MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
SEÇÃO DE ENGENHARIA DE COMPUTAÇÃO

VINÍCIUS RAMOS EDUARDO
LUIS HELDER LIMA BARBOSA

FERRAMENTA PARA ANÁLISE COMPARATIVA ENTRE SIMULAÇÃO E MODELAGEM ANALÍTICA

Rio de Janeiro
2015

INSTITUTO MILITAR DE ENGENHARIA

VINÍCIUS RAMOS EDUARDO
LUIS HELDER LIMA BARBOSA

FERRAMENTA PARA ANÁLISE COMPARATIVA ENTRE SIMULAÇÃO E MODELAGEM ANALÍTICA

Projeto de Final de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia.

Orientador: Gabriela Moutinho de Souza Dias, M.Sc.
Orientador: Anderson Fernandes Pereira dos Santos, D.Sc.

Rio de Janeiro
2015

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

004.68 Eduardo, Vinícius Ramos
E24f Ferramenta para análise comparativa entre simulação e modelagem analítica / Vinícius Ramos Eduardo, Luis Helder Lima Barbosa; orientados por Gabriela Moutinho de Souza Dias e Anderson Fernandes Pereira dos Santos - Rio de Janeiro: Instituto Militar de Engenharia, 2015.

44p.: il.

Projeto de Fim de Curso (PROFIC) - Instituto Militar de Engenharia - Rio de Janeiro, 2015.

1. Curso de Engenharia de Computação - Projeto de Fim de Curso. 2. Redes de Computadores. 3. DTN - Delay-Tolerant Network. I. Barbosa, Luis Helder Lima. II. Dias, Gabriela Moutinho de Souza. III. Santos, Anderson Fernandes Pereira dos. IV. Título. V. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

VINÍCIUS RAMOS EDUARDO
LUIS HELDER LIMA BARBOSA

FERRAMENTA PARA ANÁLISE COMPARATIVA ENTRE SIMULAÇÃO E MODELAGEM ANALÍTICA

Projeto de Final de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia.

Orientador: Gabriela Moutinho de Souza Dias, M.Sc.

Orientador: Anderson Fernandes Pereira dos Santos, D.Sc.

Aprovada em 25 de Maio de 2015 pela seguinte Banca Examinadora:

Gabriela Moutinho de Souza Dias, M.Sc., do IME

Anderson Fernandes Pereira dos Santos, D.Sc. do IME

Julio Cesar Duarte, D.Sc., do IME

Ricardo Choren Noya, D.Sc., do IME

Rio de Janeiro
2015

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

BPMN *Bussiness Process Model and Notation*

DTN *Delay/Disruption-Tolerant Network*

GCC *GNU Compiler Collection*

HTTP *HyperText Transfer Protocol*

IP *Internet Protocol*

JNA *Java Native Access*

SIR *Susceptible-Infected-Recovered*

TCP *Transmission Control Protocol*

The ONE *The Opportunistic Network Environment*

TTL *Time to Live*

RESUMO

As redes de computadores são o núcleo da comunicação moderna, novas redes surgem e seus estudos crescem a cada dia. Uma delas é a rede DTN (*Delay-Tolerant Network* ou *Disruption-Tolerant Network*), uma rede tolerante a atrasos e interrupções bastante útil para cenários de desastre ou comunicação em ambientes sem infraestrutura de telecomunicações.

Para ajudar nesses estudos cada vez mais presentes, este projeto apresenta o desenvolvimento de uma ferramenta de análise capaz de gerar resultados de simulações e modelagens de maneira simples e eficaz. O simulador utilizado foi o The ONE (*The Opportunistic Network Environment*), específico para redes DTN, e o modelo utilizado foi o desenvolvido em [1].

O objetivo desta ferramenta é otimizar o trabalho do pesquisador, possibilitando que o mesmo possa se dedicar à análise dos resultados em vez de desperdiçar tempo na obtenção destes. A simplicidade é o princípio da ferramenta, que foi desenvolvida para facilitar o trabalho do usuário.

ABSTRACT

Computer networks are the core of modern communication, new types of networks appear more and more often, and their studies grow in the same path. One of them is the DTN network, a delay and disruption tolerant network that is really helpful in disasters scenarios or in places with no network infrastructure.

In order to help those studies, this project presents the development of an analysis tool capable of generating results from simulators and analytical models in a simple and effective way. The simulator used was the The ONE (The Opportunistic Network Environment), specific for DTN networks, and the model used was the one developed in [1].

The objective of this tool is to optimize the work of the researcher in a way that one can focus on analyzing the results, instead of wasting time gathering them. Simplicity is the principle behind this tool which was developed to facilitate the job of an user.

1 INTRODUÇÃO

As redes de computadores são, nos dias de hoje, o núcleo da comunicação moderna. Milhares de pessoas utilizam a Internet, por exemplo, para os mais diversos fins, empresas e governos dependem de suas redes para o controle de seus processos e comunicação entre seus integrantes, e até o sistema de telefonia tem migrado para redes que utilizam o protocolo IP (*Internet Protocol*).

Nos últimos anos, o interesse em um tipo de rede de computadores em particular, chamada de DTN (*Delay-Tolerant Network* ou *Disruption-Tolerant Network*) ou rede tolerante a atrasos e desconexões, tem sido crescente. Ela foi desenvolvida para funcionar nos chamados ambientes desafiadores, onde os protocolos tradicionais de rede não funcionam satisfatoriamente.

No campo da pesquisa em redes de computadores, duas ferramentas são bastante importantes para que os pesquisadores possam tirar conclusões sobre o desempenho dos protocolos propostos e também sobre o comportamento da rede em si: a simulação e a modelagem analítica. No caso particular das DTNs, um simulador bastante utilizado é o The ONE (*The Opportunistic Network Environment*), que foi desenvolvido especificamente para avaliação de protocolos e modelagens para esse tipo de rede [2]. No campo da modelagem, não existe atualmente nenhum modelo analítico genérico para DTN [1], existem, porém, estudos para o desenvolvimento desse modelo, divididos em diversas abordagens, como por exemplo, o modelo epidemiológico determinístico e o epidemiológico estocástico [3] [4].

1.1 JUSTIFICATIVA

Unificar a maneira como obtemos resultados dessas duas ferramentas de forma mais simplificada pode facilitar o trabalho de pesquisadores da área, e é nesse âmbito que este projeto se foca, criar uma única ferramenta para simplificar e automatizar a maneira de obtenção de resultados. Essa automatização possibilita realizar análises de maneira mais focada nos resultados, poupando o tempo do pesquisador que era gasto com a obtenção dos mesmos.

1.2 MOTIVAÇÃO

O campo de estudo das DTNs, por ser relativamente novo, ainda precisa ser bem pesquisado e o comportamento dos nós desse tipo de rede precisa ser modelado, para que se possam desenvolver os melhores protocolos para os diversos cenários onde as DTNs são aplicáveis. O estudo de modelagens visa buscar mais conhecimento do comportamento da rede, e é de grande importância, uma vez que não existe modelo genérico proposto. Para testar a validade de uma modelagem, é necessário compará-la com algo que se assemelhe a uma rede DTN, fazendo-se necessário o uso de uma segunda ferramenta, o simulador. Obter resultados da modelagem e da simulação, para o mesmo cenário, é o objetivo do pesquisador, porém esse processo tem se mostrado trabalhoso. Portanto, é importante que os pesquisadores disponham de ferramentas que agilizem essa obtenção.

As redes DTN carecem de estudos mais aprofundados na área de simulação e modelagem, isso faz com que cada vez mais pesquisadores se interessem pela área e busquem respostas para essas perguntas. Este trabalho busca contribuir para a melhor obtenção de resultados, de forma simples e rápida, criando uma ferramenta que possibilita ao pesquisador se preocupar com o que realmente importa, os resultados e não sua obtenção.

1.3 OBJETIVOS

O objetivo do presente trabalho é desenvolver uma ferramenta, escrita na linguagem de programação Java, para automatizar o processo de obtenção de resultados a partir de duas ferramentas secundárias: simulações de cenários de redes DTN, por meio do *software* The ONE, e modelagem matemática epidêmica estocástica, desenvolvida em [1], dos mesmos cenários. A ferramenta proposta integrará tanto a simulação quanto a modelagem em uma interface só, de modo que o pesquisador foque na análise dos resultados, e não na obtenção dos mesmos.

1.3.1 OBJETIVO SECUNDÁRIO

Como objetivo secundário, definiu-se a funcionalidades que serão abrangidas pela ferramenta de integração proposta nesse projeto:

- Configuração de cenários;
- Execução do simulador;
- Execução da modelagem;

- Disponibilização dos resultados ao usuário.

Uma descrição aprofundada de cada funcionalidade será apresentada no Capítulo 3.

1.4 METODOLOGIA

O desenvolvimento deste trabalho teve início com a fundamentação teórica, apresentada no Capítulo 2. Durante a primeira documentação do projeto, adotou-se como foco uma breve introdução acerca de redes DTN, estudos de modelagem na área e do simulador The ONE. Também foi foco desenvolver uma descrição geral do sistema, bem como o desenvolvimento de uma primeira versão da ferramenta sem uma interface gráfica, somente por linha de comando, e para o sistema operacional Linux.

Já na segunda parte da documentação, foi desenvolvido um fluxo, explicando de forma gráfica o funcionamento da ferramenta, e uma descrição do desenvolvimento da mesma. Nessa parte também foi desenvolvida a segunda versão do código, que já abrangia todas as funcionalidades, porém sem interface gráfica.

Por fim, foi desenvolvida uma interface gráfica, que facilita a utilização da ferramenta. Ela traz consigo algumas funcionalidades extras para a ferramenta, como pode ser visto na Seção 3.3. Além da interface, também foi implementada a criação de gráficos comparativos, que são parte fundamental dos resultados.

1.5 ORGANIZAÇÃO DA MONOGRAFIA

O conteúdo do presente trabalho está organizado em cinco capítulos, sendo quatro de desenvolvimento e um de conclusão.

