UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
TECNÓLOGO EM REDES DE COMPUTADORES

JOAO ANTONIO FAUSTINO FREITAS

MALHA DE REDE SEM FIO COMO SUPORTE PARA AMBIENTE PARCIALMENTE UBÍQUO

QUIXADÁ
2015

JOAO ANTONIO FAUSTINO FREITAS

MALHA DE REDE SEM FIO COMO SUPORTE PARA AMBIENTE PARCIALMENTE UBÍQUO

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Tecnologia em Redes de Computadores da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Tecnólogo.

Área de concentração: computação

Orientador Prof. Msc Marcos Dantas Ortiz

QUIXADÁ
2015

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca do Campus de Quixadá

F936m Freitas, Joao Antonio Faustino
Malha de rede sem fio como suporte para ambiente parcialmente ubíquo / Joao Antonio Faustino Freitas. – 2015.
64 f. : il. color., enc. ; 30 cm.

Monografia (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Tecnologia em Redes de Computadores, Quixadá, 2015.
Orientação: Prof. Me. Marcos Dantas Ortiz
Área de concentração: Computação

1. Computação ubíqua 2. Redes de computadores 3. Redes locais sem fio I. Título.

CDD 004.6

JOAO ANTONIO FAUSTINO FREITAS

MALHA DE REDE SEM FIO COMO SUPORTE PARA AMBIENTE PARCIALMENTE UBÍQUO

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Tecnólogo em Redes de Computadores da Universidade Federal do Ceará como requisito para obtenção do grau de Tecnólogo.

Área de concentração: Computação

Aprovado em: _____ / Junho / 2015.

BANCA EXAMINADORA

_____

Prof. Msc. Marcos Dantas Ortiz (Orientador)
Universidade Federal do Ceará-UFC

_____

Prof. Dr. Arthur de Castro Callado
Universidade Federal do Ceará-UFC

_____

Prof. Msc. Michel Sales Bonfin
Universidade Federal do Ceará-UFC

Dedico a meu filho recém-nascido Erick Otávio, que foi a motivação para eu sempre continuar tentando ser uma pessoa melhor e ao meu avô Otávio Faustino (*in memoriam*), que apesar do pouco tempo de infância que passei com ele, aprendi a ser humilde, justo e nunca desistir perante a dificuldade alguma.

AGRADECIMENTOS

Agradeço primeiramente a Deus por me proporcionar conforto em momentos difíceis.

Ao prof. Marcos Dantas Ortiz, não só pela ajuda crucial na orientação desse trabalho, mas também pelo conhecimento passado a mim por ele com muita didática, objetividade e paciência nesses muitos anos de estudo.

A todos os professores da UFC com quem eu estudei e principalmente ao Professor Arthur Callado e Jeandro Mesquita, pois a maioria do tempo que passei na universidade foi tendo aula com eles.

Aos meus amigos da turma 2010.1, em especial os amigos Marcelo Miranda e Otacílio Aguiar, passamos muita coisa na universidade e nas chatas viagens diárias de Ocara a Quixadá.

Aos amigos da turma 2011.1 em especial as amigas Luclécia Correia e Atrícia Sabino.

A todos os meus familiares que acreditaram em mim, em especial, meu primo Sérgio que me ajudava de vez em quando, a minha avó Tereza (*in memoriam*), que foi uma pessoa importante na minha formação pessoal, me educou com rigor e disciplina quase militar e a minha tia Maria (*in memoriam*), que apesar de não conhecer nem uma letra do alfabeto, conhecia a importância dos estudos e estava sempre lembrado os horários de ir para escola, a minha mãe que na minha infância lia para mim toda a coleção de livros Vaga Lume me ensinado o gosto para leitura e a meu pai Zé Maia.

À Universidade Federal Do Ceará por ter me dado a oportunidade de ter minha formação profissional.

E por fim, agradecer em especial a minha esposa Natalia Faustino, por tentar me ajudar de todas as formas possíveis e impossíveis.

“Talvez não tenha conseguido fazer o melhor, mas lutei para que o melhor fosse feito.
Não sou o que deveria ser, mas Graças a Deus, não sou o que era antes”.
(Martin Luther King)

RESUMO

Este trabalho tem como objetivo simular uma malha de rede sem fio que forneça conectividade para um ambiente parcialmente ubíquo. A pesquisa se deve, pela falta de uma rede específica e que tenha baixo custo para suporte à computação ubíqua. Na simulação foram analisados os requisitos de conectividade constante e mobilidade que são dois dos principais para suporte e criação de um ambiente parcialmente ubíquo. Na simulação é definido que a conectividade constante sempre esteja atrelada a mobilidade. Espera-se que o resultado deste trabalho acadêmico possa ser útil para estudo e simulação de ambientes parcialmente ubíquos. Para o cenário de testes foi criada uma topologia de malha de rede sem fio rodando o *Hibrid Wireless Mesh Protocol*. O ambiente de simulação utilizou o *Network Simulator* 3 e o *bonnmotion* para suporte à mobilidade. No NS3 não existem módulos para simular uma rede desta magnitude e complexidade com perfeição, porém pode ser criada uma rede com características semelhantes a rede real que possam gerar os resultados necessários para satisfazer a proposta deste trabalho. Os problemas encontrados na criação deste trabalho foram as limitações do simulador relativas à conexão transparente do usuário móvel. Os testes de validação foram feitos baseados nas métricas: tempo de conexão do nó móvel na rede, tempo de simulação, troca de pacotes de conexão, distância que o usuário móvel percorreu. Os resultados coletados foram inseridos em uma base estatística para análise. Os testes mostraram que a rede simulada pode oferecer conexão constante e mobilidade necessárias para criar um ambiente parcialmente ubíquo.

Palavras chave: Malha de rede. Ambiente Ubíquo. Redes Sem Fio.

ABSTRACT

This study aims to simulate a wireless mesh network that provides connectivity to a ubiquitous environment. The research is due the lack of a specific network and has low cost to support ubiquitous computing. In the simulation, we analyzed the constant connectivity requirements and mobility are two of the main requirements needs to support and create a ubiquitous environment. In simulation, set that constant connectivity linked is always to mobility. Is expected that the outcome of this academic work may be useful for studding and simulating ubiquitous environments. For the scenario, testing created was a wireless mesh network topology running the Hybrid Wireless Mesh Protocol. The simulation environment used the Network Simulator 3 and Bonnmotion to support mobility. In NS3, there are no modules to simulate a network of this magnitude and complexity to perfection, but it can create a network with characteristics similar to real network that generate the results necessary to satisfy the purpose of this work. The problems encountered in the creation of this work were the limitations of the simulator relative to the transparent connection of the mobile user. Validation tests were based on metrics: the mobile node on the network connection time, simulation time, switching the connection packages exchange, distance traveled by the mobile user. Our results were inserted on the following a statistical basis for analysis. The tests showed that the simulated network can offer constant connection and mobility necessary to create a ubiquitous environment.

Keywords: mesh network. Ubiquitous environment. Wireless Networks

LISTA DE FIGURAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

Em 1991, Mark Weiser[1] mencionou o termo computação ubíqua[2] e pela primeira vez expôs suas características (BOLSONI, 2009). Ele descreveu um ambiente computacional onde a interação de homem e máquina seria tão comum, quase transparente. A conexão era constante e onipresente, permitindo assim que todos os dispositivos da rede se conectassem dentro do contexto.

Muitos achavam que esse tipo de tecnologia não passava de utopia. Na época, essa afirmação era certa devido às limitações da tecnologia. As limitações eram em tecnologias de *hardware* na área de redes de computadores, juntamente com dispositivos que se conectassem dentro do contexto, trocando constantemente informações criando um ambiente saturado. O fator principal era a existência de uma rede que fosse capaz de suportar este ambiente caótico, onde muita informação pudesse ser processada e envidada e que principalmente fornecesse conexão constante e mobilidade aos usuários moveis (SANATAN, 2009; HAZRA, 2015).
Atualmente, os avanços de *hardware*, na parte de processamento, de roteamento, e na tecnologia dos enlaces de rede tornam possível a criação de um ambiente parcialmente ubíquo. Em um ambiente parcialmente ubíquo, por exemplo, existe a necessidade de intervenção humana para efetuar a conexão em nível de usuário (camada de aplicação). Já em um ambiente totalmente ubíquo a intervenção humana para efetuar a conexão no nível de usuário é quase nenhuma. Na maioria dos casos existem dificuldades para criação de um ambiente totalmente ubíquo. Estas dificuldades são: falta de capacidade de expansão, dificuldade de manutenção, evolução da rede e necessidade de alto desempenho no tráfego de dados, roteamento e gerenciamento de usuários móveis, Dessa forma, ainda não é possível atingir o mundo previsto, há mais de uma década, pelos pioneiros da Computação Ubíqua (BOLSONI, 2009; SANATAN, 2009; HAZRA, 2015). Portanto, este trabalho investiga o uso de malhas de rede sem fio para criação de um ambiente com características parcialmente ubíquas.

O *WiFi* (IEEE 802.11s padrão *wireless* desenvolvido pelo *task group*) é o padrão mais indicado para criar uma malha de rede, pelo seu bom alcance e baixo custo. O protocolo mais indicado para criação desse ambiente é o HWMP *(Hybrid Wireless Mesh Protoco*l, IEEE 802.11s), por se tratar de um protocolo de controle criado especialmente para malhas de rede e por conseguir trabalhar com outros protocolos de roteamento se necessário, pode atenuar o problema da demanda de alto desempenho no tráfego de dados, roteamento e gerenciamento de

[1] Cientista chefe do Centro de Pesquisa Xerox PARC, é Considerado o pai da computação ubíqua.
[2] Ambientes saturados de dispositivos computacionais e redes de comunicação sem fio, que se integram naturalmente à atividade humana, e troncam informações constantemente.

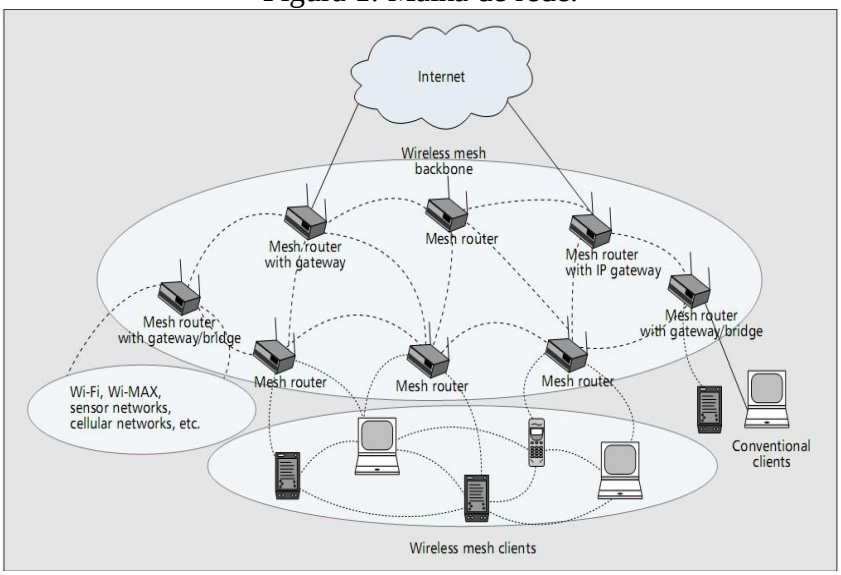
usuários móveis. As malhas de rede são descentralizadas, se auto organizam e são capazes de formar uma rede de comunicação confiável e robusta (ANDREV, 2010; VULRAL, 2013; HAZRA, 2015).

> neste tipo de rede, roteadores sem fio se comunicam uns com os outros no modo ad hoc utilizando múltiplos saltos para encaminhar mensagens ao destino apropriado. Redes mesh […] são redes comunitárias baseadas em algoritmos de roteamento cooperativos, tais quais os encontrados em redes sem fio *ad-hoc* [...]. Essas redes evoluíram a partir das redes móveis *ad-hoc (Mobil Ad-hoc Networks, ou MANETs*). Usuários finais se conectam aos pontos de acesso da rede *mesh* através rede Ethernet cabeada ou rede sem fio IEEE 802.11. (SAADE, 2007 p.1).

No trabalho proposto, o usuário terá uma cobertura sempre presente dentro dos limites da rede com possibilidade de trocar informação com qualquer ponto de acesso. A malha de rede será criada por pontos sem fios, que se comunicam através de sinal de rádio 802.11s. O protocolo HWMP define como cada nó deve interagir dentro da rede maior. Os pacotes são roteados pelo caminho mais seguro e rápido como base no roteamento dinâmico. Em uma malha de rede sem fio, somente um ponto precisa estar fisicamente ligado à *internet,* conforme Andreev (2010).

A Figura 1 ilustra uma malha de rede para compartilhamento de *Internet*, com vários dispositivos conectados. Nela são mostrados padrões de rádio interoperáveis. Apenas um ponto se liga diretamente à *Internet* e compartilha a conexão com um grupo de pontos próximos, e um desses pontos pode compartilhar conexão com um novo grupo de pontos.

Figura 1: Malha de rede.



Fonte: Adaptada de Fontoura (2007).

A figura 2 ilustra como podem ficar distribuídos os nós sem fio em uma área como um pequeno bairro. Cada ponto de acesso 802.11s precisa estar ligado a uma fonte de energia, pode ser uma fonte AC ou painéis solares, se estiver ao ar livre, onde os quais devem ser

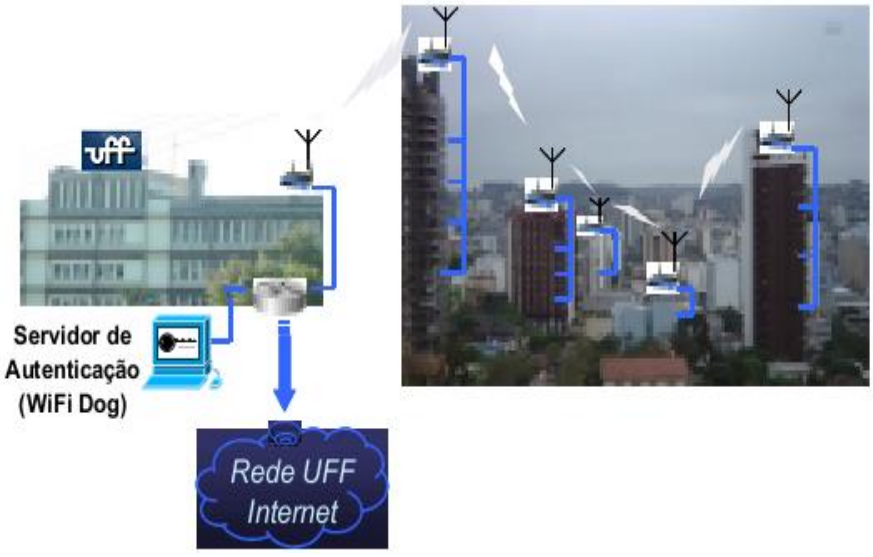
impermeabilizados. A eficácia da conectividade e compartilhamento da Internet se dá pelo número de nós da rede, Vural (2013).

Figura 2:Pequeno bairro ligado com malhas de rede.

Fonte: Adaptada de Breuer (2004).

A Universidade Federal Fluminense (UFF) vem usando desde 2006 essa tecnologia para fornecer acesso à comunidade universitária que mora nas proximidades. A Figura 3 ilustra a topologia de rede usada na UFF, que consiste em roteadores instalados no topo dos prédios. Os roteadores formam uma malha de rede sem fio que faz autenticação de usuário com o *software wifidog*[3] para autenticar quem deseja se conectar à rede, conforme Saade (2007)

Figura 3: Projeto Remesh.



Fonte: Saade (2007)

[3] Servidor de autenticação de usuários da malha de rede desenvolvido pela Universidade Federal Fluminense. (SAADE, 2007).

2 OBJETIVOS

2.1 Objetivos Gerais

Verificar se as malhas de rede sem fio podem prover características fundamentais de um ambiente parcialmente ubíquo, que são mobilidade com conectividade constante.

2.2 Objetivos Específicos

Simular no *Network Simulator 3* (NS3) uma malha de rede sem fio(802.11s). Validar no NS3 as necessidades fundamentais dos usuários móveis construir um ambiente parcialmente ubíquo sobre redes *mesh.*

3 TRABALHOS RELACIONADOS

Este trabalho faz uma análise do uso de malhas de rede para criar um ambiente parcialmente ubíquo através da observação em ambiente simulado dos seguintes requisitos de ambiente parcialmente ubíquo: conexão constante, em que o usuário deverá ter conexão constante dentro de toda amplitude da rede; e mobilidade, em que será possível que o usuário se movimente dentro de todo o campo da rede.

