hoc Veiculares (VANETs) e sua extensão através da Internet dos Veículos (IoV) se beneficiam das tecnologias de softwarização em 5G para gerenciar sua topologia dinamicamente inerente, requisitos de QoS, escalabilidade e melhor gerenciamento dos serviços.

Para alcançar o gerenciamento flexível em cenários de IoV baseados em 5G, fornecer a virtualização de funções de rede (*Network Function* (NF)) e aproximar os serviços de nuvem dos usuários finais é de suma importância. Por um lado, o NFV (*Network Functions Virtualization*) permite a implementação e o provisionamento de serviços sob demanda, trazendo inúmeros benefícios por facilitar a gestão dos recursos computacionais e reduzir significativamente os custos de CAPEX (*Capital Expenditure*) e OPEX (*Operational Expenditure*). Com NFV, os serviços de rede poderão ser virtualizados e executados como

softwares (*Virtualized Network Functions* (VNFs)) em máquinas virtuais, sem a necessidade de utilizar dispositivos de hardware real, além de possibilitar que estas funções de rede sejam encadeadas logicamente em uma plataforma de nuvem (OPERATORS, 2012). Por outro lado, a MEC (*Multi-access Edge Computing*) busca reduzir a latência para garantir um bom desempenho de aplicações que são sensíveis a atrasos, garantindo uma melhor experiência do usuário. Além disso, com a possibilidade de coletar informações contextuais os provedores de serviços também serão beneficiados com a computação em nuvem tradicional na borda da rede, pois com tais informações é possível agilizar o provisionamento de novos serviços (PATEL et al., 2014). Tanto o NFV como a MEC foram padronizados em conjunto pelo ETSI (*European Telecommunications Standards Institute*) na estrutura chamada MEC-NFV.

Como uma terceira tecnologia fundamental para fornecer a sinergia entre a MEC e o NFV acima mencionados, visando a criação de serviços de rede, temos o paradigma de fatiamento de rede (*Network Slicing*) que foi introduzido inicialmente pela NGMN (*Next Generation Mobile Network*) em (ALLIANCE, 2015). De maneira geral, uma fatia de rede (*Network Slice* (NS)) corresponde a um conjunto de recursos dedicados para a instanciação de serviços, que podem ser encadeados e isolados logicamente através de uma rede virtual. O principal objetivo do conceito de fatiamento de rede, desde sua idealização, consiste em possibilitar a implantação de diferentes tipos de serviços com requisitos heterogêneos, através de uma infraestrutura compartilhada.

Em (ALLIANCE, 2016) são apresentados mais detalhes sobre a concepção de fatiamento de rede e proposta uma terminologia comum para todas as camadas ou módulos funcionais, apontando suas principais características. Já o 3GPP (*3rd Generation Partnership Project*) apresenta uma visão arquitetural com as respectivas interfaces de comunicação (*Long Term Evolution* (LTE)) e até mesmo uma evolução dessa arquitetura (5G), com uma proposta mais modular e com um novo modelo de descoberta e comunicação dos serviços, usando a estratégia de *producer/consumer*. Com essa nova arquitetura, o 3GPP tem como objetivo fornecer mais flexibilidade, elasticidade e escalabilidade das principais entidades de rede e, além disso, facilitar a instanciação de EPCs (*Evolved Packet Core*) sob demanda e com diferentes requisitos.

1.2 PROBLEMA DE PESQUISA

Diante do que foi apresentado como motivação, este trabalho tem como problema de pesquisa viabilizar o compartilhamento de conteúdos para veículos conectados, tendo como base os princípios de fatiamento de rede e os requisitos previstos para as redes 5G. No melhor do nosso conhecimento, nenhum outro trabalho explorou diversos casos de uso para prover o fatiamento flexível da rede, o que torna este trabalho um grande diferencial. Diversos cenários são avaliados e explorados ao máximo, visando mensurar o comportamento de todos os elementos propostos na arquitetura.

1.2.1 Questões de Pesquisa

Apesar de existirem vários trabalhos e projetos com estratégias de fatiamento de rede, as soluções atuais carecem de uma visão orientada a serviços como proposto pelo 3GPP, e de uma abordagem dinâmica para lidar com cenários de IoV. Para este fim, identificamos as seguintes questões:

Q1. Como identificar os requisitos dos consumidores e tratá-los?

Q2. Quais os compromissos entre o provisionamento reativo e proativo de fatias de rede?

Q3. Como definir a capacidade de atendimento de uma instância de serviço?

Q4. Considerando o compartilhamento de conteúdos para veículos conectados, como tornar o arcabouço genérico para o uso de diferentes protocolos de transporte (TCP ou UDP)?

Q5. Quais os benefícios que técnicas de *caching* traria para um cenário de redes veiculares?

1.2.2 Hipóteses

Com base nas questões de pesquisa levantadas, identificamos algumas hipóteses com o desenvolvimento desta dissertação:

1. Torna-se necessária a adesão de uma entidade intermediadora que seja capaz de traduzir os requisitos dos usuários em recursos/serviços alocados.

2. O tipo de provisionamento deve ser considerado com base nas necessidades dos provedores e nas garantias exigidas pelos consumidores. Uma abordagem semelhante pode ser utilizada para definir a capacidade de atendimento de uma instância de serviço.

3. Através da adesão de um middleware torna-se possível construir uma camada de infraestrutura genérica, que seja capaz de lidar com diferentes tipos de protocolo de transporte.

4. O uso de estratégias de cache em redes veiculares é crucial por se tratar de ambiente altamente dinâmico e com desconexões frequentes, além de permitir a redução da latência total.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

O objetivo geral deste trabalho é propor uma arquitetura flexível para compartilhamento de conteúdos em redes veiculares, baseada nos princípios de fatiamento de rede e alinhada aos requisitos propostos pelo 3GPP para redes 5G.

1.3.2 Objetivos Específicos

- Conceber e implementar uma arquitetura para provisionamento dinâmico de fatias de rede, com base nos requisitos dos usuários.
- Projetar e implementar um modelo de troca de mensagens entre os produtores de conteúdos e os consumidores.
- Elaborar uma estratégia de compartilhamento de conteúdos para usuários de veículos conectados, considerando um mecanismo de cache e uma tabela de interesses.
- Desenvolver uma solução para prover a alocação de recursos, elasticidade e gerenciamento do ciclo de vida das fatias de rede.

1.4 ESTRUTURA DA DISSERTAÇÃO

Esta dissertação está organizada da seguinte forma: o Capítulo 1 contextualiza o trabalho e apresenta os objetivos propostos. No Capítulo 2, são apresentados os fundamentos e definições referentes à redes móveis de quinta geração, com foco em Redes Veiculares, NFV, MEC e Fatiamento de Rede. O Capítulo 4 apresenta diferentes propostas para prover o fatiamento da rede, discutindo os estudos relacionados a este trabalho. A avaliação da proposta é definida no Capítulo 5, esclarecendo o planejamento para a criação dos cenários de avaliação, as métricas utilizadas e os resultados obtidos. Por fim, o Capítulo 6 aponta as considerações finais encontradas com a realização desta pesquisa.

2 FUNDAMENTAÇÃO TEÓRICA

Neste Capítulo, os principais conceitos e tecnologias que foram utilizados nesta dissertação são discutidos. A Seção 2.1 aborda os elementos fundamentais para o entendimento das VANETs. Já nas Seções 2.2 e 2.3 são apresentadas as arquiteturas de referência do NFV e NFV-MEC, ambas propostas pelo ETSI. A concepção de Fatiamento de Rede e seus princípios estão presentes na Seção 2.4.

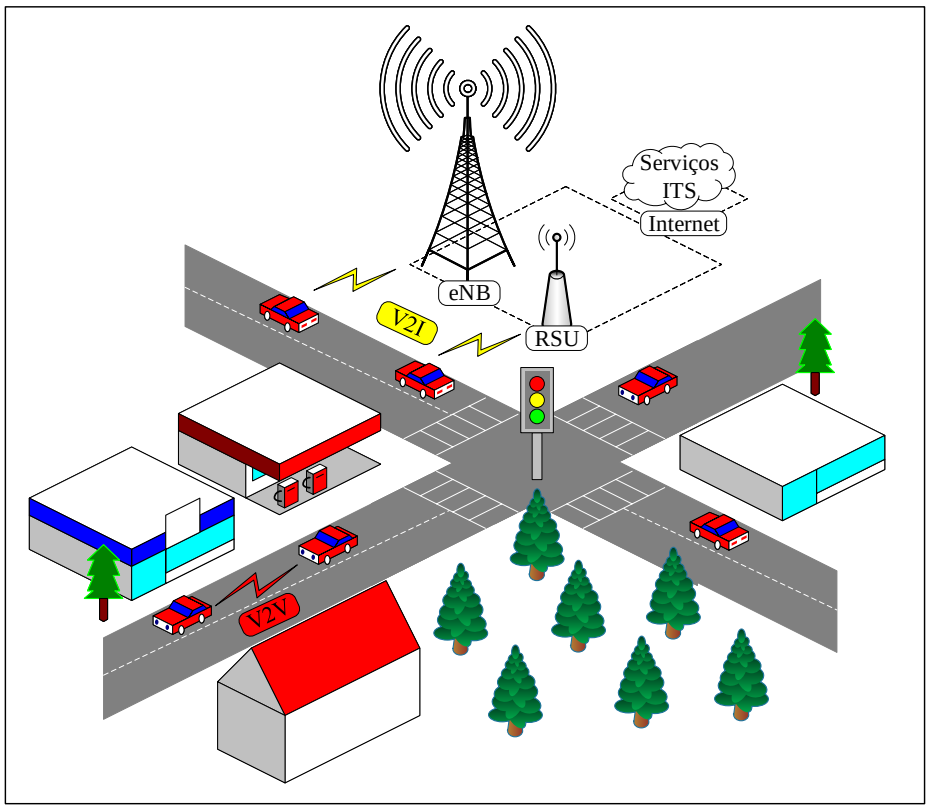
2.1 REDES VEICULARES

Nos últimos anos, foram realizados esforços significativos para o desenvolvimento das redes veiculares, visando a criação de uma subclasse das *Mobile Ad-hoc Networks* (MANETs). Vários fatores levaram a esse desenvolvimento, incluindo a ampla adoção do padrão IEEE 802.11 (redes Wi-Fi), a criação de políticas governamentais para alocação de espectro, bem como a grande aceitação por parte dos principais *players* da indústria automobilística (HARTENSTEIN; LABERTEAUX, 2008).

Apesar das MANETs apresentarem bons resultados na comunicação sem fio, elas não atendem aos requisitos necessários para fornecer a comunicação entre *Vehicle to Vehicle* (V2V) ou *Vehicle to Infrastructure* (V2I), que ocorre por meio de *Roadside Unit*s (RSUs) e estações rádio-bases em redes veiculares (Figura 1). Um exemplo disso está relacionado ao fato dos nós estarem em constante movimento causando desconexões constantes, onde a utilização de protocolos de roteamento que são comumente aplicados em MANETs não apresentam um bom desempenho em um cenário veicular. A partir dessas necessidades, foram concebidas as *Vehicular Ad-hoc Networks* (VANETs).

Uma das arquiteturas de comunicação sem fio projetadas para VANETs é chamada de *Wireless Access in Vehicular Environments* (WAVE). O WAVE foi padronizado a partir do 802.11p (*Dedicated Short Range Communications* (DSRC))/(camadas física e de enlace) e do IEEE 1609 (camadas superiores). No contexto de redes móveis celulares, o projeto 3GPP vem desenvolvendo soluções para a comunicação veicular desde o padrão *Cellular Vehicle to Everything* (C-V2X) na quarta geração. As comunicações veiculares na quinta geração, 5G-V2X, permitirão aplicações de carros autônomos por viabilizarem a redução significativa da latência na interface aérea, com valor alvo de 1ms, além do emprego de tecnologias baseadas em computação na borda e virtualização, apresentadas na sequência desse Capítulo. Além disso, estima-se que os carros autônomos do futuro terão alta capacidade de processamento e armazenamento, possuindo várias interfaces de comunicação para lidar com a heterogeneidade de cenários diversos (*e.g.,* Wi-Fi, DSRC, Bluetooth, WiMax, UMTS, LTE e 5G) (HE; ZHANG; LIANG, 2015). Na Figura 1 é ilustrado um cenário de redes veiculares usando a comunicação V2V e V2I.

Figura 1 – Cenário de Redes Veiculares



Fonte: Elaborada pela Autora

Um dos avanços mais importantes em VANETs foi a idealização da *Internet of Vehicles* (IoV), que consiste no conjunto de requisitos e tecnologias para fornecer a comunicação e o tráfego de dados de sensores e multimídia (YANG et al., 2017). Com o IoV, cada entidade da rede poderá ser conectada à Internet para compartilhamento de informações. Além disso, o IoV possibilitou a integração entre os seres humanos e veículos através da utilização de sistemas inteligentes, ultrapassando as limitações existentes em redes sem fio tradicionais (FANGCHUN et al., 2014; YANG et al., 2017).

Em virtude de todos os benefícios propiciados pelas VANETs, diversas aplicações foram desenvolvidas voltadas para o gerenciamento do tráfego, avisos de colisões, propagação de mensagens de emergência e assistência para a mudança de pista. Essas aplicações ajudarão a reduzir os engarrafamentos e o número de acidentes na via (DIETZEL et al., 2014).

Como em VANETs os veículos podem interagir uns com os outros e com a infraestrutura por meio de comunicações sem fio, o tipo de comunicação utilizada se tornou um dos principais desafios enfrentados, devido à alta mobilidade dos veículos. Os nós se deslocam em velocidades elevadas e com direções variadas, tornando a conectividade intermitente e dificultando o gerenciamento. Além disso, conforme cresce o número de veículos conectados e se expande o mercado de veículos autônomos, o tipo de comunicação *Vehicle to Everything* (V2X) se tornou ainda mais discutido e desafiador.

No contexto do 5G, as VANETs se tornaram um dos principais casos de uso para apresentar os benefícios fornecidos pelo 5G. Embora existam muitos estudos em comunicações veiculares baseadas em tecnologias de quarta geração, o 5G se tornou um potencial facilitador para combater as principais limitações dos sistemas móveis atuais, tais como: a falta de espectro, alta latência e garantia de transmissão confiável (SHAH et al., 2018).

2.2 *NETWORK FUNCTIONS VIRTUALIZATION*

Apesar da grande evolução da padronização das atuais arquiteturas de redes também chamadas de redes legadas, com o passar do tempo observou-se a inflexibilidade dos equipamentos que compõem a rede, por possuírem softwares fechados e arquiteturas proprietárias. Tais características fazem com que os equipamentos sejam disponibilizados a um alto custo, tornando ineficiente a sua utilização em novos experimentos em ambientes de estudo e produção (ROTHENBERG et al., 2010). Tal inflexibilidade também é caracterizada por fornecer pouca agilidade para implantar políticas de alto nível, isto é, caso se tenha a necessidade de alterar as configurações dos dispositivos de rede, essa ação deverá ser realizada separadamente em cada dispositivo, gerando uma alta probabilidade de falhas (KREUTZ et al., 2015). Além disso, é muito comum a falta de interoperabilidade entre os equipamentos de diferentes fabricantes, fazendo com que haja um aumento significativo nas despesas de capital (*Capital Expenditure* (CAPEX)) e despesas operacionais (*Operational Expenditure* (OPEX)) para os provedores de serviços (HERRERA; BOTERO, 2016).

Buscando soluções para contornar os problemas descritos, vem sendo estudado e aprimorado o conceito de *Network Functions Virtualization* (NFV). O desenvolvimento do NFV tem ganhado apoio da indústria e academia por ser considerado uma solução para os desafios enfrentados pelos serviços de telecomunicações (MIJUMBI et al., 2016). Tal conceito ganhou amadurecimento em 2012, através de um grupo de 13 operadoras com interesses em comum, na qual realizaram uma publicação no ETSI introduzindo novas perspectivas voltadas para a virtualização das funções de rede. Desde então, o uso de NFV tem ganhado cada vez mais popularidade.

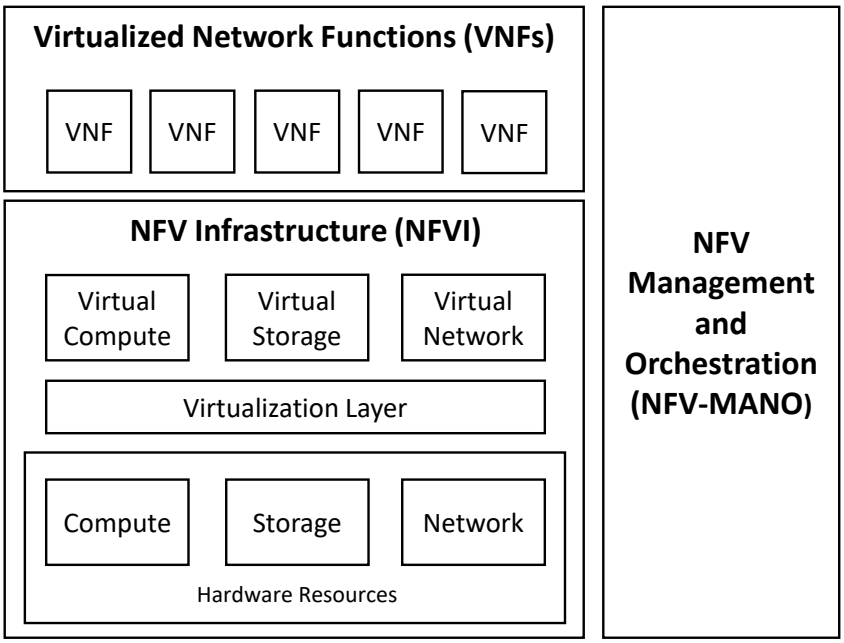
Com NFV os serviços de rede não precisarão ser implantados em dispositivos de hardware real, estes podem ser virtualizados e executados como softwares (VNF) em máquinas virtuais, além de possibilitar que os serviços sejam encadeados em uma plataforma de nuvem por meio de *Service Function Chaining*s (SFCs). Em (OPERATORS, 2012) são apresentados os principais benefícios da utilização de NFV:

- **Redução de custos:** Os custos com a aquisição de equipamentos será reduzido significativamente, além de minimizar o consumo de energia.
- **Disponibilidade:** Diversos serviços podem ser executados no mesmo substrato físico, podendo ser provisionados sob demanda.

- **Manutenção:** Facilidade no gerenciamento dos elementos que compõem a rede, flexibilizando a implantação e configuração de serviços.
- **Inovação:** Permite a experimentação de aplicações e serviços sem a necessidade de adesão a equipamentos proprietários.

Já em (ETSI, 2013), foi definido pelo ETSI um modelo arquitetural com blocos funcionais para prover a gestão e orquestração NFV (Figura 2):

Figura 2 – Arquitetura NFV Alto Nível



Fonte: ETSI (2013)

- **Funções de Rede Virtualizadas:** Atuam como serviços virtualizados que podem corresponder a serviços de redes tradicionais, tais como: *firewall*, *gateway*, servidor DNS, servidor DHCP, entre outros. De modo geral, em NFV, implementações de software ou qualquer função de rede virtualizada são chamadas de VNFs.
- **Infraestrutura NFV:** Consiste nos recursos físicos e virtuais na qual os serviços são implementados, que são: computação, armazenamento e rede. Cada conjunto de recursos podem estar distribuídos entre nós de computação distintos, mas conectados logicamente através de redes virtuais. Por meio dos *hypervisors* são criadas abstrações entre os recursos físicos e virtuais.
- **Gerenciamento e Orquestração NFV:** Este bloco funcional, também chamado de *Management and Network Orchestration* (MANO), possui como papel principal

gerenciar e orquestrar as VNFs e o *NFV Infrastructure* (NFVI). Outro papel fundamental está relacionado ao gerenciamento do ciclo de vida de toda a infraestrutura NFV.

Complementar a essa promessa, estão as *Software Defined Networking* (SDN) cuja função é comparada à virtualização das funções de redes, mas na realidade possuem papéis distintos. Enquanto SDN visa automatizar a orquestração e configuração da rede, NFV propõe automatizar e aprimorar a implantação e o controle dos recursos de rede através do uso da virtualização (ROSA et al., 2014). A virtualização tem se tornado cada vez mais proeminente nos últimos anos, e a cada dia surgem novos desafios e propostas para dinamizar a sua utilização.

2.3 *MULTI-ACCESS EDGE COMPUTING*

A cada geração das redes móveis cresce a necessidade de garantir uma melhor qualidade de experiência dos usuários de acordo com os serviços que serão consumidos. Estima-se que as cidades do futuro exigirão implantações cada vez mais distribuídas e hospedadas ainda mais próximas do usuário final, considerando a localização de quem irá produzir e consumir as aplicações e os serviços.

Com o crescimento do paradigma de Internet das Coisas e com a revolução da computação móvel em 5G, a computação em nuvem tradicional está sendo movida para a borda da rede. Os principais fatores que estão contribuindo para essa mudança, está relacionado à necessidade de redução da latência para garantir um bom desempenho de aplicações que são sensíveis a atrasos, além da necessidade de redução no consumo de energia pelos dispositivos móveis (MAO et al., 2017). Este paradigma é chamado *Multi-access Edge Computing* (MEC).

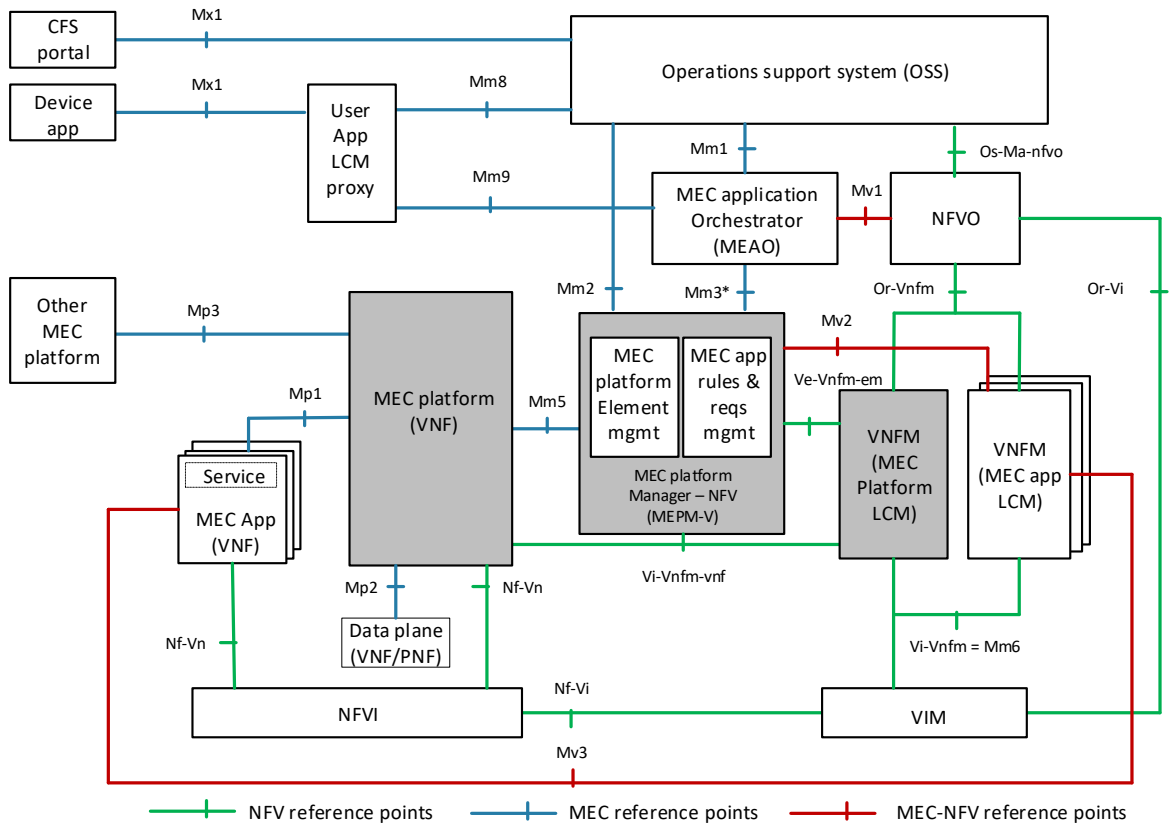
Em (PATEL et al., 2014), o ETSI abordou uma especificação sobre a utilização de MEC apontando as seguintes características:

- Premissa: Como os servidores MEC são locais, estes serão isolados de toda a rede, requisito essencial para cenários *Machine-to-Machine*. Embora
- Proximidade: Devido à proximidade com os usuários, torna-se possível capturar informações contextuais.
- Menor latência: A latência será reduzida porque os serviços passarão a ser executados na borda da rede.
- Reconhecimento local: Facilitará a identificação da localização dos usuários.
- Informações do contexto da rede: As informações sobre o estado da rede poderão ser facilmente capturadas, garantindo um melhor gerenciamento.

