UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Alexandre Martins Gama de Deus

Join-Me: Um Framework para integração entre Provedores de Serviços e Operadores de Redes Móveis

Juiz de Fora
2021

Alexandre Martins Gama de Deus

Join-Me: Um Framework para integração entre Provedores de Serviços e Operadores de Redes Móveis

Dissertação apresentada ao Programa de Pós Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Marcelo Ferreira Moreno

Coorientador: Prof. Dr. Eduardo Pagani Julio

Juiz de Fora

2021

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

de Deus, Alexandre Martins Gama.
Join-Me : Um Framework para integração entre Provedores de Serviços e Operadores de Redes Móveis / Alexandre Martins Gama de Deus. – 2021.
58 f. : il.

Orientador: Marcelo Ferreira Moreno
Coorientador: Eduardo Pagani Julio
Dissertação (Mestrado) – Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Programa de Pós Graduação em Ciência da Computação, 2021.

1. Computação de Borda. 2. Migração de Serviços. 3. Qualidade de Experiência. 4. Provisionamento de Serviços. 5. Redes Móveis. 6. Internet do Futuro. I. Moreno, Marcelo Ferreira, orient. II. Julio, Eduardo Pagani, coorient. Título.

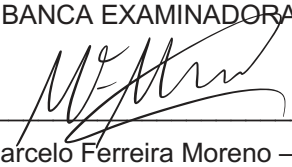
Alexandre Martins Gama de Deus

"Join-Me: Um Framework para integração para Provedores de Serviços e Operadores de Redes Móveis"

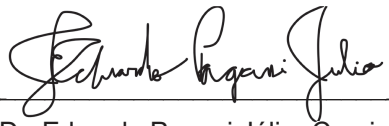
Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Ciência da Computação.

Aprovada em 05 de março de 2021.

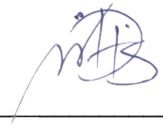
BANCA EXAMINADORA



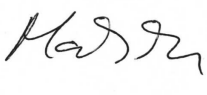
Prof. Dr. Marcelo Ferreira Moreno – Orientador
Universidade Federal de Juiz de Fora



Prof. Dr. Eduardo Pagani Júlio- Coorientador
Universidade Federal de Juiz de Fora



Prof. Dr. Mario Antonio Ribeiro Dantas
Universidade Federal de Juiz de Fora



Prof. Dr. Markus Endler
Pontifícia Universidade Católica do Rio de Janeiro

AGRADECIMENTOS

Agradeço primeiramente a Deus, Mestre dos mestres e Doutor dos doutores, por ter me ajudado até aqui. À minha esposa Roberta, por estar sempre ao meu lado me apoiando e dando o suporte emocional sempre que preciso. Aos meus familiares João Luiz, Glória e Anna Eliza, por terem contribuído com a pessoa que sou e por sempre acreditarem em mim.

Ao Centro Federal de Educação Tecnológica de Minas Gerais - CEFET/MG, juntamente com a Direção do Campus Leopoldina, Douglas Martins e José Geraldo Jr., por terem me concedido a licença capacitação, a qual foi de suma importância para a obtenção deste título.

Aos amigos os quais contribuíram com a camaradagem e a técnica no desenvolvimento do trabalho, em especial Thales Almeida e George Farias. Aos demais familiares e amigos, um muito obrigado por estarem comigo, fisicamente ou em orações.

Agradeço pelo suporte oferecido pelos profissionais do Núcleo de Recursos Computacionais - NRC, os quais se tornaram meus novos amigos. E à Rede de Pesquisa de alta velocidade - RePesq, por terem fornecido o apoio e a infraestrutura necessária para o desenvolvimento do trabalho.

Por fim, mas não menos importante, a melhor dupla de orientador e coorientador que um mestrando poderia ter, Marcelo Moreno e Eduardo Pagani. Muito obrigado pelo conhecimento compartilhado, pela paciência e dedicação ao ensinar e pelo companheirismo ao longo do trabalho.

Que Deus possa retribuir cada um de vocês!

RESUMO

Com a popularização dos *smartphones* e do acesso móvel à Internet, Provedores de Serviços e Operadores de Redes Móveis têm empregado significativos esforços para elevar a qualidade de experiência (QoE) dos usuários. Observa-se, no entanto, que em tais esforços não há uma integração entre ambas as partes, que permitiria o uso avançado e sob demanda de recursos para impactar positivamente na QoE dos usuários. Para suprir esta lacuna, este trabalho apresenta Join-Me, um *framework* que possibilita que Provedores de Serviços transfiram módulos de seus serviços para a infraestrutura dos Operadores de Redes Móveis. Com a gestão de tais módulos feita internamente pela rede, os serviços podem ser entregues tão próximos do usuário quanto necessário, contando com uma gerência dinâmica de recursos e respeitando a privacidade do usuário. Join-Me define, além dos componentes funcionais necessários, uma interface que permite aos Provedores de Serviços especificarem, para cada um dos módulos de um serviço, a demanda por uso de recursos, pela qual serão tarifados adequadamente. Um serviço de telemedicina interativa com vídeo de ultra alta definição é utilizado como prova de conceito, sendo avaliada com o método de *Video Multimethod Assessment Fusion*, de forma a demonstrar concretamente o ganho de QoE como intencionado por Join-Me.

Palavras-chave: Computação de Borda. Migração de serviços. Qualidade de Experiência. Provisionamento de serviços. Redes móveis. Integração de serviços.

ABSTRACT

With the popularization of smartphones and mobile access to the Internet, Service Providers and Mobile Network Operators have employed significant efforts to increase users' quality of experience (QoE). It is observed, however, that in such efforts there is no integration between both parties, which would allow for an enhanced and on-demand use of resources to positively impact users' QoE. To fill this gap, this work presents Join-Me, a framework that allows Service Providers to transfer modules of their services into the infrastructure of Mobile Network Operators. With the management of such modules being done internally in the network, services may be delivered as close to the user as needed, relying on a dynamic resource management and respecting user's privacy. Besides the needed functional components, Join-Me defines an interface that allows Service Providers to specify, for each module of a service, its resource usage demand, which they will be properly billed for. An interactive telemedicine service with ultra high definition video is used as a proof of concept, being evaluated with the Video Multimethod Assessment Fusion method, in order to concretely demonstrate QoE improvement as intended by Join-Me.

Keywords: Edge Computing. Service Migration. Quality of Experience. Service Provisioning. Mobile Networks. Service Integration.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3GPP	3rd Generation Partnership Project
AF	Application Function
AnyaaS	Anything as a Service
API	Application Programming Interface
AMF	Access Mobility Management Function
AMPS	Advanced Mobile Phone Service
Anatel	Agência Brasileira de Telecomunicações
AUSF	Authentication Server Function
CAPEX	Capital Expenditures
CAPIF	Common API Framework
CDMA	Code Division Multiple Access
CDN	Content Delivery Network
DASH	Dynamic Adaptative Streaming over HTTP
DLM	Detail Loss Metric
DN	Data Network
EAD	Ensino à Distância
EDGE	Enhanced Data Rate for GSM
ERB	Estação Rádio Base
ETSI	European Telecommunication Standards Institute
EVDO	Evolution-Data Optimized
FDMA	Frequency Division Multiple Access
FMC	Follow-Me Cloud
gNB	Next Generation NodeB
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HLS	HTTP Live Streaming
HSDPA	High Speed Downlink Packet Access
HSUPA	High Speed Uplink Packet Access
HTTP	Hyper Text Transfer Protocol
IaaS	Infrastructure as a Service
IoT	Internet of Things
IP	Internet Protocol
JSON	JavaScript Object Notation
LTE	Long Term Evolution
LS	Location Service
MEC	Multi-Access Edge Computing
MNO	Mobile Network Operator
NAT	Network Address Translation
NEF	Network Exposure Function
NF	Network Function

NFV	Network Function Virtualization
NRF	Network Repository Funcion
NSSF	Network Slice Selection Function
OAI	OpenAPI Initiative
OAS	OpenAPI Specification
OFDMA	Orthagonal Frequency Division Multiple Access
PaaS	Plataform as a Service
OPEX	Operational Expenditures
PoC	Proof of Concept
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
REST	Representational State Transfer
RNIS	Radio Network Information Service
SBA	Service Based Archtecture
SBI	Service Based Interface
SC-FDMA	Single Carrier Frequency Division Multiple Access
SDN	Software Defined Networks
SMF	Session Management Function
SMS	Short Message Service
SP	Service Provider
SSH	Secure Shell
TDMA	Time Division Multiple Access
ToS	Type of Service
UDP	User Datagram Protocol
UE	User Equipament
UPF	User Plane Function
UMTS	Universal Mobile Telecommunications System
UX	User Experience
V2X	Vehicle to Everything
VIF	Visual Information Fidelity
VIM	Virtualization Infrastructure Manager
VM	Virtual Machine
VMAF	Video Multimethod Assessment Fusion
vNF	Virtualized Network Function
VoD	Video on Demand
YAML	YAML Ain't Markup Language
WCDMA	Wideband Code Division Multiple Access
WIMAX	Worldwide Interoperability for Microwave Access

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, empresas de tecnologia e centros de pesquisa empregaram esforços a fim de elevar a Qualidade de Experiência (QoE) dos usuários na entrega de serviços através da Internet. Tais esforços objetivam reduzir contratempos inerentes à abordagem de melhor esforço presente na Internet atual, como congestionamento, perda e retransmissão de pacotes, alta variação estatística do retardo e entrega de pacotes fora da sequência de transmissão.

No contexto de aplicações como as de multimídia, a busca pela melhoria na QoE torna-se uma tarefa indispensável para os Provedores de Serviços (SPs - *Service Providers*), de modo que estes são obrigados a investir em serviços, normalmente terceirizados, como as Redes de Distribuição de Conteúdo (CDNs - *Content Delivery Networks*) (1). Neste caso, as CDNs permitem a extensão dos recursos de armazenamento e entrega de serviços aos SPs, uma vez que replicam o conteúdo em servidores distribuídos geograficamente, balanceando o tráfego e possibilitando a entrega de serviços em pontos de presença mais próximos ao usuário.

Em paralelo, a indústria de *hardware* tem reduzido o tamanho dos componentes eletrônicos, possibilitando que *smartphones* cada vez mais poderosos computacionalmente e com grande capacidade de armazenamento sejam lançados continuamente no mercado com custos acessíveis. A aquisição facilitada desses dispositivos alavanca também a utilização de redes móveis, o que coloca o Brasil em sexto lugar no *ranking* dos principais mercados de telefonia móvel do mundo em 2018, de acordo com a Agência Nacional de Telecomunicações (Anatel)[1]. Não foi possível encontrar dados de mercado mais recentes.

Consequentemente, a utilização de aplicativos móveis, em especial os de redes sociais e conteúdo multimídia, incluindo serviços de jogos on-line e *streaming* de áudio e vídeo impulsionam o consumo de dados. O tráfego de dados tem crescido consideravelmente nos últimos cinco anos de acordo com o relatório da DataReportal[2], caracterizando a dominância das redes móveis como um dos principais meios de acesso à Internet para usuários finais.

A tecnologia das redes móveis vem sendo aprimorada ao longo dos anos para lidar não apenas com o tráfego de dados de *smartphones* mas também de tecnologias emergentes como Internet das Coisas (IoT - *Internet of Things*) e Sistemas de Comunicações Veicular (V2X - *Vehicle to Everithing*). Dados esses novos segmentos, surge então a Quinta Geração de Redes Móveis (5G) (3), a qual promete altíssima taxa de transmissão de

[1] Relatório de acompanhamento do setor de telecomunicações - Serviço Móvel Pessoal - telefonia celular e banda larga móvel. Segundo trimestre de 2019. Disponível em https://cloud.anatel.gov.br/index.php/s/wyJqiCFMtzgJFTI

[2] Digital 2020: Global Digital Overview. Disponível em https://datareportal.com/reports/digital-2020-global-digital-overview

dados e baixa latência (requisito importante para aplicações de tempo real). Com uma arquitetura baseada em serviços, o 5G possibilita a instanciação de componentes sob demanda, tornando-a flexível e escalar.

Em paralelo, Operadores de Redes Móveis (MNOs - *Mobile Network Operators*) apostam em tecnologias como a *Multi-access Edge Computing* (MEC) (4) para descentralizar o processamento de aplicações de rede, trazendo-o para junto às Estações Rádio Base (ERBs), agilizando assim a instanciação de funções da própria rede sob demanda. Para se adequar aos requisitos de infraestrutura demandados pela MEC, é previsto um investimento financeiro por parte dos MNOs a fim de suportar aplicações MEC que possam oferecer assistência aos Provedores de Serviços por meio de recursos relacionados à própria rede móvel.

Apesar dos esforços por parte tanto dos Provedores de Serviços, quanto dos Operadores de Redes Móveis, observa-se a falta de integração entre ambas as partes. Deste modo, todas as requisições realizadas pelos usuários, necessariamente devem percorrer toda a infraestrutura dos MNOs até o serviço do SP fora da infraestrutura dos MNOs, seja ele hospedado em um servidor *cloud* ou replicado em um nó de CDN mais próximo.

Dada lacuna apresentada no parágrafo anterior, surge então o seguinte questionamento: Como possibilitar uma melhor integração para que os esforços empregados pelos Provedores de Serviços e Operadores de Redes Móveis culminem em um menor tempo de resposta às requisições dos usuários, elevem sua qualidade de experiência e respeitando a privacidade do usuário?

Neste contexto, vislumbra-se a possibilidade de se transferir parte dos serviços dos SPs para dentro da infraestrutura dos MNOs, reduzindo assim o tempo de resposta às requisições dos usuários, ao aproxima-los dos serviços em execução, elevando sua QoE e abrindo caminho para um novo meio de gerar receita para os MNOs. Com tal internalização de serviços, surge a preocupação de harmonizar os três pontos a seguir:

- **Alocação de Recursos:** alocar de modo dinâmico os recursos computacionais sem que haja desperdício dos mesmos. Tal alocação deve considerar a necessidade de isolamento no uso dos recursos entre os SPs e, ainda, oferecer suporte adequado à tarifação;
- **Atenção às demandas dos serviços:** mobilizar a infraestrutura para atender as demandas dos serviços e aplicações dos SPs, entregando-os tão próximo dos usuários quanto necessário;
- **Privacidade do usuário:** garantir que informações sensíveis relacionadas aos usuários presentes na infraestrutura dos MNOs permaneçam preservadas e inacessíveis pelas aplicações internalizadas.

Assim, este trabalho apresenta *Join-Me*, um *framework* distribuído que permite que SPs transfiram módulos de seus serviços para dentro da infraestrutura dos MNOs. Com a gestão de tais módulos feita internamente pela rede, os serviços podem ser entregues tão próximos do usuário quanto necessário, contando com uma gerência dinâmica de recursos e respeitando a privacidade do usuário. Join-Me define, além dos componentes funcionais necessários, uma interface de integração para que SPs especifiquem, para cada um dos módulos de um serviço, a demanda por uso de recursos, pela qual serão tarifados adequadamente.

Para descrever a proposta Join-Me, esta dissertação está estruturada como se segue: O Capítulo 2 apresenta os trabalhos que serviram de fundamento para o desenvolvimento deste trabalho. O Capítulo 3 reúne os trabalhos relacionados e em seu final, apresenta uma comparação entre as soluções apresentadas por estes trabalhos e a proposta apresentada por esta dissertação. A arquitetura *Join-Me*, bem como seus componentes, são detalhados no Capítulo 4. O Capítulo 5 apresenta o caso de uso utilizado para demonstrar a aplicabilidade do trabalho proposto, bem como seu funcionamento e resultados obtidos. Por fim, o Capítulo 6 conclui esta dissertação e apresenta seus trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este Capítulo é destinado a apresentar uma visão geral sobre a evolução das redes móveis e discutir trabalhos e tecnologias utilizados na fundamentação desta dissertação.

2.1 EVOLUÇÃO DAS REDES MÓVEIS

Desde sua popularização até os dias atuais, as redes móveis têm mudado o estilo de vida dos seres humanos, trazendo suporte a novos recursos a cada nova geração. Um breve histórico das gerações de redes móveis já existentes é apresentado em (3), juntamente com algumas características como taxa de transferência, tipo de comutação utilizada, suporte a diferentes tipos de aplicações, etc.

