MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

PEDRO PAULO LEAL DE ALBUQUERQUE
RENAN RODRIGUES MOREIRA RESENDE DA SILVA

UM SIMULADOR PARA REDES HETEROGÊNEAS SOB O PARADIGMA "*ALWAYS BEST CONNECTED*"

RIO DE JANEIRO
2021

PEDRO PAULO LEAL DE ALBUQUERQUE
RENAN RODRIGUES MOREIRA RESENDE DA SILVA

UM SIMULADOR PARA REDES HETEROGÊNEAS SOB O PARADIGMA "*ALWAYS BEST CONNECTED*"

Projeto de Final de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientadores: Maria Cláudia Reis Cavalcanti, D.Sc.
David Fernandes Moura, D.Sc.

Rio de Janeiro

2021

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 – Praia Vermelha
Rio de Janeiro – RJ CEP: 22290-270

de Albuquerque, Pedro Paulo Leal; da Silva, Renan Rodrigues Moreira Resende.
Um simulador para redes heterogêneas sob o paradigma "*Always Best Connected*" / Pedro Paulo Leal de Albuquerque e Renan Rodrigues Moreira Resende da Silva. – Rio de Janeiro, 2021.
62 f.

Orientadores: Maria Cláudia Reis Cavalcanti e David Fernandes Moura.

Projeto de Final de Curso (graduação) – Instituto Militar de Engenharia, Engenharia de Computação, 2021.

1. Mininet-WiFi. 2. RDS. 3. wifi. 4. redes. 5. proveniencia. 6. emulador. i. Cavalcanti, Maria Cláudia Reis (orient.) ii. Moura, David Fernandes (orient.) iii. Título

PEDRO PAULO LEAL DE ALBUQUERQUE
RENAN RODRIGUES MOREIRA RESENDE DA SILVA

Um simulador para redes heterogêneas sob o paradigma "*Always Best Connected*"

Projeto de Final de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientadores: Maria Cláudia Reis Cavalcanti e David Fernandes Moura.

Aprovado em Rio de Janeiro, 14 de outubro de 2021, pela seguinte banca examinadora:

Prof. **Maria Cláudia Reis Cavalcanti** - D.Sc. do IME - Presidente

---

Cel **David Fernandes Cruz Moura** - D.Sc. do CTEX

Júlio César Cardoso Tesolin - M.Sc. do IME

Prof. **Ricardo Choren** - D.Sc. do IME

TC **Marcelo José Camilo** - D.Sc. do CTEX

Rio de Janeiro
2021

"Um profissional é aquele que faz o seu melhor trabalho quando menos vontade tem de o fazer". - Frank Lloyd Wright

AGRADECIMENTOS

A todas as pessoas que nos incentivaram, apoiaram e possibilitaram esta oportunidade de ampliar nosso conhecimento e nossos horizontes.

A todos os amigos, familiares e mestres que sempre estiveram presentes.

Em especial a Professora Orientadora Dra. Maria Claudia Cavalcanti, ao Cel David Moura e ao Júlio Tesolin pela disponibilidade e atenção fora do comum.

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito.
(Bíblia Sagrada, Romanos 12, 2)

RESUMO

Este trabalho está relacionado à iniciativas de Pesquisa e Desenvolvimento na área de redes de comunicações sem fio, destinando-se a auxiliar pesquisas existentes e futuras nas execuções de simulações de redes. Para que seja possível atingir esse objetivo, será desenvolvido um software que se integre ao Mininet-WiFi (emulador de redes já existente) para fornecer uma interface de execução de emulações para redes, disponibilizando o histórico de experimentos com seus respectivos resultados. Por fim, ao final do desenvolvimento da ferramenta, testes com séries de emulações serão feitos e validados junto aos interessados no projeto, para assim disponibilizar o sistema para uso.

Palavras-chave: Mininet-WiFi. RDS. wifi. redes. proveniencia. emulador.

ABSTRACT

This work is related to Research and Development initiatives in the area of wireless communication networks, and is intended to assist existing and future research in network simulation runs. In order to achieve this goal, a software that integrates with Mininet-WiFi (an existing network emulator) will be developed to provide an interface for network emulalation execution, making available the history of experiments with their respective results. Finally, at the end of the tool development, tests with sequence of emulations will be made and validated with those interested in the project, to make the system available for use.

Keywords: Mininet-WiFi. SDN. wifi. networks. provenience. emulation.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

MVC	Model-View-Controller
RSSI	Received Signal Strength Indication
IP	Internet Protocol
SSID	Service Set Identifier
AP	Access Point
UML	Unified Modeling Language
SDN	Software Defined Networking
CLI	Comamand Line Interface
ER	Entidade-Relacionamento

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação

Os meios de comunicações táticos eficientes, sofisticados e seguros são imprescindíveis para garantir a capacidade do Exército Brasileiro no cumprimento de suas atividades e objetivos. Neste sentido, surgiu a necessidade de iniciativas de Pesquisa e Desenvolvimento nessa área. Ao longo do tempo, a trajetória e a evolução das comunicações possuem forte relação com essas iniciativas na área de rádios voltados para comunicações táticas.

Ainda nessa evolução, foi adotado o paradigma do Rádio Definido por Software (RDS), que não faz o uso de um ponto central de interoperabilidade, e, portanto, garante soluções mais seguras para a comunicação. No ano de 2012 foi criado o programa RDS-Defesa (3) (Rádio Definido por Software do Ministério da Defesa), com responsabilidade delegada ao Centro Tecnológico do Exército (CTEx). O programa faz uso do modelo de inovação de tripla hélice (integração com a indústria, academia e governo) para garantir seu desenvolvimento, com altos investimentos e resultados consistentes.

O programa RDS-Defesa possui uma grande abrangência, considerando os diferentes recursos possibilitados por redes cognitivas. Uma das frentes de inovação está no estudo de modelos de propagação de ondas e seus impactos nos índices de qualidade de uma rede de comunicação móvel, onde se faz o uso de diversas simulações para avaliar os resultados em inúmeros cenários e condições.

Essas simulações são feitas através de alguns softwares, como o Mininet-WiFi (2), onde o pesquisador pode controlar as condições do experimento e analisar os resultados encontrados. Porém, o desenvolvimento de novas ferramentas que auxiliem esse processo ou incrementem a produtividade faz-se necessário, pois fornece melhores condições para a realização das pesquisas e manipulação dos dados encontrados.

Além disso, o apoio aos trabalhos desenvolvidos pelos cientistas se tornam ainda mais necessários no contexto de e-Ciência, onde a rastreabilidade, confiabilidade, recuperação e comparação dos resultados se tornam mais desafiadoras devido ao volume de dados que estão sob manuseio. Este cenário demanda não somente melhorias nas ferramentas utilizadas, mas também a disponibilização de novos recursos que atendam às essas necessidades.

1.2 Objetivo

Desenvolver um software integrado ao Mininet-WiFi (emulador de redes já existente) que permita aos pesquisadores, por meio de uma interface amigável, a gerência da execução

das simulações de rede, disponibilizando o histórico de experimentos com seus respectivos resultados.

1.3 Justificativa

Atualmente, o emulador de redes Mininet-WiFi é largamente utilizado em experimentos de simulação de redes sem fio, possuindo grande capacidade de extensão de suas características e de seu escopo, confirmando assim a sua relevância neste domínio de pesquisa. Porém, durante o desenvolvimento dos trabalhos, um grande número de experimentos é feito para encontrar e confirmar os resultados a partir de condições previamente configuradas. Diante do alto número de execuções, evidencia-se algumas necessidades para a melhoria na produtividade e no acesso aos simuladores.

A primeira delas é prover uma interface que possibilite a realização (criação, execução e análise dos resultados) de experimentos sem a necessidade de conhecimento profundo de quaisquer linguagem de programação. Atualmente, as simulações no Mininet-WiFi são feitas por meio de programas escritos na linguagem Python, com a utilização de algumas ferramentas adicionais como o MiniEdit para a criação de topologia (1). Dessa forma, caso o pesquisador não esteja familiarizado com a linguagem de programação exigida, a curva de aprendizado, a produtividade e desenvolvimento das pesquisas tendem a ser lentas. Além disso, a centralização das ferramentas utilizadas e da exibição dos resultados facilita a condução do trabalho, considerando que fornece um único ponto de contato com a aplicação, eliminando a complexidade do manuseio de diferentes ferramentas e programas.

Outro aspecto que representa uma limitação para o desenvolvimento das pesquisas é a falta de uma ferramenta de captura dos experimentos feitos e resultados obtidos, sendo de inteira responsabilidade do usuário do simulador o registro e organização de suas realizações. Isso pode se tornar uma barreira para a gestão do conhecimento e do progresso obtido à medida que o número de experimentos se torna expressivo, pois pode se tornar inviável a recuperação de experimentos anteriores, caso os usuários não se comprometam com a gestão desses registros.

Nesse sentido, existem diversas iniciativas e projetos que visam a captura da proveniência dos dados de experimentos em diversas áreas do conhecimento, como proposto por Pérez, Rubio e Sáenz-Adán(4), que realiza a análise e comparação entre eles. Porém, essas iniciativas apresentam um alto grau de generalização, de forma que não conseguem atender os requisitos das simulações em redes militares de uma maneira geral. Surge a necessidade, então, de um modelo único de proveniência para esse tipo de experimentos.

1.4 Metodologia

O projeto é dividido em três etapas, que diferem entre si quanto a natureza de suas atividades. A primeira etapa consiste em um aprendizado teórico sobre conceitos básicos relacionados a Redes Móveis Cognitivas, Rádio Definidos por Software e Simulações de Redes. Inclui-se também nessa etapa o estudo sobre o emulador de redes Mininet-WiFi e sobre modelos de proveniência de dados.

A segunda etapa se trata da compreensão das necessidades que o sistema a ser desenvolvido deve atender e a respectiva viabilidade. Aqui se enquadram entregas como o levantamento de requisitos e desenvolvimento de programas simples para avaliar se é possível atender as especificações.

Por fim, realiza-se a implementação do projeto em si, seguindo as boas práticas do desenvolvimento de software. Neste momento, são feitas as modelagens, o desenvolvimento e os testes.

1.5 Contribuições esperadas

Espera-se que o software desenvolvido auxilie no desenvolvimento de pesquisas na área de redes de comunicação sem fio, sendo utilizado para a gestão e realização de experimentos e simulações, de modo a permitir uma análise comparativa entre os resultados encontrados nas diversas execuções.

1.6 Estrutura

Este documento especifica as etapas do projeto desde a sua concepção até a sua finalização, passando pelo seu desenvolvimento. Esta especificação é feita de acordo com a estrutura descrita a seguir.

O Capítulo 2 trata da fundamentação teórica que será a base para a realização do projeto. Este capítulo abrange os conceitos de Redes de Comunicação Móveis, Mininet-WiFi e Proveniência de dados. O Capítulo 3 especifica a abordagem realizada para o desenvolvimento do projeto. Já o Capítulo 4 aborda sobre o desenvolvimento do sistema. Por fim, o Capítulo 5 apresenta as conclusões e trabalhos futuros.

2 CONTEXTUALIZAÇÃO

2.1 Redes de Comunicação Móveis

Uma rede de comunicação pode ser definida como um conjunto de dois ou mais dispositivos, denominados nós de rede, capazes de se comunicar por meio de enlaces. Essa comunicação é feita através de um conjunto de processos e regras chamado de protocolo de rede.

Nesta definição, um dispositivo pode ser um *host* (ou um sistema final, como às vezes é chamado), tal como um grande computador, *desktop*, *laptop*, estação de trabalho, telefone celular ou sistema de segurança. Um nó de rede nessa definição também pode ser um dispositivo de conexão, tal como um roteador, que liga uma rede a outras redes, um *switch* (ou comutador) que liga dispositivos entre si, um modem (modulador-demodulador) que altera a forma dos dados, e assim por diante. Tais dispositivos em uma rede são conectados usando meios de transmissão com ou sem fio.

O campo das comunicações sem fio, de uma maneira geral, vem sendo um dos temas mais estudados na área de telemática. Embora tenha surgido no final do século XIX, as atividades nessa área continuam intensas até os dias de hoje devido a grande demanda por conectividade. Impulsionado, inicialmente, pela conhecida telefonia celular e, mais recentemente, por aplicativos de dados, novas formas de comunicação e compartilhamento de acesso a conteúdos, as comunicações sem fio não param de surpreender com o surgimento de novas tecnologias. Diversas delas são conhecidas e estão bastante presentes no cotidiano como o Bluetooth, WiFi (IEE 802.11), e também, mais recente e em expansão, a Internet das Coisas (IoT), que se refere à interconexão de objetos do cotidiano como celulares, lâmpadas, ventiladores, geladeiras, entre outros.

Esse grupo de tecnologia de comunicações possui características únicas que as tornam distintas das demais tecnologias de redes cabeadas. Uma delas, e certamente a mais predominante, é a propagação de ondas de rádio.

Um sinal propagando-se de um ponto a outro sofre fenômenos como a atenuação, que se refere a perda de percurso que ocorre quando o receptor se afasta da fonte. O fenômeno de perda de percurso em função da distância física entre uma estação base e um hospedeiro sem fio, a exemplo da atenuação do nível do sinal, pode ser estimado estimado por diferentes modelos de propagação da literatura (2) (Free-Space, Log-Distance, ITU) que estão disponíveis no emulador estudado em 2.2. A intensidade do sinal propagado também pode sofrer desvanecimentos lentos devido às obstruções presentes no percurso como árvores e prédios, assim como desvanecimentos rápidos devido ao fenômeno de múltiplos

caminhos percorridos. Outro fenômeno também bastante relevante nas propagações de ondas rádio é a interferência, que é causada por serviços que utilizam a mesma frequência ou frequências próximas.

Questões como a interferência impulsionam a pesquisa e desenvolvimento de rádios cognitivos. Um rádio cognitivo (5) é um rádio que percebe e está ciente do seu ambiente operacional e pode, de forma dinâmica e automática, ajustar seus parâmetros operacionais de acordo com seu ambiente, ou seja, ele muda seus parâmetros de transmissão com base na interação com o ambiente que ele está operando. Os processos de decisão podem ser feitos com base em algoritmos de inteligência artificial, utilizando cenários de experiências anteriores.

Os padrões de redes de comunicação sem fio tem basicamente duas arquiteturas(2). Na primeira, a arquitetura é definida como um conjunto básico de serviço, ou modo infraestruturado. Nela toda a comunicação entre os nós clientes deve passar por uma estação base denominada ponto de acesso, que possui uma identificado e possui certas características como associação. Na segunda arquitetura a comunicação é feita diretamente entre os clientes, sem a necessidade de um ponto de acesso. Essa disposição é definida como um conjunto básico de serviço independentes, ou modo adhoc. Essas estruturas são ilustradas na Figura 1

Apesar das diversas tecnologias de comunicações sem fio existentes, a mais conhecida e uma das mais utilizadas atualmente é o conjunto de padrões IEE802.11, popularmente conhecido como Wi-Fi, que geralmente opera em uma arquitetura infraestruturada.

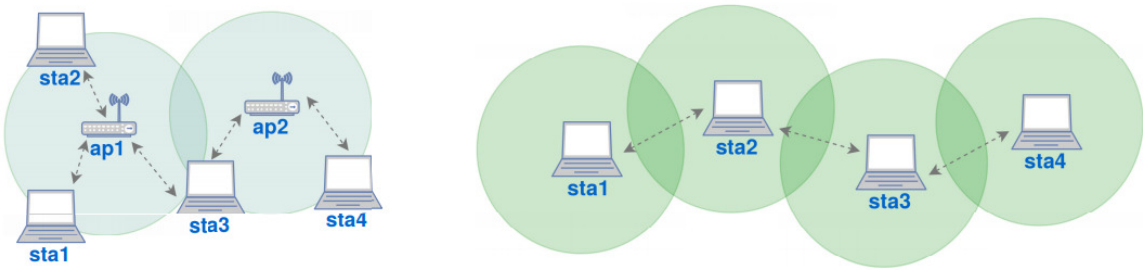

Figura 1 – Modo Infraestruturado e Modo Adhoc, Respectivamente (1)

O Wi-Fi com o passar dos anos passou por diversos processos de evolução, sendo assim existem diversos padrões do IEE802.11 que são resultados de melhorias realizadas ao longo dos anos. Entre os padrões mais antigos estão o 802.11b, 802.11a e 802.11g. Enquanto os considerados mais recentes são os exemplos do 802.11n, 802.11ac, 802.11p, etc. Sendo que de modo geral, esse conjunto de padrões operam em duas faixas de frequências principais: 2.4 GHz ou 5 GHz, onde são alocados um número de canais com uma largura específica(1). A pesquisa em cima desse conjunto é uma tendência que é de constante interesse na comunidade científica.

Outro área de pesquisa bastante relevante atualmente é a de Redes Sem Fio Definidas por Software(2) que consiste em uma abordagem que permite o controle centralizado da rede através de aplicações que não estão localizadas necessariamente em um ponto de acesso, ou seja, há controladoras com base em software ou APIs que direcionam o tráfego na rede para se comunicar com a infraestrutura de hardware subjacente, ditando assim todo o comportamento da rede. Em redes tradicionais, são usados dispositivos de hardware dedicados (roteadores e *switches*) para o controle do tráfego de rede.

O estudo na área de Redes Sem Fios Definidas por Software traz consigo outras oportunidades de pesquisa, como a possibilidade da interconexão de redes heterogêneas. Uma rede heterogênea pode ser definida como uma rede com uma arquitetura que utiliza diferentes tecnologias e padrões de comunicação de maneira colaborativa. Pode-se tomar como exemplo uma rede que utiliza em uma parte de sua topologia o conjunto de padrões IEEE802.11 enquanto em outra parte da mesma tecnologia utiliza-se o conjunto de padrões IEEE802.16 (6).

O universo das comunicações sem fio é uma vasta área de pesquisa que possui a tendência de evoluir cada vez mais com o tempo.

2.2 Mininet-WiFi

O campo de estudo na área de redes de computadores é constantemente explorado por diversos pesquisadores. Nesse contexto, a emulação de redes tem sido bastante utilizada na avaliação de desempenho, testes, depuração de protocolos, entre outros parâmetros desse campo. Nesse contexto, a emulação realista dos ambientes de redes sem fio tem o potencial de impulsionar os esforços de pesquisa e desenvolvimento reutilizando os protocolos e as pilhas de aplicativos existentes, imitando o comportamento de redes sem fio reais.

Pode-se encontrar algumas alternativas, além dos emuladores, para um pesquisador avaliar e validar protocolos de rede existentes, bem como efetuar análises, entre outros objetivos. Simuladores e testbeds também estão entre os principais métodos de avaliação que auxiliam o pesquisador. Em relação à aplicação real, todos estes métodos de avaliação são muito diferentes no seu grau de abstração, tendo cada um suas vantagens e desvantagens. Das diferenças existentes entre essas plataformas experimentais, pode-se destacar a grande vantagem do uso dos emuladores pelo fato deles utilizarem a pilha de protocolos de redes reais, possibilitando até mesmo colocar esses equipamentos emulados em comunicação com equipamentos reais de laboratórios, ou com outros dispositivos disponíveis comercialmente. Essa possibilidade de ver a modelagem ser validada por equipamentos reais não é possível com simuladores, que apenas “imitam” a atividade de algo que está simulando. Esse fato também faz com que simuladores de redes de modo geral obtenham resultados muito parecidos em simulações repetidas, o que não é vantajoso para análise de experimentos (2).

O Mininet-WiFi (2) é um emulador para redes sem fio e de código fonte aberto estendido a partir do Mininet, emulador bastante conhecido por pesquisadores que já atuam na área das redes definidas por software. Deve-se ressaltar que a emulação de redes sem fio possui características peculiares quando comparada com emuladores para redes cabeadas. Existe a necessidade de implementar características específicas das redes sem fio, como a mobilidade dos nós e a propagação do sinal, para que seja possível realizar experimentos em ambientes onde existem interferências, atenuação de sinal etc.

O fato desse emulador ser de código fonte aberto é uma outra grande vantagem em relação à maioria dos simuladores, que geralmente possuem custos bastante elevados. Outro fato é que simuladores muitas vezes possuem uma implementação complexa, necessitando de uma grande preocupação com elementos de camadas, que muitas vezes podem não ser tão relevantes para a pesquisa. A maiorias das funcionalidades do emulador são implementadas através de *scripts* relativamente compactos em relação aos simuladores existentes.

Apesar do nome Mininet-WiFi, o simulador provê suporte para outras tecnologias de comunicação sem fio. Possibilita virtualização de estações, pontos de acesso, controladoras, comutadores e seus respectivos enlaces. Todo esse processo de virtualização tem como base processos que são executados em Linux *Network Namespaces* e placas de rede virtuais.

