UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UMA ARQUITETURA DE COMUNICAÇÃO OPORTUNÍSTICA

DANILO AUGUSTO MOSCHETTO

ORIENTADOR: PROF. DR. HÉLIO CRESTANA GUARDIA

São Carlos - SP
Fevereiro/2010

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UMA ARQUITETURA DE COMUNICAÇÃO OPORTUNÍSTICA

DANILO AUGUSTO MOSCHETTO

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Sistemas Distribuídos e Redes.

Orientador: Dr. Hélio Crestana Guardia.

São Carlos - SP
Fevereiro/2010

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

M895ac

Moschetto, Danilo Augusto.
Uma arquitetura de comunicação oportunística / Danilo Augusto Moschetto. -- São Carlos : UFSCar, 2010.
87 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2010.

1. Redes de comunicação de dados. 2. Redes de computadores. 3. Sistemas de comunicação móvel. 4. Ad hoc networks (Redes de computação). I. Título.

CDD: 004.62 (20ª)

Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Programa de Pós-Graduação em Ciência da Computação

"Uma Arquitetura de Comunicação Oportunística"

DANILO AUGUSTO MOSCHETTO

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação

Membros da Banca:

Hélio Crestana Guardia

Prof. Dr. Hélio Crestana Guardia
(Orientador - DC/UFSCar)

Cesar Marcondes

Prof. Dr. Cesar Augusto Cavalheiro Marcondes
(DC/UFSCar)

Prof. Dr. Alfredo Goldman vel Lejbman
(IME/USP)

São Carlos
Fevereiro/2010

AGRADECIMENTOS

A minha família que sempre me apoiou em todas as decisões e em todos os caminhos que trilhei e que ainda irei trilhar.

Halan, nos últimos anos você passou de desconhecido a um verdadeiro irmão. Obrigado por tudo.

Aos amigos de Jaú e de São Carlos, que sempre estiveram presentes para tudo que precisei.

A todos aqueles que foram importantes durante esse período. Podemos não nos ver ou nos falar tanto, mas eu sempre serei grato pelo tempo em que estiveram em minha vida.

Aos colegas de mestrado. Muitas foram as noites de sono perdidas estudando e fazendo trabalhos, mas essas horas tão sofridas serão levadas pelo resto da minha vida como uma das fases mais importantes dela.

Não gosto de citar nomes, pois sempre nos esquecemos de alguém, mas não posso deixar de agradecer a algumas pessoas.

Roberto Rigolin e Pedro Nobile, pelas longas horas trabalhadas e pelos *Happy Hours* que se seguiam.

Ricardo, Élen, Mayra e Flávio. Sem vocês suportar o último ano teria sido muito mais difícil. Obrigado pelos inúmeros cafés, sinucas e conversas.

Aos professores que de alguma forma ajudaram na minha formação. Em especial ao Professor Hélio, que se mostrou merecedor da palavra “Orientador”. Obrigado pelos anos de convívio.

"Would you tell me, please, which way I ought to go from here?"
"That depends a good deal on where you want to get to," - said the Cat.
"I don't much care where" - said Alice.
"Then it doesn't matter which way you go," - said the Cat

Lewis Carroll -Alice no País das Maravilhas

RESUMO

Dispositivos computacionais móveis comumente apresentam elevada capacidade de processamento e disponibilidade de múltiplas interfaces de comunicação em rede. Sob o ponto de vista da transmissão de dados, redes 4G deverão prover uma infraestrutura unificada baseada no uso do protocolo IP, com suporte a segurança e QoS para esses dispositivos. Atrasos de propagação, rupturas e perdas de conectividade, contudo, ainda são comuns quando se considera a mobilidade dos usuários. Este trabalho apresenta uma arquitetura de comunicação tolerante a atrasos e conectividade intermitente, que explora o modo de encaminhamento ponto a ponto para prover transmissões em redes DTNs. A arquitetura é organizada em camadas, possibilitando o uso de diferentes políticas de encaminhamento e de diferentes tecnologias de transmissão. Ao prover o encaminhamento de objetos de transmissão, chamados *bundles*, um número ilimitado de serviços de comunicação em alto nível pode ser implementado sobre a arquitetura desenvolvida. Questões de segurança na identificação dos usuários e dos *bundles* transmitidos, bem como a confidencialidade e a autenticação das informações são considerados no projeto da arquitetura desenvolvida. O registro de operações realizadas durante os encaminhamentos, incluindo contatos entre usuários e eventos de transmissão, pode ainda servir para alimentar uma base de dados a ser explorada por diferentes algoritmos relacionados a organizações sociais e políticas de encaminhamento cientes de contexto. Uma implementação parcial das funcionalidades previstas pela arquitetura foi realizada e os resultados obtidos mostram a viabilidade de sua operação.