Existem muitas limitações para criação real da topologia proposta nesse trabalho, como por exemplo: a autenticação do servidor *wifidog,* a conexão com *internet* de alta velocidade, os custos financeiros para compra de equipamentos e implementação da rede em área geográfica extensa e necessidade de alto desempenho no tráfego de dados, roteamento e gerenciamento de usuários móveis. Por esse motivo, o cenário de avaliação foi construído e executado através do simulador de redes *Network Simulato*r 3 (NS-3). Neste simulador, apesar das limitações de processamento e alta abstração, é possível criar e simular o que está sendo proposto no trabalho, devido aos seus módulos nativos e pela possibilidade de trabalhar com outras ferramentas como o *Bonnmotion* (RODRIGUES, 2009; ASCHENBRUCK, 2010; SIEGA, 2013; HAZRA, 2015).

Muitos pesquisadores apontam a computação ubíqua como um dos maiores desafios de hoje. Todo tipo de análise e projeto de pesquisa que venha a contribuir para o campo da computação ubíqua são úteis. Atualmente, ainda não se tem uma rede específica para dar suporte aos padrões ubíquos. Roos (2008) apresenta um trabalho que propõe utilizar as malhas de redes sem fio para fornecer *internet* de qualidade (alta velocidade) para grandes áreas metropolitanas. Vural (2013), faz um levantamento sobre as malhas de rede, para verificar a possibilidade da mesma fornece um ambiente ubíquo completo em grande escala. Resultado obtido em seu trabalho é que existe a necessidade estudos aprofundado em áreas específicas da tecnologia ubíqua, como por exemplo, protocolos de roteamento, mobilidade, desempenho e escalabilidade de rede, sendo assim Vural (2013), foca seu trabalho no roteamento.

Este trabalho não tem a pretensão de abordar, nem solucionar todos os problemas das tecnologias citadas no mesmo, porém, pretende apontar uma alternativa para suprir duas das necessidades do cenário ubíquo, que são mobilidade e conexão constante. Nesse contexto, esta pesquisa pretende contribuir no campo da tecnologia de suporte para ao campo da computação ubíqua.

Projetos baseados nesse tipo de rede já estão sendo analisados por outras universidades do Brasil e há potencial para uso em cenários como o da computação ubíqua. As malhas de rede têm várias vantagens, como baixo custo, implantação incremental e tolerância a falhas. O diferencial da proposta apresentada neste trabalho é usar essa tecnologia barata e já existente não só para fornecer *Internet*, como nos casos citados anteriormente, mas também dar suporte e criação de um ambiente parcialmente ubíquo (SAADE, 2007; ROOS, 2008; FONTOURA, 2009; VULRAL, 2013).

4 FUNDAMENTAÇÃO TEÓRICA

4.1 Computação Ubíqua

As redes ubíquas por serem ambientes saturados de dispositivos móveis interconectados e trocando informação constantemente formam um cenário complexo e difícil de implementar, como visto em Hazra (2015). Das dificuldades destacam-se: suporte de *hardware*, expansão e manutenção da rede, demanda de alocação de usuários, necessidade de alto desempenho no tráfego de dados, roteamento e gerenciamento de usuários móveis, conforme Hazra (2015). O suporte para criação de uma rede totalmente ubíqua atualmente ainda é muito difícil, porém, tecnologias como malhas de rede podem dar suporte inicial à criação dessas redes, conforme Vural (2013).

Uma das tecnologias que dão suporte à criação de um ambiente parcialmente ubíquo, que serviria como esboço inicial para projetos futuros nesse campo, são as malhas de rede sem fio. Essa tecnologia é existente, relativamente barata e resolveria o problema da rede de suporte para o ambiente ubíquo, conforme Roos (2009) e Vural (2013). Com o uso de determinados protocolos, essa tecnologia pode dar suporte ao ambiente de rede pretendido. O protocolo mais recomendado para criação de um cenário ubíquo com malhas de rede sem fio, é o protocolo HWMP (*Hibrid Wireless Mesh Protocol*). Este protocolo foi desenvolvido para o suporte de redes em malha sem fios, conforme Vural (2013). Um ambiente de rede criado com o protocolo HWMP, possui características parcialmente ubíquas que podem satisfazer o pretendido neste trabalho acadêmico, que são: Mobilidade e conexão constante, por se tratar de um protocolo que cuida do roteamento e da manutenção conexão entre os nós da malha de rede, como apresentado em Andrev (2010).

O termo Computação ubíqua foi explicado primeiramente pelo professor Mark Weiser em 1991. Ele descreveu o ambiente ubíquo com características que não eram implementadas por uma rede de computador daquela época. Hoje já existe a conexão constante, mobilidade e vários dispositivos já podem se interconectar livremente dentro de determinados ambientes de redes. Porém um ambiente totalmente ubíquo ainda é muito difícil de criar conforme Bolsoni (2009), pois seria necessário garantir conexão transparente, conectividade dentro do contexto e interoperabilidade de tecnologias sem fio.

A computação ubíqua surge da necessidade da junção da mobilidade com as tecnologias existentes na computação pervasiva[4]. Nesse contexto um dispositivo computacional que está em movimento tem a habilidade de configurar seus serviços e necessidades de forma dinâmica,

[4] Diferentes dispositivos em redes distribuídos em ambiente de trabalho e ambiente caseiro de forma imperceptível ou não.

de acordo com o ambiente em que nos movemos, segundo Bolsoni (2009). Um esboço de um ambiente ubíquo, citado por Coulouris (2005), é o resultado da integração de tecnologias para criar um ambiente totalmente baseado em computação móvel,

> também chamada de nômade, nesse modelo o usuário acessa a rede fora de seu ambiente normal(casa ou escritório) através de dispositivos que ele carrega consigo. Computação ubíqua é realizada por vários dispositivos pequenos e baratos que estão presentes em nosso ambiente (casa, escritório ou qualquer outro lugar). O termo sugere uma situação que é tão grande e sua presença será ignorada, quer dizer seu comportamento será transparente. (COULOURIS, 2005 p.6).

As redes ubíquas têm por objetivo fornecer conexão constante a usuários móveis, transparência de conexão e mobilidade. Nesse tipo de ambiente, vários dispositivos de tecnologia diferentes podem interagir através da rede. A conexão é constante e onipresente dentro da área do sinal, muitos tipos de dispositivos podem se interconectar através dessa rede, desde dispositivos móveis como veículos automotores, dispositivos fixos como eletrodomésticos a até *microchips* implantados em seres humanos, de acordo com Vural (2013).

Existem muitos projetos direcionados à computação ubíqua, porém, devido à complexidade do tema, nenhum tem a pretensão de explorar ou suprir todas as necessidades dessa tecnologia. Quando a computação ubíqua ganhar força, a informação virtual estará em toda parte e não mais concentrada em um só ponto. Por esses fatores citados nesse e outros trabalhos, pode-se ver que existe muita carência na parte tecnológica para que a computação ubíqua chegue a seu ápice, e que estão abertas propostas e ideias para que o mundo previsto por Weiser se torne realidade (SANTANA, 2009; VURAL, 2013; HAZRA, 2013).

4.2 Malha de Rede Sem Fio

Com o crescimento das malhas de rede sem fio, houve a necessidade desenvolvimento de novas tecnologias para este tipo de rede. O tema está sendo amplamente estudado devido ao seu potencial de uso para criação de ambientes de redes complexos como os ambientes ubíquos. As malhas de rede sem nunca tiveram um papel tão importante quanto hoje no cenário ubíquo, pois apresenta soluções para suprir necessidades como, alta necessidade de mobilidade e conexão constante, conforme Rodrigues (2009) e Vural (2013).

As malhas de rede sem fio são compostas por vários nós de rede, cada nó ligado a um ou mais nós diferentes. O que permite a redundância de rede e vários caminhos de enlace a ser seguido. Dessa forma um nó pode se comunicar com outro mesmo que este esteja distante e fora do seu alcance de rádio, conforme Vural (2013). A característica principal das malhas de rede sem fios é a troca de informação entre nós através de um ou mais saltos, semelhante a uma rede *Ad Hoc*, Bakht (2011). Os nós podem emitir sinais de rádio de mesmo padrão ou não, desde que sejam interoperáveis, Pathak (2010). Essa tecnologia de rede está cada vez mais

presente. Está sendo usada para interligar várias cidades em alguns estados americanos, como o caso do Arizona e Manhattan, Roos (2009).

As malhas de rede sem fio comunitárias estão se tornando uma solução atraente para oferecer acesso banda larga de baixo custo à *Internet* e outros serviços, tais como computação ubíqua e telefonia IP sobre redes sem fio. Existem propostas semelhantes, que também propõem que as malhas de rede sem fio sejam a solução para um mundo continuamente conectado, conforme Pathak (2010). Dessa forma, o cenário ubíquo aos poucos está virando realidade. De acordo com um relatório da *MuniWireless.com*, baseado em Roos (2008) que foi publicado em março de 2007, em torno de 81 cidades dos Estados Unidos tinham instalado malhas de rede sem fio municipais em toda a região ou cidade, e outras 164 estão construindo tais redes. O relatório também afirma que 38 cidades norte-americanas já possuem redes *wireless* municipais para uso exclusivo da segurança pública e dos funcionários da cidade, diz Roos (2008).

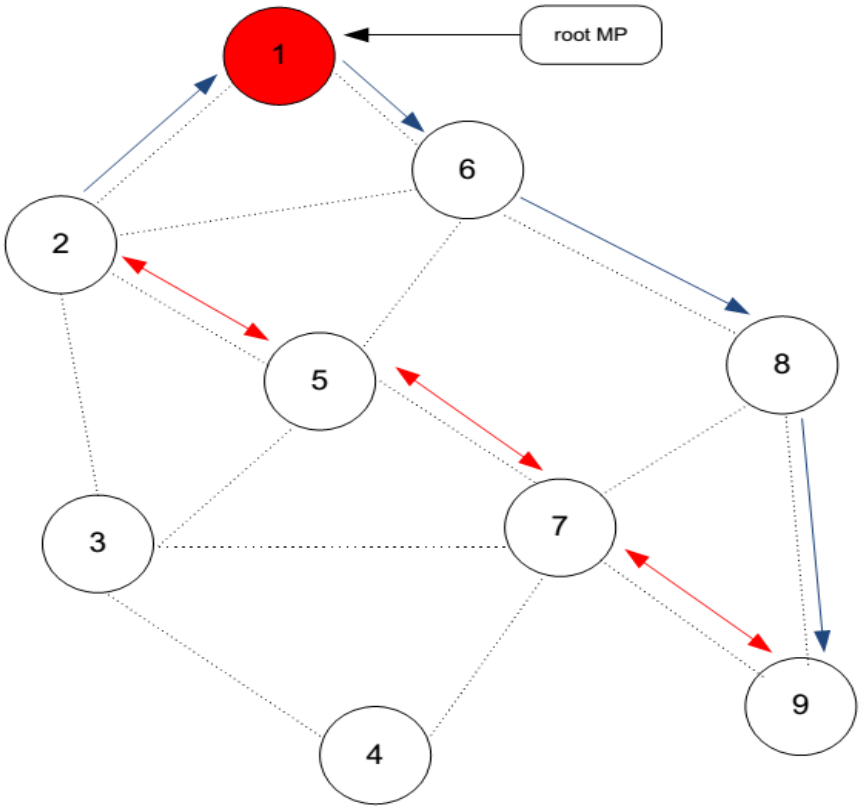
O correto funcionamento da rede depende de protocolos de roteamento multi-salto. A escolha desses protocolos vai depender das necessidades de cada rede. Existem muitos protocolos direcionados a esse tipo de rede, como por exemplo, o DSR (*Dynamic Source Routing*), RFC 4728, e o MR-LQSR (*Multi-vRadio Link-Quality Source Routing*), RFC 3561, o OLSR (*Optimized Link-State Routing*), RFC 3626. Nesse trabalho será usado o HWMP, por ser aberto e satisfazer as necessidades da rede que será simulada nesse projeto. Tais necessidades são: auto roteamento, encaminhamento dinâmico de enlaces (sempre sendo enviado pelo melhor caminho) e facilidade de conexão constante entre enlaces sem fio, conforme Andreev (2010).

O cenário deste trabalho tem necessidades de conexão constante entre os nós e mobilidade sem fio que são satisfeitos pelo HWMP. O protocolo de roteamento HWMP é destinado a grandes redes sem fio, especificamente para redes de malha sem fio. Este protocolo é um hibrido (Reativo e Proativo) que utiliza o conceito de *multipoint relaying*[3] (MPR). O MPR é uma técnica de inundação para reduzir o número de mensagem *broadcast* enviada por todos os outros nós da rede, com atualizações sobre mudanças na topologia, seu funcionamento é semelhante ao funcionamento do OLSR, porém o diferencial é que o HWMP é reativo e proativo, assim as tabelas de rotas só são solicitadas quando há a necessidade te troca de pacotes entre determinados nós da rede. No caso de falha na estrutura de *multipoint relaying*, o protocolo funciona de forma proativa enviado a tabela de rotas para todos os nós da rede, conforme Rodrigues (2009) e Andreev (2010).

A figura 4 pode ser visto a estrutura de funcionamento do HWMP, o nó 1 em vermelho é um RMP (*Root Mesh Point)*, ele é responsável por armazenar e atualizar todas as

tabelas de rotas que são enviadas para os demais nós somente quando solicitadas (setas em azul), neste modo seu funcionamento é semelhante ao do OLSR. No caso de falha do RMP o protocolo funciona de forma proativa, onde cada nó troca informações de saltos com seus vizinhos (Setas vermelhas). A estrutura de *multipoint relaying e* RMP, funciona perfeitamente com o uso do HWMP, com descrito em Rodrigues (2009) e Andrev (2010).

Figura 4: Funcionamento do HWMP.



Fonte: Extraída de Fontoura (2009).

As redes *mesh* sem fio são compostas de roteadores sem fio e clientes *mesh*, onde os roteadores têm mínima ou nenhuma mobilidade formando o *backbone* da rede. O AP 802.11s, conhecido como MP *(Mesh Point)*, quando usado numa malha de rede sem fio, estabelece ligações sem fios uns com os outros para permitir uma aprendizagem automática da topologia, criando uma configuração dinâmica dos caminhos para trocarem dados entre eles (RODRIGUES, 2009; VICENTINI, 2010).