O MEC é fundamental para cenários de redes veiculares, principalmente quando considerada a necessidade de propagar avisos de colisões, engarrafamentos ou quaisquer situações adversas que venham a ocorrer na via. Outros dois exemplos que podem se beneficiar com a computação na borda, são aplicações de reconhecimento facial e realidade aumentada, onde parte do processamento pode ser realizado na MEC para reduzir o atraso.

A arquitetura MEC originou-se a partir da arquitetura de referência do NFV, ambas foram concebidas pelo ETSI. Enquanto o NFV cria abstrações para o provisionamento dinâmico de recursos, a MEC se utiliza dos benefícios da arquitetura NFV e foca em gerenciar as aplicações que serão consumidas na borda da rede. Na Figura 3 pode-se visualizar a estrutura projetada pelo ETSI chamada MEC-NFV, dividida em três camadas: Infraestrutura, Plataforma MEC e Aplicações MEC.

Figura 3 – Arquitetura MEC-NFV



Fonte: ETSI (2019)

O relatório publicado em (ETSI, 2020) aponta as seguintes características para a arquitetura MEC-NFV: a Plataforma MEC é implantada como VNFs para instanciar diferentes tipos de aplicações; o NFVI é gerenciado por meio de um *Virtualized Infrastructure Manager* (VIM); o *MEC Platform Manager* (MEPM) é substituído por um *MEC Platform Manager - NFV* (MEPM-V) cujo papel é instanciar o *VNF Managers* (VNFM),

que realiza o gerenciamento do ciclo de vida das VNFs; o *MEC Orchestrator* (MEO) foi substituído por um *MEC Application Orchestrator* (MEAO) para viabilizar o gerenciamento de múltiplos serviços de rede. Também é possível observar na Figura 3 onde há características específicas e/ou compartilhadas entre as arquiteturas NFV e MEC, sendo diferenciadas pelas cores das interfaces de comunicação.

2.4 *NETWORK SLICING*

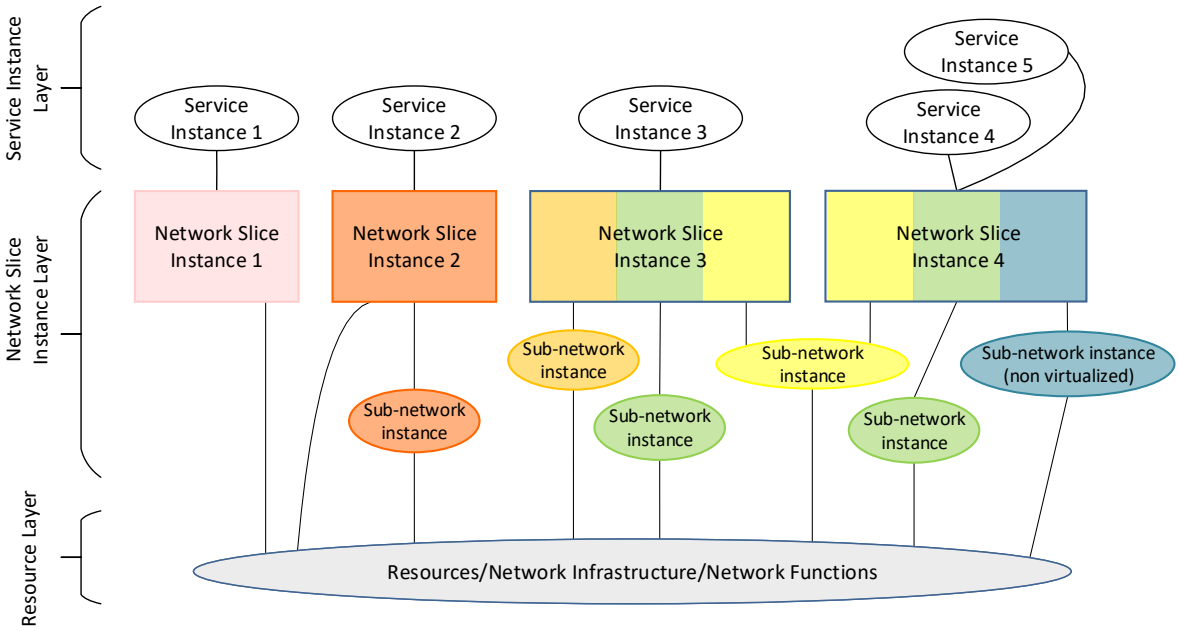
O conceito de fatiamento de rede ganhou popularidade com a ampla adesão à chamada softwarização de redes (*Network Softwarization*), conquistando ainda mais projeção com a arquitetura de referência do 5G proposta pelo 3GPP. O fatiamento de rede permite que diferentes serviços compartilhem o mesmo substrato de rede física por meio de abstrações de recursos virtuais. Originado em (ALLIANCE, 2015), sua concepção arquitetural foi dividida em três camadas principais (Figura 4):

- **Camada de serviço**: Também chamada de camada de negócios, corresponde aos serviços provisionados por provedores para os usuários finais.
- **Camada de fatia de rede**: Fornece as características exigidas por cada fatia de rede, bem como os serviços compartilhados por cada instância de serviço. Uma instância pode ser totalmente ou parcialmente provisionada através de recursos físicos e/ou lógicos.
- **Camada de recurso**: Consiste no conjunto de recursos utilizados para o provisionamento de fatias de rede, tais como recursos de computação, armazenamento e rede.

Embora o conceito de fatiamento de rede seja relativamente novo, já é possível identificar na literatura esforços significativos buscando um consenso sobre a caracterização dos seus princípios. O estudo publicado em (AFOLABI et al., 2018), que foi bastante aceito e referenciado pela comunidade, lista os seguintes princípios:

- **Automação**: Provisionamento e instanciação sob demanda, com base no SLA (*Service Level Agreement*) do usuário que irá se utilizar dos serviços que serão providos.
- **Isolamento**: Há dois tipos de isolamento: os que compartilham o mesmo substrato físico e isolados logicamente, e os que são totalmente isolados. A escolha de cada tipo irá depender de quais garantias de desempenho serão ofertadas.
- **Customização**: Utilização de recursos de forma eficiente, atendendo aos requisitos dos usuários.
- **Elasticidade**: Corresponde à capacidade de aumentar ou diminuir a quantidade de recursos de acordo com a demanda.

Figura 4 – Concepção sobre Fatiamento de Rede



Fonte: Alliance (2015) - Adaptada pela Autora

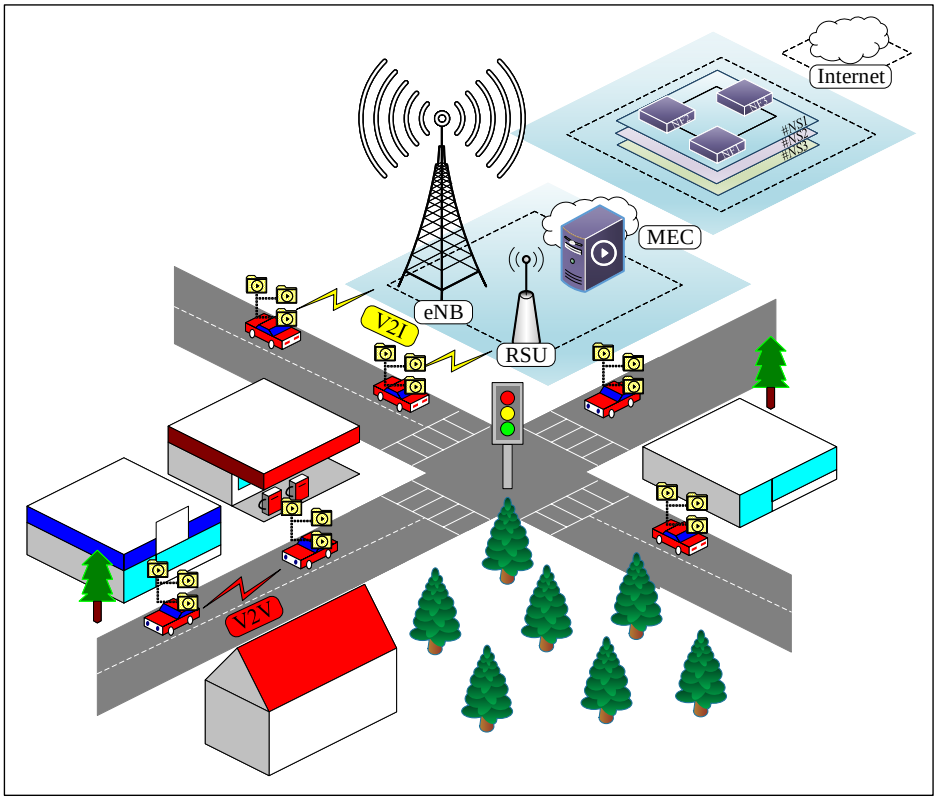
- **Programabilidade**: Consiste na modularização da arquitetura projetada para facilitar a inclusão de soluções de terceiros, como o uso de controladores SDN (*Software Defined Networking*).
- ***End-to-end***: Capacidade de fornecer o fatiamento de rede como serviço, seja considerando o EPC (*Evolved Packet Core*) virtualizando e criando cadeias de serviços das principais entidades de rede, tais como MME (*Mobility Management Entity*), HSS (*Home Subscriber Server*), PGW (*Packet Data Network Gateway*), SGW (*Serving Gateway*) e PCRF (*Policy and Charging Rules Function*), ou até mesmo criando separações lógicas até o UE (*User Equipment*) na RAN (*Radio Access Network*).
- **Abstração Hierárquica**: Abstração dos recursos físicos na qual estes podem ser compartilhados ou negociados entre diferentes fatias de rede.

3 ARQUITETURA OPENS-IOV

Este Capítulo apresenta uma visão geral da arquitetura projetada nesta dissertação, chamada OPENS-IoV, justificando as principais motivações, requisitos, detalhamento da proposta e um cenário de uso.

3.1 MOTIVAÇÃO E REQUISITOS

Figura 5 – Cenário de Uso



Fonte: Elaborada pela Autora

A quinta geração das redes móveis promete fornecer conectividade sem interrupções para pessoas e coisas, com maiores capacidades de transmissão e menores latências. Além disso, o 5G possibilitará a implantação de diferentes tipos de serviços atendendo a grandes demandas de clientes, tornando-se um potencial facilitador para a Internet das Coisas. Para os provedores de serviços, tais características permitirão trazer inovações para diversos segmentos, tais como saúde, automotivo e residencial, dentre outros. Para esse fim, o 5G promete impulsionar ainda mais a utilização do conceito de virtualização. Com a virtualização é possível abstrair as características físicas dos recursos computacionais, além de possibilitar o compartilhamento e o isolamento de acordo com a necessidade.

Muitos esforços estão sendo realizados para a criação das futuras infraestruturas de telecomunicações, onde a utilização de técnicas como SDN e NFV estão convergindo para a chamada softwarização de redes. Dentro dessa abordagem surge o conceito de fatiamento de rede com a proposta de provisionar sob demanda os elementos amplamente utilizados nas infraestruturas tradicionais, através da unificação de funções de rede (AFOLABI et al., 2018). De acordo com Afolabi et al. (2018), o conceito de fatiamento de rede se baseia nos seguintes princípios: automação, isolamento, customização, elasticidade, programabilidade, abstrações arquiteturais *end-to-end* e hierárquicas (conforme discutido na Seção 2.4).

Considerando os requisitos mencionados acima, projetamos uma arquitetura para prover o fatiamento de rede e compartilhamento de conteúdos para veículos conectados, alinhada aos requisitos previstos em 5G e utilizando MEC e NFV, por serem juntos com SDN, os principais facilitadores para projeção das arquiteturas de quinta geração. Um exemplo de cenário de uso é ilustrado na Figura 5, como uma extensão da Figura 1.

3.2 DETALHAMENTO DA PROPOSTA

O modelo implementado se baseia na divisão em camadas amplamente utilizado em diversos estudos, a qual foi detalhada em (SOENEN et al., 2017). A Figura 6 mostra a arquitetura de referência especificada pelo ETSI. Os retângulos tracejados, com os nomes de camadas, correspondem aos elementos utilizados para conceber a arquitetura proposta neste trabalho (Figura 7).

3.2.1 Camadas de Aplicação e Serviço

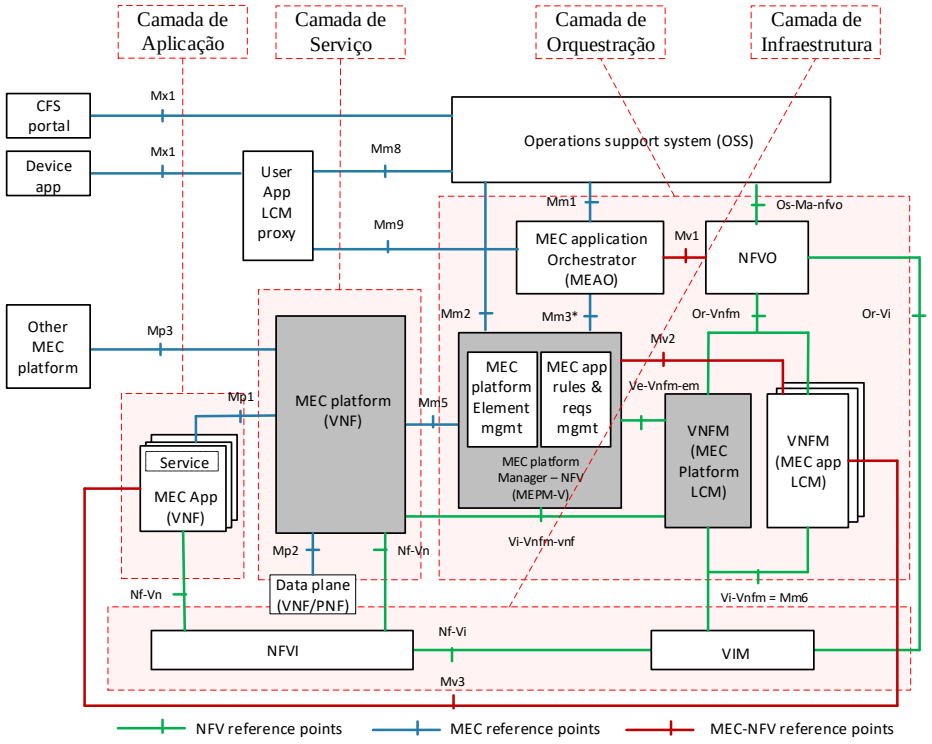
Estudos apontam que a ampla adoção do IoV aumentará a busca de serviços de informação e entretenimento multimídia (ZHOU et al., 2019). Diante disso, surge um fator crucial a ser considerado que é possibilitar uma melhor qualidade de experiência do usuário/assinante. Isso se tornou um dos principais focos com a evolução das redes móveis. Como benefícios, será possível ter uma maior largura de banda e menor latência.

Neste trabalho, criamos uma abstração da camada de aplicação por meio de um middleware integrado na camada de serviços, o que permitirá a execução de diferentes tipos de aplicações para cenários de veículos conectados. No diagrama da Figura 12 é demonstrado um fluxo de comunicação fim-a-fim, onde é possível perceber o funcionamento da aplicação implementada para a transmissão de vídeos entre os *consumers* e *producers*.

3.2.1.1 Middleware orientado a mensagens

Para gerenciar todas as solicitações de conteúdos foi projetado um middleware orientado a mensagens que possui como característica quatro elementos principais: eventos, *subscribers*, *publishers* e um servidor de eventos. Um evento pode ser uma informação a ser

Figura 6 – Abstrações utilizadas em OPENS-IoV



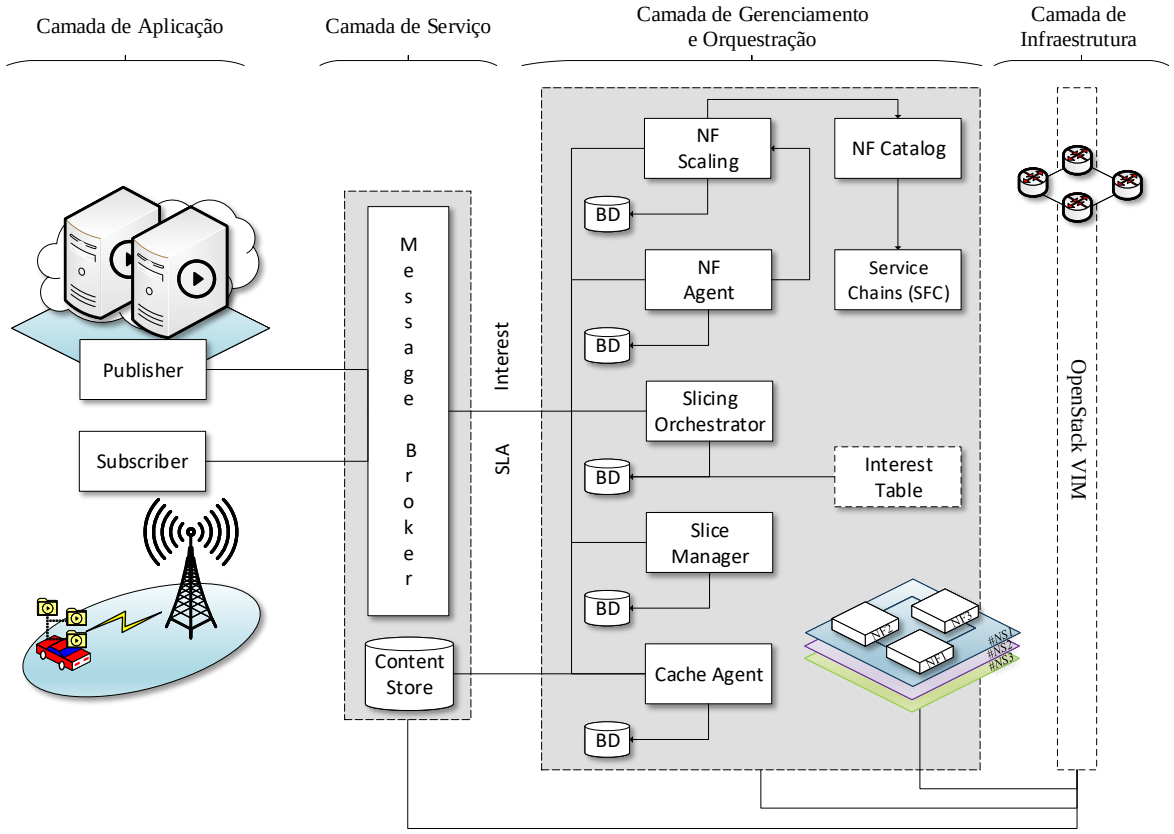
Fonte: ETSI (2019) - Adaptada pela Autora

processada ou até mesmo um conteúdo a ser fornecido, ou seja, um evento é algo de interesse de alguma entidade. Os *subscribers* registram o interesse e consomem eventos, já os *publishers* produzem os eventos que serão disponibilizados. Por outro lado, um servidor de eventos gerencia todas as subscrições e realiza o tratamento das mensagens. Adotamos a mesma abordagem em nossa proposta, os *publishers* são os produtores de conteúdo e os *subscribers* os consumidores/veículos. Para tal, seguimos os princípios da padronização *Remoting Patterns* abordado em (VÖLTER; KIRCHER; ZDUN, 2013) e reutilizamos os seguintes elementos: *Client Request Handler* e *Server Request Handler* na camada de infraestrutura, e o *Client Proxy*, *Marshaller* e o *Message Broker* na camada de distribuição, conforme descrito abaixo:

- ***Client Proxy:*** Os clientes (*producers/consumers*) interagem com o *Client Proxy* que realiza a intermediação das requisições, fornecendo assim transparência de acesso e de localização do servidor remoto.

- ***Client Request Handler:*** Para que todas requisições sejam efetuadas dos clientes para o servidor remoto é papel do *Client Request Handler* abrir e fechar as conexões de rede, bem como receber as mensagens de retorno e enviar para o cliente apropriado.

Figura 7 – Arquitetura projetada OPENS-IoV



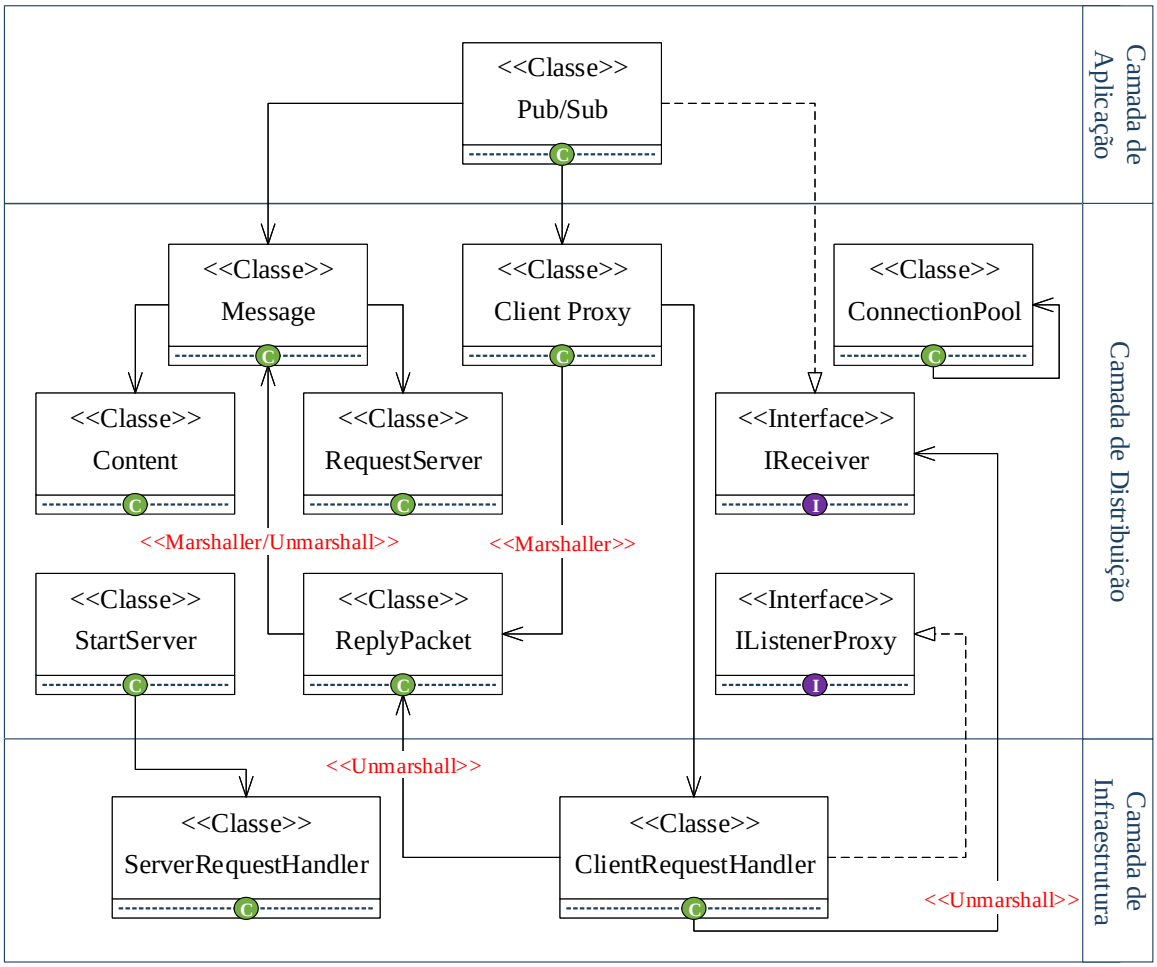
Fonte: Elaborada pela Autora

- ***Server Request Handler:*** Tem a função de receber solicitações simultâneas e efetuar o gerenciamento e estabelecimento das conexões.
- ***Marshaller:*** Todas as mensagens trafegadas na rede são serializadas em bytes antes de serem enviadas e deserializadas no recebimento. Tal processo é importante para lidar com a compatibilidade entre o cliente e o servidor. O processo de Marshaller/Unmarshall entre as entidades é demonstrado na Figura 8 na cor vermelha.
- ***Request Server:*** Recebe e trata todas as mensagens recebidas dos *producers* e *consumers*. No ato da subscrição dos *consumers*, são identificados os principais parâmetros no corpo da mensagem e armazenados em uma base de dados (e.g., ID, conteúdo, protocolo e tipo de recurso).