Palavras-chave: computação móvel, DTN, MANETs, comunicação oportunística

ABSTRACT

Mobile computing devices often present high processing power and include several wireless network interfaces. 4G networks promise to provide a unified IP based infrastructure with QoS and security support for the communications using such devices. Transmission delays, and network disruptions, however, are often common when considering the current transmissions of mobile users. This work presents a software communications architecture which is tolerant to transmission delays and intermittent connectivity by exploring a hop-by-hop forwarding mode in DTNs. The proposed architecture is organized as a set of functional layers, which allow the use of different forwarding policies and network technologies. By providing the forwarding of objects, named bundles, an unlimited number of high level communication applications can be implemented on top of the architecture. Security aspects concerning the proper identification of users and transmission bundles, as well as the confidentiality and authentication of the received information are considered in the project of the developed architecture. Log keeping of the bundle forwarding activities performed by the architecture, including the interactions among users and the data passing operations, may be further investigated for use in different context aware forwarding and social organization algorithms. A partial implementation of the developed architecture was made and the results demonstrate the viability of its operation.

Keywords: mobile computing, DTN, MANETs, opportunistic computing

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

API	*Application Programming Interface*
AES	*Advanced Encryption Standard*
DNS	*Domain Server Name*
DTN	*Delay/Disruption Tolerant Network*
GPS	*Global Positioning System*
HTTP	*Hypertext Transfer Protocol*
IP	*Internet Protocol*
IRTF	*Internet Research Task Force*
LBS	*Location-Based Services*
LTE	*Log-Term Evolution*
MAC	*Media Access Control*
MANET	*Mobile Ad hoc Network*
ms	*milissegundo*
RAM	*Random Access Memory*
RFC	*Request For Comments*
RMS	*Record Management System*
SMTP	*Simple Mail Transfer Protocol*
SQL	*Structured Query Language*
TCP	*Transmission Control Protocol*
TTL	*Time To Live*
UDP	*User Datagram Protocol*
UUID	*Universally Unique Identifier*
VANET	*Vehicular Ad hoc Network*
Wi-Fi	*Wireless Fidelity, nome dado ao padrão IEEE 802.11*
WLAN	*Wireless Local Area Network*
WPAN	*Wireless Personal Area Network*
XML	*eXtensible Markup Language*

SUMÁRIO

Capítulo 1

INTRODUÇÃO

1.1 Considerações Iniciais

O crescente número de dispositivos móveis com elevada capacidade computacional e contendo diferentes tecnologias de comunicação sem fio tem permitido o desenvolvimento de novas formas de comunicação em rede, as quais podem operar sem uma infraestrutura fixa para a troca de informações entre dispositivos.

Hoje em dia não é incomum encontrar *smartphones* e outros dispositivos móveis com processadores com velocidade próxima a 1Ghz, grande quantidade de RAM (256MB ou mais) e espaço de armazenamento (48GB ou mais).

No campo da comunicação, o Bluetooth tornou-se praticamente onipresente, estando disponível em quase todos os dispositivos. Além disso, Wi-Fi tem se tornado cada vez mais popular, estando disponível em cada vez mais aparelhos.

O uso dessas novas formas de comunicação, contudo, não é transparente para as aplicações, principalmente no que se refere à mobilidade dos usuários, à intermitência da conectividade e à seleção da tecnologia de comunicação a ser utilizada.

Esse cenário tende a se tornar cada vez mais complexo com o aumento no uso de novas tecnologias de comunicação, como as redes 4G (*Long Term*

Evolution, LTE) e WiMax (IEE 802.16) nas comunicações de longo alcance. Apesar disso, as atuais considerações de custo, cobertura e capacidade de transmissão devem continuar a existir principalmente em países em desenvolvimento, como no caso do Brasil.

Em paralelo, as tecnologias de comunicação de alcance mais restrito (para uso em WPANs e WLANs) vêm melhorando suas capacidades de transmissão e reduzindo o consumo de energia em suas operações, entre outros aspectos.

