Deployment e Testes de Controladores Inteligentes da RAN no *Testbed* do Projeto OpenRAN@Brasil

Paulo R. B. da Silva[1], João P. S. H. Lima[1], Vitalii Afanasiev[1], Michelle S. P. Facina[1], Erika C. Alves[1], Gustavo C. Lima[1]

[1]Centro de Pesquisa e Desenvolvimento em Telecomunicações (CPQD) – Campinas, SP – Brasil

{prsilva, jsales, vitaliiaf, mfacina, erikaa, gcorrea}@cpqd.com.br

Abstract. *The virtualization of network functions and the disaggregation of components require the management of the Radio Access Network (RAN) via a central controller known as the RAN Intelligent Controller (RIC). The RICs from the Open Networking Foundation (ONF), O-RAN Software Community (OSC), and OpenAirInterface (OAI) emerge as nearly complete solutions to the RAN control problem. In this paper, setups are created for each RIC for the testing of essential operations related to the monitoring of Key Performance Indicators (KPIs) and/or for the slicing of network resources. Using these setups, we monitor RAN metrics through graphical interfaces and create slices.*

Resumo.
A virtualização de funções de rede e a desagregação de componentes demandam o gerenciamento dos recursos da Radio Access Network (RAN) através de um controlador central conhecido como RAN Intelligent Controller (RIC). Os RICs da Open Networking Foundation (ONF), O-RAN Software Community (OSC) e OpenAirInterface (OAI) fornecem soluções quase completas para o gerenciamento da RAN. Neste artigo, setups são criados para cada RIC que testam operações essenciais relacionadas ao monitoramento de Key Performance Indicators (KPIs), ao slicing de recursos da rede ou a ambos. Usando esses setups, monitoramos as métricas da RAN por meio de interfaces gráficas e demonstramos a criação de slices de rede.

1. Introdução

Soluções de mobilidade estão passando por um processo de virtualização e desagregação. Soluções monolíticas estão sendo convertidas em blocos virtualizados, coordenados de forma automática e escalável. O novo paradigma normalmente usa gerenciadores como o *Kubernetes*, o que se traduz em elementos da solução embarcados em *pods* que se comunicam uns com os outros através de APIs ou protocolos de comunicação específicos [Arnaz et al. 2022].

Ferramentas descentralizadas e virtualizadas podem ser testadas sem a necessidade de altos investimentos em equipamentos e de tempo na montagem de *setups* experimentais. No entanto, os cenários de implementação e teste devem ser modificados para a validação das novas ferramentas, especialmente nas etapas de orquestração e controle da rede. Um exemplo de *testbed* seguindo os novos paradigmas inclui o projeto *Cloud Enhanced Open Software Defined Mobile Wireless testbed for City-Scale Deployment*

(COSMOS) [COSMOS 2023]. Ele tem como objetivo criar cenários de teste de larga escala espalhados por vários países. A arquitetura do COSMOS foca em largura de banda ultra larga e comunicação sem fio de baixa latência usando computação distribuída para o aumento do poder de processamento. Seu *testbed* consiste de 40 a 50 nós de rádios definidos por software conectados a redes *front-haul* e *back-haul* com infraestrutura de computação de borda [COSMOS 2023].

Outro *testbed* frequentemente estudado é o da plataforma Colosseum [Polese et al. 2022], o qual combina 128 nós de rádio padrão com um emulador de canal digital sustentado por uma rede de roteamento composta por *Field-Programmable Gate Arrays* (FPGAs) [Colosseum 2020]. Cada um dos nós de rádio fornece uma plataforma para comunicação via radiofrequência e para aplicações de aprendizado de máquina através de um servidor Dell R730 equipado com uma GPU NVIDIA K40M e um *Universal Software Radio Peripheral* (USRP) Ettus X310 equipado com uma FPGA XILINX Kintex 7 [Colosseum 2020].

Seguindo as iniciativas do consórcio Open-RAN [O-RAN Alliance 2023], podemos fragmentar a rede de acesso de rádio (Radio Access Network - RAN) em *Control Unit* (CU), *Distributed Unit* (DU) e *Radio Unit* (RU). A CU e a DU dividem entre si os protocolos tratados pela RAN e podem ser gerenciadas por meio de controladores lógicos conhecidos como *RAN Intelligent Controllers* (RICs), os quais controlam funções de rede e a lógica de operação da RAN. Entre as implementações práticas de código aberto do RIC destacam-se o μ-ONOS da *Open Networking Foundation* (ONF), o RIC da *O-RAN Software Community* (OSC) e o FlexRIC da *OpenAirInterface* (OAI) [ORAN Software Community 2023, Open Networking Foundation 2023, OpenAirInterface 2022].