No Capítulo 2, será dada uma introdução teórica acerca de redes DTN, modelagens e simulador. Nele serão desenvolvidos os seguintes tópicos: pequena introdução acerca de redes tolerantes a atrasos, tipos de modelagens em redes DTN e descrição do simulador The ONE.

No Capítulo 3, será discutido uma descrição geral da ferramenta e um fluxo de tarefas que ilustra o seu funcionamento, de forma bem detalhada. Além disso, será descrito de que maneira a ferramenta foi modelada, mostrando seu diagrama de classe, bem como uma descrição das classes mais importantes. Por fim, mostra-se como a interface foi desenvolvida, e quais as funcionalidades que ela agrega para a ferramenta. Seu funcionamento e integração com a ferramenta também será abordado.

No Capítulo 4, será mostrado um exemplo do funcionamento da ferramenta, tanto pela linha de comando quanto pela interface gráfica. Cada etapa do funcionamento será detalhada seguindo o exemplo proposto.

Para finalizar, o Capítulo 5 apresenta as conclusões obtidas e próximos passos decorrentes desse projeto.

2 FUNDAMENTAÇÃO TEÓRICA

A maioria dos protocolos de transporte, incluindo o TCP (*Transmission Control Protocol*), parte do princípio de que os nós comunicantes estão conectados durante todo o tempo em que se dá a transmissão dos dados, caso contrário a comunicação falha e os dados não são entregues. Em alguns casos, porém, essa hipótese não é satisfeita. Em uma rede espacial, por exemplo, onde os nós seriam satélites e estações terrestres, é impossível garantir que sempre exista conectividade entre dois nós que desejam se comunicar, pois os satélites ocasionalmente estarão fora de alcance. Cenários como esse são chamados de ambientes desafiadores. Esses ambientes referem-se a uma classe mais ampla de redes sem fio sem infraestrutura [1], como por exemplo, uma cidade após uma catástrofe natural.

Ambientes desafiadores são cenários de redes onde os protocolos de transporte usuais, como o TCP, não funcionam, devido a condições extremas que dificultam sua operação. Entre tais condições, destacam-se três principais:

- Conectividade intermitente: Não se pode garantir a conexão fim a fim entre nós. Os nós poderão se conectar apenas ocasionalmente.
- Atrasos longos: Os protocolos de transporte usuais também não funcionam bem se os atrasos no encaminhamento de pacotes forem muito grandes.
- Alta taxa de erros: Pode tornar inviável a operação dos protocolos de redes usuais, pois aumentaria muito o tráfego devido à correção e/ou retransmissão de pacotes.

2.1 REDES TOLERANTES A ATRASOS

As redes tolerantes a atrasos, ou DTNs, são um tipo de rede cuja arquitetura foi desenvolvida para ser eficaz em ambientes desafiadores. A inspiração para seu desenvolvimento veio do projeto IPN, ou *Interplanetary Internet*, no qual se objetivava desenvolver um protocolo que seria a base de uma possível futura rede interplanetária. Kevin Fall publicou um dos trabalhos mais importantes da área em 2003, a partir daí, o estudo de Redes Tolerantes a Atrasos e Desconexões se intensificou [1].

Para lidar com as dificuldades dos ambientes desafiadores, as DTNs utilizam uma abordagem em que os nós não somente armazenam e encaminham as mensagens, mas também são capazes de vencer os obstáculos de ambientes desafiadores pois retêm a mensagem,

movem-se e a transmitem quando houver conexão [1]. Dessa forma não é necessário haver conexão fim a fim entre o transmissor e o receptor, a mensagem viajará de nó em nó, até chegar ao seu destino. Essa técnica chama-se comutação de mensagens [5], mas também é conhecida pelo seu nome em inglês, *store-carry-forward*.

Ao receber uma mensagem, um nó precisa saber para quem ele deverá enviar a mensagem, de modo que ela chegue ao seu destino. Esse comportamento é definido pelo protocolo de roteamento utilizado em uma determinada DTN. Existem vários protocolos de roteamento possíveis, cada um visando otimizar um aspecto do funcionamento da rede, sendo o mais simples deles o chamado roteamento epidêmico, no qual o nó encaminha a mensagem para todos os nós alcançáveis. Desse modo, garante-se que o destino receberá a mensagem assim que possível. Essa abordagem aumenta o tráfego na rede e a demanda por armazenamento nos nós, porém é a mais simples de ser implementada, além de oferecer um menor atraso na entrega da mensagem.

2.2 MODELAGEM MATEMÁTICA

Apesar de ser uma área recente de estudos, as redes DTN possuem vários campos de pesquisa, e destaca-se, dentro desse ambiente, o estudo de modelagem para tal rede. Este campo abrange a construção de modelos analíticos, além de análises de desempenho para diversos protocolos de roteamento e cenários de simulação [1] [6].

Tais modelos ajudam a compreender melhor a dinâmica da rede, além de tornar possível a realização de projetos com maior precisão [1]. Porém, é necessária uma extensa validação que conta com a ajuda de simuladores. Tais *softwares* são utilizados para gerar resultados comparáveis com os modelos analíticos propostos. Tal validação é essencial para que o modelo seja aceito.

No caso desse projeto, o modelo utilizado na ferramenta usa como base o modelo SIR (*Susceptible-Infected-Recovered*), que é um modelo consagrado no ramo da epidemiologia [3] [4] e que pode ser adaptado para modelagem de troca de mensagens em DTNs epidêmicas, ou seja, DTN que utiliza roteamento epidêmico [1]. Esse modelo possui duas abordagens distintas, estocástica ou determinística, sendo a primeira utilizada no modelo proposto. Uma das diferenças entre as abordagens é que o SIR estocástico considera o número de indivíduos Suscetíveis e Infectados como variáveis aleatórias correlacionadas [1], diferentemente do modelo clássico determinístico, que consiste num sistema de equações diferenciais ordinárias, no qual o número de Suscetíveis e Infectados varia como o tempo de forma determinística.

A adaptação do modelo SIR estocástico para DTNs foi proposto por [1] e recebeu o nome de Modelo Epi-DTN. Além do modelo, um código escrito em C++ foi desenvolvido por [1], e é partindo dele que se obtêm resultados para que o modelo seja testado. Esse código gera resultados a partir dos dados contidos nos relatórios gerados pelo simulador The ONE, que será explicado na seção 2.3. O The ONE possibilita a geração de vários relatórios, mas, para a modelagem, os dados oriundos dos seguintes relatórios são utilizados:

- *CommunicationStatsReport2* - Estatísticas de comunicação entre nós.
- *EventLogReport* - Um *log* de eventos da simulação. Esses eventos incluem conexão entre nós, envio e recebimento de mensagens, entre outros.
- *TotalEncountersReport* - Um relatório da distribuição de quantos encontros cada nó realizou.

Os resultados da modelagem são armazenados em arquivos e usados como métricas para análise da eficiência do modelo.

Os resultados fornecidos por esse modelo analítico devem ser, então, comparados com um modelo proposto por um simulador, que também simula o roteamento epidêmico. Comparar pode não ser a tarefa mais difícil, às vezes, obter os resultados em primeiro lugar pode ser mais custoso, já que não existe ferramenta específica para gerar diretamente todos os resultados necessários.

2.3 SIMULADOR THE ONE

O simulador escolhido para ser a base de comparação da modelagem da rede foi o The ONE, que é uma ferramenta de simulação de eventos discretos baseada em agentes [7]. Foi desenvolvida por pesquisadores finlandeses na Universidade Tecnológica de Helsinki, é um projeto aberto, escrito em Java e vem sendo desenvolvido desde 2008.

O The ONE permite a simulação de movimento de nós baseada em diferentes modelos de movimentação, que podem ser sintéticos ou reais, e permite também o roteamento de mensagens utilizando diversos algoritmos próprios para o The ONE [7]. O simulador pode servir de base para vários tipos de modelagens, porém limitações existem, já que o simulador não abrange todos os algoritmos de roteamento, por exemplo. Apesar disso, ele se mostra suficientemente bom para o propósito deste trabalho. Outro ponto a favor do

simulador é o fato de estar disponível gratuitamente na Internet, para qualquer sistema que disponha de uma máquina virtual Java.

O funcionamento do The ONE pode ser através de interface gráfica ou linha de comando, e ele baseia-se no arquivo de configuração definido pelo usuário. Cada arquivo define um cenário, além de outras funcionalidades, como o formato de saída dos resultados e onde a mensagem a ser trocada está armazenada. Este arquivo de configuração é um simples arquivo de texto. Um exemplo pode ser visto no Apêndice A.

Os cenários são construídos definindo-se o número de nós e suas capacidades [7], bem como as dimensões do terreno. Cada conjunto único de variáveis define um cenário, por exemplo, 30 nós com alcance de 50 metros numa área de 600x600 metros quadrados em uma simulação com tempo máximo de 100 segundos. Cada simulação é executada um certo número de vezes definido pelo número de sementes, ou seja, 100 sementes é o mesmo que 100 simulações. Cada semente, utilizando um gerador de números pseudo-aleatórios, altera a posição inicial e a movimentação dos nós na área definida.

Cada arquivo de configuração caracteriza um único cenário, ou seja, se qualquer uma das variáveis mudar, um novo arquivo tem que ser gerado, e cada simulação é executada para um único cenário. Porém, para se fazer uma análise e validação da modelagem com qualidade, é preciso obter uma grande quantidade de resultados. Para se obter essa grande quantidade, variando os parâmetros da simulação, e com isso, configurando diversos cenários diferentes, se faz necessário uma grande quantidade de arquivos. Além de ser necessário executar o simulador este mesmo número de vezes. Este processo, quando não automatizado, exige um tempo considerável do pesquisador. Porém o trabalho não acaba nesse ponto, além de executar as simulações é necessário integrar os resultados gerados por ela com aqueles gerados pela modelagem. A geração de resultados, tanto da simulação quanto da modelagem, é única, para cada cenário. É justamente essa cadeia de processos que a ferramenta desenvolvida nesse projeto tem objetivo de automatizar. Além disso, a ferramenta alavanca o tempo do pesquisador no armazenamento e organização dos arquivos de configuração e resultados.