As malhas de rede sem fio podem ser estruturadas com uso do roteador da *Linksys*. Ilustrado na Figura 5. Este roteador é usado no projeto *ReMesh* da UFF, Saade (2007). Trata-se de um roteador *wireless* 802.11g com 4 MB de memória *flash* (permanente) e 8 MB de memória RAM. Este tipo de roteador, além funcionar como ponto de acesso, roteia os clientes ligados a ele tanto pela interface sem fio, como pelas suas cinco portas *ethernet* presentes. O roteador vem de fábrica com um sistema operacional da própria *Linksys* que possui uma

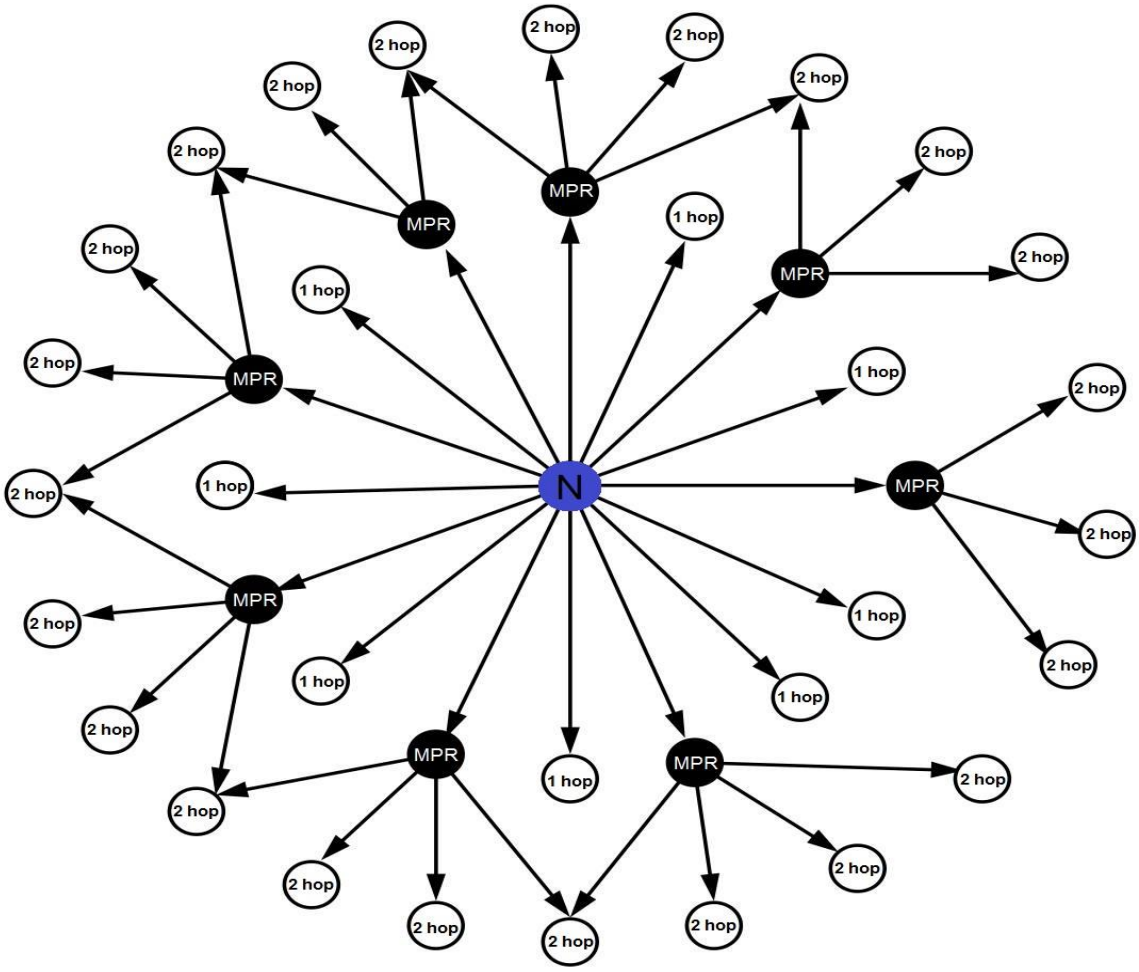
interface de administração via *web*. A adaptação para transformá-lo em um roteador de malha é feita com a instalação do *firmware* desenvolvido pelo *ReMesh*, baseado em *OpenWRT*, conforme Saade (2007). Um roteador sem fio genérico, rodando um *firware* de terceiros, que apesar de ter licença paga, pode ser uma solução mais barata e simples de usar, pois o *firmware* já implementa malhas de rede sem precisar de modificações.

Figura 5: WRT54G Linksys usado no projeto ReMesh.

Fonte: Covelo (2005).

As ligações de MP para MP criam um *backbone* sem fio que fornece ao usuário largura de banda elevada, serviços de interligação de múltiplos saltos sem falhas com um número limitado de pontos de entrada de *Internet* e interligação com outros utilizadores dentro das redes, com custo reduzido, conforme Vural (2013). O *Mesh Point* pode ser configurado como um *Root Mesh Point* (onde se instala o MPR para controle das rotas). Um *Root Mesh Point* estabelecerá uma política de manutenção de rotas, enviando um *Root Announcement* periodicamente; que é semelhante ao algoritmo de roteamento de vetor distância, para informar aos outros *Mesh Points* à tabela de rotas. Na Figura 6 são ilustrados os nós de uma malha de rede sem fio interligado, onde cada nó assume uma função. Os roteadores comuns (círculos em branco) são responsáveis por distribuir e propagar o sinal, através de um ou mais saltos, com TTL (*Time To Live*) curto para não sobrecarregar a rede. Os *Root Mesh Point* (círculos em cinza) estão instalados o MPR, que controlam e atualizam a tabela de rota, já o nó N (círculo em azul) está ligado a *Internet* e a compartilha com os demais nós da rede. Sendo assim a cobertura proposta da rede é ampla com menor custo e totalmente tolerante a falhas dos enlaces, dizem ainda, Rodrigues (2009) e Fontoura (2007).

Figura 6: Funcionamento dos Root Mesh Points.



Fonte: Adaptada de Fontoura (2009).

Características desejáveis nas malhas de rede sem fio são a auto organização e auto configuração. Esses requisitos possibilitam a manutenção das conexões dos roteadores presentes na rede de forma automática, visando à inclusão de novos roteadores na rede para o aumento da área de cobertura, conforme Vural (2013).

4.3 Network Simulator 3

O *Network Simulator* (NS3) é um simulador de rede atualmente na terceira versão. O simulador é um *software* livre, licenciado sob a GNU GPLv2, (ANDREEV 2010). A licença está disponível ao público para análise e desenvolvimento. NS3 foi projetado desde o inicio, para ser de fácil compreenção de uso, manutenção e modificação, conforme Andreev (2010). A ferramenta permite a criação de redes de computadores com ou sem fios. É uma ferramenta que requer conhecimento de linha de comando e programação.

O NS3 é muito utilizado para auxiliar na validação de pesquisas acadêmicas e no estudo na área de redes de computadores para quem não possui recursos suficientes para

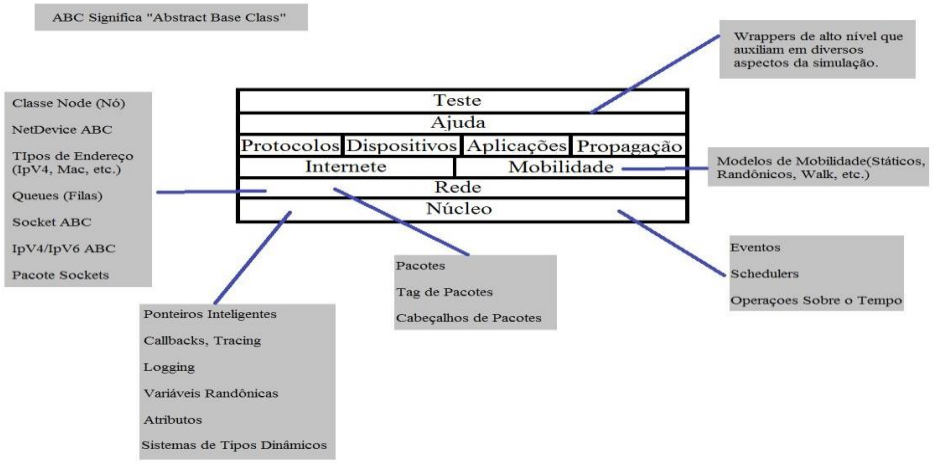
construção de ambientes reais, como visto em Siega (2013). A ferramenta NS3 pelas suas características de simulação e protocolos nativos, é indicado para a implementação de uma topologia de malha de rede sem fio. Além dos módulos nativos do simulador, tem-se a possibilidade de utilizar os módulos criados por outras entidades. O NS3 tem protocolos de roteamento, módulos e ferramentas que podem ser usados para a implementação de modelos de rede com ou sem fio, como visto em Rodrigues (2009) e Siega (2013).

A figura 7 ilustra como está disposta de forma hierárquica a estrutura dos módulos e bibliotecas quer formão o simulador. A estrutura integra componentes que são comuns em todas as partes de uma rede, como tipos de hardware, modelos de ambiente e protocolos, baseado em Siega (2013).

A estrutura principal de *software* está contida no modulo *Core* que está na base da figura 7. A partir do *Core* é executado o simulador e são carregados todas as bibliotecas e módulos. Outro ponto de destaque é o módulo *Network*, nele está contido objetos fundamentais para um simulador de rede como, pacotes, *tags* de pacotes e cabeçalhos de pacotes, com base em Siega (2013). Estes dois módulos de simulação compreendem um núcleo de simulação genérico, que pode ser utilizado por diferentes tipos de redes, e não apenas as redes baseadas na Internet, enquanto que os outros módulos são utilizados em tipos específicos de rede ou em determinados tipos de dispositivos.

Além dos recursos citados, o modulo *Mobility* e essencial para a simulação de redes móveis. Nele estão contidos recursos necessários para a simulação de mobilidade, como mecanismos para controlar e manter a posição e velocidade de um objeto, meios de identificar a mudança de curso de um nó, e métodos auxiliares que podem ser utilizadas para adicionar e configurar mobilidade em objetos, como por exemplo, carregar arquivos de mobilidade gerados pelo *bonnmotiom*. Conforme Siega (2013).

Figura 7: Estrutura de bibliotecas NS3



Fonte: Adaptada de Siega (2013).

A execução de uma simulação no NS3 gera arquivos de rastreamento (*Trace*). Nos arquivos de rastreamento são salvas todas as informações referentes aos eventos ocorridos na simulação como envio, recebimento ou perda de pacotes. Dessa forma, é possível fazer à análise do comportamento da simulação no NS3 e através dos *traces,* coletar os dados relevantes na simulação.

A análise dos eventos de simulação do NS3, pode ser feita através de ferramentas de processamento de dados baseado em XML, como por exemplo, um arquivo de execução do *NetAnim*. A vantagem de usar o NS3 em relação ao seu antecessor o Network Simulator 2 é a facilidade na leitura de arquivos *trace*, usar programas de terceiros, como o *NetAnim* e *Bonnmotion.* Houve organização e melhoria na estrutura do simulador, pois foi desenvolvido segundo padrões de engenharia de *software*, como afirma Siega (2013).

O subsistema de rastreamento é um mecanismo importantes para entender no NS3. Na maioria dos casos, o usuário do simulador terá que fazer melhorias e adaptações, a partir dos resultados observados no rastreamento *trace*, dependendo da necessidade. Sendo assim toda execução do simulador gera uma saída de dados para um estudo mais aprofundado, baseado em (NSNAM, 2015). Um ponto importante da coleta dos dados no NS3 é que diferente do *Network Simulator* 2, o NS3 coleta os dados de análise de rede dentro do próprio código da simulação. A coleta de dados é bem menos complexa, pois pode extrair na simulação somente os dados importantes para o projeto e descartar o desnecessário, baseado em (NSNAM, 2015). Este evento supre a necessidade de grande uso de processamento e memória RAM na execução e coleta de dados. Tal processo tenta evita grandes gatos de tempo e recursos para ler os enormes arquivos gerados pela simulação, conforme Rodrigues (2009). Apesar dos recursos de rastreamento do NS3, ainda não existe um modelo consistente para ser usado com todos os métodos de rastreamento do simulador, como visto em (NSNAM, 2015).

5 PLANEJAMENTO DOS EXPERIEMENTOS

O procedimento de análise deste trabalho envolveu coleta de dados, base estatística e simulação de rede. Além do NS3, foi utilizada a ferramenta *Bonnmotion* para geração de mobilidade e a ferramenta R-Studio para gerar os gráficos. Para usar o *Bonnmotion* com o NS3, foi necessária uma pequena mudança no padrão do código para seu funcionamento que será apresentada detalhadamente na seção 7. Na mudança foi criado um método que força o carregamento de um arquivo de rastreamento de mobilidade padrão do ns2 no NS3, baseado em Aschenbruck (2010).

A mobilidade gerada pela ferramenta que simula um modelo de movimentação, chamado *Manhattan Grid*, produz movimentos aleatórios como se fosse um pedestre passeando pelas ruas de *Manhattan* com sutis mudanças de direção, baseado em Aschenbruck (2010). Um *script* é gerado e nele há todas as informações de velocidade e caminhos que o nó móvel deve percorrer. Sendo assim, com um só *script* podem ser gerados vários modelos de movimentos, que podem ser usados a cada execução da simulação sem que faça grandes alterações no código fonte da implementação. O resultado gerado é o mais próximo possível de uma pessoa ou veículo transitando em um quadrante, conforme Aschenbruck (2010).

A rede simulada tem o tamanho 250 m de altura em "X" 300 m comprimento em "Y", contém 30 nós dispostos a 50 metros de distância máxima um dos outros sendo que 1 dos nós é um usuário móvel dentro da rede. A área simulada tem tamanho de 7450 m². Esta configuração satisfaz o que foi proposto neste trabalho e também pode ser modificada para uma rede de menor ou maior magnitude, como visto no Projeto *ReMesh* (SAADE, 2007). O quadro 1 mostra os tipos de sinais, sua abrangência e números de nós necessários para cobrir determinada área. A quantidade de nós se justifica porque o aumento o a diminuição deles só iria interferir na geração de dados e consumo de processamento, no contexto desse trabalho e não nas métricas, conectividade constante e mobilidade que são importantes para este trabalho, baseado em Siega (2010). Uma rede configurada com os parâmetros usados no contexto deste trabalho tanto pode ser usada para fornecer conexão constante e mobilidade, a um pequeno complexo habitacional como para uma cidade inteira, conforme Vural (2013). A grande escalabilidade se deve a interoperabilidade de sinais WiFi e de protocolos de roteamento.

Quadro 1: Projetos que usam tecnologia *mesh* sem fio.

	Google Wi-Fi	*Task Group*	*ReMesh*	*TFA*	*MadMesh*	*Roofnet*	*Dartmouth*	*UoS-Net*	*UNC*
Área de rede	*31 Km²*	*6,25 km²*	*3,5 km²*	*3 Km²*	*29,9 km²*	*6 km²*	*0.81 Km²*	*0.30 Km²*	*2.95 Km²*
Numerode nós na malha	*500 Tropos MetroMesh*	*30 Ap Routes Genericos ou outros*	*5 WRT54G Linksys*	*17 Cisco 1510*	*250 Cisco 1510*	*38 configured PCs*	*566 Cisco Aironet 350*	*444 Trapeze MP-422B*	*488 Cisco Aironet*
Ganho da antena	*7.4 dBi*	*15 dBi*	*15 dBi*	*15 dBi*	*11 dBi*	*8 dBi*	*2.2 dBi*	*15 dBi*	*2.2 dBi*
Canall	*802.11g*	*802.11s*	*802.11b/g/n*	*802.11b*	*802.11g for MAPs 802.11b for RAPs*	*APs share a single 802.11b channel*	*802.11b*	*802.11b/g*	*802.11b*
TX Potência	*≤ 100 mW*	*200 mW*	*≤ 200 mW*	*200 mW*	*200 mW*	*200 mW*	*100 mW*	*≤ 200 mW*	*100 mW*

Fonte: Adaptado de Vural (2013) e Saade (2005).

O usuário móvel pode se movimentar por toda a rede e se comunicar com todos os nós que esteja dentro do seu raio de alcance. Os movimentos são aleatórios. Os demais nós da rede são fixos e se comunicam e trocam dados de aceso e controle com seus vizinhos mais próximos.

A simulação foi feita em duas partes, Cenário 1 e Cenário 2, pois assim facilita o processo de análise dos dados coletados. Na primeira parte foram feitas 40 execuções com o tempo de 240 segundos. Na segunda parte foram feitas 20 simulações com o tempo de 180 segundo. O importante na fragmentação da execução é dar mais confiabilidade devido a repetibilidade do processo de execução e coleta, conforme Feitelson (2015).

Foram feitas quarenta (40) simulações com o tempo de 240 segundos com o nó móvel transitando no máximo de 15 km/h e mínimo de 6 km/h, velocidade estimada de um humano caminhado. Onde o mesmo pode parar em determinado ponto, diminuir ou aumentar a velocidade, que foi chamada de cenário 1. Foram feitas vinte (20) simulações com o tempo de

180 segundos com o nó móvel transitando no máximo a 80 km/h e mínimo a 30 km/h, velocidade estimada de um veículo popular motorizado. Neste cenário o nó também pode parar em determinado ponto, diminuir ou aumentar a velocidade, que foi chamado de cenário 2. Foram totalizados 3 horas e 40 minutos de coleta de dados. A variação de velocidade nos dois cenários é para aumentar a confiabilidade da análise das características ubíquas conexão constante e mobilidade, baseado em Feitelson (2015).