O objetivo deste trabalho é demonstrar a atuação dessas três implementações e apresentar os ambientes desenvolvidos para o teste de suas funcionalidades. Nós demonstramos o uso do RIC para o monitoramento de KPIs da RAN via *Command-Line Interfaces* (CLIs) ou interfaces gráficas do Grafana e Prometheus. Também avaliamos como o RIC pode fatiar os recursos físicos da rede por meio de uma aplicação que produz *slices* com mais ou menos recursos em termos de *throughput*, o que é apresentado especificamente para o RIC da ONF. O conjunto de testes apresentado se insere dentro de um *testbed* maior pertencente ao projeto OpenRAN@Brasil, que envolve tecnologias óticas de transmissão e multiplexação e tecnologias de computação de borda.

Segue a organização do restante do artigo. A Seção 2 apresenta o ambiente de teste geral do projeto OpenRAN@Brasil com o propósito de contextualizar a atuação do RIC. A Seção 3 resume as funções do RIC, assim como três de suas implementações, com aplicações, *setups* e *testbeds*. A Seção 4 se dedica aos experimentos realizados para o monitoramento de métricas da RAN e o *slicing* de recursos de rede. A Seção 5 apresenta conclusões e trabalhos futuros.

2. *Testbed* do projeto

O projeto OpenRAN@Brasil realiza a pesquisa e o desenvolvimento de *software* para uma plataforma de código aberto que controla e gerencia infraestruturas de rede programáveis compostas por equipamentos desagregados. O projeto prevê um *testbed* com integração de componentes fornecidos por diferentes fabricantes de *hardware* e *software* para dife-

rentes domínios tecnológicos [RNP 2022, OpenRAN Brasil 2022].

Domínios tecnológicos envolvendo a RAN, a *Cloud* e tecnologias óticas se integram de forma desagregada à rede. Esses incluem elementos da rede sem fio, como o RIC e o *Core*, dados de programabilidade do tipo *Programming Protocol-independent Packet Processors* (P4), *Dense Wavelengths Division Multiplexing* (DWDM), *Fiber to the 'x'* (FTTX) e infraestrutura de *Cloud/Edge Computing* [OpenRAN Brasil 2022].

As redes lógicas implantadas no *testbed* final serão (i) a rede de gerência, (ii) a rede de controle e orquestração, (iii) a rede de sincronização e (iv) a rede de dados [OpenRAN Brasil 2022]. A Figura 1 apresenta a conexão lógica entre os componentes da infraestrutura do *testbed* no qual o RIC descrito neste artigo se insere.

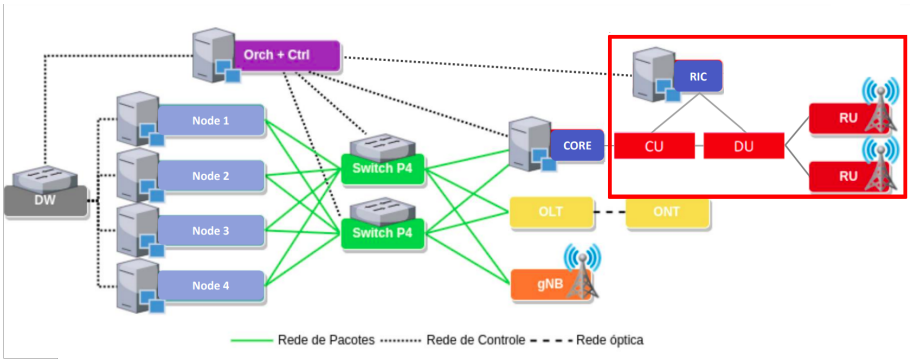

Figura 1. Esquema do *Testbed* do CPQD com a estrutura do RIC em destaque.

3. RIC

O principal objetivo do RIC é trazer inteligência para o processo de otimização de recursos de RAN. Como pode ser observado na Figura 2, há dois tipos de controladores lógicos: o RIC *near Real-Time* (near-RT RIC) e o RIC *non Real-Time* (non-RT RIC) [O-RAN Alliance 2023]. A maior distinção entre eles é o tempo de resposta e o nível de proximidade da rede. O near-RT RIC está mais próximo da RAN e age em operações que exigem processamento em tempo quase real. O non-RT RIC realiza cálculos que demandam maior capacidade de computação e tempo para o processamento. Este artigo descreve apenas o near-RT RIC, já que o non-RT RIC no momento da escrita deste artigo ainda não é uma solução madura para testes.

3.1. Near-RT RIC

Dadas as rígidas restrições de latência, o near-RT RIC se localiza na borda da nuvem de telecomunicações e opera as malhas de controle do *Radio Resource Management* (RRM), as quais requerem respostas rápidas com períodos entre 10 ms e 1 s. Ele interage com a DU e a CU na RAN, assim como com estações rádio-base (*base stations* - BS) que suportam a interface E2 [Bonati et al. 2021, ORAN Software Community 2022c].