Um breve resumo sobre as gerações de redes móveis já existentes, bem como suas principais características é apresentado a seguir:

- **1G**: surgiu no início dos anos 80, suportava apenas chamadas de voz e tinha uma taxa de transferência de até 2,4kbit/s. Uma de suas principais desvantagens era a falta de segurança, uma vez que não havia criptografia nas chamadas, possibilitando que as mesmas pudessem ser interceptadas por escâneres de rádio;

- **2G**: inicialmente com taxas de transmissão de até 10kbit/s, a segunda geração surgiu no final dos anos 90 e além de suportar chamadas de voz, passou a suportar os Serviços de Mensagens Curtas (SMS). Suportando o padrão Global System for Mobile Communications (GSM), possibilitou que usuários fossem identificados por meio de seus chips *Subscriber Identity Module* (SIM), o qual permitiu uma maior flexibilidade para os usuários, de modo que os mesmos pudessem carregar informações de assinatura e lista de contato de um dispositivo para outro;

- **2.5G**: com a adoção do padrão *General Packet Radio Service* (GPRS), a segunda geração das redes móveis passou a suportar a comutação por pacotes, possibilitando a comunicação com a Internet. Junto com o GPRS, a tecnologia *Enhanced Data rates for GSM Evolution* (EDGE) também foi adicionada para melhorar o GSM, o qual elevou a taxa de dados para até 144kbit/s;

- **3G**: surgiu ao final dos anos 2000, estabelecendo o primeiro marco na história das redes móveis. Em resposta às demandas dos serviços por alta velocidade, a *Universal Mobile Telecommunication System* (UTMS) foi escolhida como tecnologia para suceder o GSM, possibilitando taxa de transmissão de até 2Mbit/s, trazendo melhorias na Qualidade de Serviços (QoS) de voz e suporte a *Roaming* global. Com a utilização do *High Speed Packet Access* (HSPA), trouxe melhorias para infraestrutura acesso, o que possibilitou maior velocidade do tráfego de *downlink* e *uplink*, *High*

Tabela 1 – Evolução das redes móveis - Adaptado de (3)

Geração	Tecnologia de Acesso	Taxa de Transmissão	Frequência	Largura de Banda	Tipo de Comutação	Aplicações Suportadas
1G	Advanced Mobile Phone Service (AMPS) Frequency Division Multiple Access (FDMA)	2,4 kbit/s	800 MHz	30 KHz	Circuito	Voz
2G	Global System for Mobile Communications (GSM) Time Division Multiple Access (TDMA) Code Division Multiple Access (CDMA)	10 kbit/s	850/900/1800/1900 MHz	200 KHz 1,25 MHz	Circuito	Voz e Dados
2.5G	General Packet Radio Service (GPRS) Enhanced Data Rate for GSM Evolution (EDGE)	50 kbit/s 200 kbit/s		200 KHz	Circuito / Pacotes	
3G	Wideband Code Division Multiple Access (WCDMA) Universal Mobile Telecommunications Systems (UMTS) Code Division Multiple Access (CDMA) 2000	384 kbit/s	850/900/1800/1900/2100 MHz	5 MHz 1,25 MHz	Circuito / Pacotes	Voz, Dados, Vídeo Chamada
3.5G	High Speed Uplink / Downlink Packet Access (HSUPA / HSDPA) Evolution-Data Optimized (EVDO)	5 - 30 Mbit/s		5 MHz 1,25 MHz	Pacotes	Voz, Dados, Vídeo Chamada, Jogos Online e TV de Alta Definição
4G	Long Term Evolution (LTE - A) (Orthogonal / Single Carrier Frequency Division Multiple Access (OFDMA / SC-FDMA) Worldwide Interoperability for Microwave Access (WIMAX) - (Scalable Orthogonal Frequency Division Multiple Access (SOFDMA)) WIMAX Móvel	DL 3 Gbit/s UL 1.5 Gbit/s 100-200 Mbit/s	1,8 GHz 2,6 GHz 2,3 GHz, 2,5 GHz e 3,5 GHz inicialmente	1,4 MHz - 20 MHz 3,5 MHz, 7 MHz, 5 MHz, 10 MHz e 8,75 MHz inicialmente	Pacote	Jogos Online, TV de Alta Definição

Speed Downlink Packet Access (HSPDA) e *High Speed Uplink Packet Access* (HSUPA), respectivamente. Com esta evolução, surgiu o 3.5G, possibilitando o acesso a uma gama de serviços como os de Vídeos sob Demanda (VoD), compartilhamento de arquivos ponto a ponto e serviços Web;

- **4G**: a quarta geração das redes móveis foi marcada pela utilização da tecnologia *Long Term Evolution* (LTE), a qual buscou manter a compatibilidade com tecnologias de gerações passadas (GSM e HSPA) e elevou a taxa de transmissão, chegando a 200Mbit/s, graças a utilização de *Multiple Input Multiple Output* (MIMO), multiplicando a capacidade dos *links* de rádio. O 4G consolidou a convergência das redes móveis para redes sobre *Internet Protocol* (IP) e possibilitou o consumo de jogos *on-line*, vídeo conferências e conteúdo televisivo em alta definição para dispositivos móveis.

A Tabela 1 apresenta uma síntese das principais informações sobre cada geração de rede móvel. Por serem também termos mercadológicos, principalmente a partir do 3G, as denominações de cada geração e seus respectivos avanços podem se apresentar na literatura com pequenas diferenças de definição.

2.1.1 5G: A quinta geração das redes móveis

Com o surgimento de novas aplicações e a crescente demanda dos usuários por uma maior qualidade de serviço, a Quinta Geração das redes móveis surge com diversas melhorias oferecendo uma altíssima velocidade de transmissão e baixa latência. O 5G busca lidar com o aumento do tráfego de dados provocado pela crescente popularidade das aplicações já presentes no 4G e pela introdução de tecnologias emergentes como IoT, Vigilância Pública e Redes Veiculares.

2.1.1.1 Arquitetura

Uma das inovações do 5G, e de particular interesse para o presente trabalho, é a adoção de uma *Software Based Archtecture* (SBA) (6) para o (5G *Core*). SBA permite a instanciação de seus componentes sob demanda, tornando-a dinâmica e flexível para lidar com os diferentes volumes de tráfego das diversas aplicações.

Os componentes são genericamente denominados *Network Functions* (NF) e se comunicam por meio de *Service Based Interfaces* (SBI). Um dos NFs, o *Network Repository Function* (NRF) é o componente responsável por permitir que os demais componentes publiquem e descubram outras funcionalidades da rede.

A função de rede *Access and Mobility Management Function* (AMF), gerencia o ingresso e o egresso de dispositivos finais. Como já mencionado, o 5G tem uma arquitetura flexível e dinâmica para lidar com situações adversas. Neste contexto, um algoritmo de escalabilidade da AMF é demonstrado no trabalho (7), a fim de suportar o alto número de dispositivos conectados (o que é esperado no 5G) sem que estes gerem *overhead* e congestionamento na rede.

O 5G tem em seu modelo arquitetural o conceito de *Network Slice* (8), o qual permite instanciar redes lógicas com alocação exclusiva de recursos dentro de uma mesma infraestrutura física. Após o acesso de um *User Equipament* (UE) à rede, o AMF aciona o *Authentication Server Function* (AUSF) para realizar sua autenticação e em seguida, o *Session Management Function* (SMF) se encarrega do estabelecimento da sessão.

As políticas de QoS e de controle do funcionamento da rede são gerenciadas pelo *Policy Control Function* (PCF) (9), o qual se apoia em informações unificadas presentes no *Unified Data Management* (UDM), como autenticação, autorização e demais dados de registro e perfil do usuário. Tais informações se juntam àquelas contidas no *User Plane Function* (UPF), como roteamento de pacotes, interconexão com *Data Network* (DN), *buffer* de dados e demais informações relacionadas à conexão do usuário, segundo o seu pacote de dados contratado. Além disso, as políticas são aplicadas respeitando os limites informados pelo *Application Function* (AF), de modo que o mesmo seja notificado em caso de falhas ou consumo de recursos próximo do limiar estabelecido.

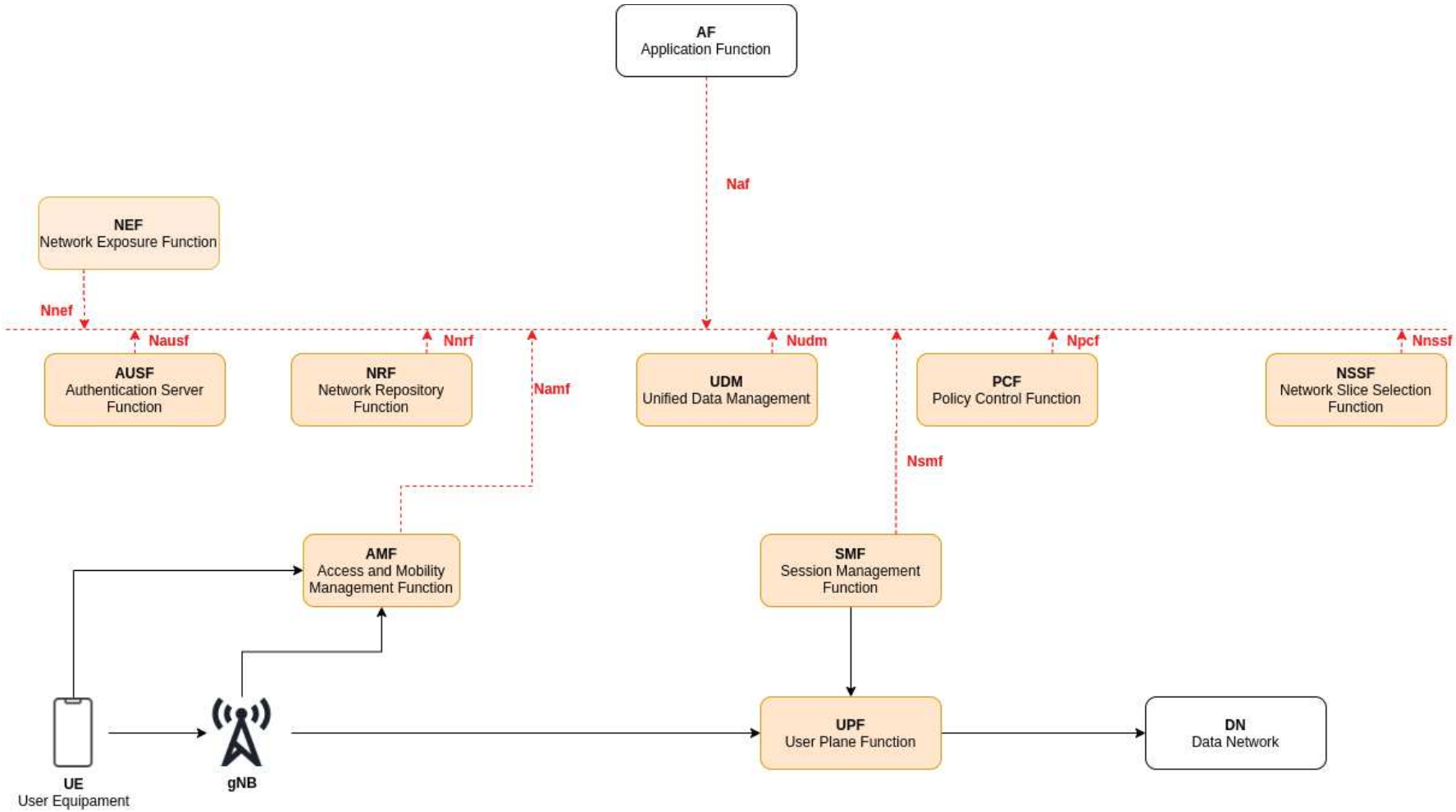
Assim como o NRF permite a descoberta de funções para os componentes internos da arquitetura do 5G, o *Network Exposure Function* (NEF) (10) atua como uma interface, expondo as funcionalidades da arquitetura por meio de *Application Programming Interfaces* (APIs), a fim de que sejam descobertas e tenham informações presentes em seus componentes utilizadas por AFs.

Uma AF por sua vez, pode interagir diretamente com a arquitetura 5G por meio do NEF. Tal ação possibilita, por exemplo, influenciar a tomada de decisão do UPF quanto ao roteamento do tráfego de dados do usuário, alterando seu local da borda da rede para

o núcleo do 5G ou mesmo possibilitando a interconexão com gerações de redes móveis anteriores, como o 4G.

Com base na aplicação em execução no UE, a AMF apresenta os requisitos de rede à função *Network Slice Selection Function* (NSSF), a qual é responsável por avaliar a viabilidade e efetuar a instanciação do *slice* de rede internamente à infraestrutura da rede 5G, a partir dos requisitos informados. A Figura 1 apresenta os componentes do 5G, além de suas interfaces de comunicação.

Figura 1 – Arquitetura do 5G *Core*



Fonte: adaptada de (11)

Nota-se que a adoção de uma arquitetura baseada em serviços, aliada a técnicas de virtualização e de programabilidade das funções de rede, vislumbra-se uma tendência de criação de novas APIs, que permitam interoperabilidade e integração dos serviços e informações providas por tais funções de rede. Assim, o *3rd Generation Partnership Project* (3GPP) inclui desde a *release* 15 a especificação técnica do *Common Application Program Interface Framework* (CAPIF) — atualmente na versão 16.9.0 (12).

CAPIF possui como objetivo padronizar o desenvolvimento e aspectos de segurança das APIs. Sua especificação possibilita que funções de diferentes tecnologias (4G, 5G, MEC) sejam publicadas, hospedadas, autenticadas e descobertas por demais serviços e aplicações de terceiros. O principal propósito é centralizar as APIs de modo que as mesmas possam expôr as funcionalidades e formas de integrar suas respectivas aplicações, permitindo que outros serviços as descubram e as invoquem por meio de um ponto central de acesso.

Até o momento, o CAPIF possui três principais componentes em sua arquitetura. O *CAPIF Core Function* (CCF) é o componente central da arquitetura, concentrando em si todas as especificações das APIs disponíveis e efetuando o gerenciamento aspectos de segurança, como autenticação e autorização, além de ser responsável pelo registro e faturamento. Já o *API Exposing Function* (AEF), é o responsável por expor as APIs armazenadas. Um exemplo de uma das APIs a ser exposta é a API NEF, presente na arquitetura 5G. Por fim, o *API Invoker* é um componente residente nos parceiros que tenham contrato estabelecido para uso do CAPIF. Ele é o responsável por interagir com CCF, a fim de realizar procedimentos de autenticação e com o AEF para acionamento de APIs presentes no CAPIF.

Um possível exemplo de utilização do CAPIF é atender o desenvolvimento de aplicações que têm por finalidade serem suportadas por múltiplas tecnologias, como por exemplo 4G e 5G. Uma vez que as respectivas estejam hospedadas pelo CAPIF, o desenvolvedor utiliza um ponto central de consulta das mesmas, sem a necessidade de realizar consultas em locais distintos.

2.2 EDGE COMPUTING

Os serviços providos por grandes centros de dados espalhados ao longo da Internet, repletos de servidores de alta performance, são popularmente chamados de *Cloud Computing*. O nome se deve ao fato de não se saber ao certo a localização do servidor responsável pelo armazenamento de um determinado dado.

Na literatura da área de computação, o termo *Cloud Computing* é defindo por (13) como:

> ... um conjunto de serviços habilitados para rede, fornecendo

> infraestruturas de computação escalonáveis, com QoS garantida, normalmente personalizadas e econômicas sob demanda, que podem ser acessadas de forma simples e abrangente. **tradução do autor**.

Ainda assim, a crescente demanda dos usuários por avanços em qualidade de serviço, experiência de usuário (UX) e aplicações cada vez mais realísticas, e de empresas demandando mais informações de seus clientes, vem impondo a necessidade de uma nova abordagem para complementar a *Cloud Computing*.

De fato, o avanço das redes móveis e dos dispositivos finais, que além de consumirem dados, passaram a gerá-los também, vem levando a um aumento ainda maior do tráfego de dados e da demanda por processamento em nuvem. Para lidar com essa mudança, a *Edge Computing* é definida em (2) como:

> ...a possibilidade de levar tecnologias que permitam o processamento para a borda da rede, próximo da origem dos dados. **tradução do autor**.

Em outras palavras, a *Edge Computing* possibilita a extensão do poder computacional já existente em *Cloud Computing*, levando-o para mais próximo dos dispositivos finais, o que impacta positivamente no tempo de resposta de processamento para aplicações que têm a baixa latência como um de seus principais requisitos.

Apesar da *Edge Computing* apresentar alguma semelhança à *Cloud Computing*, a descentralização dos servidores e a aproximação dos recursos computacionais junto à borda da rede, são dois grandes diferenciais, que ressaltam as vantagens da *Edge Computing*. São considerados dispositivos de borda qualquer dispositivo que forneça processamento, armazenamento e/ou manipulação de dados entre os servidores *cloud* e algum dispositivo de origem/destino dos dados. Um exemplo é a utilização de um *smartwatch* para monitorar um sensor cardíaco externo, como um marca-passo, pois ele agrega e processa os dados do sensor em preparação para a entrega em serviços em nuvem. Por outro lado, em uma situação em que um *smartwatch* não esteja manipulando dados de nenhum outro dispositivo externo, o mesmo é categorizado simplesmente como um dispositivo final, pois utilizaria seus próprios recursos (processamento, armazenamento...) para consumir e produzir dados, provenientes de servidores *cloud* e do próprio ambiente em que se encontra (condição climática, batimentos cardíacos e nível de estresse de seu dono...), respectivamente.

Existem, ainda, características exclusivas da *Edge Computing*, as quais ressaltam ainda mais a importância de sua adoção. Uma dessas características é a distribuição geográfica de seus servidores, possibilitando que o gerenciamento de serviços seja estrategicamente baseado na localização demandada, sem que todo o tráfego tenha que atravessar a Internet até os centros de dados. Neste contexto, análises de alto volume de

dados, bem como sistemas de tempo real, podem ser processados junto origem dos dados, proporcionando maior rapidez no processamento e entrega dos dados.