Considerando-se esse contexto, diversos mecanismos de comunicação *ad hoc* foram criados, tais como a comunicação oportunística, que complementa os serviços oferecidos por uma infraestrutura de comunicação (eventualmente indisponível), criando redes ocasionais em que cada dispositivo dentro de uma área de alcance é transformado em um ponto de acesso por onde os dados podem trafegar para chegar ao seu destino.

Para tentar aproveitar melhor eventuais oportunidades de comunicação, além de conseguir tratar melhor eventuais problemas de desconexão, uma nova classe de arquiteturas de rede vem sendo desenvolvida, comumente denominada como DTN (*Delay/Disruption Tolerant Network*).

Desenvolvidas como arquiteturas *overlay,* que operam acima das pilhas de protocolos existentes, DTNs são planejadas para prover a função de intermediários (*gateways*) no modo de encaminhamento “armazena e encaminha” (*store-and-forward*) entre nós que se encontram. Além disso, essas redes compostas por nós móveis também podem beneficiar-se do suporte de uma infraestrutura fixa de transmissão (Banerjee et al, 2008).

Apesar de diversos esforços estarem sendo empregados no desenvolvimento de redes com essas características, ou mesmo de redes *mesh*, em que nós podem intermediar as transmissões de outros nós a um ponto de acesso, não há uma solução definitiva para o problema da tolerância a atrasos e à conectividade intermitente.

Este trabalho apresenta uma nova arquitetura de comunicação, capaz de operar em conjunto com as atuais tecnologias de infraestrutura de rede e de permitir a entrega de informações através de um encaminhamento ponto a ponto. Sua utilização pode ser apropriada para diferentes aplicações de rede.

Operando principalmente em dispositivos móveis, a arquitetura desenvolvida cria uma rede oportunística que utiliza interfaces de comunicação genéricas para o envio e recebimento de informações (*bundles*). A operação da rede ocorre de maneira transparente sobre as tecnologias de comunicação atuais, sendo que os dados a serem transmitidos trafegam entre os nós sem a interferência dos usuários, valendo-se da mobilidade dos mesmos para alcançar o destino em cada caso.

A arquitetura que se apresenta foi utilizada para a criação de uma série de serviços que foram avaliados em caráter experimental. Os serviços desenvolvidos foram instrumentados para permitir a coleta de informações também sobre o cenário, sobre os encontros ocorridos entre os usuários e os intervalos de ausência de conectividade dos nós, e deverão servir para o desenvolvimento de novos protocolos e políticas de encaminhamento de dados.

1.2 Cenários

Como exemplo de cenário relevante para a arquitetura de comunicação proposta, considere um dia típico da na vida de uma pessoa, João, que trabalha num campus de uma grande universidade. Sua rotina consiste em ir de metrô até o campus, desenvolver suas atividades de suporte na área de informática e retornar ao final do dia para a sua casa.

João se levanta de manhã e utiliza sua rede sem fio para fazer o *download* dos seus e-mails para o seu celular e sai para o trabalho. No caminho, aproveita o tempo no metrô para ler e responder as mensagens. Como não há nenhuma conexão com a Internet disponível, o celular dele envia parte das mensagens via Wi-Fi a outras pessoas próximas, para que estas encaminhem os *e-mails* quando chegarem a um ponto de acesso.

Apesar de João não conhecer essas pessoas, elas fazem parte de uma mesma comunidade social na Internet, definida pela interação circunstancial constante. Por isso, ao encontrar os dispositivos dessas pessoas, o celular de

João consegue trocar informações com os dispositivos dessas pessoas e descobrir que existe uma grande chance deles se conectarem à Internet logo. Assim, alguns e-mails são encaminhados a esses dispositivos para que eles os enviem aos destinatários quando se conectarem à Internet.

Como alguns *e-mails* possuem grandes arquivos como anexos, nem todos são encaminhados, ficando parte deles armazenados na memória do celular, esperando para serem encaminhados. Ao chegar ao campus, João encontra José, um colega de trabalho para quem tinha um e-mail para enviar, e que ainda estava na memória de seu celular. Ao descobrir o celular de José, o celular de João faz o envio do e-mail diretamente através de uma conexão Bluetooth.

Durante o dia, enquanto João está num setor onde não há sinal de celular, uma emergência faz com que Maria precise se sua presença. Como ninguém sabe a localização de João e seu celular não responde, ela utiliza um sistema de localização da universidade. Esse sistema consiste de um conjunto de dispositivos (fixos e móveis, celulares, por exemplo) que periodicamente envia a um servidor a sua posição (através de GPS) e a lista dos dispositivos Bluetooth que estão próximos.