O near-RT RIC tem seu funcionamento baseado em xApps e em esquemas de operação conhecidos como modelos de serviço (*Service Models* - SM). Uma xApp é uma aplicação de software que pode ser desenvolvida por terceiros e que implementa uma funcionalidade especifica dentro do near-RT RIC (como o controle de recursos rádio). O

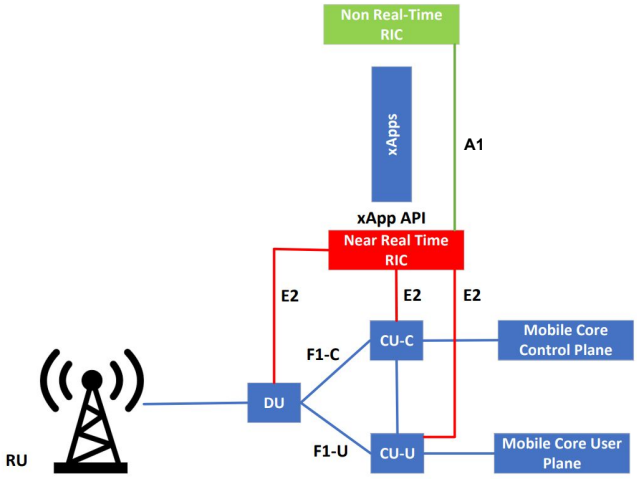

Figura 2. Arquitetura Open RAN com componentes e interfaces ligados ao RIC.

near-RT RIC usa interfaces padronizadas de comunicação, tal como a interface E2, que o conecta à DU e CU, e a interface A1, que permite ao non-RT RIC fornecer controles de ação ao near-RT RIC (ver Figura 2) [ORAN Software Community 2022b]. Além disso, SMs são usados para receber dados da RAN, processá-los e enviar respostas com ações de controle [ORAN Software Community 2022b].

Para realizar o controle, o near-RT RIC inclui (i) um banco de dados contendo informação da RAN, (ii) uma infraestrutura de envio/recebimento de mensagens para realizar a interface com diferentes componentes da rede, (iii) um sistema de comunicação por meio de APIs e (iv) um mecanismo para controlar funções da RAN com resolução de conflitos [ORAN Software Community 2022b].

3.2. RICs Analisados

3.2.1. OSC

Alguns dos principais componentes fornecidos pela plataforma da OSC são o Gerenciador de Roteamento, responsável pela comunicação dentro do RIC, o Gerenciador de Inscrição, que organiza as inscrições originárias de xApps construídos a partir de SMs existentes, e o Gerenciador de *App*, que gerencia a operação dos xApps [ORAN Software Community 2023]. O armazenamento de dados para os *User Equipments* (UEs) e as informações para a RAN são fornecidos através do banco de dados *Remote Dictionary Server* (Redis) [Redias 2009]. Simultaneamente, os xApps podem exportar métricas de desempenho para soluções externas de visualização e armazenamento, como o Prometheus e o InfluxDB, respectivamente [ORAN Software Community 2023].

Para hospedar os xApps, os microsserviços do near-RT RIC da OSC operam de forma independente em um *cluster Kubernetes*. O processo de *onboarding* das aplicações é feito usando a ferramenta *dms cli*, que foi desenvolvida para simplificar a instalação, criar *helm charts* para o ambiente *Kubernetes* a partir de arquivos de configuração e controlar as aplicações através do *App Manager* [ORAN Software Community 2023].

Após o *onboarding* e a inscrição do xApp ao RIC, o xApp pode fazer uso da infraestrutura do near-RT RIC para armazenar métricas de interesse, receber políticas para sua

operação e comunicar-se com nós E2 para a comunicação de dados de monitoramento da RAN ou para a execução de instruções de controle [ORAN Software Community 2023].

3.2.1.1 xApps da OSC

O projeto da OSC que desenvolve e divulga as aplicações para o RIC se chama RICAPP. Essa plataforma contém xApps de demonstração como o *Bouncer*, criado para avaliar gargalos de desempenho da plataforma RIC, e o *Hello-World*, o qual ilustra a estrutura de desenvolvimento básica, a ação de inscrição e a comunicação entre componentes [ORAN Software Community 2023].

O caso de uso de detecção de anomalias, o desenvolvimento mais sofisticado da OSC, engloba quatro xApps: *Anomaly Detection*, *Quality Prediction*, *Traffic Steering* e *RAN Control*. Estas funcionam em conjunto, demonstrando o uso de técnicas de aprendizado da máquina para identificar pioras na experiência de usuários da rede e realizar ações que garantam a qualidade de serviço.

Mais importante do ponto de vista de testbed, contudo, é o xApp *KPI Monitor*, que coleta métricas da CU e da DU e as armazena dentro do RIC para o uso em outras aplicações [ORAN Software Community 2023].