Para exemplificar, suponha uma DTN com 30 nós, alcance de 50 m, movendo-se em uma área de 600x600 metros quadrados. Como já mencionado, para rodar a simulação de encaminhamento de mensagens, com tais características, é preciso um arquivo de configuração. Este arquivo é lido pelo simulador para execução da simulação, que gera os resultados em forma de relatórios diversos (as opções de relatórios são configuradas também pelo usuário). Caso o pesquisador queira variar o alcance de 50 a 200 metros, com

passo 10, ele precisaria repetir esse processo 16 vezes, isso somente com a variação de um parâmetro. Pode-se notar, assim, a complexidade de obtenção de resultados de uma grande quantidade de cenários simulados.

3 DESCRIÇÃO DA FERRAMENTA

A ferramenta desenvolvida nesse projeto tem como objetivo final integrar dois processos: simulação e modelagem. Tal integração envolve todas as etapas de obtenção de resultados, desde a configuração do cenário de simulação até a geração de gráficos comparativos. Conforme descrito nas seções 2.2 e 2.3, a ferramenta de modelagem é o código desenvolvido por [1] e a ferramenta de simulação é o simulador The ONE. Os resultados da simulação e da modelagem serão disponibilizados no formato de texto, já os gráficos podem ser gerados em dois formatos: eps e pdf, com a opção de exportação para a máquina do usuário, compactados em um aquivo ZIP. O gráfico é plotado utilizando o programa Gnuplot, a partir dos resultados finais da simulação e da modelagem. O código da ferramenta foi escrito na linguagem de programação Java de maneira escalável, ou seja, se algum outro desenvolvedor desejar expandir a ferramenta ou escrever módulos separados, conseguirá fazê-lo sem alterar a estrutura básica do projeto.

Inicialmente, o sistema pede ao usuário os parâmetros para configurar o ambiente de simulação, que serão usados também pelo código da modelagem Epi-DTN. Existirá a possibilidade de, em vez de o usuário prover um valor fixo, prover um intervalo de variação e um passo, para qualquer um dos parâmetros, possibilitando assim, uma simulação integrada de vários cenários. Cada cenário ou conjunto de cenários (no caso de haver variação) gerado receberá um nome, que será armazenado internamente na ferramenta em um arquivo de configuração, tal arquivo não tem nenhuma relação com o de configuração do The ONE. Quando o usuário utilizar a ferramenta novamente, ele poderá escolher um cenário já definido e configurado anteriormente, apenas utilizando seu nome, sem necessitar configurar os parâmetros novamente.

Após a ferramenta receber os parâmetros, os arquivos de configuração do The ONE serão gerados, um para cada cenário. Cada arquivo de configuração será utilizado para rodar uma simulação, que será chamada de dentro da ferramenta, e os resultados, armazenados em arquivos, que serão acessados futuramente para tratamento estatístico. Os resultados do The ONE, em si, não trazem muitas informações úteis para comparação com a modelagem, por isso é necessário passar por um tratamento. Tal tratamento é realizado por meio de um código em C, utilizado para retirar métricas estatísticas. É esse resultado pós tratamento que será usado como resultado da simulação e posterior

comparação com o resultado da modelagem. Tal comparação é executada pelo próprio código do modelo Epi-DTN.

Uma vez de posse dos resultados da simulação, é necessário gerar os da modelagem através do código da Epi-DTN. O código usa os mesmos parâmetros que o usuário proveu no início, e além desses valores, ele também lê algumas das métricas estatísticas também geradas anteriormente. Essa leitura é usada para fazer algumas comparações e medições de erros, fazendo parte da análise entre resultados de simulação e modelagem. Tais detalhes são específicos da implementação do código, caso o código mude, esse fluxo também muda. O código gera 7 arquivos para cada cenário, que são os resultados disponibilizados ao usuário referente à modelagem.

De posse dos dois resultados finais, a próxima etapa é a geração dos gráficos comparativos. Primeiramente, a ferramenta, a partir dos resultados da simulação e modelagem, gera os arquivos com extensão .gnu, que serão lidos pelo Gnuplot, sendo um para cada cenário. A geração dos gráficos propriamente dita é feita somente após o formato de visualização ser definido, de acordo com a vontade do usuário.

Neste ponto, a ferramenta terá acesso aos dois resultados finais e aos arquivos de geração dos gráficos. Para o formato dos gráficos, o usuário tem como primeira opção a extensão eps, que é o padrão definido pela ferramenta. A segunda opção é pdf, que é um formato mais conhecido e mais simples de visualizar. Uma vez definida a extensão dos gráficos, eles são gerados com o Gnuplot. Como o número de resultados gerados por um cenário é relativamente alto, eles serão compactados para o fomato ZIP e só então, disponibilizados para exportação.

Será mostrado na seção seguinte uma imagem do fluxo do sistema, com todas as etapas descritas anteriormente. Nessa imagem pode ser visto tudo que foi descrito, facilitando assim o entendimento do funcionamento da ferramenta. Em outra seção será descrito o diagrama de classes, permitindo, assim, entender como o sistema foi modelado. Após entender como a ferramenta funciona e como ela está modelada, será detalhada como foi criada a interface gráfica e como ela auxilia a comunicação entre o usuário e a ferramenta. Por fim, uma seção de validação de resultados será mostrada, a fim de validar o correto funcionamento da ferramenta.

3.1 FLUXO DA FERRAMENTA

A figura 3.1 mostra o funcionamento básico da ferramenta de integração. Nela está detalhado o fluxo de tarefas, bem como uma descrição dos mesmos. Esse fluxo segue o

modelo BPMN (*Bussiness Process Model and Notation*) e foi desenvolvido utilizando a ferramenta Bizagi.

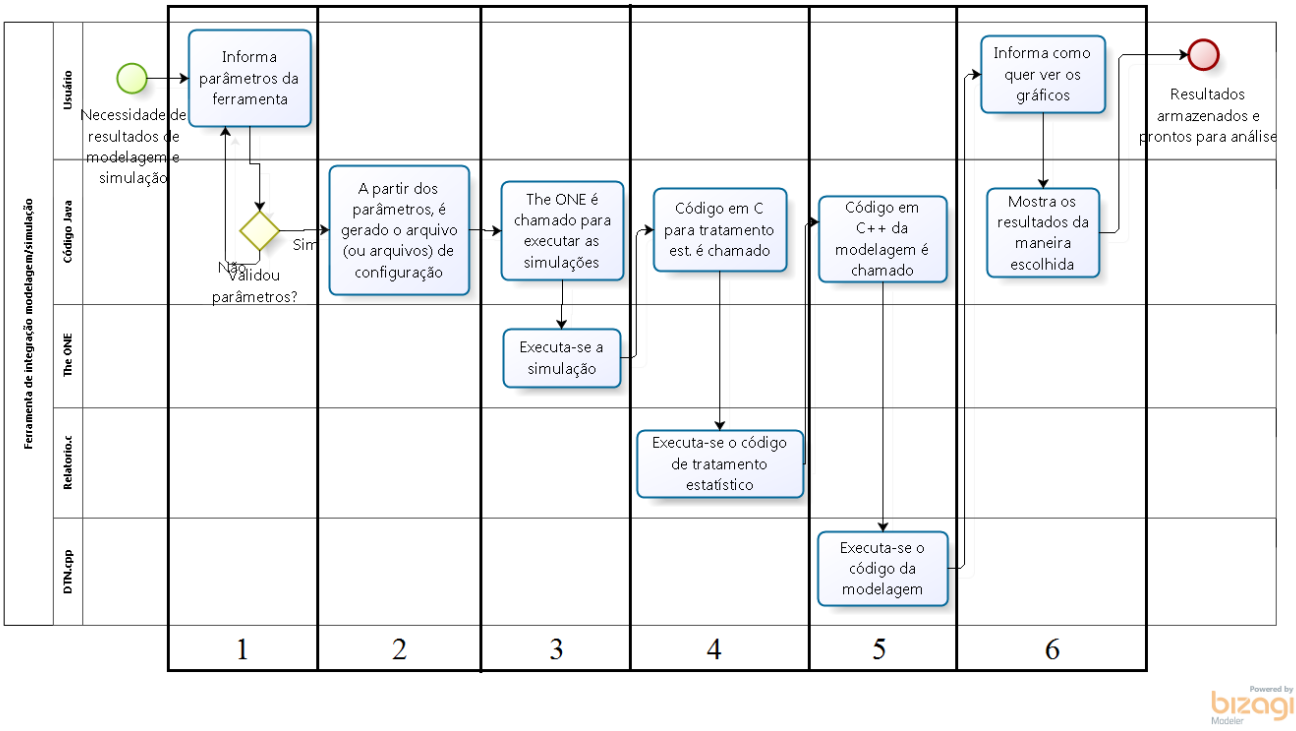

FIG. 3.1: Fluxo da Ferramenta

Para facilitar o entendimento, será detalhado o funcionamento de cada coluna numerada no fluxo. Primeiramente, cada coluna será nomeada:

1. Configuração de parâmetros de entrada;
2. Geração de arquivos de configuração;
3. Execução das simulações;
4. Tratamento estatístico dos resultados da simulação;
5. Execução da modelagem matemática;
6. Visualização dos resultados em gráficos comparativos.

O fluxo se inicia com o desejo do usuário de obter resultados da execução de uma simulação e de uma modelagem, isso ocorre antes do início do funcionamento da ferramenta.

Inicialmente, na etapa de configuração de parâmetros de entrada, ele fornece os parâmetros necessários para o funcionamento da ferramenta, eles são divididos em dois grupos:

a) Parâmetros do Cenário: Número de nós, alcance, tamanho do cenário, tamanho do *buffer* do nó, velocidade do nó e TTL da mensagem;

b) Parâmetros da Simulação: Número de sementes, nome do cenário, intervalo de atualização da posição dos nós e tempo de simulação.