O tráfego de pacotes na rede será apenas de pacotes de controle de conexão TCP/UDP, pois o que vai ser analisado é: conexão constante com mobilidade. Uma perfeita réplica da rede teria alta demanda de desempenho no tráfego de dados, desempenho de roteamento e desempenho no gerenciamento de usuários móveis, por isso está fora do escopo do trabalho. A quantidade de um usuário móvel na simulação se justifica devido ao aumento de processamento de *hardware*, leitura dos arquivos de controle, de e troca de dados entre os nós da rede. Nesse trabalho, devido a limitações de recursos de processamento e memória RAM, uma grande quantidade de nós da rede poderia gerar grande carga de trabalho, e negação de serviço por sobrecarga, comprometendo assim um dos pontos principais pesquisado nesse trabalho, a conexão constante (DUARTE, 2008; HAZRA, 2015).

6 REALIZAÇÃO DOS EXPERIMENTOS

Na simulação foram usados dois cenários de mobilidade diferentes dentro do mesmo cenário de rede. O cenário 1, um nó móvel percorreu toda a extensão da rede com velocidade média de 2,5 metros por segundo e tempo de 240 segundo (4 minutos), este cenário foi executado quarenta vezes totalizando 9600 segundos (2 horas e 40 minutos). No cenário 2 o nó móvel percorreu toda a extensão da rede com velocidade média de 13,88 metros por segundo e tempo de 180 segundos (3 minutos), este cenário foi executado vinte vezes totalizando 3600 segundos). O tempo total de simulação foi de 13200 segundos (3 horas e 40 minutos) e percorridos 74000 metros (74 km). O quadro 2 mostra resumidamente dados de tempo, velocidade e distância percorrida individualmente em cada cenário.

Quadro 2: Dados de tempo, velocidade e distância.

	Pedestre	Veiculo
Tempo	2,4 Horas	01 Horas
Velocidade	9 Km/h	50 Km/h
Distância	24 Km	50 Km

Fonte: Elaborada pelo autor.

O uso do *Bonnmotion* se deu por linha de comando, desde a geração dos cenários de mobilidade até a conversão para um formato que fosse reconhecido pelo NS3. O comando da figura 8 foi usado para criar o cenário 1. No parâmetro menos "–i" é passada uma variável aleatória, para que diferentes áreas da rede sejam percorridas pelo nó móvel para reduzir a redundância da simulação.

Figura 8: Comando *Bonnmotiom*.

```
bm -f mmg1 ManhattanGrid -n 1 -x 300 -y 250 -d 240 -i $(($RANDOM %7450) + 1)) -R 1 -o 15 -p 6 -m 10000
```

Fonte: Elaborada pelo autor.

O quadro 3, mostra os detalhes do comando e de cada parâmetro passado para criação dos cenários. No *Bonnmotiom,* os paramentos são passados por linha de comando. Se necessários comandos de *shellscript* podem ser incorporados aos comandos do *Bonnmotiom,* como foi o caso do parâmetro "-i", descrito também no quadro 3.

Quadro 3: Comandos *Bonnmotiom* para criação de cenário de mobilidade.

Bm –f	Chamada do *Bonnmotion*
mmg1	Nome do arquivo de mobilidade
ManhattanGrid	Cenário de mobilidade gerado
-n	Número de nós que se moverão
-x	Altura que o no percorrerá no eixo X
-y	Altura que o no percorrerá no eixo Y
-d	Duração dos movimentos em segundos
-i	Localização inicia do nó na simulação
-P	Velocidade mínima
-O	Velocidade Máxima
-R	Semente de aleatoriedade fixa
-m	Fator de aleatoriedade dos movimentos

Fonte: Elaborada Pelo autor e baseado em, Aschenbruck (2010).

A ferramenta *Bonnmotion* gera os movimentos por traços de rastreamento OTCL (*Object Tool Command Language*), os traços podem ser carregados em um script de mesma linguagem de programação para gerar movimentos diversos, conforme Aschenbruck (2010).

A figura 9 mostra um exemplo do código de rastreamento gerado pelo *Bonnmotiom*, nela a localização do nó móvel é dada nas condenadas “X” e “Y” da grade com reação ao tempo de execução.

Figura 9: Traços de rastreamento de localização *NetAnim*.

```
$node_(0) set X_ 246.84957015832646
$node_(0) set Y_ 170.0
$ns_ at 0.0 "$node_(0) setdest 226.91406347611314 170.0 10000.233088697805"
$ns_ at 0.001993504201891483 "$node_(0) setdest 201.91406347611314 170.0 9999.757332475301"
$ns_ at 0.004493564870244882 "$node_(0) setdest 196.91406347611314 170.0 9999.982408366146"
$ns_ at 0.004993565749828122 "$node_(0) setdest 191.91406347611314 170.0 9999.633375900368"
$ns_ at 0.005493584081705194 "$node_(0) setdest 131.91406347611314 170.0 10000.191613978022"
$ns_ at 0.011493469115521293 "$node_(0) setdest 130.0 170.0 9999.865664710673"
$ns_ at 0.011684878034429858 "$node_(0) setdest 130.0 151.91406347611317 9999.865667684411"
$ns_ at 0.013493495982402237 "$node_(0) setdest 130.0 141.91406347611317 10000.004600008038"
$ns_ at 0.014493495522401645 "$node_(0) setdest 130.0 130.0 9999.954217348119"
$ns_ at 0.01568490732461214 "$node_(0) setdest 91.91406347611317 130.0 9999.954213227124"
$ns_ at 0.019493518415401923 "$node_(0) setdest 90.0 130.0 10000.255446736623"
$ns_ at 0.01968491987372545 "$node_(0) setdest 90.0 143.08593652388683 10000.25542408897"
$ns_ at 0.020993480102333706 "$node_(0) setdest 90.0 170.0 10000.10446489398"
$ns_ at 0.023684858334490855 "$node_(0) setdest 113.08593652388683 170.0 10000.104462124871"
$ns_ at 0.025993427871071617 "$node_(0) setdest 123.08593652388683 170.0 10000.090184270673"
$ns_ at 0.02699341885272588 "$node_(0) setdest 163.08593652388683 170.0 10000.139443634833"
$ns_ at 0.030993363076049718 "$node_(0) setdest 170.0 170.0 10000.134026946047"
$ns_ at 0.03168476015707711 "$node_(0) setdest 170.0 121.91406347611317 10000.13402231615"
$ns_ at 0.036493289364443626 "$node_(0) setdest 170.0 111.91406347611317 9999.834454477692"
$ns_ at 0.037493305919269915 "$node_(0) setdest 170.0 61.91406347611317 10000.055113645409"
$ns_ at 0.042493278362599085 "$node_(0) setdest 170.0 50.0 10000.015203517685"
$ns_ at 0.04368468289885641 "$node_(0) setdest 156.91406347611317 50.0 10000.015205847627"
$ns_ at 0.04499327456142055 "$node_(0) setdest 141.91406347611317 50.0 10000.218317837109"
$ns_ at 0.046493241811445991 "$node_(0) setdest 131.91406347611317 50.0 9999.862356929154"
$ns_ at 0.04749325557895645 "$node_(0) setdest 130.0 50.0 9999.975140831142"
$ns_ at 0.04768466240238922 "$node_(0) setdest 130.0 41.91406347611317 9999.975152948584"
$ns_ at 0.0484932580638997 "$node_(0) setdest 130.0 36.91406347611317 9999.930222016112"
```

Fonte: Elaborada pelo autor.

No NS3 a simulação foi feita usando a linguagem C++ em código padronizado. Houve a necessidade de fazer uma pequena adaptação no código para que fosse carregado o cenário de mobilidade *Mandatam Grid* gerado pelo *Bonnmotion*. O mesmo permite que seja

carregado um *tracefile* em OTCL, que no *script* em C++ padronizado NS3. A adaptação teve a necessidade de ser feita por não conter módulos de mobilidade para uma implementação do *bonnmotiom* no NS3 necessários para realização do proposto nesse trabalho, conforme Aschenbruck (2010).

Para entender a mudança na padronização é preciso saber como as bibliotecas de mobilidade funcionam e como são pré-programadas. No NS3 código começa pelo carregamento de módulos através da inclusão de arquivos. Durante a construção, a biblioteca "mobility-module.h", carrega todo os arquivos incluídos nela. Como citado anteriormente, a padronização da biblioteca carrega arquivos em *python* e C++, baseando em (NSNAM, 20015). Para carregar o arquivo de rastreamento de posição gerado pelo *Bonnmotiom* em OTCL padrão do NS2, seria necessária mudar a padronização da biblioteca de mobilidade do "mobility-module.h" NS3, como visto em (NSNAM, 20015).

Para que não fosse necessário alterar a estrutura e padronização de código do NS3, foi usada a sobreposição de atributos padrão. Assim pode ser alterado o comportamento do NS3 sem precisar para carregar cenários de mobilidade alternativos sem alterar as bibliotecas e módulos padrões. Para isto o NS3 fornece um mecanismo de análise de argumentos de linha de comando, que configura automaticamente variáveis locais e globais através desses argumentos. O para usar argumentos de linha de comando é preciso declarar o analisador de linha de comandos. Isto é feito com a seguinte linha de programação, em seu programa principal, como mostra a figura 10.

Figura 10: Argumentos para sobreposição de atributos.

```
MeshScript::Configure (int argc, char *argv[])
{
CommandLine cmd;
cmd.AddValue ();
 }
```

Fonte: Elaborada pelo autor.

Para esse trabalho foi preciso carregar o arquivo de mobilidade "ConstantPositionMobilityModel" padrão da biblioteca "mobility-module.h" do NS3. As linhas da figura 11 sobrepõe os dados de "ConstantPositionMobilityModel" com os dados de mmg1.ns_movements, que foi o arquivo de mobilidade gerado pelo *Bonnmotiom* para esse trabalho. A sobreposição se deu da seguinte forma: Das linhas 239 a 250 da figura 11 o método "showPosition", atribui um número fixo nas variáveis utilizadas pelo modelo "mobility-module.h". Todas as variáveis e alocação de posições são reinicializadas. Na

linha 251 os mecanismos de análise de argumentos de linha de comando são chamados em um método principal e faz a sobreposição de atributos. O “nodeId” na linha 240 ler o arquivo de rastreamento e instancia os valores dos dados de localização e tempo das variáveis declaradas no método “showPosition”. As linhas 254 e 255 lê os atributos fixos do arquivo “traceFile”, pela chamada da biblioteca “mobility-module. H” e as linhas 257 a 251 faz a sobreposição dos atributos de tempo e posição com os dados do arquivo “mmg1.ns_movements” com o movimento *Manhatan Grid.*

A partir da linha 263 da figura 11 é carregado na simulação o nó móvel com os das do novo “traceFile” carregado. O objeto “NODE” é criado e instanciado com os valores de um no padrão da biblioteca “nodeList” do NS3. O objeto “Ns2MobilityHelper” recebe os dados do novo “traceFile”, em seguida é chamado na execução global da simulação com ns2.Install. A linha 262 pega o objeto Simulador padrão dos scripts no NS3 e imprime as variáveis do “traceFile” que interessa para a simulação que são, a posição, o tempo e o nó móvel da simulação. A linha 270 passa os valores anteriores da simulação para “t”, a linha 271 sobrepõe todos os valores de “t” finalmente a linha 272 retorna os valores sobrepostos na simulação como se estivesse feito uma chamada de um arquivo de mobilidade padrão da biblioteca “mobility-module.h”.

Figura 11: Trecho da simulação utilizada no trabalho.

```
237  void
238  showPosition (Ptr<Node> nodes, double deltaTime)
239  {
240    uint32_t nodeId = nodes->GetId ();
241    Ptr<MobilityModel> mobModel = nodes->GetObject<MobilityModel> ();
242    Vector3D pos = mobModel->GetPosition ();
243    Vector3D speed = mobModel->GetVelocity ();
244    std::cout << "At " << Simulator::Now ().GetSeconds () << " nodes " << nodeId
245              << ": Position(" << pos.x << ", " << pos.y << ", " << pos.z
246              << ");   Speed(" << speed.x << ", " << speed.y << ", " << speed.z
247              << ")" << std::endl;
248
249    Simulator::Schedule (Seconds (deltaTime), &showPosition, nodes, deltaTime);
250  }
251  int
252  main (int argc, char *argv[])
253  {
254    std::cout.precision (4);
255    std::cout.setf (std::ios::fixed);
256    double deltaTime = 100;
257    std::string traceFile = "/home/joaofaustino/mmg1.ns_movements";
258    CommandLine cmd;
259    cmd.AddValue ("traceFile", "Ns2 movement trace file", traceFile);
260    cmd.AddValue ("deltaTime", "time interval (s) between updates (default 100)", deltaTime);
261    cmd.Parse (argc, argv);
262
263    Ptr<Node> n0 = CreateObject<Node> ();
264
265    Ns2MobilityHelper ns2 = Ns2MobilityHelper (traceFile);
266    ns2.Install ();
267
268    Simulator::Schedule (Seconds (0.0), &showPosition, n0, deltaTime);
269
270    MeshScript t;
271    t.Configure (argc, argv);
272    return t.Run ();
273  }
```

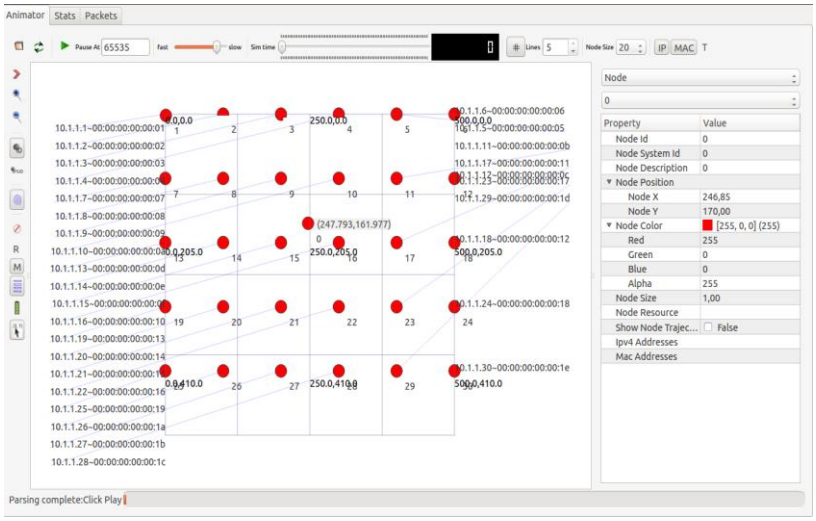
Fonte: Elaborada pelo autor.

7 COLETA DOS DADOS E ANÁLISE DOS RESULTADOS

A rede foi estrutura com 30 nós fixos e um no móvel. Para a execução foi criado 2 cenários de mobilidade. Os dois cenários são idênticos. O nó móvel troca pacotes de controle de conexão durante toda a simulação com os nós fixos. Os nós além de pacotes de controle de conexão trocam pacotes de controle de rotas entre si. O trajeto do nó móvel é em forma de grafo com passeio aberto. O nó móvel percorre um trajeto aleatório durante toda a simulação, onde cada nova execução ele parte de um ponto diferente da rede.

Na figura 12 é mostrada a posição inicial de nó “0”, onde o mesmo se encontra nas coordenadas 247.793 em “X” e 161.997 em “y”. O espaço percorrido com mobilidade é monitorado pela troca de pacotes do no móvel como os nós fixos. Neste trabalho, considere-se tempo sem conexão o tempo com ausência de trocas de pacotes.

Figura 12: Visualização da simulação no *NetAmim.*



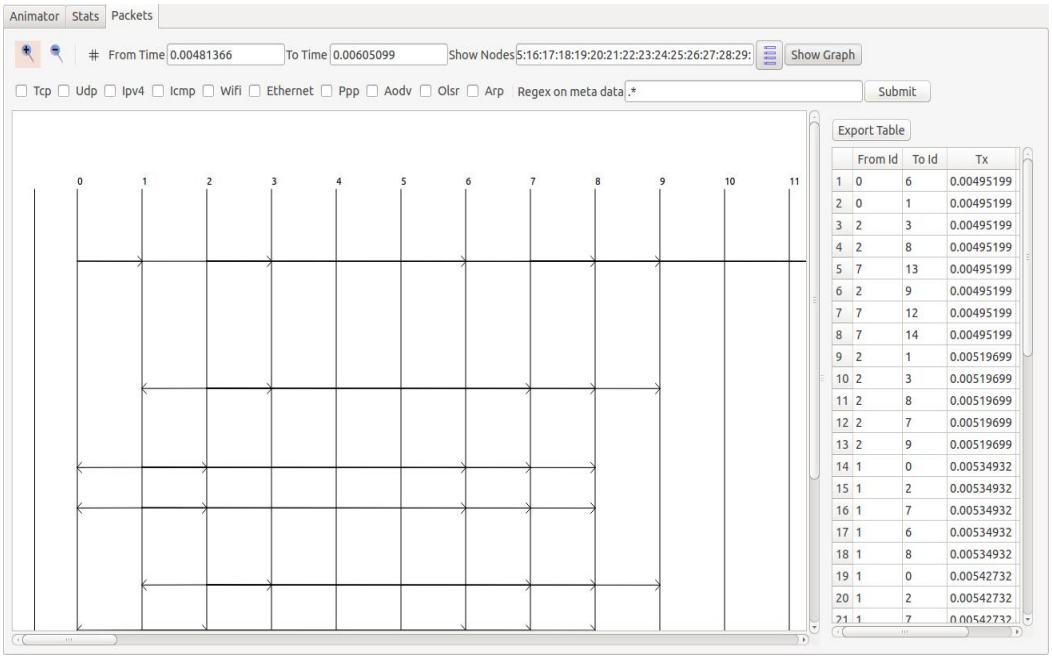
Fonte: Elaborada pelo autor.