Ao consultar o sistema, Maria descobre que a última localização de João é dentro do departamento de química, 3 minutos antes. Ela pega o telefone, faz uma ligação para o departamento em questão e consegue localizá-lo.

Durante o dia, João envia uma mensagem para um grupo de colegas de trabalho, marcando uma *Happy Hour* para depois do trabalho. Como nem todos nesse grupo têm acesso à Internet durante o trabalho, a mensagem é encaminhada diretamente a um dos colegas, via Bluetooth, quando João o encontra no almoço. A partir disso, ela vai sendo encaminhada a cada encontro entre as pessoas, sendo difundida entre todos os interessados até o final da tarde.

Essa história, apesar de simples, demonstra diversos casos em que o uso da comunicação oportunística permitiria aos usuários atingir os seus objetivos de comunicação de maneira mais rápida, objetiva ou barata.

Esse cenário pode ser generalizado conforme a Figura 1, onde uma mensagem se utiliza de diversos métodos de envio para chegar aos seus destinatários.

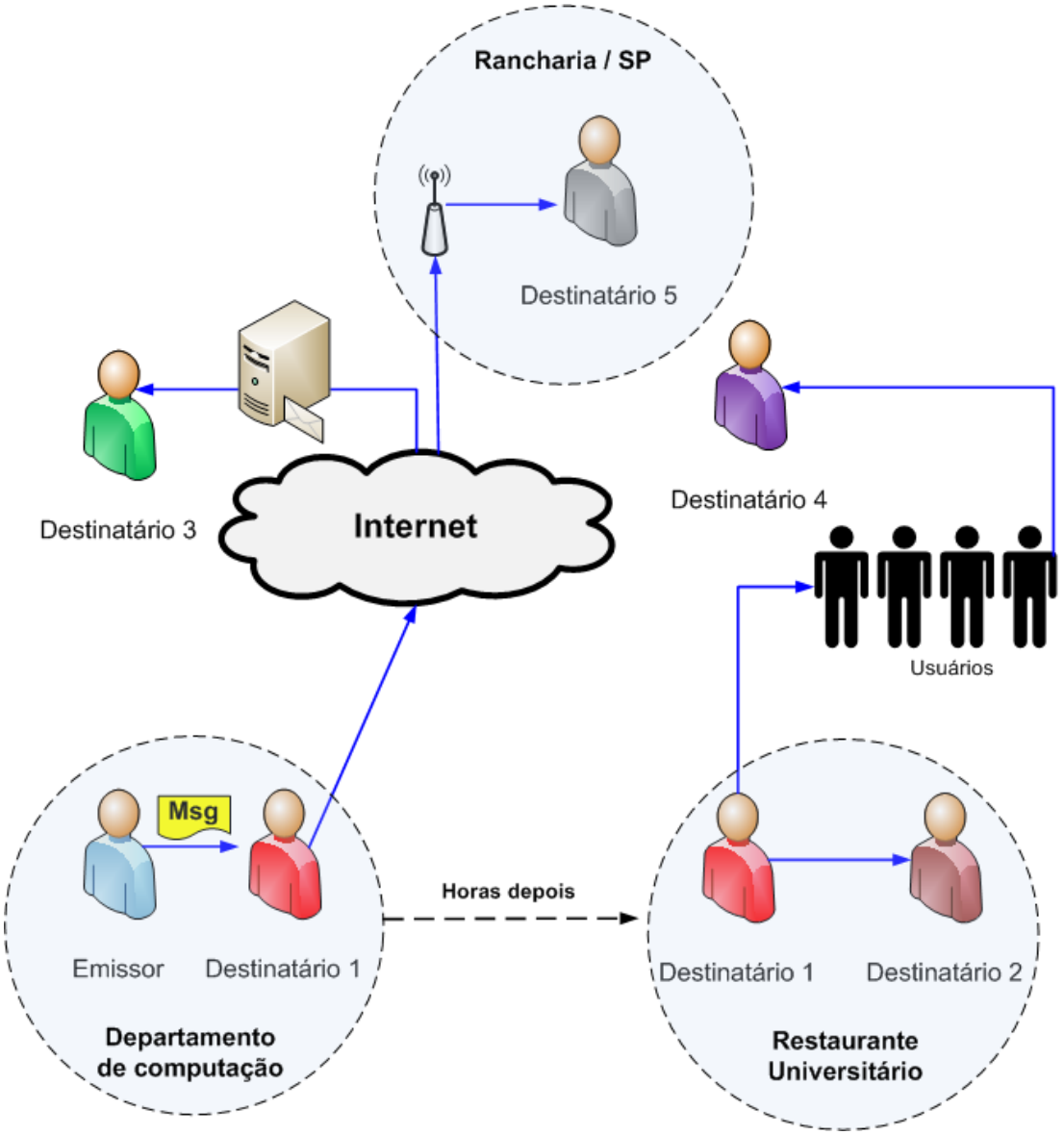

Figura 1 - Cenário de uso da arquitetura de comunicação desenvolvida

A Figura 1 apresenta de maneira ilustrativa as três formas de encaminhamento que a arquitetura desenvolvida nesse trabalho é capaz de operar. Essas formas estão descritas no capítulo 3.

A arquitetura apresentada nesse trabalho foi desenvolvida para operar em cenários como o descrito acima. Esses cenários se constituem tipicamente de ambientes urbanos, onde os participantes se encontram com alguma frequência, mesmo que por poucos instantes. Essas interações podem ser

vistas como redes de relacionamento virtuais, que são utilizadas para encaminhar dados entre os usuários, aproveitando-se da mobilidade dos próprios participantes para atingir o seu destino.

1.3 Objetivos

Nesse trabalho buscou-se o desenvolvimento de uma arquitetura de comunicação com suporte à comunicação oportunística que pudesse operar em diversos cenários, permitindo que a comunicação entre os nós pudesse ocorrer sem a intervenção do usuário.

Essa arquitetura foi desenvolvida para que a sua implantação pudesse ocorrer com o mínimo de alterações possível nos ambientes alvo. Além disso, a arquitetura deveria ser compatível com a infraestrutura de rede atual da Internet, baseada em TCP/IP, permitindo que os dados trafegados por ela sejam enviados através dos mecanismos usuais, como o email por exemplo.