A arquitetura do sistema é ilustrado na Figura 2 . O Linux *Network Namespaces*, de forma lógica, representam uma cópia da pilha de rede do sistema operacional Linux, que inclui suas próprias rotas, regras de *firewall* e dispositivos de rede. Eles atuam como se fossem verdadeiros computadores com as mesmas propriedades de rede que um computador físico pode ter.

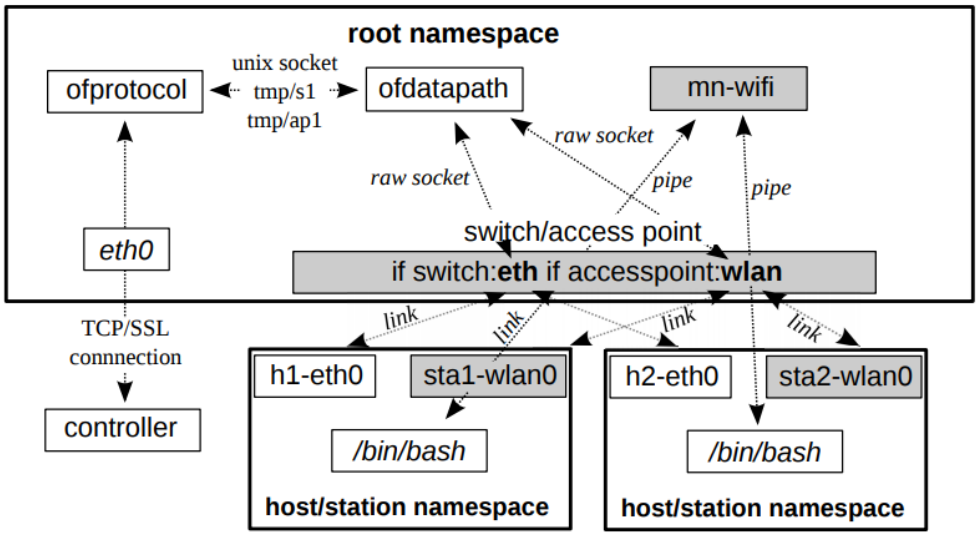

Figura 2 – Arquitetura do Mininet-WiFi (2)

2.2.1 Funcionamento

Após instalado, o Mininet-WiFi pode ser iniciado através do seguinte comando:

> sudo mn –wifi

O resultado dessa execução pode ser visto na figura 3. Além de abrir a interface de linha de comando (CLI) do Mininet-WiFi, esse comando irá criar uma topologia que consiste em 2 (duas) estações conectadas a 1 (um) ponto de acesso através de um meio sem fio com um SSID chamado de “my-ssid”, operando no canal 1 (frequência 2412MHz), valores esses que podem ser customizados. Além disso, também há a presença de 1 (um) controlador SDN que está conectado ao ponto de acesso. Essa topologia é ilustrada na figura 4.

```
*** Adding stations:
sta1 sta2
*** Adding access points:
ap1
*** Configuring wifi nodes...
*** Creating network
*** Adding controller
*** Adding hosts:

*** Adding switches:

*** Adding links:
(sta1, ap1) (sta2, ap1)
*** Starting controller(s)
c0
*** Starting L2 nodes
ap1 ...
*** Starting CLI:
mininet-wifi>
```

Figura 3 – Inicialização do Mininet-WiFi

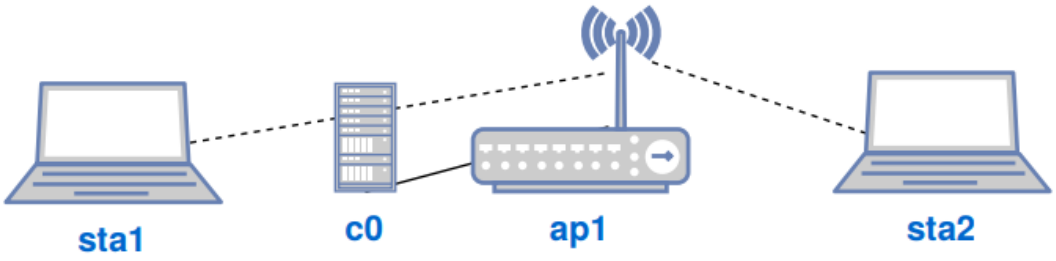

Figura 4 – Topologia Padrão do Mininet-WiFi (1)

Comandos como pings entre estações para testes de conectividade ou comandos para mensurar largura de banda entre dois nós podem ser feitos na CLI.

A criação de topologias de rede e configurações para emulação são feitas através de *scripts* escritos na linguagem python.

Na Figura 5 está representado um trecho de um *script* criando uma topologia, com estações, pontos de acesso, *switch*, *host*, controladoras e enlaces especificados com configurações personalizadas.

```
def topology():
    "Create a network."
    #net = Mininet_wifi(controller=Controller, accessPoint=OVSKernelAP)
    net = Mininet_wifi(controller=Controller, accessPoint=UserAP,
        link=wmediumd,wmediumd_mode=interference,
        noise_th=-91, fading_cof=3)
    #Wireless
    info("*** Creating nodes\n")

    sta1 = net.addStation('ue01',
    min_x=5, max_x=100, min_y=10, max_y=100, min_v=5, max_v=10,
    bgscan_threshold=-75, s_inverval=5, l_interval=10, bgscan_module="simple")

    ap1 = net.addAccessPoint('ap1', ssid='ssid-ap1', mode='a', channel='1',
                             mac='00:00:00:00:00:02', ip='10.0.0.1/8',position='15,30,0', range=40)
    ap2 = net.addAccessPoint('ap2', ssid='ssid-ap1', mode='a', channel='6',
                             mac='00:00:00:00:00:03', ip='10.0.0.2/8',position='55,80,0', range=40)
    ap3 = net.addAccessPoint('ap3', ssid='ssid-ap1', mode='a', channel='9',
                             mac='00:00:00:00:00:04', ip='10.0.0.3/8',position='80,30,0', range=40)

    #Wired
    s3 = net.addSwitch('s3')
    h1 = net.addHost('h1')
    c1 = net.addController('c1', controller=Controller)

    net.setPropagationModel(model="logDistance", exp=4)

    info("*** Configuring wifi nodes\n")
    net.configureWifiNodes()

    info("*** Creating links\n")
    net.addLink(ap1, s3)
    net.addLink(ap2, s3)
    net.addLink(ap3, s3)
    net.addLink(s3, h1)
```

Figura 5 – Trecho do *Script* Criando uma Topologia Personalizada

A documentação detalhada de como criar esses *scripts*, configurar e executar os comandos disponíveis são encontradas no repositório disponibilizado em (2).

2.3 Proveniência de Dados

A simulação de uma rede de comunicação móvel, como qualquer experimento científico, consiste em construir um ambiente com atributos específicos e, a partir dele, realizar medições de parâmetros pré-estabelecidos, com o objetivo de validar ou invalidar uma hipótese inicial. Pode-se dividir esse processo em 3 etapas: definição do experimento, realização e coleta dos resultados. Além disso, as simulações são feitas continuamente para avaliar cenários que, em muitos casos, diferem entre si em pequenos detalhes.

Com o crescente volume de dados e o avanço tecnológico, o número de experimentos feitos sobre um objeto de pesquisa para construir uma tese pode ser tão grande que torna

inviável o manuseio com as ferramentas tradicionais. O controle de todas as condições testadas, com suas respectivas saídas, somados a necessidade de consecutivas tentativas para encontrar um certo resultado e o compartilhamento das medidas encontradas se tornam desafio em contextos como esse.

Neste contexto, ganha relevância o conceito de proveniência de dados. Segundo Cruz(7), a proveniência abrange a representação, o registro e a recuperação das informações (sem ambiguidade) sobre as atividades feitas por um ou mais agentes em um determinado contexto. Em um experimento científico, a proveniência permite a recuperação de forma sequenciada dos dados históricos gerados pelas execuções feitas e o compartilhamento em um formato homogêneo. Dessa forma, uma modelagem adequada da representação dos processos envolvidos em um experimento é fundamental.

As capturas dos dados em um esquema de proveniência são divididas entre retrospectiva e prospectiva. A primeira aborda a coleta dos dados gerados por um experimento executado. Já a segunda, trata da sequência de etapas realizadas para a captura de um dado, bem como as condições iniciais. Para as simulações de redes, serão utilizadas ambos os tipos de captura, pois o interesse do usuário está em registrar tanto as condições de realização do experimento (topologia de rede, parâmetros utilizados, definição de métricas a serem coletadas, entre outros) quanto os resultados encontrados (valores para as medições configuradas, relacionamento entre os componentes da rede, tempo de execução do experimento, entre outros).

Outras questões acerca da representação dos dados são os momentos de captura dos dados e o nível. Como o interesse não está no registro do funcionamento do simulador em si, mas nas entradas e saídas do experimento, a proveniência não será aprofundada para registrar o comportamento da ferramenta. Então, apesar da simulação consistir em execuções de funções do Mininet-WiFi, não será feita a captura dessa interação e procedimentos internos durante a execução. Dessa forma, a modelagem fica focada em atender as necessidades do usuário, que abrange somente as configurações, definidas em uma etapa antecessora à execução da simulação, e aos resultados encontrados.

Duas técnicas para a captura das informações podem ser mencionadas: anotação e inversão. A inversão registra as operações realizadas, possibilitando a reconstrução dos artefatos quando requisitados. Já a anotação define os elementos a serem registrados antes da execução do experimento em si. Esta é a ideal para o projeto, pois é especificado previamente (seja pelo pesquisador ou pelo próprio programa) os dados que serão coletados na realização dos experimentos.

Existem diversas abordagens para a construção de um modelo que represente a proveniência. O PROV-DM (8) define uma estrutura base a ser estendida de forma a atender as especificidades de cada requisito. Esse modelo consiste em três elementos: Entidades, Atividades e Agentes.

As Entidades são representações de conceitos ou objetos com atributos fixos. Elas são manipuladas pelas Atividades, que podem ser definidas como operações realizadas ao longo de um período de tempo agindo sobre ou com as Entidades. Dessa forma, as Atividades podem se relacionar com as Entidades tanto para a utilização desses recursos (leitura das informações, para comunicação com outras Atividades por exemplo) quanto para a criação delas. Por fim, os Agentes são elementos que possuem a responsabilidade pela execução das Atividades. Outro relacionamento possível para um Agente é a atribuição às Entidades, como se o Agente fosse o dono, proprietário, responsável semanticamente pelo objeto ou conceito representado pela Entidade.

A partir disso, pode-se registrar a geração de dados em um processo, como sendo a manipulação de Entidades (criação, atualização e atribuição) em Atividades atreladas aos seus Agentes. O diagrama abaixo exibe os conceitos aqui apresentados.

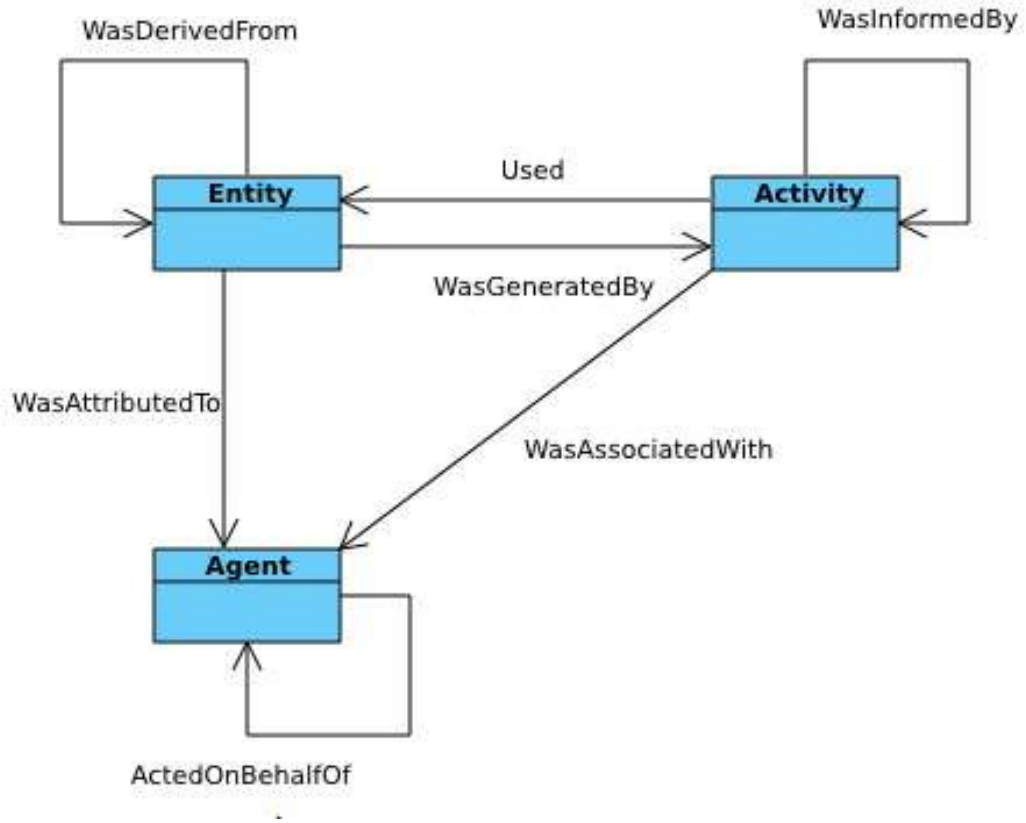

Figura 6 – Estrutura Central de um Modelo de Proveniência de Dados

É válido destacar que um esquema bem definido de proveniência de dados não se limita a uma representação abrangente e superficial, visto que deve satisfazer a semântica das informações. Portanto, para o caso abordado neste projeto, a modelagem deve abranger os conceitos de redes envolvidos nas simulações a serem feitas. Dessa maneira, é garantido o correto relacionamento entre os elementos estruturais e conceituais de um experimento.

Na figura abaixo, pode-se visualizar uma modelagem inicial de como será feita a

proveniência, utilizando os conceitos do PROV-DM (8). As rodadas consistem na execução de uma simulação, a partir de seus elementos de configuração (especificados na versão), definidos pelo usuário e gerando os respectivos resultados. Destaca-se aqui, também, o conceito de plano de teste que agrupa versões de forma a reutilizar as definições feitas. Esse conceito é importante para a praticidade do registro das atividades, utilizando-se da pouca diferenciação das entradas em experimentos consecutivos, apresentando uma alta taxa de repetição.

Um detalhe é que a relação entre a entidade Plano de Teste e Usuário gera uma redundância na modelagem. Porém essa característica será explorada para que a recuperação dos planos de teste de um usuário seja feita de forma eficiente, sem a necessidade de acessar todas as Rodadas para que encontre as versões e então os planos de teste.

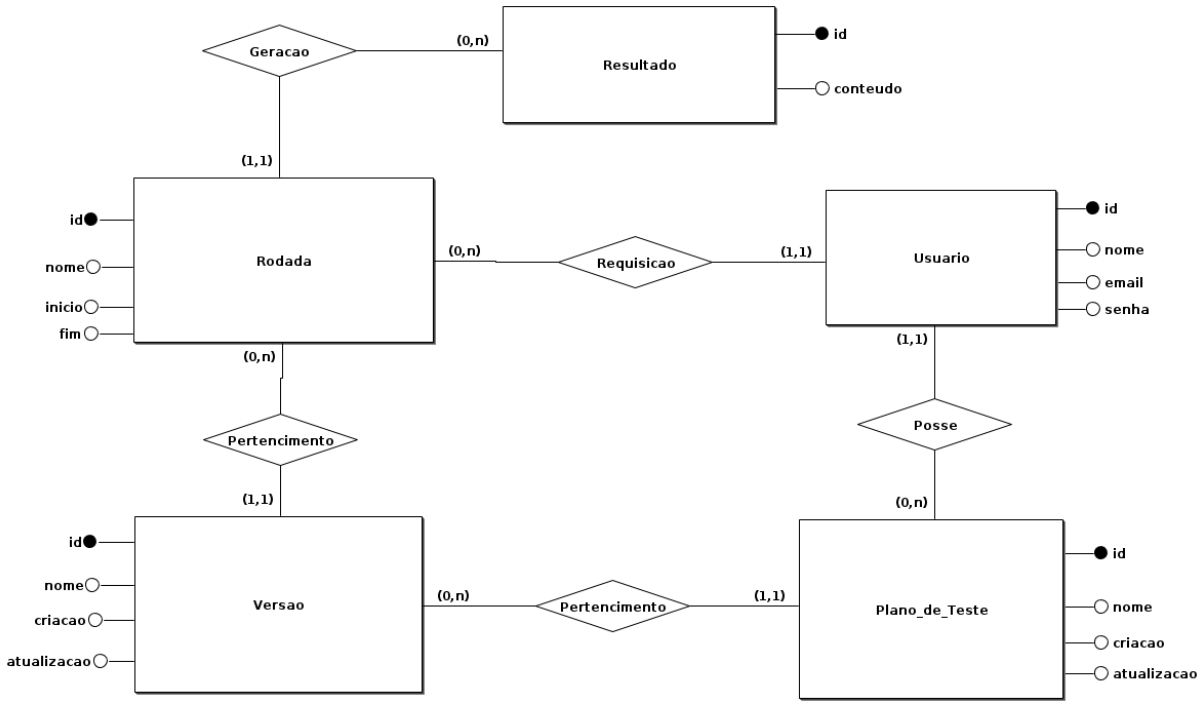

Figura 7 – Modelo ER da Estrutura Central de Proveniência Utilizada no Projeto

3 ABORDAGEM PROPOSTA

3.1 Visão geral

O ponto de partida foi o estudo sobre as definições de Rádios Cognitivas, Redes Heterogêneas, Redes Definidas por Software e outros tópicos de contextualização para o projeto. Logo após, foi feito um estudo sobre o simulador de redes Mininet-WiFi, incluindo atividades práticas como a criação de simulações de redes, justificada pela necessidade de ambientação à experiência dos potenciais usuários do projeto a ser desenvolvido, bem como a verificação da cobertura de funcionalidades e dos recursos existentes e as demandas oriundas da limitação do software. Essa atividade foi seguida pelo desenvolvimento de um estudo de viabilidade quanto a integração do Mininet-WiFi com outros softwares. Neste momento, foi desenvolvido uma prova de conceito (POC) que demonstrasse essa integração. Essa POC foi feita com a linguagem Python, utilizando a biblioteca tkinter para a interface gráfica e basicamente possibilita a realização de experimentos simples.

Como próxima etapa, foi realizado o levantamento de requisitos através de entrevistas sobre as necessidades e desafios encontrados pelos pesquisadores que utilizam o simulador. Com os requisitos já especificados, foi feito um estudo sobre possíveis abordagens para o modelo de proveniência dos dados e como poderiam ser aplicados para atender os requisitos do projeto.

Uma vez contextualizado, ciente dos requisitos do projeto e com a viabilidade checada, iniciou-se as diversas modelagens para o desenvolvimento do projeto. O diagrama e a descrição dos casos de uso foram desenvolvidos, seguido pela modelagem conceitual e lógica do banco de dados, além da decisão de qual SGBD adotar. Por fim, foi feita a idealização das telas da aplicação.

Após isso, inicia-se a implementação da modelagem feita, seguindo as boas práticas e padrões de projeto.

3.2 Levantamento de Requisitos

3.2.1 Requisitos Funcionais

- Ser capaz de configurar uma rede sem fio, simular seu comportamento, acompanhar e analisar os resultados;

- Cada pesquisador pode listar, cadastrar, alterar e excluir(CRUD) qualquer plano de teste associado a seu usuário;

- Cada pesquisador precisa ser autenticado e relacionado a um usuário;

- Cada plano de teste possui apenas uma configuração de rede, que pode ser infraestruturada ou adhoc. As rede adhoc só podem ser compostas de estações móveis. As redes infraestruturadas precisam ter pelo menos um estação móvel, um ponto de acesso e uma controladora;

- Cada plano de teste possui ao menos uma versão. Cada versão possui ao menos uma rodada;

- Em cada versão, é necessário configurar as características dos elementos de rede (potência de transmissão, bitrate, perda por propagação, entre outros), o comportamento dos elementos de rede (movimentação das estações móveis, rompimentos de enlaces, perda de energia da estação, entre outros) e os testes de desempenho que as estações móveis devem realizar (latência, taxa de transmissão, entre outros);

- Cada rodada representa uma simulação feita pelo pesquisador, seguindo as configurações definidas pela versão e plano de teste;

- O pesquisador deve ser capaz de acompanhar os resultados ao longo da rodada;

- O pesquisador deve ser capaz de exportar os resultados de cada rodada;

- A plataforma deve, se possível, prover suporte internacional (inglês).