3.2.2. ONF

A ONF possui uma implementação do near-RT RIC conhecida como μONOS-RIC. Como ele evoluiu do controlador *Software-Defined Network* (SDN) *Open Network Operating System* (ONOS), os caminhos de mensagem exigem interação direta com os *pods* pertencentes aos processos internos do ONOS. O μONOS-RIC organiza a comunicação de dados entre seus componentes internos e serviços usando o formato binário *protocol buffer* (protobuf), o qual é transportado através de *endpoints* da API do *Google Remote Procedure Call* (gRPC) [Open Networking Foundation 2023]. Além disso, o μONOS-RIC inclui a integração com ferramentas de aquisição e visualização, tais como o Prometheus e o Grafana [Open Networking Foundation 2022a].

De forma geral, o near-RT RIC da ONF pode ser observado como um chassi que conecta uma série de subsistemas representados por microsserviços e hospeda xApps.

3.2.2.1 xApps da ONF

O desenvolvimento do projeto SD-RAN é baseado em microsserviços [Open Networking Foundation 2023], e cada xApp pode ser utilizado como um pod no *cluster Kubernetes*. Os xApps podem associar informações temporais relacionadas a um UE a um estado dinâmico de informações ou compartilhá-las com outras aplicações. Dada a natureza dinâmica dos dados, a API e o sistema são projetados para permitir modificação contínua dos dados e a mínima latência possível [Open Networking Foundation 2023].

Destacamos dois xApps a seguir. O *RAN Slice Management* (RSM) é um gerenci-

ador da rede que adiciona, atualiza e remove *slices*. Ele também customiza os *slices* com diferentes características, tais como o tipo de *scheduler*, a largura de banda associada e a direção (*downlink* ou *uplink*) [Open Networking Foundation 2022c]. Por fim, atribui UEs aos *slices* criados. O *KPI Monitor*, em contrapartida, é um xApp para o armazenamento de métricas e informações do estado da rede: número de conexões RRC, tentativas de reestabelecimento de conexão, células, etc. [Open Networking Foundation 2022b].

Todos os comandos de configuração utilizados nestes xApps podem ser ativados através de um microsserviço com uma interface CLI [Open Networking Foundation 2022b]. Ela permite interagir com os xApps e executar seus comandos. É interessante notar que os xApps foram todos escritos em Golang, mas existe um *Software Development Kit* (SDK) que disponibiliza xApps semelhantes em Python, possibilitando, dessa forma, um maior número de desenvolvedores a contribuírem com a evolução da plataforma.

3.2.3. OAI

A solução desenvolvida pela OAI se chama FlexRIC. Ele consiste das bibliotecas *server* e *agent* [Schmidt et al. 2021, Schmidt 2021]. A biblioteca *server* traz aplicações internas ao controlador (*controller-**i**nternal applications* - iApps) e interfaces de comunicação [Schmidt 2021]. A composição de iApps ajuda na construção de SMs, como o *network slicing*. Já a biblioteca *agent* possibilita a extensão das funcionalidades dos elementos da RAN (conhecidos como agentes na arquitetura do FlexRIC) via chamadas de API que permitem a comunicação da RAN com serviços externos [Schmidt 2021].

O FlexRIC pode ser interpretado como a implementação de um controlador do tipo *Software-Defined RAN* (SDR), que utiliza um protocolo de comunicação compatível com interfaces de comunicação padronizadas pela O-RAN Alliance, como a interface E2. Essa interface é utilizada pelo controlador em conjunto com a biblioteca *server* para gerenciar conexões com agentes [Schmidt et al. 2021, Schmidt 2021].

A biblioteca *agent* consiste de uma interface de comunicação, uma camada de abstração para o E2 *Application Protocol* (E2AP), uma central de mensagens e uma API para funções genéricas da RAN [Schmidt et al. 2021, Schmidt 2021]. Esta API é projetada especificamente para situações com múltiplos controladores ou SMs, pois indica quais UEs estão expostos a cada controlador.

3.2.3.1 xApps da OAI

As soluções desenvolvidas pela OAI incluem iApps [OpenAirInterface 2023a] e SMs [OpenAirInterface 2023b], muitos dos quais projetados para casos de uso específicos, tal como o *slicing*. Outros SMs incluem `mac_sm`, `pdcp_sm` e `rlc_sm`, os quais coletam estatísticas das camadas RAN *Media Access Control* (MAC), *Packet Data Convergence Protocol* (PDCP) e *Radio Link Control* (RLC), respectivamente [OpenAirInterface 2023b]. Há ainda outros SMs que envolvem protocolos de sinalização, tais como o `gtp_sm`, para o *GPRS Tunnelling Protocol* (GTP), e o `KPM Monitor`, para o *Radio Resource Control* (RRC) [OpenAirInterface 2023b].