Uma outra característica da *Edge Computing* é o suporte à mobilidade. Devido ao rápido aumento na quantidade de dispositivos, é importante que estes sejam identificados por uma identificação única (*host*) e com base em sua localização, para que seja possível a comunicação direta entre a *Edge Computing* e os dispositivos finais. Com o uso de recursos na borda, pode-se suavizar a transição entre ERBs, possibilitando a continuidade na entrega de serviço. Além disso, a *Edge Computing* deve permitir que, dependendo do tipo de serviço ou aplicação, usuários tenham suas requisições atendidas pelo servidor de borda mais próximo, utilizando recursos de geolocalização presentes nos dispositivos finais ou baseado nos equipamentos de ponto de acesso ao qual o usuário final esteja conectado.

A experiência dos usuários pode ser elevada graças ao fornecimento de recursos computacionais e serviços mais próximos dos usuários. Ao mesmo tempo, provedores de serviços podem tirar proveito de informações acerca do comportamento dos usuários para que os serviços oferecidos sejam ainda mais aprimorados. A entrega de serviços na borda da rede, potencialmente, auxilia em uma redução na latência no acesso aos serviços. Dessa forma, *Edge Computing* pode ajudar no atendimento a um dos principais requisitos das aplicações sensíveis a retardo.

2.2.1 Multi-Access Edge Computing

No final de 2014, o *European Telecommunications Standards Institute* (ETSI) introduziu o conceito de *Mobile Edge Computing* (4), em meio às discussões de sobre uma nova geração de redes móveis. O conceito traz os benefícios da *Edge Computing* para os usuários e aplicações das redes móveis. Não se limitando à tecnologia de rede móvel da época (4G), vislumbrando a aplicação do conceito a outras tecnologias de acesso, como redes móveis futuras, WiFi e conexão fixa, o ETSI decidiu em 2017, alterar o nome para *Multi-Access Edge Computing* (MEC) (14). O intuito foi tonar o conceito da MEC mais inclusivo com relação às diferentes tecnologias às quais pode ser aplicado.

Com a introdução de poder computacional junto às ERBs, de forma a aliviar a sobrecarga no núcleo (*core*) da rede, os MNOs passam também a obter informações mais precisas sobre aspectos da rede, além de avançarem em seus procedimentos de gerência. Por exemplo, a obtenção de informações sobre a localização dos usuários ingressados à infraestrutura, o monitoramento de eventos relacionados às *Radio Access Networks* (RANs), além do ciclo de vida das aplicações MEC se tornam possíveis, graças ao conjunto de serviços presente na MEC e suas respectivas APIs (15), (16) (17).

Segundo (14), a ideia chave da *Multi-access Edge Computing* é:

> ... prover um ambiente de serviços de tecnologia da informação e computação em nuvem para a borda da rede, dentro da RAN e

nas proximidades dos clientes móveis. **tradução do autor**.

Dentre diversos fatores, um dos que mais contribuiu para o desenvolvimento da MEC foi a necessidade de baixa latência, um dos principais requisitos demandados pelas aplicações em execução na borda da rede. Além disso, alguns recursos presentes na MEC podem caracterizá-la com relação à sua finalidade. A proximidade dos usuários finais, por exemplo, permite definir uma maior precisão na localização de seus clientes baseada nas informações de sinal obtidas pelos seus dispositivos, para que serviços sejam instanciados em locais estratégicos, proporcionando uma melhor experiência para o usuário.

O modelo de referência arquitetural da MEC é composto por dois níveis: o nível de sistema e o nível de *host*. No nível de sistema, encontra-se o *MEC Orchestrator*, que mantém as informações referentes aos serviços e recursos (rede e computação) disponíveis dentro de uma infraestrutura MEC, além de alocá-los adequadamente para a instanciação das aplicações. Já no nível de *host* existe o Virtualization Infrastructure Manager (VIM), que é responsável pelo gerenciamento do ciclo de vida das aplicações e pelo faturamento e monitoramento dos recursos alocados.

2.2.2 MEC 5G

No contexto do 5G, graças à sua abordagem SBA, a MEC pode facilmente usufruir de várias funções de rede, desde que autorizado por meio da NEF. Isto permite fornecer serviços MEC aos seus usuários com base no plano de usuários presente na arquitetura 5G e em suas respectivas localizações.

Um usuário pode ser facilmente rastreável através da API de Localização (15), disponibilizada pelo *Location Service* (LS) da MEC. Este serviço permite que aplicações autorizadas requisitem informações (de forma anônima ou não) de um usuário específico que esteja com seu dispositivo associado a uma gNB (na especificação do *New Radio* 5G a ERB é denominada *Next Generation Node B* - gNB). O serviço de localização da MEC pode suportar tanto coordenadas geográficas quanto localização lógica na rede móvel, ou seja, através da identificação do rádio que esteja atendendo o usuário em questão. Além disso, é possível obter informações relacionadas não apenas aos usuários, mas também às zonas de cobertura e seus respectivos pontos de acesso.

Informações relacionadas às condições dos rádios podem ser obtidas por meio do *Radio Network Information Service* (RNIS) e sua respectiva API (16). O objetivo deste serviço é permitir que aplicações autorizadas sejam otimizadas com base no compartilhamento de informações da RAN, como por exemplo o *throughput* dos rádios, o qual pode interferir nos pré-requisitos de aplicações.

Com diversas aplicações sendo executadas paralelamente na borda da rede, é necessário que haja um gerenciamento efetivo da largura de banda disponível. Sem a

presença desta gerência, as aplicações iriam concorrer pela largura de banda sem a garantia de que seus pré requisitos sejam atendidos. O *Bandwith Management Service* (BWMS) possui uma API (17) para que aplicações autorizadas obtenham informações acerca da largura de banda disponível, bem como solicitar a alocação de uma porção da mesma, a fim de atender seus próprios requisitos.

2.3 NETWORK FUNCTION VIRTUALIZATION

Com o rápido avanço tecnológico, o ciclo de vida dos equipamentos responsáveis pelas funções de rede diminuiu consideravelmente, tornando onerosa a constante atualização da infraestrutura dos MNOs. Neste contexto, a *Network Function Virtualization* (20) possibilita o desacoplamento entre a função de rede e seu *hardware* dedicado, tornando possível sua implantação em servidores de propósito geral, por meio de tecnologias de virtualização, reduzindo, assim, as despesas de *Capital Expenditures* (CAPEX) e *Operational Expenditures* (OPEX).

Essa abordagem permite que funções de rede de diferentes fabricantes funcionem ao mesmo tempo no mesmo *hardware*, possibilitando a instanciação de funções e serviços de maneira ágil e escalável. Além disso, com as funções de rede sendo providas como *software*, é possível tornar mais granular as opções individuais de cada instância, a fim de elevar seu desempenho.

O ETSI (19) apresenta três principais elementos que compõem a arquitetura NFV. A *Network Function Virtualization Infraestructure* (NFVI), como o próprio nome já diz, é a infraestrutura composta pelos componentes de *hardware* e de *software* capazes de comportar e suportar a implementação das funções virtualizadas de rede.

Já as *Virtualized Network Functions* (VNFs) são todas e quaisquer funções de rede virtualizadas que possam ser implantadas e instanciadas através de recursos virtuais dentro de uma NFVI. Cada VNF possui internamente os componentes responsáveis pelo seu ideal funcionamento. Serviços como servidores DHCP, *firewalls* e roteamento, são exemplos comuns de VNF.

Para que as VNFs sejam provisionadas com suas demandas devidamente atendidas, o NFV *Management and Orchestration* (NFV MANO) é responsável por gerenciar os recursos físicos e virtuais presentes na infraestrutura, além do ciclo de vida das próprias VNFs. Além disso, o NFV MANO dispõe de interfaces de comunicação, as quais permitem a comunicação entre instâncias distintas e também funções de rede em equipamentos legados.

Partindo da mesmo conceito por trás da NFV, uma abordagem similar é apresentada em (21), na qual a virtualização é realizada não por meio de Máquinas Virtuais (VMs), mas sim por contêineres, o que eleva o desempenho das funções de rede devido à sua leveza

e agilidade na instanciação, se comparada à abordagem convencional.

Os benefícios trazidos pela NFV possibilitam que recursos presentes no 5G se tornem realidade, graças ao rápido provisionamento das funções de rede, o que impacta diretamente no tempo de respostas das aplicações. Um grande exemplo disso é a *Network Slice*, que pode ser dinamicamente instanciada, escalada e garantindo qualidade de serviço, de acordo com a demanda dos serviços e aplicações.

3 TRABALHOS RELACIONADOS

Este capítulo é destinado a apresentar trabalhos os quais apresentam soluções similares ou de mesmo interesse que esta dissertação. Em seu final, é apresentada uma tabela com as principais características dos trabalhos apresentados e confrontadas com a solução aqui proposta.

Com os benefícios trazidos pelas funções de rede virtualizadas, principalmente a rápida disponibilidade de novas instâncias e leveza dos contêineres, uma prova de conceito relacionada à migração de vNFs, como *firewalls*, é apresentada em (22). Essa abordagem permite que vNFs encapsuladas dentro de contêineres sejam migradas para diferentes localidades da borda da rede, a fim de aliviar o tráfego de dados em uma determinada localização que esteja sobrecarregada. Apesar da utilização das funções de rede baseadas em contêineres impactar positivamente na economia de recursos dos MNOs, é possível notar que o trabalho se limita apenas aos serviços provenientes do próprio MNO, não explorando a possibilidade de oferecer a leveza provida por esta abordagem para provedores de serviços.

Observando-se o sucesso da adoção da NFV para os MNOs, um novo leque de possibilidades se abre para que novas funcionalidades presentes nas redes sejam virtualizadas. Isso é comprovado em (23), no qual o conceito de *Anything as a Service* (AnyaaS) é definido como a possibilidade de se ter qualquer coisa como serviço e apresenta duas soluções, denominadas *Mobile CDN as a Service* (CDNaaS) e *Traffic Offload as a Service* (TOFaaS). Ambas são instanciadas sob demanda para atender pontos estratégicos da rede, a fim de evitar problemas como congestionamento e sobrecarga de link, permitindo, implicitamente, que a entrega de conteúdo esteja mais próxima do usuário.

No mesmo trabalho, é possível notar que os serviços instanciados e devidamente gerenciados pelo *AnyaaS Service Orchestrator* são exclusivamente de interesse dos MNOs, de modo que os provedores de serviços ficam limitados a se beneficiarem apenas das configurações providas pelos mesmos. Além disso, a alocação de recursos para a instanciação de um novo serviço, bem como a verificação da possibilidade de acomodação do serviço em um outro ponto da rede não são esclarecidas no trabalho.

Nota-se, também, que, apesar dos serviços serem migrados ou mesmo estendidos para novos pontos da rede, a qualidade de serviço entre estas instâncias é questionável, uma vez que não há menção sobre o assunto. A CDNaaS poderia utilizar de recursos de computação além de armazenamento, possibilitando o processamento de dados próximo ao usuário e potencializando a redução do tempo de resposta das requisições.

Os desafios enfrentados para prover a migração de serviços em redes IP são brevemente apresentados em (24), o qual traz o conceito de *Follow Me Cloud* (FMC), possibilitando que serviços sejam migrados entre centros de dados federados, de modo

que estes permaneçam em execução sem que haja interrupção em sua entrega. Para que, durante a migração, não haja o congelamento da sessão estabelecida inicialmente entre o cliente e um servidor específico, o trabalho apresenta como uma de suas contribuições um mapeamento de pontos de acesso e suas respectivas localidades. A abordagem sugerida tem como base o *Locator / Identifier Separation Protocol* (LISP) (25), para auxiliar na identificação da localização da sessão do serviço e a identificação do serviço em si, possibilitando a migração de uma maneira mais suave, mitigando problemas interrupção.

Para que o serviço seja identificado pela abordagem Follow Me Cloud, se faz necessária a instalação de uma aplicação de suporte no dispositivo do usuário final e nos servidores dos centros de dados. Sendo assim, qualquer requisição proveniente do usuário final tem uma espécie de sessão IP (comunicação *socket* origem e de destino) "substituída"pela identificação da sessão de serviço introduzida pelo trabalho. A identificação da sessão de um serviço é formada pela identificação do dispositivo final juntamente com a identificação do serviço solicitado, ambas fornecidas pelas aplicações de suporte instaladas. Embora a abordagem seja interessante, a necessidade do usuário final ter uma aplicação de suporte em seu dispositivo, levanta o questionamento no tocante à sua privacidade, a qual não é abordada pelo trabalho.

Um trabalho complementar ao FMC é apresentado em (26), que apresenta uma implementação baseada em *OpenFlow* como alternativa ao LISP. Em seu ambiente experimental, o trabalho descreve a presença de dois centros de dados que compartilham o mesmo espaço de armazenamento. Essa abordagem favorece o experimento, uma vez que não haveria uma migração física do serviço, impactando positivamente no tempo de migração. É interessante utilizar espaços de armazenamento de dados e serviços distintos para que o tempo de migração dos mesmos se aproxime da realidade, considerando que os centros de dados estejam geograficamente distantes.

A fim de aprimorar a tomada de decisão para a migração de serviços, um algoritmo baseado no processo de decisão de Markov (27) é apresentado em (28), e tem sua aplicabilidade comprovada analisando ambas as alternativas de implementação do *Follow Me Cloud* (24) (26). O algoritmo visa prever o deslocamento do usuário e, com base em métricas pré estabelecidas, como o custo de migração e distância entre usuário final e um novo centro de dados, elencar a nova localização para hospedar o serviço a ser migrado.

Com diversas aplicações sendo providas na rede, cada uma com seus respectivos requisitos para o ideal funcionamento, é esperada uma sobrecarga nos recursos dos MNOs, sejam eles na borda ou no núcleo da rede. Visando mitigar esses possíveis contratempos, a migração de serviços é também abordada em (29). Uma arquitetura baseada na MEC para ambientes 5G é apresentada para discutir as políticas de realocação e migração de aplicações e VNFs, a fim de possibilitar uma melhor utilização dos recursos para atender aplicações que demandam ultra baixa latência.

Segundo (29), uma aplicação é considerada um conjunto de VNFs criadas por meio de Virtual Machines (VM), com seus respectivos requisitos. Essas aplicações foram classificadas tendo como base a demanda mínima de latência, resultando em aplicações de tempo real, aplicações convencionais e aplicações híbridas. As aplicações de tempo real são aquelas que possuem a latência como quesito crucial para seu funcionamento. Já as aplicações convencionais, como o próprio nome diz, são as que suportam um certo nível de latência. E as híbridas consistem em um conjunto de VNFs dividido em dois grupos, um tendo a latência como quesito crucial e o outro tolerável à latência.

Devido à limitação de recursos presente nos servidores MEC, (29) define que aplicações tolerantes à latência devem, se necessário, ser migradas para uma localidade distinta (outro servidor MEC ou servidores *cloud*), a fim de garantir que os requisitos de latência das aplicações de tempo real sejam respeitados. Já as aplicações hibridas podem ser instanciadas parcialmente tanto nos servidores *cloud*, como nos MEC. No núcleo da rede, junto aos servidores *cloud* está localizado um *NFV Orchestrator*, responsável por aceitar ou rejeitar as requisições para instanciar novas aplicações, bem como realocá-las, segundo os requisitos de recursos computacionais. Além disso, é também responsável por realizar a migração das aplicações quando necessária e definir o local ideal para a acomodação das mesmas.

Apesar de apresentar uma preocupação com os requisitos das aplicações e realizar sua migração de acordo com a disponibilidade dos recursos dos MNOs, nota-se que a abordagem proposta não é aproveitada para instanciar as aplicações em locais estratégicos na borda da rede, o que possibilitaria efetuar a entrega de serviços o mais próximo possível do usuário.

Como apresentado neste Capítulo, há diversos trabalhos com o objetivo de alocar sob demanda os recursos computacionais e agilizar a entrega de serviços, impactando positivamente na QoE do usuário em redes móveis. Entretanto, observa-se que não há uma proposta de solução única que inclua i) a preocupação com os requisitos das aplicações, de modo que estes sejam supridos pela infraestrutura dos MNOs; ii) a alocação dos recursos de modo dinâmico, ou seja, instanciar, funções de rede e alocar recursos de comunicação e computação sem que os mesmos fiquem ociosos, provocando desperdício indesejado de recursos; e iii) a preocupação com a privacidade do usuário, de modo que informações sensíveis presentes nos MNOs permaneçam inacessíveis por demais aplicações.

A solução proposta no presente trabalho contempla de maneira unificada atender os requisitos demandados por cada aplicação, a fim de garantir seu ideal funcionamento. A alocação dos recursos é efetuada de maneira dinâmica, evitando o desperdício dos mesmos e os liberando em caso de ociosidade. Utilizando da técnica de *network slice*, as aplicações do próprio MNO, juntamente com as informações manipuladas permanecem restritas a um *slice* exclusivo, evitando que aplicações terceiras obtenham quaisquer informações neste

contexto.