Além dos elementos descritos, outras entidades foram adicionadas ao middleware proposto, tais como:

- ***StartServer:*** Instancia o servidor e aguarda novas conexões.

Figura 8 – Arquitetura do Middleware



Fonte: Elaborada pela Autora

- ***Content:*** Detalha as características dos conteúdos de acordo com as necessidades da camada de aplicação, como nome, tamanho e sua localização.
- ***Message:*** Específica os parâmetros que serão passados no corpo das mensagens, tais como ID, protocolo, conteúdo, status e entre outros. Além disso, o formato das mensagens é do tipo *Reply Packets*.
- ***IListenerProxy:*** Devido à necessidade de estabelecer múltiplas requisições simultâneas, o *IListenerProxy* tem a função de implementar chamadas de retorno independentes e assíncronas.
- ***IReceiver:*** Herda os comportamentos do *IListenerProxy* e trata as mensagens de retorno de acordo com as requisições realizadas pelo solicitante.
- ***ConnectionPool:*** Em toda aplicação que faz o uso de um banco de dados, com inúmeros acessos simultâneos, o tratamento do número de conexões que serão re-

alizadas é essencial. De maneira geral, a cada consulta o servidor abre uma nova conexão com o banco de dados e a fecha em seguida, tornando ineficiente tal utilização em larga escala. Para evitar que isso aconteça e buscando reduzir a latência, um *pool* de conexões foi implementado para reaproveitar as conexões ativas.

A Figura 8 apresenta a organização de cada camada do middleware implementado e os relacionamentos entre suas entidades. É importante salientar que na camada de aplicação pode ser utilizado qualquer tipo de aplicação entre os *producers* e *consumers*, para este trabalho foi adotado a transmissão de vídeos. As abstrações do middleware estão presentes na arquitetura da Figura 7 por meio dos seguintes elementos: (i) o *Publisher* e *Subscriber* da arquitetura equivalem ao *pub/sub* presentes na arquitetura do middleware; (ii) o *Message Broker* da arquitetura corresponde as camadas de distribuição e infraestrutura do middleware.

Um *consumer* ao se subscrever passa no corpo da mensagem seus interesses, tais como o nome e o tipo do conteúdo (*Standard Definition* (SD) ou 4K), o protocolo de comunicação para o recebimento do conteúdo (TCP ou UDP) e se deseja ou não compartilhar recursos. Vale lembrar que essa abordagem foi adotada inicialmente e que será aprimorada em trabalhos futuros. Já os *producers* serão notificados quando houver alguma requisição de conteúdos sem saber quem o solicitou. Em seguida, a transmissão do vídeo para o cache é iniciada após a instanciação completa da fatia de rede correspondente, de acordo com os parâmetros já definidos, como o protocolo de comunicação e a porta para a transmissão.

3.2.1.2 Content Store

Semelhante ao conceito de NDN, as primeiras requisições para cada tipo de conteúdo/protocolo são registradas na *Interest Table*, com as informações de *Service Level Agreement* (SLA) dos *subscribers*. Durante o processo de subscrição, verifica-se se já existe o mesmo interesse registrado com status de pendente ou concluído, caso contrário, um novo interesse é registrado. Quando todo o conteúdo estiver disponível em cache, inicia-se a transmissão do vídeo para todos os *subscribers* que já expressaram o desejo de consumir tal conteúdo.

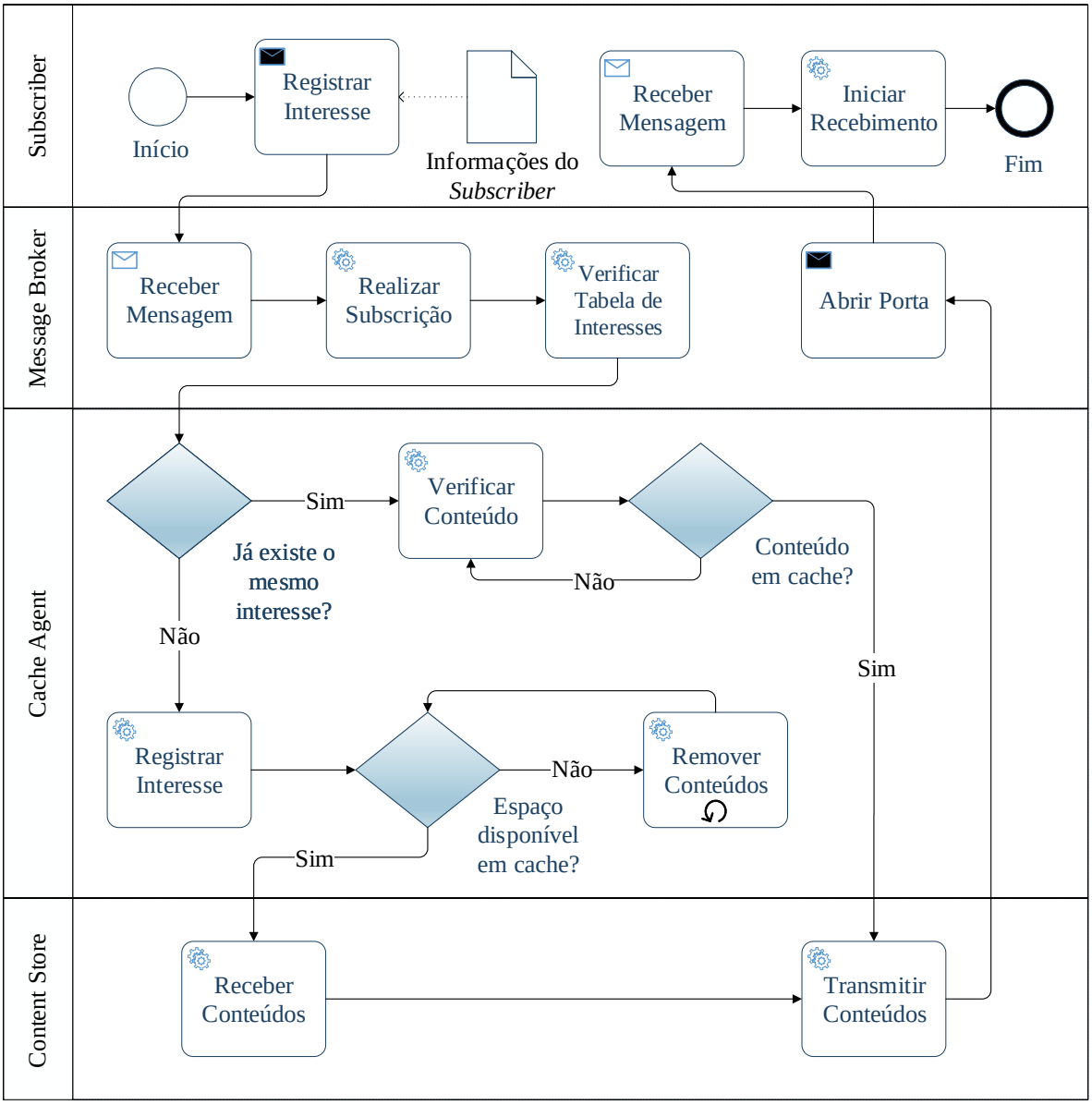
Uma ilustração sobre a estratégia de cache de conteúdos é apresentada através do fluxograma da Figura 9, com os seguintes passos:

1. O *subscriber* registra um interesse passando suas informações.

2. O *Message Broker* valida se o mesmo interesse já foi registrado, caso contrário, registra um novo interesse.

3. Antes dos *publishers* enviarem os conteúdos solicitados para o cache, uma mensagem de status com o nome do conteúdo, protocolo de comunicação e seu tamanho é encaminhada para o *Message Broker*. Este passo não está presente na Figura 9, mas é através dele que é determinado o tamanho dos conteúdos.

Figura 9 – Estratégia de Cache de Conteúdos



Fonte: Elaborada pela Autora

4. O *Cache Agent* verifica o tamanho atual do cache e se o tamanho disponível suporta o novo conteúdo. Caso não seja possível adicionar, é iniciado o processo de verificação para identificar quais os conteúdos que estão sendo menos utilizados e que não estão sendo transmitidos nos últimos minutos, consideramos o intervalo de uma hora em nossos cenários de avaliação. Se não houver nenhum conteúdo dentro da faixa especificada, o fator de decisão passa a ser apenas a taxa de requisição atual, sendo removidos do *Content Store* os conteúdos com menor taxa de requisição até que o novo conteúdo possa ser adicionado e, dessa forma, também é feito o gerenciamento do ciclo de vida dos conteúdos.

5. Depois que o conteúdo está presente no *Content Store* ocorre o processo de transmissão para todos os *subscribers* que registraram o interesse.

3.2.2 Camada de Gerenciamento e Orquestração

Na arquitetura proposta, o NFVO (NFV *Orchestrator*) é formado por cinco entidades: (i) *Database*, (ii) *Slicing Orchestrator*, (iii) *NF Agent*, (iv) *Slice Manager* e (v) *NF Scaling*. O funcionamento do algoritmo de decisão para a alocação de recursos é detalhado no Algoritmo 1. Abaixo são apresentadas as principais características de cada módulo:

- ***Database*:** Persiste todas as informações de redes, VNFs, SFCs e *consumers*, com os relacionamentos necessários e os estados de cada processo.
- ***Slicing Orchestrator*:** Identifica as requisições recebidas pelo *Message Broker* e trata de acordo com o SLA do usuário solicitante e o conteúdo requerido. Caso o conteúdo não esteja em cache, será instanciada uma fatia de rede para o usuário de acordo com o tipo de recurso e protocolo definidos. Após serem instanciadas todas as NFs e criada toda a SFC, o *producer* saberá que já é possível iniciar a transmissão do vídeo através da cadeia correspondente e de acordo com o classificador de fluxo SFC.
- ***NF Agent*:** É de responsabilidade do *NF Agent* identificar quais NFs serão atribuídas para cada fatia de rede instanciada pelo *Slicing Orchestrator*. Em cenários que o usuário solicitante não exige recursos totalmente dedicados, essa decisão é tomada com base no percentual de utilização das NFs já instanciadas. As NFs com menor utilização são sempre as primeiras candidatas consideradas no processo de alocação.
- ***NF Scaling*:** O módulo *NF Scaling* é responsável por realizar o processo de elasticidade caso as NFs selecionadas estejam acima do limiar de utilização. Para tal, considera-se a mesma capacidade computacional das NFs selecionadas e estas serão designadas para as próximas requisições. Vale lembrar que, quando se inicia alguma elasticidade, todas as demais solicitações com compartilhamento de recursos aguardam até as NFs estarem alocadas, tornando possível selecioná-las para compor novas fatias de rede. Através dessa estratégia fazemos elasticidade horizontal das NFs sob demanda.

3.2.2.1 Gerenciamento do Ciclo de Vida

O ciclo de vida das VNFs é gerenciado de forma dinâmica com base no status das fatias de rede instanciadas. A cada finalização da transmissão dos vídeos, após novos interesses, o status da fatia de rede é atualizada, possibilitando ao *Slice Manager* identificar todas as desalocações necessárias (e.g., VNFs, SFCs, redes, dentre outros). Para a desalocação,

Algoritmo 1: Algoritmo de Decisão

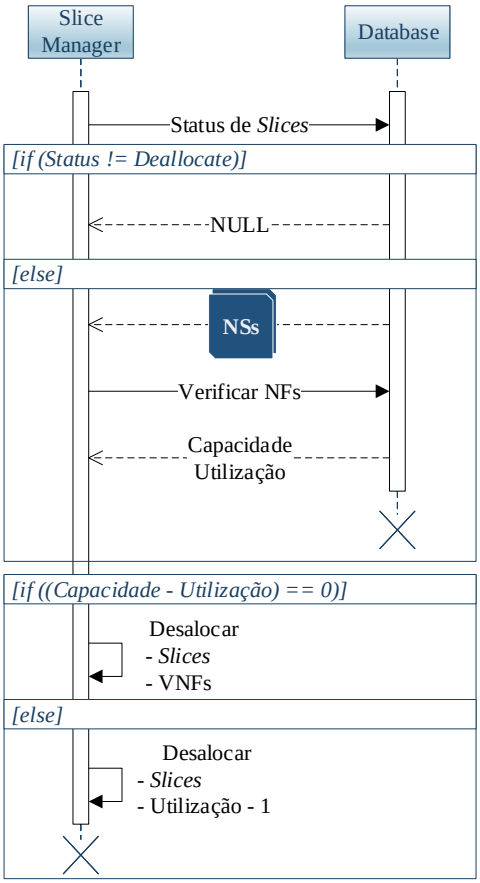
```
1  repita
2      selecione Recurso faça
3          caso "Dedicado" faça
4              Lista_NFs[] ← AlocarNFs(Recurso);
5              InstanciarSlice(Dedicado, Lista_NFs[], Protocolo, Porta);
6              Lista_NFs[] ← NULL;
7          fim caso
8          caso "Compartilhado" faça
9              RecursoAlocado ← VerificarRecurso();
10             se RecursoAlocado = True então
11                 Lista_NFs[] ← SelecionarNFs();
12                 para i ← 1 até i ≤ Lista_NFs.Tamanho() faça
13                     Utilização ← VerificarUtilização(Lista_NFs[i]);
14                     Capacidade ← VerificarCapacidade(Lista_NFs[i]);
15                     se Utilização ≥ Limiar então
16                         NF ← EscalarNF(Lista_NFs[i], Capacidade);
17                         Lista_NFs.Remover(i);
18                         Lista_NFs.Adicionar(NF);
19                     fim se
20                 fim para
21                 AtualizarUtilização(Lista_NFs[]);
22                 InstanciarSlice(Compartilhado, Lista_NFs[], Protocolo, Porta);
23                 Lista_NFs[] ← NULL;
24             senão
25                 Lista_NFs[] ← AlocarNFs(Recurso);
26                 AtualizarUtilização(Lista_NFs[]);
27                 InstanciarSlice(Compartilhado, Lista_NFs[], Protocolo, Porta);
28                 Lista_NFs[] ← NULL;
29             fim se
30         fim caso
31     fim selecione
32 até ∞;
```

inicialmente é identificado o tipo de fatia de rede (se compartilha recurso ou não) e se cada NF está atendendo a mais alguma solicitação. Essa informação é importante para saber se será removido todo mapeamento criado (e.g., rede, portas, classificadores SFC) e as NFs ou apenas reduzidas suas taxas de utilização. Um exemplo de como ocorre o processo descrito é exibido na Figura 10, com os seguintes passos:

1. O *Slice Manager* monitora os status das fatias de rede até identificar quais precisam ser desalocadas.

2. Ao identificar que uma fatia de rede precisa ser desalocada, o *Slice Manager* verifica quais VNFs compõem a fatia correspondente e seus respectivos percentuais de utilização.

Figura 10 – Gerenciamento do Ciclo de Vida



Fonte: Elaborada pela Autora

3. Durante o processo de desalocação é reduzido o percentual de utilização de cada VNF, sendo também desalocadas se não estiverem atendendo a nenhuma outra fatia de rede.

3.2.3 Camada de Infraestrutura

A plataforma de nuvem OpenStack vem ganhando popularidade entre a comunidade acadêmica desde seu surgimento em 2010. Inicialmente, foi projetada por uma iniciativa da *Rackspace Hosting*[1] e da NASA (LEDYAYEV; RICHTER, 2014), mas ao longo dos anos ganhou notoriedade formando uma das maiores comunidades de código aberto. Com o OpenStack é possível implantar nuvens públicas e privadas para fornecer serviços diversos em um ambiente de *IaaS* (*Infrastructure as a Service*), tornando possível projetar soluções em cenários de estudo ou produção.

Na arquitetura OPENS-IoV é proposto a utilização do OpenStack como VIM para a virtualização dos recursos físicos. Juntamente com o OpenStack propomos uma solução de orquestrador NFV e gerenciador de VNFs baseado no framework ETSI NFV MANO.

[1] Rackspace: <https://www.rackspace.com>

Por meio do orquestrador é estabelecido comunicações *Representational State Transfer* (REST) através da *Application Programming Interface* (API) do OpenStack implantado.

3.3 CONSIDERAÇÕES FINAIS

A Figura 11 mostra um resumo com as principais contribuições da arquitetura proposta neste trabalho, com base em cada camada implementada. Por último, no diagrama de sequência da Figura 12, apresenta-se um exemplo com o fluxo de comunicação de todas as entidades da arquitetura OPENS-IoV.

Figura 11 – Principais Contribuições

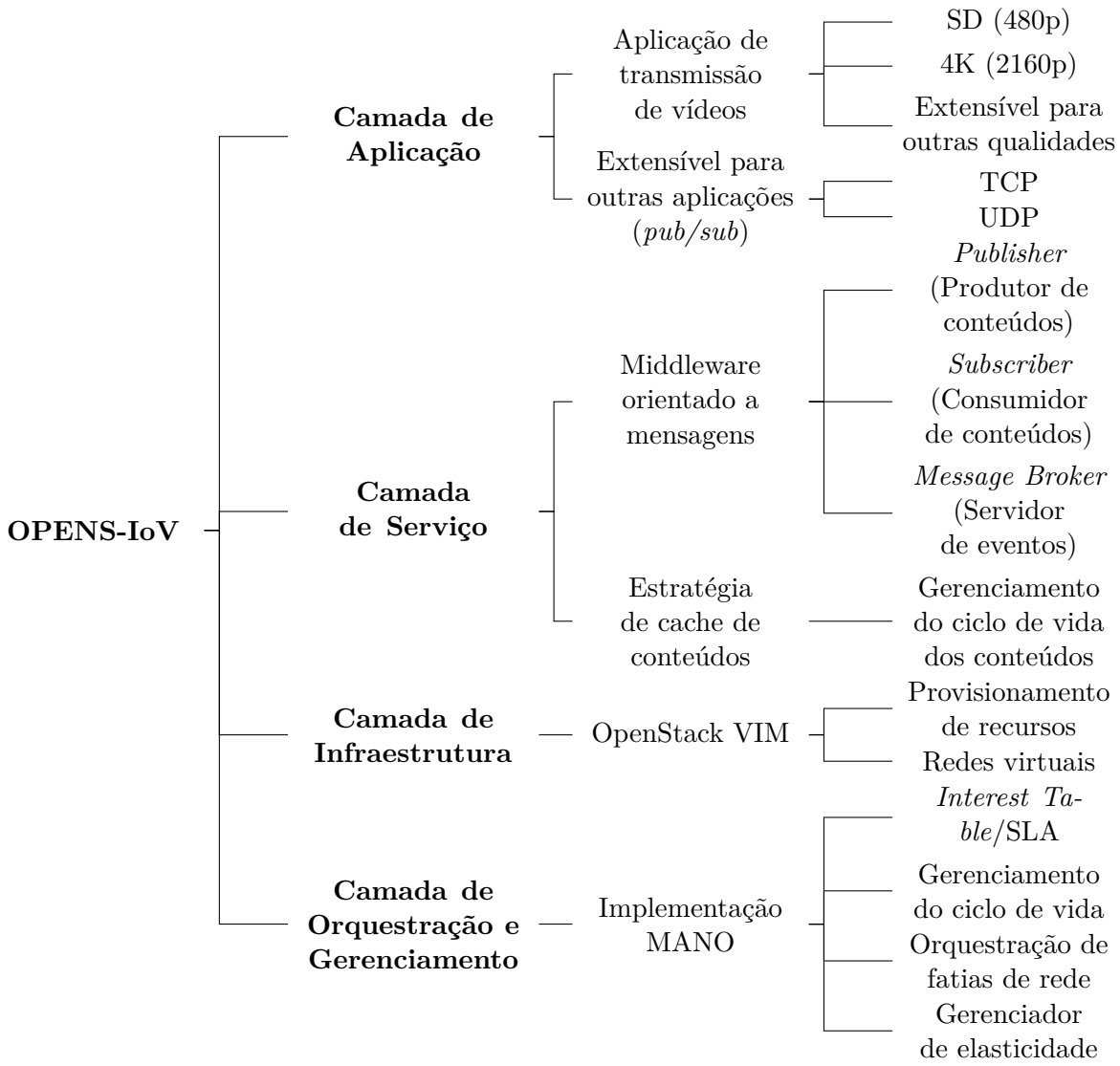
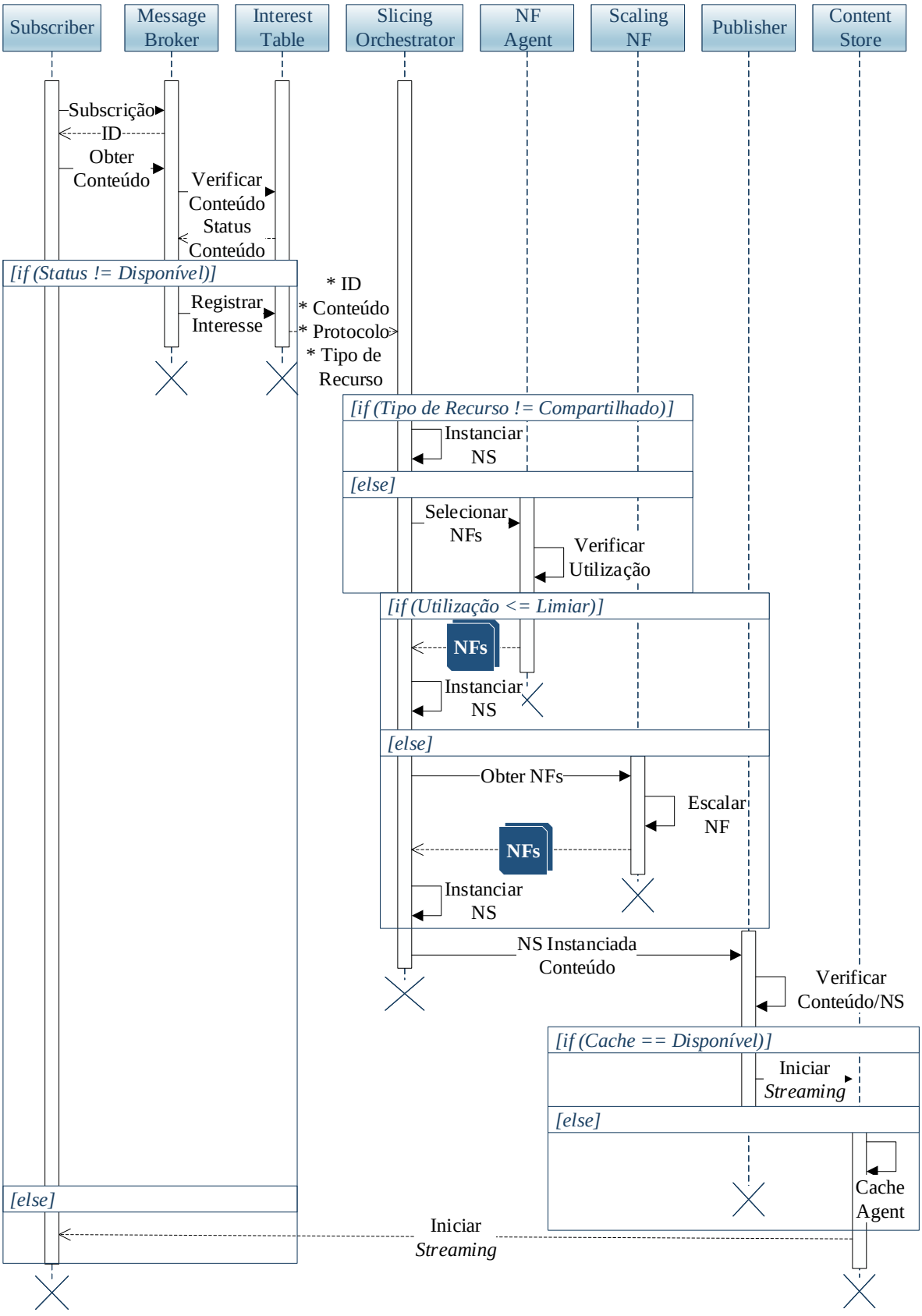


Figura 12 – Fluxo de Comunicação



Fonte: Elaborada pela Autora

4 TRABALHOS RELACIONADOS

Este Capítulo descreve os avanços recentes sobre concepções de arquiteturas ou soluções aplicáveis ao conceito de fatiamento de rede, encontrados na literatura durante a formulação do objeto de pesquisa desta dissertação. Todos os estudos discutidos neste Capítulo foram selecionados com base nos princípios de *Network Slicing* listados na Seção 2.4, além de trabalhos que exploram a sinergia entre tais princípios com os desafios previstos em redes veiculares. Uma análise comparativa é realizada nas Seções 4.1 e 4.2, elencando os principais requisitos atendidos por cada estudo.