O FlexRIC pode também implementar soluções criadas através de xApps [Schmidt 2021]. Os xApps são desenvolvidos em linguagem C mas podem ser convertidos para Python usando a ferramenta de desenvolvimento *Simplified Wrapper and Interface Generator* (SWIG) [Schmidt 2021]. O *HelloWorld* `hw.c`, a aplicação de *slicing* `xapp_slice_moni_ctrl.c`, o monitor do protocolo GTP `xapp_gtp_moni.c`, o monitor de métricas da RAN `xapp_kpm_moni.c` e o monitor de estatísticas MAC, RLC e PDCP `xapp_mac_rlc_pdcp_moni.c` são exemplos de xApps, construídos com base em iApps e os SMs associados [OpenAirInterface 2023a].

4. Experimentos e Testbeds Utilizados com os xApps

4.1. OSC

Nós usamos um computador com as seguintes especificações para a instalação do RIC OSC: i) processador i7 de 2.8 GHz com 8 núcleos, ii) 32 GB de memória RAM e iii) 50 GB de HD.

O xApp *kpimon-go* está disponível no ecossistema do RIC OSC e pode ser usado com outros xApps para fornecer dados de desempenho [ORAN Software Community 2022d]. Ele usa o SM E2 KPM, o qual foi criado para coletar métricas da RAN e de dispositivos associados a ela.

O *setup* atual utiliza uma gNB física e UEs simulados. O *kpimon-go* se conecta com o simulador de E2 (e, por extensão, de UEs) *e2-sim* para criar usuários capazes de se inscrever aos indicadores de desempenho destinados ao RIC. O *e2-sim* envia medidas com informações a nível de UE e a nível de célula para o *kpimon-go*. Essas são armazenadas no banco de dados InfluxDB [ORAN Software Community 2022a]. Uma conexão com o Grafana permite a visualização dos resultados.

A Figura 3a mostra a relação do volume de tráfego *downlink* com os *Physical Resource Blocks* (PRB) disponíveis, enquanto a Figura 3b mostra as medições de potência para diferentes UEs. O gráfico da Figura 3a é importante para a análise de capacidade do canal, pois demonstra a redução de recursos da célula (em termos de PRBs) em função do crescimento do tráfego transmitido. O gráfico da Figura 3b sugere ser fácil detectar usuários com sinal fraco. Isso pode ser usado para ativar o *handover* a uma célula vizinha que possua maior taxa de transmissão.

4.2. ONF

Para a implementação do RIC ONF, verifica-se adequado o uso de um computador ou máquina virtual com as seguintes características: processador Intel Broadwell (ou microarquitetura mais recente) de pelo menos 6 núcleos, memória RAM de 32 GB ou mais e disco rígido de 100 GB ou mais.

Uma característica do SD-RAN é sua integração com a pilha OAI *Long-Term Evolution* (LTE), que permite a implementação de um *setup* físico com USRP e UEs do tipo *Commercial off-the-shelf* (COTS). O uso de um setup físico possibilita a exploração de cenários com o plano de usuário.

O *setup* com simulador não inclui o plano de usuário. No entanto, é capaz de acessar o plano de controle e proporcionar experimentos em ambientes complexos, tais

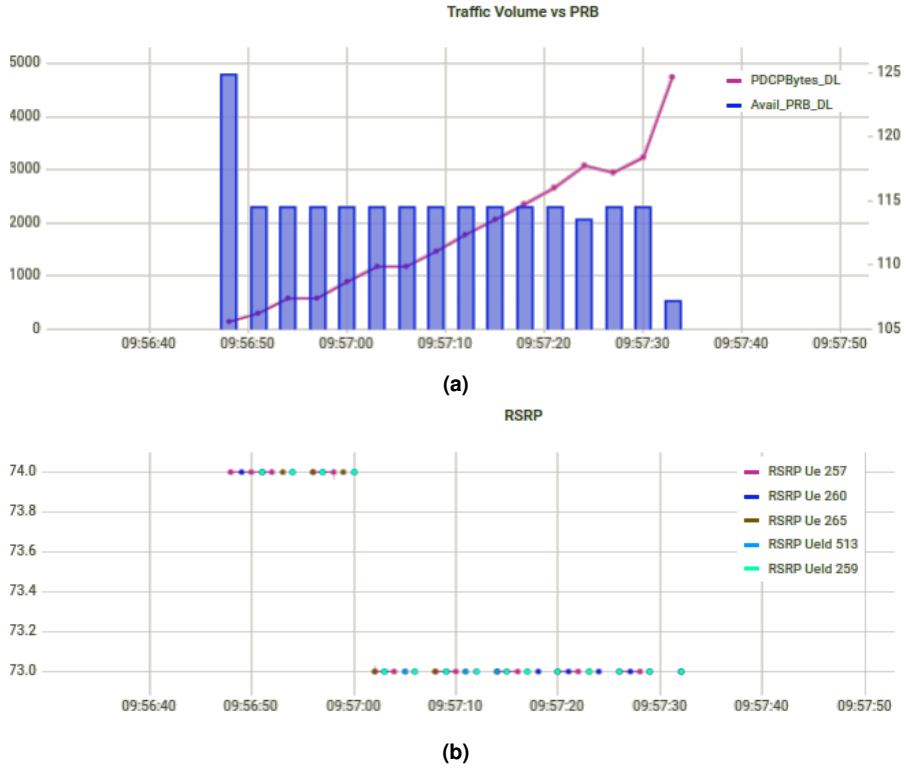

Figura 3. Métricas do xApp *kpimon-go* da OSC: a) relação entre o tráfego e o nível de PRBs por célula e b) medidas de potência para diferentes UEs.

como cenários com mais de uma centena de UEs em um arranjo de sete células hexagonais [Open Networking Foundation 2023].