3.2.2 Requisitos Não Funcionais

- O simulador deve ser construído obedecendo a arquitetura MVC;

- A interface de utilização pode ser feita tanto com tecnologias Web quanto Desktop. Porém, elas devem ser de fácil portabilidade (a instalação da plataforma de execução de simulações não deve apresentar dificuldades consideráveis na sua instalação);

3.2.3 Integração com o Mininet-WiFi

O escopo do projeto não irá abranger todas as funções existentes no Mininet-WiFi. Foi especificado durante o levantamento de requisitos quais funcionalidades do emulador atenderão as necessidades do pesquisador cliente. Para essa especificação, o projeto irá utilizar os programas de exemplo, que estão contidos no código fonte (2), como referência. O projeto deverá contemplar, pelo menos, a execução dos seguintes scripts:

-telemetry.py - Esse script exemplifica a criação de uma topologia de rede e a exibição gráfica em tempo real de parâmetros dos nós da emulação. As informações dos nós que serão cobertos no projeto serão: nome, RSSI, canal, banda, SSID, taxa de transmissão, associação à outro nó e endereço IP.

-position.py - Esse script mostra como definir o comportamento espacial dos nós existentes na topologia da rede, como a posição inicial e movimentação.

-handover_bgscan.py - Esse script aborda o conceito do "always best connected", no qual o dispositivo fica realizando uma varredura de sinal em segundo plano, verificando se dentro de seu alcance existe sinal de algum ponto de acesso, e se existir, qual o sinal de melhor qualidade para estabelecer conexão.

O paradigma "always best connected" abordado pelo simulador é a escolha para conexão do ponto de acesso que possui o melhor nível de sinal, porém existem outras abordagens, como por exemplo, sendo a melhor escolha o ponto de acesso que possui uma melhor relação sinal ruído. Vale ressaltar que não será o objetivo do projeto realizar alterações no funcionamento e no comportamento do Mininet-WiFi.

3.3 Arquitetura

De forma a atender os requisitos, a arquitetura será composta por uma aplicação seguindo o padrão MVC conectado à um banco de dados para a persistência.A arquitetura pode ser divida em alguns componentes de acordo com sua natureza.

O primeiro componente é o Mininet-WiFi, simulador de redes já abordado anteriormente e que será integrada à aplicação principal (MiniManager). Já o segundo se trata do banco de dados, responsável por persistir os dados da aplicação. O Sistema Gerenciador de Banco de Dados escolhido foi o PostgreSQL e um maior detalhamento será feito em um capítulo posterior. O terceiro e último componente é o MiniManager, uma aplicação web escrita na linguagem Python seguindo o paradigma MVC (Model-View-Controller) e implementada através do framework Django. Esta aplicação é fragmentada em alguns módulos de acordo com suas responsabilidades.

O primeiro módulo, cujo nome é “Configurator”, abrange a configuração de simulações. O seu escopo inclui a criação e visualização de Planos de Testes e Versões. Como a criação de novas Versões inclui a configuração de um experimento, esse módulo possui essa atribuição também. Então, pode-se listar as seguintes funcionalidades: geração de formulários para cadastros de Planos de Testes e Versões, gestão das configurações disponíveis no sistema, validação dos dados de entrada do usuário e registro das informações coletadas no banco de dados.

O “Experimenter” se trata do componente que faz o gerenciamento dos experimentos que estão e serão executados no sistema. Sendo assim, ele possui uma interface com o usuário (Cientista) para recebimento de requisições de execução de novas Rodadas, integração com o módulo “Configurator” para recuperação da configuração da simulação solicitada.

O módulo "Experimenter" também faz a comunicação com o módulo “MininetWi-

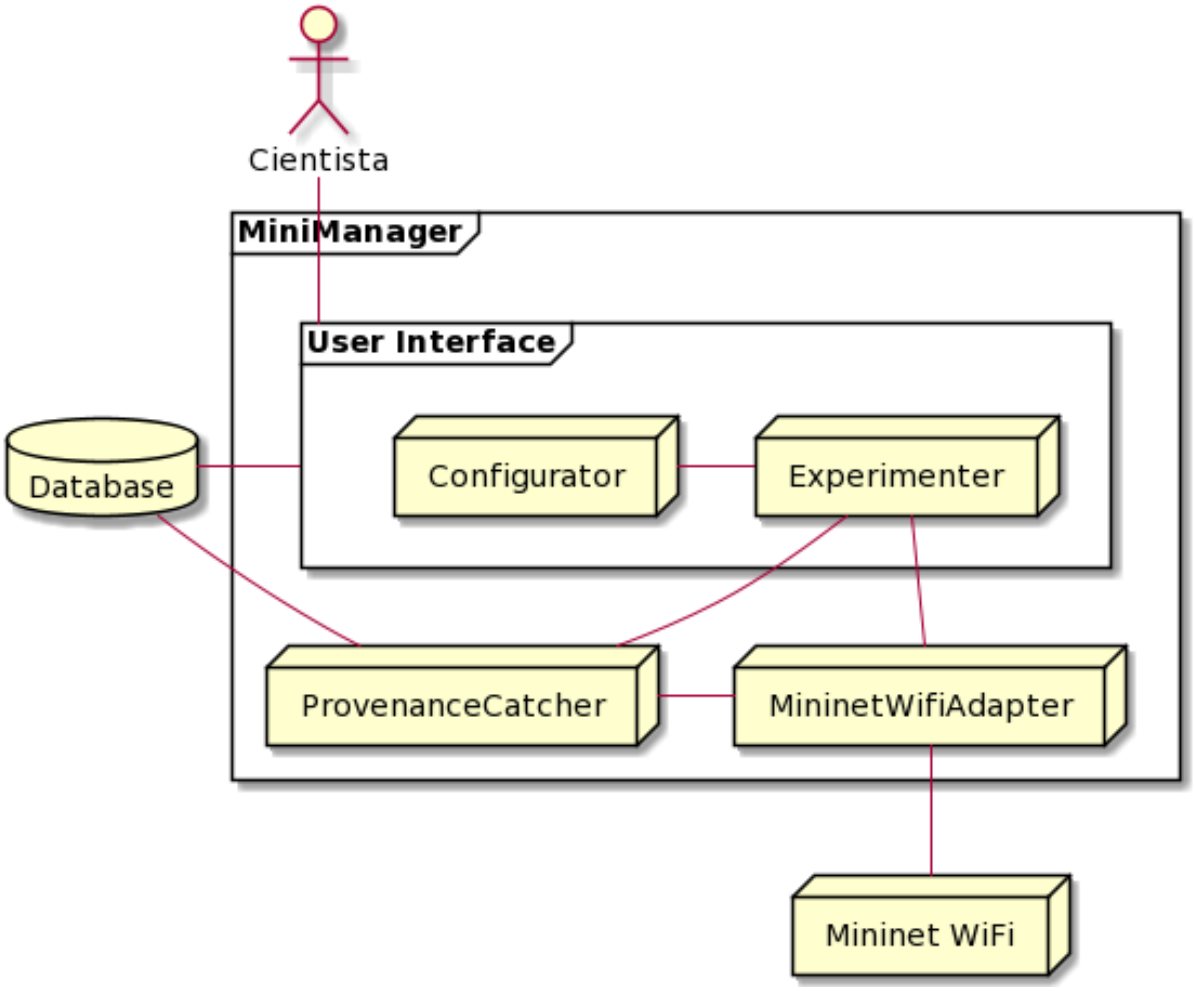

Figura 8 – Arquitetura do Projeto

fiAdapter" para a execução da simulação propriamente dita, acessa ao Banco de Dados para o registro de novas rodada e realiza acionamentos à alguns métodos do módulo "ProvenanceCatcher" para anunciar a execução de rodadas, e também para a recuperação dos resultados encontrados em rodadas anteriores. Então suas funcionalidades serão: recebimento de requisições do usuário para novas simulações, gerenciamento da fila de experimentos que serão executados, solicitação de início de um experimento ao "MininetWifiAdapter", envio de atualizações para o usuário, exibição de rodadas já concluídas, exportação dos resultados e comparação de resultados de rodadas diferentes (podendo ser de versões diferentes).

Já o "ProvenanceCatcher" é o responsável por coletar os dados que estão sendo gerados das simulações e salvá-los de forma organizada. Além disso, esse módulo disponibiliza uma interface (métodos) para a recuperação de dados de experimentos já executados em diferentes formatos, de forma que atenda os diferentes casos de uso especificados.

Por fim, o "MininetWifiAdapter" é o responsável pela comunicação direta com o Mininet-WiFi. Esse módulo funciona como uma camada, dentro do MiniManager,

entre o simulador e os outros módulos da aplicação. Esse módulo possui como entrada as configurações do experimento e possui a função de, além da comunicação com o simulador, o envio dos resultados em tempo real para outros módulos do sistema. Como característica, esse módulo possui um baixo grau de acomplamento com o restante do sistema MiniManager, de forma que seja removível caso se opte pela utilização de outro emulador dr redes.

3.4 Casos de Uso

Os casos de uso podem ser divididos com a natureza de seu escopo, bem como os seus agentes. A modelagem apresenta apenas dois agentes: Pesquisador e o Mininet-WIFI. O primeiro representa o usuário do sistema, especificando as diferentes formas com que este pode interagir com o sistema. Já o segundo representa o emulador de mesmo nome, exibindo como ocorre sua integração com o sistema a ser desenvolvido.

O primeiro pacote de casos de uso trata da gestão das simulações de um usuário. Pode-se subdividir naqueles responsáveis por manter as informações dos planos de teste (Criar Plano de Teste, Listar Plano de Teste e Excluir Plano de Teste) e das versões (Criar Versão, Listar Versões, Excluir Versões, Clonar Versão, Importar Versão e Serializar Versão).

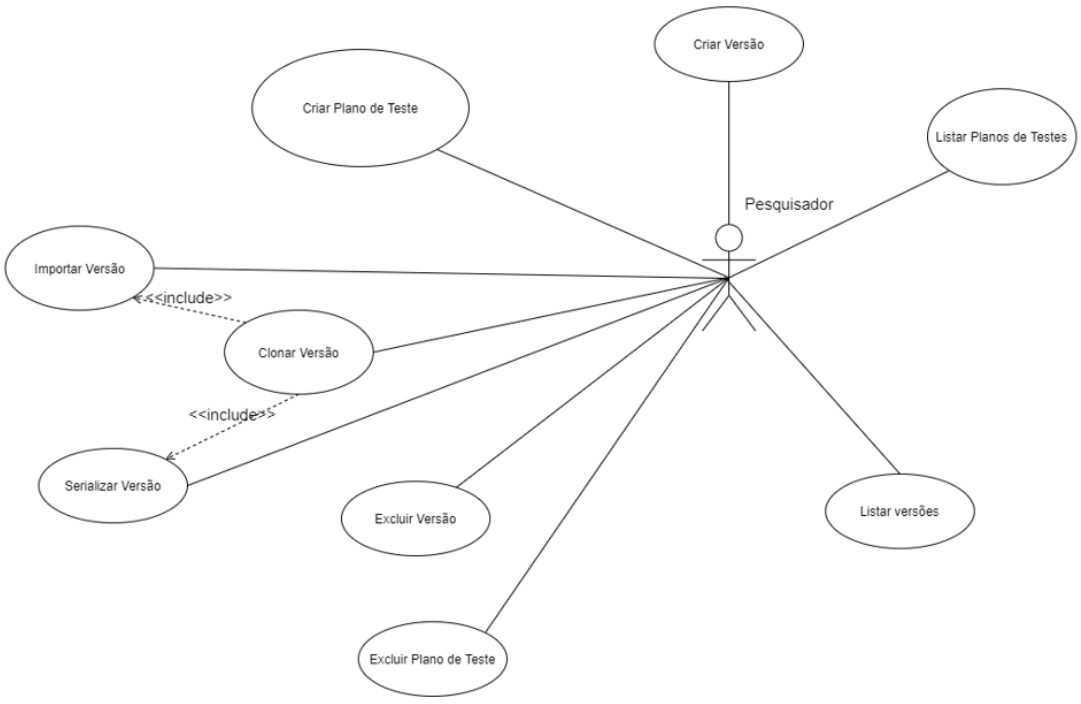

Figura 9 – 1º Pacote de Casos de Uso

O segundo pacote representa os casos de uso diretamente relacionados com experimentos de rede. Ele abrange aqueles responsáveis pela realização de uma simulação

(Executar simulação e Finalizar simulação) e pela gestão das informações das rodadas (Listar Rodadas, Exibir Rodada, Comparar Rodadas e Exportar Rodadas).

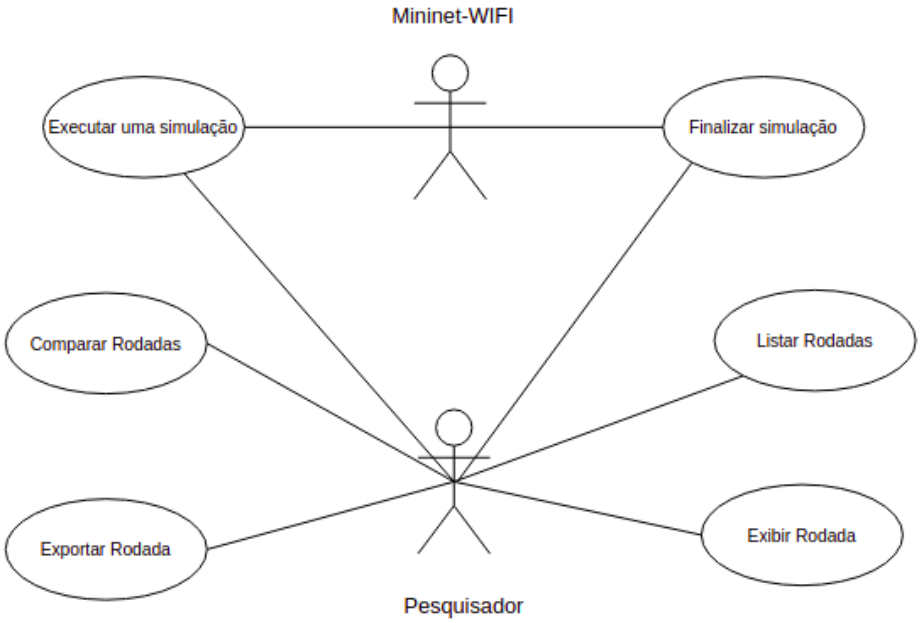

Figura 10 – 2º Pacote de Casos de Uso

A descrição detalhada dos casos de uso pode ser encontrada no **??**.

3.5 Modelagem do Banco de Dados

3.5.1 Modelo Conceitual

O diagrama do modelo conceitual do banco de dados pode ser encontrado abaixo, divido em diferentes partes de forma a facilitar a visualização. O dicionário de dados está documentado no Apêndice B.

A primeira parte do modelo conceitual aborda os conceitos de Planos de Teste, Versão e Rodada. Algumas decisões sobre relacionamentos foram explicitados na seção de Proveniência de Dados. É importante destacar que como o plano de teste define algumas configurações que serão utilizadas em todas as suas versões, existe um relacionamento com a entidade de configuração. Esse relacionamento tem o papel de definir um modelo de configuração. A versão também se relaciona com a configuração, porém o objetivo definir a configuração que será usada, de fato, para executar uma simulação. Dessa forma, a versão realiza uma cópia da configuração do plano de teste (modelo) e faz o preenchimento das informações restantes.

Observando a entidade Configuração, verifica-se que sua finalidade é representar de forma reunida todos os elementos que serão utilizados como entrada para a execução de uma simulação no Mininet-WiFi. Neste sentido, essa entidade se relaciona com a entidade de Redes, Modelo de Mobilidade, Modelo de Propagação e Medida.

Por último, detalhando a entidade Rede, encontra-se seu relacionamento a entidade Node(ou Nó), de forma a especificar todos os elementos presentes na topologia da rede definida. Como esses elementos podem variar entre si, houve a necessidade da especialização dessa entidade em Acess Point, Switch, Estação e Host. Essa especialização é mandatória, ou seja, um entidade Node não pode existir sem uma especialização atribuída, pois representaria um elemento genérico.

O mesmo pode ser aplicado para a entidade protocolo, pois não existe um significado semântico para um protocolo sem uma especialização definida. Porém nessa caso as especializações possuem algumas similaridades entre si, de forma que podem ser reunidas em alguns grupos. Dessa forma, foram criados alguns níveis de especialização, dependendo da tecnologia utilizada.

A ligação entre dois elementos é representado pela relação Link entre duas Interfaces. Apesar de não explicitado no modelo conceitual, deve-se garantir que uma interface não se relacione com ela mesma através de um link, pois não possuiria sentido semântico. Outra regra de negócio a ser aplicada é a equivalência de protocolos entre duas Interfaces conectados, isto é, devem possuir a mesma tecnologia e atributos compatíveis entre si.