4.1 CONCEPÇÕES ARQUITETURAIS

Em (CAMPOLO et al., 2018), os autores exploram o conceito de *Network Slicing* voltado para redes veiculares. Os autores propuseram uma arquitetura para lidar com os principais desafios em um cenário veicular, que são: alta mobilidade, conexão intermitente e o suporte a aplicações diversas com requisitos distintos. Assim como no nosso trabalho, os autores também adotaram uma arquitetura com uma divisão em camadas, sendo elas: Negócios, Orquestração, Serviço e Infraestrutura.

Diferente do trabalho de Campolo et al. (2018), optamos pela integração da camada de negócios com o orquestrador MANO, já que ambas as camadas estão totalmente interligadas. Enquanto o papel da camada de negócios é identificar os requisitos dos usuários, a camada de orquestração e gerenciamento identifica as requisições e as trata com base no tipo de serviço solicitado. De acordo com Soenen et al. (2017), a camada de negócios necessita de uma solução que seja capaz de traduzir todos os requisitos dos usuários em recursos alocados, dentro dos critérios de utilização previamente definidos. Já o TR 29.801 do 3GPP, em sua arquitetura de referência, aponta que a entidade responsável por realizar esta tradução consiste no componente CSMF (*Communication Service Management Function*). Para tal fim, consideramos que o paradigma *Named Data Networking* (NDN) poderia ser aproveitado, ainda que não seja em sua totalidade, por concluirmos que a ideia de utilizar uma tabela de interesses atenderia à necessidade mencionada em cenários diversos.

Em NDN a comunicação é baseada em pacotes de interesses e de dados, usando o modelo *publish/subscribe*. Uma tabela de interesses, também chamada de *Pending Interest Table* (PIT), registra novas solicitações quando os dados requeridos não estão disponíveis no *Content Store*. Um roteador NDN, por sua vez, sempre verifica o *Content Store* antes de encaminhar o interesse para o produtor de dados/conteúdos (ZHANG et al., 2014). Trazendo para o contexto de fatiamento de rede, uma tabela de interesses pode ser utilizada na camada de negócios ou orquestração para persistir as informações dos usuários, com os

requisitos e o tipo de serviço contratado. Neste trabalho, utilizamos tal abordagem e agregada a uma estratégia de cache de conteúdos.

Semelhante ao trabalho de Campolo et al. (2018), adicionamos funções de rede no *Core* e a Rede de Acesso com os veículos conectados na camada de serviço. Porém, este trabalho se diferencia com a inclusão de um middleware para intermediar a troca de mensagens entre os clientes e o servidor remoto. Também foi através da inclusão de um middleware orientado a mensagens que adicionamos uma camada de aplicação na arquitetura proposta, que foi idealizada com base nas seguintes características:

- O modelo *publish/subscribe* pode ser utilizado em qualquer tipo de aplicação que faça o uso de eventos.
- Permite a adequação dos tipos de parâmetros que formarão o corpo da mensagem, de acordo com os requisitos da camada de aplicação.
- Todas as mensagens são serializadas antes de serem enviadas.
- A camada de infraestrutura do middleware implementado permite o tráfego de dados usando o TCP ou UDP.
- Através do uso de um proxy é fornecida a transparência de acesso e de localização.

Por fim, na camada de orquestração e gerenciamento os autores discutem a necessidade de utilização de quatro entidades principais: *Slice Manager*, *Slicing Orchestrator*, *SDN Controller* e *Infrastructure Manager*. Tal abordagem se assemelha ao que é proposto em nosso trabalho, exceto no quesito de gerenciamento de mobilidade destacado nas características do *SDN Controller*, por não ser ainda o nosso principal objeto de estudo. Por outro lado, focamos no gerenciamento do ciclo de vida das fatias de rede, alocação de recursos, tomada de decisão com base nos requisitos dos usuários e elasticidade das instâncias de serviço.

Embora a discussão realizada por Campolo et al. (2018) esteja alinhada com os requisitos previstos na arquitetura de referência do 3GPP, nenhum resultado concreto foi apresentado. Os autores utilizaram apenas alguns experimentos como *Proof-of-Concept* (PoC) para justificar a relevância da arquitetura apresentada, fundamentando a importância do tema e atribuindo como desafio a complexidade no nível de implementação.

Outro modelo arquitetural é apresentado em (Afolabi et al., 2019), com um foco mais amplo em serviços móveis para 5G como um todo. O trabalho exibe uma proposta de arquitetura baseada nas especificações do 3GPP, subdividindo em três domínios: rede de acesso, núcleo e transporte. Apesar de apresentar uma avaliação de desempenho com um escopo reduzido, a principal contribuição do trabalho consiste em discutir os desafios de pesquisa dentro do conceito de fatiamento de rede, apontando em cada domínio uma

concepção sobre possíveis soluções. Ainda assim, nosso trabalho se assemelha com a abordagem utilizada para provisionamento de fatias de rede de forma reativa, considerando as informações dos usuários.

Em (MEDEIROS et al., 2019), apesar do foco principal do trabalho ser a estratégia de elasticidade implementada, os autores também propuseram uma nova arquitetura com três blocos principais: *Cloud-Network*, *Data Center* e WAN. O bloco *Cloud-Network* atua como orquestrador e gerenciador do ciclo de vida das fatias de rede, além de realizar elasticidade através do componente *Slice Resource Orchestrator*. Já a alocação de recursos é gerenciada pelos blocos *Data Center* e WAN, no entanto, os autores não deixaram claro a estratégia utilizada. Semelhante ao nosso trabalho, o estudo abordou a importância do uso de uma base de dados que tenha o conhecimento completo de todas as fatias de rede provisionadas. Em nossa proposta persistimos todas as informações de redes, VNFs, SFCs e *subscribers*, com os relacionamentos necessários e os estados de cada processo.

4.2 SOLUÇÕES APLICÁVEIS

Entendemos como soluções aplicáveis em *Network Slicing* os trabalhos que atendem algum dos requisitos listados na Seção 2.4. Além disso, também consideramos os trabalhos com requisitos específicos de redes veiculares, que estejam diretamente relacionados com a nossa proposta.

4.2.1 Estratégias de Alocação de Recursos

A gestão dos recursos computacionais é crucial para os provedores de serviços, que buscam frequentemente novos mecanismos de otimização. Com o advento da computação em nuvem e com ampla adesão de centros de dados, o monitoramento dos recursos computacionais passou a ser considerado uma tarefa desafiadora, como abordado inicialmente em (BARI et al., 2012). Com o NFV, a utilização de algoritmos de alocação de recursos se tornou fundamental para definir quais VNFs serão selecionadas ou provisionadas, com base nos critérios de desempenho adotados pelos provedores de serviços (HERNANDEZ-VALENCIA; IZZO; POLONSKY, 2015).

Os autores de (CELDRÁN et al., 2019) mostram uma estratégia de otimização através de três modelos complementares baseados em ontologia, como uma extensão do trabalho (CELDRÁN et al., 2018). As ontologias *Network Slice Blueprint* e *Sub-network Blueprint* realizam a tradução dos requisitos dos usuários e identificam os recursos que serão alocados, respectivamente. O conjunto de dados da primeira ontologia são os elementos de entrada para a segunda. Por outro lado, a ontologia *Network Slice Administration* atua no gerenciamento de mobilidade e avalia as métricas de desempenho.

Já em (CHIEN et al., 2019) a estratégia de alocação de recursos é baseada no *framework* implementado pelos autores, com o foco em arquiteturas MEC e foi denominado: JECRS

(*Joint Edge and Central Resource Slicer*). A estrutura do *framework* possui uma interface que identifica os requisitos dos usuários e os trata através de um orquestrador de recursos. Também possui um módulo que realiza o monitoramento das fatias de rede, permitindo a criação/atualização/exclusão de tarefas. Assim como em nossa proposta, as informações de cada mapeamento são persistidas em um banco de dados.

Assim como os autores de (CELDRÁN et al., 2019) e (CHIEN et al., 2019), em nosso trabalho também consideramos as informações dos usuários para o gerenciamento de recursos, mas com uma abordagem diferente. Cada VNF possui uma capacidade máxima de atendimento de acordo com o *flavor* associado, e seu percentual de utilização é atualizado sempre que for escolhida no processo de seleção. Realizando o gerenciamento dessa forma se avalia não só a quantidade de recursos computacionais alocados, mas também os requisitos necessários para atender cada serviço provisionado.

4.2.2 Estratégias de Elasticidade

A capacidade de tornar um sistema elástico é um requisito essencial para a criação de soluções em computação em nuvem, conforme definido em (MELL, 2011). Conceitualmente, a elasticidade consiste na capacidade de ajustar os recursos de forma dinâmica com base na carga atual, seja adicionando mais recursos computacionais às instâncias já alocadas (Elasticidade Vertical), ou até mesmo provisionando novas instâncias (Elasticidade Horizontal). Embora o conceito de elasticidade já tenha sido amplamente explorado em plataformas computacionais, dentro do paradigma de fatiamento de rede a elasticidade é considerada um dos princípios fundamentais, conforme abordado em (AFOLABI et al., 2018).

Uma estratégia de elasticidade vertical é proposta em (MEDEIROS et al., 2019). Os autores discutem a ideia de doação de recursos entre fatias de rede, atuando sempre como doadores os que estão com recursos ociosos. A utilização de memória, CPU e rede foram os critérios adotados para a tomada de decisão, considerando o SLA do usuário. A quantidade total de recursos foi considerada outro fator pelos autores, por entenderem que pequenos *players* carecem de um maior controle dos seus recursos computacionais.

A estratégia de elasticidade utilizada em nosso trabalho difere em diversos aspectos do que é proposto em (MEDEIROS et al., 2019). Adotamos o tipo horizontal com base no percentual de utilização das VNFs, fazendo elasticidade de forma reativa sempre que a VNF escolhida estiver acima do limiar previamente definido. Também consideramos a necessidade de evitar recursos ociosos através da implementação do módulo *Slice Manager*.

4.3 CONSIDERAÇÕES FINAIS

A Figura 13 lista os princípios de *Network Slicing* atendidos por cada trabalho discutido neste Capítulo, visando realizar um comparativo com a nossa proposta. É possível perce-

ber que a arquitetura proposta neste trabalho atende boa parte dos princípios de *Network Slicing*, na qual apenas o princípio programabilidade ainda não é atendido. É importante destacar que todos os trabalhos listados neste princípio possuem apenas concepções sobre o uso de SDN, como uma alternativa para prover a programabilidade da rede. Os demais aspectos arquiteturais ou requisitos específicos de redes veiculares são comparados na Tabela 1. Concluímos, portanto, que o trabalho de (CAMPOLO et al., 2018) é o que mais se aproxima do que é proposto nesta dissertação, apesar de apresentarem casos de uso diferentes.

Figura 13 – Princípios de *Network Slicing*

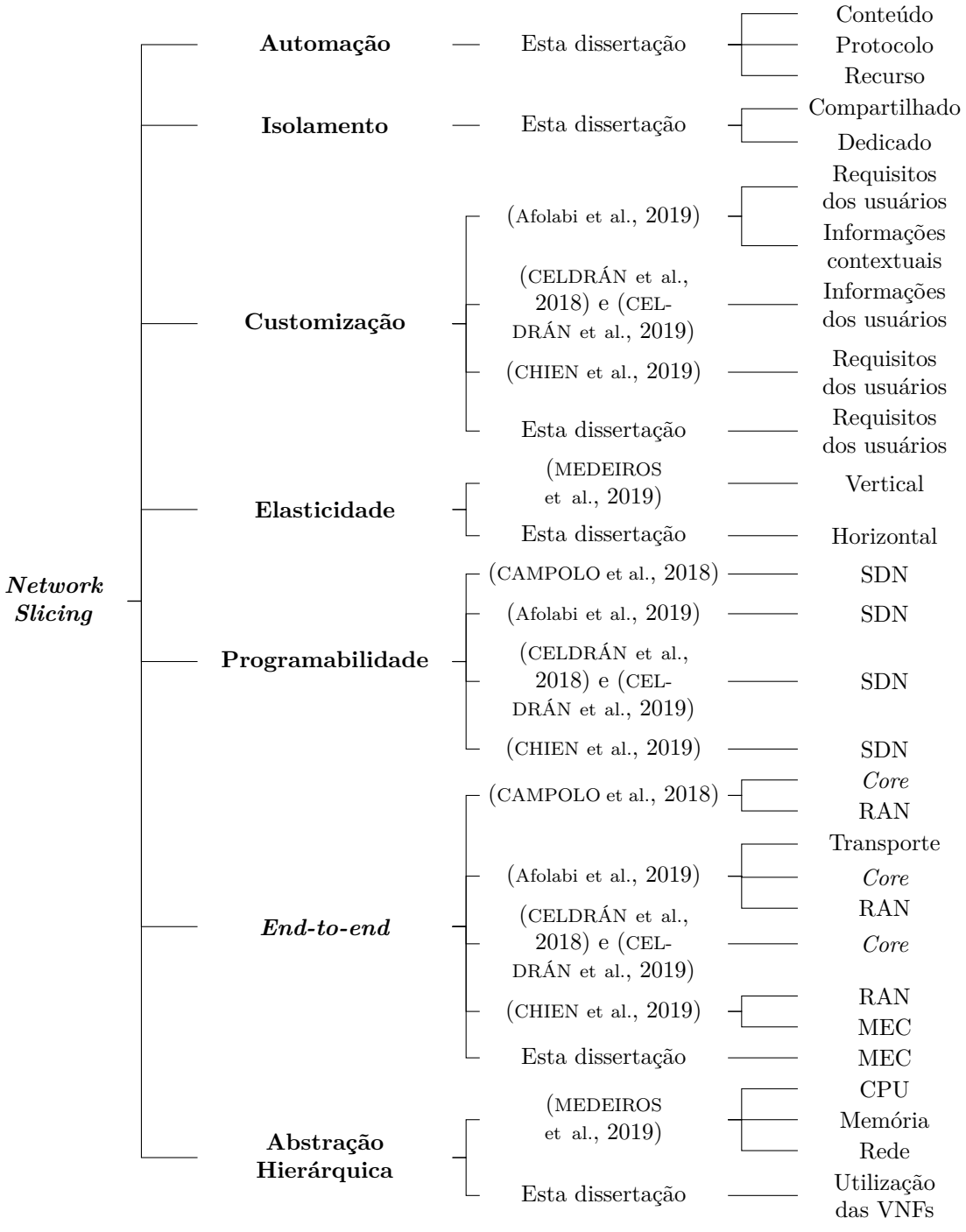


Tabela 1 – Comparativo entre os trabalhos relacionados

Trabalhos	**Objetivo Principal**	**Critérios Arquiteturais**					**Critérios de VANETs**	**Ambiente de Avaliação**
		Modelo Arquitetural	**SFC**	**Alocação de Recursos**	**Ciclo de Vida**	**Elasticidade**	**MEC**	
(CAMPOLO et al., 2018)	Propor uma arquitetura capaz de realizar o gerenciamento de mobilidade em redes veiculares	Camadas de negócios, serviços, infraestrutura e orquestração	Controladores SDN		Concepção sobre o uso de um *Slice Manager*		Aplicações de gerenciamento de mobilidade	Mininet-Wifi e Ryu
(Afolabi et al., 2019)	Realizar a orquestração de fatias de rede com requisitos heterogêneos	Divisão em RAN, *core* e transporte	Controladores SDN		Concepção sobre o uso do NSMF			OpenStack e FlexRAN
(MEDEIROS et al., 2019)	Propor uma estratégia de elasticidade vertical	*Cloud-Network*, *Data Center* e WAN			Concepção sobre o uso de um *Slice Orchestrator*	Vertical		
(CELDRÁN et al., 2018) e (CELDRÁN et al., 2019)	Otimizar a alocação de recursos e adotar políticas de gerenciamento do ciclo de vida	Arquitetura de referência do ETSI, incluindo módulos para o gerenciamento de mobilidade	Controladores SDN	Modelagem baseada em ontologia	Inclusão de políticas *intra-slices* e *inter-slices*			OpenStack, Open Baton, e OpenDaylight

(Continuação)

Trabalhos	Objetivo Principal	Critérios Arquiteturais					Critérios de VANETs	Ambiente de Avaliação
		Modelo Arquitetural	SFC	Alocação de Recursos	Ciclo de Vida	Elasticidade	MEC	
(CHIEN et al., 2019)	Propor uma solução de alocação de recursos em duas camadas	Divisão em RAN e MEC (*central office* e *transport network*)	Controladores SDN	Criação de um *framework* para o provisionamento de recursos em dois níveis			Provisionamento de recursos para a implantação de serviços 5G	Emulação de célula LTE, OpenStack, Tacker e OpenDaylight
Esta dissertação	Propor uma arquitetura que realiza o provisionamento de fatias de rede, efetua o gerenciamento do ciclo de vida e de elasticidade; Estratégia de cache de conteúdos	Camadas de aplicação, serviço, infraestrutura e orquestração	API Networking-SFC	Tipo de isolamento e percentual de utilização das VNFs	Status das fatias de rede e o tipo de isolamento	Horizontal	Estratégia de cache de conteúdos baseada no conceito de NDN	OpenStack e Mininet-Wifi

5 AVALIAÇÕES E RESULTADOS

Este Capítulo apresenta e discute os resultados de desempenho para a avaliação da proposta. A Seção 5.1 apresenta uma descrição abrangente sobre o processo completo de criação de SFCs na plataforma OpenStack, bem como os detalhes de todo ambiente de experimentação e os resultados preliminares. Os cenários avaliados com as respectivas métricas, fatores e níveis estão elencados na Seção 5.2. Por último, a Seção 5.3 apresenta os resultados obtidos com a pesquisa e algumas considerações finais.

5.1 PLANEJAMENTO E TESTES INICIAIS

Para avaliar a arquitetura proposta uma nuvem privada foi implantada com a versão *Train* do OpenStack em servidores do grupo de pesquisa IANG (*Intelligent and Advanced Networking Research Group*), no Centro de Informática da Universidade Federal de Pernambuco. Já a geração de carga foi feita usando o emulador Mininet-Wifi[1] com o simulador de mobilidade urbana SUMO[2] na versão 1.0. As configurações dos servidores utilizados para o ambiente de experimentação são apresentadas na Tabela 2.

Tabela 2 – Ambiente de Experimentação

Servidores	**Sistema**	**Memória**	**Processador**	**Disco**
OpenStack	Ubuntu 18.04.3	90 GB	2 - 8 Core Intel Xeon E5530s	300 GB
Mininet-Wifi	Ubuntu 18.04.3	32 GB	8 Single Core Intel Xeon E5530s	50 GB

Fonte: Elaborada pela Autora

A implantação do OpenStack foi do tipo *All-In-One*, isto é, com a função de *controller* e *compute* no mesmo nó. Foram configurados apenas os serviços essenciais do tipo Core, compostos pelo *Nova, Keystone, Glance, Swift, Cinder* e o *Neutron* com a API de encadeamento de funções de serviço (*networking-sfc*). Tratando-se do ambiente de simulação, utilizamos 10 carros e 6 RSUs com um mapa cobrindo parte da cidade de New York exportado através do OpenStreetMap[3].

No Mininet-Wifi é possível ter a comunicação V2V e a V2I que foi o tipo de comunicação adotada em nosso ambiente de experimentação. Também adicionamos um nó para atuar como NAT para fornecer acesso à Internet aos nós emulados. Em virtude disso, conforme cada veículo entra no ambiente de simulação verifica-se se estão associados a

[1] Mininet-Wifi: <https://intrig.dca.fee.unicamp.br/mininetwifi>
[2] SUMO: <https://sumo.dlr.de/docs/index.html>
[3] OpenStreetMap: <https://www.openstreetmap.org>

alguma antena e se estão com acesso à Internet para efetuar as requisições. Cada veículo gera inúmeras requisições durante todo o processo de simulação, utilizando sementes aleatórias com intervalos exponenciais entre cada requisição por meio da biblioteca *gsl-randist*. Buscamos explorar o máximo no número de requisições para mensurar o comportamento da nossa aplicação. Abaixo apresenta-se um exemplo de execução da aplicação dos consumidores, destacando como as sementes são geradas.

Código Fonte 5.1 – Exemplo de execução da aplicação do consumidor

```
#Gerando a primeira semente
seed=`( echo $$ ; time ps ; w ; date ) | cksum | cut -f1 -d" "`

while true
do
    # Executando app do consumidor
    sudo java -jar consumer.jar &

    tempo_final=$(gsl-randist $seed 1 exponential 0.5)

    #Alterando a semente
    seed=` expr \( $seed \* 9301 + 4929 \) % 233280 `

    sleep $tempo_final
done
```

A arquitetura inicialmente foi projetada voltada para cenários dinâmicos como o de redes veiculares usando uma estratégia de alocação de recursos reativa. Como é possível perceber na Figura 14 e com os resultados preliminares, o *setup* de provisionamento inicial das VNFs pode impactar no tempo de resposta de um determinado serviço. Conforme existe a necessidade de aumentar o número de VNFs e de acordo com o tipo de serviço que será instanciado, esse tempo poderá crescer consideravelmente.

Vale lembrar que grandes *players* do mercado de *cloud* adotam estratégias de alocação proativa buscando reduzir o tempo de resposta dos seus serviços. Considerando esse fator e com as questões descritas, passamos a utilizar a alocação proativa como uma nova abordagem para melhorar o desempenho da solução.

Mesmo adotando também a alocação proativa, ainda assim, o tempo de resposta foi relativamente alto, devido ao enfileiramento das requisições ainda não atendidas pelo produtor de conteúdos. Analisando a situação, podemos considerar duas abordagens para a redução da latência: aumentar o número de produtores de conteúdos para reduzir o enfileiramento ou criar uma estratégia de cache de conteúdos. Buscando aproveitar os benefícios proporcionados pela evolução da computação móvel e aprimorados com o conceito de MEC, bem como visando um diferencial em termos de eficácia para a arquitetura proposta neste trabalho, optou-se pela criação de uma estratégia de cache de conteúdos.

Para mensurar o tempo para alocação de recursos, criação de SFCs e a definição do roteamento entre as VNFs na plataforma OpenStack, a Figura 14 exibe os valores médios obtidos no ambiente de experimentação deste trabalho, considerando os recursos

apresentados na Tabela 2. Em todos os cenários com alocação reativa, esse tempo de provisionamento inicial está inserido no tempo de resposta. Os elementos envolvidos na criação de SFCs são enumerados a seguir. Esse itens também estão relacionados aos passos necessários para a criação de SFCs no Openstack.

1. **Instanciação de VNFs:** As VNFs são funções de rede virtuais cujo objetivo é prover serviços sob demanda através do uso da virtualização. A Figura 14 apresenta o tempo de *deploy* de uma VNF usando virtualização total, considerando uma imagem com o Ubuntu Server 18.04 LTS do repositório Ubuntu *Cloud Images*, um *flavor* com 2 vCPUs, 256 MB de RAM e 10 GB de disco.

2. **Criação de Redes Virtuais:** Possibilita a criação de redes virtuais e fornece a conectividade entre as VNFs de acordo com o tipo de rede utilizada, seja de camada 2 ou de camada 3. O tipo de rede a ser escolhida depende da localização das VNFs, se estão alocadas no mesmo servidor ou até mesmo em diferentes servidores. Em nosso ambiente de experimentação adotamos o tipo VXLAN com o Open vSwitch.

