

UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Matheus Ronssani de Figueiredo

Implementação do Protocolo de Comunicação DTN
Aplicado para Cidades Inteligentes

Santa Maria, RS, Brasil

2018

Implementação do Protocolo de Comunicação DTN Aplicado para Cidades Inteligentes

Por

Matheus Ronssani de Figueiredo

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Mestre em Ciência da Computação**.

Orientador: Prof. Dr. João Baptista dos Santos Martins

Santa Maria, RS, Brasil

2018

Figueiredo, Matheus Ronssani de
Implementação do Protocolo de Comunicação DTN Aplicado
para Cidades Inteligentes / Matheus Ronssani de
Figueiredo.- 2018.
72 p.; 30 cm

Orientador: João Baptista dos Santos Martins
Dissertação (mestrado) - Universidade Federal de Santa
Maria, Centro de Tecnologia, Programa de Pós-Graduação em
Informática, RS, 2018

1. Cidades Inteligentes 2. DTN 3. Rede de Sensores I.
Martins, João Baptista dos Santos II. Título.

sistema de geração automática de ficha catalográfica da ufsm. dados fornecidos pelo
autor(a). sob supervisão da direção da divisão de processos técnicos da biblioteca
central. bibliotecária responsável paula schoenfeldt vatta cxa 10/1728.

© 2018

Todos os direitos autorais reservados a Matheus Ronssani de Figueiredo. A reprodução de partes ou do todo deste trabalho só poderá ser feita com autorização por escrito do autor.

Endereço: Avenida Maurício Sirostky Sobrinho, n. 165, Santa Maria, RS. CEP: 97020-440.

Fone (0xx)55 98109-0060; E-mail: mfigueiredo@inf.ufsm.br.

Matheus Ronssani de Figueiredo

Implementação do Protocolo de Comunicação DTN Aplicado para Cidades Inteligentes

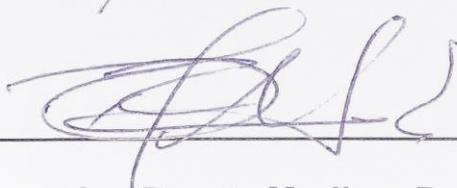
Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Mestre em Ciência da Computação**.

Aprovado em 31 de agosto de 2018



João Baptista dos Santos Martins, Dr. (UFSM)

(Presidente/Orientador)



Roseclea Duarte Medina, Dra. (UFSM)



Sergio Jose Melo de Almeida, Dr. (UCPel)

Santa Maria, RS

2018

RESUMO

Implementação do Protocolo de Comunicação DTN Aplicado para Cidades Inteligentes

AUTOR: Matheus Ronssani de Figueiredo

ORIENTADOR: João Baptista dos Santos Martins

No contexto de Cidades Inteligentes, coletar e distribuir informações relevantes para fornecer os mais variados tipos de serviço é mais que necessário. Para conseguir essa conectividade a rede deve estar preparada para suportar um grande tráfego de dados com as limitações dos protocolos de comunicação e condições físicas, como a grande movimentação dos nós e falhas nos links. Uma abordagem para esse desafio é o uso da arquitetura DTN (*Delay/Disruption Tolerant Networking*), baseada na inserção de uma camada de agrupamento de funcionalidades na pilha do protocolo com a finalidade de garantir a qualidade de serviço da comunicação. Esse trabalho foca em avaliar e modelar uma versão de DTN otimizada para a comunicação em cidades inteligentes.

Palavras-Chave: Redes de Sensores, DTN, Cidades Inteligentes.

ABSTRACT

Implementation of the DTN Communication Protocol Applied to Smart Cities

AUTHOR: Matheus Ronssani de Figueiredo

ADVISORS: João Baptista dos Santos Martins

In a Smart City context, collecting and distributing relevant information to provide every type of service is far beyond necessary. To achieve this connectivity the network have to be prepared to sustain a heavy traffic of data with the limitations of communications protocols and physical conditions, as high node mobility and link failure. An approach to this challenge is the DTN (Delay/Disruption Tolerant Networking) architecture, which based in the “bundle” layer in the stack of the protocol to maintain the QoS (Quality of Service) of the communication. This paper focus on evaluating and modeling a DTN version optimized for smart cities communications.

Keywords: Sensor Network, DTN, Smart Cities.

LISTA DE FIGURAS

Figura 3-1 Comparação entre a pilha do TCP/IP e do DTN. Fonte: Farrel, 2006.	23
Figura 3-2. Exemplo de uma rede tolerante a atrasos. Fonte: Farrel, 2006.	24
Figura 3-3Figura 3 3 Troca de mensagens no roteamento Epidêmico. Fonte: VAHDAT, 2000.	26
Figura 4-1 Formatação do Bloco Primário. Fonte: autor.	38
Figura 4-2 Bloco de carga útil. Fonte: Autor.	39
Figura 4-3 Bits de controle do protocolo. Fonte: Autor.	39
Figura 4-4 Visualização da camada de conversão. Fonte: Autor.	41
Figura 5-1 Modelo de implementação do simulador ONE. Fonte: KERÄNEN, A.2008.	42
Figura 5-2 Interface gráfica do simulador ONE. Fonte: Autor.	43
Figura 6-1 Novo modelo de pilha proposto para cidades inteligentes. Fonte: Autor.	47
Figura 6-2 Fluxograma da metodologia utilizada por este trabalho para a implementação de um protocolo DTN. Fonte: Autor.	51
Figura 6-3 Novo modelo energético utilizado. Fonte: Autor.	57
Figura 6-4 Área utilizada para o cenário de simulação. Fonte: Autor.	58
Figura 7-1 Consumo de energia do protocolo Epidemic. Fonte: Autor.	62
Figura 7-2 Consumo de energia do protocolo PRoPHET. Fonte: Autor.	62
Figura 7-3 Consumo de energia do protocolo Spray-and-Wait. Fonte: Autor.	63

LISTA DE TABELAS

Tabela 6-1 Comparação das funções de cada tipo de nó. Fonte: Autor.	48
Tabela 6-2 Resultados obtidos do protocolo implementador pelo simulador. Fonte: Autor.	54
Tabela 6-3 Resultados obtidos do protocolo proposto. Fonte: Autor.	55
Tabela 6-4 Comparação entre as características dos nós de cada grupo. Fonte: Autor.	59
Tabela 7-1 Performance do Bundle Layer Protocol com 305 nós. Fonte: Autor.	64
Tabela 7-2 Performance do Protocolo Proposto com 305 nós. Fonte: Autor.	64
Tabela 7-3 Performance do Bundle Layer Protocol com 585 nós. Fonte: Autor.	65
Tabela 7-4 Performance do Protocolo Proposto com 585 nós. Fonte: Autor.	66
Tabela 7-5 Performance do Bundle Layer Protocol com 1120 nós. Fonte: Autor.	66
Tabela 7-6 Performance do Protocolo Proposto com 1120 nós. Fonte: Autor.	67

LISTA DE ABREVIATURAS

ARP	Address Resolution Protocol
BAB	Bundle Authentication Block
BSP	Bundle Security Protocol
CONDOR	Command and Control On-the-move Network Digital Over-the-horizon Relay
DD	Direct Delivery
DSL	Digital Subscriber Line
DTN	Delay/Disruption Tolerant Network
ESB	Extension Security Block
FC	First Contact
FTP	File Transfer Protocol
GENA	General Event Notification Architecture
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IEFT	Internet Engineering Task Force
IoT	Internet of Things
ICMP	Internet Control Message Protocol
IP	Internet Protocol
LAN	Local Area Network
MAC	Medium Access Control
MANET	Mobile Ad hoc Network
MQTT	Message Queue Telemetry Transport
M2M	Machine to machine
ONE	Opportunistic Network Environment
OSI	Open Systems Interconnection
PCB	Payload Confidentiality Block
PHY	Physical Layer
PIB	Payload Integrity Block

PROPHET	Probabilistic Routing Protocol using History of Encounters and Transitivity
QoS	Quality of Service
RARP	Reverse Address Resolution Protocol
RFC	Request for Comments
SA	Security-Aware
SDNV	Self-Delimiting Numeric Values
SSDP	Simple Service Discovery Protocol
SSH	Secure Shell
SSP	Scheme-Specific Part
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UPnP	Universal Plug and Play
UTC	Universal Time Coordinated
VANET	Vehicular Ad hoc Network
WAVE	Wireless Access in the Vehicular Environment
XMPP	Extensible Messaging and Presence Protocol

LISTA DE LISTAGENS

Listagem 1 Parâmetros de caracterização de um grupo de nós no ONE. Fonte: Autor.....	44
Listagem 2 Pseudocódigo da validação do protocolo. Fonte: Autor.	53

SUMÁRIO

1. INTRODUÇÃO	14
1.1. OBJETIVOS	14
1.2. JUSTIFICATIVA	15
1.3. ORGANIZAÇÃO	15
2. REVISÃO BIBLIOGRÁFICA	17
2.1. REDES VEICULARES	17
2.1.1. Padrões	18
2.2. CIDADES INTELIGENTES E INTERNET DAS COISAS	18
2.2.1. Desafios da Internet das Coisas	19
3. REDES TOLERANTES A ATRASOS	22
3.1. TIPOS DE CONTATOS	24
3.2. ROTEAMENTO	25
3.2.1. Roteamento Epidêmico	26
3.2.2. Roteamento PROPHET	27
3.2.3. Roteamento MaxProp	28
3.2.4. Spray and Wait	29
3.3. CLASSES DE SERVIÇO	30
3.4. NÓS DE UMA DTN	30
3.5. STORE-AND-FORWARD	31
3.6. TRANSFERÊNCIA DE CUSTÓDIA	32
3.7. APLICAÇÕES DE UMA DTN	33
3.8. MÉTRICAS DE AVALIAÇÃO E PERFORMANCE DE UMA DTN	34
3.9. SEGURANÇA	35
4. BUNDLE LAYER PROTOCOL	36
4.1. O PROTOCOLO	36
4.1.1. Bits de controle	39
4.2. FRAGMENTAÇÃO	40
4.3. A CAMADA DE CONVERSÃO	40
5. O SIMULADOR ONE	42
5.1. CARACTERÍSTICAS DOS NÓS	44
5.2. MODELOS DE MOVIMENTO	45
5.3. ROTEAMENTO	45
6. NOVO PROTOCOLO	46

6.1. A PROPOSTA	46
6.1.1.A pilha do protocolo.....	47
6.1.2.Fragmentação	49
6.2. METODOLOGIA	49
6.2.1.VALIDAÇÃO	52
6.2.2.O CENÁRIO DE TESTE.....	58
7. RESULTADOS.....	61
7.1. RESULTADOS DE ENERGIA	61
7.2. RESULTADOS DE PERFORMANCE.....	63
8. CONCLUSÃO E TRABALHOS FUTUROS.....	68
8.1. TRABALHOS FUTUROS.....	69
9. REFERÊNCIAS BIBLIOGRÁFICAS.....	70

1.INTRODUÇÃO

Cada dia uma enorme quantidade de informações é coletada e oferecida às pessoas através de seus dispositivos. Devido ao grande volume de dados, grande parte dessas informações é apresentada de maneira bruta ou até incompreensível. Logo, é preciso desenvolver meios de filtrar, organizar, espalhar e utilizar essas informações de maneira inteligente. Nesse aspecto surge o conceito de *Smart Cities*, que são definidas através de uso de informações-chaves para os sistemas centrais da cidade, focando no bem-estar do cidadão (KEHUA et al, 2011).

Nesse cenário fica evidente a disseminação de objetos conectados aos mais diversos dispositivos de rede, como televisores, automóveis e sistemas de iluminação. Essa onda da computação ubíqua é chamada de Internet das Coisas (IoT, do inglês *Internet of Things*), que visa um modelo onde qualquer objeto, pessoa ou evento pode ser uma fonte de dados em potencial para ser acessada, processada e utilizada para uma melhor experiência (TAN AND WANG et al, 2010).

1.1.OBJETIVOS

O principal objetivo deste trabalho é explorar o protocolo DTN (*Delay/Disruption Tolerant Networking*) de comunicação entre dispositivos para uma rede urbana utilizada em um cenário de Cidades Inteligentes. Para isto serão necessárias alterações deste protocolo, tendo em vista que o mesmo é normalmente utilizado para aplicações em comunicação de satélites e seu paradigma tem sido utilizado em aplicações com redes desafiadoras em cenários onde não há uma rede estruturada em cidades ou ambientes de campo.

1.2.JUSTIFICATIVA

Com a difusão em massa da internet das coisas a realidade das cidades inteligentes está cada vez mais presente na vida das pessoas, e com isso o tamanho e a complexidade das redes aumentam conjuntamente com o número de dispositivos que começam a fazer desse novo modelo de computação (HRIBERNIK et al. 2011).

Os protocolos utilizados em redes convencionais, como o TCP/IP, acabam sofrendo uma série de dificuldades que surgem com o novo modelo de computação ubíqua que cresceu com a quantidade de dispositivos móveis existentes atualmente. Grandes atrasos, interrupções frequentes e a impossibilidade de garantir um caminho de ponta-a-ponta entre os dispositivos acabam afetando a performance dos protocolos de comunicação padrão (FARREL et al. 2006).

Visando as necessidades de uma cidade inteligente em manter uma conectividade estável para poder fornecer os diversos serviços previstos, é possível utilizar a arquitetura de protocolos DTN para atingir um nível de qualidade de serviço (QoS) da rede aceitável aliando uma melhor performance na comunicação entre os dispositivos bem como uma maior eficiência energética.

1.3.ORGANIZAÇÃO

A dissertação encontra-se organizada conforme a descrição a seguir.

O capítulo 1 apresenta uma Introdução ao tema, define os Objetivos e a Justificativa para este trabalho.

No capítulo 2 é feita uma Revisão Bibliográfica sobre temas relevantes a esse trabalho, apresentando conceitos e algumas aplicações existentes. É feita

uma breve introdução a temas que serão utilizados na conceituação dos temas abordados nos capítulos seguintes.

No capítulo 3 é feita uma revisão sobre redes tolerantes à atrasos e desconexões, *DTN*. São apresentados os conceitos, características, trabalhos já realizados na área, bem como protocolos de roteamento, métricas de performance, segurança, entre outros.

O capítulo 4 apresenta o *Bundle Layer Protocol* em maiores detalhes, bem como os modelos de frame e o funcionamento do protocolo. Também é apresentada a arquitetura proposta pela RFC 5050.

No capítulo 5 é apresentado o simulador ONE, utilizado no desenvolvimento desse trabalho.

No capítulo 6 é apresentada a proposta de um protocolo, objetivo desse trabalho, bem como metodologia e o ambiente utilizado no desenvolvimento, implementação e validação do protocolo DTN. Neste capítulo é apresentada uma proposta de contribuição para melhorar o rendimento do protocolo.

Os resultados são apresentados no capítulo 7, onde são comparados com o desempenho de outro protocolo no mesmo ambiente.

Finalmente, no capítulo 8 são apresentadas as conclusões dessa dissertação e a proposição de trabalhos futuros.

2.REVISÃO BIBLIOGRÁFICA

Este capítulo contém a revisão dos conceitos envolvidos neste trabalho como por exemplo, redes de computadores, cidades inteligentes, redes tolerantes a atrasos e redes de sensores em cidades inteligentes. Um enfoque maior será dado ao protocolo de comunicação DTN a ser trabalhado nesta dissertação. Ao final deste capítulo também é apresentada uma modelagem da camada de serviços relacionado ao trabalho.