A tabela 2 apresenta um resumo dos trabalhos apresentados nos Capítulos 2 e 3 e os pontos cobertos por cada um. É importante ressaltar que o termo "migração de serviços"é utilizado na tabela para referir a qualquer tipo movimentação de serviços ou funções de rede para uma outra localidade da rede.

Tabela 2 – Resumo dos Trabalhos dos Capítulos 2 e 3

	ALOCAÇÃO DE RECURSOS	PRIVACIDADE DO USUÁRIO	INTEGRAÇÃO COM SPs	VIRTUALIZAÇÃO	SLICING	MIGRAÇÃO DE SERVIÇOS	APLICABILIDADE
5G (6) (7) (8) (10)	SIM	SIM	-	SIM	SIM	-	Serviços MNO
MEC (14)	-	SIM	-	SIM	SIM	-	Serviços MNO
NFV (20),(21)	-	SIM	-	SIM	-	SIM (21)	Serviços MNO
AnyaaS (23)	-	-	-	SIM	-	SIM	Serviços MNO
Follow-Me Cloud (24) (26) (28)	SIM	-	-	SIM	-	SIM	Serviços MNO / SP
Migração de App. e VNF (29)	SIM	-	-	SIM	-	SIM	Serviços MNO
CAPIF (12)	-	SIM	-	-	-	-	Descoberta de Serviços (MNO)
Join-Me (este trabalho)	SIM	SIM	SIM	SIM	SIM	SIM	Serviços MNO / SP

4 O FRAMEWORK JOIN-ME

Como apresentado na Capítulo 2, houve uma evolução tecnológica das redes móveis e computação em nuvem. Consequentemente, novas abordagens para lidar com os requisitos de serviços e aplicações, além da demanda dos usuários, foram surgindo ao longo dos últimos anos. No entanto, é notória a falta de uma integração entre MNOs e SPs, pela qual vislumbra-se a possibilidade de melhorar a QoE das aplicações dos SPs.

O termo *Join-Me* surge com um convite dos MNOs para que os SPs internalizem módulos de suas aplicações dentro da infraestrutura das redes móveis, possibilitando assim a entrega de serviços mais próxima dos usuários, sempre que demandado. Dessa forma, abrem-se aos SPs não somente os recursos de núcleo e de borda da rede de um MNO, mas também a negociação de garantias de uso desses recursos, que também virão a contribuir com melhoria na QoE.

Um módulo é qualquer tarefa auto-contida do *workflow* de um SP, e pode tomar a forma de um componente de software, contêiner, máquina virtual, ou outra estruturação de fácil assimilação e instanciação pela infraestrutura de um MNO. É importante mencionar que nem todo o *workflow* de um SP precisa ser internalizado em MNOs de interesse. Além disso, mesmo que internalizado, cada módulo pode ser instanciado em partes diferentes da infraestrutura de um MNO, com base na parametrização definida no momento de sua internalização.

A partir da internalização dos módulos de um SP, a gerência e administração dos mesmos passam a ser de responsabilidade única e exclusiva do MNO. Tal gerenciamento independente contribui com a proteção da privacidade dos usuários da rede móvel. Além disso, *Join-Me* se apoia na técnica de *Network Slicing* (8) para prover o isolamento entre as aplicações internalizadas pelos SPs e os serviços provenientes da própria infraestrutura, de modo que informações sensíveis aos usuários não sejam acessadas pelas instâncias dos módulos dos SPs.

O *Framework* em questão conta com um conjunto de APIs responsável por permitir a integração entre os SPs e MNOs. No momento da internalização de seus módulos, os SPs poderão customizar os recursos a serem alocados pelo MNO, visando garantir que os requisitos de suas aplicações sejam atendidos. Após tal procedimento, o MNO mobilizará sua infraestrutura para lidar com a aplicação em questão e, junto com informações relacionadas à localização do usuário, efetuar a entrega em conformidade com as políticas de alocação de recursos e de proximidade do usuário requeridas pelos SPs para cada um de seus módulos.

4.1 ARQUITETURA

A arquitetura do framework Join-Me é projetada para funcionar de maneira distribuída, de modo que enquanto uma parte de seus componentes é implementada no núcleo da rede, a outra é implementada na borda da rede, junto às ERBs. Sua implementação é flexível e independente de tecnologia, possibilitando que os MNOs possam implantá-la a qualquer momento, adaptando-a às tecnologias presentes em sua infraestrutura ou mesmo às gerações de redes móveis futuras.

Especificamente, em um contexto de implantação de *Join-Me* em uma rede 5G, é esperada a integração entre *Join-Me* e a NEF (10), a fim de obter informações relacionadas aos usuários, como autenticação, autorização e políticas de controle baseados em seu pacote de dados contratado.

Além das informações providas pela arquitetura do 5G, é previsto também que *Join-Me* se beneficie de informações disponibilizadas pelo conjunto de APIs presentes na MEC 5G (15), (16) e (18). Essas informações são de suma importância para que os serviços e aplicações dos SPs sejam entregues de forma eficiente. Informações relacionadas ao histórico de deslocamento dos usuários estão presentes na infraestrutura dos MNOs e podem ser acessadas por meio do serviço de localização (15) da arquitetura MEC. Uma vez que esta integração seja possível, os recursos podem ser previamente alocados e os módulos dos SPs instanciados, dada a predição da rota do usuário segundo as informações obtidas.

4.1.1 Componentes da Arquitetura

Os componentes que integram a arquitetura Join-Me estão ilustrados na Figura 2. A figura evidencia a distribuição dos componentes Join-Me entre núcleo da rede e borda da rede. O fundo verde é usado para delimitar o escopo da arquitetura, enquanto o fundo amarelo delimita as infraestruturas do *5G Core* e da MEC, tendo em mente uma integração de Join-Me com a geração atual de redes móveis. Por serem definidos como parte de um *framework*, os componentes de Join-Me podem ser adaptados se integrarem a funções análogas das próximas gerações de redes móveis.

Para que um SP usufrua dos serviços providos pelo *Join-Me*, é necessário que o mesmo esteja devidamente cadastrado no MNO. Deste modo, o *Service Provider Account Manager* é responsável por admitir novos SPs juntamente com seus respectivos usuários operadores. Estando cadastrado no MNO, o SP pode dar continuidade em sua interação, se autenticando por meio do componente *Service Provider Authentication*, que verifica as credenciais de acesso, bem como os privilégios do usuário operador, aplicando todos os mecanismos de segurança necessários.

Join-Me fornece um conjunto de APIs denominado *Join-Me API*, o qual permite

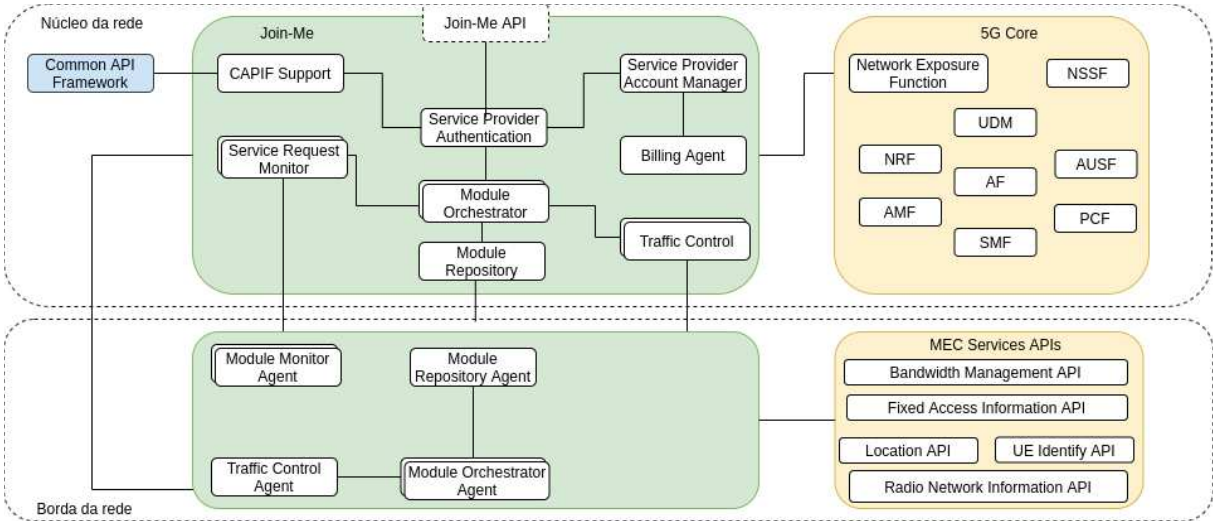


Figura 2 – Arquitetura *Join-Me* e seus respectivos módulos

que o SP, uma vez autenticado e autorizado, interaja com a infraestrutura, efetuando a internalização e acompanhamento do estado dos módulos. É possível obter informações acerca dos recursos consumidos, além de fazer a manutenção dos seus módulos (e.g. mudanças de estado, atualizações de código, atualizações de especificações de recursos). *Join-Me API*, bem como suas funcionalidades, encontram-se detalhadas na Seção 4.2.

No processo de internalização dos módulos na infraestrutura do MNO, recursos de armazenamento e comunicação são utilizados, uma vez que há a necessidade de transferir os dados dos módulos e armazenar suas respectivas imagens localmente. As imagens dos módulos são armazenadas e gerenciadas pelo *Module Repository*, deixando-as disponíveis para a execução no núcleo ou na borda da rede, conforme requerido pelo SP.

Quando o SP ativar sua aplicação, instâncias iniciais dos módulos são criadas e passam a consumir recursos de processamento, memória e comunicação, também em conformidade com a alocação demandada. Tal consumo de recursos é contabilizado pelo componente *Billing Agent*, que disponibiliza relatórios de consumo, juntamente com o valor cobrado correspondente.

É importante ressaltar que a abordagem de faturamento de recursos consumidos de Join-Me tem o potencial de expandir a cadeia de valor de telecomunicações. Abre-se a possibilidade de que não apenas SPs consumam recursos, mas também MNOs consumam recursos de outros, e tal integração pode fomentar a criação de federações de MNOs.

Visando a interoperabilidade com demais serviços e arquiteturas presentes nas redes móveis, o *CAPIF Support* é responsável por disponibilizar o conjunto de APIs *Join-Me* para que sejam descobertas por serviços terceiros em um ambiente de rede 5G. Dado esse suporte ao CAPIF, é possível que novos SPs, tecnologias legadas e demais serviços, tenham interesse em utilizar as funcionalidades fornecidas por *Join-Me*.

O componente *Common API Framework* em azul, apenas referencia a ligação da arquitetura *Join-Me* com o CAPIF, uma vez que o mesmo se encontra no núcleo da rede junto às tecnologias diversas, funciona de modo independente, com objetivo único e exclusivo de ser um repositório centralizado de APIs.

Quando um dispositivo final, que está conectado à infraestrutura de um MNO com suporte a *Join-Me*, requisita um determinado serviço, o componente *Service Request Monitor* intercepta esta requisição, obtendo como informação o endereço da ERB à qual o dispositivo final está associado e o endereço de destino (serviço/aplicação solicitada). Com base nas informações passadas no momento da internalização, o *Module Orchestrator* ou *Module Orchestrator Agent* é acionado, caso o módulo demande ser instanciado no núcleo ou na borda da rede, respectivamente.

Ao ser acionado, o *Module Orchestrator* consulta no *Module Repository* a imagem correspondente ao módulo a ser instanciado. Posteriormente, aloca os recursos pré determinados no momento da internalização e coloca o módulo em execução.

Caso o módulo deva ser executado na borda da rede, o *Module Orchestrator Agent* é quem será acionado, que por sua vez consultará a imagem do módulo no *Module Repository Agent*. Por estar na borda, o *Module Repository Agent* possui menor capacidade de armazenamento e mantém como uma cache as réplicas da imagens de módulos recentemente usados. Na ausência da réplica de uma réplica demandada, o *Module Repository Agent* realiza a cópia da mesma a partir do *Module Repository* e então, o *Module Orchestrator Agent* coloca o módulo em execução com seus recursos demandados devidamente alocados.

Traffic Control e *Traffic Control Agent* são responsáveis pelo roteamento das requisições aos serviços para a borda e para o núcleo da rede, respectivamente. Além disso, realizam a alocação dos recursos de rede, obedecendo os requisitos de QoS fornecidos no processo da internalização dos módulos. Ambos componentes alternam a rota das requisições aos serviços, a fim de poupar recursos na borda da rede caso não haja usuários consumindo tais serviços. Uma vez que um novo usuário se conecte a uma ERB a qual já tenha o serviço requisitado em execução, as requisições deste usuário serão atendidas diretamente pelo *Module Monitor Agent*, sem a necessidade de ser interceptada pelos componentes presentes no núcleo.

Com função semelhante ao *Service Request Monitor*, o *Module Monitor Agent* se diferencia ao periodicamente verificar a atividade atual de um determinado serviço. Uma vez que um período preestabelecido de tempo seja atingido sem que nenhuma atividade seja detectada para o serviço, assume-se que não há mais nenhum usuário consumindo o mesmo. A partir de então, o *Traffic Control Agent* é acionado para alterar a rota do serviço para o núcleo, caso ele tenha essa configuração estabelecida no processo de internalização.

A fim de economizar os recursos na borda da rede, o *Module Orchestrator Agent* encerra a execução do módulo e libera os recursos respectivamente alocados, uma vez que

não há mais trafego destinado ao módulo na borda.

Caso um usuário esteja em processo de *handover* com destino a uma ERB na qual o serviço encontra-se em meio ao período de *timeout*, o mesmo passará a ter sua atividade na nova ERB, que deverá interromper o *timeout* e o serviço continuará em execução. Além disso, baseado em informações relacionadas ao deslocamento do usuário, o *Module Orchestrator* e os *Module Orchestrator Agents* podem realizar a pre alocação de recursos e instanciação do serviço para atender o usuário ao longo de seu deslocamento.

Dadas as funções dos componentes apresentados, é possível notar a preocupação e suporte de *Join-Me* à harmonização dos principais requisitos que nortearam sua concepção: i) a alocação de recurso: é executada pelo *Module Orchestrator* e *Module Orchestrator Agent* no momento da instanciação dos módulos e consequentemente no encerramento dos mesmos em caso de ociosidade na borda da rede; ii) a atenção aos requisitos dos serviços dos SPs: no momento da internalização dos módulos através da *Module Internalization API*, parte do componente *Join-Me API*, é fornecida uma gama de opções para a parametrização dos recursos a serem consumidos pelos módulos, possibilitando seu funcionamento ideal; e iii) privacidade do usuário: a gerência dos módulos é feita de forma independente pelo próprio MNO, não cabendo ao SP acesso direto a informações da infraestrutura. Além disso, do mesmo modo que um *slice* de rede é criado para cada serviço internalizado, os serviços internos ao MNO, juntamente com as informações dos usuários internamente manipuladas são isoladas em um *slice* exclusivo, garantindo que as mesmas permaneçam inacessíveis por aplicações e serviços terceiros.

4.2 JOIN-ME API

Para permitir a interação dos SPs com os MNOs e possibilitar a internalização e gerência dos módulos na infraestrutura, foi desenvolvido um conjunto de APIs denominado *Join-Me API*, que se encarrega de acionar a infraestrutura para alocar os recursos necessários e disponibilizar cada serviço para os usuários, mediante as parametrizações especificadas pelos SPs no momento da internalização. As APIs que compõem *Join-Me API* foram desenvolvidas seguindo as especificações da *OpenAPI Specification* (OAS) da *OpenAPI Initiative*[1] (OAI).

Apesar da especificação de recursos a serem alocados para atender as demandas das aplicações ser comumente notada em plataformas de nuvem pública, as APIs *Join-Me* oferecem recursos adicionais. Criação de *slices* de rede, definição de local de instanciação e migração de módulos entre núcleo e borda da rede, além de priorização no tráfego de pacotes dentro da infraestrutura, são algumas das funcionalidades providas pelo *Join-Me Framework*. Além disso, graças as informações obtidas por meio da comunicação entre

[1] OpenAPI Initiative (OAI) - https://www.openapis.org/about

Join-Me e MEC, é possível permitir que instâncias dos módulos "sigam"o usuário, assim como previsto em (24).

A comunicação com a API é realizada por meio do protocolo *Hyper Text Transfer Protocol* (HTTP) (30), transportando mensagens no formato *JavaScript Object Notation* (JSON) (31). Join-Me API utiliza o *Representational State Transfer* (REST) (32) como estilo arquitetural para explorar os recursos dos protocolos Web e se alinhar às APIs já desenvolvidas no 5G (10).

Os métodos do HTTP GET, POST, PUT e DELETE foram definidos para possibilitar as variações de manipulação específicos de cada API, possibilitando respectivamente, consultar, inserir, alterar e remover informações. Assim como a padronização dos métodos em questão, os códigos de retorno (*HTTP status*) 200 e 404 foram definidos para lidar, respectivamente, com sucesso e exceções nas operações. No caso de exceções, além do código genérico informando que ocorreu um erro durante uma determinada operação, um segundo código e mais informações relacionadas ao erro em questão são incorporados ao corpo da mensagem de resposta.