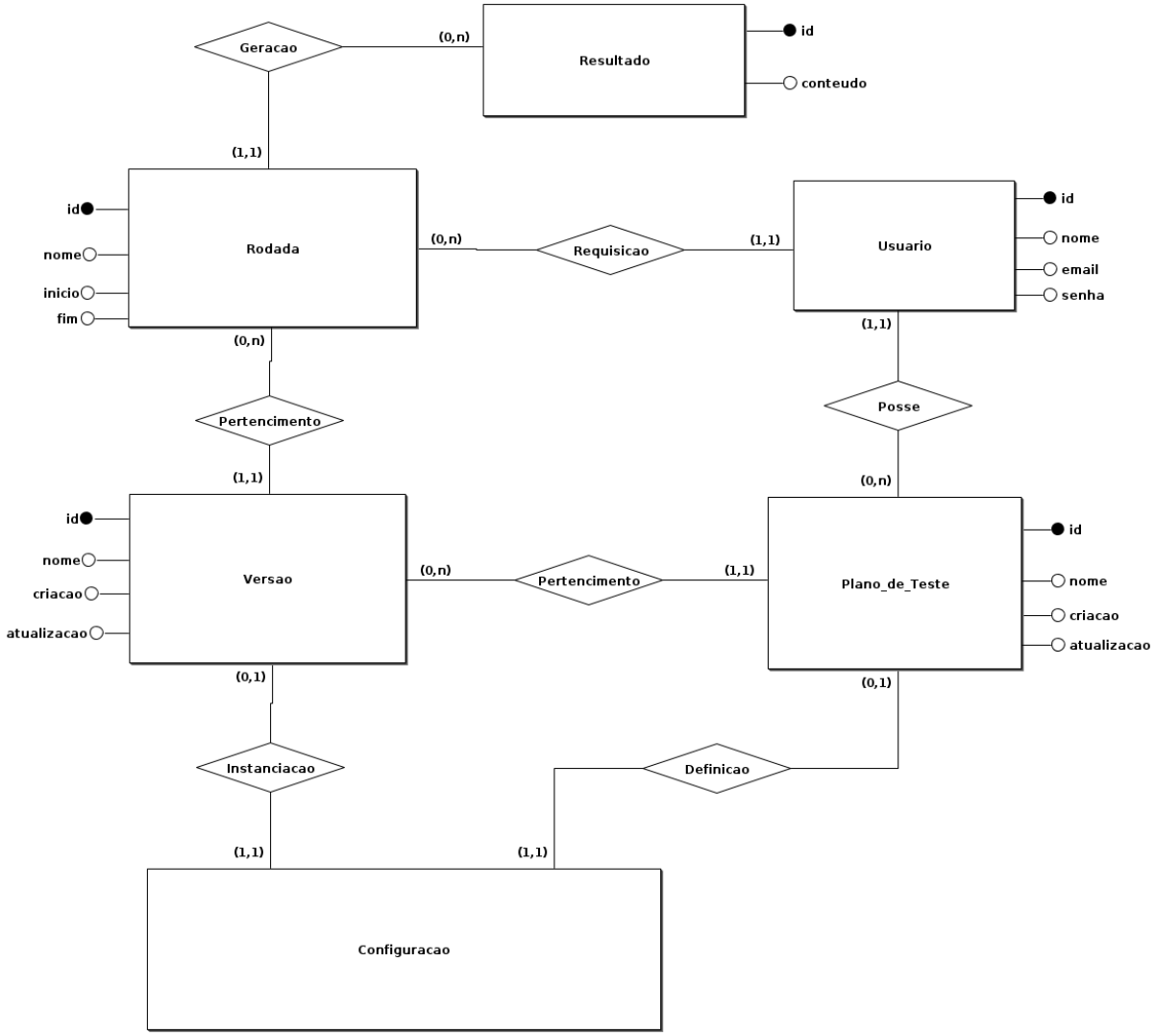

Figura 11 – 1º Parte do Modelo Conceitual - Representação ER

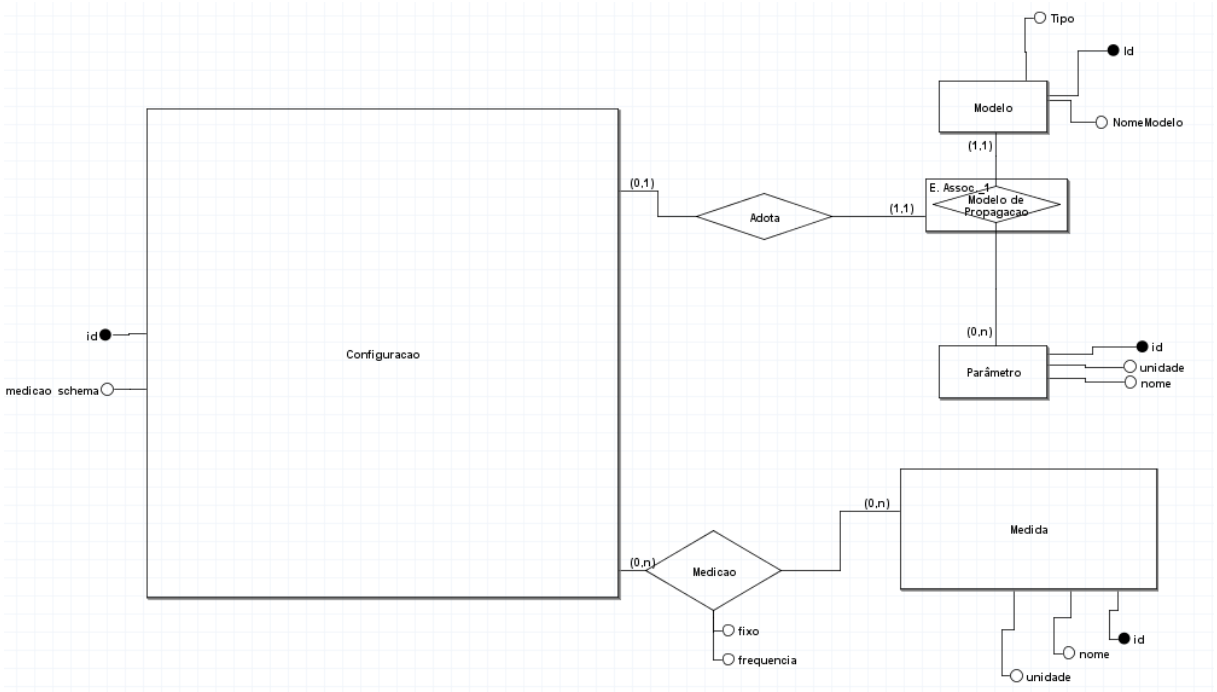

Figura 12 – 2º Parte do Modelo Conceitual - Representação ER

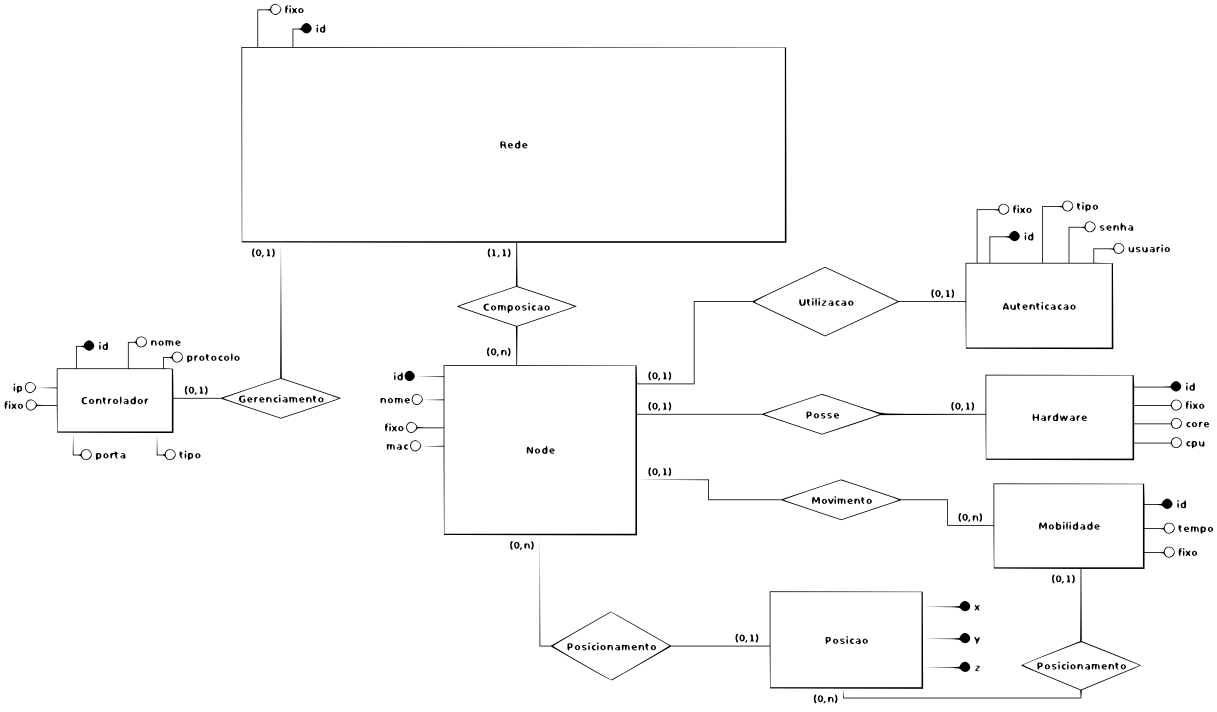

Figura 13 – 3º Parte do Modelo Conceitual - Representação ER

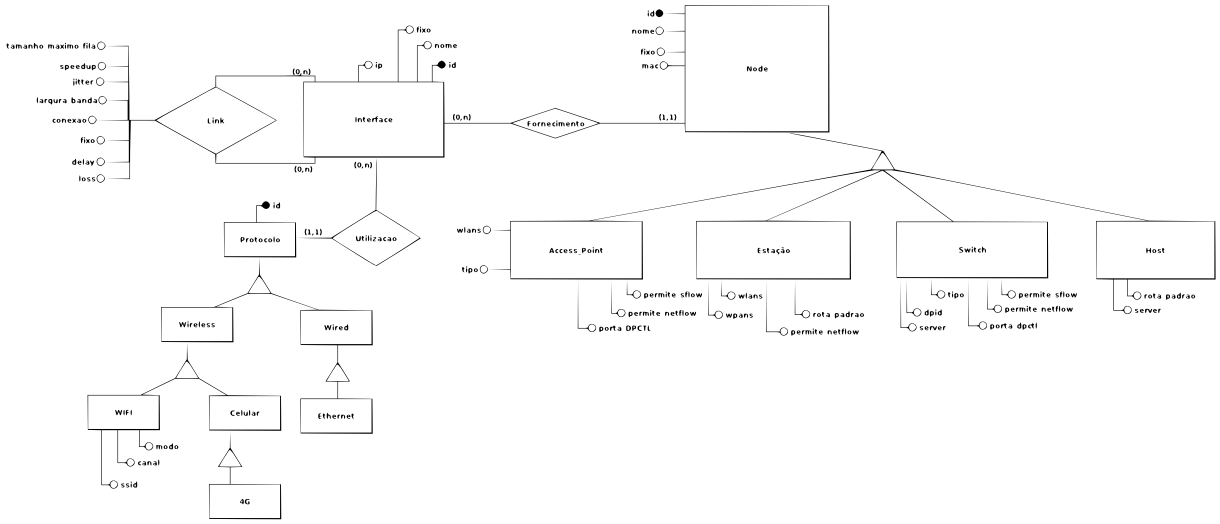

Figura 14 – 4º Parte do Modelo Conceitual - Representação ER

3.5.2 Modelo Lógico

O modelo lógico, concebido a partir do modelo conceitual apresentado anteriormente possui algumas decisões de projeto e modelagem associados. A primeira delas é quanto a especialização das entidades Protocolo e Nó. Optou-se pela utilização de uma tabela para cada entidade (seja ela a representação da especialização ou não). Essa decisão foi tomada considerando os relacionamentos que a entidade que sofreu especializações possuía inicialmente e o número de atributos das entidades especializadas. O segundo dificulta a agregação em apenas uma entidade, enquanto a primeira torna não vantajoso a utilização de entidades separadas (sem uma tabela que represente a generalização das entidades especializadas).

Outro ponto de destaque é a criação de uma tabela para o relacionamento Mobilidade-Posição, devido ao alto número de chaves primárias na tabela posição.

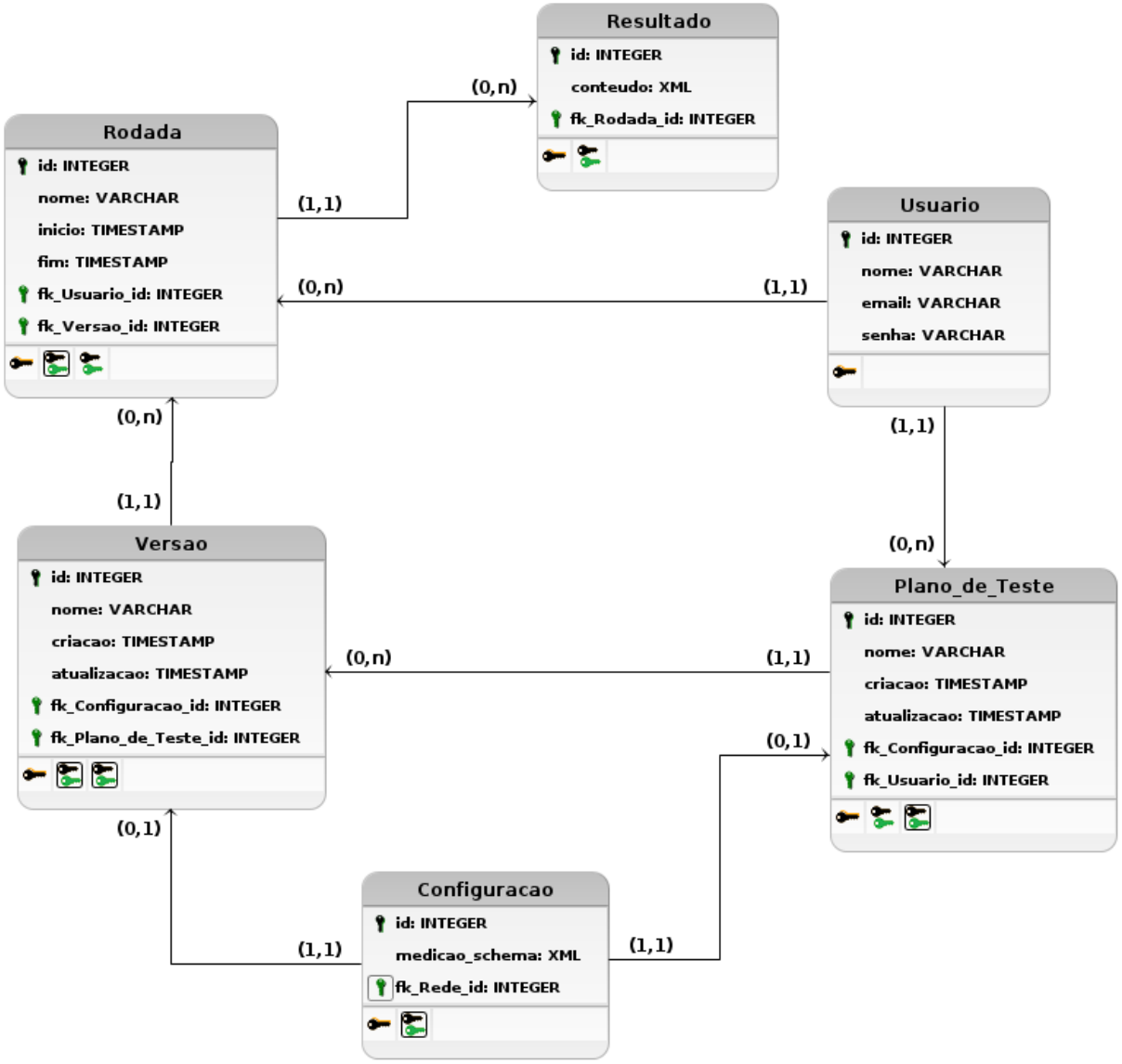

Figura 15 – 1º Parte do Modelo Lógico

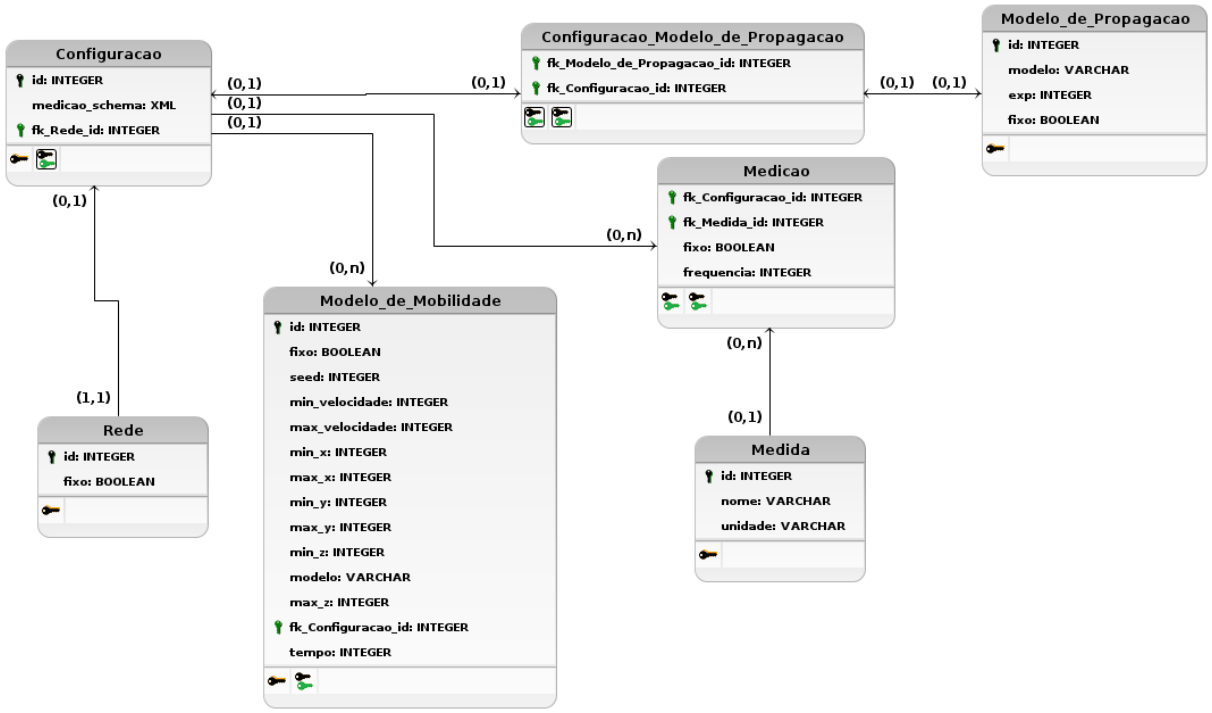

Figura 16 – 2º Parte do Modelo Lógico

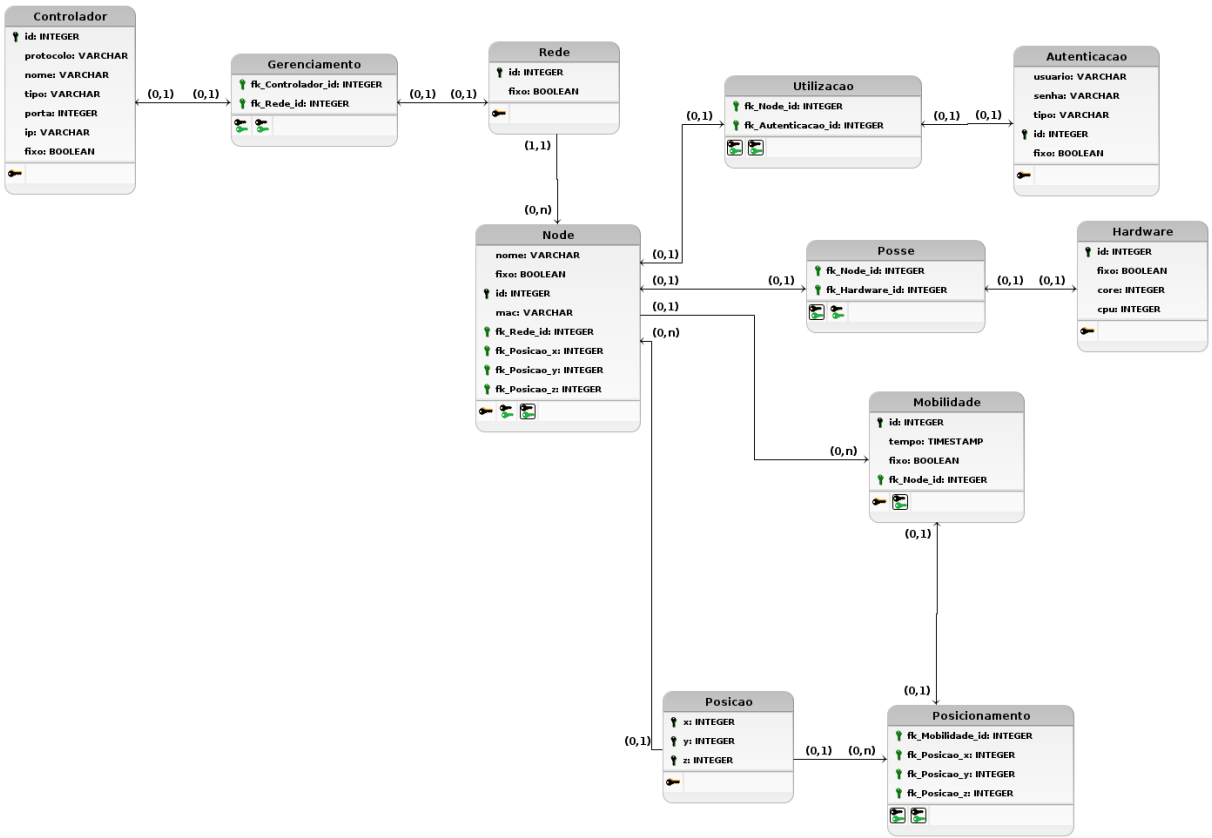

Figura 17 – 3º Parte do Modelo Lógico

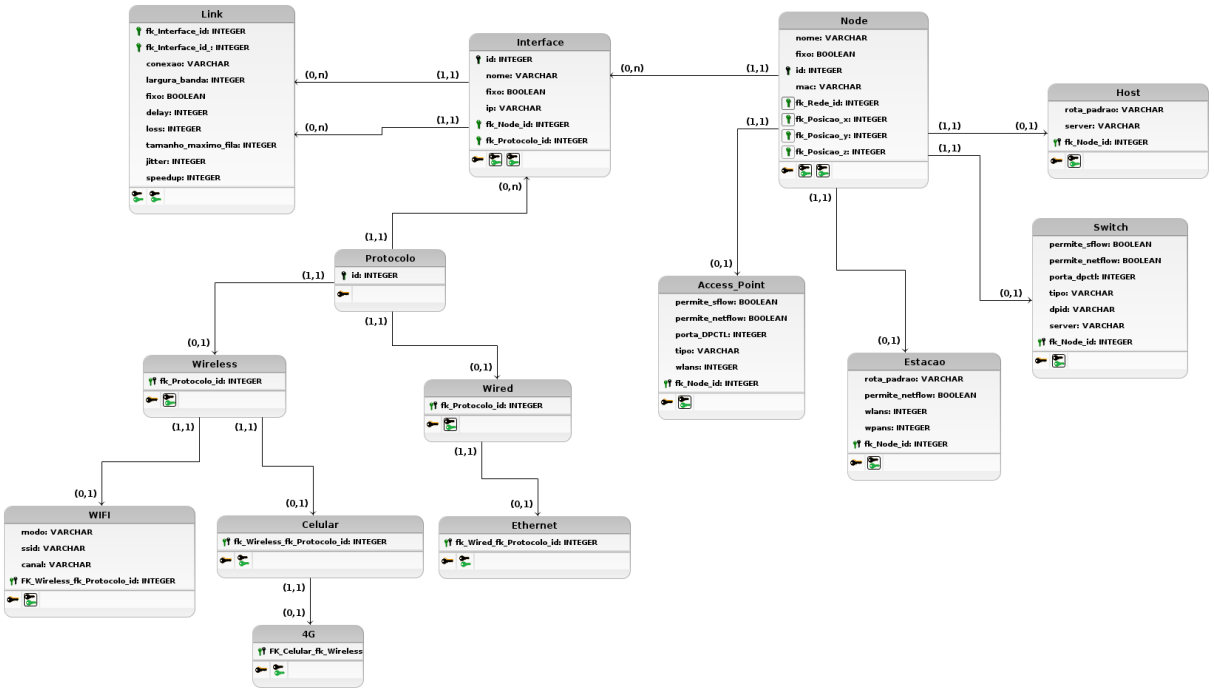

Figura 18 – 4º Parte do Modelo Lógico

Assim como especificado no dicionário de dados, o modelo lógico apresenta os atributos "medicao schema" de Configuração e "conteúdo" de Resultado como XML. A decisão por um modelo mais flexível para registrar essas informações baseia-se na heterogeneidade do formato dos resultados de uma rodada, dificultando a aplicação de um modelo relacional clássico.