Um desses parâmetros é de suma importância, que é o nome do cenário. É a partir dele que a ferramenta salva os cenários, para caso o usuário deseje executá-los novamente, o faça de maneira rápida, uma vez que existirá uma lista de nomes de cenários previamente executadas. Para acessar essa lista, no menu inicial da ferramenta, ao invés de executar um novo cenário, o usuário deve escolher "Abrir Simulação Salva". Existe outro que pode variar mas ele é definido pela própria ferramenta, o diretório de saída dos resultados. Todos os outros parâmetros vistos no Apêndice A são fixos, como os tipos de relatórios gerados, velocidade de transmissão da mensagem e o tipo de roteamento. Uma vez de posse dos parâmetros é feito uma validação, que testa basicamente a compatibilidade de tipos, por exemplo, se no número de nós não foi colocado um caractere.

Com os parâmetros validados, inicia-se a segunda etapa de geração dos arquivos de configuração. Nessa etapa a ferramenta irá gerar um arquivos de configuração para cada conjunto único de parâmetros. Por exemplo, se existirem parâmetros que variem em um intervalo de valores, seria gerado um arquivo de configuração do The ONE para cada combinação linear distinta dos parâmetros. No caso mais simples, onde nenhum parâmetro varia, somente um arquivo seria gerado.

Os próximos passos são repetidos para cada cenário criado a partir de um arquivo de configuração e correspondem à terceira etapa do fluxo, a execução das simulações. Inicialmente a ferramenta chama o The ONE, tal processo não é tão simples, pois devido a limitações do simulador, ele só é executado corretamente quando chamado dentro do próprio diretório onde se encontra. Para contornar esse problema, é criado um .jar que engloba todo o projeto e é colocado na pasta do simulador, ou seja, a ferramenta será executada como se estivesse no diretório do simulador. Um script foi criado para automatizar esse procedimento, basta chamá-lo uma vez, antes de usar a ferramenta pela primeira vez. Foi criado também um comando "TheONE", no intuito de facilitar a chamada do simulador; esse comando executa o "one.sh", o script que efetivamente executa a simulação. Uma vez chamado o simulador corretamente, o cenário da vez é executado, gerando resultados na forma de relatórios (Nesse projeto, todos os resultados terão o mesmo formato, como dito anteriormente).

Após a execução do simulador, é hora de fazer a quarta etapa, o tratamento estatístico dos resultados da simulação. O responsável pelo tratamento é o código Relatorio.c, ele é chamado pela ferramenta usando JNA (*Java Native Access*) que permite acesso a bibliotecas nativas e compartilhadas, ou seja, foi necessário gerar uma dessas bibliotecas a partir do código C para ter acesso às suas funcionalidades. A tecnologia utilizada para gerar uma biblioteca nativa foi o compilador GCC (*GNU Compiler Collection*). Para chamar o código, usa-se alguns parâmetros fornecidos no começo: o alcance, o tamanho da área, tempo de execução e o nome, adicionados ao nome do diretório de resultados, ao número mínimo e ao máximo de nós. Além dos parâmetros de entrada, esse código realiza leitura em 3 relatórios originários do The ONE: "CommunicationStatsReport2", "EventLogReport" e "TotalEncountersReport".

Os resultados do tratamento estatístico possuem uma característica peculiar, alguns deles são gerados para o conjunto de nós, ou seja, ao invés de criar uma saída para cada cenário, é criada uma saída para cada grupo de cenários agrupados pelo número de nós, ou seja, é como se a variação do número de nós se comportasse como um valor fixo. Nesse caso particular, é gerado um arquivo para cada combinação linear distinta dos parâmetros, com excessão do número de nós.

Depois da execução do tratamento estatístico é iniciada a quinta etapa do fluxo, que é a execução da modelagem matemática, através do código DTN.cpp. Sua chamada ocorre da mesma maneira que Relatorio.c e sua execução é similar também. Os parâmetros de entrada são os mesmos, já os arquivos de leitura provêm do tratamento estatístico, que no caso do DTN.cpp, são quatro. Alguns resultados da modelagem também são agrupadas pelo número de nós, em um total de dois arquivos, e os que não são agrupados, num total de cinco arquivos.

Por fim, inicia-se as tarefas da última etapa, que é a visualização dos resultados em gráficos comparativos . Inicialmente, é necessário criar os arquivos de geração dos gráficos comparativos, que são os últimos resultados que faltam. A ferramenta gera, para cada cenário, um arquivo de extensão .gnu que será usado no futuro para criar os gráficos. Esse arquivo serve para gerar cinco grupos de arquivos em três escalas diferentes de tempo (segundos, minutos e hora), ou seja, serão 15 gráficos para cada cenário. Os cinco grupos são: "Nós Disponíveis na Simulação", "Nós Disponíveis na Modelagem", "Nós Transmissores na Simulação", "Nós Transmissores na Modelagem" e "Comparação dos Nós Transmissores na Modelagem Estocástica, Determinística e Simulação".

Nesse ponto, já se possui todos os dados necessários para mostrar os resultados finais

integrados ao usuário, falta decidir a maneira como os gráficos serão visualizados. Tal decisão fica a cargo do usuário e pode variar entre .eps e .pdf. Após a escolha, os gráficos são gerados no formato escolhido, permitindo a criação de um arquivo ZIP contendo todos os resultados do cenário, que será disponibilizado para exportação, caracterizando o final do funcionamento da ferramenta.

3.2 MODELAGEM DO SISTEMA

Para ilustrar como a ferramenta foi modelada, será apresentado o diagrama de classe da mesma. Primeiramente será mostrada uma divisão do diagrama, separada em grupo, para facilitar a explicação e a visualização das classes como um todo, e depois, especificando cada um deles. O intuito desse capítulo não é detalhar a fundo o código da ferramenta, e sim mostrar classes e métodos que realizam as tarefas descritas no fluxo da ferramenta.

As classes da ferramenta podem ser divididas em três grupos:

- O grupo Principal, que possui a classe principal, que contém o método *main*, e algumas classes auxiliares de uso geral.
- O grupo Entradas, que trata todas as entradas, bem como modela os parâmetros, além de fornecer *input* ao grupo Núcleo, explicado a seguir.
- O grupo Núcleo, que é o responsável por todas as tarefas principais realizadas durante o funcionamento da ferramenta.

A seguir, cada grupo será mostrado com um pouco mais de profundidade.

3.2.1 GRUPO PRINCIPAL

Na figura 3.2, pode ser visto uma visão mais detalhada do grupo.

A classe Ferramenta é a que contém a função *main*, nela também é criado o link simbólico para o diretório /bin com o intuito de criar o comando "TheONE". Nesse grupo existem também as interfaces Dtn e Relatório, responsáveis por chamar os códigos em C referentes à modelagem e ao tratamento estatístico, respectivamente. Já a classe Serializer guarda os cenários já executados para, caso o usuário deseje, poderem ser visualizados no futuro, para isso, basta usar o método *"deserialize"* com o objeto. Os métodos que compõem essas classes são chamados em vários lugares do código.

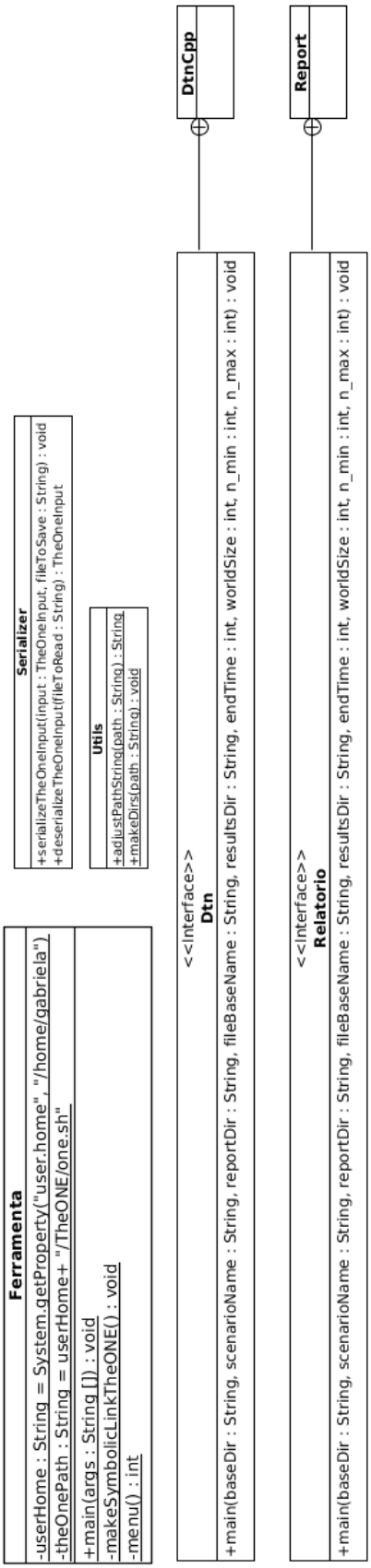

FIG. 3.2: Diagrama de Classes (Grupo Principal)

3.2.2 GRUPO ENTRADAS

O grupo Entradas é bem simples, sendo composto por uma única classe chamada TheOneInput, mostrada na figura 3.3, que é a responsável por definir os parâmetros configuráveis, da simulação e da modelagem, tanto os que terão somente um valor fixo quanto os que terão um intervalo de variação. Existem vários outros que possuem valor padrão imutável e não necessitam ser definidos a cada simulação, como os que definem os relatórios gerados pelo simulador. Nessa classe também é feita a breve validação de tipagem dos parâmetros. Essa classe fornece *inputs* para a classe Scenario, vista na seção seguinte.