A Figura 4 ilustra a utilização do simulador de RAN com o xApp *KPI Monitor* para avaliar um cenário denso com cem conexões RRC ativas usando cinco células e três nós E2. Elaborado a partir de arquivos *.yaml*, o cenário pode ser usado para explorar a otimização da área de cobertura, o balanceamento ótimo de tráfego, a predição de falhas de conexão, entre outros. Além disso, a integração com o Prometheus e o Grafana permite o armazenamento de métricas e a visualização gráfica dos dados.

Apesar de denso e exigir esforço nas etapas de depuração, criação de configuração de rede e manipulação dos arquivos de instalação, este *setup* permite a exploração de características do plano de dados, tais como o acesso aos dados de *throughput* e ao recurso do *slicing* de rede.

O xApp RSM é utilizado para realizar um teste de vazão com uso da aplicação *iperf3* em uma rede LTE com um único UE COTS. Um *slice* é configurado com maior largura de banda para uma taxa que gira entre 5 e 7 Mbits/sec. A aplicação é então comutada para um outro *slice*, configurado com uma largura de banda menor, o que reduz o tráfego por um fator entre 4 e 6 para uma taxa de aproximadamente $1,2$ Mbits/sec. É possível observar essa redução na Figura 5, que mostra o xApp com um *slice* de taxa mais elevada entre 3s e 12s e com um *slice* de taxa reduzida entre 40s e 49s.

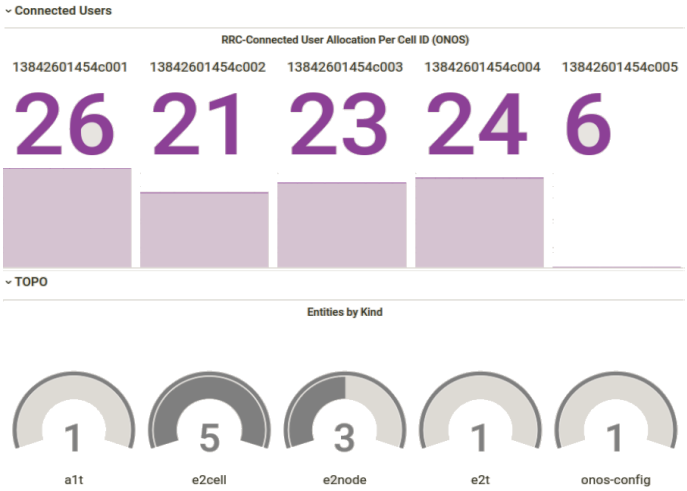

Figura 4. *Setup* com o simulador da ONF RAN utilizando o KPI Monitor em um cenário denso e integrado ao Prometheus e Grafana.

4.3. OAI

O FlexRIC é usado em uma máquina com 16 GB de memória RAM, processador Intel Core i7 e 256 GB de disco rígido. Ele oferece a possibilidade de integração com as pilhas OAI e *software radio system RAN* (srsRAN) através de *patches* que permitem a comunicação dessas pilhas com o RIC.

A fim de demonstrar o xApp de monitoramento de KPIs das camadas MAC, RLC e PDCP (*xApp MAC-RLC-PDCP Monitor*) no FlexRIC, a rede de rádio 5G *Standalone* (SA) é utilizada com uma pilha OAI. Apontamos para o fato de o FlexRIC e a pilha estarem na mesma máquina. Um UE COTS é conectado à rede 5G, e o *patch* do FlexRIC para a OAI é aplicado.

As KPIs coletadas são guardadas no banco de dados InfluxDB. O aplicativo Grafana é usado para consultar a base de dados e mostrar o valor atual e a série temporal de cada KPI. A Figura 6 mostra um painel do Grafana com algumas métricas possíveis de serem monitoradas pelo xApp *MAC-RLC-PDCP Monitor* quando um UE COTS está conectado à rede 5G SA rodando uma aplicação de *streaming* de vídeo. O painel permite

Figura 5. Teste de *slicing* com o *slice* de maior largura de banda à esquerda e o de menor largura de banda à direita.

monitorar informações como a *Block Error Rate* (BLER), os esquemas de modulação e codificação utilizados, os RNTIs dos UEs, a SNR do canal *Physical Uplink Shared Channel* (PUSCH), o *Hybrid Automatic Repeat Request* (HARQ), o número de pacotes retransmitidos e o número de bytes transmitidos.