2.1.REDES VEICULARES

Um caso específico de redes *ad hoc*, são as redes veiculares, pois possuem algumas características em comum, como por exemplo a constante mudança em sua topologia devido a mobilidade dos nós, o que pode acarretar em frequentes desconexões. As *Vehicular Ad hoc Networks* (VANETs), surgem da aplicação dos princípios de uma *Mobile Ad hoc Network* (MANET), porém pelo fato de seus nós e estações serem compostos por veículos, elas possuem alguns padrões de mobilidades particulares (BAUMANN, 2004).

Esse conjunto de nós pode ser influenciado por diversas variáveis, como rotas, horários, condições climáticas e a posição dos outros nós. Para essas características existem dois tipos de avaliação das VANETs, modelos microscópicos, que fazem uso de velocidades individuais, manobras, medidas de segurança entre os veículos, pedestres, entre outros e existem as abordagens macroscópicas, que modelam o tráfego através de fluxos e padrões (BAUMANN, 2004).

Uma característica importante das VANETs é fonte de energia, que devido a capacidade das baterias automotivas que são constantemente recarregadas, acaba tornando-se praticamente ilimitada.

2.1.1.Padrões

A padronização das redes veiculares ocorreu em 2004, o IEEE iniciou um grupo de trabalho voltado a redes veiculares dentro do IEEE 802.11, o padrão é conhecido como IEEE 802.11p WAVE. *Wireless Access in the Vehicular Environment* (WAVE) é obtido através da composição de um conjunto de documentos, entre eles:

- IEEE P1609.1: padrão que controla a comunicação entre as duas unidades que auxiliam o fluxo de dados pela rede. Unidades de Acostamento, estações fixas ao longo da rodovia, e as Unidades de Bordo, unidades acopladas aos veículos.

- IEEE P1609.2: responsável pela segurança. Define os processos de mensagens e protocolos de segurança que deverão ser realizados.

- IEEE P1609.3: define os serviços das camadas de rede e transporte através de um plano de gerenciamento, onde são realizados o plano de dados, serviços de comunicação e as configurações da rede.

- IEEE P1609.4: define os canais da arquitetura.

- IEEE 802.11.a: Padrão de redes veiculares locais.

- IEEE 802.11.p: baseado no padrão IEEE 802.11.a, define as responsabilidades das camadas físicas e o controle de acesso (MAC).

A arquitetura WAVE opera em faixa de frequências próxima à alocada as redes veiculares militares.

2.2. CIDADES INTELIGENTES E INTERNET DAS COISAS

A Internet das Coisas é conceituada como um conjunto de coisas, não operadas por seres humanos, conectadas à internet. A forma mais comum de comunicação via Internet atual é por interação humana, porém com a futura aprendizagem de máquina a máquina (M2M, do inglês *Machine to Machine*), será possível automatizar dispositivos e comunicações, fornecendo então conectividade ampla a todos objetos (WAHER, 2015).

Conexão autônoma, troca de dados entre dispositivos, segurança e aplicações para o dia a dia nas cidades, esses são conceitos básicos da IoT, que visa incorporar a inteligência da máquina em objetos conectados para se comunicarem, trocar informações, tomar decisões e fornecer serviços para a população de uma cidade. Serviços esses que podem ser desde fornecer internet para os habitantes, bem como automatizar serviços de saúde, iluminação, segurança, entre outros (FAN e CHEN, 2010).

2.2.1.Desafios da Internet das Coisas

Com o objetivo de incorporar todos dispositivos móveis e implantar micro controladores em objetos diários, a IoT apresenta algumas dificuldades, entre as quais é muito importante citar a garantia de segurança e a padronização dos protocolos. Como ainda não existe protocolo padrão definido para todas aplicações, visto que os protocolos existem visam ser eficientes em um foco de problemas, existe um grande número de protocolos utilizados para IoT aplicadas aos pequenos dispositivos como MQTT, HTTP, UPnP, XMPP, entre outros, porém cada um com uma aplicabilidade restrita (WOODSIDE CAPITAL PARTNERS, 2015).

2.2.1.1.MQTT

Message Queue Telemetry Transport (MQTT) é um protocolo que faz uso de agente de mensagens, baseado no método PUBLISH-SUBSCRIBE, onde cliente, ou *Subscriber*, informam interesse em mensagens de um tipo ou classe, então agentes intermediários encarregam-se de filtrar as mensagens enviadas pelos geradores de mensagens, ou *Publisher*, responsáveis por retirar informações de eventos (HIVEMQ, 2015).

Este protocolo é utilizado principalmente para a disseminação de informações já que está acessível a todos dispositivos conectados à rede, já

que não é endereçado via IP, evitando que uma rede inteira seja tratada como um único cliente através de um único endereço. Pode ser adaptado para possuir níveis diferentes de qualidade de serviço, o primeiro onde as mensagens são enviadas em *multicast* para todos assinantes, sem confirmação de envio e então sem possibilidade de reenvio em caso de perda ou erro. Um segundo nível onde o *Publisher* espera a confirmação do assinante sobre o recebimento da mensagem, e caso ocorra alguma falha, podendo ou não ser reenviadas mediante a solicitação do assinante. E um terceiro nível onde a entrega da mensagem é assegurada através de uma confirmação de cada assinante, sendo requisitada pelo usuário final ou não (WAHER,2015).

2.2.1.2.HTTP

O *Hypertext Transfer Protocol* (HTTP), protocolo amplamente utilizado para navegação web que se comunicam com servidores, é também utilizado para a comunicação M2M em serviços de automação industrial, por ser de fácil acesso e uso. O HTTP baseia-se em solicitação e resposta, onde um pedido é efetuado por um cliente solicitando informações a um servidor, então o servidor, após estabelecida uma conexão envia as informações conforme solicitadas. Essa conexão é feita através do TCP com o endereçamento do IP, modo mais comum utilizado em redes Ethernet.

Um protocolo normalmente utilizado na IoT para adicionar uma comunicação ou suporte web para publicação final dos dados obtidos de uma rede. O protocolo se torna uma boa opção para manipulação e apresentação dos dados e eventos, entretanto a necessidade de sincronização, declarações de bloqueio e conexões contínuas acabam impactando na obtenção de dados momentâneos por diversos dispositivos simultaneamente (KUROSE E ROSS, 2016).

2.2.1.3.UPnP

Universal Plug and Play (UPnP), é um protocolo ou arquitetura que se baseia na utilização do *Simple Service Discovery Protocol* (SSDP), um protocolo para varredura e descoberta de dispositivos na mesma rede, e o *General Event Notification Architecture* (GENA), um protocolo que serve para notificações de eventos e subscrições de eventos. Ambos foram baseados no HTTP, e servem para fazer uma varredura nos dispositivos da mesma rede, para avaliar seu estado, detectar conexões ou desconexões e notificar sobre serviços disponíveis. O UPnP é usado em sua maioria por dispositivos eletrônicos ativados pela rede, como alto-falantes, dispositivos de áudio e vídeo e aparelhos Smart (WAHER, 2015).

2.2.1.4.XMPP

O *Extensible Messaging and Presence Protocol* (XMPP) é um protocolo padronizado pela IETF, foi projetado para mensagens instantâneas em aplicativos de tempo real, onde as mensagens são compostas por pequenas informações para serem enviadas com o menor atraso possível.

Baseado no mesmo sistema de PUBLISH-SUBSCRIBER, assim como o MQTT, ele também apresenta um padrão chamado REQUEST-RESPONSE, padrão de comunicação ponto a ponto que permite que duas aplicações ou servidores conversem entre si através de um canal bidirecional. Possibilitando assim uma maior delegação, precisão e qualidade nas mensagens enviadas aos usuários finais (XMPP COMMUNITY, 2002).

3. REDES TOLERANTES A ATRASOS

Inicialmente proposto para comunicação de satélites, o paradigma *Delay Tolerant Network* (DTN) fornece comunicação em situações que apresentam redes desafiadoras, que possuem conectividade intermitente, atrasos longos e variados além de uma alta taxa de erros (SCOTT, 2005). Essas redes apresentam características opostas as redes comuns que fazem o uso do TCP/IP, que apresentam conectividade contínua, baixa taxa de erros, atrasos pequenos e pouca perda de pacotes.

O paradigma DTN prevê a adição de uma nova camada na pilha da internet, chamada de camada de agregação (*Bundle Layer*), cujo objetivo é prover os serviços necessários para complementar os protocolos utilizados nas redes comuns. Podemos ver na figura 3-1 a diferença entre a pilha do protocolo TCP/IP e do protocolo DTN, onde é feita a inserção de uma 6ª camada, a chamada camada de agregação, onde são implementados todos os serviços oferecidos pelo protocolo. Essa nova camada é igual a todos os dispositivos da rede, enquanto as outras camadas podem ser diferentes entre si.

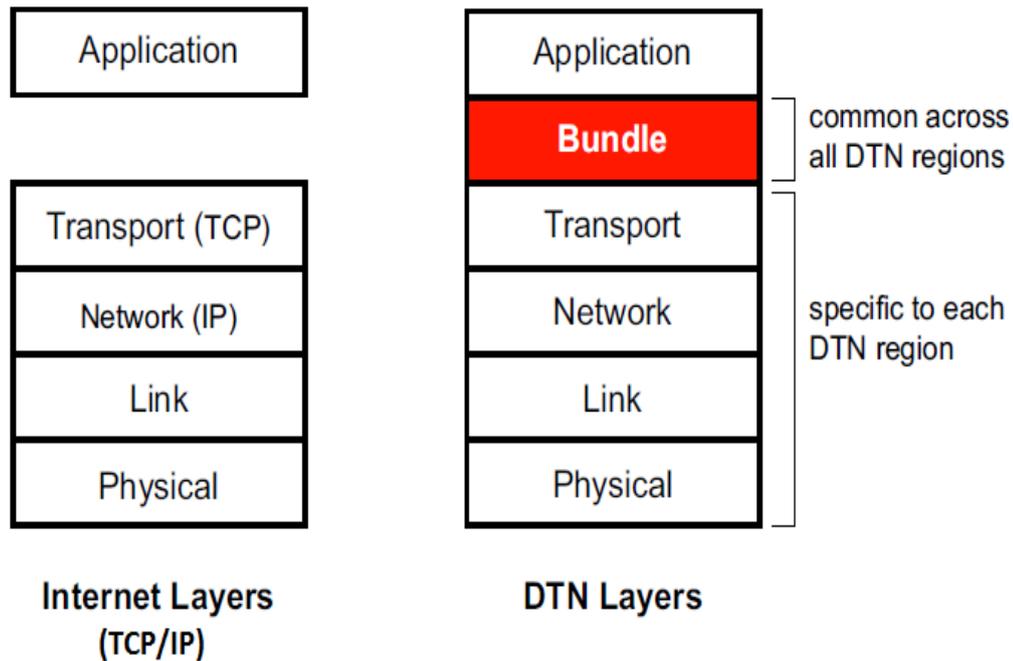


Figura 3-1 Comparação entre a pilha do TCP/IP e do DTN. Fonte: Farrel, 2006.

Como pode-se ver na figura 3-2, alguns exemplos de adversidades que redes de comunicação estão suscetíveis. Primeiramente um caso onde há uma desconexão entre nós que compõem as rotas entre o ponto inicial (*Source*) e o nó final (*Destination*). Abaixo é exemplificado na mesma rede o caso de atrasos (*Delays*) na rede, onde o caminho de nó inicial ao final é de algumas horas, e o caminho do final ou inicial acaba levando dias para chegar a nós intermediários. Ao fim exemplifica-se uma rede com uma grande taxa de erros, onde há uma grande perda de pacotes ou a entrega de pacotes que sofreram alguma alteração devido a falhas na comunicação.

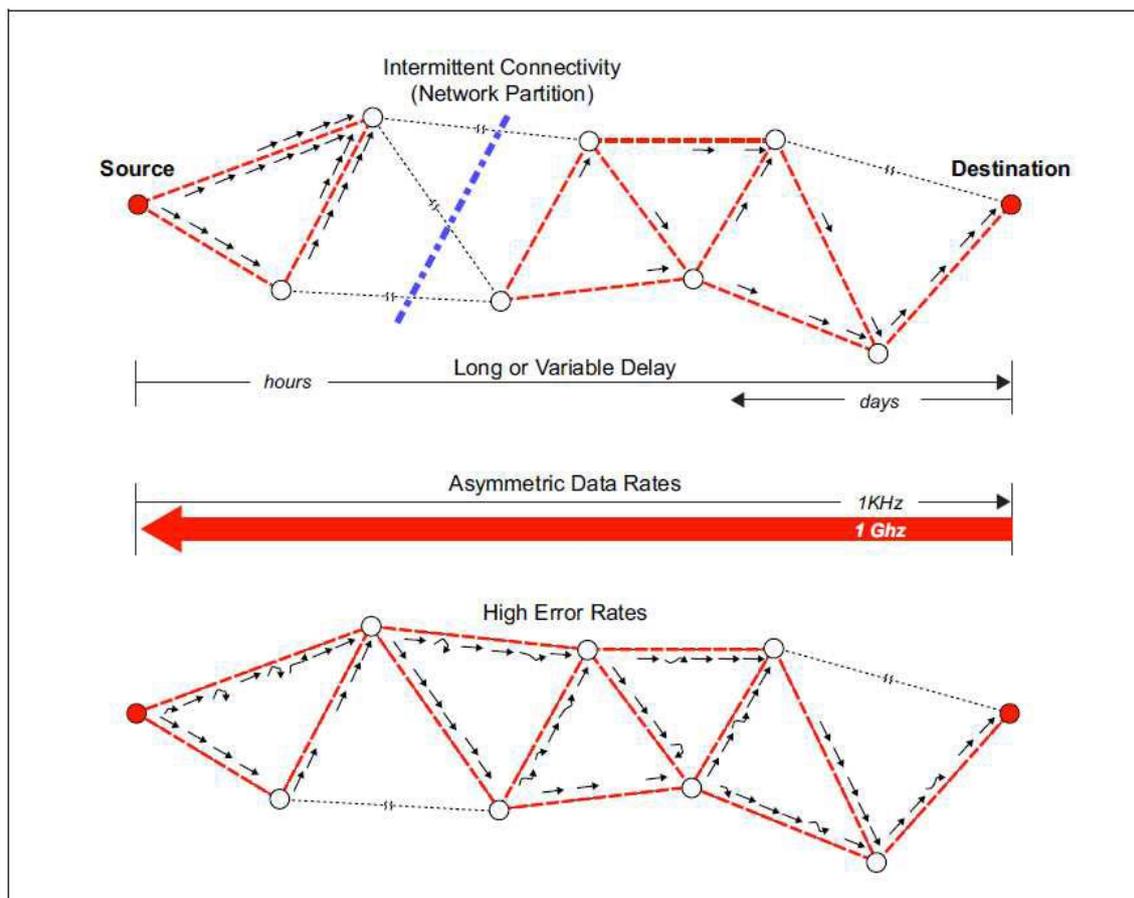


Figura 3-2. Exemplo de uma rede tolerante a atrasos. Fonte: Farrel, 2006.

3.1. TIPOS DE CONTATOS

Uma das características de uma rede DTN é o fato de não se pode garantir que a todo momento todos os nós sejam alcançáveis, mas em algum momento esses nós podem ser contatados por alguns instantes, dessa maneira, um conceito avaliado no paradigma DTN são os tipos de contato. A transferência de mensagens está ligada diretamente ao tipo de contato, já que quando dois nodos se encontram, o contato permite a transferência ocorrer. A arquitetura DTN classifica os contatos em: programados, persistentes, sob demanda, previsíveis e oportunistas, cada um deles explicado a seguir (CERF, 2007).