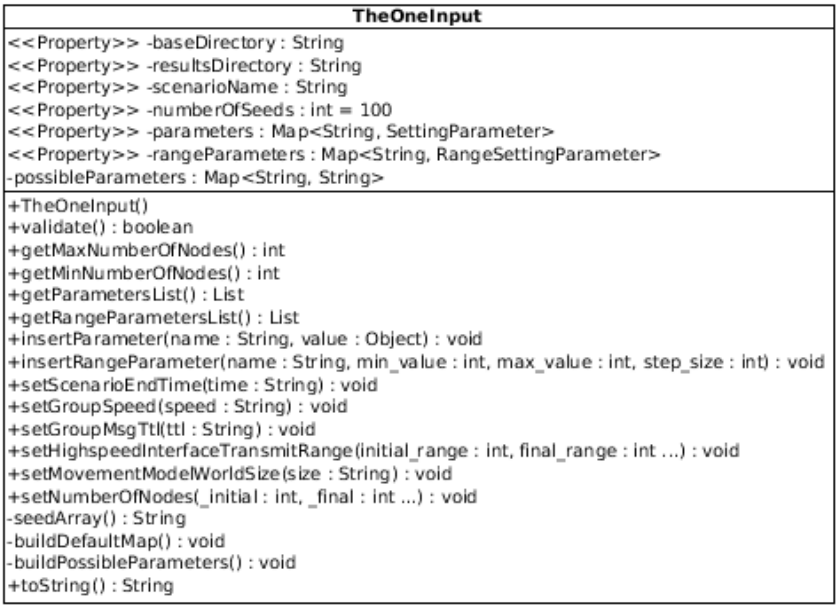

FIG. 3.3: Diagrama de Classes (Grupo Entradas)

3.2.3 GRUPO NÚCLEO

O grupo mais importante da ferramenta, que executa todas as suas funcionalidades, pode ser visto na figura 3.4. Sua principal classe é a Scenario. Cada cenário possui um objeto dessa classe, e ela é responsável por integrar todas as atividades da ferramenta, como rodar a simulação, o tratamento estatístico e a modelagem para aquele cenário específico. Outros métodos são importantes na classe Scenario, como o que junta os parâmetros com e sem intervalos, e o que salva o cenário (com a ajuda da classe *serializer* no grupo Principal). Note que um de seus parâmetros é um objeto de TheOneInput, ou seja, é aqui que acontece a comunicação com a clase do grupo Entradas.

FIG. 3.4: Diagrama de Classes (Grupo Núcleo)

A classe SettingsFile é a que gera o arquivo de configuração para o cenário específico, a partir dos parâmetros do objeto da classe Scenario. Um dos seus *inputs* é outra classe, chamada SettingParameter, que modela cada parâmetro do cenário, e possui dois filhos, que só existem pois possuem o método *toString* diferente. Se o parâmetro possuir um intervalo, como o alcance do nó no exemplo do capítulo passado, ao invés de ser tratado por SettingParameter, é tratado por RangeSettingParameter. Esse tratamento diferenciado é importante na hora de juntar os dois tipos de parâmetros e de formar os nomes dos arquivos gerados.

3.3 INTERFACE GRÁFICA

A interface gráfica da ferramenta de integração é a porta de entrada dos parâmetros fornecidos pelo usuário, bem como a porta de saída para os resultados gerados por aquele grupo de parâmetros. Ela não é parte fundamental da ferramenta, que ainda pode ser executada pela linha de comando direto na máquina que a hospeda, porém algumas funcionalidades se perdem ao optar pela não utilização da interface.

As principais funcionalidades que a interface agrega à ferramenta são:

- Visualização dos cenários: Para cada grupo de parâmetros será possível saber quantos cenários foram executados corretamente, quantos falharam e o quantos faltam ser executados;
- Validação dos parâmetros: Apesar de existir validação dentro do código da ferramenta, ela será extendida na interface, abrangendo uma gama maior de verificações. Isso ajudará a garantir a consitência das análises e o bom funcionamento da ferramenta;
- Gráficos *on-demand*: Os gráficos serão gerados segundo a necessidade do usuário, que escolherá um cenário específico para obter os gráficos. A única informação que precisa ser fornecida é o tipo de formato do gráfico, eps ou pdf.

Na próxima seção, será abordada as especifições técnicas da interface, que abrange todas as tecnologias usadas para seu correto funcionamento. Posteriormente, será detalhado o funcionamento da interface, como ela se integra com a ferramenta e como ela facilita a vida do usuário.

3.3.1 ESPECIFICAÇÕES TÉCNICAS

A interface gráfica escolhida foi uma interface Web, escrita na linguagem Python, usando o *framework* de desenvolvimento web Django e o banco de dados relacional MySql. A escolha da interface na internet se deve ao fato de que ela funciona não importa onde a ferramenta esteja hospedada, bastando saber o endereço IP da máquina que a hospeda.

O *framework* Django foi escolhido devido a sua rica biblioteca de validação, facilidade de utilização e o fato de possuir um servidor interno para testes. O servidor HTTP que será utilizado quando a ferramenta for disponibilizada na internet, é o Nginx, que é um servidor leve e de fácil implementação, e satisfaz as necessidades da ferramenta.

Idealmente, o servidor web deveria ficar na mesma máquina que a ferramenta, para facilitar sua execução a partir da interface. Como isso nem sempre é possível, utiliza-se Fabric, que é uma biblioteca em Python usada para chamar remotamente comandos e rotinas. O Fabric também permite o download de arquivos remotamente, funcionalidade que será utilizada largamente no momento de disponibilizar os resultados de cada cenário.

O funcionamento da ferramenta é bem demorado, chegando a várias horas, o que geraria um *timeout* no servidor web, pois o mesmo ficaria esperando o resultado da ferramenta, que foi chamada remotamente. Para contornar esse problema, o Fabric, ao fazer a chamada remota, a fará de maneira assíncrona, através do Celery, que é um gerenciador de tarefas baseado na passagem distribuída de mensagens.

Cada cenário é enviado assíncronamente, e uma vez terminado, outro é enviado. É esse gerenciamento de tarefas que é feito pelo Celery, e conta com a ajuda de um banco de dados não-relacional chamado Redis, utilizado para armazenar todas as tarefas que ainda não foram executadas.

3.3.2 FUNCIONAMENTO

Primeiramente o usuário fornece os parâmetros necessários para rodar a simulação e modelagem. A interface fará todas as validações necessárias, e a primeira é a de tipagem, que é feita naturalmente pelo navegador. A segunda validação é verificar se o valor de cada parâmetro está dentro de um intervalo aceitável, por exemplo, não faz sentido um tempo de simulação negativo ou muito alto, como 100 horas. A terceira validação é feita para aqueles parâmetros que possuirão um intervalo de variação, e será a checagem do passo, para garantir que haja uma partição inteira. Essa ultima validação garante que o usuário não escolha uma parâmetro que varie de 10 a 20 com passo 3, por exemplo.

Após a validação, a interface é levada à página do grupo de cenários. É nessa página

que é possível rodar a ferramenta, bem como visualizar os status do cenário. Existem quatro estados:

- WAITING: Estado inicial, significa que a ferramenta ainda não foi rodada, é o estado que aparece assim que o grupo de cenário é criado;
- PENDING: Significa que o comando foi mandado ao Celery, porém ainda está pendente na fila de tarefas;
- SUCCESS: Significa que a ferramenta rodou corretamentamente para aquele cenário;
- FAILURE: Ferramenta apresentou algum problema ao ser rodada para aquele cenário.

A figura 3.5 ilustra como esses estados se relacionam:

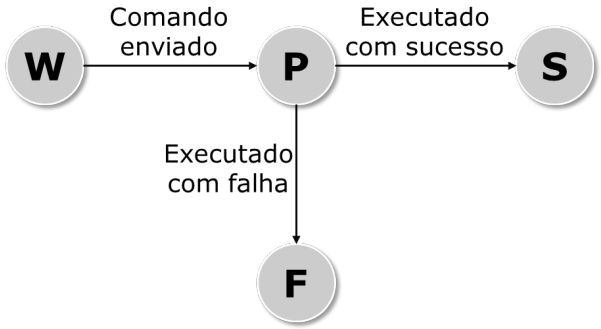

FIG. 3.5: Diagrama de Estados de um Cenário

Quando o usuário escolhe rodar a ferramenta, a interface monta todos os comandos que serão passados remotamente pelo Fabric. Como somente um cenário é executado por vez, será feito um comando para cada cenário, e são esses comandos que efetivamente rodam a ferramenta de integração. O Fabric, utilizando o Celery, chama a ferramenta assíncronamente, para todos os cenários de uma vez. O Celery, juntamente com o Redis, faz o gerenciamento de todos os comandos, chamando um de cada vez. Ao fim de cada execução, a ferramenta retorna o status, permitindo assim, saber se houve alguma falha na execução bem como os cenários que ainda faltam ser rodados. Quando não restar mais nenhum cenário com o status "PENDING", significa que todos já foram executados.

Na página do grupo de cenários, é possível clicar em cada cenário para ver a tela de detalhes de um cenário. Nessa tela, há todas as informações do cenário, como os parâmetros que o caracterizam, o grupo a que ele pertence e seu *status*. Nessa tela também é possível exportar os resultados desse cenário. Para isso, seleciona-se qual formato de

visualização dos gráficos e clica-se no botão *Download*. Um arquivo ZIP compactado é então disponibilizado, contendo todos os resultados daquele cenário.

A interface também provê uma lista com todos os grupos de cenários já rodados. Ela é gerenciada usando o banco de dados MySql já citado, que armazena todos os parâmetros de cada grupo de cenário. Essa lista alavanca o tempo do usuário caso ele deseje visualizar um grupo de cenários novamente ou obter resultados de cenários que ele ainda não exportou. A página que contém a lista também permite uma organização dos grupos de cenários rodados, evitando execuções duplicadas.

A figura 3.6 apresenta o fluxo de telas descrito nesse capítulo.