O protocolo HWMP faz a manutenção e estabelece rota entre os nós da rede. Os pacotes que circulam na rede são pacotes TCP, UDP e pacotes padrão de controle de Conexão. Os pacotes TCP e UDP não são gerados em demasia para não sobrecarregar a rede. A possibilidade de uma sobrecarga de pacotes poderia gerar negação de serviço na conexão e gerar resultados imprecisos para o trabalho. A presença de transito de pacotes é mínima, pois testar o desempenho da rede relativo a escalabilidade de usuários móveis, tráfego de pacotes e QoS (*Quality Of Service*) da rede não está dentro do escopo do trabalho, pois implicaria em outras métricas ubíquas como, escalabilidade para ambientes saturados e conexão dentro de contexto. Ambientes saturados de nós móveis, vão gerar grande carga de dados para rede e consequentemente negação de serviços da mesma. Situações desse tipo vão comprometer a

conexão constante, que é uma métrica importante para esse trabalho (JAIN, 1991; DUARTE, 2009; SIEGA, 2010; HAZRA, 2015).

A ferramenta *NetAnim* foi uma ferramenta crucial para visualização da rede, pois diminui a necessidade de abstração relativa a topologia. A ferramenta tem funções como, animação gráfica da topologia para diminuir a abstração e mapeamento da conexão entre os nós que podem ajudar na visualização e coleta. Na figura 13 e 14 pode ser visto o mapeamento e encaminhamento de pacotes nos segundos inicias da simulação.

Figura 13: Mapeamento troca de pacotes.



	From Id	To Id	Tx
1	0	6	0.00495199
2	0	1	0.00495199
3	2	3	0.00495199
4	2	8	0.00495199
5	7	13	0.00495199
6	2	9	0.00495199
7	7	12	0.00495199
8	7	14	0.00495199
9	2	1	0.00519699
10	2	3	0.00519699
11	2	8	0.00519699
12	2	7	0.00519699
13	2	9	0.00519699
14	1	0	0.00534932
15	1	2	0.00534932
16	1	7	0.00534932
17	1	6	0.00534932
18	1	8	0.00534932
19	1	0	0.00542732
20	1	2	0.00542732
21	1	7	0.00542732

Fonte: Elaborada pelo autor.

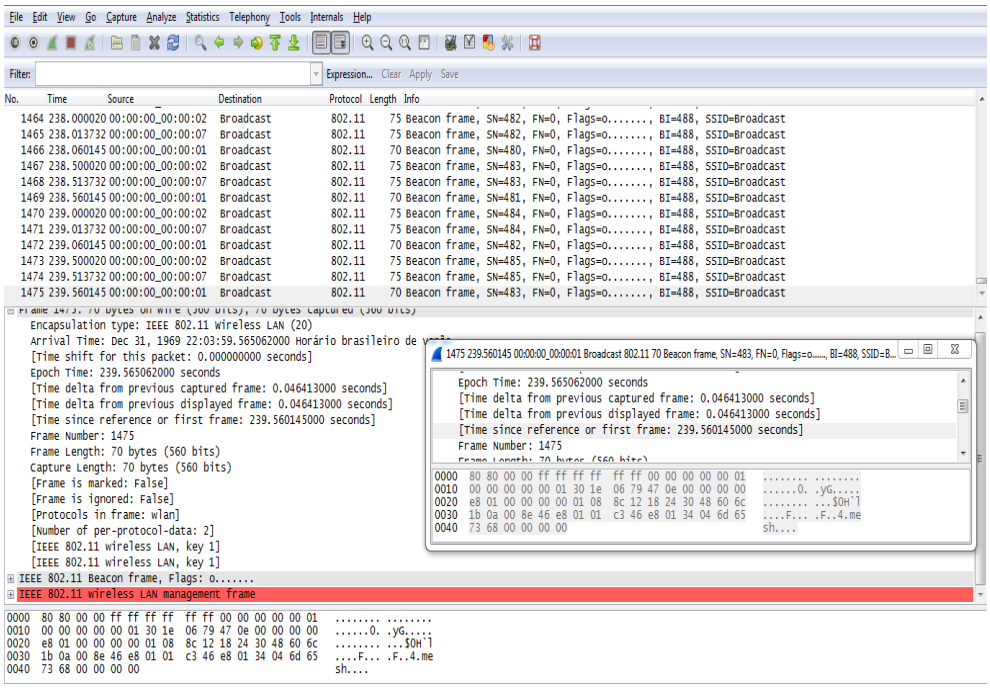
Figura 14: Mapeamento troca de pacotes, visualizado com *NetAnim*.

```
From Id To Id   Tx  Meta
0    6   0.00495199
0    1   0.00495199
2    3   0.00495199
2    8   0.00495199
7    13  0.00495199
2    9   0.00495199
7    12  0.00495199
7    14  0.00495199
2    1   0.00519699
2    3   0.00519699
2    8   0.00519699
2    7   0.00519699
2    9   0.00519699
1    0   0.00534932
1    2   0.00534932
1    7   0.00534932
1    6   0.00534932
1    8   0.00534932
1    0   0.00542732
1    2   0.00542732
1    7   0.00542732
1    6   0.00542732
1    8   0.00542732
2    1   0.00557565
2    3   0.00557565
2    8   0.00557565
2    7   0.00557565
2    9   0.00557565
```

Fonte: Elaborado pelo autor.

Para verificação das taxas de tempo e pacotes foi usado o *WireShark* sobre os dados coletados e armazenados já processados em um arquivo *trace*. O rastreamento de tempo de simulação e pacotes enviados e recebidos se deu pela análise desses arquivos. Para determinar o tempo de conexão da simulação foi analisado o último evento de processamento do *WireShark*, nele está o número total de pacotes transmitidos e o tempo total de simulação. A figura 15 ilustra que foi falado no parágrafo anterior. Nela pode-se ver que o último evento de execução foi o de número 1475, no qual foram capturados 70 *bytes* em 239.56 segundos. Como o tempo determinado para este evento era de 240 segundos (Cenário 1 da simulação), considera-se que o restante do tempo o nó móvel não teve conexão e nem transmissão de pacotes.

Figura 15: Leitura de dados da com *wireshark*.



Fonte: Elaborado pelo autor

A visão geral dos dados de velocidade, mobilidade e conexão constante foram as seguintes:

- Cenário 1: Foi definida a velocidade média de 2,5 m/s, o nó móvel percorreu 24000 metros em 9600 segundos com conexão constante. Porém, o total percorrido com conexão foi 23937, 024 metros no tempo de 9574,81 segundos.

- Cenário 2: Foi definida velocidade média de 13,88 m/s, o nó percorreu 50000 metros em 3600 segundos com conexão constante. Porém, o total percorrido com conexão foi 49602,956 metros no tempo de 3573,7 segundos.

- A conclusão final da coleta pode-se resumir que o espaço total percorrido com conexão foi 99.37% do tempo total de simulação. Sendo 99.60% do tempo total de simulação o nó móvel esteve conectado.

7.1 Cenário 1

Este Cenário simula um pedestre caminhando. A média de tempo conectado e de espaço percorrido com conexão foi acima do esperado para um cenário complexo, pois esperava-se que com a alta taxa de mobilidade aplicado na simulação o tempo desconectado seria maior. Os dados são apresentados nas tabelas e gráficos nas seções 7.1.1 e 7.1.2.

7.1.1 Tempo Cenário 1

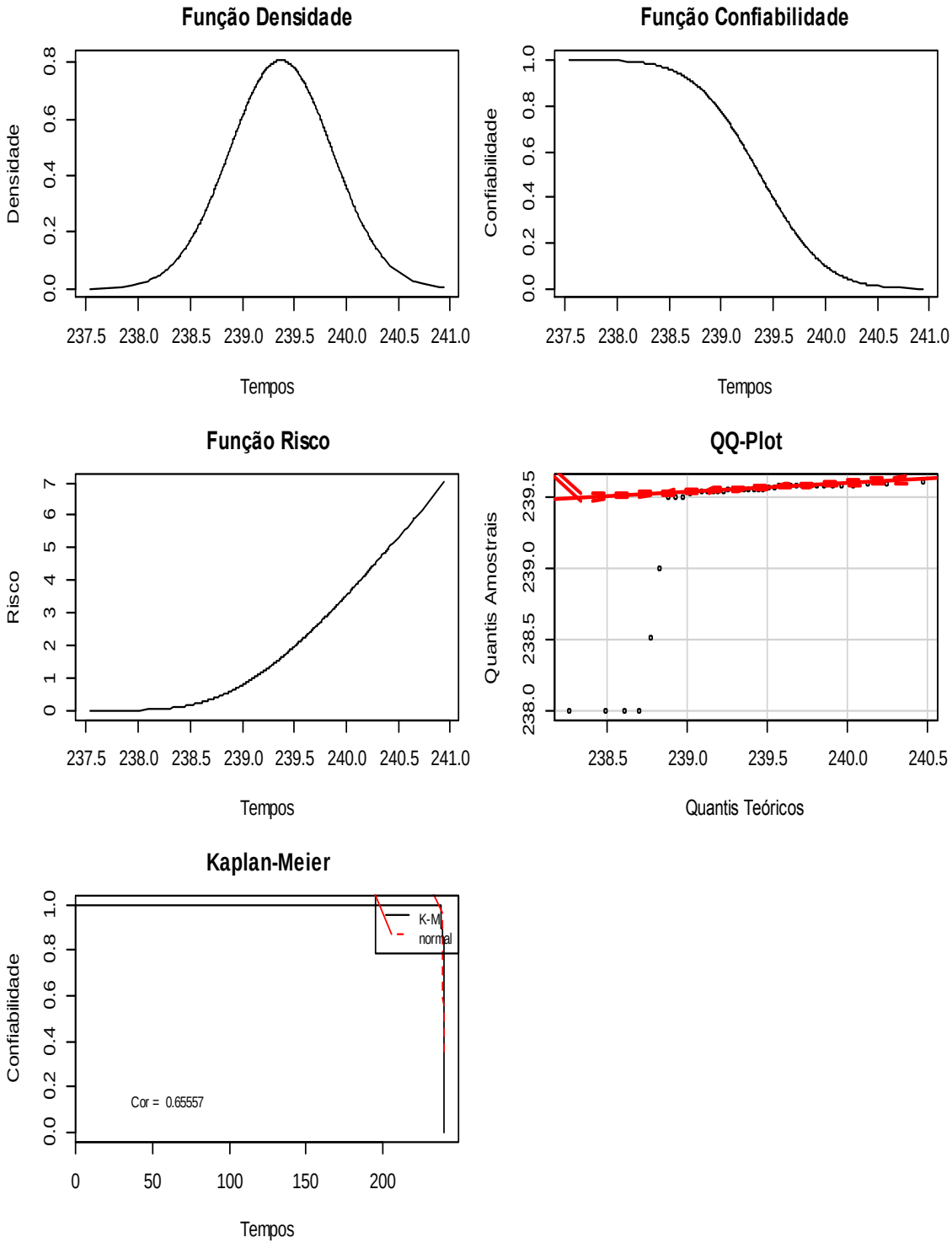
No quadro 4 e figura 16 estão descritos os dados de confiabilidade por processo. Em ambas estão ilustrados a relação de confiabilidade do tempo com relação densidade de amostras, função de risco, QQ-plot (Quartil-Quartil) e a estimativa de sobrevida das amostras com base na população (Gráfico Kaplan-meier). Com base nos dados pode ser observado que as amostras seguem um padrão de normalidade. O padrão de normalidade evita *out-liers* e que sejam apresentados dados extremamente afastado do padrão normal, baseado em Jain (1991) e Martins (2011). Os dados gráficos mostram que o tempo de simulação do trabalho foi uniforme, já que foi estabelecido o tempo de 240 segundos em uma população amostral de 40 execuções.

Quadro 4: Dados de Confiabilidade.

Método	***Distribuição***	
Máxima Verossimilhança	Normal	
	Estimativas	***Lim. Inf.***
Médias	239,37025	239,2174364
Desvio Padrão	0,493109965	0,385056465
Percentis (%)	***Estimativas***	***Lim. Inf.***
0,1	238,7383042	238,5320819
0,5	239,37025	239,2174364
0,9	240,0021958	239,7959736
Índices	***Valor***	
Desvio Padrão	0,493109965	
Mediana	239,37025	
1º Quartil	239,0376524	
3º Quartil	239,7028476	

Fonte: Elaborado pelo autor.

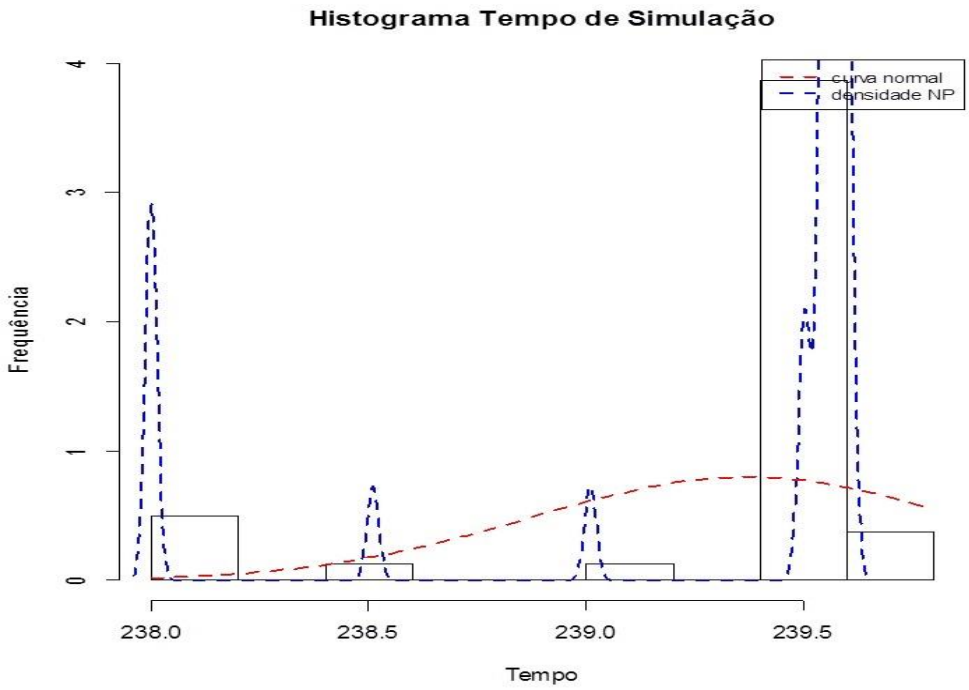
Figura 16: Gráfico de confiabilidade cenário 1.



Fonte: Elaborada pelo auto e baseado em Silva (2006) e Martins (2011).

A figura 17 demostra o diagrama de frequência dos intervalos de tempo previamente tabulado e dividido em classes uniformes. O valor da classe de maior frequência está bem próximo do tempo de 240 segundos que foi determinado em cada segmentação da simulação, baseado em Martins (2011).

Figura 17: Histograma cenário 1.



Fonte: Elaborada pelo autor, baseada em Silva (2006) e Martins (2011).