3. ***Ports*:** Cada VNF deve possuir um par de portas/interfaces para cada SFC associada, sendo elas de origem e outra de destino. O agente DHCP do OpenStack (*neutron-dhcp-agent*) se encarrega de atribuir os endereços IPs para as interfaces com base na rede virtual designada.

4. ***Routing*:** Para a criação dos encadeamentos lógicos entre as VNFs é preciso configurar as tabelas de roteamento e atribuir o gateway padrão para cada interface de rede.

5. ***Port Pairs*:** Cada par de portas possui um identificador que será utilizado para a formação das SFCs, e além disso, é especificado qual porta será a de entrada e a de saída durante sua criação.

6. ***Port Pair Group*:** Consiste na junção dos pares de portas formando grupos lógicos de VNFs.

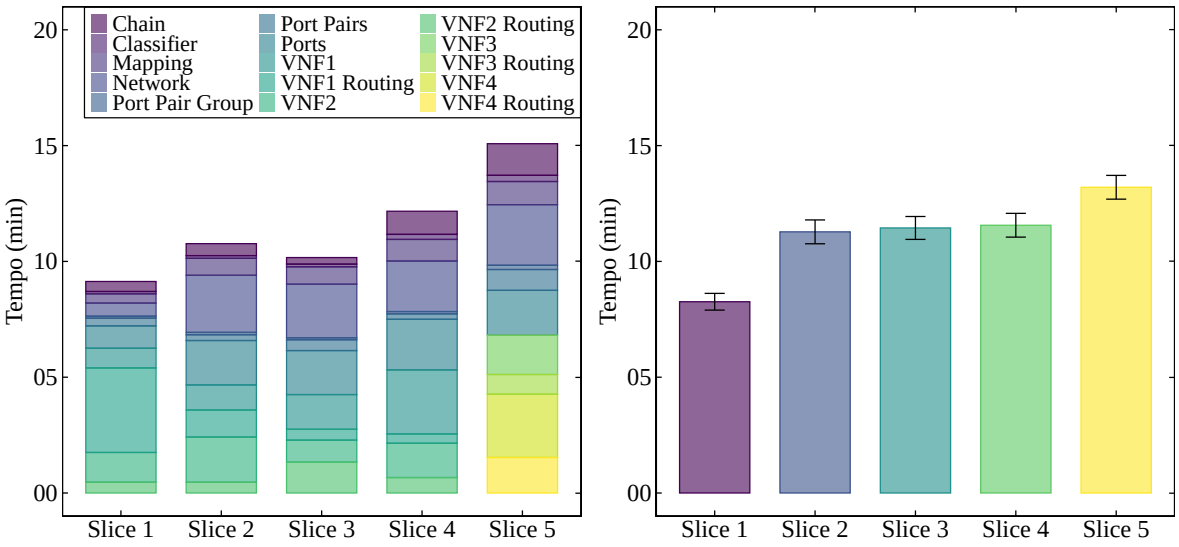
7. ***Classifier*:** O classificador de fluxo específica a versão do protocolo IP e o protocolo da camada de transporte que serão utilizados, a faixa de portas da camada de aplicação, além dos extremos com os endereços e portas de origem e de destino.

8. ***Chain*:** Por fim, após a criação e a configuração de todos os passos anteriores é possível criar uma SFC, especificando o classificador de fluxo e o grupo de pares de portas. Também é possível atribuir outros parâmetros como o tipo de fluxo, se será simétrico ou assimétrico.

Os tempos de cada passo descrito anteriormente é indicado através da criação de cinco fatias de rede, cada uma com duas VNFs, de um total de quatro VNFs para todo o sistema,

na Figura 14 (gráfico à esquerda). Por outro lado, no gráfico à direita, temos as médias que foram obtidas com o provisionamento total, por meio de 30 amostras com as mesmas características do gráfico à esquerda. Também é importante destacar que, os resultados ilustrados na Figura 14, foram gerados a partir de cenários com compartilhamento de recursos e com a estratégia de elasticidade. Tais características de cenários são detalhadas nas próximas Seções.

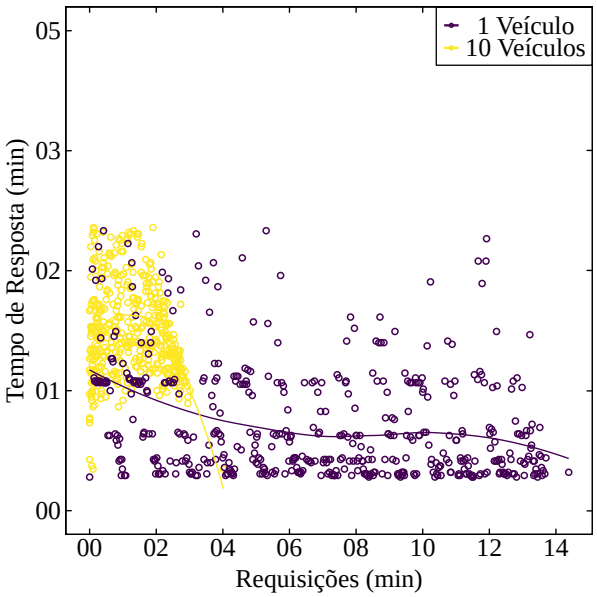
Figura 14 – Tempo de Alocação/Provisionamento



Fonte: Elaborada pela Autora

Com os resultados preliminares, também foi possível perceber que o número de veículos emulados gerando carga simultaneamente causa um impacto considerável que reflete no tempo de resposta, já que cada requisição corresponde a uma nova thread no servidor onde os nós foram emulados, compartilhando então CPU, memória e o link de transmissão. Por esse motivo, foram utilizados apenas 10 veículos. Para demonstrar esta observação, a Figura 15 apresenta um exemplo comparativo do tempo de resposta com 1 e 10 veículos emulados. Esta avaliação representa os tempos de resposta obtidos em uma amostra, cujo cenário avaliado possui as seguintes características: alocação proativa, uso de cache e transmissão de conteúdos em SD por meio do protocolo TCP. O resultado obtido nesta avaliação aponta que utilizando apenas um veículo a quantidade de requisições atendidas pelo tempo de avaliação é maior. Também é possível perceber que a linha de tendência do tempo de resposta é menor no cenário com apenas um veículo.

Figura 15 – Tempo de Resposta - Número de Veículos



Fonte: Elaborada pela Autora

5.2 PROJETO DOS EXPERIMENTOS

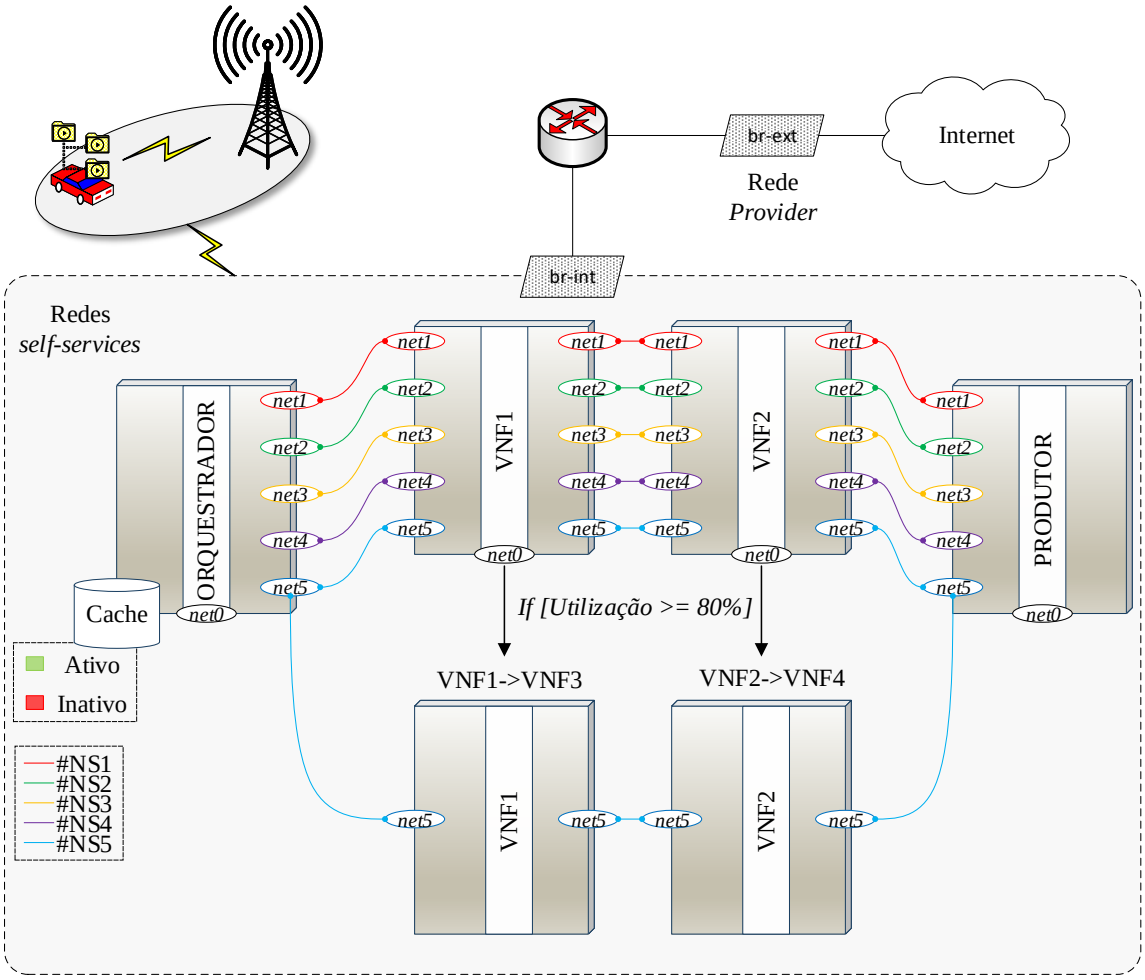
Diversos experimentos preliminares foram realizados visando mensurar a quantidade de requisições que uma VNF pode atender com recursos mínimos, com a finalidade de definir sua capacidade máxima. Para tal, foram avaliadas utilizações de CPU e de memória aceitáveis de acordo com o número de requisições, durante todo o ciclo de vida das VNFs. Foi possível identificar que uma VNF com 256 MB de RAM, 2 vCPUs e 10 GB de disco pode atender até 5 requisições e, atribuímos portanto, um limiar máximo de 80% (4 requisições) para que seja possível ocorrer elasticidade.

A Figura 16 ilustra todas as variações nos cenários de avaliação, que são: (i) Sem Elasticidade e sem Cache (SESC), (ii) Com Elasticidade e sem Cache (CESC), (iii) Sem Elasticidade e com Cache (SECC) e (iv) Com Elasticidade e com Cache (CECC). Para cada fatia de rede é criada uma rede do tipo *self-service*[4] como rede privada, estas são interligadas a um roteador virtual criando assim uma conexão entre as redes internas com a rede externa (*provider*). Na Figura 16 cada rede virtual representa a fatia de rede à qual as VNFs estão associadas, exceto a *net0* que consiste na rede de gerenciamento.

O status do *Content Store* representado na Figura 16 indica seus tipos de variações nos cenários avaliados, sendo eles ativo ou inativo. Dessa forma, o processo de registrar um novo interesse na *Interest Table* sempre ocorrerá caso o status seja inativo, e o *Publisher* enviará um novo conteúdo a cada requisição. Já o uso da elasticidade é demonstrado através das VNFs 3 e 4, estas são provisionadas quando as VNFs 1 e 2 estão acima do limiar de utilização definido.

[4] OpenStack: <https://docs.openstack.org/neutron/train/admin/deploy-ovs-selfservice.html>

Figura 16 – Ambiente de Experimentação



Fonte: Elaborada pela Autora

Antes de avaliar os resultados obtidos é importante entender o funcionamento do fluxo de comunicação de todas as entidades envolvidas em cada ambiente de experimentação. A Figura 12 ilustra o processo de requisição de conteúdos pelos *subscribers*, assumindo que já estejam dentro do *range* de sinal e associados a alguma antena. Para a avaliação, cada tipo de conteúdo foi replicado cinco vezes (com diferentes nomes) e cada um corresponde a fatias de rede distintas. A aplicação criada seleciona a qualidade do vídeo e obtém um nome de conteúdo da lista de conteúdos replicados, tornando assim a quantidade de requisições por fatia de rede aleatória.

O fluxo de comunicação apresentado na Figura 12 considera um cenário totalmente reativo, onde a alocação de recursos e criação de fatias de rede ocorrem de forma dinâmica. Caso as fatias de rede já estejam alocados, o processo de transmissão é iniciado logo após a subscrição.

Com o intuito de reduzir o escopo de avaliação, em todos os cenários avaliados foi utilizado apenas um produtor de conteúdo transmitindo vídeos usando o TCP ou o UDP,

conforme o tipo de protocolo requerido pelos consumidores. O vídeo escolhido para ser transmitido possui 15 segundos de duração e foi utilizado em duas qualidades, em SD (480p - 1,3 MB) e em 4K (2160p - 10,9 MB).

A Tabela 3 apresenta um sumário dos experimentos, detalhando as métricas de avaliação e os fatores com seus níveis que foram considerados e analisados em cada cenário. A alocação proativa não será avaliada para todos os fatores e níveis compostos no sumário dos experimentos, também não será utilizado o tipo de recurso dedicado para minimizar o escopo de avaliação.

Tabela 3 – Sumário dos Experimentos

Fatores	Tráfego	Recurso	Alocação	Cache	Elasticidade	Qualidade
Níveis	TCP UDP	Dedicado * Compartilhado	Reativa Proativa	Sim Não	Sim Não	SD (480p) 4K (2160p)
Projeto	Cada experimento foi repetido 30 vezes, por 20 minutos cada, para a combinação de níveis dos fatores, exceto para a alocação proativa e o tipo de recurso dedicado					
Métricas	Utilização de CPU, Utilização de Memória e Tempo de Resposta					

Nota: As células com os níveis destacados com um asterisco (*) não serão avaliadas, mas podem ser considerados em trabalhos futuros.

Fonte: Elaborada pela Autora

5.3 RESULTADOS

Para cada avaliação realizada buscamos medir a utilização de CPU, utilização de memória e o tempo de resposta. Para cada métrica avaliada os resultados obtidos são apresentados de duas formas: o comportamento individual de apenas uma amostra e o valor médio de todas as amostras com seus respectivos intervalos de confiança, utilizando um nível de confiança de 95%.

As subseções a seguir exibem os gráficos gerados e uma análise dos resultados obtidos. Seguem abaixo algumas informações importantes para facilitar o entendimento dos gráficos:

- Utilização de CPU e Memória
  - Nos gráficos de CPU e Memória, com uma amostra, são apresentados os valores individuais de utilização das VNFs com os instantes iniciais e finais de cada SFC, que também expressa seus ciclos de vida quando o tipo de alocação é reativa.

  - Nos gráficos com a média das amostras, as utilizações são agregadas pelo tempo ou de acordo com a quantidade de fatias de rede compartilhadas por cada VNF.

- Tempo de Resposta
  - Tratando-se dos gráficos com os tempos de resposta, foi calculado o percentil 90 para todos os resultados e a única diferença consiste em como são exibidos. Com apenas uma amostra é exposto os valores que correspondem a 90% do que foi coletado, removendo assim os *outliers*. Já nos gráficos com a média das amostras, são indicados apenas os limites obtidos com o percentil 90, através de linhas tracejadas na horizontal.
  - Em gráficos com 30 amostras, cada ponto representa a média obtida com os respectivos intervalos de confiança, calculados a cada 10 segundos.
  - As linhas de tendência são representadas de acordo com a cor da fatia de rede correspondente.
  - Os intervalos de confiança são expostos com a cor de fundo suavizada.
  - As linhas pontilhadas na vertical representam os instantes que os conteúdos ficam disponíveis no cache.

5.3.1 Alocação Reativa

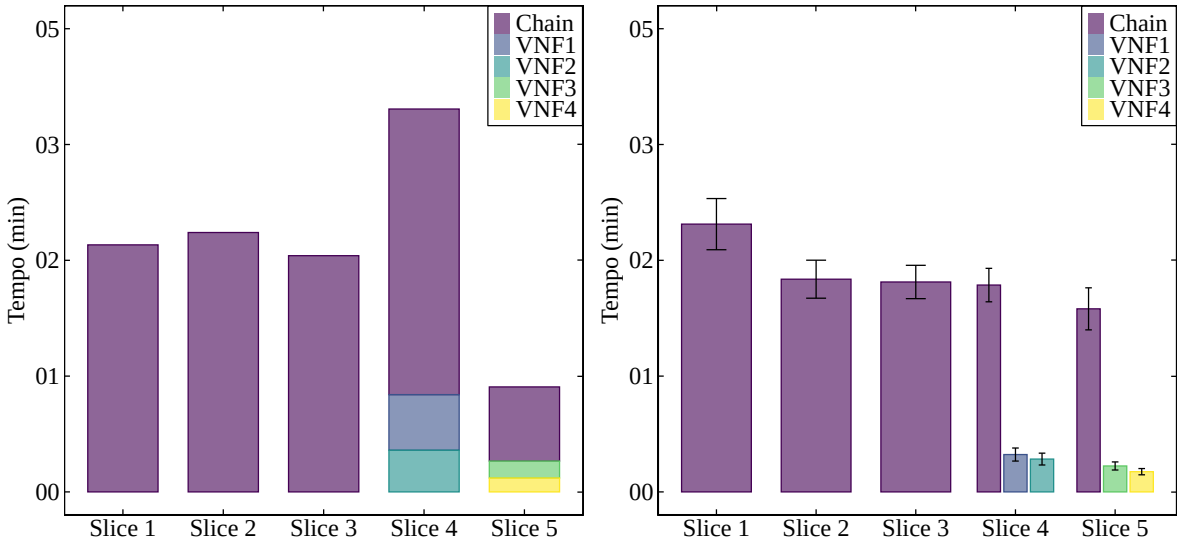
A alocação reativa consiste no provisionamento dinâmico dos recursos, conforme o orquestrador recebe novas requisições. Em cenários com este tipo de alocação o ciclo de vida das VNFs é gerenciado através da implementação do módulo *Slice Manager*, que é acionado durante o início de cada avaliação. Na Figura 17 é apresentado o tempo que o *Slice Manager* leva para desalocação de VNFs e SFCs (ilustrado na cor roxa) em um ambiente com compartilhamento de recursos e, por esse motivo, as VNFs são desalocadas apenas quando não houver mais fatias de rede associadas. Logo, como as VNFs 3 e 4 fazem parte apenas da fatia 5 e foram geradas através do processo de elasticidade, o tempo de desalocação da fatia 5 sempre será menor.

O processo de desalocação apresentado acima traz impactos em alguns dos resultados descritos nas próximas Seções e, por esse motivo, ele deve ser considerado em todos os resultados com o tipo de alocação reativa.

5.3.1.1 Sem Elasticidade e sem Cache

De maneira geral, a utilização de CPU apresentou maiores valores em todos os resultados se comparada com a utilização de memória, já que não há aplicações consideradas ”pesadas” operando nas VNFs. Também foi possível perceber que as maiores utilizações, em sua maioria, foram causados pelos *setups* iniciais das VNFs, sendo identificado nos

Figura 17 – Tempo de Desalocação



Fonte: Elaborada pela Autora

gráficos nos instantes iniciais ou quando há apenas uma fatia de rede instanciada, já que também significa que outras quatro fatias de rede ainda estão sendo provisionados simultaneamente.

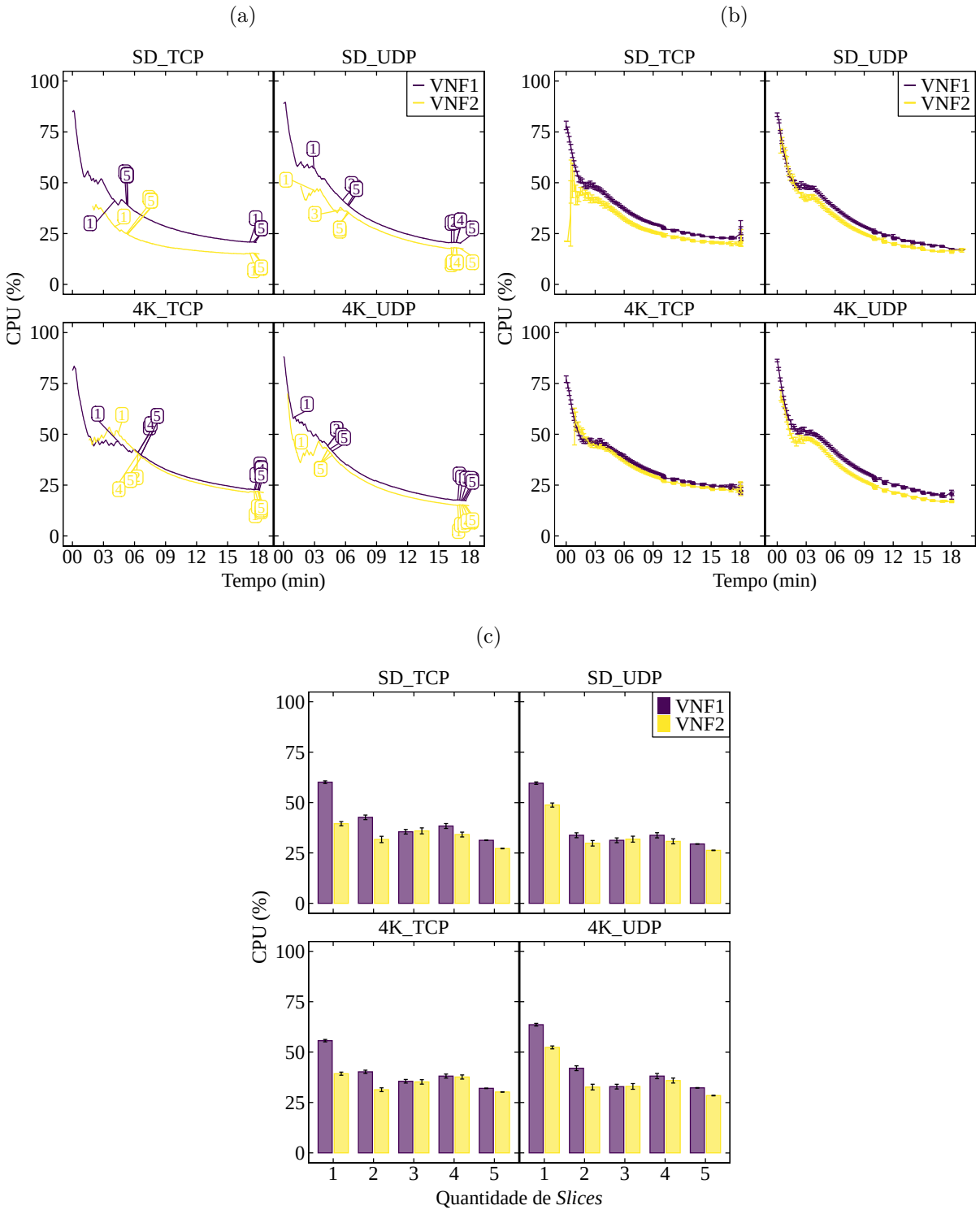
Todos os resultados de utilização de CPU e memória obtiveram os limites inferiores e superiores próximos à média. Em experimentos SESC, a maior média atingida de CPU foi de 86,27% com uma transmissão em 4K por meio do protocolo UDP (Figura 18b). Tal valor representa 10,64% a mais se equiparado com a mesma qualidade do vídeo sendo transmitido sobre o TCP (Figura 18b), ambos percentuais foram obtidos pela VNF1. Já as menores utilizações de CPU vieram da VNF2, sendo o menor valor 15,81% correspondente aos resultados expostos na Figura 18b, onde também representa 18,46% a menos em relação à Figura 18b.

Diferente dos resultados de utilização de CPU, a maior utilização de memória ocorreu com a transmissão em SD usando o protocolo UDP (Figura 19b), com a média de 43,38%. Todos os resultados obtidos em experimentos SESC mostram no máximo uma média de 40,96% de utilização de memória. Os valores apresentados correspondem aos gráficos de utilização em função do tempo, visto que os gráficos em função do número de fatias de rede apresentam percentuais de utilização inferiores e com maior variação dos limites em relação às médias.