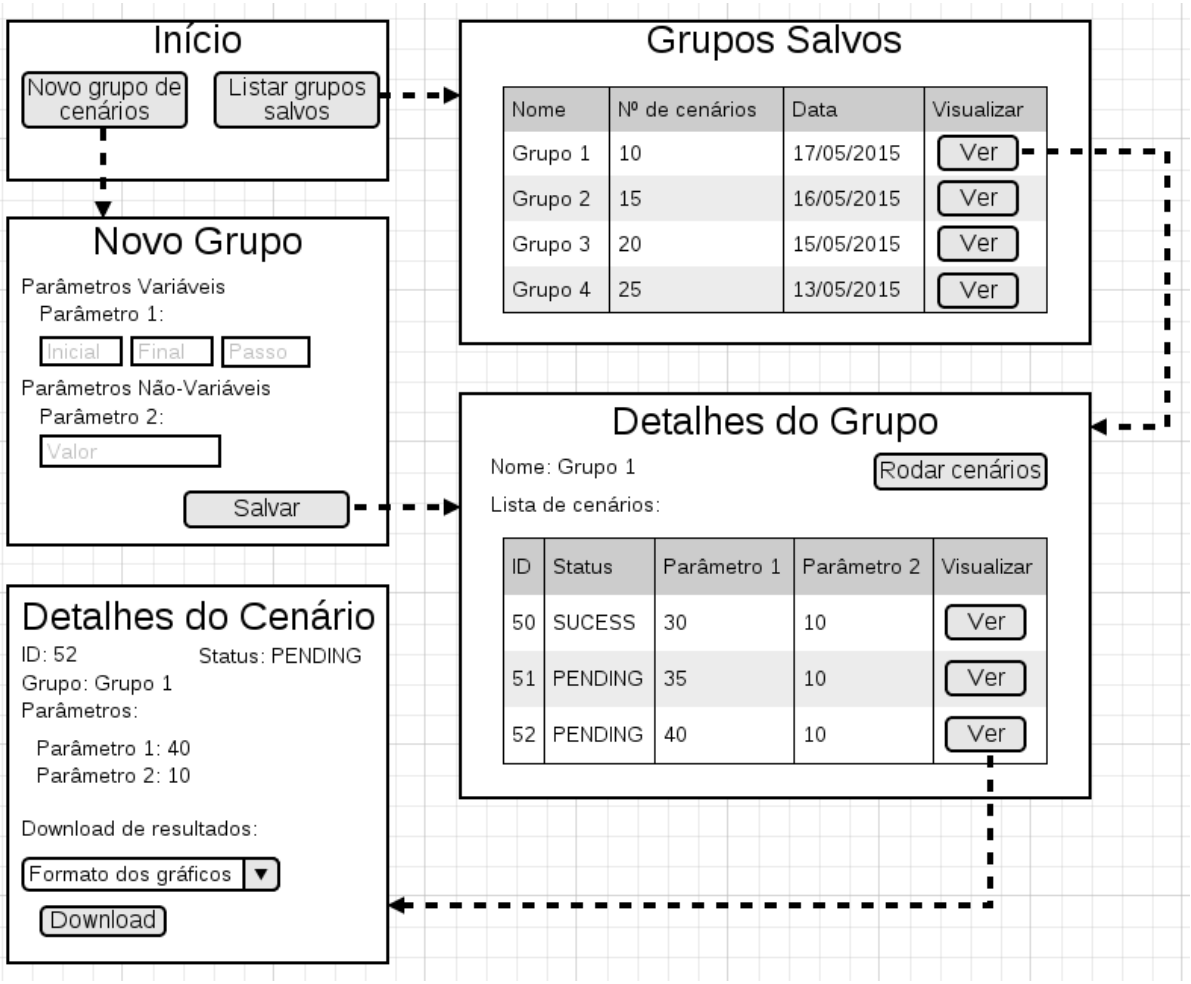

FIG. 3.6: Fluxo de Telas

No próximo capítulo poderá ser visto um exemplo onde cada tela do fluxo é mostrada, simulando o funcionamento real da ferramenta.

3.4 VALIDAÇÃO DOS RESULTADOS

Como parte fundamental para comprovar o bom funcionamento da ferramenta de integração desenvolvida nesse projeto, foi rodado o mesmo cenário tanto manualmente, pelo método antigo, quanto automaticamente, pela ferramenta, e seus resultados foram comparados. O cenário utilizado foi o seguinte: Número de nós variando de 10 a 200 com passo 5, utilizando um raio de transmissão de 20m em uma área de 4km^2, com um tempo de simulação de uma hora. Os demais parâmetros, descritos na seção anterior, possuem os mesmos valores do exemplo.

Um gráfico de "Nós disponíveis na Modelagem" com a dimensão de tempo em segundos, mostrado na figura 3.7, foi construído para os dois métodos. Pode-se constatar, por meio do gráfico, que os resultados gerados pela ferramenta são equivalentes aos gerados pelo método manual.

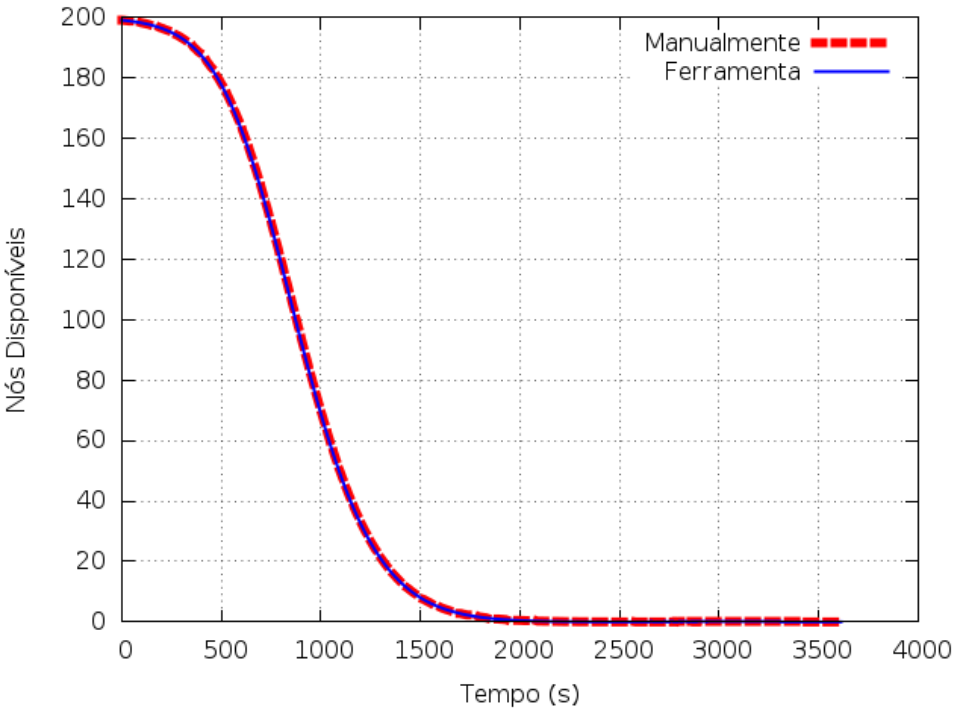

FIG. 3.7: Gráfico comparativo de "Nós disponíveis na Modelagem"

Note que as curvas se sobrepoem, mostrando que não existe diferença entre os resultados comparativos obtidos a partir da ferramenta ou manualmente. Ainda mais, o gráfico mostra que a ferramenta não interfere de maneira alguma no funcionamento da simulação ou da modelagem.

4 EXEMPLO DE USO

Nesse capítulo serão apresentados dois exemplos de uso da ferramenta. Um sem o uso da interface gráfica, por meio da linha de comando do Linux, e outro com o uso da interface gráfica.

Os parâmetros do cenário que utilizaremos como exemplo serão os seguintes:

a) Número de nós variando de 20 até 30, com passo 5;

b) Alcance dos nós de 50m;

c) O tamanho da área disponível para movimentação dos nós será de 300 x 300 m^2;

d) Cada nó terá um *buffer* de 50 MB;

e) A velocidade mínima do nó é 4m/s e a máxima é 10m/s;

f) O TTL da mensagem em cada nó é de 2 minutos;

Além dos parâmetros que configuram o grupo de cenários, existem os parâmetros que definem a simulação:

a) A simulação tem 100 sementes, ou seja, será executada 100 vezes, variando a posição inicial e a movimentação dos nós;

b) O tempo total de cada simulação é de 200 segundos;

c) O nome do grupo de cenários é "exemplo";

d) O intervalo de atualização da posição dos nós é de 1s, ou seja, a cada segundo, a posição é atualizada.

A seguir, será apresentado como se dá o uso da ferramenta pela linha de comando e pela interface gráfica, utilizando os parâmetros que escolhemos.

4.1 USO PELA LINHA DE COMANDO

Primeiramente, o usuário deve entrar na pasta onde a ferramenta está, que deve ser a mesma pasta do The ONE. No nosso caso, a ferramenta estava salva no formato .jar, sob o nome "pfc.jar". Também é necessário passar como parâmetro para a ferramenta as pastas onde serão salvos os resultados da simulação e da modelagem. Todos os parâmetros devem ser passados na ordem que a ferramenta espera. A figura 4.1 mostra o comando completo sendo executado. Note que, ao rodar o comando, foi definida uma variável de ambiente chamada LD_LIBRARY_PATH. Ela é necessária para que o JNA saiba onde encontrar os programas em C da modelagem.