4.2.0.1 ***Service Creation API***

A partir do registro de um SP à infraestrutura, o mesmo passa a ter acesso às funcionalidades da Join-Me API. Antes da internalização dos módulos, um serviço deve ser criado na infraestrutura, a fim de associar os módulos que devem ser internalizados ao serviços em questão. Ao criar um serviço, um *slice* da rede também é criado, visando, principalmente, isolar a comunicação entre os módulos de mesmo contexto dos demais módulos de outros serviços ou de outros SPs. Uma vez que o *slice* esteja criado, o mesmo pode ser utilizado para a parametrização de QoS na comunicação dos módulos presentes neste *slice*. As características de um objeto do tipo *Service* podem ser observadas na Tabela 3, em meio à especificação resumida da *Service Creation API*.

A *Service Creation API* ainda permite que os SPs manipulem o funcionamento dos serviços, possibilitando que todos os módulos referentes ao serviço sejam finalizados, inciados ou reiniciados. Note que para os dois últimos casos, os módulos serão inicialmente disponibilizados no núcleo da rede. Isso vale para todo tipo de módulo, mesmo os que são especificados para instanciação na borda da rede. A instanciação na borda somente ocorrerá quando alguma solicitação ao serviço for identificada no núcleo da rede.

4.2.0.2 ***Module Internalization API***

A partir da criação de um serviço, o SP está apto a internalizar seus módulos e a *Module Internalization* API é a responsável por permitir que isto aconteça. No momento da internalização dos módulos, o SP tem a possibilidade de configurar diversos parâmetros que auxiliarão o MNO a mobilizar sua infraestrutura, a fim de atender os requisitos para

Tabela 3 – Especificação resumida da *Service Creation API*

Service Creation API		
Solicitação Método: POST Rota: https://{server}/{version}/services Conteúdo: Objeto Service		
Resposta Status: 200 Content-Type: application/json Conteúdo: Objeto Service		
Nome do Campo	Valor	Descrição
id	int	Identificador do Serviço (incluído apenas na resposta)
sp_id	int	Identificador do SP
name	string	Define o nome do serviço
fqdn	string	URL de acesso ao serviço
Outras Ações		
Método: GET Iniciar - https://{server}/{version}/services/{id}/start Parar - https://{server}/{version}/services/{id}/stop Reiniciar - https://{server}/{version}/services/{id}/restart Status - https://{server}/{version}/services/{id}/status Exibir configuração - https://{server}/{version}/services/{id}/getConf		
Método: PUT Alterar serviço - https://{server}/{version}/services/{id}		
Método: DELETE Remover serviço - https://{server}/{version}/services/{id}		

o ideal funcionamento dos módulos. A Tabela 4 apresenta a especificação resumida da *Module Internalization API*, incluindo a parametrização disponível e demais características do objeto *Module*.

Após a submissão de um módulo, o *Module Repository* realiza o *download* da imagem do módulo utilizando o valor apresentado pelo campo *"download_url"* e a armazena localmente. Em seguida, o *Module Orchestrator* aguarda o comando para instanciação individual do módulo ou de todos os módulos integrantes de um serviço, feita pelo SP através da *Module Internalization API* ou da *Service Creation API*, respectivamente.

Quando o campo *"is_edge"* é definido com o valor *"1"*, a arquitetura, por meio do *Service Request Monitor*, aguarda até que alguma requisição seja recebida, e então notifica a ERB mais próxima do usuário, onde o *Module Repository Agent* realiza a cópia da imagem do módulo a partir do núcleo da rede. Logo em seguida, o *Module Orchestator Agent* coloca o módulo em execução, dando continuidade à entrega do serviço ao usuário, dali em diante realizada na borda da rede.

Tabela 4 – Especificação resumida da *Module Internalization API*

Module Internalization API		
Solicitação Método: POST Rota: https://{server}/{version}/services/{id}/modules Conteúdo: Objeto Module		
Resposta Status: 200 Content-Type: application/json Conteúdo: Objeto Module		
Nome do Campo	Valor	Descrição
id	int	Identificador do Módulo (incluído apenas na resposta)
service_id	int	Identificador do serviço
sp_id	int	Identificador do SP
name	string	Define o nome do módulo
fqdn	string	URL de acesso ao módulo
port	int	Porta de acesso
is_exposed	int	Acessível ao usuário (0 - não, 1 - sim)
mem	int	Requisito de memória
max_mem	int	Máximo de memória permitida
cpu	int	Requisito de CPU
max_cpu	int	Máximo de CPU permitido
max_storage	int	Espaço máximo ocupado
download_url	string	Localização remota do módulo
is_edge	int	Acessível na borda? (0 - não, 1 - sim)
multi_users	int	Múltiplos usuários? (0 - não, 1 - sim)
max_users	int	Usuários atendidos (0 - ilimitado)
prior	int	Prioridade no tráfego de dados
share	string	Link espaço em disco x módulo
inclusion_date	date	Data de internalização do módulo
Outras Ações		
Método: GET Iniciar - https://{server}/{version}/services/{id}/modules/{id}/start Parar - https://{server}/{version}/services/{id}/modules/{id}/stop Reiniciar - https://{server}/{version}/services/{id}/modules/{id}/restart Status - https://{server}/{version}/services/{id}/modules/{id}/status Exibir configuração - https://{server}/{version}/services/{id}/modules/{id}/getConf		
Método: PUT Alterar módulo - https://{server}/{version}/services/{id}/modules/{id}		
Método: DELETE Remover módulo - https://{server}/{version}/services/{id}/modules/{id}		

Para evitar a massiva parametrização no momento da internalização do módulo, é possível disponibilizar *templates* de configurações pré definidas para módulos que requerem uma menor personalização. Além disso, configurações pré definidas podem ser uma opção

financeiramente mais atrativa para SPs e MNOs com um menor poder aquisitivo.

Assim como os serviços, é possível que os SPs manipulem individualmente cada módulo. No entanto, tal operação pode causar um mau funcionamento do serviço e deve ser utilizada com cautela. O *Join-Me* possibilita esta funcionalidade a fim de prover uma fácil atualização dos módulos, uma vez que imagens mais recentes sejam disponibilizadas. Uma vez que os serviços e seus respectivos módulos estejam criados e internalizados, torna-se possível obter informações acerca das configurações determinadas, como nome do *slice* de rede, níveis de QoS, dentre outras características de cada objeto.

Tanto o processo de criação de serviço, bem como o *workflow* de internalização do módulo na infraestrutura do MNO por meio das APIs *Join-Me* podem ser observados na Figura 3.

4.2.0.3 ***Billing API***

Uma vez que os módulos dos serviços dos SPs estejam devidamente instanciados e consumindo recursos providos pela infraestrutura do MNO, este consumo passa a ser contabilizado pelo componente *Billing Agent* da arquitetura *Join-Me*. A *Billing API* é responsável por prover o acesso aos relatórios financeiros relacionados aos recursos consumidos pelos seus serviços aos SPs.

Join-Me estabelece uma precificação pré definida para recursos como memória, processamento e armazenamento. Tais valores podem ser customizados e novos recursos podem ser disponibilizados e terem sua contabilidade implementada junto à API.

A *Billing API* deve ser capaz de atender às requisições gerando respostas contendo o relatório de consumo por serviço e também por módulo. A Tabela 5 apresenta as propriedades do objeto *Billing*, em meio à especificação resumida da *Billing API*.

O relatório de consumo por serviço apresenta informações de consumo de recursos do ponto de vista do serviço. Neste caso, é reportada uma visão geral sobre o serviço, incluindo a quantidade de módulos que o compõe, custo de consumo por recurso, juntamente com consumo por recurso utilizado, mensurados dentro de um período pré determinado. Por fim, o valor total de recursos consumidos pelo serviço é incluído.

Do ponto de vista de um módulo, é reportada a quantidade de instâncias em execução do módulo e seus respectivos consumos e custos. É importante salientar que não deve ser informada a localização do módulo instanciado, a fim de garantir a segurança da infraestrutura dos MNOs e a privacidade dos usuários.

Figura 3 – Processo de criação de serviço e internalização de módulos

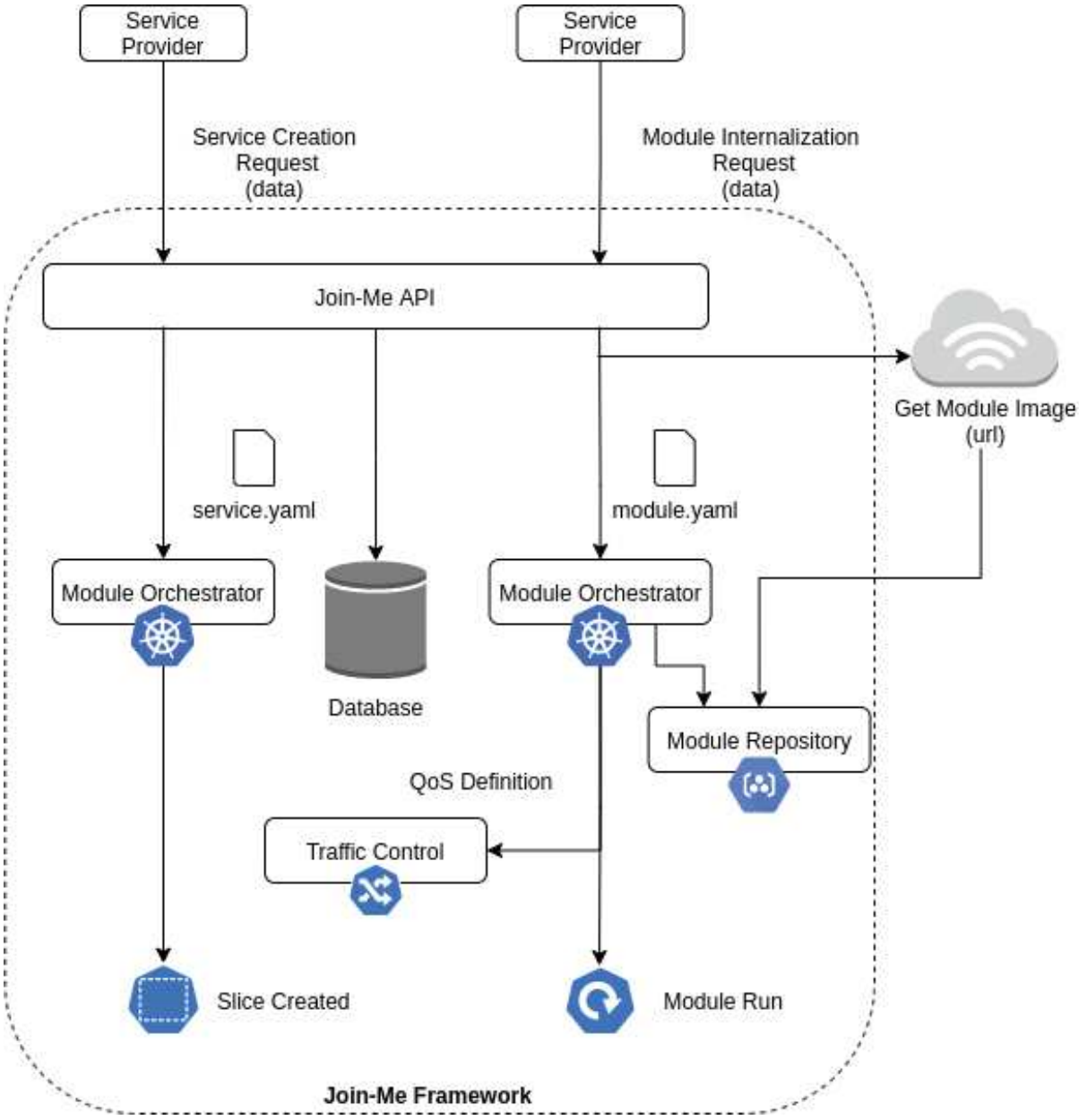


Tabela 5 – *Billing API*

Billing API		
Solicitação Método: GET Rota: https://{server}/{version}/services/{id}/billing		
Resposta Status: 200 Content-Type: application/json Conteúdo: Objeto Billing (Serviço)		
Nome do Campo	Valor	Descrição
id	int	Identificador do Relatório
sp_id	int	Identificador do SP
service_name	string	Define o nome do serviço
modules_count	int	Qtde de módulos no serviço
report_starts	date	Período do relatório (inicio)
report_ends	date	Período do relatório (fim)
cpu_consumption	int	CPU consumido (milicore)
memory_consumption	float	Memória consumida
storage_consumption	float	Armazenamento consumido
avg_bandwidth_consumption	float	Vazão média de dados consumido
data_transferred	float	Total de dados transferidos
memory_cost	float	Custo de uso de memória
cpu_cost	float	Custo de uso de cpu
storage_cost	float	Custo do uso de armazenamento
data_transfer_cost	float	Custo pela transferência de dados
total_cost	float	Custo total

Billing API (Continuação)		
Solicitação Método: GET Rota: https://{server}/{version}/services/{id}/modules/{id}/billing		
Resposta Status: 200 Content-Type: application/json Conteúdo: Objeto Billing (Módulo)		
Nome do Campo	Valor	Descrição
id	int	Identificador do Relatório
sp_id	int	Identificador do SP
service_name	string	Define o nome do serviço
module_name	string	Define o nome do módulo
instances_count	int	Qtde de instâncias ativas
report_starts	date	Período do relatório (inicio)
report_ends	date	Período do relatório (fim)
cpu_consumption	int	CPU consumido (milicore)
memory_consumption	float	Memória consumida (MB)
storage_consumption	float	Espaço consumido (MB)
avg_bandwidth_consumption	float	Vazão média de dados consumido
data_transferred	float	Total de dados transferidos
memory_cost	float	Custo de uso de memória
cpu_cost	float	Custo de uso de cpu
storage_cost	float	Custo do uso de armazenamento
data_transfer_cost	float	Custo pela transferência de dados
total_cost	float	Custo total

5 CENÁRIOS DE CASO DE USO E PROVA DE CONCEITO

Este Capítulo tem por finalidade introduzir alguns cenários em que *Join-Me* pode ser implantado. Por fim, é apresentada uma Prova de Conceito (PoC) à qual o *framework* foi submetido visando demonstração de sua funcionalidade e resultados.

5.1 CENÁRIOS DE CASO DE USO

5.1.1 Aquisição e distribuição de conteúdo em locais super populosos

Grandes eventos são comumente transmitidos pela Internet por meio de empresas credenciadas pelo próprio evento. É o caso de shows, finais de campeonato de futebol, exposições, onde além de ter uma transmissão oficial, há também um grande volume de pessoas que produzem e consumem o conteúdo, seja ele oficial ou não oficial.

A fim de realizar a cobertura de um determinado evento, a empresa credenciada pode previamente solicitar que recursos sejam alocados nas gNBs[1] próximas ao local do evento, visando efetuar o pré processamento do conteúdo capturado. Em paralelo, módulos poderiam ser instanciados junto à borda para realizar a segmentação e entrega do conteúdo a usuários no local do evento.

É importante ressaltar a necessidade de processamento em tempo real próximo ao evento, ou seja, junto às gNBs a fim de atender os requisitos computacionais para transcodificação e segmentação em múltiplas resoluções. Além disso, aplicar políticas de QoS na comunicação entre os módulos instanciados pela empresa credenciada, bem como pelos módulos provenientes de provedores de serviços terceiros, responsáveis pela criação e consumo de conteúdo não oficiais.

5.1.2 Caching e adaptação de Videos on-Demand - VoD

Grandes empresas provedoras de conteúdo multimídia disponibilizam constantemente obras cinematográficas para entreter seus assinantes. Estes, por sua vez, consomem tais conteúdos no conforto de suas casas, no deslocamento para o trabalho ou mesmo no próprio trabalho. Dados estes distintos ambientes, mecanismos para adaptação da resolução de vídeo baseado na condição da rede do usuário, como *HTTP Live Streaming* (HLS) e *Dynamic Adaptative Streaming over HTTP* (DASH) são executados nos *players* dos dispositivos finais a fim de evitar a ocorrência interrupções na reprodução de mídia contínua.

Para que a mídia seja exibida em diferentes resoluções para o usuário, diferentes instâncias de processos de segmentação de vídeo (cada uma com sua respectiva resolução)

[1] Neste capítulo de cenários de uso e prova de conceito, é usada a denominação gNB (*Next Generation Node B*) dada pelo *New Radio* 5G à ERB

deve ser executada no servidor da empresa provedora do conteúdo e posteriormente, estes segmentos são distribuídos pelas CDNs contratadas e então entregues aos usuários.

Com a adoção de *Join-Me*, a segmentação e o armazenamento do conteúdo passam a ser realizados na borda da rede, possibilitando ampliar o alcance das CDNs. A partir de uma estratégia previamente definida, torna-se possível que o conteúdo seja armazenado e entregue por servidores de borda considerados ótimos para grupos de assinantes com interesses em comum.