- Programado: é definido programado quando uma troca de informações pode ser previamente calculada, sabendo-se o horário e a duração da comunicação. Como por exemplo a rota de dois satélites, que possuem órbitas previamente calculadas, mantém contato programados periodicamente. É considerado como um contato intermitente.

- Persistente: são contatos que sempre estão disponíveis, onde não há necessidade de inicialização ou varredura de possíveis nós para contato. Redes cabeadas ou conexões *Digital Subscriber Line* (DSL).

- Sob Demanda: necessita de instanciamento prévio, então passa a operar semelhante a um contato persistente até encerrado. Comumente utilizado em redes de sensores, onde um nó central envia mensagens do tipo *Wake* para inicializar os nós vizinhos.

- Previsível: cada nó mantém uma tabela, com dados sobre contatos anteriores, para então fazer previsões sobre futuros contatos, com horário e duração. Essas previsões possuem um grau de incerteza elevado já que, contando com um certo nível de segurança, é arriscado prever a mobilidade de nós que foram contatos apenas uma ou duas vezes. (OLIVEIRA, 2009).

- Oportunista: ocorre quando dois nós efetuam um encontro não programado, sendo então possível efetuar uma transferência de dados. Comum entre redes que possuem comunicação *wireless*, seja via interface *Bluetooth*, infravermelho, entre outros.

3.2.ROTEAMENTO

As redes DTN surgiram como forma de conectar redes que apresentavam desafios para as estruturas convencionais, como atrasos extremamente longos e desconexões frequentes. Juntamente com as desconexões surge o desafio da topologia da rede variar muito ao longo do tempo. Logo é possível de presumir que dois nós específicos da rede não tenham um caminho fim-a-fim entre eles, e dessa forma nunca sejam conectados.

Essas dificuldades resultam na necessidade de uma estratégia de roteamento eficiente, embora os desafios tornem essa tarefa complicada. De maneira geral, as informações de uma rede e sua topologia interferem diretamente sobre a performance do roteamento. Porém em redes oportunísticas esse conhecimento é complexo de se obter. Assim é necessária ser feita uma troca entre performance e conhecimento da rede (JAIN, 2004).

Diferentemente do padrão de transferências da internet o roteamento do protocolo DTN é concluído na camada de agregação (*Bundle Layer*), enquanto as outras camadas possuem o mesmo comportamento, controlando os *hops* intermediários, apenas a camada de agregação conhecerá o destino final do pacote através da DTN.

3.2.1. Roteamento Epidêmico

Nesse roteamento as mensagens são enviadas pela rede de uma maneira semelhante a um vírus, de onde vem seu nome. Um nó com a mensagem, gerada ou recebida, armazena-a no buffer local, para que possa ser encaminhada em um momento posterior. Esse nó é então chamado infectado. Um nó apenas pode ser infectado caso ainda não tenha recebido a mensagem.

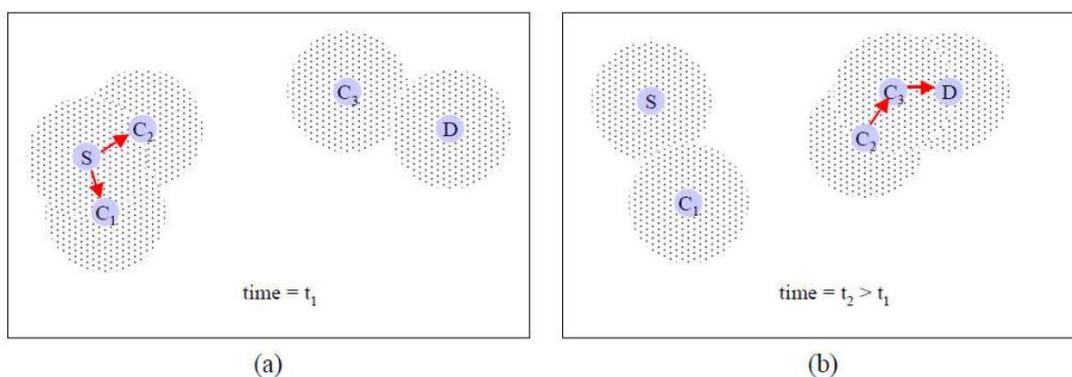


Figura 3-3 Troca de mensagens no roteamento Epidêmico. Fonte: VAHDAT, 2000.

Um nó infectado é curado uma vez que tenha repassado a mensagem ao nó de destino. Assim ele também é considerado imune e não mais receberá ou encaminhará a mensagem.

Na figura 3-3 apresenta um exemplo do roteamento epidêmico, onde o nó inicial S, gera uma mensagem e envia para os nós com quem tem contato, C1 e C2, é possível observar que no momento inicial t_1 não há caminhos entre o inicial e o nó destino D. Em um momento subsequente, o nó C2, encontra-se no alcance do nó C3 e então “infecta” ele com a mensagem, que por sua vez encaminha a mensagem para o nó final, completando assim o roteamento.

Após um determinado intervalo de tempo, o espaço em buffer e o número de cópias da mesma mensagem na rede acabam ocasionando a possibilidade da perda de mensagem por falta de espaço ou uma sobrecarga de mensagens na rede que geram um atraso muito grande. Caso maior que o tempo de vida da mensagem, fará que mensagens comecem a ser descartadas. Esses fatores acabam levando a uma baixa escalabilidade da rede ou a uma rede com uma baixa taxa de probabilidade de entrega de mensagens (VAHDAT, 2000).

3.2.2. Roteamento PROPHET

Uma solução encontrada para a baixa escalabilidade e a alta utilização de recursos do roteamento epidêmico foi o *Probabilistic Routing Protocol using History of Encounters and Transitivity* (PROPHET). Que embora ainda utilize um mecanismo parecido ao epidêmico, é efetuado um controle por intermédio de uma “previsibilidade de entrega”, $P(a, b) \in [0,1]$, em cada nó a para cada destino b . Essa informação é referente a probabilidade de um nó a entregar uma mensagem ao nó b .

Toda vez que os nós a e b se encontram, o valor de $P(a, b)$ é incrementado. Caso os nós não se encontrem frequentemente esse valor decai com o tempo. Essa probabilidade possui uma propriedade transitiva, que se baseia na premissa: se um nó a encontra diversas vezes um nó b , e esse nó

frequentemente encontra um nó c , então as chances de uma mensagem enviadas por a chegarem ao nó c serão altas se enviadas pelo nó b (LINDGREN, 2003).

Semelhante ao roteamento epidêmico, quando um grupo de nós entram em contato, eles atualizam suas tabelas, que contém a informação da previsibilidade. Quando um nó recebe essa informação do vizinho, ele calcula a previsibilidade de entrega para cada mensagem que ainda não possui armazenada em sua memória estável. Em seguida, para cada mensagem, o nó compara a probabilidade indicada na sua lista com a probabilidade indicada na lista recebida do vizinho.

Essa comparação é realizada para verificar qual dos dois nós possui a maior probabilidade de entrega. Feito isso, devem ser realizados três procedimentos. Primeiramente, o nó deve enviar um pedido das mensagens não armazenadas que possuem uma maior probabilidade de serem entregues através dele. Em seguida, recebe o pedido de mensagens do vizinho e as envia. E por último, apaga do buffer todas as mensagens cujas probabilidades de entrega sejam maiores quando a entrega é realizada através dele.

3.2.3.Roteamento MaxProp

O algoritmo *MaxProp* é baseado em replicação, ou *flooding*. Nesse roteamento, sempre que um contato é feito, todas mensagens que não surgirem desse contato serão replicadas e transferidas se possível. Esse roteamento determina quais mensagens devem ser enviadas e quais devem ser descartadas através de uma fila de prioridades baseada no tempo de vida e na possibilidade de um caminho através dos nós que já foram conectados até o nó destino. Para que essa estimativa seja calculada é utilizado um vetor contendo o número de nós existentes na rede, e cada vez que dois nós se encontram, é atualizado o valor de estimativa de conexão entre os nós, e é então repassado para seus contatos o devido valor atualizado.

3.2.4.Spray and Wait

Os protocolos baseados em utilidade, como a previsibilidade do PROPHET, trouxeram uma abordagem para reduzir o *overhead* que o *flood* de mensagens do roteamento epidêmico gera e para melhorar sua performance, que consiste em encaminhar uma cópia com uma probabilidade $p < 1$, baseada na utilidade de cada nó na rede para o nó de destino. Apesar dessa abordagem apresentar uma performance melhor e possuir melhores decisões de encaminhamento do que o roteamento epidêmico (LINDGREN, 2003), esses protocolos ainda são baseados em *flood*. Além disso, ainda há o dilema de se escolher um limite para a utilidade. Um limite muito baixo e o protocolo se comportará como epidêmico (puramente *flood*); muito alto e a latência aumentará significativamente e a probabilidade de mensagem alcançar seu destino diminuirá.

O protocolo Spray and Wait se propõe a reduzir significativamente o *overhead* de transmissão dos esquemas baseados em *flooding* e apresenta performance melhor no que diz respeito ao atraso de entrega na maioria dos cenários, além de não utilizar qualquer informação da rede (SPYROPOULOS, 2005). Para isso, o protocolo dissemina um número de cópias prefixado da mesma mensagem para outros nós na rede e aguarda até que uma dessas cópias atinja o destino.

O Spray and Wait consiste de duas fases. Na primeira (fase *Spray*), para cada mensagem originada em um nó, n cópias são geradas e então espalhadas para outros nós para serem entregues ao nó de destino. Um nó é considerado ativo quando possui $n > 1$ cópias da mensagem em seu buffer e, portanto, ainda está na fase de Spray. Quando um nó ativo a encontra outro nó b , a entrega para b $F(n)$ cópias da mensagem e retém para si as outras cópias, calculadas por $n - F(n)$. F é uma função, baseada na chance de um nó a encontrar-se com um outro nó b para que possa haver uma troca de mensagens, que define o processo de espalhamento. Quando as cópias são pulverizadas através dos nós de modo que um único nó mantém apenas uma única cópia da mensagem, o algoritmo entra em sua segunda fase (fase *Wait*). Nessa fase, cada um dos n nós com uma única cópia só pode encaminhar a mensagem para o seu destino final.

3.3. CLASSES DE SERVIÇO

Uma rede DTN pode prover quatro classes de serviço em uma comunicação. Estes serviços são independentes das outras camadas da rede. Cada uma dessas classes refere-se à uma condição ou característica das mensagens da rede, como a prioridade de uma mensagem, duração de uma mensagem, sinais de confirmação de entrega e responsabilidades de um nó.

- Transferência de custódia: repassa a responsabilidade sob uma mensagem para outro nó, que não seja o de origem.

- Sinal de confirmação: confirmação do destino para a origem, sobre o recebimento de uma mensagem. Utilizado apenas em alguns casos, como a transferência de custódia.

- Tempo de vida: delimita se a mensagem possui um tempo de vida útil, antes de ser apagada pela rede.

- Prioridade de entrega: Baixa, Normal ou Alta, defini a prioridade de envio e recebimento pelos nós de uma mensagem.

3.4. NÓS DE UMA DTN

Em uma rede DTN um nó é qualquer dispositivo com a camada de agregação presente. Todos os nós podem apresentar até três funções, origem, transmissor ou destino. Todos os tipos possuem a capacidade de receber transferências de custódia e possuem armazenamento persistente.

- Origem: um nó apenas responsável por gerar e enviar mensagens para a rede.

- Destino: nó responsável por receber uma mensagem e repassar suas informações para a camada de aplicação.

- Transmissor um nó transmissor tem como tarefa reenviar uma mensagem para um nó seguinte, porém eles apresentam dois subtipos:

- Roteador: esse nó transmite as mensagens para outros nós que funcionam na mesma sub rede, por exemplo, se ele receber uma mensagem cujo protocolo de transporte é o TCP, ele apenas poderá enviar para outro nó que opera na mesma rede também com o protocolo TCP.
- Gateway: este tipo de nó possui a capacidade de comunicar duas sub redes diferentes ao retransmitir a mensagem. Se ele receber uma mensagem vinda de um nó operando com o protocolo TCP, ele poderá converter essa mensagem, e enviá-la a nós que utilizam o protocolo UDP.

3.5. STORE-AND-FORWARD

Um dos problemas associados a conexões intermitentes, com longos atrasos, ou alta taxa de erros é a falha na entrega a destinação final. A arquitetura DTN soluciona esse problema através da técnica de troca de mensagens chamada Store-And-Forward, ou armazene-e-repasse, onde blocos inteiros ou fragmentos de uma mensagem são enviados de um *storage* de um nó para outro *storage* em outro nó, até eventualmente chegar ao seu nó destino.

Enquanto roteadores normais usam memória de curto prazo para armazenar as mensagens, como *buffers* ou memória *flash*, já que o tempo de repasse é muito curto, nós de uma rede DTN usam armazenamento persistente, visto que o tempo de repasse pode ser muito longo devido a fatores como a comunicação com o próximo nó não estar disponível por algum tempo, mensagens assíncronas entre os nós ou a necessidade de reenviar uma mensagem em caso de erro ou falha de um nó subsequente.

3.6. TRANSFERÊNCIA DE CUSTÓDIA

A transferência de custódia, é um caso especial do *Store-and-Forward*. Um serviço previsto em uma rede DTN, já que protocolos de transporte não operam de ponta-a-ponta e em uma rede DTN nem sempre pode-se garantir esse caminho. Esse serviço ajuda a garantir a confiabilidade na entrega na camada de agregação.

Essa transferência ocorre da seguinte maneira: o nó inicial faz uma solicitação ao próximo nó que será responsável pela custódia, não necessariamente o próximo nó que receberá a mensagem, então estabelece um tempo para confirmar a entrega. Então começam as transmissões até chegar ao nó que receberá a custódia. Esse então, ao receber a mensagem, armazena-a e após, transmite um sinal de confirmação ao nó inicial. Caso o sinal de confirmação chegue ao nó inicial dentro de um tempo de vida estipulado, ele então remove sua responsabilidade sobre a mensagem. Entretanto, se o sinal de confirmação não retornar no tempo estipulado, o nó inicial reinicia o processo reenviando a mensagem para o mesmo nó, ou então ele altera o nó que receberá a custódia e tenta uma nova transmissão.