Com os atributos em formato XML, pode-se definir uma estrutura comum a experimentos que realizam a medição dos mesmos parâmetros com a mesma frequência no tempo. Essa estrutura é especificada pelo atributo "medicao schema" da tabela Configuração, cujo conteúdo é um schema XML e é gerado a partir dos parâmetros selecionados para serem coletados durante a execução do experimento.

O atributo "conteúdo" da tabela Resultado faz o registro dos dados coletados durante a simulação no formato especificado pela "medicao schema". Ainda, deve se garantir em nível de aplicação que o primeiro possui o formato e segue as regras definidas pelo segundo. Abaixo podemos encontrar um exemplo desse par de atributos (de tabelas diferentes).

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="result">
  <xs:complexType>
    <xs:all>
      <xs:element name="radioFrequency">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="instant" maxOccurs="unbounded">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="station" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                      <xs:all>
                        <xs:element name="position" type="xs:string"/>
                        <xs:element name="rssi" type="xs:string"/>
                        <xs:element name="channel" type="xs:string"/>
                        <xs:element name="band" type="xs:string"/>
                        <xs:element name="ssid" type="xs:string"/>
                        <xs:element name="txpower" type="xs:string"/>
                        <xs:element name="associatedto" type="xs:string"/>
                        <xs:element name="ip" type="xs:string"/>
                      </xs:all>
                      <xs:attribute name="name" type="xs:string" use="required"/>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
                <xs:attribute name="time" type="xs:string" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="performance">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="instance" minOccurs="0" maxOccurs="unbounded">
              <xs:complexType>
                <xs:all>
                  <xs:element name="value" type="xs:string"/>
                </xs:all>
                <xs:attribute name="name" type="xs:string" use="required"/>
                <xs:attribute name="time" type="xs:string" use="required"/>
                <xs:attribute name="source" type="xs:string" use="required"/>
                <xs:attribute name="destination" type="xs:string" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:all>
    <xs:attribute name="roundID" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Figura 19 – Exemplo de Schema XML, Valor Assumido Pelo Parâmetro Medicao Schema da Tabela Configuracao

```xml
<?xml version="1.0" encoding="utf-8" ?>
<result roundID="148">
  <radioFrequency>
    <instant time="3">
      <station name="sta1-wlan0">
        <rssi>-78.0</rssi>
        <channel>36</channel>
        <band>20</band>
        <ssid>new-ssid</ssid>
        <txpower>14</txpower>
        <ip>10.0.0.1</ip>
        <position>[16.34, 55.32, 0.0]</position>
        <associatedto>ap1-wlan1</associatedto>
      </station>
      <station name="sta2-wlan0">
        <rssi>-59.0</rssi>
        <channel>36</channel>
        <band>20</band>
        <ssid>new-ssid</ssid>
        <txpower>14</txpower>
        <ip>10.0.0.2</ip>
        <position>[23.29, 27.03, 0.0]</position>
        <associatedto>ap1-wlan1</associatedto>
      </station>
    </instant>
    <instant time="4">
      <station name="sta1-wlan0">
        <rssi>-79.0</rssi>
        <channel>36</channel>
        <band>20</band>
        <ssid>new-ssid</ssid>
        <txpower>14</txpower>
        <ip>10.0.0.1</ip>
        <position>[17.75, 56.51, 0.0]</position>
        <associatedto>ap1-wlan1</associatedto>
      </station>
      <station name="sta2-wlan0">
        <rssi>-58.0</rssi>
        <channel>36</channel>
        <band>20</band>
        <ssid>new-ssid</ssid>
        <txpower>14</txpower>
        <ip>10.0.0.2</ip>
        <position>[22.31, 26.45, 0.0]</position>
        <associatedto>ap1-wlan1</associatedto>
      </station>
    </instant>
  </radioFrequency>
  <performance>
    <instance time="4" source="sta1" destination="sta2" name="ping">
      <value> 10 packets transmitted, 10 received, 0% packet loss, time 9050ms
      rtt min/avg/max/mdev = 1.056/312.685/2049.986/654.469 ms, pipe 3
      </value>
    </instance>
  </performance>
</result>
```

Figura 20 – Exemplo de XML, Valor Assumido Pelo Parâmetro Conteudo da Tabela Resultado

3.5.3 Modelo Físico

O SGBD escolhido para o projeto foi o PostgreSQL, devido tanto a sua natureza relacional quanto a sua flexibilidade e suporte a atributos mais complexos, como textos em formato JSON e XML. Além disso, as XML functions, existentes nesse SGBD, é um outro fator que motiva essa decisão, pois, junto a gravação dos resultados em formatos XML definidos por um schema, habilita a expansão de funcionalidades para suportar consultas mais complexas em um conjunto de rodadas.

O código de criação das tabelas no banco de dados pode ser encontrado no **??**.

4 DESENVOLVIMENTO

4.1 Estrutura

Dentre as possíveis alternativas de implementação do projeto, foi escolhido o framework de desenvolvimento Django.

A escolha dessa tecnologia teve como influência as características descritas em sua documentação (9). Ela foi desenhada para tornar as tarefas comuns do desenvolvimento Web rápidas, fáceis e com um design limpo e pragmático. O framework cuida de grande parte, que costuma ser desgastante, do desenvolvimento da Web, para que o desenvolvedor possa se concentrar na escrita do aplicativo em si.

O Django utiliza uma arquitetura nomeada *Model View Template*, que é bem semelhante da já conhecida MVC. O mesmo consiste em um *objetic-relational mapper* (ORM) que faz a mediação entre os modelos de dados (definidos como classes de Python) e um banco de dados relacional ("*Model*"), um sistema para processamento de solicitações HTTP com um sistema de modelos da web ("*View*") e um despachante de URL baseado em expressão regular ("*Controller*").

Um *Model* é a fonte única e definitiva de informações sobre seus dados. Ele contém os campos e comportamentos essenciais dos dados que você está armazenando. É a camada relacionada ao banco de dados. Já uma *View* é uma função Python que recebe uma solicitação da Web e retorna uma resposta da Web. Essa resposta pode ser o conteúdo HTML de uma página da Web, ou um redirecionamento, um erro 404, um documento XML, uma imagem, entre outros. Por fim, um *Template* contém as partes estáticas da saída HTML desejada, bem como alguma sintaxe especial que descreve como o conteúdo dinâmico será inserido. No template que a renderização dos dados acontece, vindo das views. É como o Django mostra as solicitações para os usuários.

Toda a modelagem física do banco de dados é feita pelo Django através das *migrations*, que a partir dos Models, cria de forma automática os comandos SQL para criação das tabelas e afins.

4.2 Módulo Experimenter

O "Experimenter" caracteriza-se por ser um elemento central do sistema, pois possui comunicação com todos os outros módulos, além da interação com o usuário e acesso ao banco de dados. Sua função é fazer a gestão dos experimentos que foram, estão e serão realizados, podendo ser destrinchados nas funcionalidades mencionadas no tópico

de arquitetura. Nesta seção, cada uma dessas funcionalidades será detalhada.

4.2.1 Execução de Rodadas

Inicialmente, será abordada o recebimento de solicitações de rodadas pelo usuário, a eventual execução e exibição dos resultados. Para isso, deve-se relembrar a falta de suporte do Mininet à execução de mais de um experimento em paralelo. Dessa forma, surgiu-se a necessidade da implementação de uma fila de experimentos, para que mantenha-se o isolamento entre eles e seja possível a execução contínua sem erros de concorrência de recursos. Então, temos a seguinte sequência:

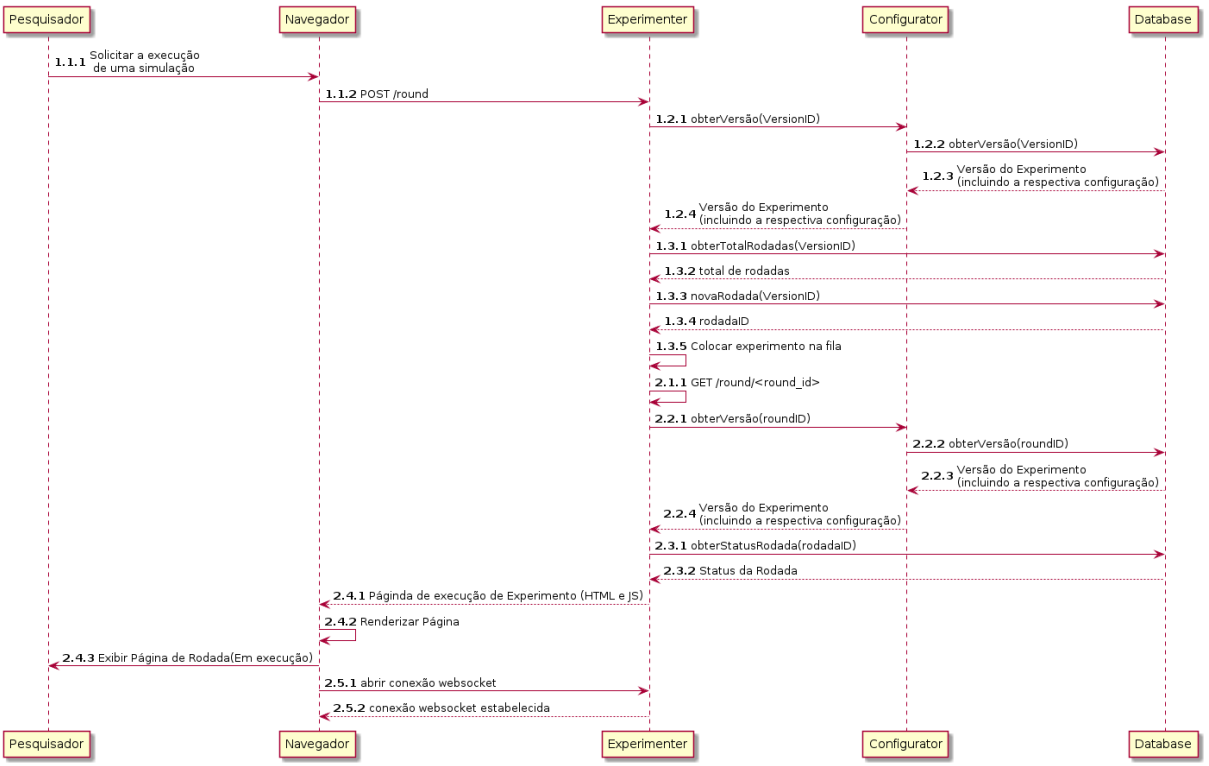

Figura 21 – Diagrama de Sequência para a Execução de Rodadas 1

A sequência inicia-se com o usuário solicitando a execução de uma simulação através de um botão fornecido pela interface do módulo Experimenter. Logo após isso é feito uma requisição HTTP com método POST e path "/round" para o sistema. Ao processar essa requisição, o módulo solicita ao "Configurator" as informações sobre a versão do experimento a ser realizado.

Munido dessa informação, o "Experimenter" realiza uma consulta do banco de dados sobre a quantidade de rodadas executadas naquela versão. Esse resultado será utilizado para definir o nome da rodada, por exemplo, se houver cinco rodadas e o nome da rodada for "Teste" o nome será composto pelo nome da versão, adicionando " - rodada 6", resultando em "Teste - rodada 6".

Definido o nome da rodada, é feito uma escrita no banco da nova rodada criada e o experimento é colocado na fila de experimentos, onde será processado quando chegar no topo. O usuário então é redirecionado para a página de rodada.

Para que seja exibido as informações corretamente, é feito uma verificação das configurações e do status da rodada. Além disso, uma conexão web socket é estabelecida entre o navegador e o servidor, para que o usuário seja informado em tempo real dos resultados do experimento.

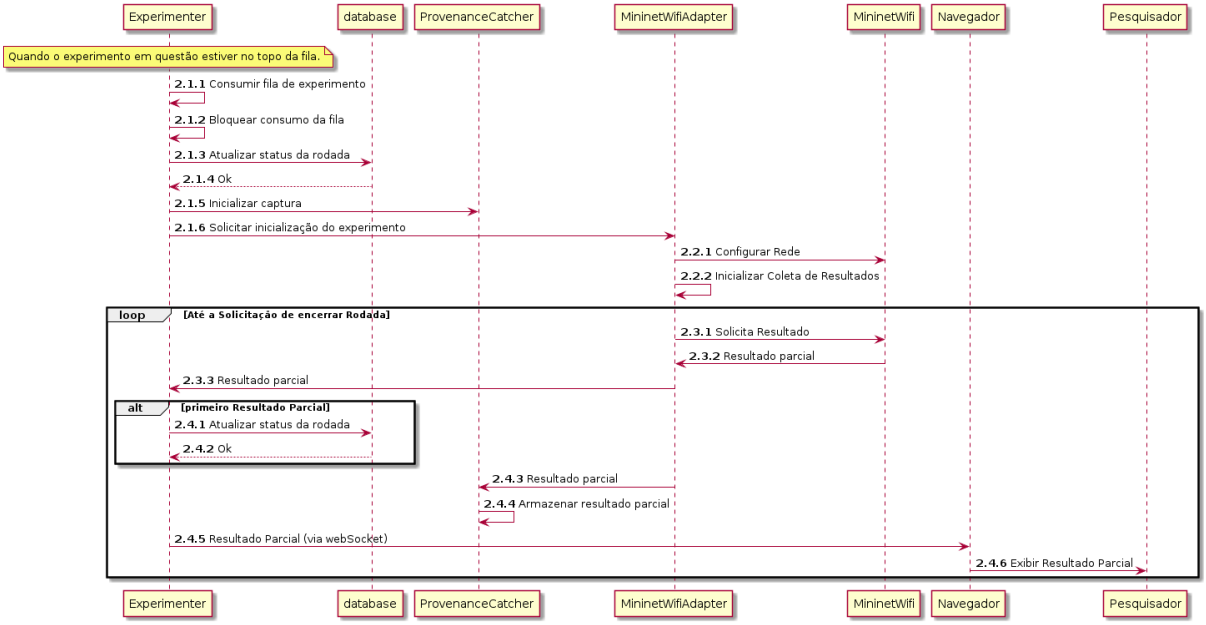

Figura 22 – Diagrama de Sequência para a Execução de Rodadas 2

Quando chegar o momento da rodada ser processada e o experimento executado, os seguintes passos são feitos: bloqueio da fila (para que nenhum outro experimento seja processado simultaneamente), atualização do status da rodada (para informar ao usuário que a execução da rodada foi iniciada) e a inicialização do experimento em si.

Esse último passo é feito através do módulo "MininetWifiAdapter" e será mais detalhadamente abordado em um momento posterior. Conforme o adaptador coleta os resultados da simulação, isso é passado tanto para o "Experimenter" quando para o "ProvenanceCatcher". O primeiro envia os resultados ao navegador através da conexão web socket aberta anteriormente. Já o segundo armazena os resultados parciais para o posterior registro. A exceção à esse comportamento ocorre no momento do primeiro resultado parcial, quando é feita uma atualização do status da rodada, indicando que ela está sendo executada.

Este procedimento é executado em loop até a finalização da rodada que pode ocorrer de duas maneiras. A primeira delas é através de uma solicitação do usuário, uma requisição HTTP com o método POST para o path "/finish-round". A segunda possui

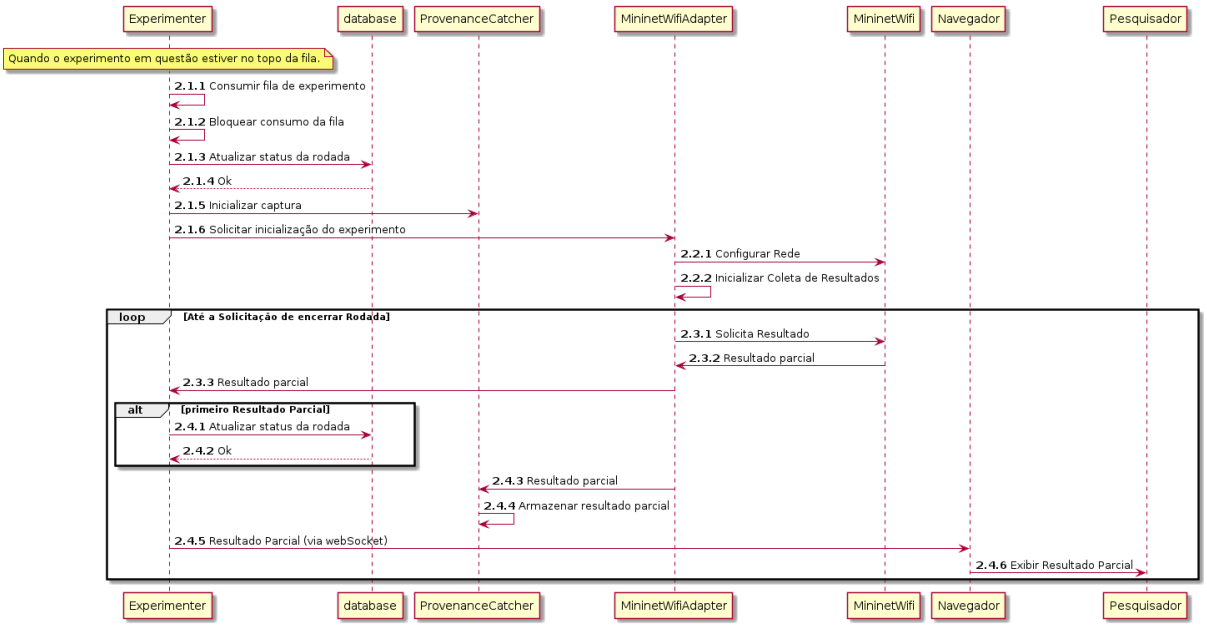

Figura 23 – Diagrama de Sequência para a Execução de Rodadas 3

como gatilho a própria aplicação, quando a simulação atinge o tempo limite configurado.

Neste momento, o experimento é finalizado pelo “MininetWifiAdapter” e esse evento é disparado para o “Experimenter” e “ProvenanceCatcher”. O segundo salva (no banco de dados) os resultados que vinham sendo armazenados e o primeiro informa envia uma atualização ao navegador para que corrija as informações exibidas e encerre a conexão web socket.

Caso a finalização seja oriunda da requisição feita pelo usuário, haverá um redirecionamento para a tela de rodadas que recuperará, novamente, as informações (configuração e status) e exibirá para o Cientista. Neste momento, como o status aponta para a rodada finalizada, então não ocorre a abertura de conexão web socket.

A seguir, observa-se o diagrama de classes responsável por implementar a sequência exibida. Para a implementação da classe que representa a fila de Experimentos, foi utilizado o padrão criacional Singleton, de forma que não seja possível a instanciação de duas filas em uma execução do sistema e, dessa forma, garantindo que cada experimento será realizado isoladamente. Essa classe possui diversas dependências e uma agreação à classe MininetWifiExp do módulo adaptador, que será detalhada posteriormente.

Verifica-se também que as classes FinishRoundView e RoundView são as responsáveis por implementar a interface com o usuário e Round realiza a comunicação com o banco de dados para a persistência. Para fins de simplificação na representação, foi suprimido as informações sobre quais classes dependem de Round.

O resultado parcial é enviado pelo "MininetWifiAdapter" aos dois módulos ("Experimenter" e "ProvenanceCatcher") através do padrão comportamental de projeto Observer

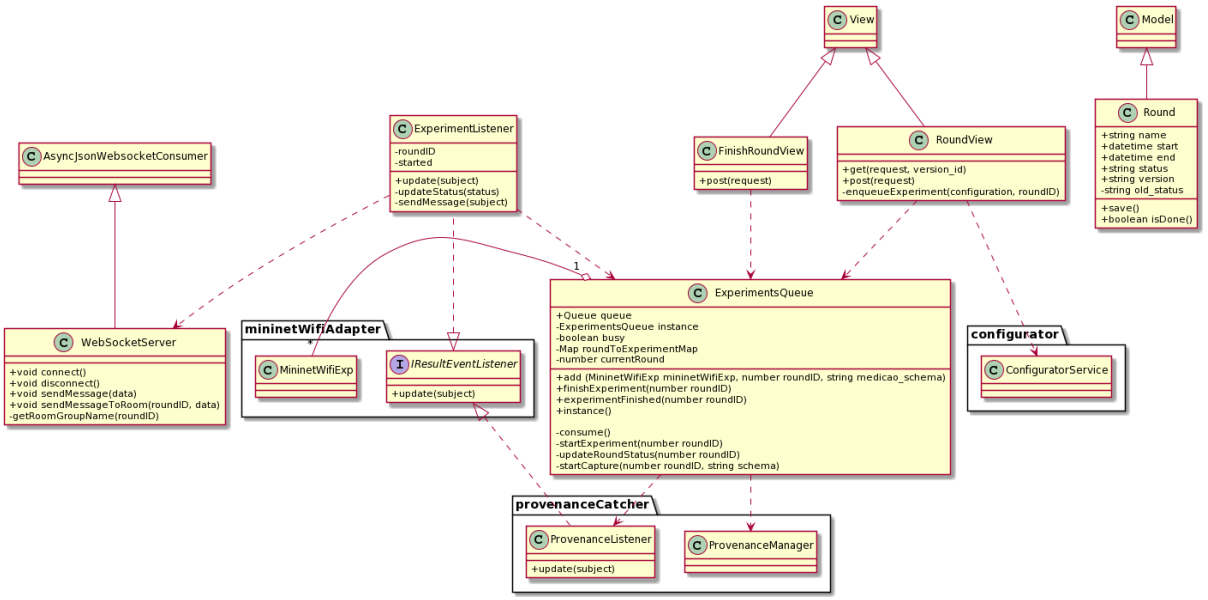

Figura 24 – 1º Parte do Diagrama de Classe para o Módulo Experimenter

e será melhor abordado posteriormente. Para isso, ambos os módulos implementam a suas próprias classes listeners que receberão essas atualizações. Além disso, como esse módulo é o responsável pela instanciação de uma nova simulação, o próprio faz as incrições dos subscribers (tanto do próprio módulo quando do "ProvenanceCatcher") ao Publisher do módulo adaptador. O ExperimenterListener possui como dependência a classe que implementa a comunicação web socket com o navegador do usuário, chamado de WebSocketServer.