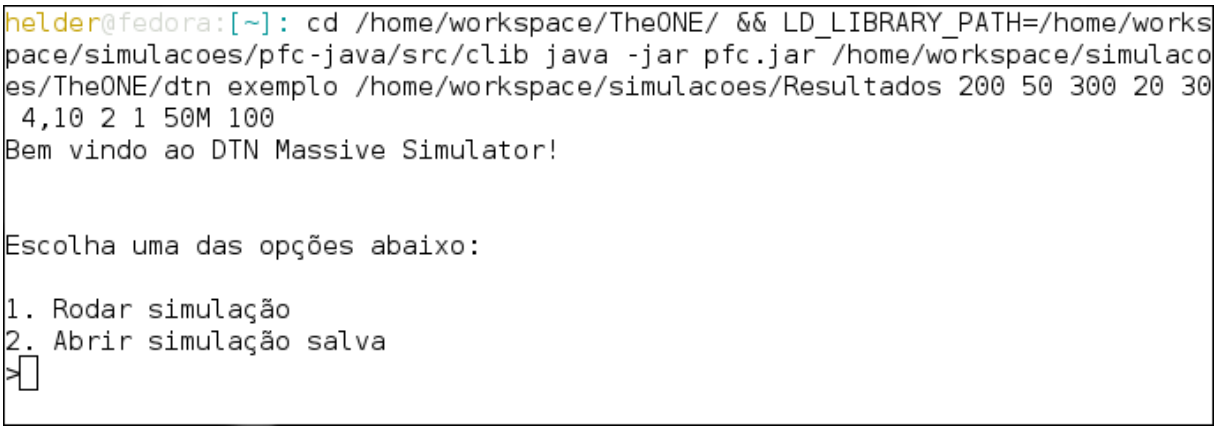

FIG. 4.1: Execução da ferramenta na linha de comandos do Linux

Após executar o comando, a ferramenta pergunta se deseja-se rodar a simulação, ou abrir uma simulação salva. Escolhendo a primeira opção, a simulação e a modelagem serão executadas automaticamente, bastando ao usuário, ao final da execução, conferir os resultados na pasta especificada por ele. Se for escolhida a segunda opção, ele listará o nome de todas as simulações que já foram rodadas por meio da linha de comando, e carregará aquela que for escolhida, listando seus parâmetros, e dando a opção de que algum deles seja alterado e o cenário modificado seja rodado novamente.

4.2 USO PELA INTERFACE GRÁFICA

Para utilizar a ferramenta por meio da interface gráfica, primeiramente o usuário deve abrir um navegador e entrar no endereço onde a interface web está rodando. No caso do nosso exemplo, ela estava rodando localmente na porta 8000. Ao entrar, será apresentada a tela inicial, com duas opções, como mostra a figura 4.2.

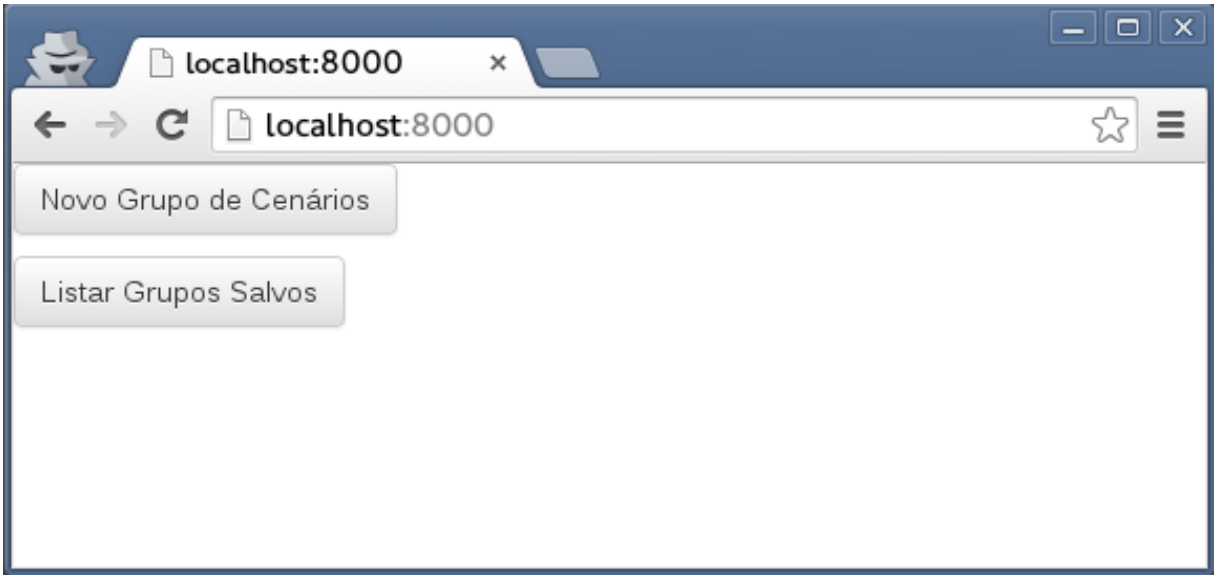

FIG. 4.2: Tela inicial da interface gráfica

Para rodar a nossa simulação, escolhemos a opção "Novo Grupo de Cenários". Um formulário é apresentado, pedindo os parâmetros do grupo de cenários a ser executado. Para demonstrar a capacidade que a ferramenta tem de executar vários cenários simultaneamente, faremos com que o alcance dos nós varie entre 50m e 70m, com passo de 5m, e o tempo de simulação varie entre 200 segundos e 500 segundos, com passo de 100 segundos. A figura 4.3 apresenta duas telas que mostram alguns dos parâmetros do formulário já preenchidos com os valores escolhidos. Note que, se o usuário não deseja que um parâmetro varie, basta preencher apenas o campo "Inicial" referente a ele, deixando os campos "Final" e "Tamanho do Passo" em branco.

Ao salvar o grupo de cenários que criamos, seremos direcionados para a tela de detalhes do grupo, mostrada na figura 4.4. Nela, pode-se ver que todas as combinações entre os parâmetros variáveis que definimos foram feitas, e para cada combinação foi criado um cenário. Nessa tela, clicando no botão "Rodar Simulações", a ferramenta irá automaticamente rodar todos os cenários desse grupo em sequência, e na coluna "Status" o usuário poderá ver quais simulações já rodaram, quais falharam e quais ainda estão pendentes.

Ainda na tela de detalhes do grupo, existe um botão ao lado de cada cenário da tabela, que serve para entrar na tela de detalhes do cenário, que é mostrada na figura 4.5. É nessa tela que o usuário poderá fazer o *download* dos resultados do cenário, quando a execução do mesmo já tiver sido concluída.