Utilizando da mesma estratégia, é possível oferecer um melhor suporte à mobilidade do usuário, de modo que auxiliado por informações dos usuários contidas nos MNOs, juntamente com suas preferências por determinados tipos de conteúdos presentes nestas empresas, o armazenamento de conteúdo seja distribuído ao longo da rota de deslocamento efetuada pelo usuário cotidianamente.

5.1.3 Telemedicina interativa com vídeo em ultra alta resolução

No cenário pandêmico vivenciado no ano de 2020, atividades remotas como vídeo conferência e ensino à distância (EAD) têm sido adotadas como alternativas para que atividades profissionais e educacionais sejam mantidas e ao mesmo tempo garantir a segurança dos envolvidos, respeitando o distanciamento social. Neste contexto, a formação de novos profissionais da saúde, bem como as discussões de casos clínicos por especialistas, os quais muitas vezes residem geograficamente distantes uns dos outros, se tornam imprescindíveis para auxiliar tanto na recuperação dos enfermos, quanto nos estudos e pesquisas.

Join-Me pode possibilitar que especialistas geograficamente distantes participem de modo passivo de cirurgias remotas, de modo a interagir com o vídeo em busca de detalhes relevantes em seu diagnóstico e orientação. Em uma da possíveis formas de implantar tal serviço, no local do evento (sala de discussão de casos, centro cirúrgico...), uma câmera de ultra-alta resolução (4K/8K) realiza a captura do vídeo e o envia em tempo real para o núcleo da rede de um MNO, onde módulos de retransmissão, segmentação e entrega estão instanciados, em uma espécie de *workflow*. Enquanto recebe o fluxo de vídeo, o módulo de retransmissão o encaminha em *multicast* dentro da infraestrutura. Em paralelo, o módulo de segmentação recebe o fluxo em *multicast* e o segmenta, juntamente com o processamento de *downscale*, para uma resolução mais adequada à entrega, em 1080p. O módulo de entrega fica responsável pela entrega final dos segmentos ao usuário móvel.

Quando um usuário iniciar uma interação com o vídeo, como, por exemplo, ampliar a visualização de uma determinada região, os módulos de segmentação e entrega serão instanciados também na gNB à qual o dispositivo final do usuário está associado. No entanto, ao receber o fluxo *multicast*, o módulo de segmentação realizará a segmentação do vídeo com sua área de visualização cortada (*cropped*), conforme seleção do usuário. Assim, possibilita-se a visualização da área selecionada em alta definição, sem que o

dispositivo final tenha que receber o vídeo de ultra-alta definição e tenha que fazer todo o processamento localmente.

5.2 PROVA DE CONCEITO

Visando demonstrar o funcionamento e viabilidade de *Join-Me*, foi desenvolvida uma Prova de Conceito (PoC) abordando o caso de uso descrito na Seção 5.1.3. O propósito desta PoC não é simular com exatidão os componentes presentes em redes 5G ou na MEC 5G, mas, sim, demonstrar de forma geral o comportamento da rede e algumas das funcionalidades que *Join-Me* proporciona, tanto para os SPs quanto para os MNOs. Ainda assim, é possível conduzir uma análise dos resultados atingidos, para melhor expor o potencial ganho da adoção do *framework*.

O cenário da PoC ilustrado pela Figura 4 conta com um servidor central, representando o núcleo da rede, e gNBs espalhados na borda da rede. Cada gNB é composta por um servidor de menor capacidade conectado à um ponto de acesso móvel. Para simplicidade do ambiente de experimentação em laboratório, este foi implantada como uma rede móvel baseada em IEEE 802.11n (33).

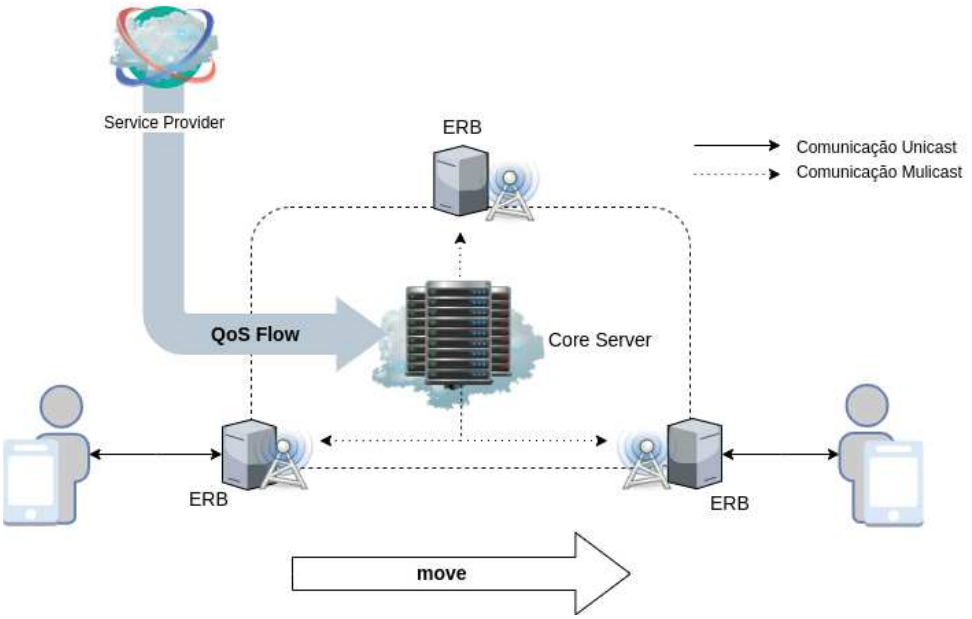


Figura 4 – Cenário Prova de Conceito - *Join-Me*

Para o funcionamento dos componentes da arquitetura *Join-Me*, foram utilizadas algumas ferramentas já existentes. No núcleo da rede, o Kubernetes[2] foi escolhido para atuar como o *Module Orchestrator* e consequentemente, os módulos foram internalizados sob a forma de contêineres *Docker*[3]. O *Service Request Monitor* foi incorporado por meio da

[2] Kubernetes Container Orchestrator - https://kubernetes.io/docs/home
[3] Docker Containers - https://www.docker.com/resources/what-container

ferramenta *HTTPry*[4] para monitorar as requisições a serviços através do protocolo HTTP. O *Traffic Control* foi desenvolvido em *Shell Script* (34) para lidar com o roteamento do tráfego das solicitações aos serviços. A fim de armazenar a imagem dos módulos internalizados, para o *Module Repository* foi utilizada a implementação do *Docker Registry*[5]. Por fim, o conjunto de APIs *Join-Me API* foi desenvolvido utilizando a versão 3 da linguagem de programação *Python* [6], juntamente com o seu *microframework Flask* [7]

Já na borda da rede, o *Module Orchestator Agent* foi implementado utilizando Kubernetes, tal como o *Module Orchestrator*. O *Module Repository Agent* utiliza o armazenamento local provido pelo *Docker* após efetuar a cópia dos módulos. O *Module Monitor Agent*, também desenvolvido em *Shell Script*, conta com recursos do IPTables[8] para monitorar as requisições aos módulos instanciados na borda. O *Traffic Control Agent*, também em Shell Script, controla o roteamento da borda para o núcleo da rede (detalhes na Seção 5.2.1). O protocolo *Secure Shell* (SSH) (35) foi escolhido para prover uma maior segurança na comunicação entre os componentes do núcleo e da borda da rede, e na transferência das imagens dos módulos.

Dado o cenário reduzido da PoC, o recurso de *Network Address Translation* (NAT) (37) foi utilizado nas gNBs para identificar a origem das requisições. Deste modo, o endereço de origem da requisição é traduzido para o endereço da gNB, o qual passa a ser monitorado pelo *Service Request Monitor* e utilizado no processo de roteamento pelo *Traffic Control* e *Traffic Control Agent*. Para a implantação do *framework* em ambientes maiores, se faz necessária a utilização de abordagens que ofereçam uma maior inteligência e autonomia no roteamento de tráfego, como *Software Defined Networks* (SDN)(36), visando por exemplo o roteamento de tráfego entre servidores de borda adjacentes, sem a necessidade da intervenção do servidor do núcleo da rede.

Os módulos desenvolvidos para a PoC são os seguintes. O módulo *sp-live-streaming* é o responsável por receber, em tempo real, o vídeo do procedimento cirúrgico, enviado ao núcleo da rede, e por retransmiti-lo, por meio do protocolo *User Datagram Protocol*(UDP) em *multicast*, dentro da infraestrutura do MNO. O módulo *sp-segmentation* é utilizado em três diferentes instâncias, sendo a primeira denominada *sp-segmentation-4k*, para realizar a segmentação do fluxo conforme recebido em *multicast*. A segunda instância, de nome *sp-segmentation-1080* realiza a transcodificação (com *downscale*) e segmentação do vídeo de visão geral em alta resolução. Já a terceira instância, denominada *sp-segmentation-zoom*,

[4] HTTP logging and information retrieval tool - https://github.com/jbittel/httpry
[5] The Docker Registry 2.0 implementation for storing and distributing Docker images https://hub.docker.com/_/registry
[6] Python is a programming language that lets you work quickly and integrate systems more effectively - https://www.python.org/
[7] Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications.
[8] The netfilter.org project - https://www.netfilter.org

realiza a transcodificação e segmentação do vídeo limitado à área selecionada pelo usuário. Por fim, o módulo *sp-delivery* realiza a entrega do vídeo segmentado ao usuário final.

É importante ressaltar que, para cada instância do módulo *sp-segmentation*, há uma instância do módulo de entrega *sp-delivery*, uma vez que cada segmentação foi realizada em local distinto na rede. Apenas o módulo *sp-segmentation-zoom* e seu respectivo módulo de entrega são instanciados na borda da rede. Sendo assim, ao solicitar o vídeo ampliado, tais módulos são instanciados na borda da rede, o que não acontece para a entrega do vídeo de visão geral, o qual permanece sendo entregue a partir do núcleo da rede.

Por se tratar de um serviço de *Streaming* Adaptativo ao vivo baseado em HTTP, não há uma preocupação com o estado da aplicação em execução, uma vez que o dispositivo do usuário final fica encarregado de efetuar diversas requisições do tipo HTTP/GET para cada segmento do vídeo a ser entregue. O controle da continuidade do *streaming* recebido é baseado em uma *playlist* baixada pelo dispositivo, a qual possui informações acerca dos segmentos disponíveis no servidor. Uma vez que novos segmentos são disponibilizados, a *playlist* é atualizada e o usuário final baixa esta atualização possibilitando a continuidade da cópia e reprodução dos novos segmentos.

5.2.1 Funcionamento

Partindo do pressuposto de que um determinado SP já esteja devidamente cadastrado e autenticado pelo *framework*, o mesmo realiza a criação do serviço de acompanhamento interativo de cirurgia remota por meio da *Service Creation API*.

Em seguida, incia o procedimento de internalização de seus módulos via *Module Internalization API*. Ao serem enviadas as solicitações de internalização de cada módulo mencionado na Seção anterior, os dados presentes em cada requisição são traduzidos para um arquivo no formato *YAML Ain't Markup Language* (YAML) (38). O componente *Module Orchestrator* recebe e interpreta tal arquivo, e, em seguida, aciona o *Module Repository* para efetuar a transferência inicial da imagem do módulo, desde o local definido pelo SP no campo *download_url* na respectiva solicitação de internalização.

Quando o SP iniciar o serviço, o *Module Orchestrator* instancia os módulos para execução, com os recursos devidamente alocados. Em caso de ausência de recurso disponível, a execução da instância do módulo não é iniciada e o SP é notificado.

Quando requisitos de QoS são definidos no momento da internalização de um módulo, o *Traffic Control* e *Traffic Control Agent* são acionados e as regras de priorização de tráfego são aplicadas. Isso é possível graças ao campo *Type of Service* (ToS) presente no cabeçalho do protocolo IP. Deste modo, o IPTables realiza a marcação do pacote relacionado ao serviço e aplica a prioridade informada no momento da internalização. Para garantir a neutralidade da rede relacionado ao tráfego de dados, as políticas de QoS são aplicadas apenas na comunicação entre os módulos, mantendo a entrega final do serviço

ao usuário final sob abordagem convencional de melhor esforço.

Para detalhar o funcionamento da plataforma do ponto de vista do usuário, a Figura 5 ilustra a interação do usuário com o serviço, e, de forma transparente, com componentes Join-Me. A figura ilustra, também, a comunicação entre os componentes para garantir a entrega do serviço acompanhando a movimentação do usuário, bem como gerenciando o uso dos recursos computacionais.

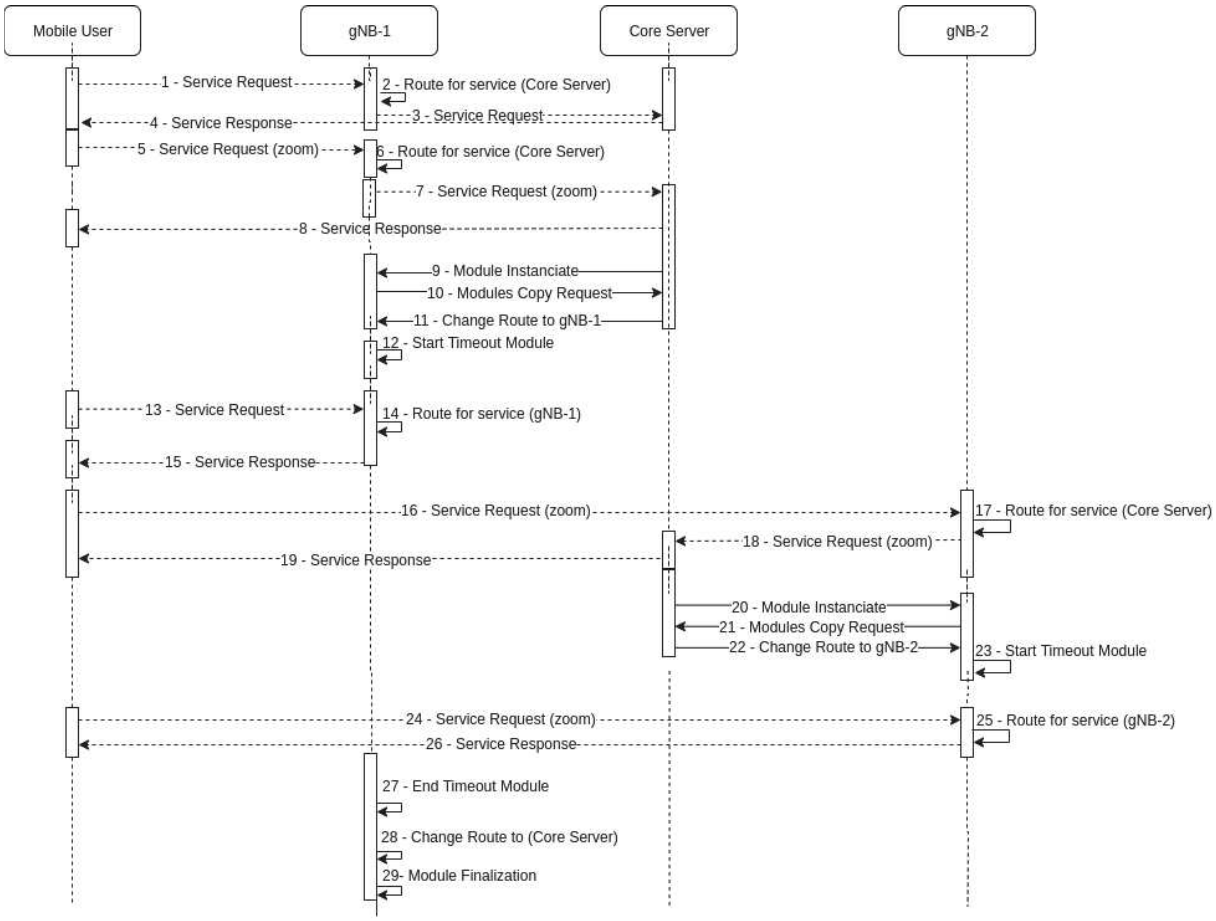


Figura 5 – *Workflow - Join-Me*

Uma vez conectado à infraestrutura do MNO através da gNB-1, o usuário solicita o acesso ao vídeo ao vivo, conforme o passo 1 na 5. Inicialmente, todos os serviços têm suas rotas definidas para o servidor no núcleo da rede, onde os módulos já se encontram instanciados e em execução. Deste modo, a requisição passa pela borda (passo 2) até chegar ao núcleo da rede, onde o *Service Request Monitor* inicia seu monitoramento e a resposta é devolvida ao usuário (passo 4).

Quando o usuário inicia sua interação com o vídeo, ou seja, solicita a ampliação de uma determinada área (passo 5), a requisição é atendida pelo núcleo da rede e (passos 7 e 8) o *Service Request Monitor* aciona o *Module Orchestrator Agent* na borda da rede, informando o serviço solicitado pelo usuário e o respectivo módulo a ser instanciado (passo 9). Recebendo esta informação, o *Module Repository Agent* é acionado para efetuar a cópia

da imagem do módulo presente no *Module Repository* e em seguida o módulo é colocado em execução (passo 10).