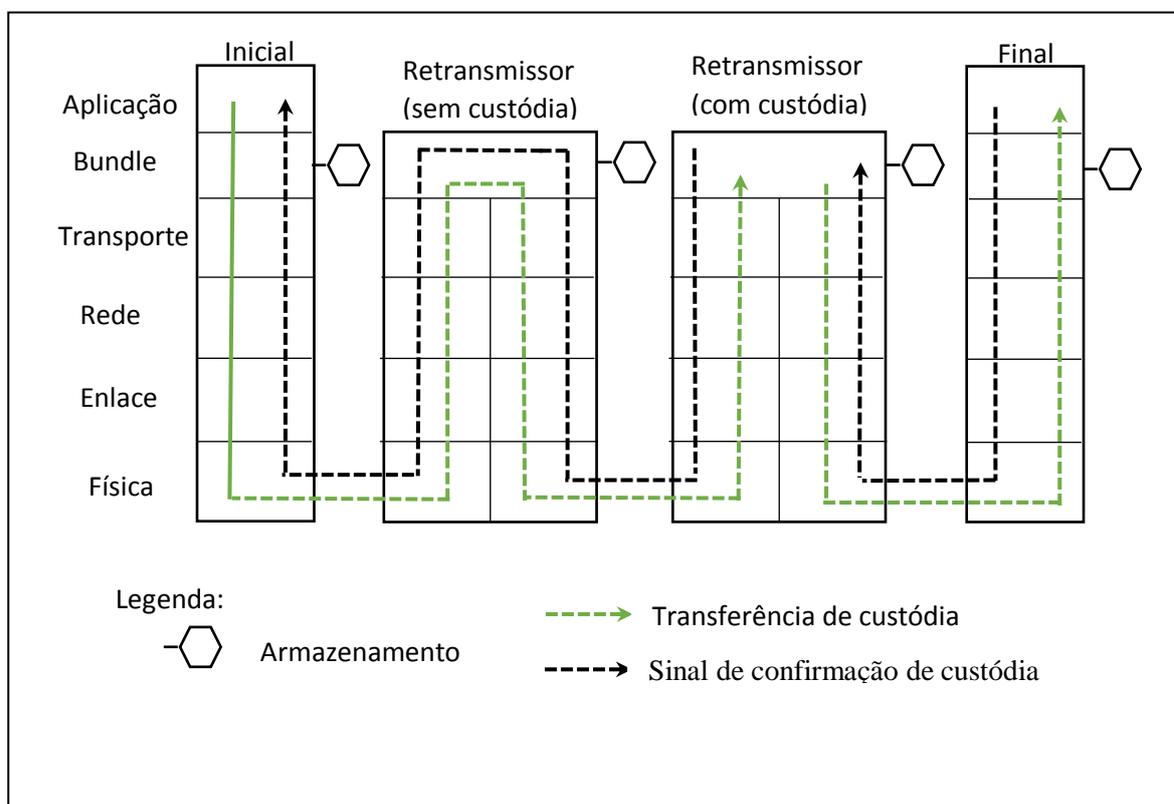


Figura 3-4 Exemplo de transferência de custódia em uma rede DTN. Fonte: autor.

Na figura 3-4 é possível observar um exemplo de transferência de custódia. No exemplo um nó inicial irá repassar custódia para um nó intermediário antes do nó final. Então ele envia a mensagem, o primeiro nó retransmissor apenas repassa para o nó que irá receber a custódia. Então o nó destinado a recebe a custódia ao receber a mensagem armazena ela em sua memória persistente, em seguida retorna um aviso de confirmação ao nó de origem, para que esse possa liberar-se da responsabilidade sobre a mensagem. Então o nó com a custódia atual da mensagem, realiza o processo de transmissão da mensagem para o nó final, como se fosse o nó de origem.

A transferência de custódia aumenta a confiabilidade de entrega de ponta-a-ponta numa rede, porém não garante a entrega.

3.7.APLICAÇÕES DE UMA DTN

- CONDOR (*Command and Control On-the-move Network Digital Over-the-horizon Relay*): projeto realizado pela MITRE Corporation para prover comunicação através veículos táticos para tropas em missões. O projeto conta com comunicação via satélite, mas para momentos de desconexões é utilizada uma DTN, desta maneira há uma maior chance do recebimento das mensagens caso a comunicação com o satélite seja interrompida de alguma maneira (CONDOR, 2017).

- DieselNet: a Universidade de Massachusetts Amherst projetou uma rede composta por 40 ônibus que cobrem uma área de aproximadamente 250 km² ao redor da cidade, onde os veículos comunicam-se de forma intermitente, utilizando rádios que funcionam no padrão 802.11b e microcomputadores. O projeto visa fornecer acesso à rede a passageiros e pedestres, porém para aumentar a conectividade é feito o uso de redes estacionárias ao longo do percurso que possuem comunicação via satélite (DIESELNET, 2017).

- Zebranet: a Universidade de Princeton desenvolveu um projeto para monitoramento de zebras em seu habitat. Os animais são primeiramente capturados e recebem então uma coleira de rastreamento. As coleiras

possuem memória *flash*, um microprocessador, um GPS e transmissores sem fio. Como as regiões de estudo não possuem serviços de comunicação, as coleiras funcionam como uma rede ponto-a-ponto, até que os dados sejam recebidos por estações bases, que podem ser computadores dos pesquisadores se movendo pela região ou estações sede localizadas em alguns pontos da região. Entretanto, como a mobilidade dos animais é variável, alguns nós podem não conseguir uma comunicação em algum momento, ocasionando assim a perda dos dados (JUANG, 2005).

3.8.MÉTRICAS DE AVALIAÇÃO E PERFORMANCE DE UMA DTN

Levando em conta as características das redes DTN, alguns pontos são muito importantes para se avaliar de forma significativa o comportamento dessas redes. Em muitos casos, métricas tradicionais são incorporadas à avaliação de redes DTN, porém é necessário entender que existem variações e visões diferentes as redes tradicionais. As métricas mais comumente utilizadas na literatura são (LINDGREN, 2009) (SPYROPOULOS, 2008):

- Mensagens Entregues:** avalia a quantia de mensagens corretamente entregue ao destino final.

- Overhead:** é definido como o número médio de cópias de cada mensagem gerada na rede. Como redes DTN normalmente possuem recursos limitados, é necessário otimizar para alcançar um baixo nível de *overhead*.

- Atraso:** medida do tempo desde a criação da mensagem até a entrega ao destinatário. Esse valor refere-se apenas as mensagens que são entregues.

Junto a essas métricas, existem outras que ajudam a avaliar melhor o desempenho da rede em relação aos recursos ou comportamentos inesperados dos nós. Sendo possível avaliar (NUNES, 2009):

- Uso do Buffer:** mede o número médio de mensagens armazenadas por um nodo, pode-se também medir a média para todos os nodos.

- Tempo médio no *Buffer*: avalia o tempo médio que uma mensagem ficou sob custódia de novo antes de ser repassada.

- Mensagens Descartadas: considera o número de mensagens descartadas, normalmente por *buffer* cheio. Usada em conjunto com as duas métricas anteriores para detectar a performance da alocação do *buffer*.

- Entregas Redundantes: número médio de cópias repetidas entregues ao destino final, após a primeira já ter sido recebida. Normalmente utilizada para avaliar o uso desnecessários de recursos numa rede.

3.9. SEGURANÇA

A segurança de uma rede DTN é feita por um subprotocolo, o *Bundle Security Protocol* (BSP), ou protocolo de segurança de agregação, que através de troca de mensagens com os nós da rede, pode averiguar os fatores de segurança. Este protocolo assegura a integridade dos dados, autenticação e confidencialidade. Apenas nós com *Security-Aware* (SA), ou reconhecimento de segurança, possuem a capacidade de originar e processar esses blocos de segurança BSP.

Existem 4 tipos de blocos de segurança:

- Bundle Authentication Block* (BAB): responsável por garantir a autenticidade e integridade da mensagem ao longo do caminho, verificado sempre por nós que possuem SA.

- Payload Integrity Block* (PIB): verifica a integridade e veracidade de uma mensagem utilizando uma verificação do signatário.

- Payload Confidentiality Block* (PCB): responsável pelas chaves criptografia de uma mensagem.

- Extension Security Block* (ESB): estende os serviços de segurança para mensagens de controle.

4.BUNDLE LAYER PROTOCOL

Este capítulo apresenta o protocolo *Bundle Layer protocol*. São também apresentados detalhes sobre o funcionamento e a estrutura do protocolo desde seu frame até algoritmo em alto nível.

4.1.O PROTOCOLO

O protocolo de agregação (*Bundle Layer*) está descrito na RFC 5050 [Scott e Burleigh, 2006].

Cada *Bundle*, ou conjunto de dados, é composto por no mínimo dois blocos. Ao invés de cabeçalho é utilizada a nomenclatura bloco, pois um bloco pode estar tanto no início como no fim de um conjunto de dados. O primeiro bloco, também chamado bloco primário, é obrigatório em todos os *bundles*, já que contém as informações básicas para realizar a transferência até o destino. O bloco primário não possui nenhuma carga útil (*payload*) de dados, apenas os blocos seguintes podem contar dados.

Para uma maior adaptação do protocolo, vários campos possuem tamanhos variáveis e utilizam a codificação denominada Valores Numéricos Auto-Delimitantes (*Self-Delimiting Numeric Values – SDNVs*). O SDNV codifica números binários sem sinais em octetos, grupos de sete bits de valores e com um bit mais significativo de cada octeto igual a 1, exceto o último octeto cujo bit mais significativo é igual a 0 para indicar o fim do SDNV. Desse modo o tamanho do campo é delimitado apenas pela própria codificação que indica continuação até encontrar um bit mais significativo igual a 0.

Na figura 4-1 é apresentado o bloco primário. Como é utilizado a codificação SDNV, o número de bytes não é apresentado devido a arbitrariedade dos valores. O bloco primário é composto por diversos campos, em todos os campos exceto no campo versão e matriz de octetos do dicionário são codificados em SDNV. Cada um destes campos encontra-se detalhado a seguir:

- Versão: Campo de 1 byte indicando a versão do protocolo utilizada no bloco. A versão atual descrita é a 0x06.

- Bits de Controle: Contém as informações referentes ao controle do *bundle*.

- Tamanho do Bloco: Contém o tamanho total do restante dos campos do bloco.

- Deslocamento do esquema do reportar-para: Contém o deslocamento no dicionário para a ID do nó para o qual os *status reports* sobre o *bundle* devem ser transmitidos.

- Deslocamento da SSP do reportar-para: Contém o deslocamento dentro do dicionário da parte específica do esquema do nó para qual deve ser reportado.

- Deslocamento do esquema de custódia: Contém deslocamento dentro do dicionário para encontrar o ID do nó atual que aceitou o bloco primário para a transmissão por custódia.

- Deslocamento da SSP de custódia: Contém o deslocamento dentro do dicionário para encontrar SSP do nó atual que aceitou o bloco primário para a transmissão por custódia.

- Tempo da estampa de tempo de criação (*Creation Timestamp Time*): Guarda o primeiro instante da estampa de criação do *bundle*.

- Número de sequência da estampa de tempo de criação: Consiste em um número inteiro positivo monotonicamente crescente que pode ser zerada a cada segundo, armazenado na escala UTC. Juntamente com o Tempo da estampa de tempo de criação, serve para identificar o *bundle*.

- Tempo de vida: Armazena, em segundos, o tempo a partir da criação do *bundle*.

- Comprimento do dicionário: Contém o tamanho da matriz de octetos do dicionário.

•Matriz de octetos do dicionário: Formada pela concatenação dos nomes e esquemas de todos os pontos de extremidades utilizados no Bloco Primário.

Versão	Bits de controle
Tamanho do Bloco	
Deslocamento do esquema do destino	Deslocamento da SSP do destino
Deslocamento do esquema da fonte	Deslocamento da SSP da fonte
Deslocamento do esquema do reportar-para	Deslocamento da SSP do reportar-para
Deslocamento do esquema de custódia	Deslocamento da SSP da custódia
Tempo da estampa de tempo de criação	
Número de sequência da estampa de tempo de criação	
Tempo de vida	
Comprimento do dicionário	
Matriz de octetos do dicionário (Variável)	
Deslocamento do fragmento*	
Comprimento total da unidade de dados de aplicação*	

Blocos com codificação SDNV.

Blocos sem codificação SDNV.

* Blocos existentes apenas se especificado nos bits de controle.

Figura 4-1 Formatação do Bloco Primário. Fonte: autor.

Os campos seguintes são utilizados apenas no caso de fragmentos.

•Deslocamento do fragmento: Indica o deslocamento do fragmento em relação ao início da unidade.

•Comprimento total da unidade de dados de aplicação: Corresponde ao tamanho total da unidade de dados utilizados pela aplicação.

Já na figura 4-2, é apresentado o bloco de carga útil do *bundle*. Esse bloco contém 4 campos, um identificador do tipo de bloco, os bits de controle, o comprimento do bloco e a carga útil ou dados, esse sendo o único com tamanho variável nesse bloco.



Figura 4-2 Bloco de carga útil. Fonte: Autor

4.1.1. Bits de controle

O bloco de bits de controle está presente em todos os blocos do *bundle*. O valor codificado nesse SDNV é um conjunto de bits usados para invocar características selecionadas do bloco de processamento e controle. Na figura 4-3 os bits são apresentados em ordem do menos significativo (0) ao mais significativo (6).

Bit	Função
0	Bloco deve ser replicado em cada fragmento
1	Transmitir relatório de estado se o bloco não puder ser processado
2	Deletar o <i>bundle</i> se o bloco não puder ser processado
3	Último bloco
4	Descartar o bloco se não puder ser processado
5	Bloco foi encaminhado para estar processado
6	Bloco contém uma referência do ponto final

Figura 4-3 Bits de controle do protocolo. Fonte: Autor.

4.2.FRAGMENTAÇÃO

Como um bloco de mensagem pode ser transferido entre regiões heterogêneas, cada uma pode possuir uma limitação máximo de tamanho de bloco, seja pelo fluxo de dados do canal, capacidade do *buffer*, entre outros. Dessa maneira, é necessário que o protocolo fragmente o bloco para que seja possível enviá-lo, ou caso seja impraticável, rever possíveis nós disponíveis para enviar por uma rota alternativa que comporte o tamanho do bloco. Para isso, existem duas maneiras de fragmentação definidas:

- Fragmentação Proativa: nesse caso, o nó analisa os anteriormente a capacidade do nó destino e a capacidade do canal entre os dois. Então com essa informação ele fragmenta o bloco no maior tamanho possível, para máximo aproveitamento do canal antes de estabelecer a comunicação.

- Fragmentação Reativa: Essa fragmentação ocorre apenas quando o bloco é dividido em pedaços menores, do que a divisão feita pela camada DTN, pelos protocolos inferiores que então avisa que o bloco foi enviado parcialmente.

4.3. A CAMADA DE CONVERSÃO

O *Convergence Layer* (Camada de Conversão) serve para permitir que o *Bundle Protocol* opere sobre protocolos de transporte diferentes. Algumas vezes, existe a necessidade do uso de algum protocolo de transporte específico o que apresenta um dos desafios de uma rede DTN heterogênea. A principal tarefa dessa camada, é permitir o melhor uso de uma comunicação e uma abstração dos protocolos de utilizados na rede.

Pertencente à camada DTN, a camada de conversão funciona sobre protocolos de transmissão, como o TCP e o UDP, e abaixo da camada de serviços do *Bundle Protocol*. Na figura 4-4 é apresentada uma visualização da camada de conversão, e suas possibilidades.

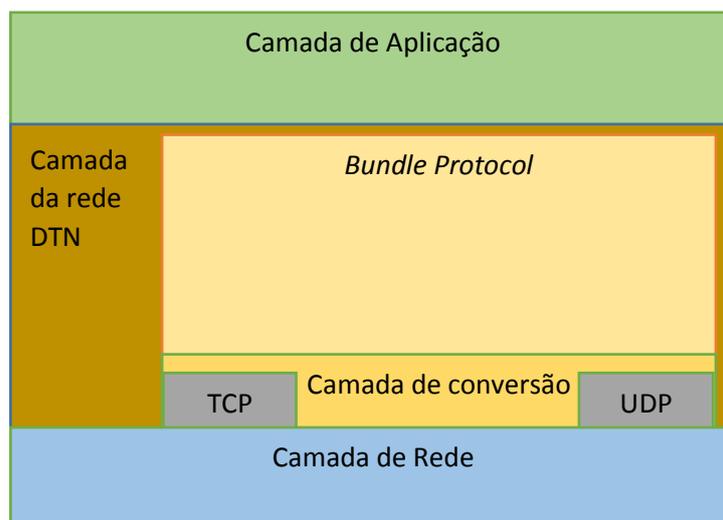


Figura 4-4 Visualização da camada de conversão. Fonte: Autor.