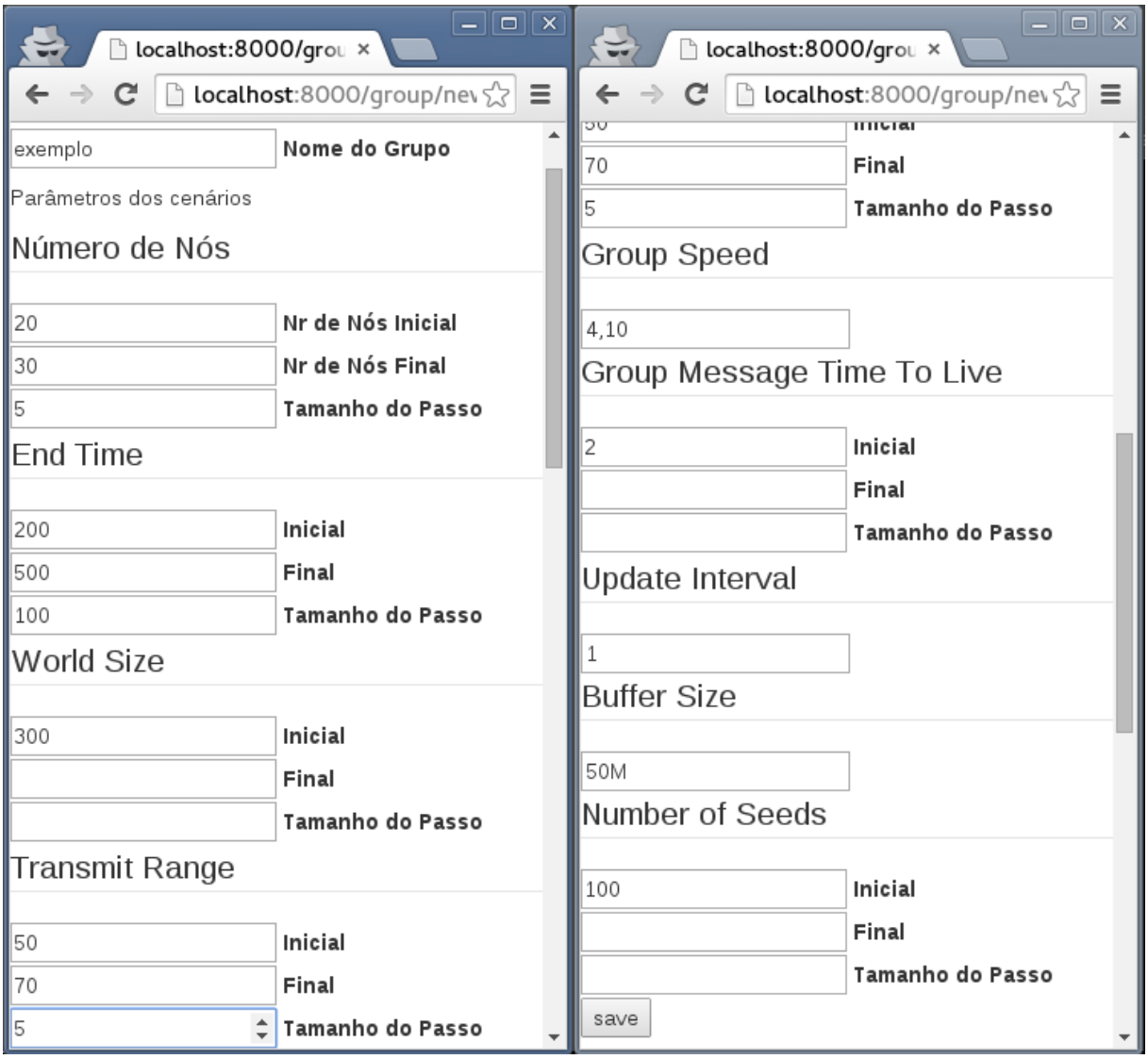

FIG. 4.3: Formulário de criação de cenários

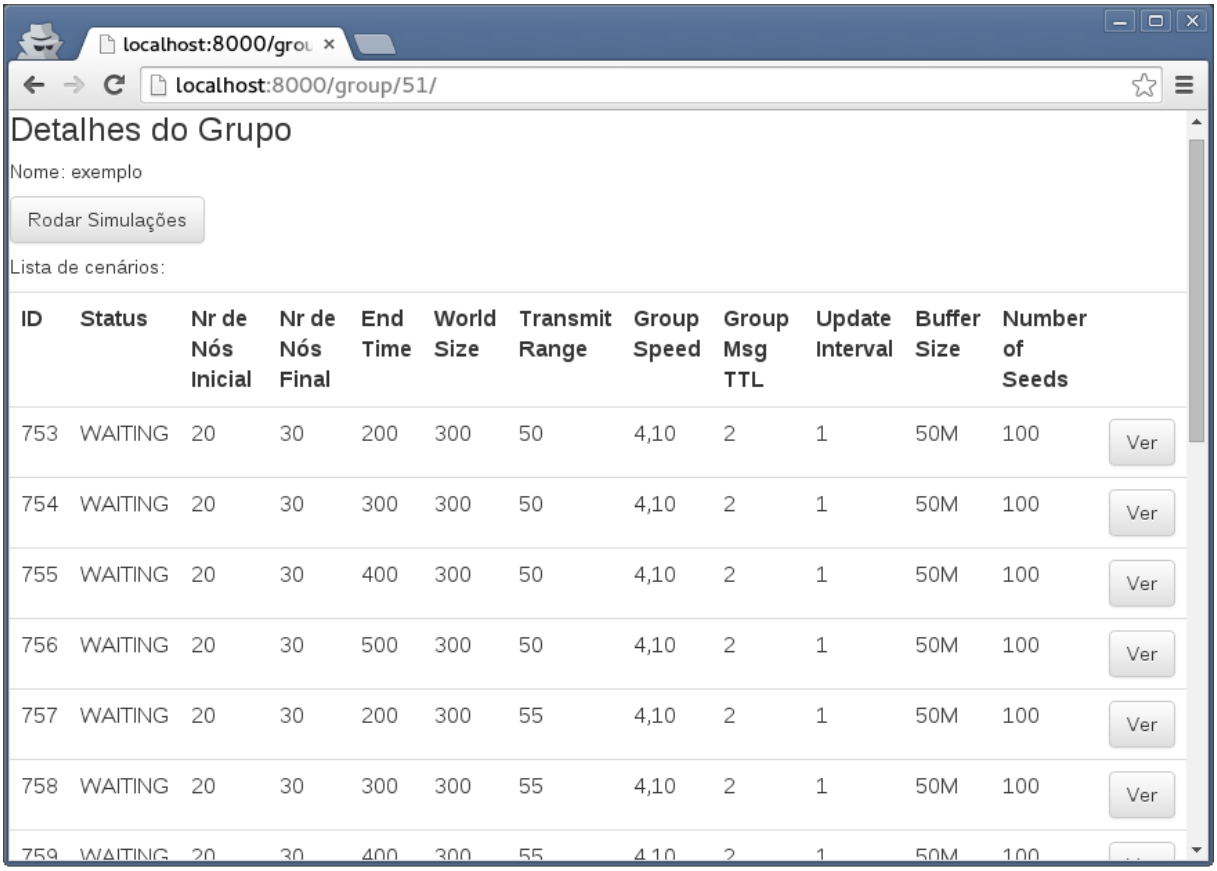

FIG. 4.4: Tela de detalhes de um grupo de cenários

FIG. 4.5: Tela de detalhes de um cenário

5 CONCLUSÃO

É evidente que o estudo de redes de computadores está sempre em voga nos centros de pesquisa ao redor do mundo. Novas redes surgem com as mais diferentes finalidades, e uma delas é a rede DTN, tolerante a atrasos e interrupções. Essa rede em particular tem bastante potencial, por ser uma rede capaz de comunicação mesmo sem conexão fim-a-fim, não fazendo uso de uma infraestrutura fixa. Tais redes despertam estudos que têm como finalidade entender seu comportamento e desenvolver protocolos para que possam ser utilizadas. O estudo da modelagem é essencial para atingir o primeiro objetivo de conhecimento comportamental, e, visando isso, muitos estudos são feitos na área. A análise de resultados deve ser o foco desses estudos, porém muitas vezes a obtenção de resultados é uma tarefa tão difícil quanto. Visando esse problema, esse projeto propôs uma ferramenta justamente para simplificar, de maneira eficaz, essa obtenção, deixando ao pesquisador somente a parte que importa, a análise.

A obtenção de resultados, em certos casos, pode demorar mais de 24 horas, e somente para um cenário. Esse fato faz com que o número de análises não seja muito alto, prejudicando o resultado final da pesquisa. A proposta da ferramenta é automatizar esse processo, dando a possibilidade de obter resultados de vários cenários simultaneamente, da forma que o pesquisador achar melhor. Nesse contexto, as limitações do número de análise praticamente desapareceriam, sendo definido pelas necessidades do pesquisador e não por outros motivos.

Com a ferramenta, o usuário atualmente é capaz de rodar simulações para cenários onde os parâmetros variam de acordo com um intervalo de valores passado, e não apenas um valor de cada vez. Desse modo ele pode otimizar seu tempo, pois com apenas uma entrada de dados ele roda simulações para vários cenários. Para cada cenário simulado, a ferramenta também faz, automaticamente, o tratamento estatístico dos dados e a modelagem, disponibilizando os resultados no formato texto.

A interface gráfica provê um ambiente gráfico, que é conhecido ao usuário, uma página na Internet. Esse fator pode facilitar o uso da ferramenta para usuários mais leigos. A interface também traz ao usuário uma solução gráfica de organização dos cenários rodados, em contrapartida aos inúmeros diretórios abarrotados de arquivos. A validação mais concisa e a obtenção de resultados de forma integrada, já que é permitido obter todos

os resultados para um cenário específico de uma só vez, também otimizam o tempo do usuário.

A ferramenta, apesar de ser específica para um tipo de rede e modelagem (DTN e a epidêmica estocástica e determinística), otimiza bastante o tempo do pesquisador. Desenvolver novas versões, para novas redes e modelagens, poderia otimizar o tempo de um número maior de pesquisadores, o que teria bastante utilidade. Tal tarefa pode ser considerada uma ótima área de estudo futuro. Porém, antes deve-se refinar a ferramenta já desenvolvida, acrescentando um sistema de autenticação para aumentar a segurança. Outra possível melhoria seria avisar ao usuário quando todos seus cenários tiverem sido executados, mostrando quais falharam e quais foram sucedidos, por meio de um envio de e-mail, por exemplo.

6 REFERÊNCIAS BIBLIOGRÁFICAS

[1] DIAS, G. M. d. S. *Modelo Epidemiológico SIR Aplicado a Redes Tolerantes a Atrasos e Desconexões*. 134 p. Dissertação (Mestrado em Sistemas e Computação) — Instituto Militar de Engenharia, Rio de Janeiro, 2013.

[2] KERÅNEN, A.; OTT, J.; KÅRKKÅINEN, T. The one simulator for dtn protocol evaluation. In: . [S.l.]: ACM, 2010.

[3] DALEY, D. J.; GANI, J. *Epidemic Modelling: An Introduction*. [S.l.]: Cambridge Studies in Mathematical Biology, 2001.

[4] HETHCOTE, H. The mathematics of infectious diseases. *SIAM review*, JSTOR, p. 599–653, 2000.

[5] TANEMBAUM, A. S. *Redes de Computadores*. 5. ed. São Paulo: Pearson Prentice Hall, 2011. 583 p. Bibliografia: p. 376–378. ISBN 978-85-7605-924-0.

[6] KHABBAZ, M. J.; ASSI, C. M.; FAWAZ, W. F. Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges. *IEEE Communications Surveys & Tutorials*, v. 14, p. 607–640, 2012.

[7] KERäNEN, A.; OTT, J.; KäRKKäINEN, T. The ONE Simulator for DTN Protocol Evaluation. In: *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. New York, NY, USA: ICST, 2009. ISBN 978-963-9799-45-5.

7 APÊNDICES

7.1 APÊNDICE A

Este apêndice mostra um exemplo de arquivo de configuração utilizado como entrada para o simulador TheONE. Eles configura apenas alguns dos muitos parâmetros de configuração utilizáveis no TheONE.

```
Scenario.name = [name\_20\_1;name\_20\_2;...;name\_20\
    _99;name\_20\_100]
Scenario.simulateConnections = true
Scenario.updateInterval = 1
Scenario.endTime = 100
Scenario.nrofHostGroups = 1
highspeedInterface.type = SimpleBroadcastInterface
highspeedInterface.transmitSpeed = 1M
highspeedInterface.transmitRange = 50
Group.movementModel = RandomWaypoint
Group.router = EpidemicRouter
Group.bufferSize = 50M
Group.waitTime = 0, 10
Group.nrofInterfaces = 1
Group.interface1 = highspeedInterface
Group.speed = 4, 10
Group.msgTtl = 2
Group.resetTtl = true
Group.noRepeatedReceive = true
Group.nrofHosts = 20
Group1.groupID = p
Events.nrof = 1
Events1.class = ExternalEventsQueue
Events1.filePath = dtn/1mensagem.txt
MovementModel.rngSeed = [1;2;3;...;99;100]
MovementModel.worldSize = 600, 600
Report.nrofReports = 6
Report.warmup  = 0
```

```
Report.reportDir = /home/gabriela/TheONE/dtn/Relatorios/
    name\_reports/name\_20/
Report.report1 = CommunicationStatsReport2
Report.report2 = TotalEncountersReport
Report.report3 = ContactTimesReport
Report.report4 = InterContactTimesReport
Report.report5 = TotalContactTimeReport
Report.report6 = EventLogReport
Optimization.cellSizeMult = 5
Optimization.randomizeUpdateOrder = true
GUI.EventLogPanel.nrofEvents = 100
```