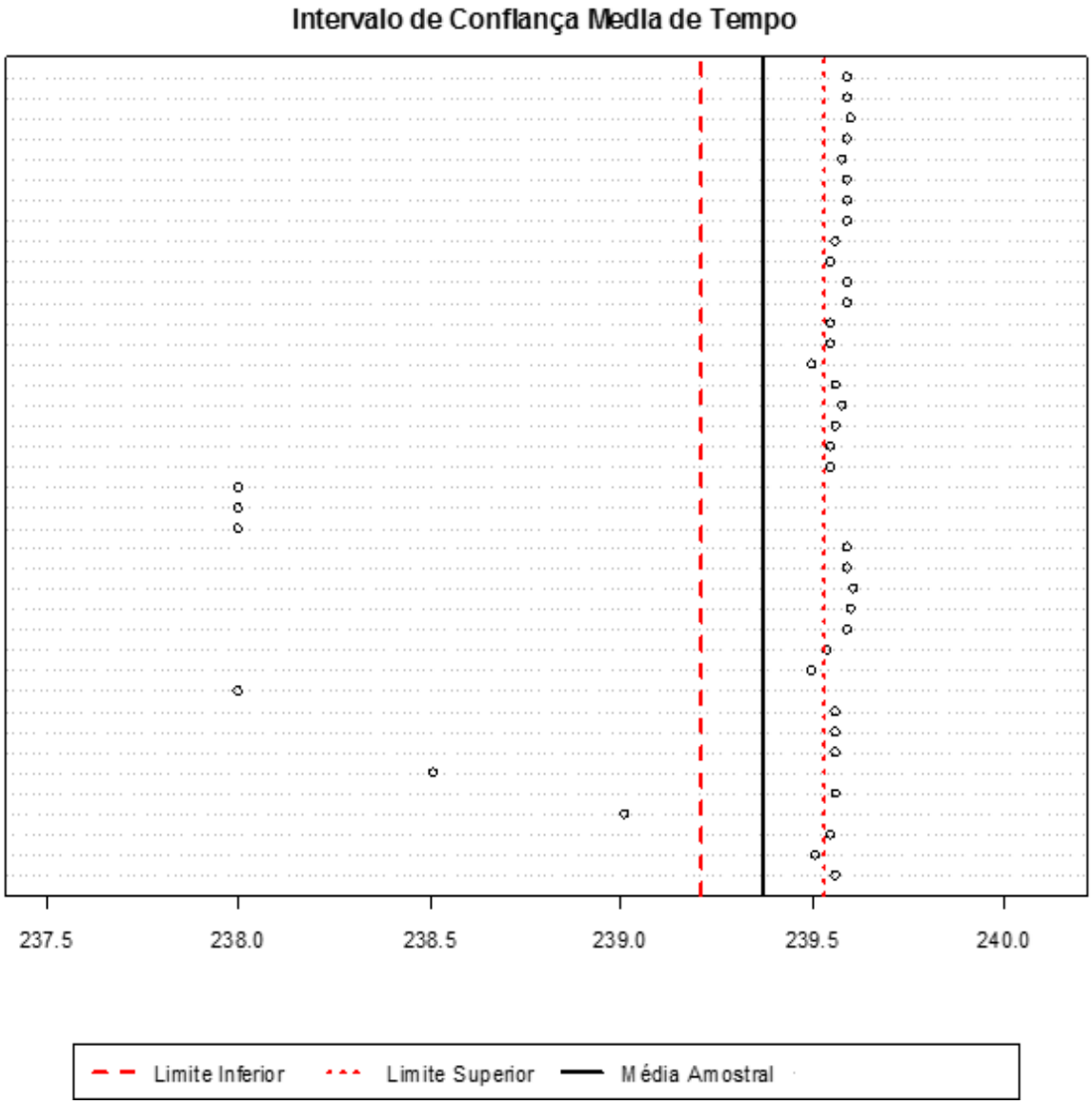
No quadro 5 está resumido os dados estatísticos básicos. Na análise dos dados foi atribuído nível de confiança de 95%, com grau de liberdade de número total de amostra menos um. Com trinta e nove parâmetros estatísticos comparativos que são o número de amostra menos um. Os dados obtidos foram muito próximo ao tempo de 240 segundos estabelecido para o cenário 1. A figura 18 ilustra a média das amostras e o intervalo de confiança por processo de execução.

Quadro 5: Dados estatísticos básicos.

Informação	*Valor*
Graus de Liberdade	39
Média Amostral	239,37025
Desvio padrão amostral	0,499391874
Tamanho da amostra	40
Intervalo de Confiança	95%
Limite Inferior	239,2105367
Limite Superior	239,5299633

Fonte: Elaborado pelo autor.

Figura 18: Media amostral e intervalo de confiança cenário.



Fonte: Elaborada pelo autor, baseada em Silva (2006) e Martins (2011).

7.1.2 Visão Geral Envio e Recebimento de Pacotes Cenário 1

A base estatística para apresentação dos dados deste tópico foram as mesmas da seção 7.1.1, por isso será apresentado resumidamente em tabelas e gráficos.

No quadro 6 estão resumidos os dados de confiabilidade da contagem de pacotes por processo de execução. A tabela contém os dados estatísticos básicos de confiabilidade. Os de confiabilidade servem para evitar que padrões fora da comum entrem na base estatística amostral. Podemos ver no quadro 6 que o número de pacotes trocados foi estável e teve uma média estável, apesar de parte dos pacotes terem sido perdidos. O processo de contagem e coleta foram os mesmos usados para base estatística de tempo, baseado em Martins (2011).

Quadro 6: Confiabilidade contagem de pacotes.

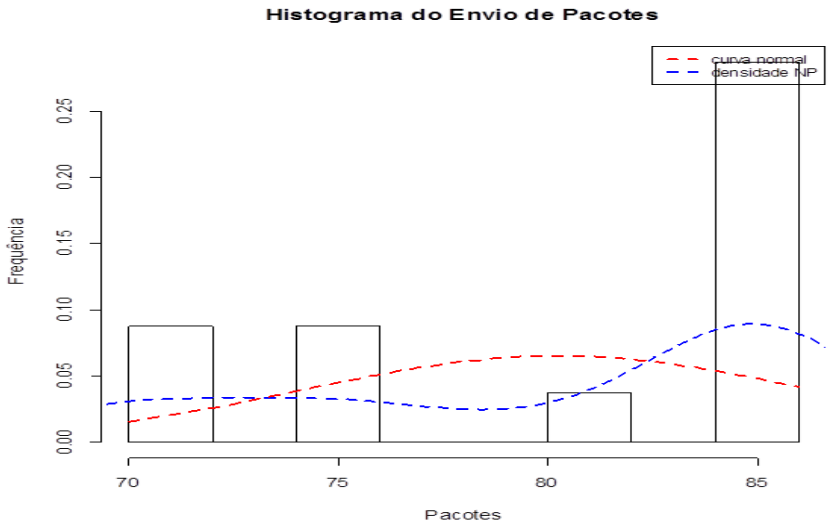
Método	*Distribuição*			
Máxima Verossimilhança	Normal			
	Estimativas	*Desvio Padrão*	*Lim. Inf.*	*Lim. Sup.*
Médias	80,25	0,951150619	78,38577904	82,11422096
Desvio Padrão	6,015604708	0,672564967	4,697401595	7,333807821
Percentis (%)	*Estimativas*	*Desvio Padrão*	*Lim. Inf.*	*Lim. Sup.*
0,1	72,54069237	1,283590711	70,0249008	75,05648393
0,5	80,25	0,951150619	78,38577904	82,11422096
0,9	87,95930763	1,283590711	85,44351607	90,4750992
Índices	*Valor*			
Desvio Padrão	6,015604708			
Mediana	80,25			
1º Quartil	76,19253628			
3º Quartil	84,30746372			

Fonte: Elaborado pelo autor.

A figura 19 mostra o histograma do envio e recebimento dos pacotes. No gráfico pode ser vista uma curva normal e sem muita alteração no traço de frequência de envio e recebimento. Comparando os dados de tempo do tópico anterior pode ser visto que o envio de pacotes permanece estável com relação a variação de tempo.

O nó móvel da simulação começa a transmitir pacotes no evento 1 do tempo de simulação, este processo foi descrito com detalhes na seção 7, pois a variação da troca de pacotes não está sujeita ao tempo e sim ao grau de mobilidade da rede e ao grau de velocidade do nó móvel. Por esse motivo, o trabalho teve dois cenários, porém em cada um o nó móvel transita com uma velocidade diferente, baseado em Vural (2013).

Figura 19: Histograma troca de pacotes cenário 1.



Fonte: Elaborada pelo autor, baseado em Silva (2006) e Martins (2011).

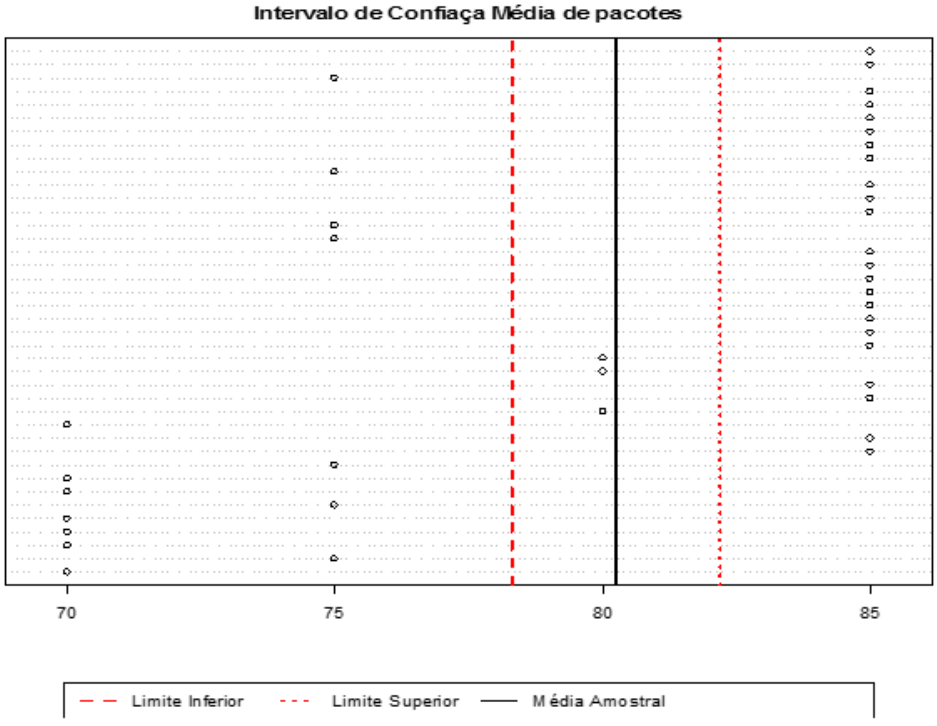
No quadro 7 e figura 20 estão respectivamente descritos e ilustrados, os dados estatísticos básicos de intervalo de confiança da média, com nível de confiança de 95%. Pode ser visto que foi transferida uma média de 80,25 pacotes de controle por cada execução de cenário.

Quadro 7: Dados estatísticos da troca de pacotes cenário 1.

Informação	*Valor*
Graus de Liberdade	39
Média Amostral	80,25
Desvio padrão amostral	6,092239704
Tamanho da amostra	40
Intervalo de Confiança	95%
Limite Inferior	78,30160722
Limite Superior	82,19839278

Fonte: Elaborado Pelo Autor.

Figura 20: Intervalo de confiança da média de pacotes Cenário 1.



Fonte: Elaborada pelo autor, baseada em Silva (2006) e Martins (2011).

7.2 Cenário 2

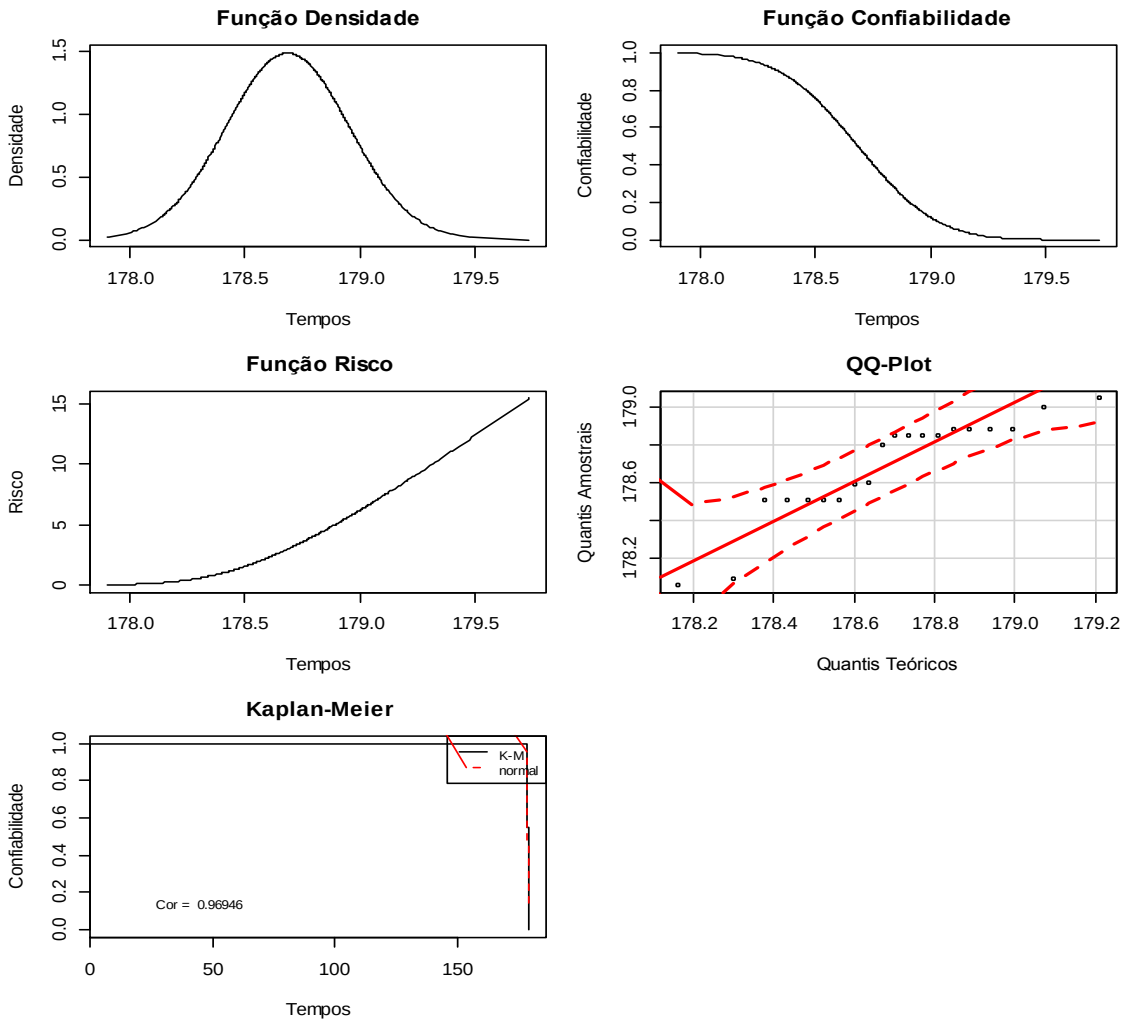
A apresentação dos dados coletados do cenário 2. Este Cenário simula um veículo transitando em média velocidade. As medidas de velocidade e troca de pacotes sofreu uma variação neste cenário. Nele o tempo conectado e a e a troca de pacotes foi um pouco menor

com relação ao cenário 1, devido a maior velocidade do usuário móvel simulado. Os dados serão mostrados nos quadros e figuras nas seções 7.2.1 e 7.2.2.

7.2.1 Tempo Cenário 2

Na figura 21 e quadro 8 estão descritos os dados de confiabilidade por processo. Como na seção 7.1.1, ambas ilustram a relação de confiabilidade do tempo com relação densidade de amostras, função de risco, QQ-plot (Quartil-Quartil) e a estimativa de sobrevida das amostras com base na população (Gráfico Kaplan-meier), com base em Jain (1991) e Martins (2011). Com base nos dados pode ser observado que neste cenário as amostras também seguem um padrão de normalidade semelhante à seção 7.1.1. Os dados gráficos mostram que o tempo de simulação do trabalho foi uniforme, porém a proximidade com relação ao tempo estabelecido de 180 segundos está mais afastada da meta que que foi estabelecida no cenário 2.

Figura 21: Gráfico de confiabilidade cenário 2.



Fonte: Elaborada pelo autor, baseado em Silva (2006) e Martins (2011).