Os tempos de resposta da Figura 19 refletiu em valores esperados, por tais motivos:

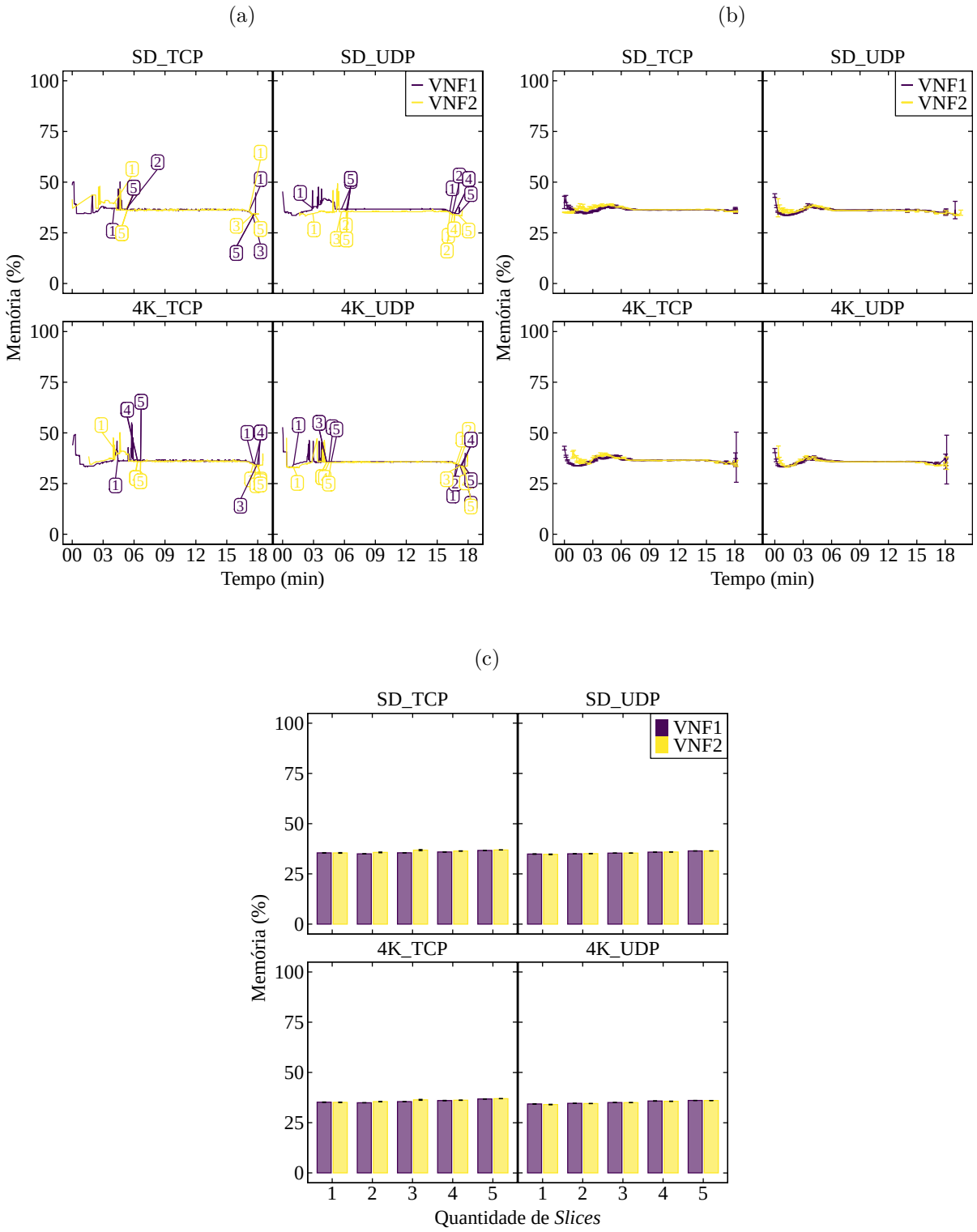
- O NS1 apresentou melhores resultados se comparado com os demais, já que é o primeiro a ser alocado.
- Enquanto as fatias de rede são provisionadas as requisições são enfileiradas para

Figura 18 – Utilização de CPU - Cenário SESC



serem atendidas após cada finalização. Portanto, o tempo máximo de aproximadamente 16 minutos que foi obtido na maioria dos resultados pelas fatias de rede NS2 ao NS5, corresponde ao período de provisionamento agregado ao tempo de atendimento.

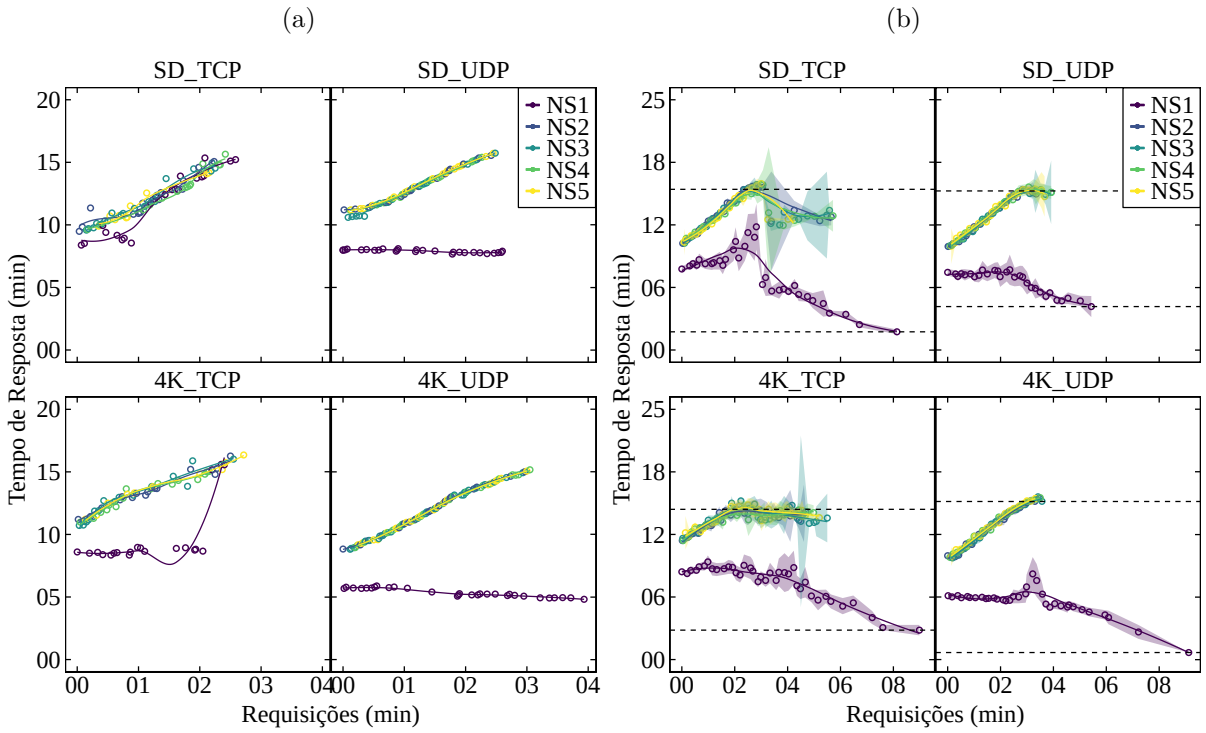
Figura 19 – Utilização de Memória - Cenário SESC



- Como já abordado nas Seções anteriores, apesar de 16 minutos ser relativamente alto também está relacionado ao fato da taxa de transmissão do link ser compartilhada, já que se trata de um ambiente de avaliação emulado em um único *host* físico.

Uma forma de tentar responder tais questionamentos é considerá-los em ambientes

Figura 19 – Tempo de Resposta - Cenário SESC



reais. Em cenários reais temos de um lado o assinante e do outro o fornecedor de serviços, e as questões levantadas são consideradas de acordo com o que foi contratado por cada assinante, se atendem ou não suas necessidades. Por esse motivo, serão apresentadas diversas abordagens avaliadas com ou sem o uso da elasticidade. Na próxima Seção são apresentados quais os ganhos e perdas na adesão da elasticidade horizontal, que é realizada através do módulo *NF Scaling* e que foi adicionado à arquitetura proposta neste trabalho. Realizar elasticidade vertical pode ser um trabalho futuro a ser implementado e investigado com o módulo *NF Scaling*, utilizando informações contextuais das VNFs por exemplo.

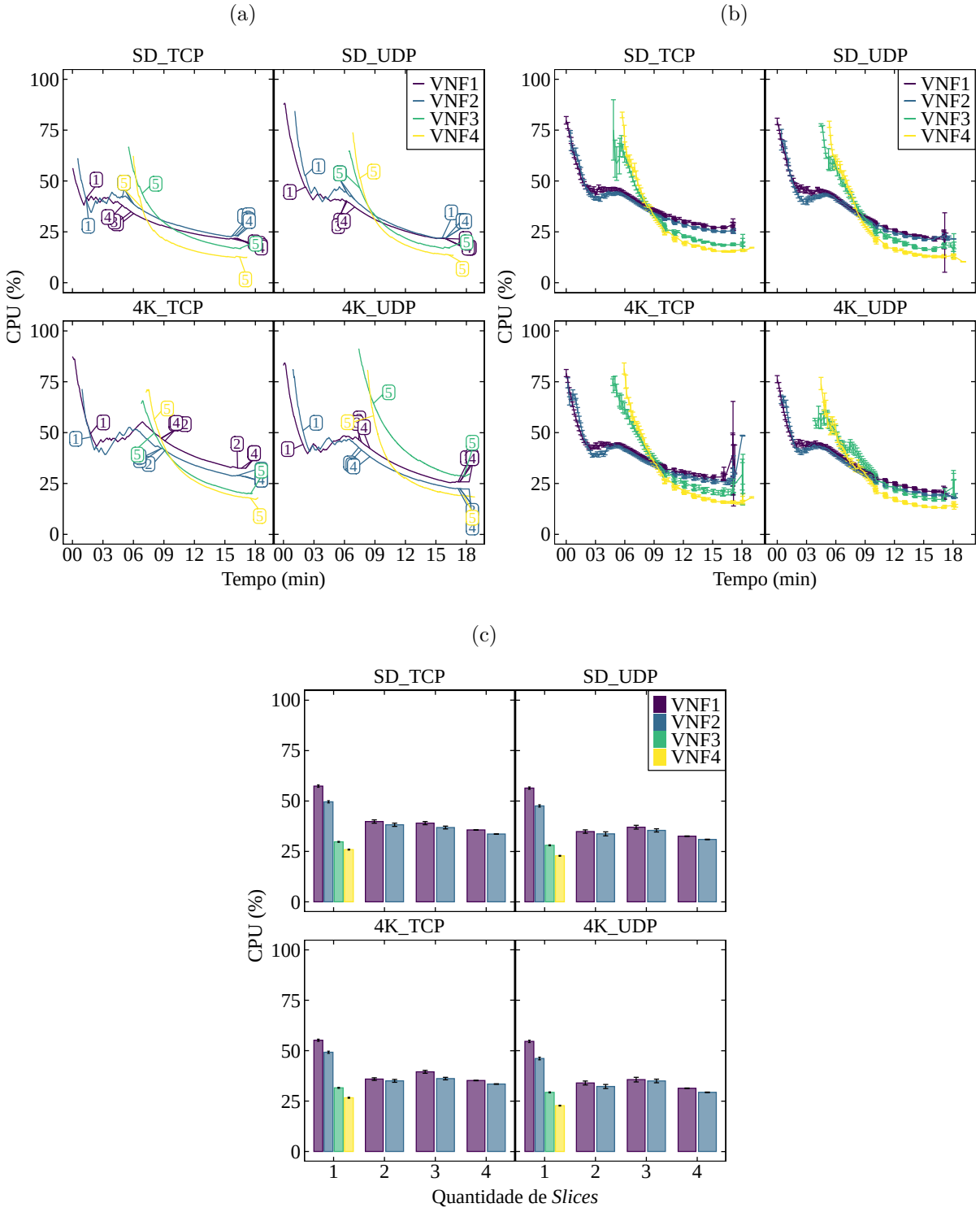
5.3.1.2 Com Elasticidade e sem Cache

Com o uso da elasticidade horizontal observou-se que os instantes iniciais das utilizações de CPU ficaram com valores próximos às VNFs já alocadas. No entanto, é possível perceber que as linhas de tendências declinam após os *setups* realizados nas novas VNFs, ficando maior parte do tempo abaixo do percentual de utilização em relação às VNFs referenciadas (Figura 20).

Como foi adotado o uso de VNFs com *flavors* reduzidos e não operando aplicações que necessitam de um grande poder computacional, o ganho com o uso da elasticidade não foi tão significativo em todos os resultados. Mesmo assim, foi possível obter uma redução de até 51,12% no uso de CPU entre a VNF2 e a VNF4, conforme ilustrado na Figura 20b.

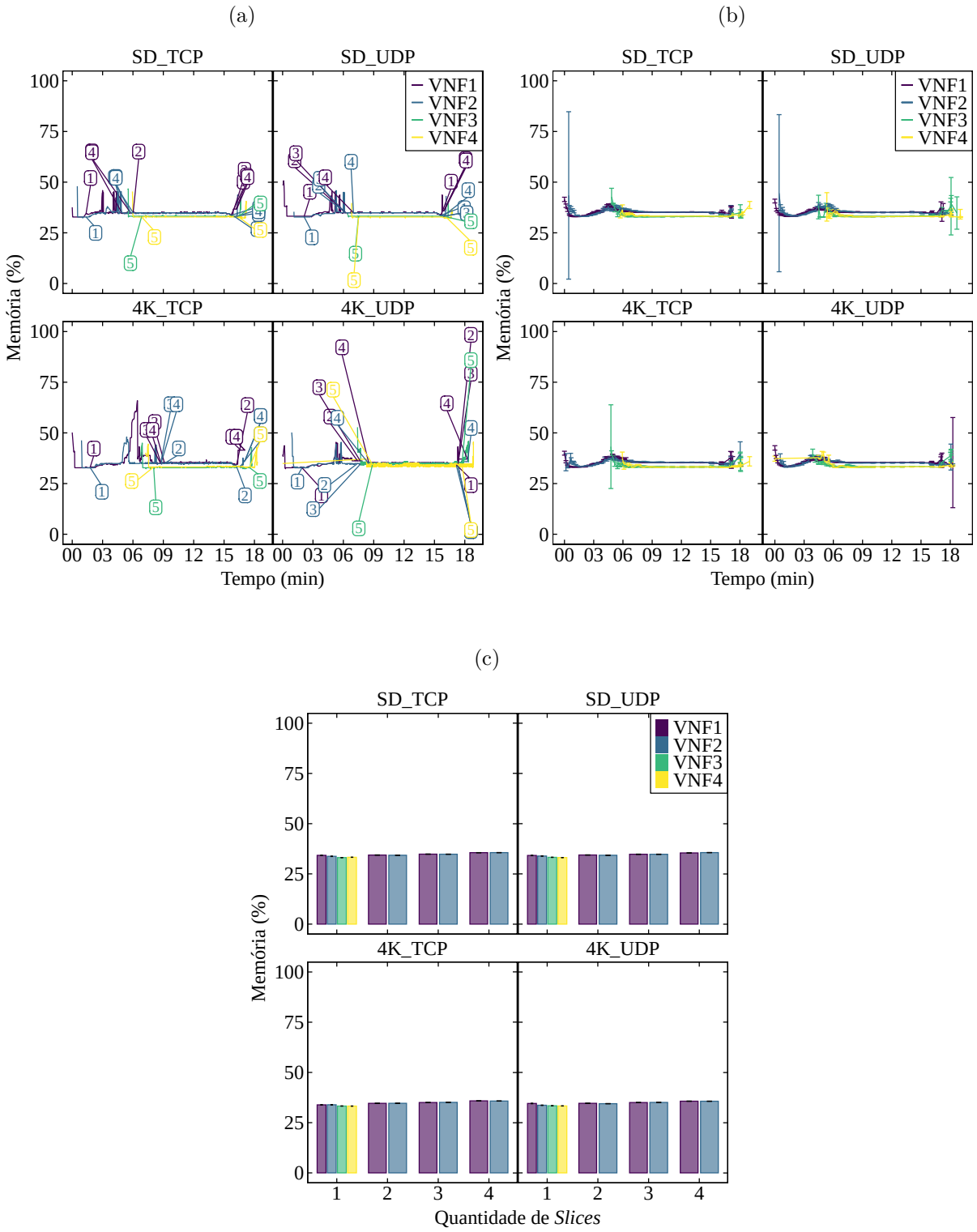
A menor diferença atingida com o uso de CPU foi de 5,43% entre a VNF1 e VNF3, no experimento usando o UDP com transmissões em 4K (Figura 20b).

Figura 20 – Utilização de CPU - Cenário CESC



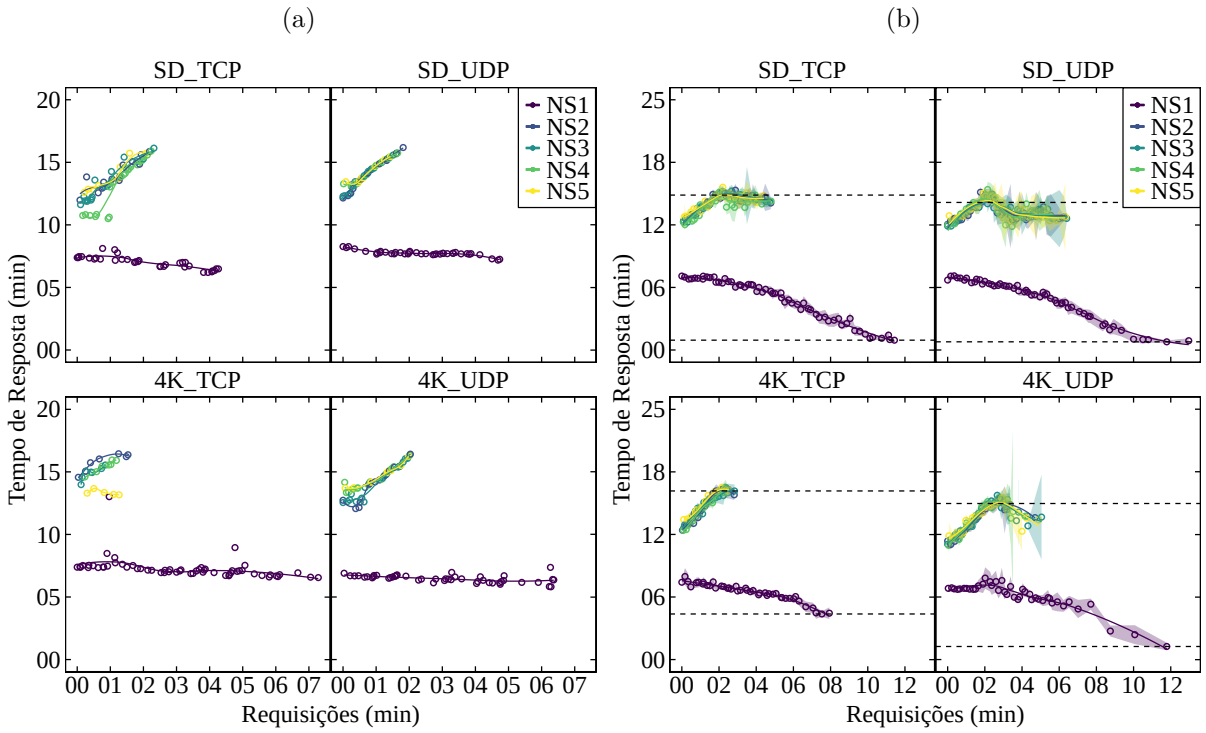
Por outro lado, a utilização de memória obteve uma redução de até 15,15% entre a VNF2 e a VNF4, com transmissões em SD e usando o UDP (Figura 21b). Nos demais resultados o impacto não foi tão expressivo, contudo, é possível estimar que essa diferença

Figura 21 – Utilização de Memória - Cenário CESC



poderia ser ainda mais significante com o uso de aplicações que necessitam de grandes recursos computacionais, bem como atendendo um grande número de clientes simultaneamente.

Figura 21 – Tempo de Resposta - Cenário CESC



5.3.1.3 *Sem Elasticidade e com Cache*

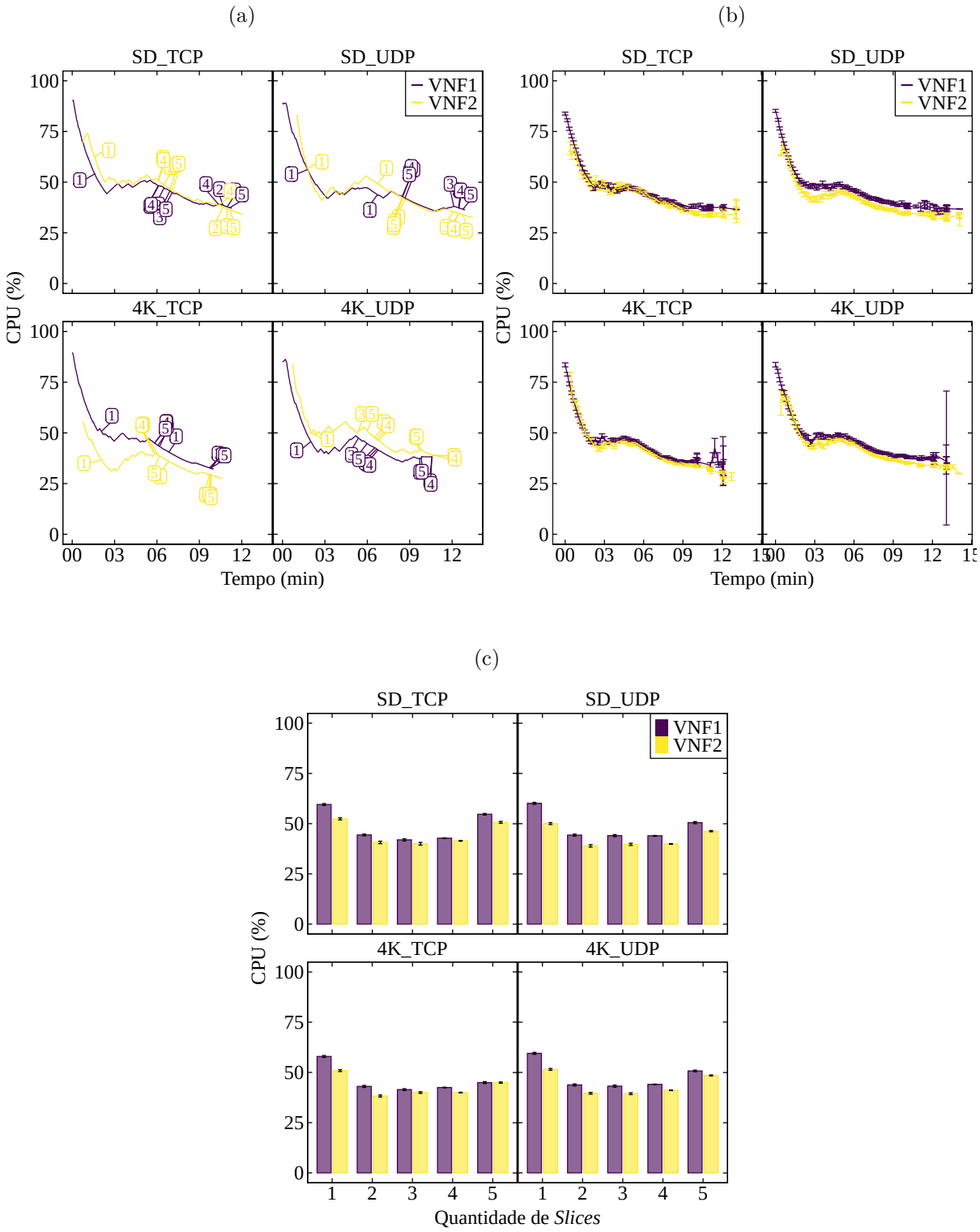
Conforme já discutido no início deste Capítulo e com o objetivo de reduzir o tempo de resposta, que é um requisito proposto em redes de quinta geração, buscamos adotar o uso da computação na borda e um mecanismo de cache de conteúdos. Diversos estudos abordam o uso do conceito de NDN como referência para a implementação de estratégias de cache, na qual pode ser aplicada em diferentes contextos. Neste trabalho, adotamos o uso de uma *Interest Table* agregada a um mecanismo de controle do ciclo de vida dos conteúdos, originando assim o *Content Store* e o Cache *Agent* adicionados à arquitetura proposta.