5.0 SIMULADOR ONE

O simulador *Opportunistic Network Environment* (ONE) foi projetado especificamente para o estudo de redes DTN. Implementado em Java para maior portabilidade, foi desenvolvido pela universidade de Helsinki, e permite ao usuário descrever múltiplos cenários, compostos por nodos, modelos de movimentação, protocolos de roteamento, protocolos de aplicação, interfaces, entre outros (KERÄNEN,2008).

O ONE oferece uma estrutura extensível de simulação, suportando a geração de eventos, troca de mensagens, roteamentos externos, novos protocolos, modelos de movimentação gerados por ferramentas externas, uma noção básica do consumo de energia, além de visualização da simulação e ferramentas de pós-processamento para os relatórios das simulações.

Como é possível ver na figura 5-1, sua implementação é modular, o que permite a implementação de novos modelos, tanto de mobilidade, roteamento, aplicações, protocolos e eventos. Uma das facilidades desse modelo, é que cada módulo já foi válido individualmente, garantindo assim uma maior

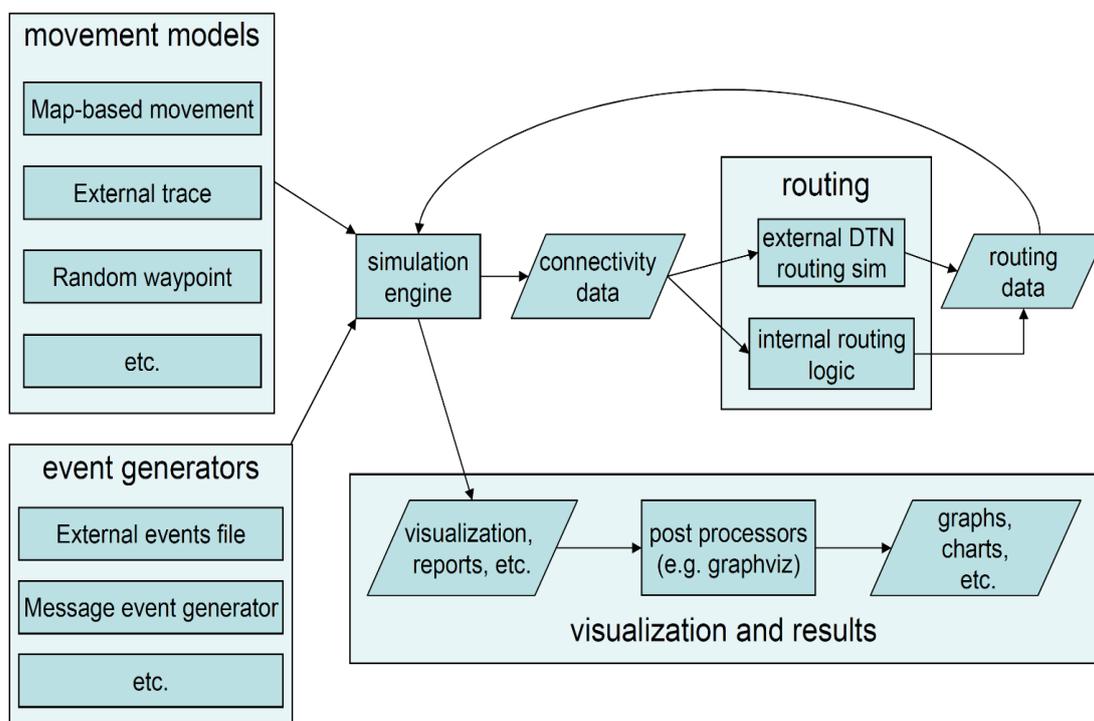


Figura 5-1 Modelo de implementação do simulador ONE. Fonte: KERÄNEN, A.2008.

confiança nos resultados obtidos das simulações.

O simulador apresenta uma interface de usuário gráfica (*Graphical User Interface, GUI*), apresentada na figura 5-2, que permite a visualização dos eventos em tempo de simulação, apresentando localização, mensagens, comunicações ativas e área de alcance de cada nó. Na imagem, os nós são os pontos p , os círculos verdes, apresentam o alcance de comunicação de cada nó.

Também é possível alterar a velocidade da visualização, bem como obter informações mais detalhadas de cada nó, como histórico de mensagens ou nós contatados, ou o caminho percorrido por uma mensagem, bem como a localização de seus outros fragmentos.

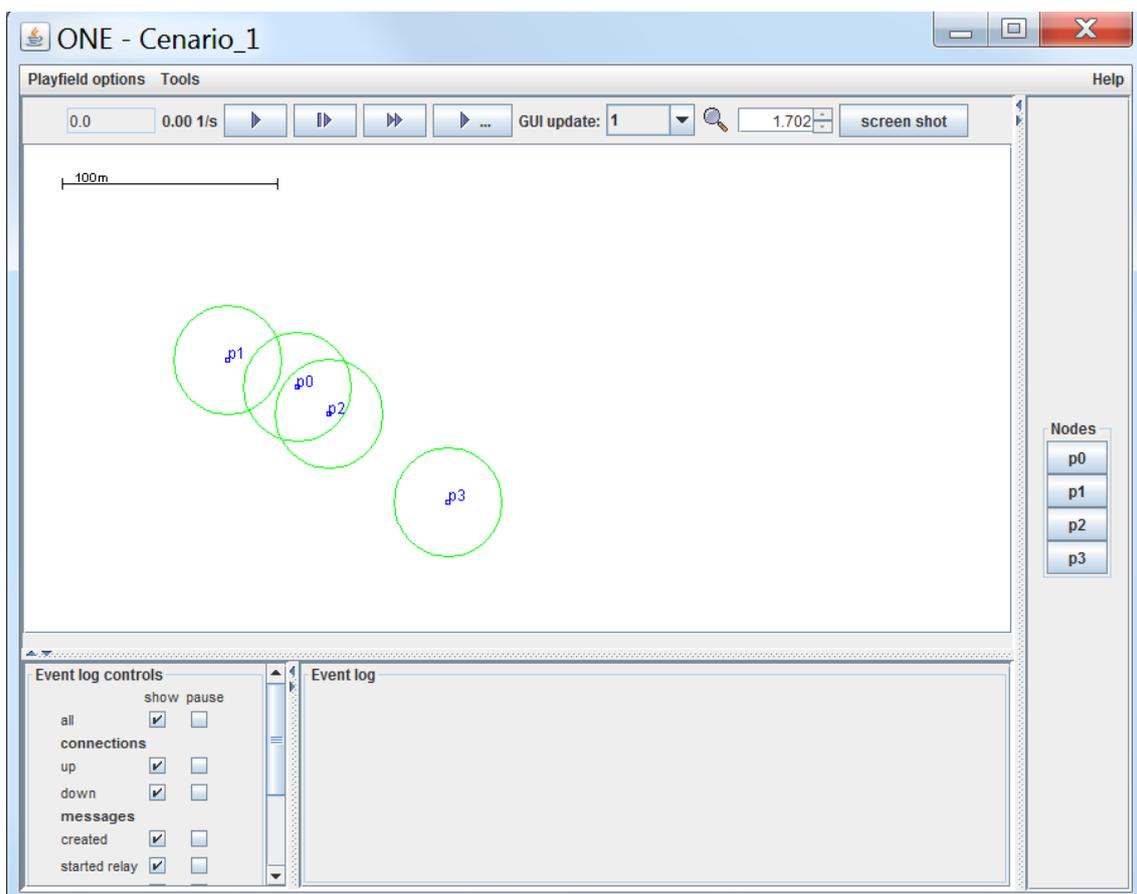


Figura 5-2 Interface gráfica do simulador ONE. Fonte: Autor.

5.1.CARACTERÍSTICAS DOS NÓS

Os agentes do simulador são chamados nós. Cada nó é representado por um modelo que possui informações básicas sobre suas características, como interfaces de comunicação, armazenamento, modelo de energia, movimentação e roteamento das mensagens. Na listagem 1 é apresentado o trecho de código referente a configuração de um grupo de nós, através da parametrização dos valores referentes.

Listagem 1 Parâmetros de caracterização de um grupo de nós no ONE. Fonte: Autor.

```
#Movimentação
Group.movementModel = RandomWaypoint

#Tipos de roteamento
Group.routeType = 1

#Protocolos de roteamentos
Group.router = EpidemicRouter

#Tamanho do buffer
Group.bufferSize = 5M

#Tempo de espera entre tentativas de contato
Group.waitTime = 0, 120

#Interfaces disponíveis
Group.nrofInterfaces = 1

#Definição das interfaces
Group.interface1 = bluetoothInterface

#velocidade de movimento
Group.speed = 0.5, 1.5

#tempo de vida dos pacotes
Group.msgTtl = 3
```

5.2.MODELOS DE MOVIMENTO

No simulador, a capacidade de movimento dos nós é implementada através de modelos de mobilidade. Esses modelos de mobilidade definem algoritmos e regras para gerar caminhos através das informações do cenário. No ONE, três modelos de mobilidades básicos já estão implementados:

- *Random Movement* (movimento aleatório): Como o próprio nome já indica, não possui regras ou destinos fixados, os nós caminham livremente pelo cenário.

- *Map Constrained Random Movement* (movimento aleatório restrito pelo mapa): uma otimização do *Random Movement*, onde os nós possuem conhecimento de áreas como construções, rios, lagos, etc.

- *Human behavior Based Movement* (Movimento baseado no comportamento humano): baseado em escolhas comuns, como fluxos e agrupamentos de pessoas, também leva em conta informações do cenário como o *Map Constrained Random Movement*.

Além desses modelos de movimento, ainda é possível criar modelos novos baseados em dados externos de movimento, como rotas predefinidas, pontos de encontro, etc.

5.3.ROTEAMENTO

O simulador já oferece um conjunto de 6 roteamentos já implementados, porém novos modelos de roteamento são facilmente suportados, sendo necessário apenas implementar uma classe definindo as regras e o algoritmo do novo protocolo. Alguns dos protocolos de roteamento disponibilizados, já implementados e validados pelo simulador, são: *Direct Delivery (DD)*, *First Contact (FC)*, *Spray-and-Wait*, *PRoPHET*, *MaxProp*, *Epidemic*.

6. NOVO PROTOCOLO

Este capítulo apresenta as alterações propostas no protocolo *Bundle Layer*. Ao final é apresentada a metodologia, a validação e o ambiente utilizado no desenvolvimento e na validação do protocolo.

6.1.A PROPOSTA

Cidades inteligentes, são zonas densamente povoadas com dispositivos capazes de estabelecer comunicação com outros dispositivos em uma certa área de alcance. Muitos desses dispositivos estão em constante movimento, o que pode, em conjunto com a elevada densidade da região, aumentar as chances de existirem possíveis caminhos de roteamento entre dois dispositivos distantes (ZHANG, 2007).

Considerando-se então as necessidades apresentadas para a comunicação em cidades inteligentes, de comunicação entre todos, a qualquer momento e em qualquer lugar com confiabilidade e segurança, foi então projetado um modelo, que suprisse as regras da arquitetura de uma DTN, com uma camada comum a todos os nós, permitindo assim uma comunicação entre todos, para manter os serviços básicos de retransmissão que oferecidos pelo *Bundle Layer Protocol*, além da inserção de duas novas camadas, uma camada de segurança e uma camada de conversão.

A ideia principal é o balanceamento de carga, acumulando tarefas mais pesadas para nós que possuam recursos computacionais abundantes, enquanto tarefas menores são delegadas aos nós com menos recursos. Possibilitando dessa maneira uma divisão mais consciente em termos de eficiência e recursos. A seguir são apresentadas as principais modificações feitas no protocolo.

6.1.1.A pilha do protocolo

Através do balanceamento de carga citado anteriormente, surgiu uma proposta para a nova implementação da pilha do protocolo, para satisfazer as condições necessárias há uma rede DTN e ao mesmo tempo para obter-se uma melhor performance na rede. Para isso, foram delegadas funções diferentes para os nós, possibilitando assim a modularização das responsabilidades de cada tipo de nó, sendo os nós separados em 3 categorias, nós Agentes, nós Estações e nós Completos.

Na figura 6-1 é possível perceber que se manteve uma camada comum entre todos os nós de uma rede DTN. Na camada comum, estão implementadas as funções principais citadas anteriormente, como a transferência de custódia e *Store-and-Forward*. Então todos os nós são capazes de transmitir e armazenar mensagens, são os chamados nós Agentes.

Já os nós considerados Estações, possuem implementadas as funções mais avançadas, como a conversão entre interfaces, fazendo a função de *Gateway* e são deliberados como preferencias para escolha de nós com

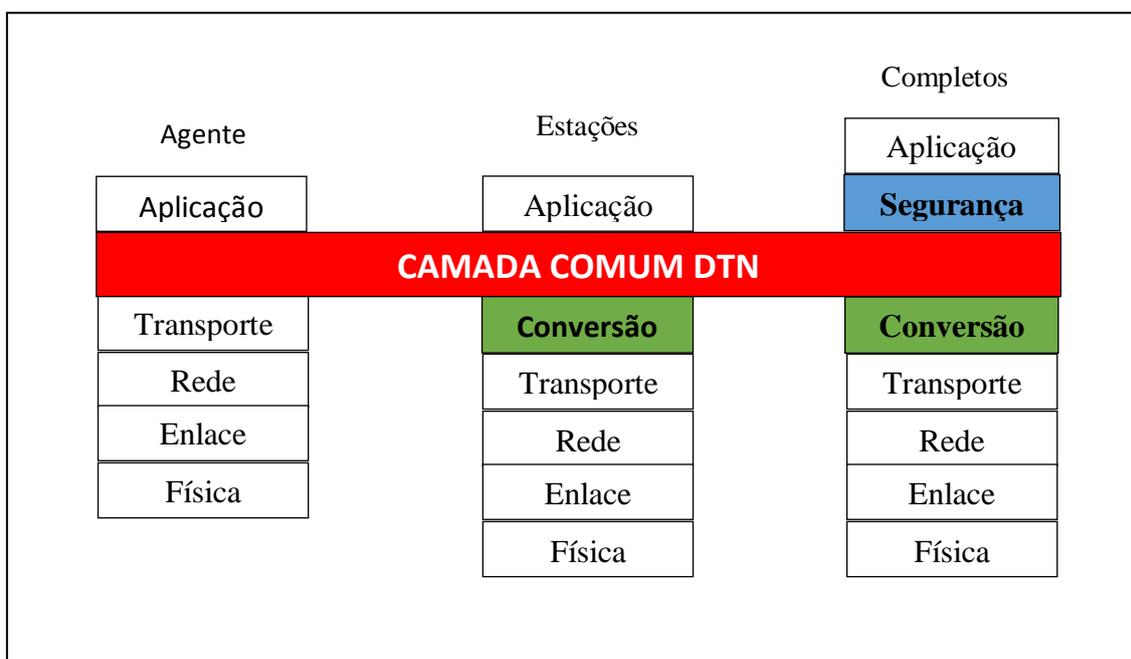


Figura 6-1 Novo modelo de pilha proposto para cidades inteligentes. Fonte: Autor.

custódia.