Estando os módulos em execução na borda da rede, o *Traffic Control* realiza a alteração da rota presente na gNB-1 para o módulo *sp-segmentation-zoom* e seu respectivo *sp-delivery* (passo 11), o qual passa a ser responsável pela entrega do vídeo ampliado ao usuário (passos 13 ao 15).

Quando o usuário realiza o processo de *handover* da gNB-1 para a gNB-2, o processo de instanciação dos módulos na borda da rede se repete (passos 16 ao 26), no entanto, como não há mais requisições na gNB-1, o *Module Monitor Agent*, responsável pelo monitoramento das requisições na gNB-1, inicia um processo de *timeout* (passo 12). Caso haja alguma solicitação durante este intervalo de tempo, o mesmo é interrompido e o serviço continua sendo entregue pela gNB-1. Entretanto, caso nenhuma requisição seja recebida, o processo de *timeout* iniciado no passo 12 se encerra (passo 27). O *Module Request Monitor* aciona o *Traffic Control Agent*, o qual altera a rota do serviço novamente para o servidor no núcleo da rede (passo 28) e posteriormente aciona o *Module Orchestrator Agent*, que se encarrega de finalizar a execução dos módulos e liberar os recursos alocados, conforme mostrado no passo 29.

É importante ressaltar que, durante o processo de *handover* e nos passos subsequentes, o serviço em questão é migrado entre os servidores de borda, possibilitando que o mesmo "siga"o usuário à medida em que ele se desloca. O trabalho (39), parte desta dissertação, apresenta a migração do serviço de *Streaming* adaptativo ao vivo entre núcleo e bordas da rede, à medida em que o usuário percorre pelas gNBs ao longo de seu deslocamento.

5.2.2 Experimentos e Resultados

Os módulos *sp-live-streaming* e *sp-segmentation* foram desenvolvidos utilizando a ferramenta FFmpeg[9] portada para seu funcionamento por meio de contêineres *Docker*. Já o módulo *sp-delivery* foi implantado utilizando a imagem oficial do servidor web NGINX[10], também como contêiner *Docker*.

Durante os experimentos, o vídeo capturado foi simulado por um arquivo de vídeo na resolução de 4K (3840 x 2160p) e retransmitido via UDP *multicast* e *unicast* pelo *sp-live-streaming*. Como o objetivo dos experimentos é observar comportamento da rede mediante o uso do *framework* proposto, tal simulação não gera impacto nos resultados

[9] FFmpeg, A complete, cross-platform solution to record, convert and stream audio and video. - https://ffmpeg.org

[10] NGINX is a free, open source HTTP web server, mail proxy server, and reverse proxy and load balancer for HTTP, TCP, and UDP traffic. NGINX is known for its high performance, stability, rich feature set, simple configuration, and low resource consumption. - https://docs.nginx.com/nginx/

obtidos.

Numa abordagem de entrega de vídeo convencional, a comunicação *unicast* entre cliente e servidor percorre toda a infraestrutura do MNO até chegar ao servidor do SP ou a um nó de CDN mais próximo. À medida que mais usuários estejam conectados e solicitando a entrega do vídeo, um novo tráfego *unicast* é gerado, o que em uma larga escala pode sobrecarregar a largura de banda disponível nos MNOs. As interações do usuário para a ampliação da área de visualização do vídeo seriam executadas localmente no dispositivo móvel, que receberia o tráfego em 4K para permitir qualidade na visualização ampliada.

Para demonstrar o serviço convencional exposto acima, o módulo *sp-live-streaming-unicast* envia fluxo de vídeo em resolução 4K, enquanto o *sp-segmentation-4K* realiza a segmentação e o *sp-delivery* se encarrega de entregar o vídeo ao usuário final. Foram realizadas capturas de pacotes entre a gNB-1, onde o usuário estava conectado e o núcleo da rede, onde os módulos estavam instanciados. A captura de pacotes durou aproximadamente 150 segundos, acarretando em um tráfego médio levemente superior a 25 Mbit/s. Neste mesmo contexto, um novo usuário se conectou à gNB-1 e uma nova captura de pacotes foi realizada, respeitando o mesmo intervalo de tempo. Ao final da captura, nota-se que o tráfego de dados praticamente é dobrado, conforme ilustra a Figura 6. Dado este cenário, assume-se que o volume do tráfego de dados no núcleo da rede é incrementado a cada novo usuário conectado, o que pode resultar em sobrecarga e congestionamento no núcleo da rede em algum momento.

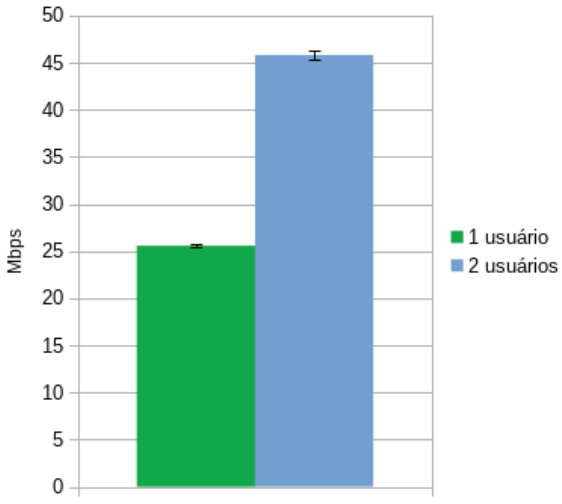


Figura 6 – Tráfico médio gNB x núcleo da rede - resolução 4k

A fim de minimizar o impacto na largura de banda de um MNO devido ao aumento de usuários conectados, *Join-Me* pode ser usado com a abordagem a utilização de retransmissão do *streaming unicast* para *multicast*, conforme descrito na Seção anterior. Com essa abordagem, visa-se reduzir a quantidade de comunicações *unicast* fim a fim,

ocasionando uma redução no consumo de largura de banda.

Ao receber o fluxo do vídeo em *multicast*, os segmentadores realizam a transcodificação alterando a resolução do vídeo de 4K para 1080p (1920 x 1080p), resolução suportada pela grande maioria dos dispositivos móveis e portáteis e, sem dúvida, com menor taxa de dados. Além da alteração da resolução, os segmentos foram definidos com o tamanho de quatro segundos, o que, apesar de gerar um maior número de segmentos, reduz a latência do serviço dado um menor tamanho dos arquivos e tempo para gerá-los. Tal decisão é tipicamente observada em serviços de vídeo ao vivo.

Inicialmente, o vídeo foi segmentado e entregue no núcleo da rede em sua resolução original, conforme apresentado na Figura 7A. Nesta primeira opção de implantação do serviço, o dispositivo do usuário fica responsável pelo processamento da ampliação do vídeo. Apesar do processamento demandado pela segmentação do vídeo original ser consideravelmente baixo, esta abordagem pode se mostrar inviável, uma vez que apenas dispositivos de alto poder computacional e de comunicação conseguiriam reproduzir o vídeo ao vivo nesta resolução. Isso limitaria o consumo do serviço apenas a uma pequena porção dos potenciais usuários. Além disso, a alta demanda de processamento para a reprodução do vídeo impactaria diretamente no consumo energético do dispositivo, degradando a durabilidade de sua bateria.

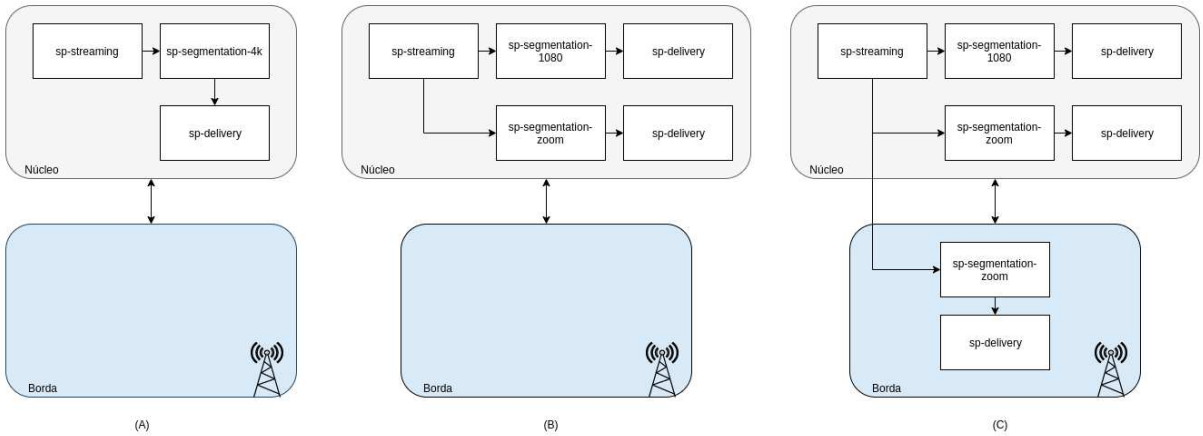


Figura 7 – Posicionamento dos módulos ao longo dos experimentos

Na segunda opção de implantação do serviço, vide Figura 7B, tanto o vídeo completo, quanto da área selecionada pelo usuário foram segmentados e entregues a partir do núcleo da rede. Devido à transcodificação no processo de *downscale* do vídeo, junto à segmentação, é esperado um aumento considerável no consumo de recursos pelo módulo *sp-segmentation-1080*. Já o módulo *sp-segmentation-zoom* também teve um consumo de recursos alto, inclusive superior ao *sp-segmentation-1080*. Isso se deve ao fato de que a área selecionada nos testes possui uma alta frequência de movimentação. Assim, aquilo que no vídeo original seria apenas um pequeno detalhe em uma determinada região, torna-se

uma grande porção de área no vídeo ampliado. O consumo dos recursos utilizados pelos módulos podem ser observados na Figura 8.

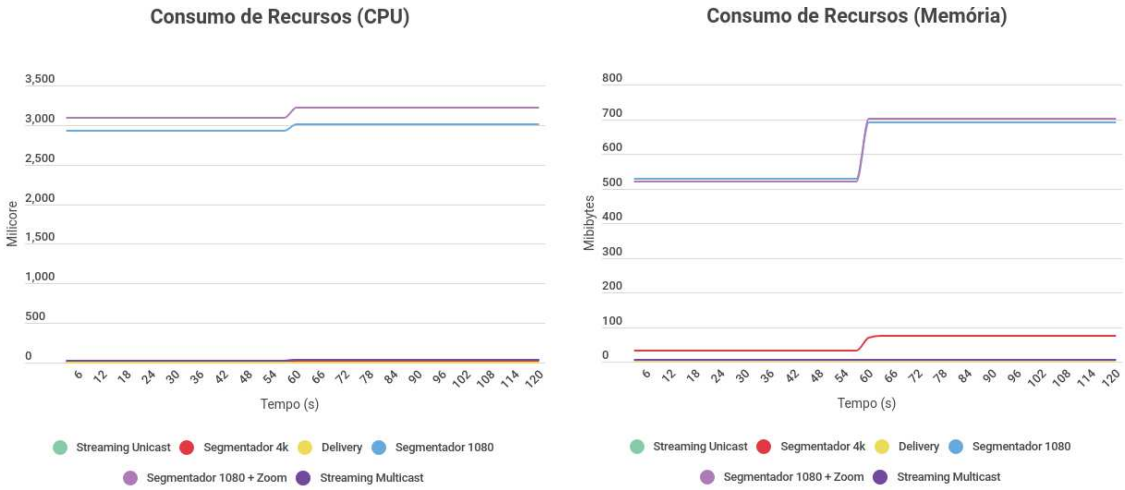
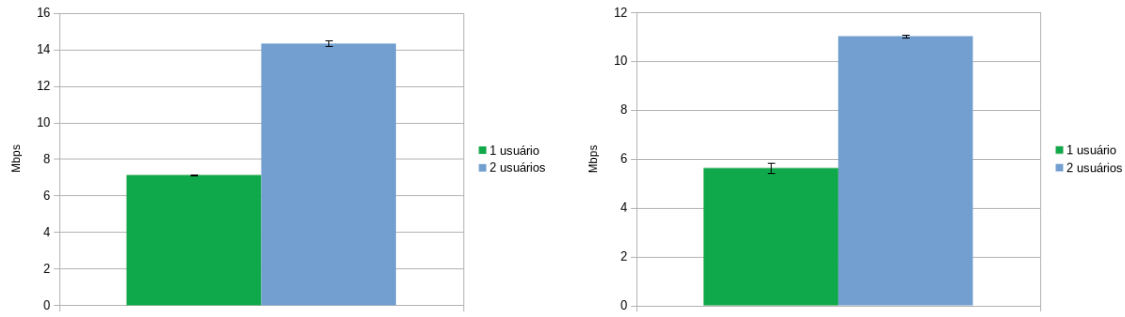


Figura 8 – Consumo de recursos (CPU e Memória)

A Figura 7C representa uma terceira opção de implantação, em que a área selecionada pelo usuário inicialmente é segmentada pelo núcleo da rede. No entanto, dada a solicitação do usuário, os módulos de segmentação e de entrega são instanciados na borda da rede, de modo que haja a migração de parte do serviço a fim de atendê-lo com uma maior proximidade. Como esperado, a Figura 9 mostra que o tráfego de dados proveniente dos módulos *sp-segmentation-1080* e *sp-segmentation-zoom* é inferior se comparado ao *sp-segmentation-4K*. Além disso, com a abordagem de *multicast* dentro da infraestrutura do MNO, a largura de banda consumida pelo *streaming* é semelhante a consumida por um único usuário utilizando o módulo *sp-segmentation-4K*, sob a abordagem de entrega de vídeo convencional. Dentro do MNO, a largura de banda utilizada se dá pelo quantidade de usuários consumindo o *streaming* no núcleo da rede, além do consumo, se necessário, da área selecionada para ampliação pelo usuário na borda da rede.



(a) sp-segmentation-1080 (b) sp-segmentation-zoom

Figura 9 – Tráfego médio

A qualidade de experiência do usuário se dá pela percepção do usuário ao receber

um determinado serviço, podendo ela ser positiva ou negativa. O setor de padronização do *International Telecommunication Union* (ITU) (41), define QoE como (em tradução livre):

> "O grau de satisfação ou aborrecimento do usuário de um aplicativo ou serviço."

Como apresentado nos Capítulos 1 e 2, esforços têm sido empregados por SPs e MNOs para elevar a qualidade de experiência do usuário, seja por distribuição de conteúdo pelas CDNs ou aplicações de políticas de qualidade de serviço.

A fim de mensurar a qualidade de experiência do usuário relacionada ao serviço de entrega de vídeo oferecido pela PoC, foi utilizado o método denominado *Video Multimethod Assessment Fusion* (VMAF) (42), desenvolvido pela *Netflix* para medir a qualidade de seus serviços de *streaming*. O VMAF é composto por um conjunto de métricas de análise de qualidade de vídeo e através de técnicas de aprendizagem de máquina une seus pontos fortes em uma métrica final. Por exemplo, é feita uma composição entre a análise de componentes de luminância e de um índice de similaridade estrutural entre imagens. O modelo de aprendizado de máquina é treinado e testado com base no *NFLX Video Dataset*, experimento interno à Netflix. A ferramenta ffmpeg suporta a biblioteca libvamf, a qual é responsável por aplicar o VMAF por meio do ffmpeg (ffmpeg-vmaf). Para que tal biblioteca seja suportada pelo ffmpeg, é necessário recompilar o código da ferramenta.

O processo de comparação de vídeos realizado pelo VMAF consiste em métricas de qualidade de imagens entre dois vídeos. Métricas como *Visual Information Fidelity* (VIF) e *Detail Loss Metric* (DLM) são utilizadas para análise de características de imagens, objetivando a fidelidade de informações e a menor perda de detalhes quadro a quadro de cada vídeo, resultando na visibilidade do conteúdo. Com relação à continuidade e diferença temporal entre os vídeos, é utilizado o recurso de movimento, analisando características referentes a luminância entre quadros adjacentes de cada vídeo e comparado entre eles.

Para realizar a medição da qualidade de experiência do usuário, o processo de transcodificação do vídeo (*crop* da área selecionada e alteração de resolução) e segmentação foram realizadas de maneira *offline*, ou seja, localmente e fora da arquitetura *Join-Me*. A partir dos segmentos gerados, foi realizada a concatenação dos segmentos resultando em um vídeo final, o qual foi denominado como vídeo-original-zoom e passou a ser utilizado como referência na comparação do vídeo transcodificado no núcleo da rede e o recebido pelo usuário final (transcodificado na borda da rede).

Duas instâncias dos módulos *sp-segmentation-zoom* e seu respectivo *sp-delivery* foram colocadas em execução no núcleo da rede e na borda, respectivamente. Em seguida, o *sp-live-streaming-multicast* iniciou a transmissão do vídeo original em 4k (ou seja, sem transcodificação), o qual simultaneamente passou a ser transcodificado e segmentado no

núcleo e na borda da rede. Na borda da rede, em particular, os segmentos foram entregues ao dispositivo final.