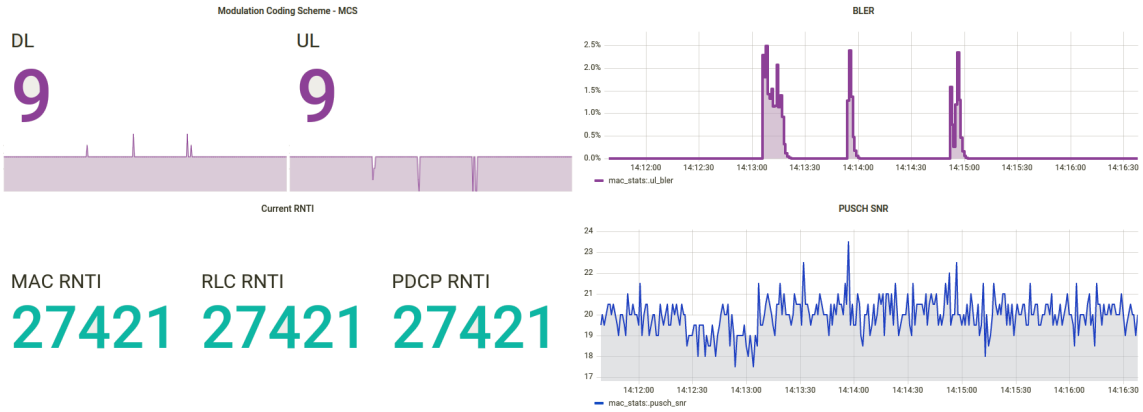

Figura 6. Informações de operação produzidas ou captadas pela pilha OAI, incluindo a BLER, os esquemas de modulação e codificação, os RNTIs relacionados aos SMs da MAC, RLC e PDCP, respectivamente, e a SNR do canal PUSCH.

Além de facilitar o processo de depuração em tempo quase real, os dados armazenados podem ser usados por outros xApps. Considerando que a condição do canal (informada pela SNR) e a taxa de transmissão de cada UE são as entradas de algumas aplicações, os SMs (tal como o da MAC e do PDCP) podem ser adaptados para incluir tais métricas. Com esses KPIs, a detecção de falhas no enlace de rádio, o balanceamento de carga, a detecção de cobertura, a predição do volume de dados, a detecção de interferência e muitas outras aplicações podem ser desenvolvidas na forma de xApps.

4.4. Discussão

Constatamos dos experimentos que cada RIC possui pontos fortes, tendo aplicações em problemas específicos. Considerando que os setups atuais ainda utilizam poucos elementos físicos (gNodeBs ou UEs), entendemos que a velocidade de processamento ou de envio de mensagens dos diferentes RICs ainda não constituem um critério relevante de comparação. Parâmetros mais relevantes incluem possíveis casos de uso, facilidade de implementação destes casos e a compatibilidade com pilhas RAN. O RIC da OSC, por exemplo, demonstra o uso de técnicas de IA e interage com uma maior quantidade de elementos da arquitetura OpenRAN. Em contrapartida, não oferece integração simples com pilhas RAN reais. O RIC da OAI oferece simplicidade de instalação e de reprodução de setups físicos. Por não usar Docker ou Kubernetes, o desenvolvimento e a depuração são mais simples, mas sem vantagens como escalabilidade e estabilidade. O RIC da ONF possui um bom equilíbrio entre facilidade de instalação e desenvolvimento, porém os casos de uso disponíveis ainda não apresentam grande utilidade prática. Deste modo, a utilidade atual dos RICs analisados depende do problema e dos recursos disponíveis.

5. Conclusões e trabalhos futuros

Neste artigo resumimos a função do RIC e três de suas implementações mais relevantes: as versões da OSC, ONF e OAI. Além disso, detalhamos os *testbeds* utilizados para

cada uma delas e especificamos os equipamentos e *setup* utilizados. Para demonstrar o monitoramento de métricas da RAN e o procedimento de *slicing* de recursos físicos de rede, fizemos o *deploy* de *pods* para as soluções OSC e ONF, utilizamos bancos de dados temporais como o InfluxDB e realizamos a integração com ferramentas gráficas como o Grafana e o Prometheus. Com o FlexRIC, utilizamos pilhas da RAN por meio de *patches* e verificamos a possibilidade de alterar a pilha e o próprio FlexRIC a fim de incorporar novas métricas.

Pretendemos realizar como trabalhos futuros (i) a criação de um *setup* de maior escala utilizando no mínimo três gNBs e 64 ou mais UEs, dos quais no mínimo 16 dispositivos físicos, e (ii) incorporar o non-RT RIC, permitindo a orquestração das funções do near-RT RIC. O intuito do novo *testbed* é o de explorar aplicações como a previsão de estatísticas da camada PHY e condições do canal, o monitoramento e a previsão da *Quality of Experience* (QoE) sob condições de *network slicing*, a criação de algoritmos de *scheduling* para *slicing*, a criação de rotinas de localização *indoor* (de onde deriva a necessidade de múltiplas gNBs, isto é, para realizar uma triangulação) e a criação de estratégias de economia de energia da RAN.