Os nós Completos, pela definição do nome, possuem todas os serviços implementados, podendo atuar como nós Agentes ou nós Estações, e ainda possuindo acesso aos subprotocolos de segurança e processamento final dos dados coletados.

Esse novo modelo, permite então deliberar melhor uma função a capacidade de cada nó. Nós com menor armazenamento, menos tempo estimado de vida e poder computacional, como celulares ou sensores, são deliberados como agentes, poupando assim de executarem funções que consomem muitos recursos. Nós intermediários ou estrategicamente colocados, podem receber a funções de nós estações, podendo ser microcomputadores ou nós de uma rede veicular. A função de nó completo seria reservada a uma estação principal de coleta de dados, ou uma central, que dispõe de muitos recursos e não possui riscos altos de falta de energia, memória ou indisponibilidade.

Na tabela 6-1 são apresentadas as funções delegadas a cada nó.

Tabela 6-1 Comparação das funções de cada tipo de nó. Fonte: Autor.

	Agentes	Estações	Completos
Store-and-Forward	X	X	X
Transferência de custódia	X	X	X
Fragmentação	X	X	X
Sinais de confirmação	X	X	X
Conversão de Protocolos		X	X
Processamento Final			X
Protocolo de segurança			X

6.1.2.Fragmentação

Entre os nós de uma rede DTN normal, sempre era possível ambos os tipos de fragmentação, tanto proativa quanto reativa. No novo modelo, os nós agentes não mais usaram a fragmentação proativa, deixando para que os protocolos inferiores gerenciem esse quesito. Assim não são gastos recursos para obter conhecimento prévio dos nós vizinhos, suas capacidades, tempo de duração da comunicação ou do fluxo de dados do canal.

6.2.METODOLOGIA

A proposta da arquitetura de protocolo desenvolvido por esse trabalho utilizou a metodologia apresentada no fluxograma exibido na figura 6-2. A etapa de especificação consistiu em avaliar os requisitos apresentados propostos por (WAHER,2015) (FAN e CHEN, 2010) e (BARNAGHI, 2014), sobre cidades inteligentes:

- Comunicação entre qualquer coisa;
- Comunicação em qualquer lugar;
- Comunicação a qualquer tempo;
- Confiabilidade;
- Segurança;

O software ONE *Simulator* foi utilizado tanto nas etapas de simulação funcional e simulação de performance. O software é disponibilizado pela universidade de Helsinki, para uso de pesquisa. Na etapa da simulação funcional foi feita a validação dos resultados obtidos, melhor apresentado na seção seguinte.

Após o término da etapa de validação, foi então dado início a fase de testes de performance. Utilizando o mesmo software foram realizados diversos testes, em cenários com capacidades diferentes para a análise do

comportamento do protocolo em situações diferentes. Esses testes e cenários serão detalhados na seção 6.2.2.

Uma vez realizada a etapa de simulação de performance, os dados coletados foram então analisados para avaliar o desempenho do protocolo.

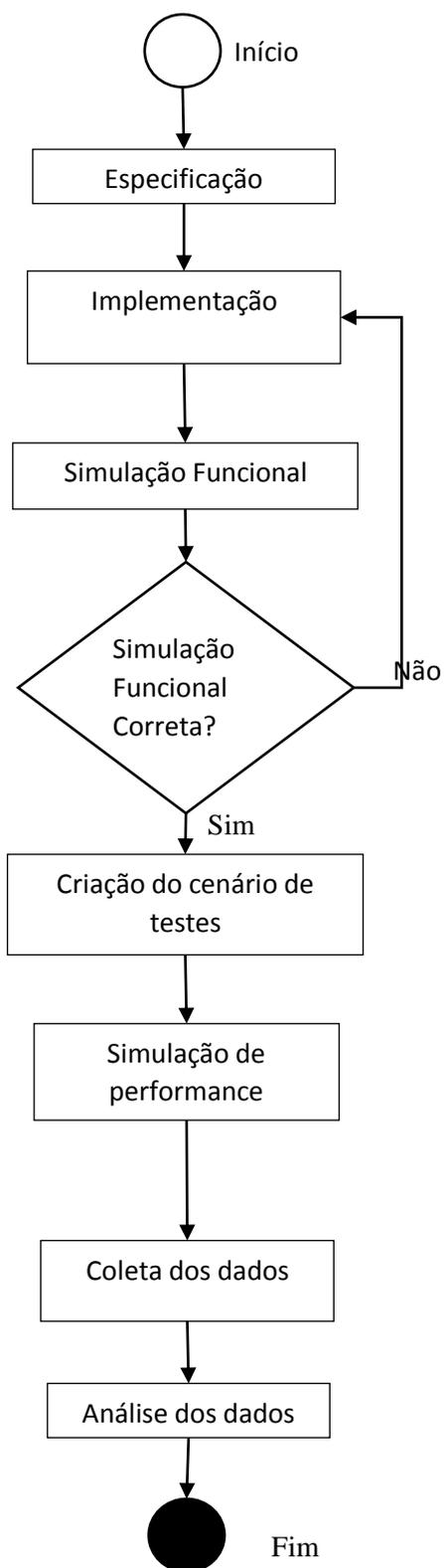


Figura 6-2 Fluxograma da metodologia utilizada por este trabalho para a implementação de um protocolo DTN. Fonte: Autor.

6.2.1.VALIDAÇÃO

Simulações de padrões e protocolos de comunicação possuem a necessidade de validação a fim de garantir que o comportamento do modelo está correto e de acordo com o esperado. A validação de um modelo consiste em certificar-se que o modelo produza resultados semelhantes ao que seriam alcançados em sistemas reais. Porém em sistemas reais há uma grande quantidade de variáveis envolvidas, então é necessário um certo nível de abstração (JAIN, 1991).

A validação da proposta de redução de protocolo foi feita baseada no sistema de validação do simulador ns-2. Devido ao fato de seu amplo uso para verificação, simulação e validação de diversos tipos de protocolos. O conjunto de testes consiste em executar diversos conjuntos de entradas e comparar os arquivos de saída gerados com os arquivos contidos na distribuição (ANTONIO, 2004).

Na listagem 2 é apresentado o pseudocódigo de como foi feita a validação no simulador, sendo avaliado então pelas tabelas geradas pelo próprio simulador para controle e criação das mensagens. No fim são comparadas as tabelas geradas entre o protocolo original do simulador e o protocolo proposto, para uma maior confiabilidade nos testes.

Listagem 2 Pseudocódigo da validação do protocolo. Fonte: Autor.

```

Início
Para cada nó a na lista de nós faça:
{
    Para cada mensagem m enviada pelo nó a faça
    {
        Se nó destino b é diferente do nó b1 que recebeu a mensagem m então
            Retorne erro;
        Se m é diferente de m1 recebida por b então
            Retorne erro;
        Se o Checksum de m1 for diferente de m então
            Retorne erro;
        Senão
            Retorne sucesso;
    }
}

```

No caso, as adaptações dos testes foram feitas para comparar os arquivos contendo as informações sobre as mensagens, comunicações e eventos gerados pelo simulador ONE, descrito a seguir. Comparando-se então os resultados do protocolo oferecido pelo simulador, que já havia sido validado, e os resultados obtidos com a implementação do protocolo proposto.

Pode-se observar na tabela 6-2 algumas linhas dos resultados sobre as mensagens do protocolo implementado pelo simulador, enquanto na tabela 5-3 são apresentados os resultados equivalentes obtidos com o protocolo proposto. A tabela é composta pelas seguintes informações em ordem:

- Tempo de criação da mensagem: Informa o tempo, em segundos, no instante em que a mensagem é criada na simulação.

- Validade da mensagem: informa se a mensagem foi entregue corretamente (C), ou se possui algum erro (F), levando em conta apenas integridade e veracidade dos valores de verificação.

- Identificador da mensagem: identifica o número de mensagem para avaliação individual caso encontrado algum erro. Composto por M seguido da posição em que foi criada.

- Identificador do nó de origem: apresenta o número correspondente ao nó que originou essa mensagem.

- Identificador do nó destino: apresenta o número do nó que deve receber a mensagem.

- Tamanho da mensagem: contém o tamanho da mensagem, em bytes, para análise de *buffer* ou quebra de mensagens.

A tabela 6-2 apresenta os resultados do teste de validação aplicado ao protocolo DTN implementado pelo simulador ONE. Os valores gerados nesse teste, serão comparados aos valores apresentados na tabela 6-3, coletados pela validação do protocolo proposto. Esses valores servem para identificar se as mensagens criadas, serão entregues corretamente para os nós de destino, verificando integridade, perda de pedaços da mensagem ou entrega a nós incorretos.

Tabela 6-2 Resultados obtidos do protocolo implementado pelo simulador. Fonte: Autor.

Tempo de criação (em segundos)	Validade	Identificador da mensagem	Identificador do nó de origem	Identificador do nó destino	Tamanho da mensagem (em bytes)
285.7	C	M1	163	509	550704
558.6	C	M2	752	241	100161
1081.0	C	M3	214	255	542599
1221.9	C	M4	405	423	389867
1266.9	C	M5	388	927	671217
1736.4	C	M6	510	124	805696
2203.0	C	M7	522	877	33705
4478.6	C	M8	79	734	94375
5192.6	C	M9	201	288	65191

Comparando os resultados apresentados nas tabelas, é possível perceber que o protocolo se comporta de maneira esperada, reproduzindo os resultados obtidos no protocolo validado pelo simulador. Foram realizados diversos testes, com várias configurações, e os resultados obtidos pelo protocolo proposto sempre eram compatíveis àqueles apresentados pelo protocolo do simulador. Algumas das configurações testadas foram:

- 10 nós e 10 mensagens;
- 10 nós e 25 mensagens;
- 50 nós e 50 mensagens;
- 50 nós e 125 mensagens;
- 50 nós e 200 mensagens;
- 100 nós e 100 mensagens;
- 100 nós e 500 mensagens;

Tabela 6-3 Resultados obtidos do protocolo proposto. Fonte: Autor.

Tempo de criação (em segundos)	Validade	Identificador da mensagem	Identificador do nó de origem	Identificador do nó destino	Tamanho da mensagem (em bytes)
213.7	C	M1	163	509	550704
424.1	C	M2	752	241	100161
1251.0	C	M3	214	255	542599
1340.3	C	M4	405	423	389867
1786.9	C	M5	388	927	671217
2529.8	C	M6	510	124	805696
4158.3	C	M7	522	877	33705
5453.2	C	M8	79	734	94375
5883.0	C	M9	201	288	65191

Com o protocolo validado, passou-se então a fase de testes, para avaliação da performance do protocolo.

6.2.1.1.O MODELO ENERGÉTICO

O modelo energético utilizado para as simulações é baseado no próprio modelo oferecido pelo simulador, porém constatou-se a necessidade de algumas modificações, para obter uma maior precisão nos resultados. Enquanto o modelo atual baseava-se apenas na energia usada para transmissão e resposta, o novo modelo proposto avalia também energia consumida durante períodos ociosos e ao iniciar varreduras.

O novo modelo torna mais preciso a avaliação da eficiência energética dos dispositivos, já que leva em conta a energia consumida em tempos de espera, um fator que não era levado em conta no modelo inicial, e que pode ser impactante avaliando o tempo que uma mensagem permanece ociosa no buffer. Outra vantagem do novo modelo é a medição do consumo de energia durante varreduras, visto que adicionando o consumo de energia da varredura de nós vizinhos é possível adicionar ao resultado o quanto de energia é gasta enquanto um nó não encontra outro nó para a transmissão da mensagem, o que impacta na sua vida útil.

Na figura 6-3 é apresentado como ficou o novo modelo energético utilizado nos dispositivos móveis e sensores.

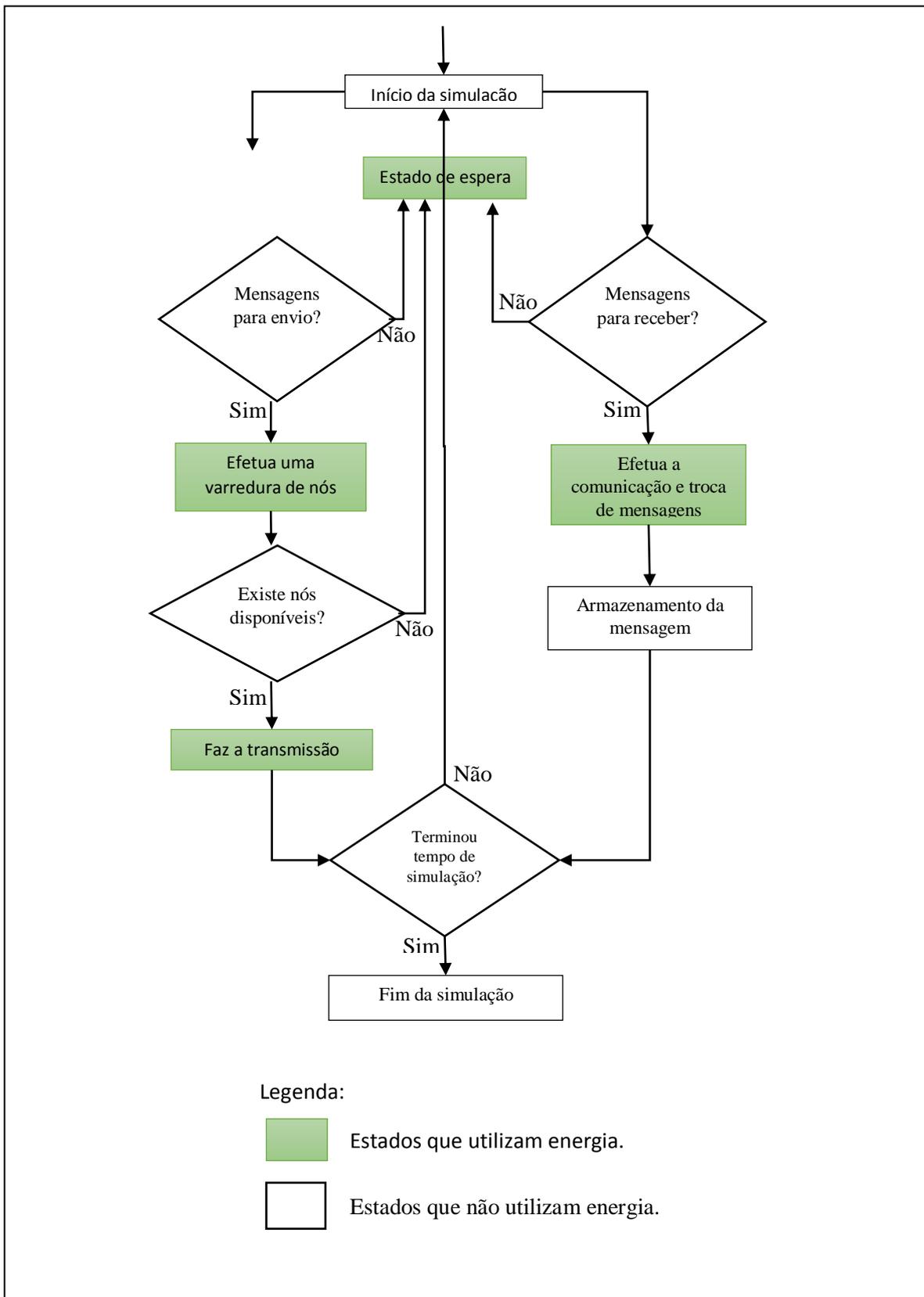


Figura 6-3 Novo modelo energético utilizado. Fonte: Autor

O cenário possui 3 tipos diferentes de nós. Existem os nós do grupo 1, equivalentes aos pedestres, grupo 2, veículos de mobilidade urbana e grupo 3, centrais espalhadas pela cidade. Na tabela 6-4 é comparada as características de cada grupo.