Nesta Seção é abordado o uso do cache sem o módulo *NF Scaling*, visando comparar os resultados obtidos SESC e SECC. Também é discutido a redução do ciclo de vida das VNFs após a inclusão do cache.

Com a inclusão do cache é possível obter ganhos relacionados ao ciclo de vida das VNFs. Como não é mais necessário que o produtor envie o mesmo conteúdo a cada nova requisição, o ciclo de vida das VNFs se torna menor através do gerenciamento realizado pelo módulo *Slice Manager*. Enquanto em experimentos SESC as instâncias ficam ativas em torno de 18 minutos, com SECC estas ficam provisionadas por até 14 minutos (conforme os resultados obtidos).

Por outro lado, a utilização de CPU apresentou um aumento significativo de até 52,18% pela VNF1, com a transmissão em SD usando o UDP (Figuras 18b e 22b). Tal aumento

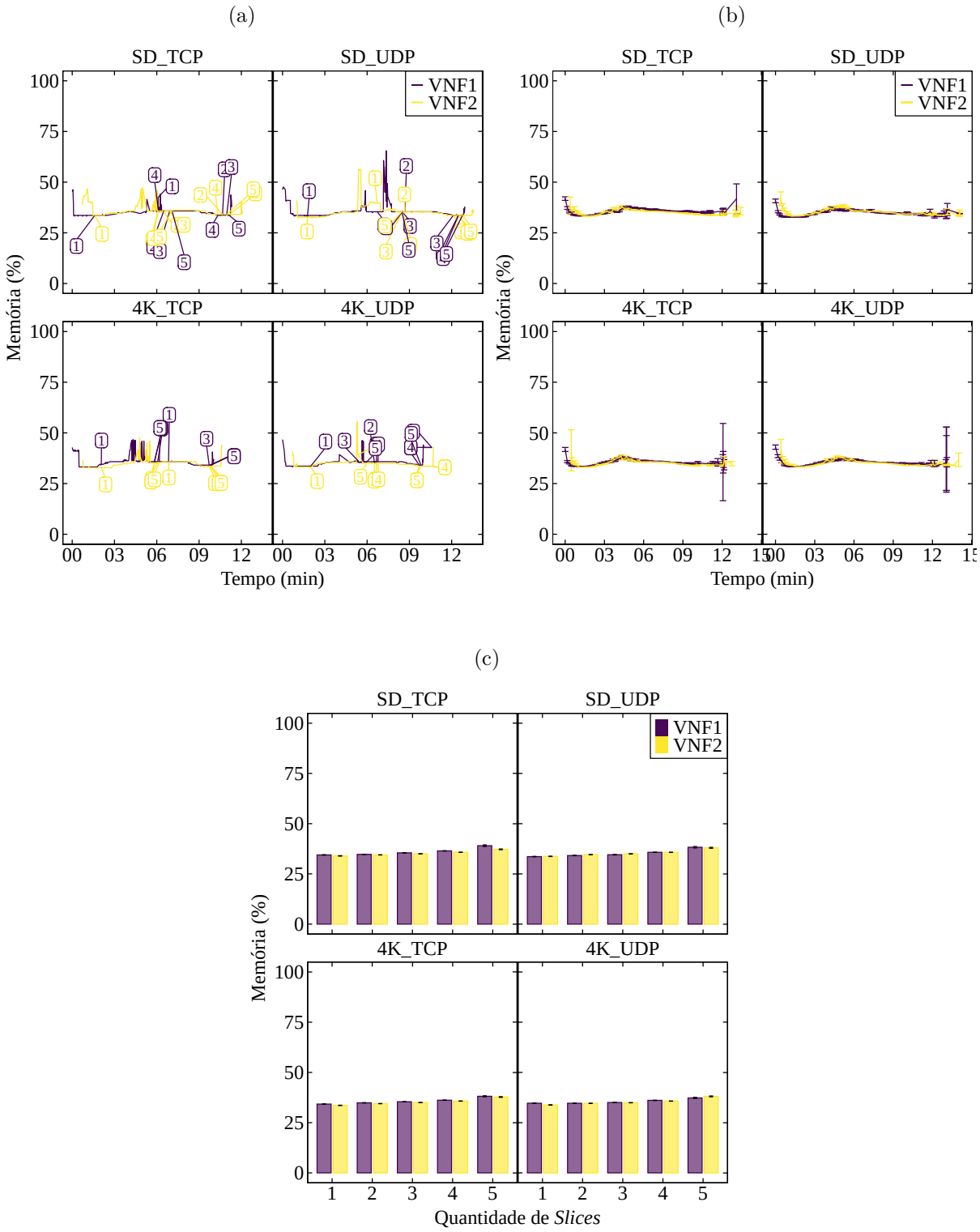
Figura 22 – Utilização de CPU - Cenário SECC



está diretamente relacionado ao fato do módulo *Slicing Orchestrator* estar trabalhando ao mesmo tempo que o *Slice Manager*, alocando e desalocando fatias de rede simultaneamente. Já a utilização de memória se manteve constante e não obteve uma diferença expressiva, na comparação com os resultados entre os experimentos SESC e SECC.

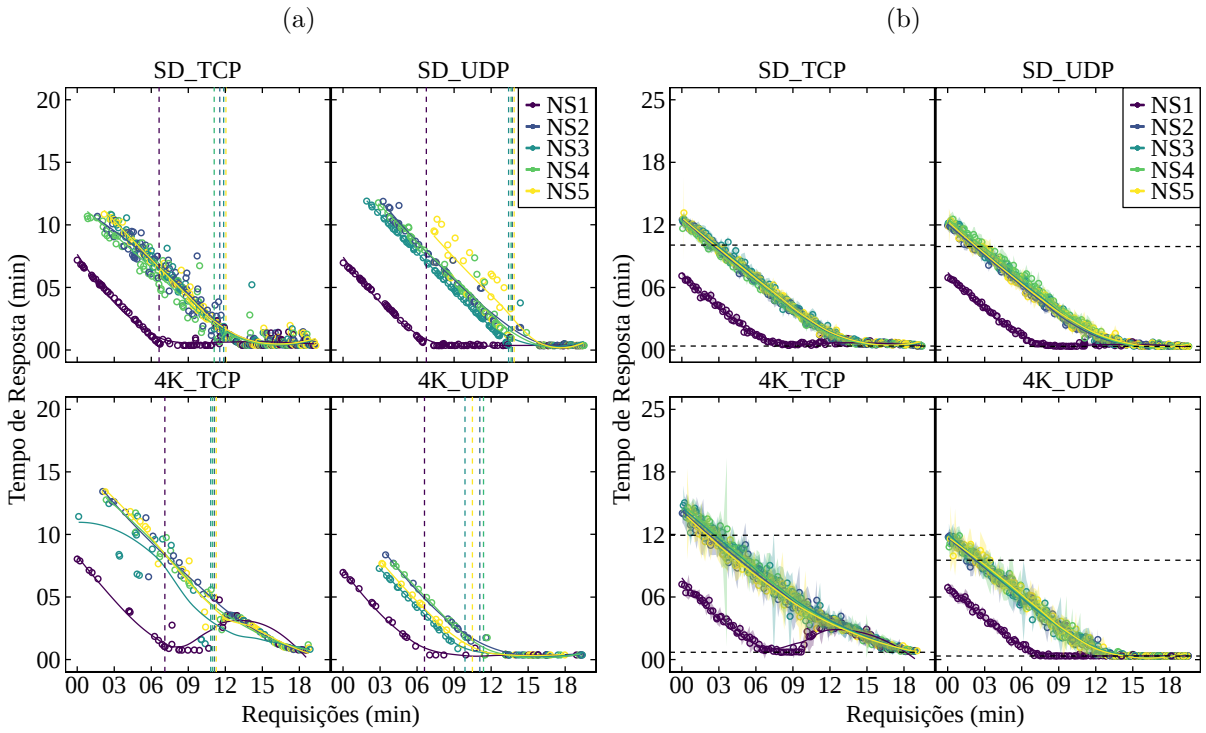
O principal ganho da utilização de estratégias de cache está relacionado ao tempo de

Figura 22 – Utilização de Memória - Cenário SECC



resposta (Figuras 22a, 22b). A Tabela 4 apresenta um comparativo do tempo de resposta entre os experimentos SESC e SECC. Com a utilização do cache foi possível obter uma redução de até 96,56% no tempo de resposta mínimo, e até 39,84% no tempo de resposta máximo. Além disso, pode-se observar que as menores médias e intervalos de confiança foram obtidas em experimentos usando o UDP.

Figura 22 – Tempo de Resposta - Cenário SECC



Por não realizar o *handshake*, como o TCP, o UDP consegue se sobressair quando o quesito é tempo de resposta. Apesar do UDP não garantir confiabilidade, ainda assim é amplamente usado quando se trata de fluxos de dados em tempo real ou quando perdas de pacotes são aceitáveis.

Tabela 4 – Tempo de Resposta - Mínimo e Máximo

Network Slices	**SD TCP**		**SD UDP**		**4K TCP**		**4K UDP**	
	SESC	**SECC**	**SESC**	**SECC**	**SESC**	**SECC**	**SESC**	**SECC**
NS1	109,0 ± 13,73	24,47 ± 1,62	260,33 ± 63,27	21,63 ± 0,22	176,33 ± 31,48	43,67 ± 2,87	42,00 ± 0,00	21,33 ± 0,54
	77,55%		91,69%		75,24%		49,21%	
	738,55 ± 136,90	444,34 ± 19,16	480,78 ± 90,97	433,40 ± 14,97	585,40 ± 38,55	451,06 ± 17,73	513,59 ± 100,67	431,31 ± 14,30
	39,84%		9,86%		22,95%		16,02%	
NS2	639,32 ± 11,74	23,73 ± 1,01	620,79 ± 14,26	21,45 ± 0,21	713,32 ± 16,82	49,17 ± 2,69	616,51 ± 20,86	21,36 ± 0,34
	96,29%		96,54%		93,11%		96,53%	
	980,43 ± 28,47	781,40 ± 29,47	962,40 ± 17,34	746,43 ± 30,51	899,76 ± 41,81	884,29 ± 78,58	967,75 ± 18,66	742,00 ± 41,95
	20,30%		22,44%		1,72%		23,33%	

(Continuação)

Network Slices	**SD TCP**		**SD UDP**		**4K TCP**		**4K UDP**	
	SESC	**SECC**	**SESC**	**SECC**	**SESC**	**SECC**	**SESC**	**SECC**
NS3	640,41 ± 9,87	23,83 ± 2,16	623,16 ± 13,18	21,50 ± 0,20	726,63 ± 18,48	50,78 ± 5,68	609,36 ± 21,61	21,14 ± 0,35
	96,28%		96,55%		93,01%		96,53%	
	992,33 ± 3,79	778,35 ± 46,69	961,00 ± 38,12	767,08 ± 38,96	951,18 ± 21,64	940,00 ± 40,02	975,60 ± 9,52	737,50 ± 77,95
	21,56%		20,18%		1,18%		24,41%	
NS4	646,29 ± 17,18	23,67 ± 1,70	626,15 ± 20,15	21,52 ± 0,24	713,32 ± 25,18	49,92 ± 2,28	613,17 ± 25,72	21,30 ± 0,35
	96,34%		96,56%		93,00%		96,53%	
	990,00 ± 88,94	783,84 ± 34,81	977,08 ± 23,94	784,56 ± 45,46	907,53 ± 49,17	911,00 ± 80,21	970,20 ± 11,36	742,64 ± 56,69
	20,82%		19,70%		0,38%		23,45%	
NS5	647,25 ± 78,95	25,23 ± 2,84	622,67 ± 106,39	21,52 ± 0,25	745,75 ± 21,65	49,00 ± 2,00	617,00 ± 56,07	21,25 ± 0,80
	96,10%		96,54%		93,43%		96,56%	
	997,83 ± 12,74	822,26 ± 32,04	964,50 ± 47,92	775,07 ± 38,56	927,63 ± 40,99	887,17 ± 55,06	966,33 ± 12,41	724,33 ± 198,08
	17,60%		19,64%		4,36%		25,04%	

Nota: As células apresentam as Médias ± CI em segundos, bem como o percentual com a diferença entre os resultados.

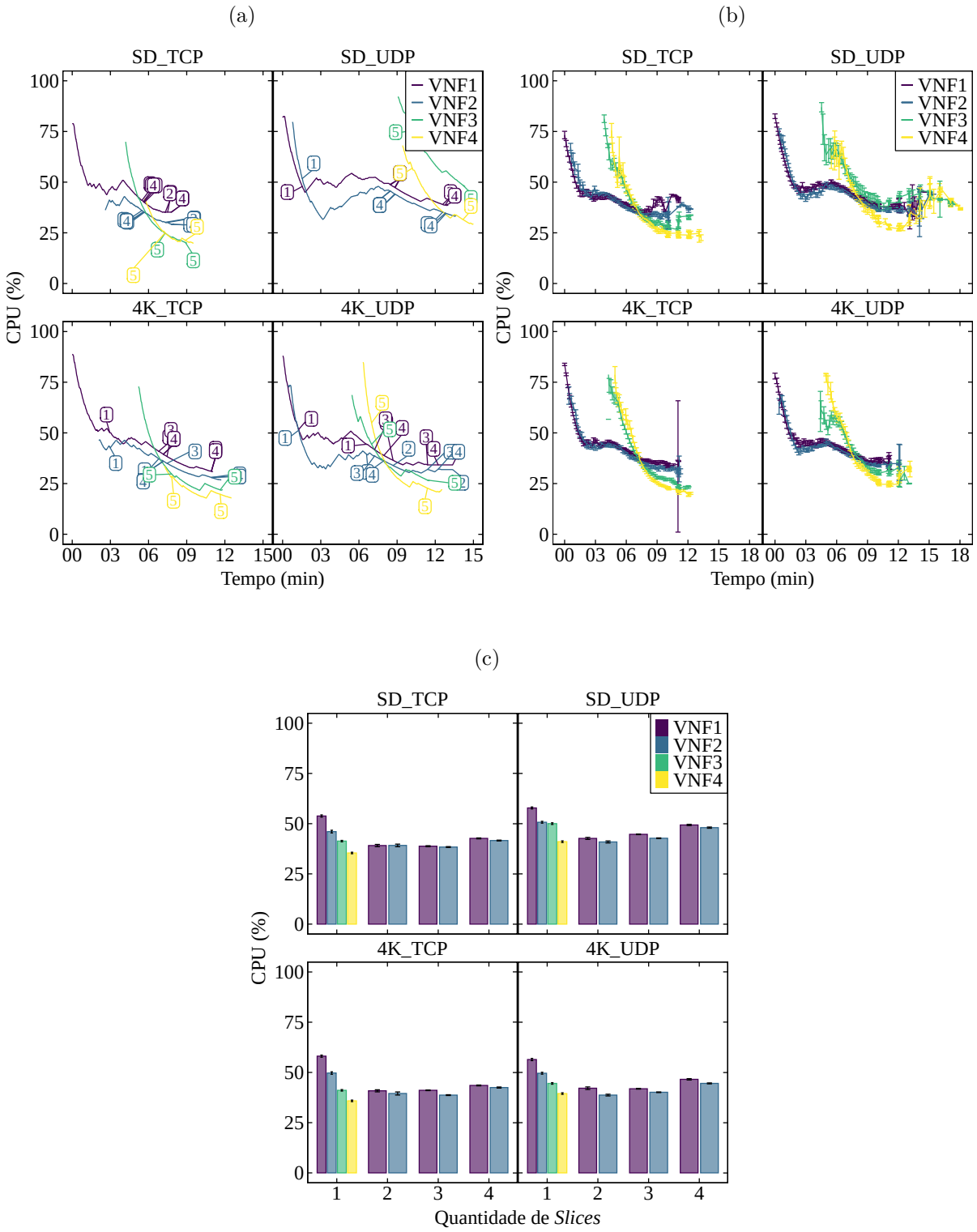
5.3.1.4 Com Elasticidade e com Cache

As Seções anteriores detalharam os ganhos e perdas com a utilização da elasticidade e cache. Nesta Seção, a avaliação conjunta da duas abordagens é realizada. Diferente dos experimentos CESC no qual foi possível atingir uma redução de até 51,12% na utilização de CPU com a elasticidade, após a inclusão do cache o maior ganho obtido foi de 36,94%, conforme exposto na Figura 23b pelas VNFs 1 e 2. Da mesma forma, a utilização de memória ganhou no máximo 10,54% (Figura 23b, VNF1 e VNF3), diferente dos experimentos CESC que foi de 15,15%.

Tratando-se do tempo de resposta, a menor média atingida foi de 00:21 segundos (NS3 - 4K_UDP), já a maior média foi de 15:56 minutos (NS5 - 4K_TCP). Ambos os resultados estão ilustrados no gráfico da Figura 23b.

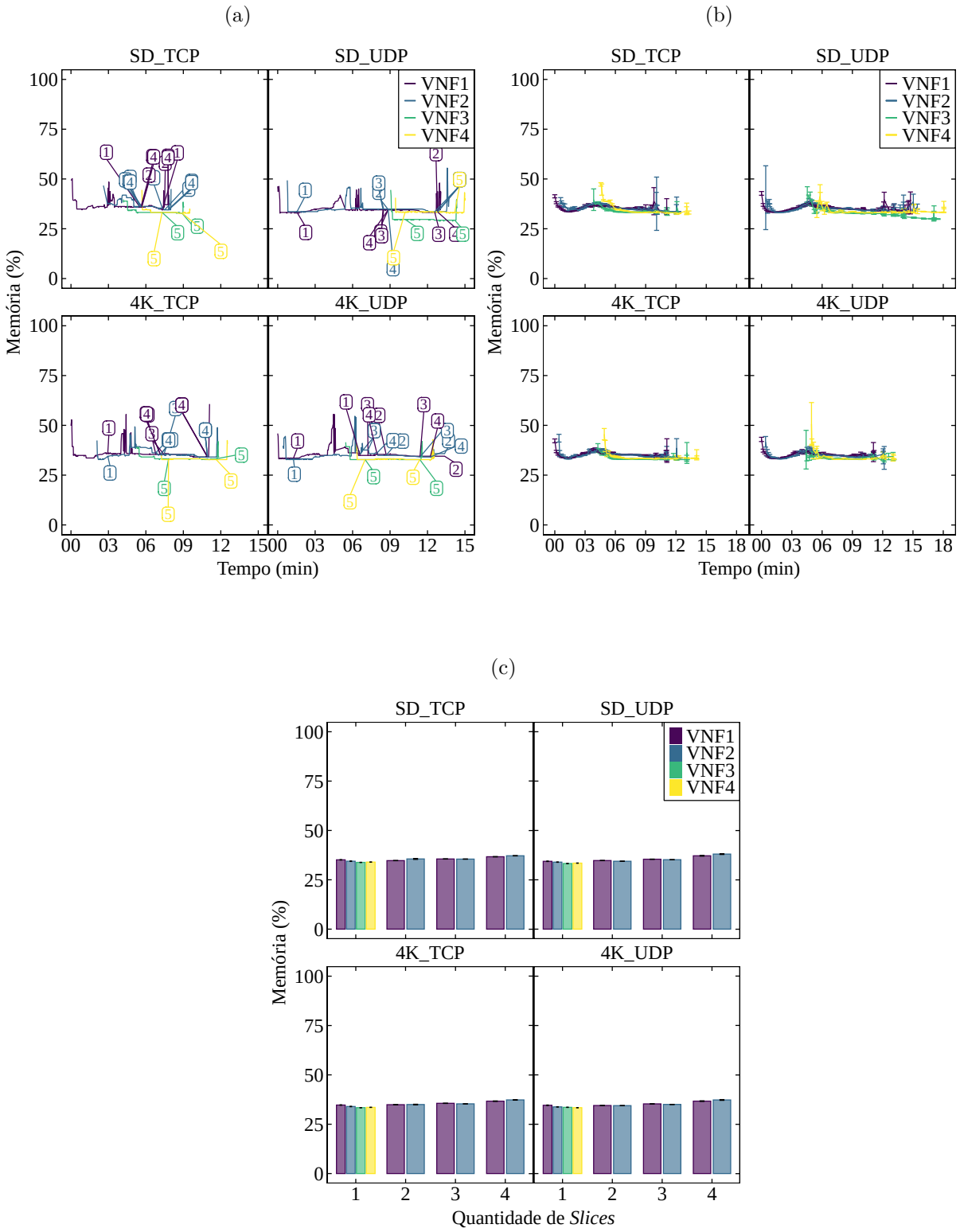
A Tabela 5 compara o tempo de resposta mínimo e máximo entre os cenários CESC,

Figura 23 – Utilização de CPU - Cenário CECC



SECC e CECC. Concluímos, portanto, que o cenário CECC obteve o menor tempo de resposta mínimo e, por outro lado, o uso da elasticidade impactou diretamente no tempo de resposta máximo, ficando próximo ao resultado obtido em CESC.

Figura 23 – Utilização de Memória - Cenário CECC



5.3.2 Alocação Proativa

Até a Seção anterior, todos os resultados apresentados e discutidos utilizaram a estratégia de alocação reativa, considerando o cenário ilustrado na Figura 16 e o tempo médio de provisionamento detalhado na Figura 14. Nesta Seção, são mostrados dois resultados para

Figura 23 – Tempo de Resposta - Cenário CECC

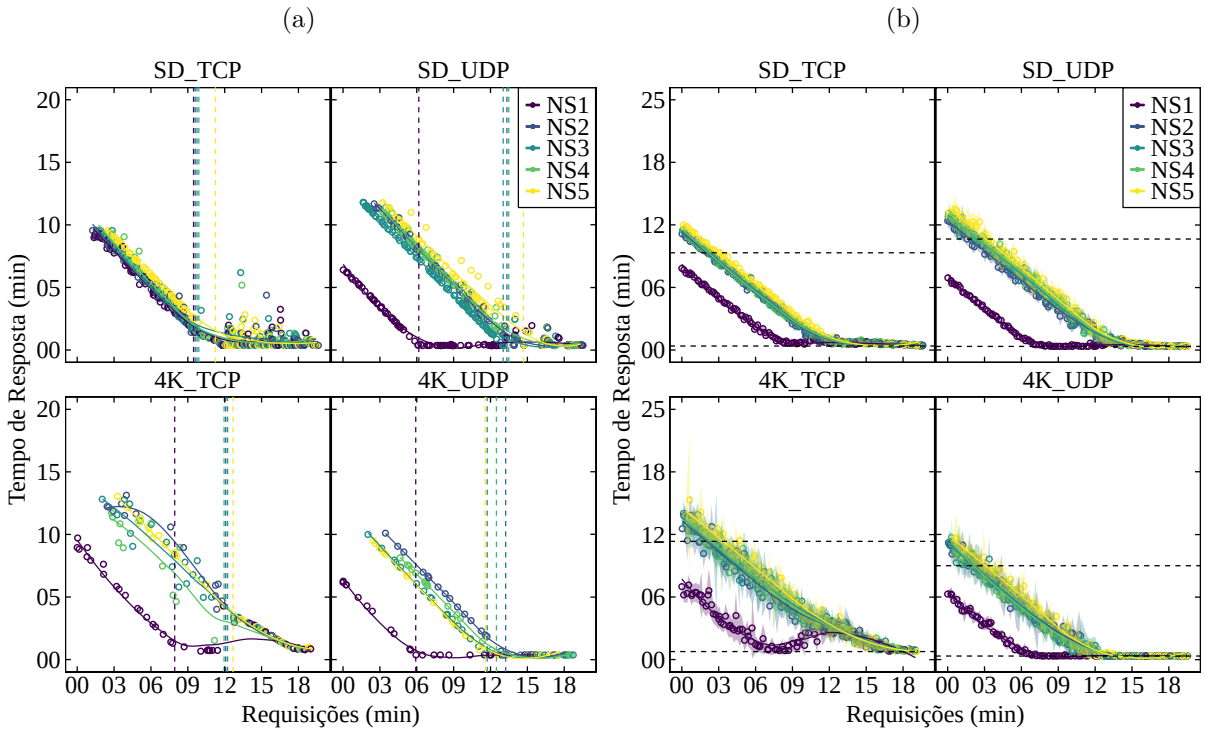


Tabela 5 – Comparativo entre tempos de respostas

CESC			SECC			CECC		
49 ± 8,04	NS1	SD_UDP	21,14 ± 0,35	NS3	4K_UDP	21 ± 0	NS3	4K_UDP
1033,11 ± 29,21	NS5	4K_TCP	940 ± 40,02	NS3	4K_TCP	956,67 ± 418,59	NS5	4K_TCP

Nota: As células apresentam as Médias ± CI em segundos.

o tempo de resposta usando a alocação proativa.