No dispositivo final, o segmentos foram baixados pela ferramenta ffmpeg, tendo como entrada de dados a URL do *sp-delivery* e por meio da opção -re, o *download* dos segmentos era realizado à medida em que os mesmos seriam reproduzidos, possibilitando assim uma maior aproximação com a realidade de um reprodutor de vídeo. Após a conclusão do *download* dos segmentos na borda e a cópia (offline) dos gerados no núcleo, os mesmos também foram concatenados. Os vídeos concatenados foram comparados pela biblioteca ffmpeg-vmaf com o vídeo-original-zoom. Assumiu-se que o núcleo da rede estava com sobrecarga e por isso, os segmentos gerados foram copiados sem a utilização do *framework*. Já na borda da rede, houve a segmentação do vídeo e a entrega ao dispositivo final, onde foi efetuado a concatenação do segmentos.

Decidiu-se pela utilização do ffmpeg no dispositivo final, para evitar a utilização de *softwares* terceiros para a coleta de resultados. Estes *softwares* utilizam técnicas de interceptação de tráfego, de modo que tal prática acrescenta o tempo de leitura das informações, o que impacta na perda de segmentos, ocasiona interrupções e falta de continuidade do vídeo, levando a uma redução da qualidade de experiência do usuário.

Foram realizadas dez execuções do procedimento para cada ambiente (núcleo e borda). Para cada video resultante, executou-se a ffmpeg-vmaf, a qual realizava a comparação entre o vídeo obtido pelo ambiente com o vídeo-original-zoom. Ao final da comparação, a ferramenta apresenta um valor numérico referente à qualidade do vídeo gerado. Quanto maior for o valor ao final da comparação, maior a qualidade do vídeo recebido. Após o termino das execuções, obteve-se um resultado médio de 11,77% para a segmentação no núcleo e 31,88% para os segmentos recebidos pelo usuário final, provenientes da borda da rede. Este resultado expressa um ganho superior a 270% na comparação da entrega do vídeo pelo núcleo e borda da rede. A Figura 10 apresenta os resultados relacionados à QoE do usuário.

É possível notar que a alternância dos resultados obtidos em cada experimento realizado na borda da rede é mais expressiva que no núcleo. Isso se deve ao *overhead* do próprio dispositivo cliente (aplicações em *background* consumindo recursos de rede e que utilizam o mesmo meio de comunicação que o tráfego dos segmentos). Uma vez que não foram definidas alternativas de resolução do vídeo entregue (situação comum em ambientes de vídeo sob demanda, o chamado *streaming* adaptativo), a interferência deste *overhead* no recebimento dos segmentos de alta resolução impacta diretamente na qualidade do vídeo recebido. No entanto, mesmo com a interferência apresentada, a melhoria na qualidade no vídeo entregue pela borda da rede é inquestionável.

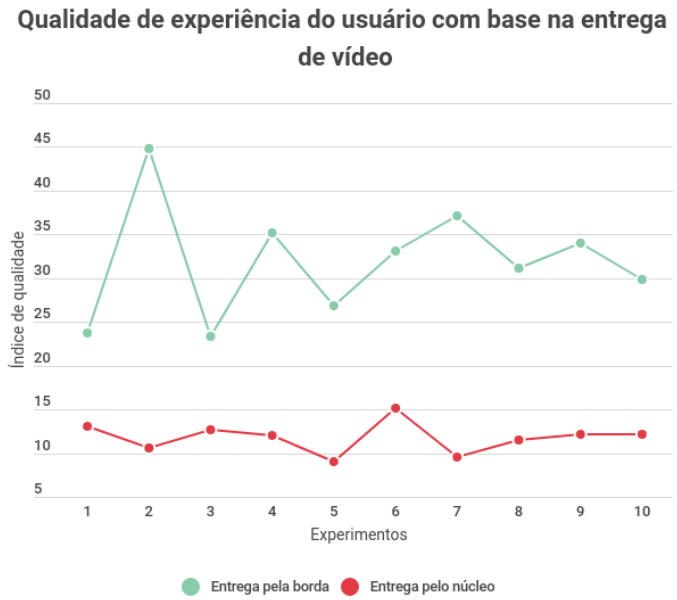


Figura 10 – Resultado dos experimentos relacionados à QoE do usuário

6 CONCLUSÃO

Este trabalho apresentou o *Join-Me*, um *Framework* distribuído e independente de tecnologias específicas, que permite que Provedores de Serviços (SPs) internalizem seus serviços na infraestrutura dos Operadores de Redes Móveis (MNOs), por meio de módulos. Para cada módulo, o SP informa, por meio de opções parametrizadas, a porção necessária de recursos a serem consumidos e a localização desejada para a instanciação daquela tarefa de um serviço. Tal trabalho possibilitou uma melhor integração entre SPs e MNOs, além de trabalhar de maneira harmoniosa o atendimento dos requisitos dos serviços e aplicações, a alocação dinâmica de recursos e a privacidade do usuário.

A privacidade do usuário pôde ser preservada, através do gerenciamento autônomo dos módulos internalizados, sob a responsabilidade do MNO. Tal gerenciamento possibilitou que aplicações dos SPs não interagissem com as aplicações internas da infraestrutura. Além disso, a utilização de *slices* de rede, possibilitou o isolamento de informações sensíveis, das demais aplicações presentes nos MNOs e das internalizadas pelos SPs. Para evitar a má utilização dos recursos computacionais, estes foram devidamente alocados à medida em que os módulos eram instanciados, e liberados sempre que os serviços não eram mais demandados na borda da rede. A possibilidade de os SPs parametrizarem os requisitos para o ideal funcionamento de suas aplicações é um grande diferencial deste trabalho, uma vez que tal atenção não é dada amplamente aos SPs. Normalmente, eles ficam fadados a utilizar apenas o que é disponibilizado pelos provedores *Plataform as a Service* (PaaS) ou *Infrastructure as a Service* (IaaS).

A fim de avaliar sua aplicabilidade, um exemplo de serviço de telemedicina interativa com vídeo em ultra-alta resolução foi utilizado como prova de conceito, o qual apresentou uma redução na largura de banda fim-a-fim por meio da abordagem *multicast*, processamento e entrega do serviço o mais próximo possível do usuário. A migração do serviço também é explorada, possibilitando que o mesmo "siga"o usuário à medida em que ele se desloca entre as estações rádio base. Para avaliar a qualidade de experiência do usuário com o conteúdo recebido, foi utilizado o método de *Video Multimethod Assessment Fusion*, aplicado ao conteúdo segmentado na borda da rede e recebido pelo usuário, além do segmentado no núcleo da rede e ambos comparados com conteúdo original.

A utilização de um único fluxo *unicast* em altíssima resolução para dentro da infraestrutura do MNO, mostrou que é possível reduzir a sobrecarga da infraestrutura de rede presente na Internet atual. Além disso, internamente nos MNOs, também é possível perceber a redução da sobrecarga por meio da abordagem de *multicast* entre servidores *cloud* e *edge*. Por fim, notou-se a melhoria na QoE do usuário, uma vez que o serviço era entregue o mais próximo possível do mesmo, apresentando assim um ganho superior a 270% da qualidade se comparado com a entrega no núcleo da rede.

6.1 TRABALHOS FUTUROS

O cenário pandêmico vivido no ano de 2020 limitou o desenvolvimento e experimentação do trabalho apresentado, de modo que experimentos adicionais como a análise do comportamento da entrega de serviços com utilização de QoS em uma rede congestionada, medições apuradas de retardo, entre outros, não puderam ser efetuados a contento. Isso levaria à necessidade de um número maior de pessoal envolvido e tempo de uso do laboratório presencialmente. De igual modo, a entrega de mais serviços em paralelo, bem como a escalabilidade do *framework* podem ser destacado como trabalhos futuros.

Para lidar com cenários maiores, se faz necessária a utilização de novas abordagens de roteamento e inteligência da rede, como a *Software Defined Networks* (SDN). Com a utilização deste tipo de abordagem, vê-se a oportunidade de unificar os componentes *Traffic Control* e *Traffic Control Agent* em um único controlador de tráfego. Ainda visando cenário maiores, é importante realizar experimentos do *framework* e analisar seu desempenho com tecnologias e equipamentos reais de redes móveis, uma vez que os experimentos apresentados no trabalho foram realizados com tecnologia WiFi.

A fim de garantir a entrega do serviço fim a fim, é prevista a utilização do IPv6 para novas versões da implementação do *framework*. Essa mudança tem como principal objetivo possibilitar identificar o dispositivo do usuário independente de sua localização, removendo técnicas intermediárias como NAT, o qual é utilizado atualmente. Sendo assim, novas abordagens para identificação das ERBs que provêem as requisições devem ser adotadas, sejam elas inseridas ao *framework* ou através de informações obtidas por arquiteturas terceiras, como a própria MEC.

Do ponto de vista dos Provedores de Serviços, é interessante prover um *dashboard* para o acompanhamento real de seus serviços e respectivos módulos internalizados. Visa-se também ampliar a gama de opções a serem configuradas, visando cobrir um número maior de serviços e aplicações de diversas naturezas. Para disponibilizar essas novas opções, se faz necessário também realizar experimentos com novos tipos de serviços, os quais tenham demandas distintas, como acesso a bases de dados, migração de contexto, entre outras características.

REFERÊNCIAS

1 PENG, Gang. **CDN: Content distribution network**. arXiv preprint cs/0411069, 2004.

2 SHI, Weisong; DUSTDAR, Schahram. **The promise of edge computing**. Computer, v. 49, n. 5, p. 78-81, 2016.

3 GUPTA, Akhil; JHA, Rakesh Kumar. **A survey of 5G network: Architecture and emerging technologies**. IEEE access, v. 3, p. 1206-1232, 2015.

4 HU, Yun Chao et al. **Mobile edge computing—A key technology towards 5G**. ETSI white paper, v. 11, n. 11, p. 1-16, 2015.

5 PATEL, Milan et al. **Mobile-edge computing introductory technical white paper**. White paper, mobile-edge computing (MEC) industry initiative, p. 1089-7801, 2014.

6 MADEMANN, Frank. **The 5G system architecture**. Journal of ICT Standardization, v. 6, n. 1, p. 77-86, 2018.

7 ALAWE, Imad et al. **On the scalability of 5G Core network: the AMF case**. In: 2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2018. p. 1-6.

8 ROST, Peter et al. **Network slicing to enable scalability and flexibility in 5G mobile networks**. IEEE Communications magazine, v. 55, n. 5, p. 72-79, 2017.

9 3GPP TS 23.503: **Policy and charging control framework for the 5G System (5GS**); version 15.2.0 Release 15

10 MAYER, Georg. **RESTful APIs for the 5G service based architecture.** Journal of ICT Standardization, v. 6, n. 1, p. 101-116, 2018.

11 **5G; System Architecture for the 5G System** (3GPP TS 23.501 version 15.2.0 Release 15)

12 TANGUDU, Narendranath Durga et al. **Common Framework for 5G Northbound APIs**. In: 2020 IEEE 3rd 5G World Forum (5GWF). IEEE, 2020. p. 275-280.

13 Wang, L., Von Laszewski, G., Younge, A., He, X., Kunze, M., Tao, J., & Fu, C. (2010). **Cloud computing: a perspective study**. New generation computing, 28(2), 137-146.

14 PHAM, Quoc-Viet et al. **A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art**. IEEE Access, v. 8, p. 116974-117017, 2020.

15 **Mobile Edge Computing (MEC); Location API** ETSI Multi-access Edge Computing (MEC). Group Specification: ETSI GS MEC 013, 2017. V1.1.1. Disponível em: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/013/01.01.01_60/gs_MEC013v010101p.pdf. Acesso em: 8 mar 2019.

16 **Mobile Edge Computing (MEC); Radio Network Information API** ETSI Multi-access Edge Computing (MEC). Group Specification: ETSI GS MEC 012, 2017. V1.1.1. Disponível em: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/012/01.01.01_60/gs_MEC012v010101p.pdf. Acesso em: 8 mar 2019.

17 **Mobile Edge Computing (MEC) Bandwidth Management API** ETSI Multi-access Edge Computing (MEC). Group Specification: ETSI GS MEC 015, 2017 V1.1.1. Disponível em: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/015/01.01.01_60/gs_MEC015v010101p.pdf

18 **Mobile Edge Computing (MEC); Mobile Edge Management; Part 2: Application lifecycle, rules and requirements management**. ETSI Multi-access Edge Computing (MEC). Group Specification: MEC GS 010-2, 2017. V1.1.1. Disponível em: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/01002/01.01.01_60/gs_MEC01002v010101p.pdf. Acesso em: 8 mar. 2019.

19 **Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV**. NFV GS 003, 2014, V1.2.1. Disposnível em: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/003/01.02.01_60/gs_NFV003v010201p.pdf. Acesso em 15 mar. 2019.

20 MIJUMBI, Rashid et al. **Network function virtualization: State-of-the-art and research challenges**. IEEE Communications surveys & tutorials, v. 18, n. 1, p. 236-262, 2015.

21 CZIVA, Richard; PEZAROS, Dimitrios P. **Container network functions: bringing NFV to the network edge**. IEEE Communications Magazine, v. 55, n. 6, p. 24-31, 2017.

22 CZIVA, Richard; JOUET, Simon; PEZAROS, Dimitrios P. **Roaming edge vNFs using glasgow network functions**. In: Proceedings of the 2016 ACM SIGCOMM Conference. 2016. p. 601-602.

23 TALEB, Tarik; KSENTINI, Adlen; JANTTI, Riku. **"Anything as a Service"for 5G Mobile Systems**. IEEE Network, v. 30, n. 6, p. 84-91, 2016.

24 TALEB, Tarik; KSENTINI, Adlen. **Follow me cloud: interworking federated clouds and distributed mobile networks**. IEEE Network, v. 27, n. 5, p. 12-19, 2013.

25 FULLER, V.; FARINACCI, D. *Locator/ID separation protocol (LISP) map-server interface*. RFC 6833, January, 2013.

26 TALEB, Tarik; HASSELMEYER, Peer; MIR, Faisal Ghias. **Follow-me cloud: An OpenFlow-based implementation**. In: 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing. IEEE, 2013. p. 240-245.

27 Taleb, T.; Ksentini, A. (2013, December). **An analytical model for follow me cloud.** In 2013 IEEE Global Communications Conference (GLOBECOM) (pp. 1291-1296). IEEE.

28 TALEB, Tarik; KSENTINI, Adlen; FRANGOUDIS, Pantelis. **Follow-me cloud: When cloud services follow mobile users**. IEEE Transactions on Cloud Computing, 2016.

29 SARRIGIANNIS, Ioannis et al. **Application and network VNF migration in a MEC-enabled 5G architecture**. In: 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD). IEEE, 2018. p. 1-6.

30 FIELDING, Roy et al. RFC 2616: **Hypertext transfer protocol–HTTP/1.1**, June 1999. Status: Standards Track, v. 1, n. 11, p. 1829-1841, 1999.

31 BRAY, Tim. **The I-JSON Message Format**. RFC 7493, DOI 10.17487/RFC7493, March 2015,< http://www. rfc-editor. org/info/rfc7493, 2015.

32 FIELDING, Roy T.; TAYLOR, Richard N. **Architectural styles and the design of network-based software architectures**. Irvine: University of California, Irvine, 2000.

33 Xiao, Y. (2005). **IEEE 802.11 n: enhancements for higher throughput in wireless LANs**. IEEE Wireless Communications, 12(6), 82-91.

34 Neves, J. C. (2008). **Programação Shell Linux 9a Edição**. Brasport.

35 YLONEN T. **The Secure Shell (SSH) Connection Protocol**. Disponível em: https://tools.ietf.org/rfc/rfc4254.txt. Acesso em: 8 mar 2019.

36 Benzekki, K., El Fergougui, A., Elbelrhiti Elalaoui, A. (2016). **Software-defined networking (SDN): a survey**. Security and communication networks, 9(18), 5803-5833.

37 SRISURESH, Pyda; EGEVANG, Kjeld. **Traditional IP network address translator (Traditional NAT)**. 2001.

38 BEN-KIKI, Oren; EVANS, Clark; INGERSON, Brian. **Yaml ain't markup language (yaml™)** version 1.1. Working Draft 2008-05, v. 11, 2009.

39 DE DEUS, Alexandre Martins Gama; JULIO, Eduardo Pagani; MORENO, Marcelo Ferreira. **Join-Me: Uma arquitetura para integração entre operadores de redes móveis e provedores de serviços**. In: Anais do Workshop de Pesquisa Experimental da Internet do Futuro. SBC, 2020. p. 26-31.

40 POSTEL, Jon. RFC0768: **User Datagram Protocol**. 1980.

41 ITU-T Telecommunication Standardzation Sector of ITU - **Vocabulary for performance, quality of service and quality of experience**, P.10/G.100, 2017.

42 The Netflix Tech Blog **Toward A Practical Perceptual Video Quality Metric** https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652. Acesso em: 20 dez.
2020

43 The Netflix Tech Blog **VMAF: The Journey Continues** https://netflixtechblog.com/vmaf-the-journey-continues-44b51ee9ed12. Acesso em: 20 dez. 2020