O modelo de movimento dos nós é baseado no sistema de transporte público de santa maria, na Figura 6-4, com as vias destacadas em vermelho, com os correspondentes códigos das linhas (SIM,2018). As centrais não possuem movimento, são estações fixas. Os nós correspondentes aos pedestres possuem o modelo de mobilidade aleatória, podendo mover-se livremente pelo mapa, com a possibilidade de entrar em prédios ou caminhar pelas vias alternativas bem como as vias do transporte público.

Tabela 6-4 Comparação entre as características dos nós de cada grupo. Fonte: Autor.

	Grupo 1	Grupo 2	Grupo 3
Tamanho de Buffer	2Mb	10Mb	30Mb
Interfaces	Bluetooth (802.11g)	Bluetooth(802.11g) e Wireless Interface(802.11n)	Bluetooth(802.11g) e Wireless Interface(802.11n)
Velocidade de movimento (em km/h)	Varia entre 0 e 6	Varia entre 0 e 40	0
Tempo de espera para reenvio (em segundos)	0 a 120	0 a 300	0 a 500

A velocidade dos nós é baseada na velocidade média de movimento de uma pessoa adulta no caso do grupo 1. No grupo 2 a velocidade é definida pela máxima permitida na zona urbana.

As interfaces foram selecionadas devido a sua ampla utilização, sendo predominante em aparelhos de celulares e roteadores de uso geral. O grupo 1 foi simulado através de smartphones, portanto a escolha da interface Bluetooth. O grupo 2 foi equipado com um Raspberry Pi 3 (RASPBERRY, 2018). O grupo 3 foi equipado com roteadores wireless TL-WR940N (TP-LINK, 2018).

Todas as configurações citadas foram testadas utilizando os 2 protocolos, o *Bundle Layer Protocol*, implementado pelo simulador, e o protocolo proposto pelo trabalho. E cada um dos dois protocolos foi simulado utilizando 3 protocolos de roteamento diferentes, para que fosse possível avaliar seu desempenho em casos diferentes:

- *Epidemic*;
- *PRoPHET*;
- *Spray-and-Wait*;

Foram feitas simulações para cada uma das 6 combinações possíveis, cada uma incrementando o número de nós de cada grupo, para avaliar o desempenho da rede conforme a densidade. A quantidade de nós de cada grupo é dada pela seguinte variação:

- Número de nós Agentes: 250,500,1000;
- Número de nós Estações: 50,75,100;
- Número de nós Completos: 5,10,20;

Dessa maneira a primeira combinação para as simulações é um grupo de 250 Agentes, 50 Estações e 5 Completos. O segundo grupo é composto por 500 Agentes, 75 Estações e 10 Completos. O terceiro grupo são 1000 nós Agentes, 100 Estações e 20 Completos.

7.RESULTADOS

Este capítulo apresenta os resultados obtidos nas simulações, de energia e performance, nas seções 6.1 e 6.2 respectivamente. Esses resultados são baseados na metodologia de métricas de (LINDGREN, 2009) e (SPYROPOULOS, 2005). São comparados os dois protocolos, *Bundle Layer Protocol* e protocolo proposto.

Os resultados apresentados foram obtidos através dos testes realizados descritos anteriormente. Os valores são decorrentes de uma média de 5 simulações, onde o pior e o melhor resultado foram retirados do cálculo, sendo então calculada a média simples dos 3 resultados restantes.

7.1. RESULTADOS DE ENERGIA

Para obter os resultados referentes a energia, cada nó é analisado a cada iteração do simulador. Ao final é feita a diferença do valor inicial do final, para obter-se a quantia consumida durante o período de simulação. Os resultados de energia são focados nos nós do grupo 1 da simulação, os smartphones e dispositivos móveis, visto que são os únicos que apresentam um suprimento limitado de energia. A unidade de energia que o simulador trabalha é em Joule (J).

É possível observar o comportamento nas figuras 7-1,7-2 e 7-3, que apresentam o consumo dos protocolos *Epidemic*, *PRoPHET* e *Spray-and-Wait*, respectivamente. Em todos os casos, o consumo de energia foi reduzido, porém em alguns testes não de maneira significativa.

No melhor dos casos, utilizando o roteamento *Spray-and-Wait*, com 250 nós, o consumo reduziu em torno de 10%. Porém avaliando isso num dispositivo móvel padrão, com uma bateria de 3000mAh, que funciona com uma voltagem de 3.7V, essa economia seria de menos de 1% do total da bateria.

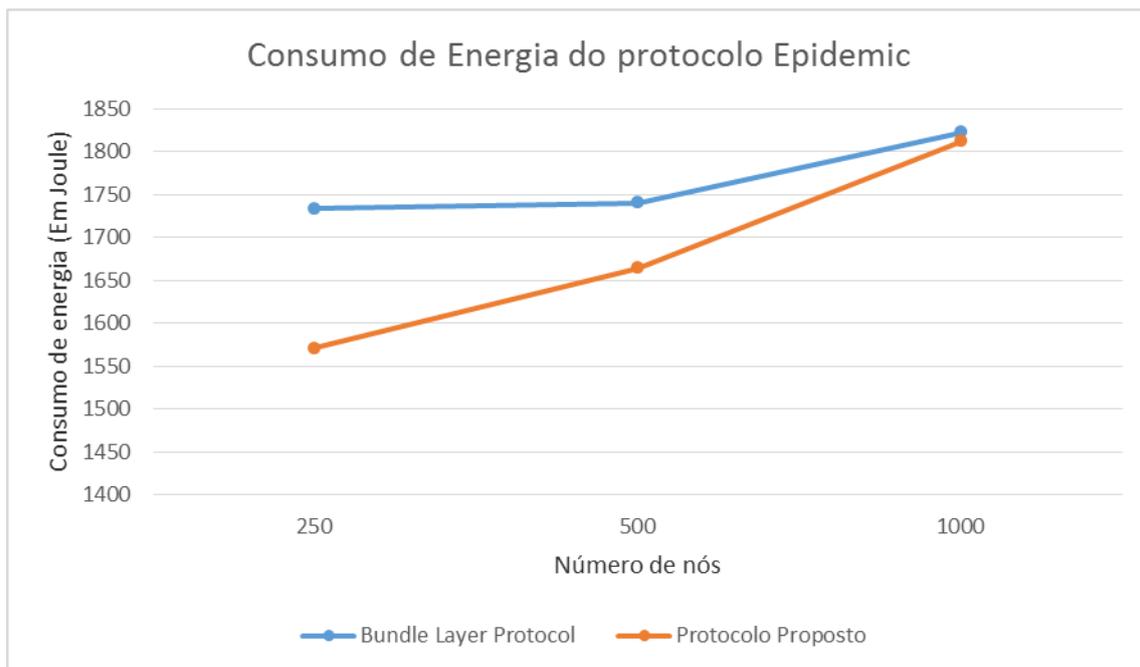


Figura 7-1 Consumo de energia do protocolo Epidemic. Fonte: Autor.

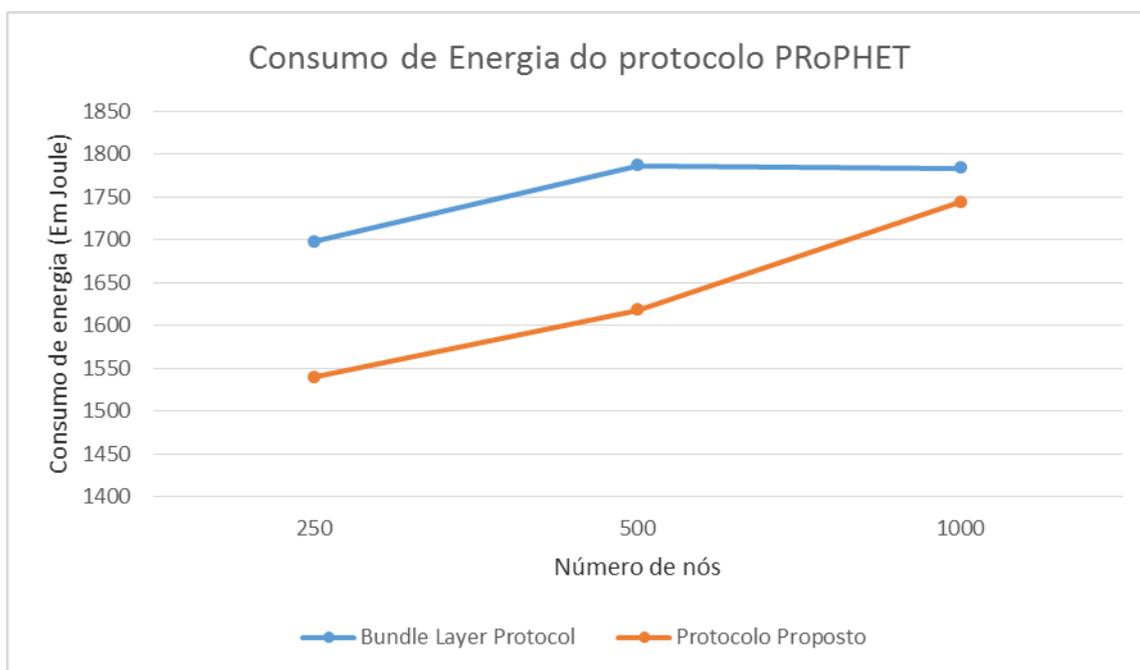


Figura 7-2 Consumo de energia do protocolo PProPHET. Fonte: Autor.

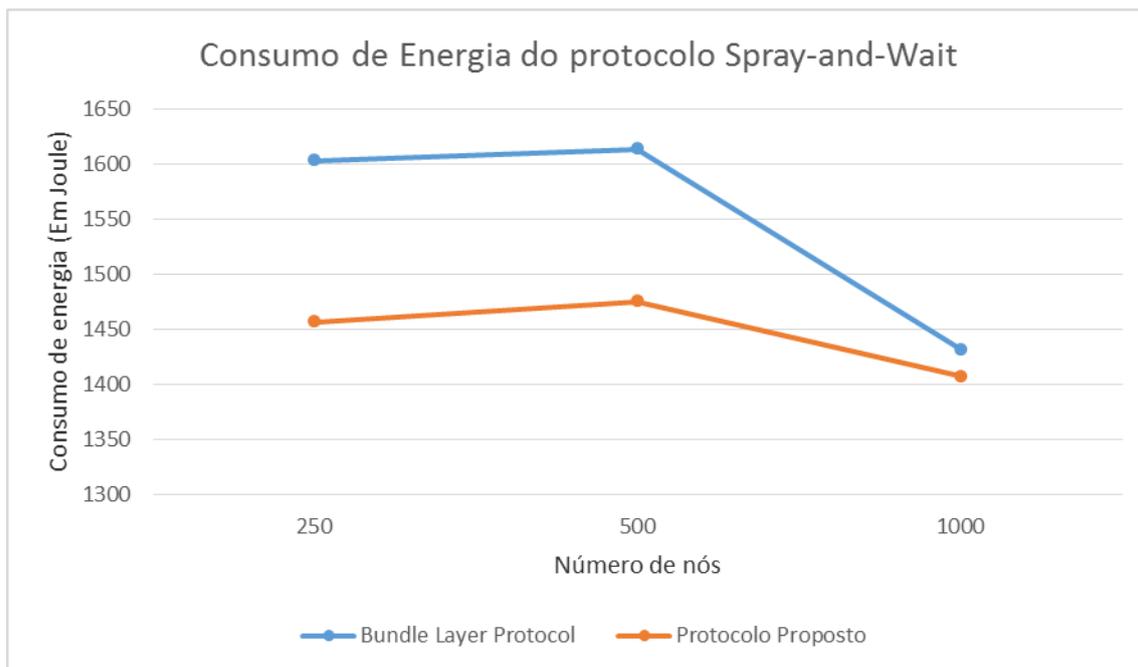


Figura 7-3 Consumo de energia do protocolo Spray-and-Wait. Fonte: Autor.

7.2.RESULTADOS DE PERFORMANCE

Juntamente com os testes de eficiência energética, foram realizados os testes de performance da rede. Esses testes visam analisar o desempenho em relação a entrega, roteamento e confiabilidade da rede. Nesses testes são avaliados o número de mensagens criadas, o número de mensagens entregues, a probabilidade de uma mensagem ser entregue com sucesso na rede, o atraso na entrega ao destino final, o *overhead* (sobrecarga) de mensagens repetidas entregues, o número médio de saltos (*hops*), e o tempo médio que uma mensagem fica no *buffer*.

Analisando as tabelas 7-1, 7-2, 7-3, 7-4, 7-5 e 7-6, é possível perceber que o protocolo proposto não obteve resultados muito diferentes do *Bundle Layer Protocol*. Embora vê-se um grande desperdício de recursos, já que o *overhead* muitas vezes chegou a níveis exorbitantes, o que acaba gerando um custo elevado em energia desperdiçada em enviar mensagens repetidas.

Tabela 7-1 Performance do Bundle Layer Protocol com 305 nós. Fonte: Autor.

	Epidemic	PRoPHET	Spray-and-Wait
Mensagens Criadas	487	487	487
Mensagens Entregues	82	116	330
Probabilidade de Entrega (%)	16,84	23,28	67,76
<i>Overhead</i>	1028,23	675,4	7,19
Número de saltos (<i>Hops</i>)	10,42	6,55	2,07
Tempo médio no buffer (em s)	54,33	41,05	337,62
Atraso (em s)	1059,34	2086,63	1892,23

Tabela 7-2 Performance do Protocolo Proposto com 305 nós. Fonte: Autor.

	Epidemic	PRoPHET	Spray-and-Wait
Mensagens Criadas	487	487	487
Mensagens Entregues	83	112	331
Probabilidade de Entrega (%)	17,04	22,99	67,96
<i>Overhead</i>	1217,68	695,3	7,70
Número de saltos (<i>Hops</i>)	11,68	7,17	2,19
Tempo médio no buffer (em s)	54,13	41,75	362,71
Atraso (em s)	1067,56	2053,63	1901,91

Comparando as tabelas 7-1 e 7-2 vemos os resultados em uma rede com poucos nós, pode-se perceber que o protocolo proposto acabou obtendo uma performance mais baixa, isso se deve ao fato de que com o número de

nós menor, a probabilidade de um nó com a mesma interface entre os nós agentes também diminui, fazendo mais evidente a necessidade de um caminho maior até chegar a um nó capaz de efetuar a conversão de interface se necessário.

Analisando as tabelas 7-3 e 7-4 é possível ver que ainda existe essa diferença de performance, porém a diferença entre eles é menor, e o roteamento P_{Ro}PHET inclusive obteve um desempenho melhor que o protocolo Bundle Layer em número de saltos, visto que com um número maior de nós disponíveis, a chance de um não ser necessária conversão para a realização da entrega aumenta.

Tabela 7-3 Performance do Bundle Layer Protocol com 585 nós. Fonte: Autor.

	Epidemic	P _{Ro} PHET	Spray-and-Wait
Mensagens Criadas	487	487	487
Mensagens Entregues	74	117	338
Probabilidade de Entrega (%)	15,20	24,02	69,40
<i>Overhead</i>	31725,33	16218	7,13
Número de saltos (<i>Hops</i>)	15,60	8,96	2,24
Tempo médio no buffer (em s)	35,84	30,51	4550,19
Atraso (em s)	1016,46	1913,64	2354,50

Já nas tabelas 7-5 e 7-6, uma rede com densidade elevada, pode-se ver como a diferença de performance entre os protocolos é mínima, devido a quantidade de nós com interfaces semelhantes por todo caminho de roteamento do ponto inicial ao final.