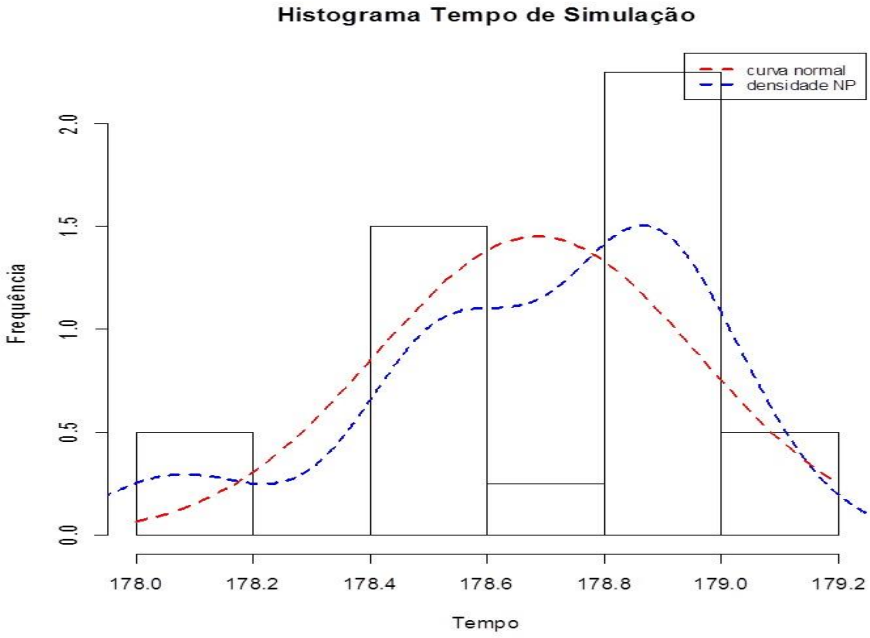
Quadro 8: Dados de confiabilidade por processo.

Índices	*Valor*
MTTF	178,685
Desvio Padrão	0,268020522
Mediana	178,685
1º Quartil	178,5042229
3º Quartil	178,8657771

Fonte: Elaborado pelo autor.

A figura 22 demostra o diagrama de frequência dos intervalos de tempo. O valor da classe de maior frequência está mais afastado para menos do tempo de 180 segundos que foi determinado em cada segmentação para simulação, baseado em Martins (2011).

Figura 22: Histograma cenário 2.



Fonte: Elaborada pelo autor, baseada em Silva (2006) e Martins (2011).

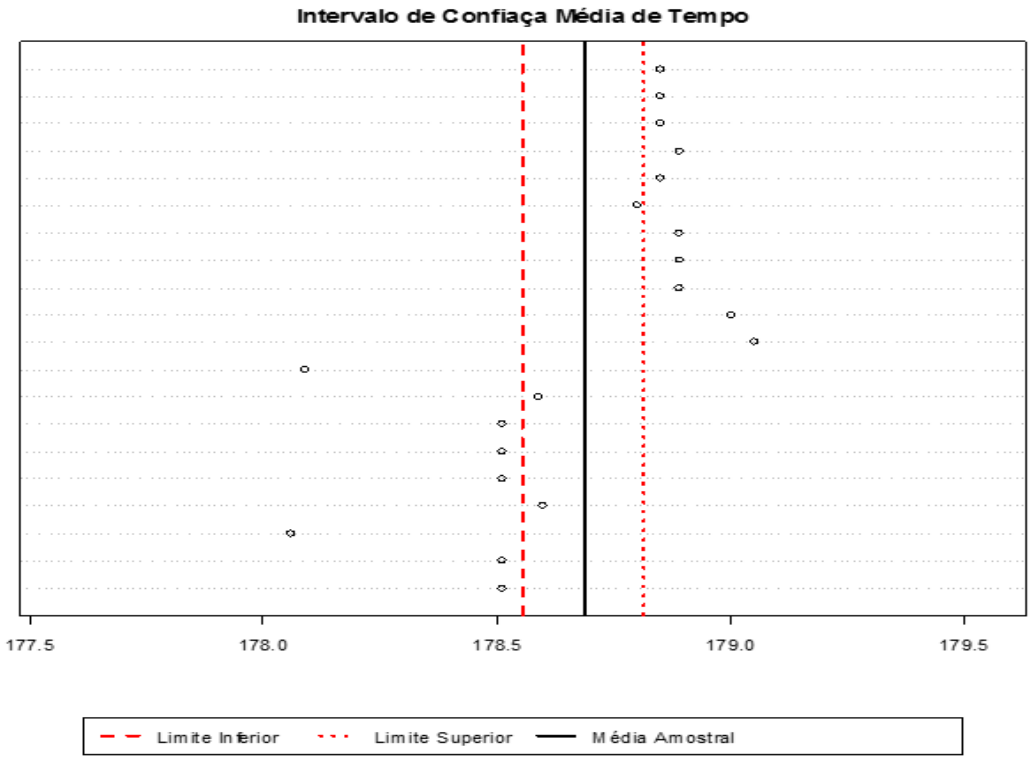
No quadro 9 está resumido os dados estatísticos básicos da média das amostras de tempo do cenário 2. Na análise dos dados foi atribuído nível de confiança de 95%. O grau de liberdade tem o número total de amostra menos um. Tendo portando, dezenove parâmetros estatísticos para serem analisados. Pode ser observado uma média amostral menor que 180 segundos. A figura 23 ilustra a média das amostras e o intervalo de confiança por processo de execução, com base em Jain (1991). Com a média aritmética de 178,685, ficou quase dois segundos abaixo do tempo determinado de 180 segundos.

Quadro 9: Média das amostras de tempo cenário 2.

Informação	*Valor*
Graus de Liberdade	19
Média Amostral	178,685
Desvio padrão amostral	0,274983253
Tamanho da amostra	20
Intervalo de Confiança	95%
Limite Inferior	178,5563039
Limite Superior	178,8136961

Fonte: Elaborada pelo autor.

Figura 23: Intervalo de confiança da média de pacotes Cenário 2.

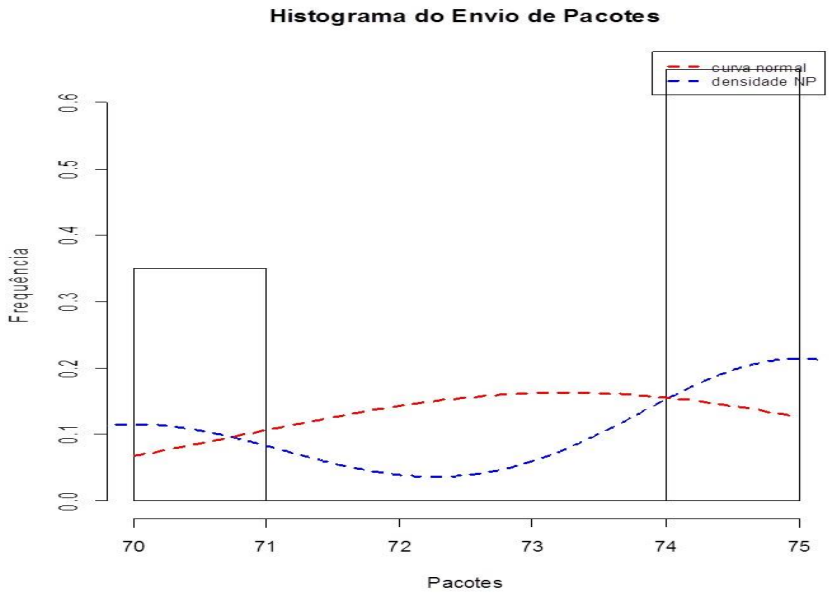


Fonte: Elaborada pelo autor baseada, em Silva (2006) e Martins (2011).

7.2.2 Visão Geral Envio e Recebimento de Pacotes Cenário 2

A base estatística para apresentação dos dados referente ao envio de pacotes de rede coletados para amostra na simulação foi a mesma usada nas seções anteriores, por isso é relevante o fato de que o tempo e o envio de pacotes foram reduzidos pelo aumento da velocidade do nó móvel. A figura 24 mostra que o histograma de pacotes foi estável, porém a troca de pacotes foi reduzida pelo aumento da velocidade.

Figura 24:Histograma de pacotes cenário 2.



Fonte: Elaborada pelo autor, baseado em Silva (2006) e Martins (2011).

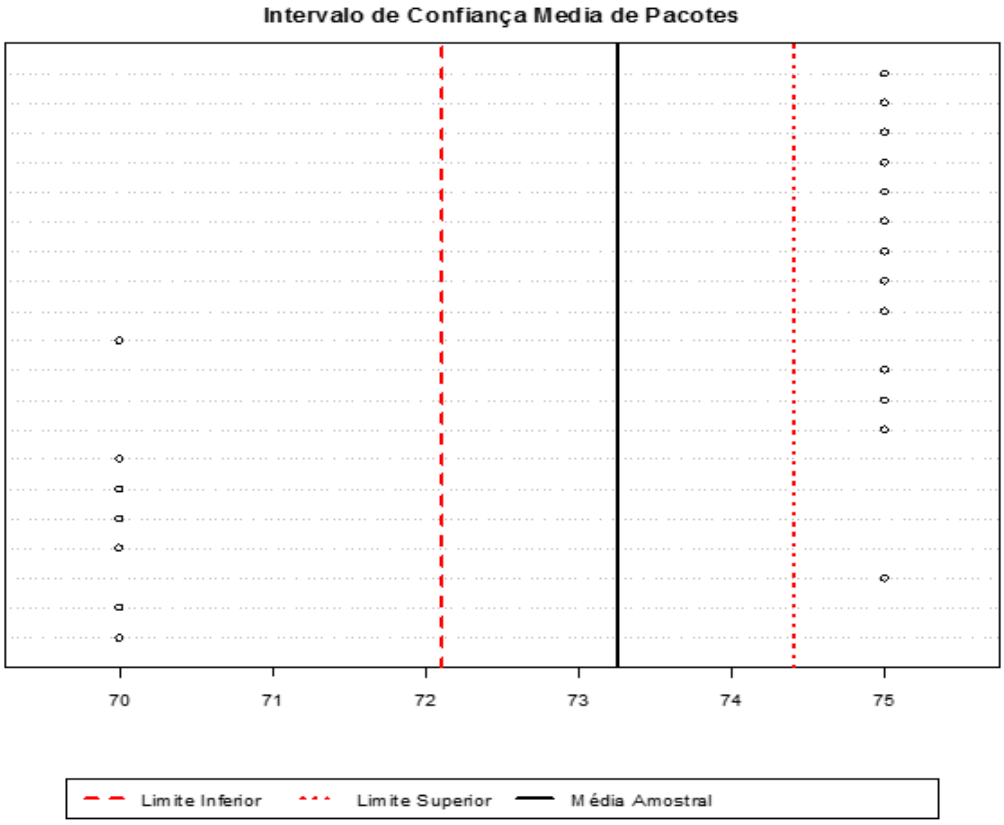
O quadro 10 resume os dados estatísticos básicos da troca de pacotes cenário 2. Na análise dos dados foi atribuído nível de confiança de 95% como no cenário anterior. Com grau de liberdade de Número total de amostra menos um. Tendo portando, dezenove parâmetros estatísticos para serem analisados. A figura 25 ilustra a média das amostras e o intervalo de confiança por processo de execução. A média amostra de 73, 24 pacotes enviado e recebido é estável porém menor que a média de amostras se comparado com o cenário 1 devido a influência da velocidade.

Quadro 10: Dados estatísticos média da troca de pacotes cenário 2.

Informação	*Valor*
Graus de Liberdade	19
Média Amostral	73,25
Desvio padrão amostral	2,446802425
Tamanho da amostra	20
Intervalo de Confiança	95%
Limite Inferior	72,10486122
Limite Superior	74,39513878

Fonte: Elaborado pelo autor.

Figura 25: Intervalo de confiança da média de pacotes Cenário 2



Fonte: Elaborada pelo autor, baseada em Silva (2006) e Martins (2011).

8 CONCLUSÃO

A verificação foi feita comparando este trabalho acadêmico com outros trabalhos relacionados, em que cada ponto de relevância foi fundamentado e referenciado bibliograficamente.

Este trabalho criou uma rede simulada usando o NS3, onde foi verificado se uma malha de rede sem fio poderia dar suporte a um ambiente parcialmente ubíquo. A necessidade de desempenho no tráfego de dados, roteamento e gerenciamento de usuários móveis, não teve a necessidade de ser medido pois estava fora do escopo. A malha de rede sem fio criada neste trabalho forneceu ao usuário móvel conexão constante junto com mobilidade. O que foi testado e analisado neste trabalho serve para abrir portas para estudo na área ubíqua, pois os seguimentos abordados aqui têm muito mais a oferecer e a ser estudado. Sendo assim este trabalho acadêmico pode ser usado como ponto de partida para um estudo mais aprofundado na área da tecnologia ubíqua. O esperado para esse trabalho foi atingido, pois a rede simulada de 7450 Km² deu suporte parcialmente ubíquo com conexão constante e mobilidade a um usuário móvel. A análise poderia ser bem mais incrementada e completa porém, os recursos de simulação são limitados. Um trabalho de magnitude completa de uma rede ubíqua simulada é viável, porém necessita de mais pesquisas na nas áreas, pois são muitos segmentos para serem abordados em um só trabalho acadêmico (DUARTE, 2009; SIEGA 2010; VURAL, 2013; HAZRA, 2015).

O objetivo geral deste trabalho acadêmico, como previsto, foi que as redes *mesh* podem ser usadas para dar suporte de conectividade à computação parcialmente ubíqua pois em 99.60% do tempo total de simulação o nó móvel esteve conectado com alta mobilidade.

A coleta dos dados, processamento e apresentação estão devidamente referenciados teoricamente, e foram baseados em dados de trabalhos semelhantes. O trabalho não é um projeto de âmbito completo em nenhuma das tecnologias que foram abordadas para criação do mesmo e nem tem a pretensão de resolver todos os problemas dessas tecnologias, porém que pode ser útil como ponto de partida para simular ambientes parcialmente ubíquos.

Por não ser um trabalho de caráter totalmente teórico, houve muitas dificuldades na área prática, com relação a simulação e coleta de dados. Uma grande dificuldade foi o grande volume de dados gerado pela simulação. Foi muito difícil de coletar e tratar de forma clara a enorme carga de dados. Para contornar esse problema o trabalho teve que abrir mão de outros aspectos como, QoS e necessidade de alto desempenho no tráfego de dados, roteamento e gerenciamento de usuários móveis. Outro fator importante que possibilitou a coleta dos dados

foi a fragmentação na execução da simulação. A fragmentação da simulação permitiu, dividir o todo em pequenas partes possíveis de coletar e analisar, processar e tratar para que fosse feita uma apresentação de dados aceitável.

Como possíveis trabalhos futuros, pretende-se investigar ainda a conexão transparente das redes ubíquas e a real capacidade de expansão e processamento no desempenho e envio de dados. Estas são áreas de pesquisa que até o presente momento são difíceis de serem recriadas com perfeição nos simuladores existentes. Estes segmentos de pesquisa ficam para trabalhos futuros na área da tecnologia ubíqua.

REFERÊNCIAS

ANDREEV, Kirill; BOYKO, Pavel. IEEE 802.11s Mesh Networking NS-3 Model. Russia Institute for Information Transmission Problems Bolshoy Karetny per. 19, Moscow, Russia. *Workshop on ns3*. 2010.

ASCHENBRUCK,Nils; ERNST, Raphael; GERHARDS-PADILLA, Elmar; SCHWAMBORN, Matthias. Bonnmotion: a mobility scenario generation and analysis tool. Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). Germany. 2010.

BOLSONI, Evandro Paulo; CARDOSO, Paula; DE SOUSA, Carlos Henrique Medeiros. Computação Ubíqua, Cloud Computing e PCL para Continuidade Comunicacional de Desastres, Anais Eletrônicos- Artigos. São Paulo. Nov. 2009.

BAKHT, Humayun et al. Survey of routing protocols for mobile ad-hoc network. International Journal of Information and Communication Technology Research, v. 1, n. 6, 2011.

Breuel, Cristiano Malanga. "Redes em malha sem fios." *Instituto de Matemática e Estatística, USP, http://grenoble. ime. usp. br/movel/Wireless_Mesh_Networks. pdf. Dezembro* (2004).