4.2.2 Exibição e Exportação dos Resultados

Além de gerir a fila e a execução dos experimentos, outra funcionalidade do módulo é a exibição/exportação dos resultados para o cientista. Nesse tópico podemos dividir em três tipos: exibição em tempo real, exibição de rodada finalizada e exportação de resultado.

A primeira difere das outras pois exige a abertura da conexão *web socket* para o recebimento de informações em tempo real, como exibido no diagrama de sequência. Essa modalidade ocorre quando é exibido uma rodada que ainda está na fila ou que está sendo executada.

Os resultados exibidos podem ser divididos em três principais espaços. O primeiro se trata das medidas de Rádio Frequência realizada durante o experimento. O segundo se trata das medidas de performance e o último espaço trata a exibição das posições dos nós de rede configurado. As medições que serão exibidas serão aquelas pré configuradas para configuradas para o experimento e o catálogo com quais estão disponíveis, bem como os passos para adicionar uma nova, serão mencionado na seção sobre o Módulo Configurator.

A exibição de uma rodada finalizada possui como principal característica a não

exibição das posições dos nós. Essa decisão de projeto foi feita baseada na utilidade dessa informação para um experimento já executado, considerando que a visualização possui seu valor apenas quando em tempo real. Caso o usuário queira saber as posições dos nós para fins de análise posterior, é possível configurá-la como um dos parâmetros de Rádio Frequência.

Por fim, a exportação da rodada é feita disponibilizando o XML já salvo após o final da rodada para o usuário (detalhado na seção sobre o módulo "ProvenanceCatcher"). Assim como a exibição da rodada esse arquivo não possui dados das posições dos nós a menos que o usuário configure a simulação para realizar essa coleta.

A implementação dessas funcionalidades seguem o diagrama abaixo, exceto para a visualização em tempo real, contemplada com a figura anterior.

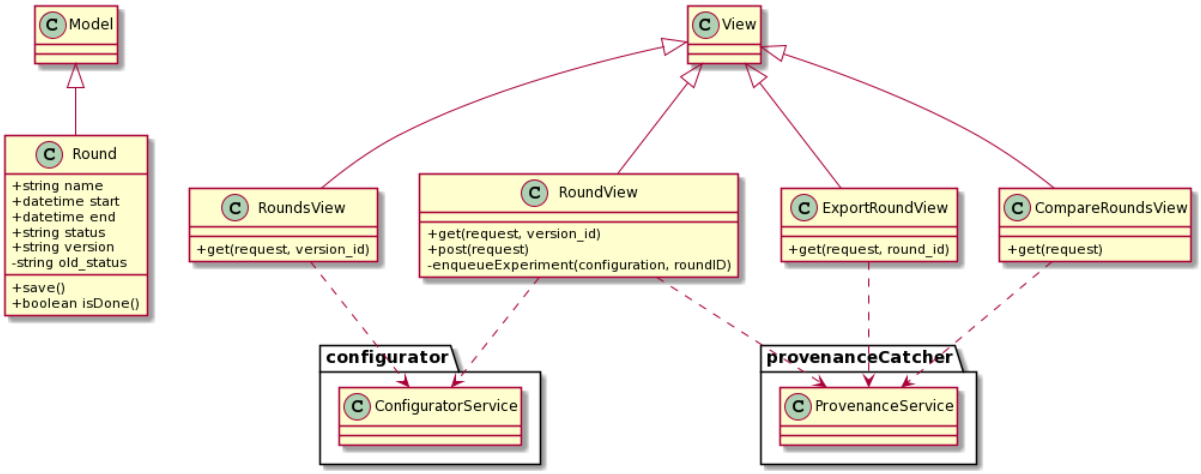

Figura 25 – 2º Parte do Diagrama de Classe para o Módulo Experimenter

Observa-se que a única classe que não depende da classe ProvenanceService, do módulo provenanceCatcher, é o RoundsView. Isso se deve pois, diferentemente das outras interfaces, essa não exibe as informações dos resultados encontrados, mas apenas recupera uma lista com as rodadas já executadas. Já a comparação de rodadas, como representado, possui uma estrutura similar as outras visualizações e será melhor detalhado a seguir.

4.2.3 Comparação de Rodadas

A comparação de Rodadas fornece ao Cientista as diferenças entre as linhas de resultado entre duas rodadas. É importante destacar que essa funcionalidade não se limita a rodadas da mesma versão e é especialmente útil na comparação entre rodadas de versões distintas com auto grau de similaridade.

A geração desse relatório é feita em três estapas. A primeira delas é a obtenção das Versões e Configurações relativas às rodadas solicitadas, a segunda é o cálculo da diferença

entre os resultados, feita pelo "ProvenanceCatcher", e a terceira é a exibição de forma intuitiva para o usuário.

4.3 Módulo Configurator

O módulo configurator é responsável por receber do usuário todas as informações referentes às configurações do experimento que o usuário deseja realizar, e salvá-las no banco de dados. Essas informações vão desde a criação de um plano de teste e versão até as configurações dos nós, modelos de mobilidade e propagação e parâmetros de medidas que o usuário deseja para o experimento.

O Mininet-WiFi é um emulador de redes sem fio bastante completo, onde consegue simular diversos cenários com panoramas bastante específicos. Essa completude torna o projeto de interface e de registro de experimentos bastante complexo. Portanto, para o nosso protótipo inicial foram feitas algumas simplificações, cujo objetivo é desenvolver um piloto para o sistema que tem projeções de ser aperfeiçoado em trabalhos futuros.

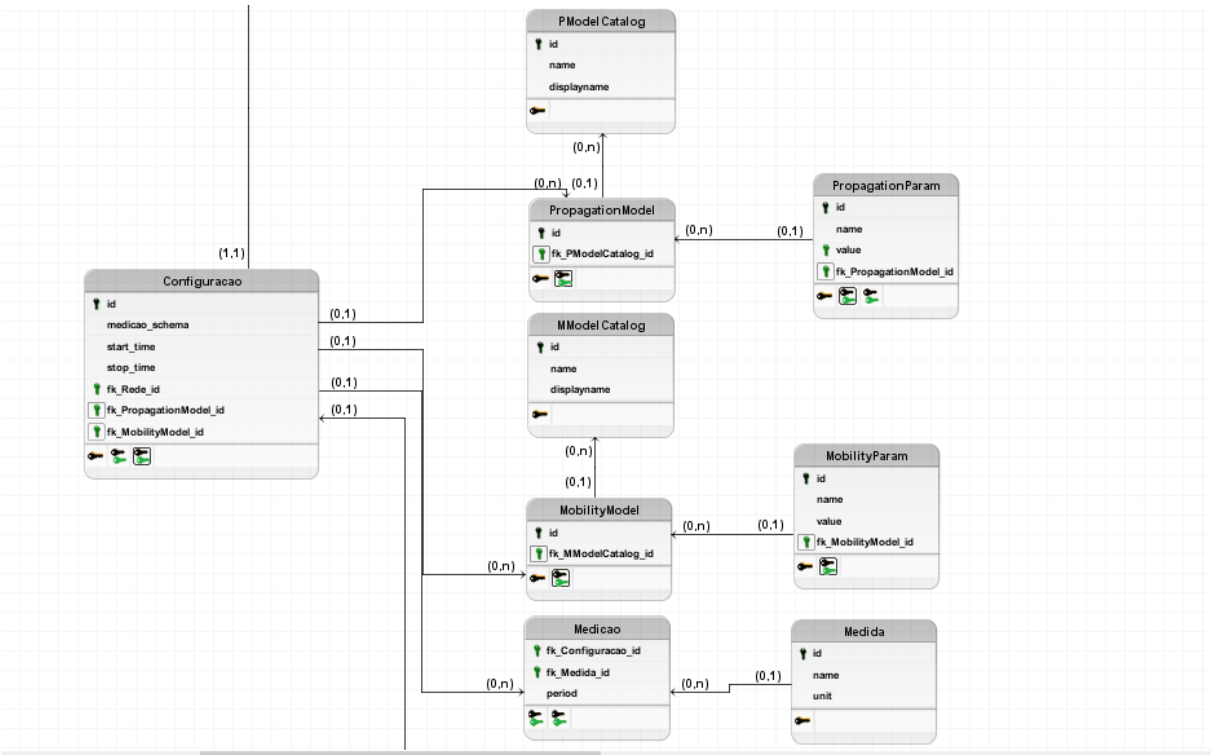

Figura 26 – Modelo Lógico dos Parâmetros de Medida Simplificado

Com base nesse modelo, ainda houve a implementação de medidas de performance, adicionando mais duas tabelas, MedicaoPerformance e MedidaPerformance. Inicialmente, a única medida suportada é o ping, porém a modelagem foi feita para que em trabalhos futuros, caso ocorram, outras medidas de performance sejam implementadas sem maiores dificuldades.

Todos os atributos do tipo "fixo" não se tornam necessários quando o plano de testes deixa de ser um *template* para um conjunto de experimentos, e sim um agrupamento de versões.

Observa-se que uma parte da configuração para realização de um experimento, consiste na escolha dos modelos de propagação e de mobilidade dos nós.

Os modelos de propagação são modelos matemáticos determinísticos ou empíricos que tentam, por aproximação, simular o comportamento das ondas eletromagnéticas durante a propagação em determinado meio. Já os modelos de mobilidade são modelos matemáticos que ditam o comportamento dos nós em relação às suas posições no espaço ao longo do tempo.

Apesar do emulador oferecer suporte a alguns modelos de propagação e de mobilidade, para o nosso projeto, foram escolhidos para serem implementados apenas dois modelos de propagação e um de mobilidade.

Modelos de Propagação:

- Log-Distance (Modelo padrão do mininet-wifi)

- Free-Space

Modelo de mobilidade:

- Random Direction

A decisão de escolha dos modelos teve como base os diversos exemplos existentes no repositório do emulador, selecionando assim os mais utilizados.

Apesar dessas escolhas , o sistema foi modelado de forma que caso ocorra futuramente o interesse de se implementar um outro modelo que o emulador tenha suporte, isso possa ser feito sem muitas dificuldades.

O banco de dados do sistema possui uma tabela com um catálogo de modelos para cada tipo, propagação ou mobilidade. Cada tabela, MModelCatalog e PModelCatalog, contém dois campos, um com o nome do modelo para o emulador, e outro com o nome do modelo que é exibido na interface do usuário. Então, para realizar o incremento de um novo modelo basta adicionar um registro nessa tabela com o novo modelo e adicionar um formulário na página estática do *Template* dos modelos, que está contido no módulo do sistema de configuração. O novo formulário deve seguir o padrão semelhante ao da Figura 27.

Outras simplificações que foram feitas para o protótipo foram em relação aos nós da rede. A modelagem inicial foi realizada de forma a se aproximar da melhor maneira possível de um cenário de rede real. Para experimentos mais simples no Mininet-WiFi, levamos em conta somente alguns atributos que são passados como parâmetros em um script de

```
<div id="RandomDirection"  class="hidde-element form-group" style="display: none;">
  <div class="form-group">
    <input type="hidden" name="RandomDirectionattribute" value="seed,min_velocidade,max_velocidade,min_x,max_x,min_y,max_y,min_z,max_z,time">
    <label for="ifprop">seed</label>
    <input type="number" name="seed" class="form-control" id="ifprop">
    <label for="ifprop">min_velocidade</label>
    <input type="number" name="min_velocidade" class="form-control" id="ifprop">
    <label for="ifprop">max_velocidade</label>
    <input type="number" name="max_velocidade" class="form-control" id="ifprop">
    <label for="ifprop">min_x</label>
    <input type="number" name="min_x" class="form-control" id="ifprop">
    <label for="ifprop">max_x</label>
    <input type="number" name="max_x" class="form-control" id="ifprop">
    <label for="ifprop">min_y</label>
    <input type="number" name="min_y" class="form-control" id="ifprop">
    <label for="ifprop">max_y</label>
    <input type="number" name="max_y" class="form-control" id="ifprop">
    <label for="ifprop">min_z</label>
    <input type="number" name="min_z" class="form-control" id="ifprop">
    <label for="ifprop">max_z</label>
    <input type="number" name="max_z" class="form-control" id="ifprop">
    <label for="ifprop">time</label>
    <input type="number" name="time" class="form-control" id="ifprop">
  </div>
</div>
```

Figura 27 – Formulário do Modelo de Mobilidade

simulação do emulador. Para experimentos mais complexos seria necessário inicialmente escalar o modelo ao feito inicialmente.

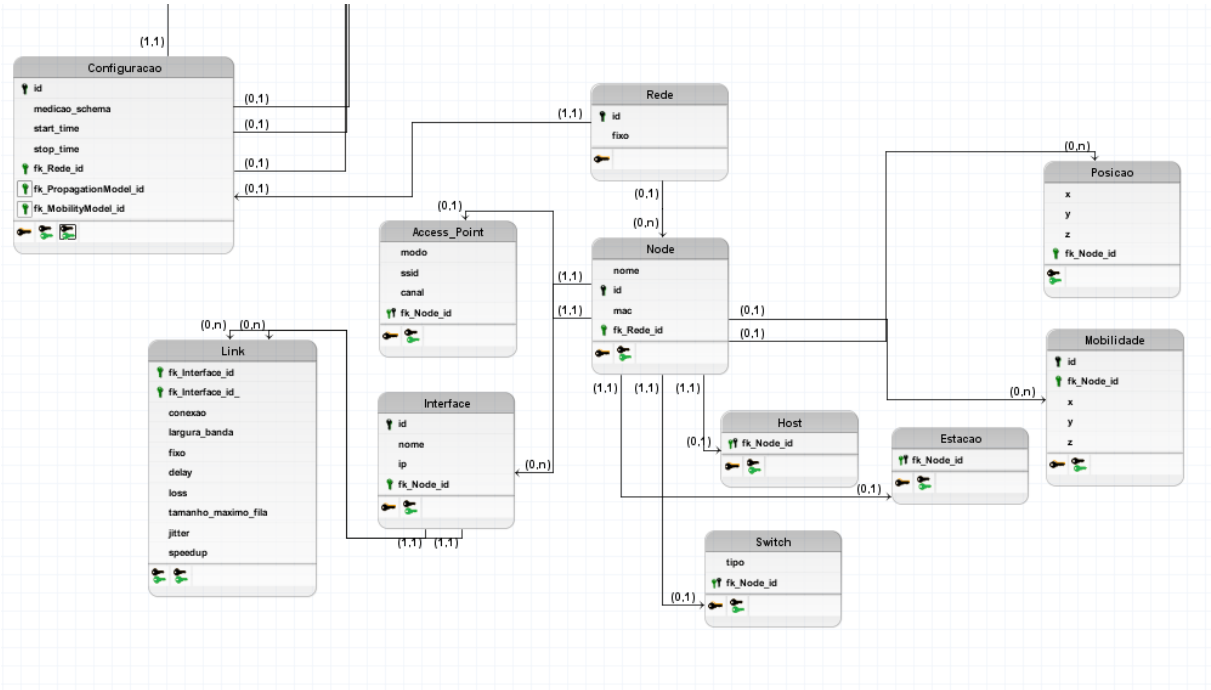

Figura 28 – Modelo Lógico dos Nós Simplificado

Como decisão para o protótipo, levamos em conta que o nó possui somente uma interface de conexão. Além disso, em relação à versão,a exportação de uma fornece ao cientista um arquivo no formato JSON cujo conteúdo apresenta a configuração do experimento que será executado naquela versão. Esse arquivo pode ser compartilhado com outros cientistas que são usuários do sistema e são usados para importar uma configuração para uma versão, conforme especificado pelo caso de uso.

A clonagem de uma versão, então, pode ser feita através da exportação de uma versão e a consecutiva importação no momento de criação da versão.

4.4 Módulo mininetWifiAdapter

Se o módulo "Experimenter" tem a finalidade de gerenciar as execuções que estão sendo feitas, o "mininetWifiAdapter" possui como responsabilidade realizar a comunicação do MininetWiFi com o restante da aplicação. Então, esse módulo é capaz de configurar redes no simulador, coletar as métricas requisitadas, enviar atualizações para os módulos interessados, interromper coleta e limpar as configurações feitas.

Como já mencionado anteriormente, o módulo recebe uma configuração de experimento como entrada para a realização do experimento. A classe principal da implementação é o MininetWifiExp, que possui uma agregação com a classe ResultNotifier.

Esta classe é um dos componentes da implementação do padrão de projeto Observer, que realizará o disparo das atualizações para as classe listeners inscritas. Esses listeners se tratam de implementações da interface IResultEventListener e podem estar inseridos em outros módulos, de acordo com seus respectivos comportamentos. As chamadas ao listeners é feitas através de aberturas de threads para cada integrante inscrito, de forma que sejam processados concorrentemente e isoladamente.

Como o MininetWiFi não possui suporte a realização de diversos experimentos de forma contínua do mesmo processo, a realização desses no sistema se dá através da criação de novos processos que executarão os programa especificado pela classe MininetScript. Além disso, a saída desses processos gerados é configurada para ser capturada pelo processo pai (MininetExp), então o script pode disparar mensagens e esta serão processadas.

Esse script é criado através da utilização da bilbioteca pexpect, que não somente os cria mas também monitora as mensagens geradas, capturando o seu conteúdo de acordo com as expressões regulares pré configuradas.

Ou seja, conforme o MininetScript imprime mensagens, estas são checadas contra expressões regulares e, caso, ocorra uma combinação, o conteúdo é redirecionado para a especialização da classe OutputHandler correspondente. Estas também possuem uma agregação com ResultNotifier para o disparo de atualizações para outras classes.

Para que o processo que executa o MininetScript tenha acesso a configuração da simulação, esta é serializada pelo MininetExp no mesmo formato feito pelo módulo configurator na exportação da versão e salva em um arquivo de formato JSON cujo nome é "config.json". Optou-se por essa solução para que seja reutilizado os artefatos já construídos. Emtão o MininetScript, como primeiro passo, realiza a leitura desse arquivo.