Porém, ao fazer uma análise geral dos resultados, é possível ver como os roteamentos *Epidemic* e *P_{Ro}PHET* acabam gerando uma quantidade de

mensagens extras bastante elevado, o que acaba ocasionando um custo em termos energéticos com mensagens desnecessárias.

Tabela 7-4 Performance do Protocolo Proposto com 585 nós. Fonte: Autor.

	Epidemic	PRoPHET	Spray-and-Wait
Mensagens Criadas	487	487	487
Mensagens Entregues	73	117	348
Probabilidade de Entrega (%)	14,98	24,02	71,45
<i>Overhead</i>	31781,16	16130,44	7,34
Número de saltos (<i>Hops</i>)	15,81	8,58	2,49
Tempo médio no buffer (em s)	35,89	30,86	4205,78
Atraso (em s)	1053,62	1303,20	2589,69

Tabela 7-5 Performance do Bundle Layer Protocol com 1120 nós. Fonte: Autor.

	Epidemic	PRoPHET	Spray-and-Wait
Mensagens Criadas	487	487	487
Mensagens Entregues	59	79	346
Probabilidade de Entrega (%)	12,11	16,22	71,05
<i>Overhead</i>	90433,49	49722,62	7,01
Número de saltos (<i>Hops</i>)	17,45	11,10	2,29
Tempo médio no buffer (em s)	29,46	27,08	4780,89
Atraso (em s)	759,35	1689,71	2242,15

Tabela 7-6 Performance do Protocolo Proposto com 1120 nós. Fonte: Autor.

	Epidemic	PRoPHET	Spray-and-Wait
Mensagens Criadas	487	487	487
Mensagens Entregues	59	79	346
Probabilidade de Entrega (%)	12,11	16,22	71,05
<i>Overhead</i>	90315,16	46034,44	7,01
Número de saltos (<i>Hops</i>)	17,76	11,13	2,23
Tempo médio no buffer (em s)	29,61	27,58	4780,89
Atraso (em s)	784,90	1671,14	2264,32

8.CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho apresentou a implementação de um protocolo DTN voltado para cidades inteligentes. Esse protocolo foi implementado num simulador para efetuação da validação e de testes de performance. Este protocolo foi desenvolvido com o intuito de melhorar a eficiência energética dos dispositivos de uma rede no âmbito de uma cidade inteligente, bem como aperfeiçoar a performance da própria rede.

Os resultados apresentados no Capítulo 7 demonstraram que o protocolo proposto apresentou uma melhora no consumo de energia, houve uma redução do gasto de até 10%. Isso mostra que mesmo com uma simplificação, talvez um novo paradigma seja necessário para maximizar a eficiência dos dispositivos numa cidade inteligente.

Também é possível perceber pelos resultados, que a escolha de um protocolo de roteamento é mais impactante na performance da rede, porém, como foi apresentado nos Capítulos 2 e 3, a maioria desses roteamentos ainda são focados para um tipo de aplicação ou modelo de rede, o que acaba tornando sua padronização um problema.

Foi possível também analisar algum dos fatores que influenciam a performance, o consumo energético de uma rede DTN e o reflexo das características de cada protocolo de roteamento em diferentes níveis de densidade de uma rede. Em relação à confiabilidade da rede pode-se perceber que muito ainda pode ser feito, para que se possa melhorar a garantia de entrega das mensagens.

Ao final, é possível garantir, que a densidade de uma rede afeta muito o resultado final de seu desempenho. Logo uma cidade inteligente, onde exista um povoamento muito grande de sensores, dispositivos móveis, centrais, entre outros, espalhados pela cidade, é possível imaginar que a performance de redes ponta-a-ponto aumentará gradativamente com o aumento dos nós disponíveis, tornando o uso cada vez melhor para o usuário final, com mais serviços prestados mais rapidamente, melhor tempo de resposta, entre outros.

8.1. TRABALHOS FUTUROS

- Otimizar a fragmentação. O procedimento de fragmentação utilizado numa rede DTN pode impactar o seu desempenho, evitando a perda de mensagens ou a necessidade de reenvio por falta de um canal que suporte o tamanho do bloco. Uma alternativa seria encontrar uma maneira otimizada de realizar essa fragmentação.

- Estudo de pontos críticos. Usando-se dos resultados de performance obtidos, pode-se perceber que alguns roteamentos e os protocolos sofreram impacto com o aumento do número de nós. Portanto, uma análise mais aprofundada sobre o comportamento de cada roteamento, em diferentes densidades e situações, poderia oferecer um maior conhecimento sobre possíveis fatores que podem ser aperfeiçoados para uma melhora no desempenho.

- Implementação de um protocolo de roteamento. Como foi possível visualizar nos resultados, um protocolo de roteamento pode impactar bastante na performance da rede. Modelar um protocolo que sirva para o maior número possível de aplicações numa rede DTN pode ser uma maneira de padronizar o uso da rede, aumentando sua visibilidade em projetos.

9.REFERÊNCIAS BIBLIOGRÁFICAS

ANTONIO de Oliveira Domingues, Marco; Fawzi Hadj Sadok, Djamel. Uma abordagem para validação de protocolos de comunicação em ambientes de simulação. 2004. Dissertação (Mestrado). Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Pernambuco, Recife, 2004.

A. BUJARI, C. E. Palazzi, D. MAGGIORINI, C. QUADRI, G. P. Rossi, "A Solution for Mobile DTN in a Real Urban Scenario", *WCNC 2012 Workshop*, April 2012.

ARAÚJO, H. S; CASTRO, W. L.; HOLANDA FILHO, R "Simulação em RSSF para Protocolos de Roteamento usando uma Abordagem Geocast." ERCEMAPI 2009 – Escola Regional de Computação Ceará – Maranhão – Piauí, 2009.

BARNAGHI, Payam; SHETH, Amit. The Internet of Things: The Story So Far. IEEE – Internet of Things. 09 set. 2014. Disponível em: <<http://iot.ieee.org/newsletter/september-014/theinternet-of-things-the-story-so-far.html>>. Acesso em: agosto. 2017.

Baumann, R., *Vehicular Ad Hoc Networks - Engineering and simulation of mobile ad hoc routing protocols for VANET on highways and in cities*, M.Sc. dissertation, Swiss Federal Institute of Technology Zurich, 2004.

Buttler, Peter. CSO magazine from International Data Group, <https://www.idg.com/news> julho, 2017.

Cerf, V.; Burleigh, S.; Hooke, A.; Torgerson, L.; Durst, R.; Scott, K.; Fall, K.; Weiss, H. "Delay-Tolerant Networking Architecture". Request for Comments: 4838, 2007, 35p.

CONDOR, "United States Marine Corps - Command and Control On-the-move Network Digital Overthe-horizon Relay (CONDOR)", Capturado em: <http://www.cengen.com/DEFENSE/condor.htm>, Outubro 2017.

DOSTÁLEK, L.; KABELOVÁ, A. *Understanding TCP/IP: A Clear and Comprehensive Guide*. [S.l.]: Packt Publishing, 2006. Paperback. ISBN 190481171X.

Fan, T. Chen, Y. A Scheme of Data Management in the Internet of Things 2nd IEEE International Conference on Network Infrastructure and Digital Content, 2010.

FARREL, Stephen. Delay and Disruption Tolerant Networking, Oct. 2006.

HiveMQ Enterprise MQTT Broker, <http://www.hivemq.com/> Maio, 2017

HRIBERNIK, Karl A.; GHRAIRI, Zied; HANS, Carl; THOBEN, K. First experiences in the participatory design of intelligent products with arduino. Proceedings of the 2011 17th International Conference on Concurrent Enterprising (ICE 2011).

KARIMI, R., ITHNIN, N., RAZAK, S., NAJAFZADEH, S., "DTN Routing Protocols for VANETs: Issues and Approaches", *IJCSI International Journal of Computer Science Issues*, v. 8, pp. 89-93, 2011

KEHUA, S. Smart City and the Applications. International Conference on Electronics, Communications and Controls (ICECC 2011).

KERÄNEN, A. Opportunistic Network Environment Simulator. Special Assignment report, Helsinki University of Technology, Department of Communications and Networking, May 2008.

Kurose, James F; Ross, Keith W. Redes de Computadores e a Internet: uma abordagem top-down. 3. ed. São Paulo: Person Addison Wesley, Brasil, 2006.

KUROSE, J. F.; ROSS, K. W. Computer Networking a Top-Down Approach. 5th edition. ed. [S.l.]: Addison-Wesley, 2009. Paperback. ISBN 0136079679.

- JAIN, R. *The art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modelling*. Wiley-Interscience, New York, NY, abr. 1991.
- JAIN, S., FALL, K., PATRA, R., "Routing in a Delay Tolerant Network". In: *ACM SIGCOMM (2004)*, pp. 145–158, New York, 2004.
- JUANG, P.; Oki, H.; WANG, Y.; MARTONOSI, M.; PEG, L. S.; RUBENSTEIN, D. "Energy-efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet". In: *10th Annual Conference on Architectural Support for Programming Languages and Operating Systems, 2002*, pp. 96–107.
- LAM, P.-K.; LIEW, S. UDP-liter: an improved UDP protocol for real-time multimedia applications over wireless links. In: *1st International Symposium on Wireless Communication Systems, 2004*. [S.l.: s.n.], 2004. p. 314–318.
- Lindgren, A.; Doria, A.; Schelén, O. "Probabilistic Routing in Intermittently Connected Networks". *SIGMOBILE Mobile Computing and Communications Review*, vol. 7-3, Julho 2003, pp. 19–20.
- Lindgren, A.; Doria, A.; Davies, E.; Grasic, S. "Probabilistic Routing Protocol for Intermittently Connected Networks". Internet-Draft: draft-irtf-dtnrg-prophet-04, 2009, 66p.
- MEHDI, KAMAK. CupCarbon: a Multi-Agent and Discrete Event Wireless Sensor Network Desing and Simulation Tool. SIMUTools 14th, Proceedings of the 7th ICST. 2014. Pages126-131.
- Nunes, C. M.; Link, E.; Dotti, F. L. "Evaluating the Impact of an acknowledgment Strategy for APRP". In: *5th International Latin American Networking conference, 2009*, pp. 77–86.
- Oliveira, C. T., Uma Proposta de Roteamento Probabilístico para Redes Tolerantes a Atrasos e Desconexões. M.Sc. dissertation, Universidade Federal do Rio de Janeiro, março 2008.
- Oliveira, C. T. de; Moreira, M. D. D.; Rubinstein, M. G.; Costa, L. H. M. K.; Duarte, O. C. M. B. "Redes Tolerantes a Atrasos e Desconexões". In: *Minicursos do XXV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2009*, pp. 203–256.
- PARKER, T.; SIYAN, K. *TCP/IP Unleashed*. 3rd edition. ed. [S.l.]: Publishing, 2002. Paperback. ISBN 0672323516.
- PAULO R. PEREIRA, JOEL J. P. C. RODRIGUES, Joan Triay, "From Delay-Tolerant Networks to Vehicular Delay-Tolerant Networks", *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, 2012.
- Pelusi, L., Passarella, A., Conti, M., "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks". *Communications Magazine, IEEE*, v. 44, pp. 134-141, 2006.
- RASPBERRY, Raspberry Pi 3. Disponível em: <https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/>. Julho ,2018
- RFC-EDITOR. Request for Comments Repository. Agosto 2017. Disponível em: <http://www.rfc-editor.org>.
- Rizvi, S., Olariu, S., Weigle, M., Rizvi, M., "A novel approach to reduce traffic chaos in emergency and evacuation scenarios". In: *IEEE Vehicular Technology Conference (VTC-Fall)*, pp. 1937–1941, Baltimore, 2007.
- S. SAGAR, S. H. AHMED, Z. A. KHAN, U. QASIM, I. I. AWAN, M. Y. DURANI, S. AHMED, M. D. SABIR, N. JAVAID, "Link and Path Duration of Routing Protocols in Mobile Ad-hoc Networks and Vehicular Ad-hoc Networks", 2014.
- SAYED FAWAD ALI SAHA, MOHAMMAD HASEEB ZAFAR, IVAN ANDONOVIC, TARIQULLAH JAN, "Hybrid routing scheme for Vehicular Delay Tolerant Networks", *Computer Science and Electronic Engineering (CEEC) 2016 8th*, pp. 158-163, 2016
- SCOTT, K. and BURLEIGH, S. (2006). Bundle protocol specification. Disponível em: <http://www.rfc-editor.org/rfc/rfc5050.txt>

Scott, K. "Disruption Tolerant Networking Proxies for On-the-Move Tactical Networks". In: Military Communication Conference, 2005, pp. 3226–3231.

SIM, Sistema Integrado de Mobilidade. Disponível em: <http://www.simsm.com.br>. Julho, 2018.

SPYROPOULOS, T., PSOUNIS, K., RAGHAVENDRA, C., "Spray and wait: an efficient routing scheme for intermittently connected mobile networks". In: *WDTN '05 Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 252-259, New York, 2005.

Spyropoulos, T.; Psounis, K.; Raghavendra, C. "Efficient Routing in intermittently Connected Mobile Networks: the Multiple-copy Case". *IEEE/ACM Transactions on Networking*, vol. 16-1, Fevereiro 2008, pp. 77–90.

SPURGEON, C. Ethernet: The Definitive Guide. 1st edition. ed. [S.l.]: O'Reilly Media, 2000. Paperback. ISBN 1565926609.

TANENBAUM, A. S. Computer Networks. 4th edition. ed. [S.l.]: Prentice Hall, 2002. Paperback. ISBN 0130384887.

TAN, L. and WANG, N. "Future Internet: The Internet of Things", 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE 2010), vol.5, pp.V5-376-V5-380, 20-22.

TAVARES, Pedro L. Redes de Sensores Sem Fio, *IEEE Communications Magazine*, 41(2):116–125, February 2003.

THE ONE, Simulator for opportunistic networks. Disponível em: <http://akeranen.github.io/the-one/>. Acesso em 7 de junho de 2017.

TP-LINK, TP-Link Products, Disponível em: https://www.tp-link.com/br/products/details/cat-9_TL-WR940N.html#specifications. Julho, 2018.

"UMass DieselNet", Capturado em: <http://prisms.cs.umass.edu/dome/umassdieselnet>, outubro 2017.

Waher, Peter. Learning Internet of Things Paperback. Packt Publishing Ltd. Birmingham Mumbai, 2015.

WHARTMAN, F. "Delay-Tolerant Networks (DTNs), A Tutorial, V1.1," May 2003. Available: http://www.ipnsig.org/reports/DTN_Tutorial11.pdf. Acesso em: março 2017.

Woodside Capital Partners. Disponível em: <http://www.woodsidecap.com/ma-and-the-internet-of-things-how-will-it-shake-out/>. Maio, 2017

XMPP community. Disponível em: <https://xmpp.org/about/technology-overview.html>. Julho 2017.

Zhang, X.; Kurose, J.; Levine, B. N.; Towsley, D.; Zhang, H. "Study of a Bus-based Disruption-Tolerant Network: Mobility Modeling and Impact on Routing". In: The Annual International Conference on Mobile Computing and Networking, 2007, pp. 195–206.