COVELO, Rui Pedro Figueira; VENDA, Pedro João Lopes. Segurança no Acesso a Sistemas Embebidos. Universidade Técnica de Lisboa Instituto Superior Técnico (licenciatura em Engenharia Eletrotécnica e de Computadores). 2005.

COULOURIS, Dollimore e Kingberg. Sistemas distribuídos: Conceitos e Design. 4. ed: Peason Education, 2005.

Duarte, J. *Escalabilidade, Gerência e Mobilidade para Redes Mesh de Acesso à Internet*. Diss. Dissertação de Mestrado, instituição Universidade Federal Fluminense–UFF, 2008.

FEITELSON, Dror G. From repeatability to reproducibility and corroboration. **ACM SIGOPS Operating Systems Review**, v. 49, n. 1, p. 3-11, 2015.

FONTOURA, Antonio Carré. Análise de protocolos de roteamento em uma rede mesh Baseada em um backbone universitário. Instituto De Ciências Exatas E Geociências Curso De Ciência Da Computação. Universidade de Passo Fundo. 2007.

HAZRA, Swarnali, and S. K. Setua. "Trusted Service Discovery against Identity Spoofing in Ubiquitous Computing Environment." *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*. Springer International Publishing, 2015.

IEEE 802.11s. D1.07. *Draft STANDARD for Information TechnologyTelecommunications and information exchange between system-Local and metropolitan area networks-Specific requirements.* 2007.

JAIN, Raj. The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. Wiley professional computing. Publisher by Wiley. 1991

KUROSE, James; ROSS, Keith. Redes de Compotadores e a Internet: Uma abordagem Top Down. Ed: Peason Education, 2004.

MARTINS, Gilberto de Andrade & DOMINGUES, Osmar. *Estatística Geral e Aplicada*. 4a ed. São Paulo: Atlas. 2011.

NSNAM Network Simulator 3. Disponível em: www.nsnam.org/overview/what-is-ns-3/ . Acesso em abr, 2015.

PATHAK, Parth H.; DUTTA, Rudra. A survey of network design problems and joint design approaches in wireless mesh networks. **Communications Surveys & Tutorials, IEEE**, v. 13, n. 3, p. 396-428, 2011.

PASSOS, Diego; ALBUQUERQUE, Célio V. N. Uma Abordagem Unificada para Métricas De Roteamento e Adaptação Automática de Taxa em Redes em Malha Sem Fio Instituto de Computação. Universidade Federal Fluminense (UFF). 27º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. 2009.

ROOS, Dave. Como funcionam as redes mesh sem fios. Artigo - Publicado em 20 de junho de 2007(atualizado em 03 de janeiro de 2008) Disponível em: http://informatica.hsw.uol.com.br/rede-mesh-sem-fio.htm/. Acesso em: 31 de fev 2015.

RODRIGUES, Nuno José Pereira Farias. Redes mesh sem fio. Tese Mestrado em Engenharia Eletrotécnica e de Computadores - Faculdade de Engenharia da Universidade do Porto. Porto Portugal. 2009.

SAADE, Débora C. Muchaluat; ALBUQUERQUE, Célio V. N.; MAGALHÃES, Luiz

Claudio Schara; PASSOS, Diego; DUARTE, Jairo; VALLE, Rafael. Mesh Network Performance Measurements. *XXV Simpósio Brasileiro de Telecomunicações-SBrT*. 2006.

SANTANA, Luiz Henrique Zambom; DO PRADO, Antonio Francisco; DE SOUSA, Wanderley Lopes. Ubick: Um framework baseado em agentes de software para computação ubíqua. Universidade Federal De São Paulo. São Paulo. 2009.

SIEGA, Marcos De Melo. Monografia de graduação: Integração Das Ferramentas Vanetmobisim e NS-3 Para Simulação De Redes Vanets. Universidade Tecnológica Federal Do Paraná – UTFPR. Paraná. 2013.

SILVA, Ana Alexandrino. Gráficos e Mapas - Representação de informação estatística. Portugal, Ed. Lidel. 2006.

VICENTINI, Cleverton Juliano Alves; ARÁUJO, Roberson Cesar Alves; FONSECA, Mauro Sérgio Pereira. Proposta De Uma Métrica de Roteamento Para Redes Wireless Mesh com Tráfego Voip. Tese de mestrado - Programa de Pós-Graduação em Informática Aplicada-(PPGia). Pontifícia Universidade Católica do Paraná (PUCPR). Paraná. 2010.

VURAL, Serdar; WEI, Dali; MOESSNER, Klaus. Survey of experimental evaluation studies for wireless mesh network deployments in urban areas towards ubiquitous Internet. **Communications Surveys & Tutorials, IEEE**, v. 15, n. 1, p. 223-239, 2013.

APÊNDICES

APÊNDICE A – SCRIPTS USADO

```
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/network-module.h"
#include "ns3/applications-module.h"
#include "ns3/wifi-module.h"
#include "ns3/mesh-module.h"
#include "ns3/mobility-module.h"
#include "ns3/mesh-helper.h"
#include "ns3/netanim-module.h"
#include "ns3/mobility-helper.h"

#include <iostream>
#include <sstream>
#include <fstream>
using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("MeshScript");

class MeshScript
{
public:
  //Iniciar Script
  MeshScript ();
  /// Configurar argumentos de linha de comando
  void Configure (int argc, char ** argv);
  /// Incia a criação do script e declara as variaveis.
  int Run ();
private:
  int        m_xSize;
```

```
  int        m_ySize;
  double     m_step;
  double     m_randomStart;
  double     m_totalTime;
  double     m_packetInterval;
  uint16_t   m_packetSize;
  uint32_t   m_nIfaces;
  bool       m_chan;
  bool       m_trace;
  bool       m_pcap;
  std::string m_stack;
  std::string m_root;
  /// Lista os nós da rede
  NodeContainer nodes;
  /// Lista todos os dispositovos mesh da rede
  NetDeviceContainer meshDevices;
  //Atribui endereço as interfaces
  Ipv4InterfaceContainer interfaces;
  // Instancia o modulo meshHelper
  MeshHelper mesh;
private:
  /// Cria os nós e configura os dados de mobilidade
  void CreateNodes ();
  /// Instala internet nos nos
  void InstallInternetStack ();
  /// Instala as aplicações
  void InstallApplication ();
  /// retorna o diagnóstico e localização na grade dos dispositivos mesh
  void Report ();
};
///Inicializa as variaveis declaradas em Run
```

```
MeshScript::MeshScript () :
  m_xSize (6),
  m_ySize (5),
  m_step (100.0),
  m_randomStart (0.1),
  m_totalTime (180.0),
  m_packetInterval (0.1),
  m_packetSize (1024),
  m_nIfaces (1),
  m_chan (true),
  m_trace (false),
  m_pcap (true),
  m_stack ("ns3::Dot11sStack"),
  m_root ("ff:ff:ff:ff:ff:ff")
{
}
void
MeshScript::Configure (int argc, char *argv[])
{
  CommandLine cmd;
  cmd.AddValue ("x-size", "Numeros de nos na grade. [6]", m_xSize);
  cmd.AddValue ("step",   "Tamanho da distância em mentros dos nos na
grade. [50m]", m_step);
  /*
   * Assim que inicia simulação os nos começão a troca de informação de
localização.
   *
   */
  cmd.AddValue ("start",  "Maximo do atraso aleatorio, em segundos. [0.1 s]",
m_randomStart);
```

```
  cmd.AddValue ("time",  "Tempo de simulação, em segundos [240 s]",
m_totalTime);
  cmd.AddValue ("packet-interval",  "Intervalo entre apcotes UDP e Ping, em
segundos [0.001 s]", m_packetInterval);
  cmd.AddValue ("packet-size",  "Tamanho dos pacotes UDP e ping",
m_packetSize);
  cmd.AddValue ("interfaces", "Número de interfaces de rádio utilizados por
cada ponto de malha. [1]", m_nIfaces);
  cmd.AddValue ("channels",   "Usa diferentes canais de freqüência para
diferentes interfaces. [0]", m_chan);
  cmd.AddValue ("pcap",   "Enable PCAP traces on interfaces. [0]", m_pcap);
  cmd.AddValue ("stack",  "Tipo de pilha de protocolos. ns3::Dot11sStack by
default", m_stack);
  cmd.AddValue ("root", "Endereço MAC dos nos RootMP so protocolo HWMP
", m_root);
  cmd.AddValue ("trace",   "habilitar captura ascii. [0]", m_trace);
  cmd.Parse (argc, argv);
  NS_LOG_DEBUG ("Grade:" << m_xSize << "*" << m_ySize);
  NS_LOG_DEBUG ("Tempo de Simulação: " << m_totalTime << " s");

}
void
MeshScript::CreateNodes ()
{
  /*
   * Cria as coordenadas m_ySize*m_xSize para criar a topologia na grade
   */
  nodes.Create (m_ySize*m_xSize);
  // Configura YansWifiChannel
  YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default ();
  YansWifiChannelHelper wifiChannel = YansWifiChannelHelper::Default ();
```

```
  wifiPhy.SetChannel (wifiChannel.Create ());
  /*
   * O mesh helper instala os atributos de rede
   * Stack crea e instala todos os protocolos necessarios para criação da rede
mesh.
   */
  mesh = MeshHelper::Default ();
  if (m_trace)
  {
    AsciiTraceHelper ascii;
    wifiPhy.EnableAsciiAll (ascii.CreateFileStream ("mesh.tr"));
  }
  if (!Mac48Address (m_root.c_str ()).IsBroadcast ())
    {
      mesh.SetStackInstaller (m_stack, "Root", Mac48AddressValue
(Mac48Address (m_root.c_str ())));
    }
  else
    {
      //Se o root MP não for atribuído, será atribuído um Root Mp
      //isso ocorrerá se não for atribuido um RMP em 11s
      mesh.SetStackInstaller (m_stack);
    }
  if (m_chan)
    {
      mesh.SetSpreadInterfaceChannels (MeshHelper::SPREAD_CHANNELS);
    }
  else
    {
      mesh.SetSpreadInterfaceChannels (MeshHelper::ZERO_CHANNEL);
    }
```

```
  mesh.SetMacType ("RandomStart", TimeValue (Seconds (m_randomStart)));
  // altera o número de interfaces para o padrão mesh do simulador
  mesh.SetNumberOfInterfaces (m_nIfaces);
  // Instala os pretocolos e retona o conteudo do MeshPointDevices
  meshDevices = mesh.Install (wifiPhy, nodes);
  // Configuração da mobilidade cria uma topologia de grade estatica.
  MobilityHelper mobility;
  mobility.SetPositionAllocator ("ns3::GridPositionAllocator",
                                 "MinX", DoubleValue (0.0),
                                 "MinY", DoubleValue (0.0),
                                 "DeltaX", DoubleValue (m_step),
                                 "DeltaY", DoubleValue (m_step),
                                 "GridWidth", UintegerValue (m_xSize),
                                 "LayoutType", StringValue ("RowFirst"));
  mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
  mobility.Install (nodes);


  if (m_pcap)
    wifiPhy.EnablePcapAll (std::string ("mp-mesh"));

}
void
MeshScript::InstallInternetStack ()
{
  InternetStackHelper internetStack;
  internetStack.Install (nodes);
  Ipv4AddressHelper address;
  address.SetBase ("10.1.1.0", "255.255.255.0");
  interfaces = address.Assign (meshDevices);
}
```

```
void
MeshScript::InstallApplication ()
{
  UdpEchoServerHelper echoServer (30);
  ApplicationContainer serverApps = echoServer.Install (nodes.Get (0));
  serverApps.Start (Seconds (0.0));
  serverApps.Stop (Seconds (m_totalTime));
  UdpEchoClientHelper echoClient (interfaces.GetAddress (0), 30);
  echoClient.SetAttribute ("MaxPackets", UintegerValue
((uint32_t)(m_totalTime*(30/m_packetInterval))));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds
(m_packetInterval)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (m_packetSize));
  ApplicationContainer clientApps = echoClient.Install (nodes.Get
(m_xSize*m_ySize-30));
  clientApps.Start (Seconds (0.0));
  clientApps.Stop (Seconds (m_totalTime));
}


void
showPosition (Ptr<Node> nodes, double deltaTime)
{
  uint32_t nodeId = nodes->GetId ();
  Ptr<MobilityModel> mobModel = nodes->GetObject<MobilityModel> ();
  Vector3D pos = mobModel->GetPosition ();
  Vector3D speed = mobModel->GetVelocity ();
  std::cout << "At " << Simulator::Now ().GetSeconds () << " nodes " <<
nodeId
            << ": Position(" << pos.x << ", " << pos.y << ", " << pos.z
            << ");   Speed(" << speed.x << ", " << speed.y << ", " << speed.z
```

```
            << ")" << std::endl;

  Simulator::Schedule (Seconds (deltaTime), &showPosition, nodes,
deltaTime);
}
int
main (int argc, char *argv[])
{
  std::cout.precision (4);
  std::cout.setf (std::ios::fixed);
  double deltaTime = 100;
  std::string traceFile = "/home/joaofaustino/mmg1.ns_movements";
  CommandLine cmd;
  cmd.AddValue ("traceFile", "Ns2 movement trace file", traceFile);
  cmd.AddValue ("deltaTime", "time interval (s) between updates (default
100)", deltaTime);
  cmd.Parse (argc, argv);

  Ptr<Node> n0 = CreateObject<Node> ();

  Ns2MobilityHelper ns2 = Ns2MobilityHelper (traceFile);
  ns2.Install ();

  Simulator::Schedule (Seconds (0.0), &showPosition, n0, deltaTime);

  MeshScript t;
  t.Configure (argc, argv);
  return t.Run ();
}

void
```

```
MeshScript::Report ()
{

///Captura os dados da simulação e joga em um arquivo .xml e reporta no
temimal se houver erro.
  unsigned n (0);
  for (NetDeviceContainer::Iterator i = meshDevices.Begin (); i !=
meshDevices.End (); + + i, + + n)
    {
      std::ostringstream os;
      os << "trace-Analise-" << n << ".xml";
      std::cerr << "Resultado gerado" << n << " coleta salva em " << n << os.str
() << "₩n";
      std::ofstream of;
      of.open (os.str ().c_str ());
      if (!of.is_open ())
        {
          std::cerr << "Erro: Arquivo não pode ser aberto " << os.str () << "₩n";
          return;
        }
      mesh.Report (*i, of);

      of.close ();
    }

}

int
MeshScript::Run ()
{
  CreateNodes ();
```

```
  InstallInternetStack ();
  InstallApplication ();
  Simulator::Schedule (Seconds (m_totalTime), &MeshScript::Report, this);
  Simulator::Stop (Seconds (m_totalTime));

  AnimationInterface anim ("mesh.xml");
  anim.SetConstantPosition (nodes.Get(0), 1.0, 2.0);



  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

APÊNDICE B – SCRIPTS REMANEJAMENTO DOS TRACES

Cenário 1

```
#!/bin/bash

$RANDOM

int x=0
int y=0



while : x=! 40 ;



do
```

```
bm -f mmg1 ManhattanGrid -n 1 -x 400 -y 400 -d 240 -i
$(($RANDOM %5000) +  1)) -R 1 -o 15 -p 0 -m 10000

bm NSFile -f mmg1

~/repos/ns-3-allinone/ns-3-dev$/ ./waf --run scratch/malha-de-rede



for (( y=1; y>0; y++ ));do

scp ~/repos/ns-3-allinone/ns-3-dev$ ./waf --run scratch/ *.xml, *.pcap
~/Dados-Simulação-Tcc-2-Joao-Faustino/Cenario-1-Pedestre/ mkdir
canario-$y

break

done

done

#Cenário 2
#!/bin/bash

$RANDOM

int x=0
int y=0


while : x=! 40 ;
```

```
do


bm -f mmg1 ManhattanGrid -n 1 -x 300 -y 250 -d 180 -i
$(($RANDOM %7450) +  1)) -R 1 -o 15 -p 6 -m 10000

bm NSFile -f mmg1

~/repos/ns-3-allinone/ns-3-dev$/ ./waf --run scratch/malha-de-rede



for (( y=1; y>0; y++ ));do

scp ~/repos/ns-3-allinone/ns-3-dev$ ./waf --run scratch/ *.xml, *.pcap
~/Dados-Simulação-Tcc-2-Joao-Faustino/Cenario-1-Veiculo/ mkdir
canario-$y
break

done
```