Além de processar a configuração de entrada do experimento, o processo (executando

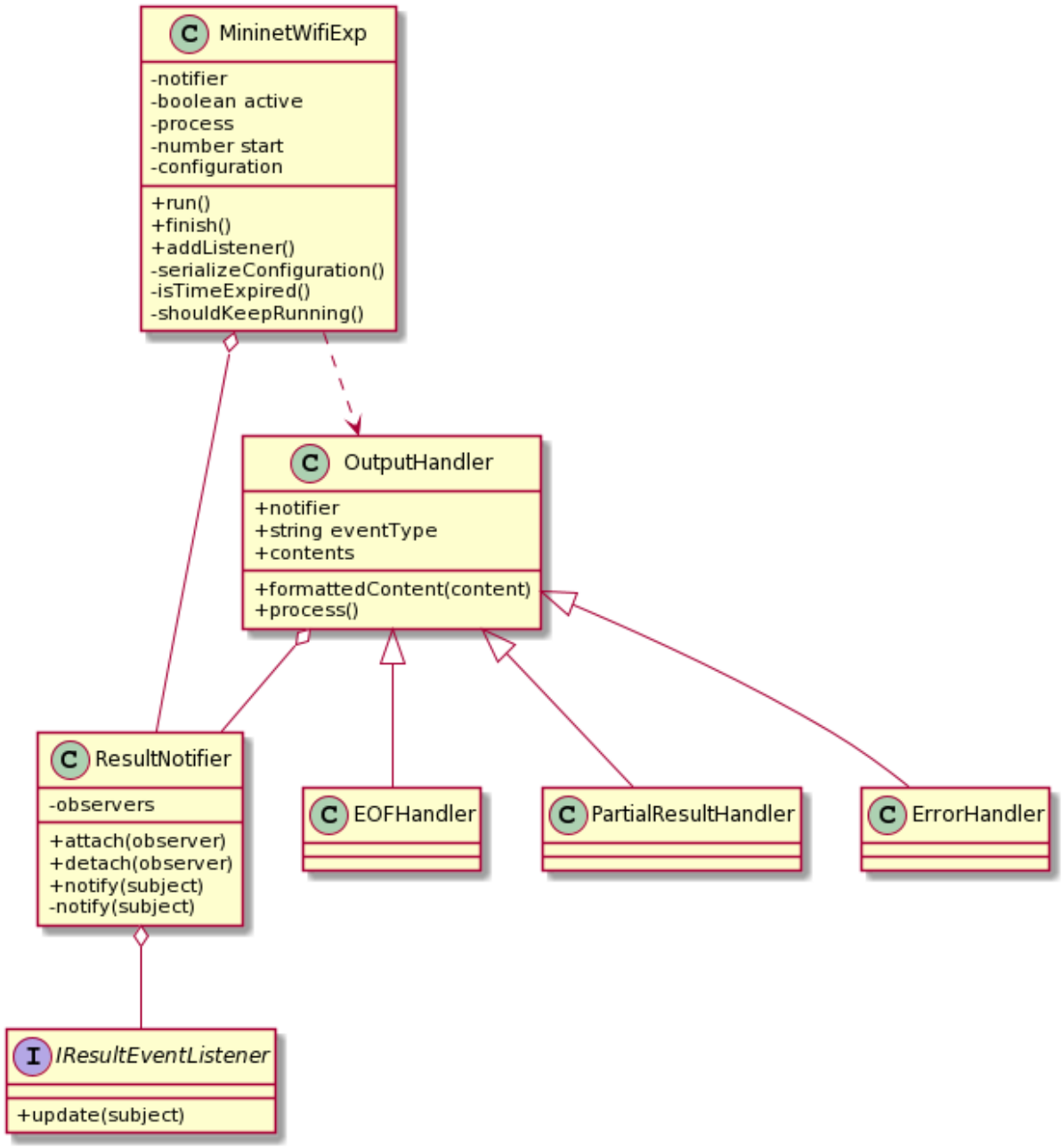

Figura 29 – 1º Parte do Diagrama de Classe para o Módulo mininetWifiAdapter

métodos da classe MininetScript e suas dependências) possui mais dois passos para a realização do experimento.

O primeiro se trata da definição e inicialização da rede, de acordo com a configuração previamente processada. Esta etapa é implementada utilizando o padrão estrutural Decorator. Onde cada decorator é responsável por estabelecer alguma configuração específica na rede que está sendo construída.

Com a rede inicializada, o próximo e último passo é a realização de medições, que também estão especificadas na configuração do experimento. Portanto o MininetScript

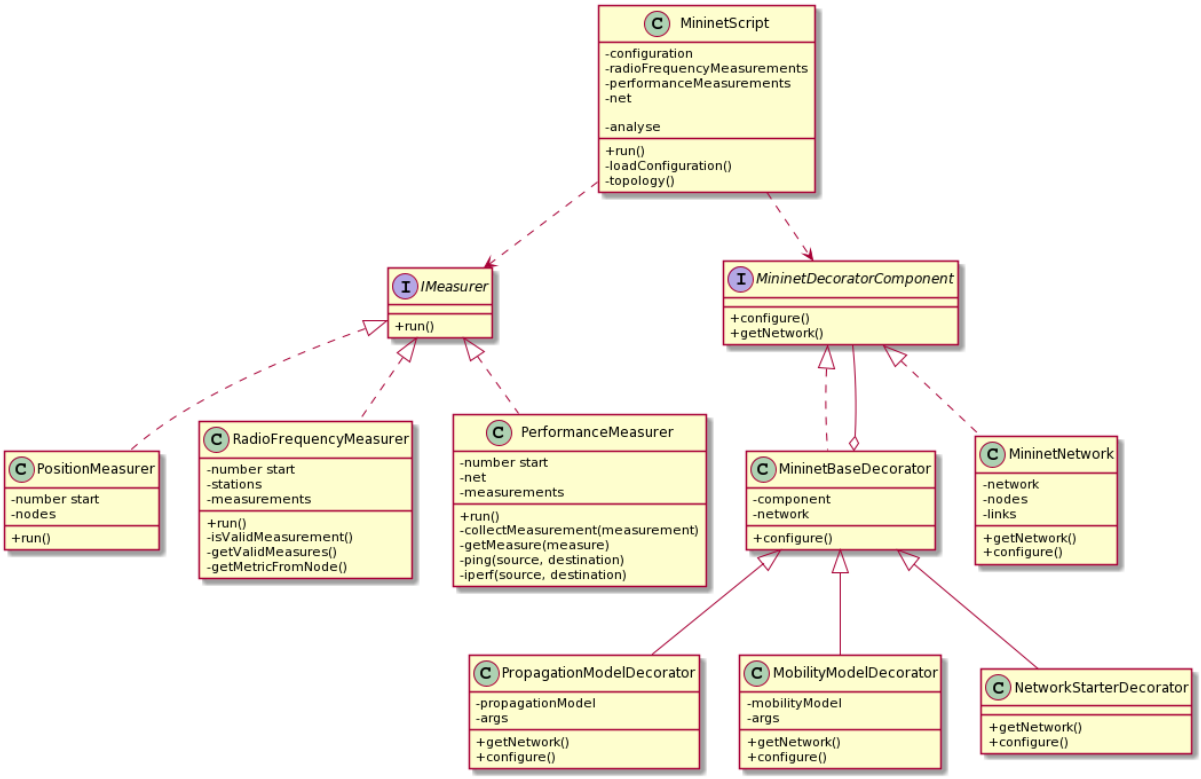

Figura 30 – 2º Parte do Diagrama de Classe para o Módulo mininetWifiAdapter

possui uma dependência às interface IMeasurer, que possui diferentes implementações de acordo com a natureza das medições. Destaca-se, aqui, que cada implementação da interface IMeasurer possuirá uma execução contínua e periódica (de acordo com o período especificado pelo usuário) e isoladas em threads separadas.

4.5 Módulo provenanceCatcher

A função principal do módulo é a coleta e registro das informações que estão sendo geradas pelo simulador de redes. A coleta é feita a partir da classe ProvenanceListener, que implementa a interface IResultEventListener do módulo mininetWifiAdapter, seguindo o padrão Observer como já foi mencionado. Essa classe possui uma composição com a classe ProvenanceManager e esta relação será detalhada a seguir.

Basicamente, o listener recebe as atualizações e realiza chamadas distintas ao ProvenanceManager para que seja feito esse registro. O primeiro tipo de chamada ocorre quando o listener recebe um resultado parcial da Rodada que está sendo executada. Quando essa informação chega ao Manager, ele checa se o resultado se trata de uma medição de Rádio Frequência ou Performance e insere em uma estrutura hierárquica de forma adequada. Essa hierarquia deve seguir o esquema XML configurado para a rodada, então no início da execução o manager recebe essa informação.

Durante o experimento, esses resultados são armazenados em memória e são persistidos apenas ao final da rodada, que é caracterizado-se por um sinal de finalização enviado ao listener. Neste momento, a estrutura é convertida para um texto no formato XML e o valor é armazenado no banco de dados.

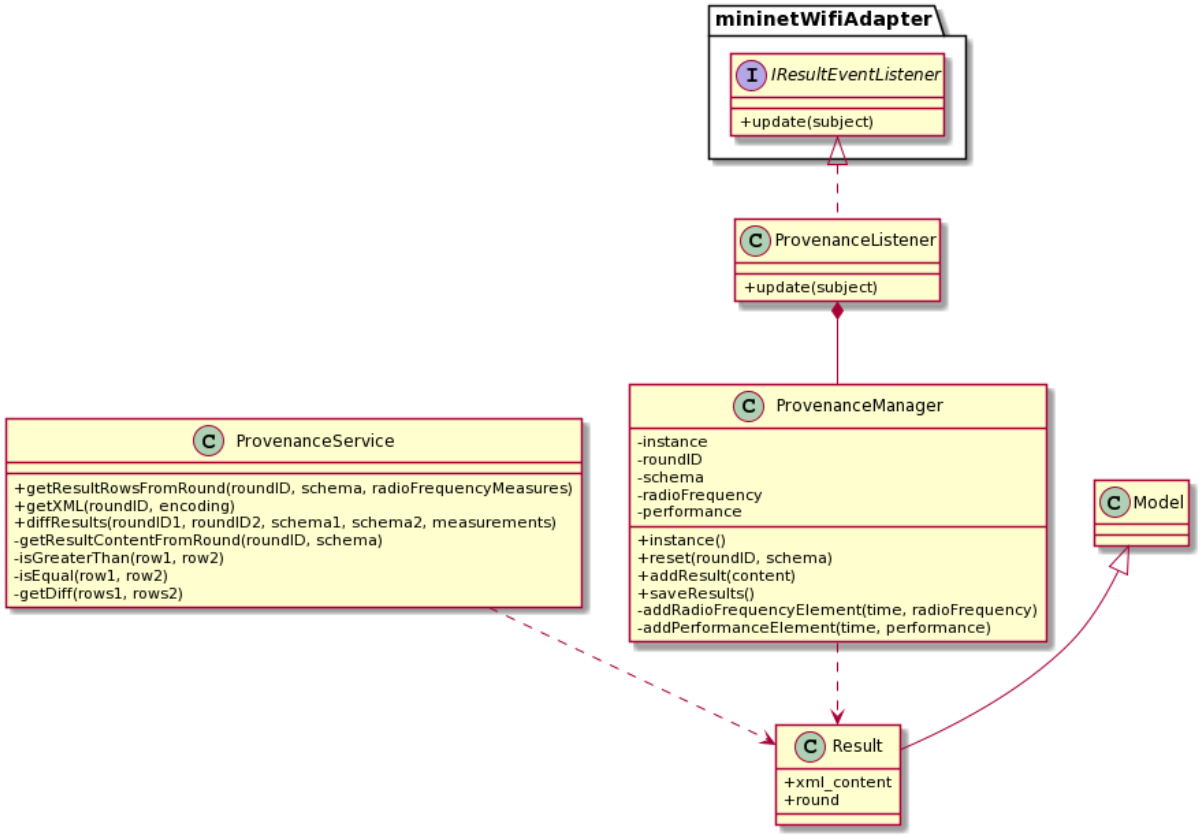

Figura 31 – Diagrama de Classe para o Módulo Provenance Catcher

Uma responsabilidade adicional desse módulo é fornecer métodos para a recuperação dos resultados de uma rodada em diferentes formatos. O primeiro e mais simples desses métodos retorna o conteúdo do arquivo XML previamente armazenado sem nenhum processamento adicional.

Já o segundo, utiliza a biblioteca xmlschema para leitura do resultado salvo e conversão para um objeto com uma estrutura hierárquica, através do esquema definido para a rodada em questão. Esse objeto, então, é processado novamente e transformado em uma estrutura de lista que permita uma fácil exibição ao usuário.

O terceiro e mais complexo utiliza-se dessa estrutura em lista de duas rodadas diferentes e realiza a comparação entre ambas, possibilitando a visualização da diferença entre os resultados encontrados em ambas.

5 CONCLUSÕES E TRABALHOS FUTUROS

Esse trabalho foi idealizado com o objetivo de criar um software integrado ao Mininet-WiFi (emulador de redes já existente) que permita aos pesquisadores, por meio de uma interface amigável, a gerência da execução das simulações de rede, disponibilizando o histórico de experimentos com seus respectivos resultados.

O objetivo inicial seria desenvolver o sistema trabalhando em cima do *handover*, que seria o procedimento empregado em redes sem fio para tratar a transição de uma unidade móvel (UM) de uma célula para outra de forma transparente ao utilizador. Porém nas reuniões com as partes envolvidas no projeto, chegou-se a conclusão que seria mais interessante para o usuário, em um primeiro momento, focar o projeto no contexto da proveniência de dados.

Assim, o trabalho desenvolvido teve início com uma abordagem teórica necessária para contextualização do projeto, seguido de uma modelagem detalhada do sistema com base nos seus requisitos. Posteriormente iniciou-se a fase de desenvolvimento, implementando um modelo protótipo funcional do sistema simplificado. Em termos gerais, o projeto realizado cumpriu com os objetivos realizando uma ambientação na área de redes sem fio e rádios cognitivos e tendo como um produto final um software integrado ao Mininet-WiFi com uma interface amigável, uma gerência da execução das simulações de rede, e que disponibiliza o histórico de experimentos com seus respectivos resultados.

Com base nos requisitos e com as simplificações feitas no desenvolvimento, foram realizadas algumas capturas de telas de navegação de interface do sistema elaborado. Tal conteúdo pode ser observado através do Apêndice A.

Além da execução e implementação do sistema, foi levantado alguns aspectos que podem sofrer melhorias em projetos futuros. Uma delas é a implementação de um módulo de usuários, definido nos requisitos iniciais do projeto. Esse componente é importante para a segmentação dos experimentos de acordo com o cientista proprietário daquelas execuções. Para isso, tanto a modelagem conceitual quanto o modelo lógico definidos dão suporte, bastando a implementação na aplicação web, tanto das interfaces quanto da integração com os outros módulos.

Outro item que foi suprimido do projeto mas pode ser incrementado é a definição de um *template* de configuração no plano de teste. Neste momento um plano de teste possui utilidade apenas para organizar os experimentos, rotulando as versões cadastradas. Com essa melhoria, que também é abrangida pela modelagem conceitual e lógica definida inicialmente, será possível um melhor reuso das configurações entre versões sem a necessidade de clonar versões. Isso seria feito através de uma atualização no módulo configurator

e poderia ser seguida de outras funcionalidades que facilitem a gestão de planos de testes e versões, como renomear e excluir.

Ainda no configurator, projetos futuros podem aumentar a abrangência de configurações de redes que podem ser criadas. Essa aumento da cobertura pode ser feito em diferentes direções. O primeiro deles é o aumento de modelos de propagação e mobilidade disponíveis, seguindo a breve explanação contida na respectiva seção do módulo. Outra forma de aumentar a flexibilidade é o suporte a novas medições, tanto de rádio frequência quanto de performance, inserindo novos valores nas respectivas tabelas de catálogo e atualizando o as classes que implementam a interface IMeasurer no módulo mininetWifiAdapter. A terceira direção, e mais complexa, refere-se a novas configurações na topologia e nos nós da rede. Para isso, será necessário tanto atualizações no módulo configurator, atualizando os formulários que coletam essa informação do usuário, quanto no mininetWifiAdapter, incrementando os decorators utilizados.

No módulo de proveniência, as sugestões tratam da coleta de novas informações sobre o funcionamento do sistema e novos formatos de disponibilização dos resultados para o usuário. Este último item requisitaria, também, atualizações no módulo Experimenter para que forneça novas interfaces, possibilitando consultas complexas nos resultados e renderizações de gráficos que facilitem a visualização dos dados encontrados.

Afim de garantir execuções de experimentos simultaneamente, sem a necessidade de espera pelos disponibilidade dos recursos do MininetWifi, o móodulo mininetWifiAdapter pode sofrer transformações mais profundas, deixando interagir com a instância local do MininetWifi. Essa comunicação seria substituída por uma interação com diversas instâncias do MininetWifi, sendo executadas em diferentes máquinas virtuais (garantindo o isolamento delas). Dessa forma, não seria mais necessário a implementação de fila de experimentos no módulo experimenter e garantia maior escalabilidade e robustez ao software.

REFERÊNCIAS

1 FONTES, R.; ROTHENBERG, C. In: Wireless Network Emulation with Mininet-WiFi. Christian Esteve Rothenberg, 2019. p. 9. Disponível em: <https://mininet-wifi.github.io/book/>. Acesso em: 19 maio 2021.

2 FONTES, R. R.; AFZAL, S.; BRITO, S. H. B.; SANTOS, M. A. S.; ROTHENBERG, C. E. Mininet-wifi: Emulating software-defined wireless networks. In: Network and Service Management (CNSM), 2015 11th International Conference on. Barcelona: [s.n.], 2015. p. 384–389.

3 FILHO, H. V. P.; GALDINO, J. F.; MOURA, D. F. C. Pesquisa e desenvolvimento de produtos de defesa: Reflexões e fatos sobre o projeto rádio definido por software do ministério da defesa à luz do modelo de inovação em tríplice hélice. Revista Militar de Ciência e Tecnologia, v. 34, 2017. Disponível em: <http://rmct.ime.eb.br/arquivos/RMCT_1_sem_2017/artigo1_2017.pdf>.

4 PéREZ, B.; RUBIO, J.; SáENZ-ADáN, C. A systematic review of provenance systems. Knowledge and Information Systems, v. 57, 2018. Disponível em: <https://doi.org/10.1007/s10115-018-1164-3>.

5 CAMILO, M. J.; MOURA, D. F. C.; SALLES, R. M. Redes de comunicações militares:desafios tecnológicos e propostas para atendimento dos requisitos operacionais do exército brasileiro. Revista Militar de Ciência e Tecnologia, v. 37, 2020. 31 set. de 2020. Disponível em: <http://rmct.ime.eb.br/arquivos/RMCT_3_tri_2020/RMCT_47218.pdf>.

6 HOSSAIN, E. Ieee802.16/wimax-based broadband wireless networks: Protocol engineering, applications, and services. In: Fifth Annual Conference on Communication Networks and Services Research (CNSR '07). [S.l.: s.n.], 2007. p. 3–4.

7 CRUZ, S. AN APPROACH TO SUPPORT THE MANAGEMENT OF DATA PROVENANCE OF SCIENTIFIC EXPERIMENTS. 214 p. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 08 2011.

8 MOREAU, L.; MISSIER, P. Prov-dm: The prov data model. W3C Recommendation, 2013. Disponível em: <https://www.w3.org/TR/2013/REC-prov-dm-20130430/>.

9 FOUNDATION, D. S. Django Documentation. 2021. 01 out. de 2021. Disponível em: <https://docs.djangoproject.com/en/3.2/>.