Consideramos como alocação proativa as fatias de rede já estarem alocados antes das requisições serem realizadas. Dessa forma, as solicitações são registradas na tabela de interesses e já é identificado que a fatia de rede associada a cada conteúdo já se encontra ativa. Neste tipo de alocação analisamos o ganho com a redução do tempo de resposta apenas para a configuração CECC, realizando transmissões em SD usando o protocolo TCP e UDP.

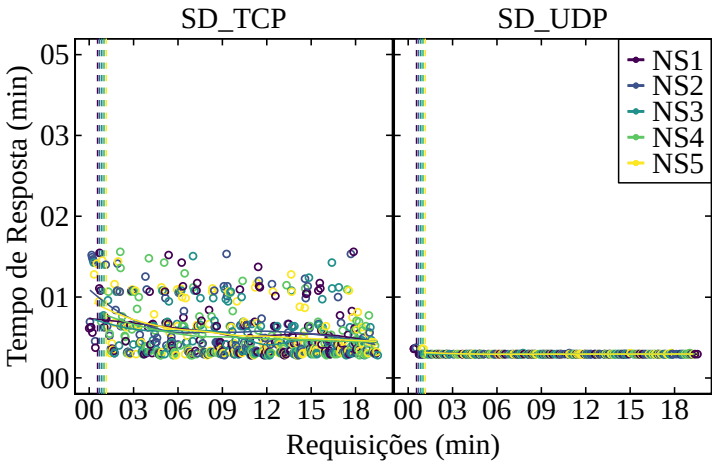
Com a alocação proativa já é possível desconsiderar o tempo de provisionamento entre 08:00 a 12:00 minutos amplamente discutido nas Seções anteriores e destacado na Figura 14. Mesmo se não forem utilizadas VNFs com virtualização total que ocupam mais espaço em disco e demorar mais tempo para inicializar, realizar *setups* de forma reativa pode acarretar um longo tempo de resposta, podendo ser ainda maior de acordo com o tipo de serviço que será instanciado.

As Figuras 24a e 24b apresentam os melhores resultados obtidos se comparado com

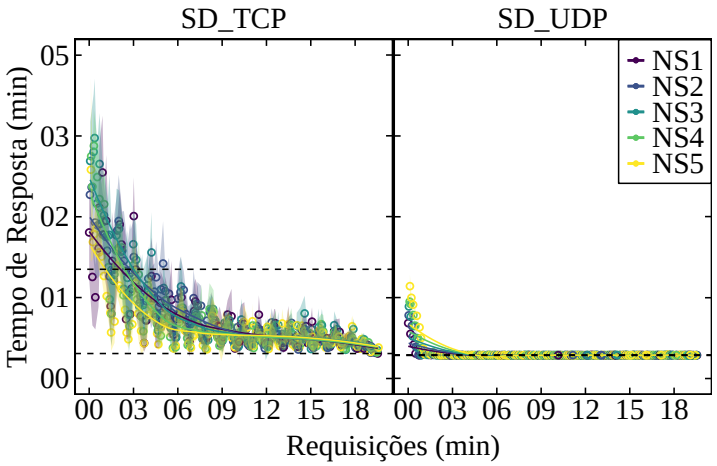
os demais, exibindo uma redução significativa do tempo de resposta. Usando o protocolo TCP as médias obtidas estão entre 00:38 segundos até 04:00 minutos, se considerados 100% dos valores coletados nas amostras. Esses valores são inferiores se analisados os limites obtidos com o percentil 90. Já com o UDP a maioria dos resultados está entre 00:20 a 00:30 segundos.

Figura 24 – Tempo de Resposta - Cenário CECC

(a)



(b)



Com a alocação proativa os percentuais de utilização se mantiveram constantes, sem os picos iniciais ocasionados pelos *setups* das VNFs. No geral, ficaram em torno de 10% de utilização de CPU e 30% de utilização de memória.

5.4 CONSIDERAÇÕES FINAIS

Neste Capítulo avaliamos a implementação da proposta desta dissertação, com base no sumário dos experimentos que foi planejado ao longo do desenvolvimento desta pesquisa. Buscamos validar todos os módulos implementados e seus respectivos impactos em cada

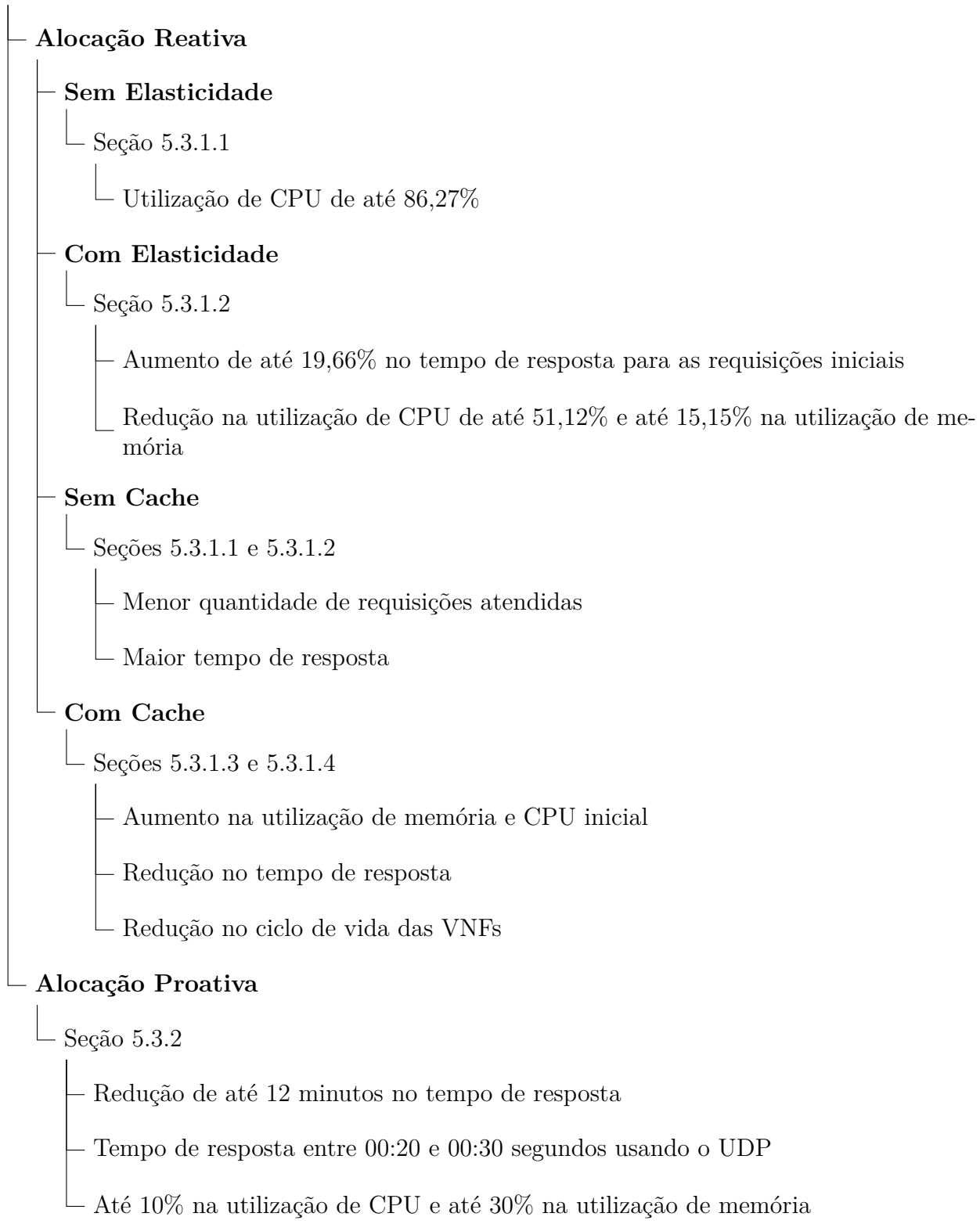
cenário avaliado. Dividimos nosso escopo de avaliação em duas vertentes principais: alocação proativa e reativa. Tratando-se da alocação reativa, foram avaliados quatro cenários: SESC, CESC, CESC e CECC, por meio de métricas como a utilização de CPU, utilização de memória e o tempo de resposta. Em todos os cenários utilizamos uma aplicação para a transmissão de vídeos em SD e 4K, usando os protocolos de transporte TCP e UDP. De modo geral, os resultados demonstraram a capacidade dos elementos que compõem a arquitetura proposta nesta dissertação, sendo eles o *NF Scaling*, *Content Store* com a *Interest Table*, além da estratégia utilizada para a alocação e o gerenciamento do ciclo de vida dos recursos computacionais.

Com todos os cenários avaliados destacamos os principais resultados obtidos na Figura 25. De maneira geral, as utilizações de CPU e memória foram maiores em cenários sem a elasticidade, o que valida a implementação do módulo *NF Scaling* e sua integração na arquitetura proposta. Como apontado na Figura 25, a maior utilização de CPU atingida foi de 86,27% nos instantes iniciais das VNFs. Por outro lado, tais utilizações se comportaram diferente em cenários onde o *Content Store* foi inserido, devido às modificações realizadas nas VNFs durante o processo de desalocação que é gerenciado pelo módulo *Slice Manager*. Os resultados de utilização de memória não tiveram impactos expressivos.

Através do uso do *Content Store* tivemos uma redução significativa nos tempos de resposta, sendo ainda maior com a alocação proativa com uma redução de até 12 minutos. O uso do protocolo UDP, em especial, apresentou os melhores resultados de tempo de resposta em todos os cenários avaliados, com os valores entre 00:20 e 0:30 segundos. Também foi possível constatar que a quantidade de requisições atendidas aumenta expressivamente, já que o conteúdo não precisa ser requisitado diretamente do produtor.

Outra característica crucial observada trata-se do tempo de permanência das VNFs alocadas em cenários com cache. Como cada VNF é desalocada assim que todas requisições são atendidas, e com o uso do cache apenas o primeiro interesse é considerado, tais parâmetros resultam em uma redução expressiva no ciclo de vida das VNFs (redução em torno de 04:00 minutos). Podemos concluir, portanto, que o uso do *Content Store* agregado à estratégia da *Interest Table* impacta diretamente no gerenciamento de recursos.

Figura 25 – Principais Resultados



6 CONCLUSÃO

Este Capítulo sumariza os principais pontos discutidos nesta dissertação. As considerações finais e as contribuições desta pesquisa são apresentadas nas Seções 6.1 e 6.2, respectivamente. Por último, listamos na Seção 6.3 algumas direções futuras e questões de pesquisa que não foram aprofundadas neste trabalho.

6.1 CONSIDERAÇÕES FINAIS

Neste trabalho, revisamos na literatura os principais desafios vigentes para as futuras infraestruturas de veículos conectados. Dentre os desafios encontrados, focamos em projetar uma arquitetura flexível para o compartilhamento de conteúdos na borda da rede. Para tal, identificamos que o conceito de fatiamento de rede agregada aos benefícios previstos pelas redes de quinta geração, se tornarão potenciais facilitadores. Após identificar os princípios essenciais para prover o fatiamento de rede, projetamos a arquitetura OPENS-IoV alinhada às características de redes de quinta geração descritas pelo 3GPP.

Além disso, foram apresentados os principais conceitos e as tecnologias utilizadas para conceber a arquitetura proposta. Para validar a nossa solução, criamos um *testbed* e avaliamos o comportamento dos módulos implementados em cenários diversos. Com os resultados obtidos após o conjunto de avaliações, listamos e discutimos todas as características e direções futuras observadas, além de todos os compromissos identificados.

6.2 CONTRIBUIÇÕES

Abaixo são listadas as principais contribuições desta dissertação, trazendo detalhes específicos de cada camada implementada na arquitetura OPENS-IoV, bem como as respostas para as questões de pesquisa elencadas em 1.2.1.

- Concepção e desenvolvimento de uma arquitetura para IoV
  - **Camada de Aplicação:** Implementada uma aplicação para a transmissão de vídeos entre produtores e consumidores de conteúdos. Com a abstração criada nesta camada, é possível estendê-la para outros tipos de aplicação que usam o modelo *publish/subscribe*.
  - **Camada de Serviço:** Criação e integração de um middleware orientado a mensagens para intermediar a troca de informações entre produtores e consumidores, além de uma estratégia de cache de conteúdos na borda da rede inspirada nos princípios NDN.

    Q4. Através da camada de infraestrutura do middleware foi criado uma abstração que permite o uso de diferentes protocolos de transporte.

Q5. Proposta uma estratégia de *caching* de conteúdos para veículos conectados.

– **Camada de Infraestrutura:** Utilizado a plataforma OpenStack como VIM para o provisionamento de recursos computacionais.

– **Camada de Orquestração e Gerenciamento:** Implementação MANO capaz de gerenciar o provisionamento e o ciclo de vida dos recursos; orquestrar fatias de rede sob demanda com base nos requisitos dos usuários; e por fim, fazer a elasticidade horizontal das VNFs.

Q1. Por meio do formato das mensagens foi possível identificar os requisitos dos usuários e tratá-los.

Q2. Os compromissos entre o provisionamento reativo e proativo foram avaliados e discutidos.

Q3. Foi apresentado o uso dos componentes *Slice Manager* e *NF Agent* para gerenciar o ciclo de vida das instâncias de serviço.

- Avaliação de desempenho considerando vários cenários IoV na transmissão de vídeo SD e 4K, por meio dos protocolos de transporte TCP e UDP.

6.3 TRABALHOS FUTUROS

Alguns elementos da arquitetura que não foram avaliados:

- Avaliar a taxa de perda de pacotes em cenários usando o UDP, para identificar o impacto refletido nos tempos de resposta.
- Avaliar os compromissos para a alocação de recursos dedicados.
- Avaliar o gerenciamento do ciclo de vida dos conteúdos no cache, já que este fator não foi incluído no escopo de avaliação deste trabalho.

Com as avaliações e os resultados obtidos com a construção desta dissertação, identificamos possíveis extensões à pesquisa desenvolvida através das seguintes questões de pesquisa e direções futuras:

- Quais os compromissos entre a elasticidade horizontal e vertical?
- Quais as melhores estratégias de desalocação de recursos?
- Investigar a viabilidade de usar o *Quick UDP Internet Protocol* (QUIC) na camada de infraestrutura do middleware.
- Estender a estratégia de elasticidade utilizada através do uso de novas informações contextuais das VNFs, tais como utilização de CPU e memória, para decidir o momento oportuno da elasticidade.

- Pesquisar e avaliar a utilização de *containers* como instâncias de serviço.
- Propor uma solução que leve em conta o *handover* dos veículos e das fatias de rede.

REFERÊNCIAS

Afolabi, I.; Taleb, T.; Frangoudis, P. A.; Bagaa, M.; Ksentini, A. Network slicing-based customization of 5G mobile services. *IEEE Network*, v. 33, n. 5, p. 134–141, Sep. 2019.

AFOLABI, I.; TALEB, T.; SAMDANIS, K.; KSENTINI, A.; FLINCK, H. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Communications Surveys & Tutorials*, IEEE, v. 20, n. 3, p. 2429–2453, 2018.

ALLIANCE, N. Next Generation Mobile Networks. *NGMN 5G White Paper*, v. 1, 2015.

ALLIANCE, N. Description of Network Slicing Concept. *NGMN 5G P1 Requirements & Architecture, Work Stream End-to-End Architecture*, v. 1, 2016.

BARI, M. F.; BOUTABA, R.; ESTEVES, R.; GRANVILLE, L. Z.; PODLESNY, M.; RABBANI, M. G.; ZHANG, Q.; ZHANI, M. F. Data center network virtualization: A survey. *IEEE Communications Surveys & Tutorials*, IEEE, v. 15, n. 2, p. 909–928, 2012.

CAMPOLO, C.; FONTES, R. dos R.; MOLINARO, A.; ROTHENBERG, C. E.; IERA, A. Slicing on the road: Enabling the automotive vertical through 5G network softwarization. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 18, n. 12, p. 4435, 2018.

CELDRÁN, A. H.; PÉREZ, M. G.; CLEMENTE, F. J.; IPPOLITI, F.; PÉREZ, G. M. Dynamic network slicing management of multimedia scenarios for future remote healthcare. *Multimedia Tools and Applications*, Springer New York LLC, 2019. ISSN 15737721.

CELDRÁN, A. H.; PÉREZ, M. G.; CLEMENTE, F. J. G.; IPPOLITI, F.; PÉREZ, G. M. Policy-based network slicing management for future mobile communications. In: IEEE. *2018 Fifth International Conference on Software Defined Systems (SDS)*. [S.l.], 2018. p. 153–159.

CHIEN, H.-T.; LIN, Y.-D.; LAI, C.-L.; WANG, C.-T. End-to-end slicing as a service with computing and communication resource allocation for multi-tenant 5G systems. *IEEE Wireless Communications*, IEEE, v. 26, n. 5, p. 104–112, 2019.

DIETZEL, S.; PETIT, J.; KARGL, F.; SCHEUERMANN, B. In-network aggregation for vehicular ad hoc networks. *IEEE communications surveys & tutorials*, IEEE, v. 16, n. 4, p. 1909–1932, 2014.

ETSI, G. Network Functions Virtualization (NFV): Architectural framework. *ETSI Gs NFV*, v. 2, n. 2, p. V1, 2013.

ETSI, G. *Multi-access Edge Computing (MEC); Framework and Reference Architecture - ETSI GS MEC 003 V2.1.1*. 2019. Disponível em: <https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf>. Acesso em: 14 jun. 2020.

ETSI, G. *Multi-access Edge Computing (MEC); Framework and Reference Architecture - ETSI GS MEC 003 V2.2.1*. 2020. Disponível em: <https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.02.01_60/gs_MEC003v020201p.pdf>. Acesso em: 14 jun. 2020.

FANGCHUN, Y.; SHANGGUANG, W.; JINGLIN, L.; ZHIHAN, L.; QIBO, S. An overview of Internet of vehicles. *China Communications*, IEEE, v. 11, n. 10, p. 1–15, 2014.

HARTENSTEIN, H.; LABERTEAUX, L. A tutorial survey on vehicular ad hoc networks. *IEEE Communications magazine*, IEEE, v. 46, n. 6, p. 164–171, 2008.

HAWKING, S. *An Earth Day message from Professor Stephen Hawking and ESA*. 2018. Disponível em: <https://stephenhawkingfoundation.org/an-earth-day-message-from-professor-stephen-hawking-and-esa>. Acesso em: 01 jun. 2021.

HE, Z.; ZHANG, D.; LIANG, J. Cost-efficient heterogeneous data transmission in software defined vehicular networks. In: IEEE. *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*. [S.l.], 2015. p. 666–671.

HERNANDEZ-VALENCIA, E.; IZZO, S.; POLONSKY, B. How will NFV/SDN transform service provider OPEX? *IEEE Network*, IEEE, v. 29, n. 3, p. 60–67, 2015.

HERRERA, J. G.; BOTERO, J. F. Resource allocation in nfv: A comprehensive survey. *IEEE Transactions on Network and Service Management*, IEEE, v. 13, n. 3, p. 518–532, 2016.

KREUTZ, D.; RAMOS, F. M.; VERISSIMO, P. E.; ROTHENBERG, C. E.; AZODOLMOLKY, S.; UHLIG, S. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, IEEE, v. 103, n. 1, p. 14–76, 2015.

LEDYAYEV, R.; RICHTER, H. High performance computing in a cloud using openstack. *CLOUD COMPUTING*, p. 108–113, 2014.

MAO, Y.; YOU, C.; ZHANG, J.; HUANG, K.; LETAIEF, K. B. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, IEEE, v. 19, n. 4, p. 2322–2358, 2017.

MEDEIROS, A.; NETO, A.; SAMPAIO, S.; PASQUINI, R.; BALIOSIAN, J. End-to-end elasticity control of cloud-network slices. *Internet Technology Letters*, Wiley Online Library, v. 2, n. 4, p. e106, 2019.

MELL, T. G. P. The NIST Definition of Cloud Computing. National Institute of Standards and Technology, SP 800-145, 2011.

MIJUMBI, R.; SERRAT, J.; GORRICHO, J.-L.; BOUTEN, N.; TURCK, F. D.; BOUTABA, R. Network function virtualization: State-of-the-art and research challenges. *IEEE Communications Surveys & Tutorials*, IEEE, v. 18, n. 1, p. 236–262, 2016.

OPERATORS, N. Network functions virtualization, an introduction, benefits, enablers, challenges and call for action. In: *SDN and OpenFlow SDN and OpenFlow World Congress*. [S.l.: s.n.], 2012.

PATEL, M.; NAUGHTON, B.; CHAN, C.; SPRECHER, N.; ABETA, S.; NEAL, A. et al. Mobile-edge computing introductory technical white paper. *White Paper, Mobile-edge Computing (MEC) industry initiative*, 2014.

ROSA, R.; SIQUEIRA, M.; BAREA, E.; MARCONDES, C.; ROTHENBERG, C. Network function virtualization: Perspectivas, realidades e desafios. *Minicursos do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC*, 2014.

ROTHENBERG, C. E.; NASCIMENTO, M. R.; SALVADOR, M. R.; MAGALHÃES, M. F. Openflow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes. *Cad. CPQD Tecnologia, Campinas*, v. 7, n. 1, p. 65–76, 2010.

SHAH, S. A. A.; AHMED, E.; IMRAN, M.; ZEADALLY, S. 5G for vehicular communications. *IEEE Communications Magazine*, IEEE, v. 56, n. 1, p. 111–117, 2018.

SOENEN, T.; BANERJEE, R.; TAVERNIER, W.; COLLE, D.; PICKAVET, M. Demystifying network slicing: From theory to practice. In: IEEE. *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. [S.l.], 2017. p. 1115–1120.

STATISTA. *Connected Cars - Statistics & Facts.* 2018. Disponível em: <https://www.statista.com/topics/1918/connected-cars>. Acesso em: 01 jun. 2021.

STATISTA. *Estimated shipments of connected cars globally from 2017 to 2020 (in million units).* 2018. Disponível em: <https://www.statista.com/statistics/743400/estimated-connected-car-shipments-globally>. Acesso em: 01 jun. 2021.

STATISTA. *Connected vehicle fleet from 2025 to 2035, by key region.* 2020. Disponível em: <https://www.statista.com/statistics/442529/connected-car-market-size-by-region>. Acesso em: 01 jun. 2021.

STATISTA. *CSize of the global connected car market between 2020 and 2025.* 2020. Disponível em: <https://www.statista.com/statistics/725025/connected-cars-global-market-size-projection>. Acesso em: 01 jun. 2021.

VÖLTER, M.; KIRCHER, M.; ZDUN, U. *Remoting patterns: foundations of enterprise, Internet and realtime distributed object middleware*. [S.l.]: John Wiley & Sons, 2013.

YANG, F.; LI, J.; LEI, T.; WANG, S. Architecture and key technologies for internet of vehicles: a survey. 2017.

ZHANG, L.; AFANASYEV, A.; BURKE, J.; JACOBSON, V.; CLAFFY, K.; CROWLEY, P.; PAPADOPOULOS, C.; WANG, L.; ZHANG, B. Named data networking. *ACM SIGCOMM Computer Communication Review*, ACM New York, NY, USA, v. 44, n. 3, p. 66–73, 2014.

ZHOU, H.; CHENG, N.; WANG, J.; CHEN, J.; YU, Q.; SHEN, X. Toward dynamic link utilization for efficient vehicular edge content distribution. *IEEE Transactions on Vehicular Technology*, IEEE, v. 68, n. 9, p. 8301–8313, 2019.