Referências

Arnaz, A., Lipman, J., Abolhasan, M., and Hiltunen, M. (2022). Toward integrating intelligence and programmability in open radio access networks: A comprehensive survey. *IEEE Access*, 10:67747–67770.

Bonati, L., D'Oro, S., Polese, M., Basagni, S., and Melodia, T. (2021). Intelligence and learning in O-RAN for data-driven nextG cellular networks. *IEEE Communications Magazine*, 59(10):21–27.

Colosseum (2020). About Colosseum. disponível em:https://colosseumneu.freshdesk.com/support/solutions/articles/61000269573-about-colosseum. acessado: 2023-02-12.

COSMOS (2023). Cloud Enhanced Open Software Defined Mobile Wireless Testbed for City-Scale Deployment. Disponível em: https://www.cosmos-lab.org. Acessado: 2023-03-13 .

O-RAN Alliance (2023). O-RAN Architecture Description 8.0. Disponível em: https://orandownloadsweb.azurewebsites.net/specifications. Acessado: 2023-03-17.

Open Networking Foundation (2022a). onos-exporter. `https://docs.sd-ran.org/master/onos-exporter/README.html?highlight=prometheus`. Accessado: 2022-08-14.

Open Networking Foundation (2022b). onos-kpimon. `https://docs.sd-ran.org/master/onos-kpimon/README.html`. Acessado: 2022-02-11.

Open Networking Foundation (2022c). onos-rsm. `https://docs.sd-ran.org/master/onos-rsm/README.html`. Acessado: 2022-03-19.

Open Networking Foundation (2023). Architecture. `https://docs.sd-ran.org/master/architecture.html?highlight=grpc`. Accessado: 2022-12-23.

Open Networking Foundation (2023). RAN Simulator. `https://docs.sd-ran.org/master/ran-simulator/README.html`. Acessado em: 2023-02-05.

Open Networking Foundation (2023). SD-RAN Documentation. Disponível em: https://docs.sd-ran.org/master/index.html. Acessado em: 2023-02-18.

OpenAirInterface (2022). O-RAN Near-Real-Time RIC. Disponível em: https://gitlab.eurecom.fr/mosaic5g/flexric. Acessado: 2022-12-10.

OpenAirInterface (2023a). Flexric. https://gitlab.eurecom.fr/mosaic5g/flexric/-/tree/dev/examples/xApp. Accessado: 2023-01-13.

OpenAirInterface (2023b). Flexric. https://gitlab.eurecom.fr/mosaic5g/flexric/-/tree/dev/src/sm. Accessado: 2023-01-13.

OpenRAN Brasil (2022). Chamada Pública - Programa OpenRAN@Brasil. https://docs.sd-ran.org/master/architecture.html?highlight=grpc. Accessado: 2022-03-04.

ORAN Software Community (2022a). E2 simulator. https://wiki.o-ran-sc.org/display/SIM/E2+Simulator. Acessado: 2022-11-21.

ORAN Software Community (2022b). Near Real time RAN Intelligent Controller (RIC). Disponível em: https://docs.o-ran-sc.org/en/latest/projects.htmlnear-realtime-ran-intelligent-controller-ric. Acessado: 2022-09-09 .

ORAN Software Community (2022c). OSC Installation Guides. https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#overview, note = Acessado: 2022-12-10.

ORAN Software Community (2022d). ric-app-kpimon-go. https://github.com/o-ran-sc/ric-app-kpimon-go. Acessed: 2022-11-21.

ORAN Software Community (2023). Near Realtime RAN Intelligent Controller (RIC). Disponível em: https://wiki.o-ran-sc.org/pages/viewpage.action?pageId=1179659. Acessado: 2023-01-13.

Polese, M., Bonati, L., D'Oro, S., Basagni, S., and Melodia, T. (2022). ColO-RAN: Developing Machine Learning-based xApps for Open RAN Closed-loop Control on Programmable Experimental Platforms. *IEEE Transactions on Mobile Computing*, pages 1–14.

Redias (2009). Redias database. https://redis.io/. Acessado: 2023-06-05.

RNP (2022). OpenRAN@Brasil. https://www.rnp.br/projetos/openranbrasil. Acessado em: 2022-12-15.

Schmidt, R. (2021). *Slicing in heterogeneous software-defined radio access networks*. Theses, Sorbonne Université.

Schmidt, R., Irazabal, M., and Nikaein, N. (2021). FlexRIC: An SDK for next-Generation SD-RANs. In *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '21, page 411–425, New York, NY, USA. Association for Computing Machinery.