APÊNDICE A – DEMONSTRAÇÃO DE INTERFACE DO SISTEMA

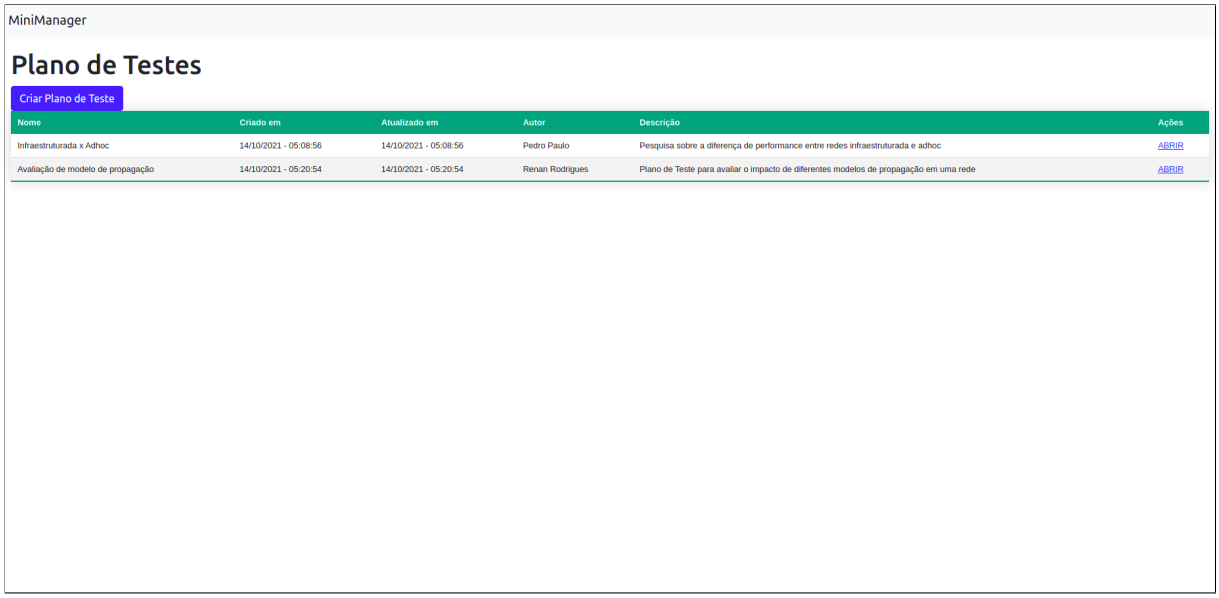

Figura 32 – Tela de Visualização dos Planos de Testes

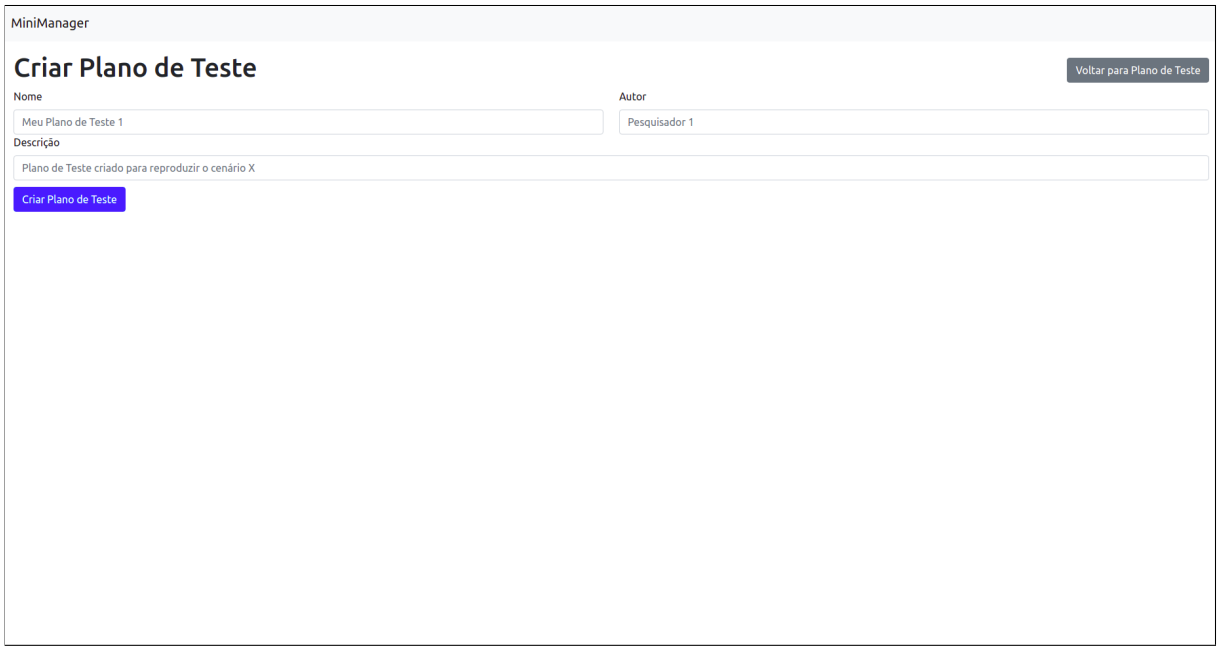

Figura 33 – Tela de Criação dos Planos de Testes

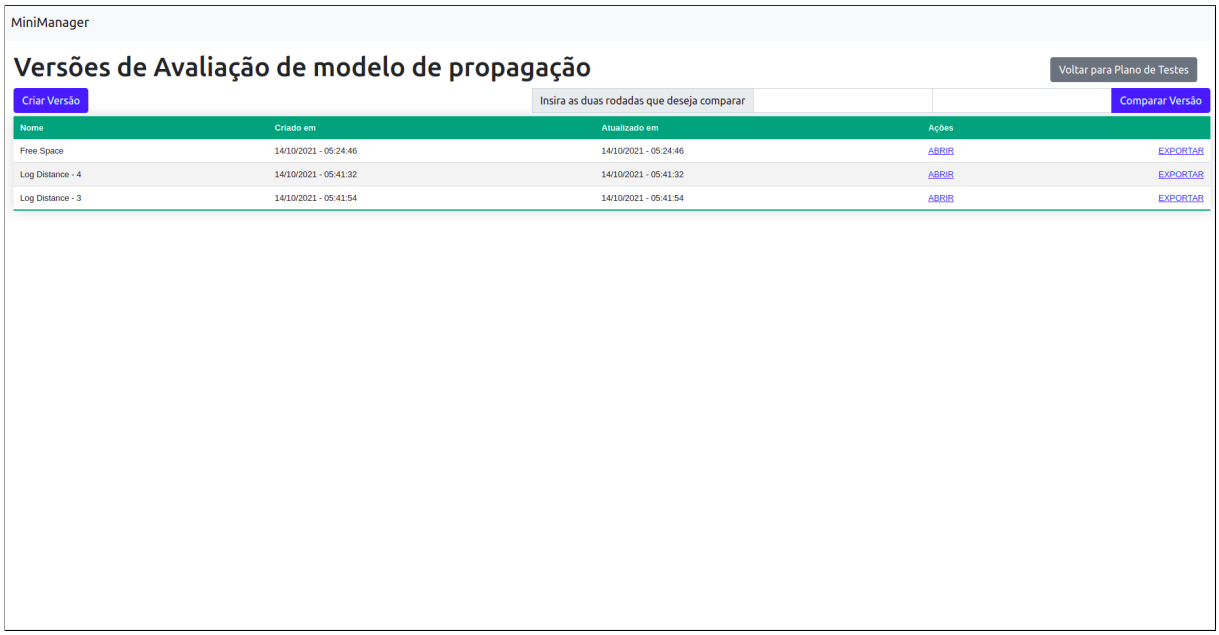

Figura 34 – Tela de Visualização das Versões

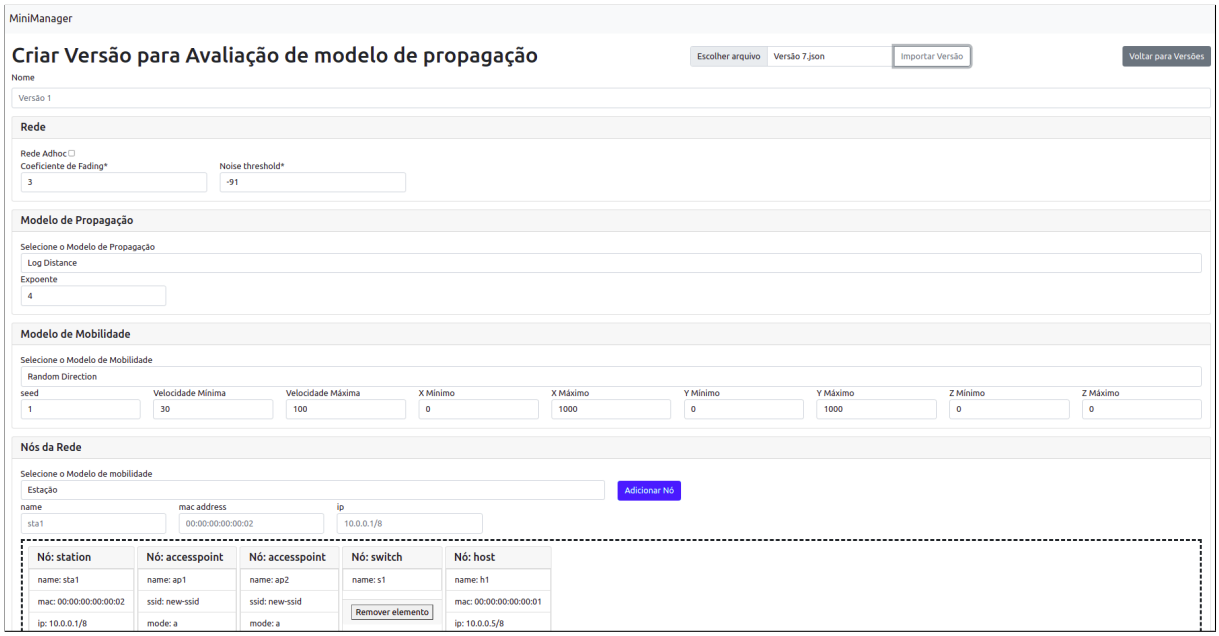

Figura 35 – 1º Parte da Tela de Criação de Versões dos Planos de Testes

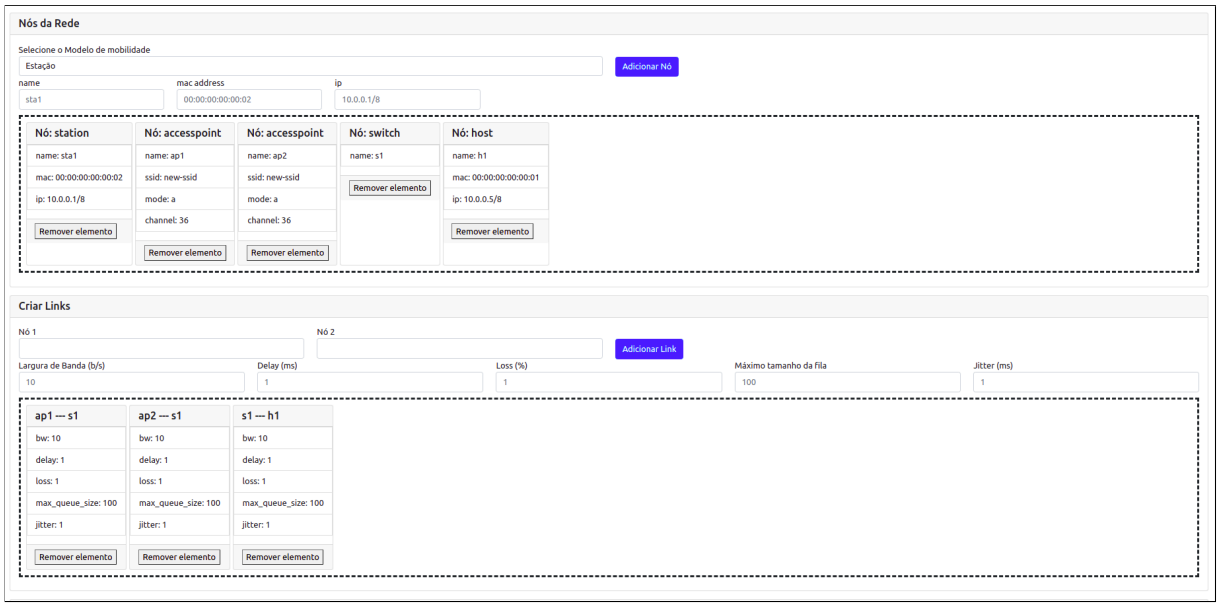

Figura 36 – 2º Parte da Tela de Criação de Versões dos Planos de Testes

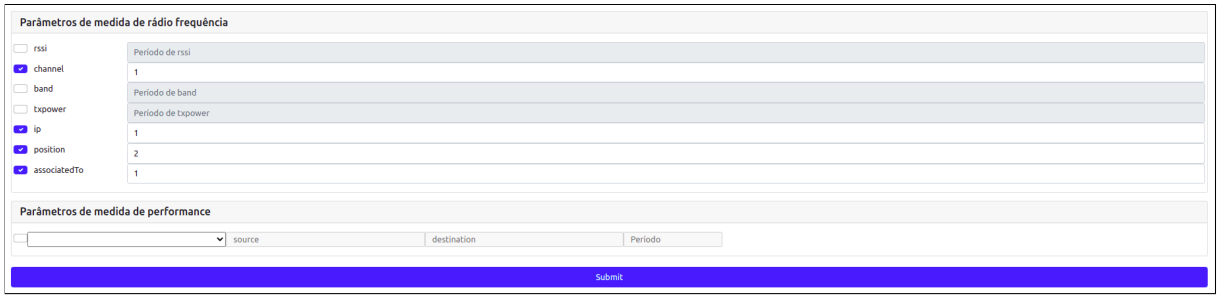

Figura 37 – 3º Parte da Tela de Criação de Versões dos Planos de Testes

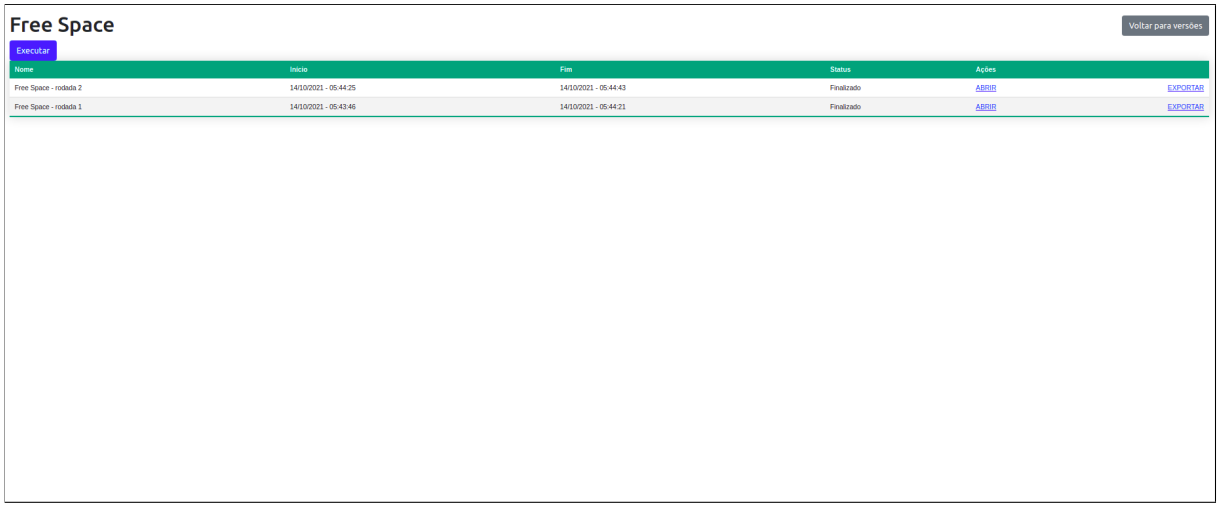

Figura 38 – Tela de Visualização das Rodadas

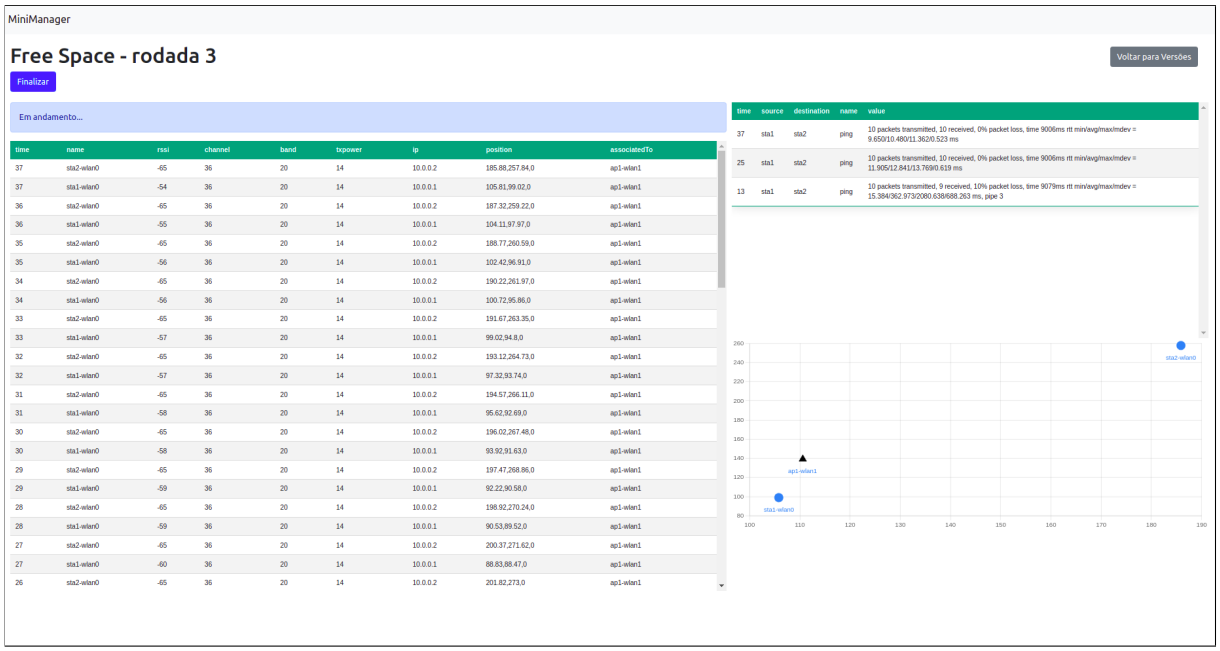

Figura 39 – Tela de Execução das Rodadas

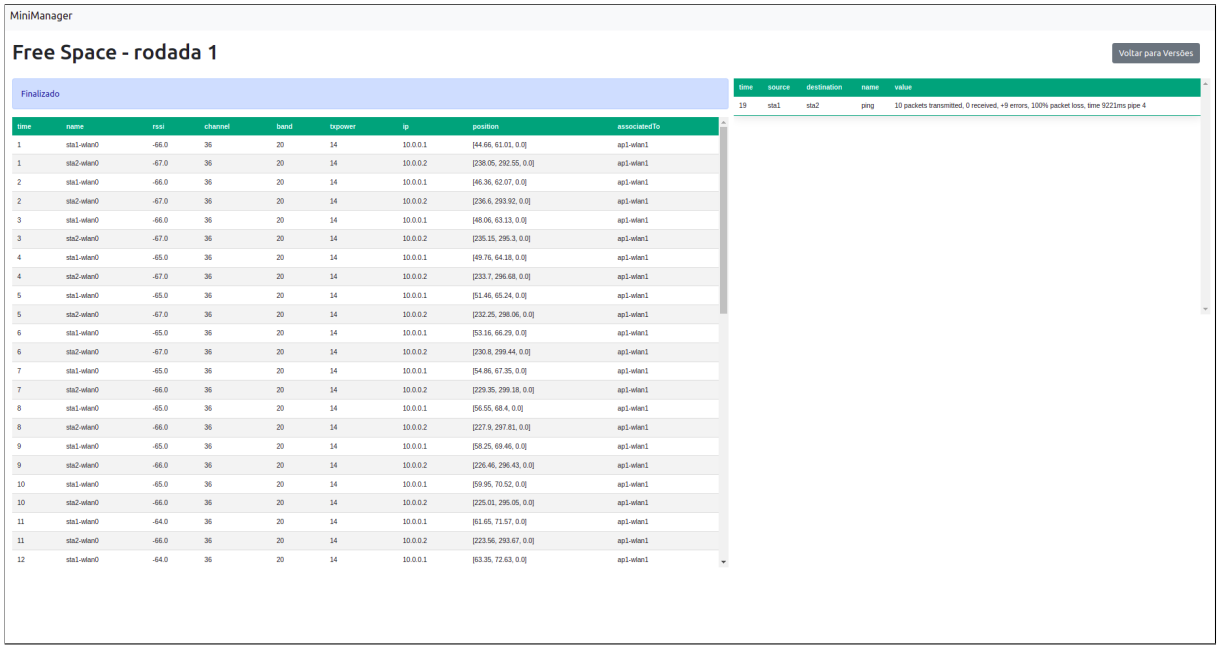

Figura 40 – Tela de Visualização das Rodadas

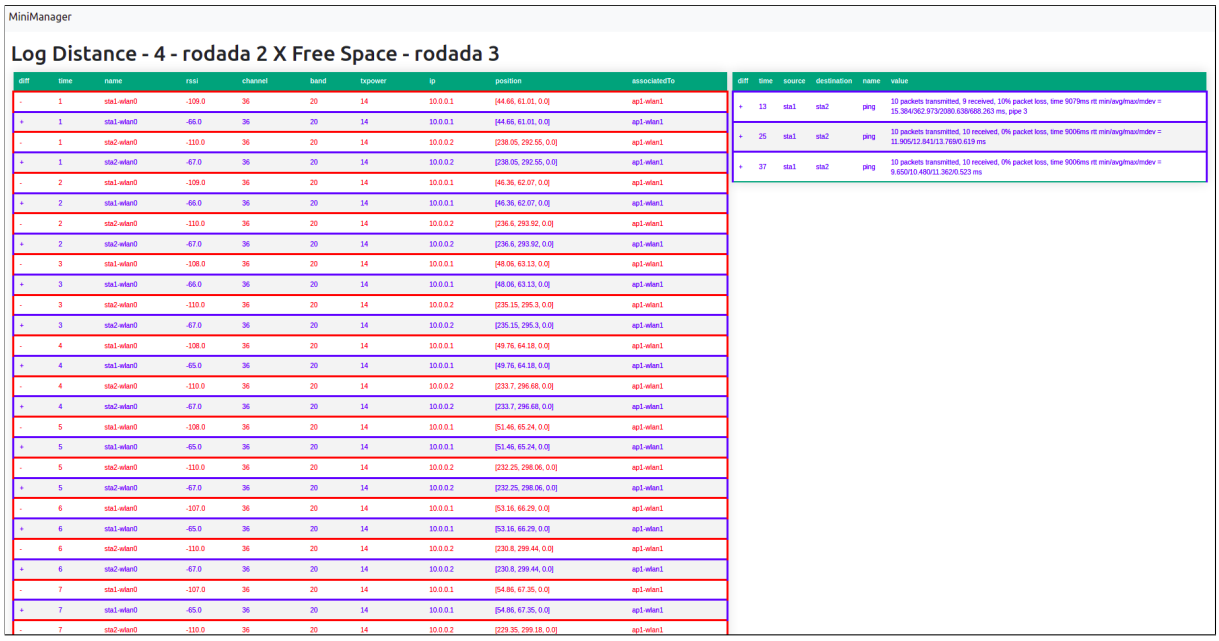

Figura 41 – Tela de Comparação de Rodadas