A arquitetura de comunicação desenvolvida, num primeiro momento, permite a comunicação ponto a ponto entre os nós participantes, bem como a comunicação com alguns pontos de infraestrutura.

1.4 Metodologia

A estratégia de implementação foi baseada num primeiro momento num levantamento bibliográfico das arquiteturas de comunicação oportunística e em suas características funcionais. Isso incluiu a pesquisa de protocolos para a definição e a transmissão de objetos e algoritmos relacionados ao encaminhamento desses objetos.

Questões relacionadas à identificação dos nós e aplicações, bem como à segurança das transmissões no que se refere à autenticidade dos nós

comunicantes e à confidencialidade das transmissões também foram investigadas.

Numa etapa posterior, foram investigados os serviços necessários para que cada elemento da arquitetura desenvolvida pudesse operar de maneira coordenada e em conjunto. Como resultado, buscou-se a formalização de uma arquitetura de serviços para comunicações em ambientes oportunísticos, que foi então implementada e avaliada.

Os testes foram feitos em ambientes reais. O uso do ambiente real, apesar de permitir apenas testes mais restritos, serviu para testar o comportamento real da implementação da arquitetura e o seu comportamento. Um emulador foi utilizado durante o desenvolvimento, mas a relação direta do trabalho com o comportamento dos dispositivos impediu o seu uso a coleta de dados durante os testes.

1.5 Organização do trabalho

No capítulo 2, é feita uma revisão sobre o atual estado das áreas de comunicação oportunísticas e MANETs. Alguns projetos têm suas funcionalidade e características descritas a fim de demonstrar uma visão geral do ponto atual da área.

O capítulo 3 apresenta as diversas características que as arquiteturas oportunísticas podem apresentar. Depois é apresentada a arquitetura de comunicação desenvolvida no trabalho, juntamente com as suas características e funcionalidades.

Uma implementação de referência foi desenvolvida e está descrita no capítulo 4. Além disso, esse capítulo traz a descrição e os resultados das avaliações de viabilidade e desempenho.

O capítulo 5 apresenta as conclusões e discussões sobre o trabalho, além de tópicos para trabalhos futuros.No capítulo 6 estão as referências do trabalho.

Capítulo 2

MANETS E REDES OPORTUNÍSTICAS

2.1 MANETs e Redes Oportunísticas

Comunicações circunstanciais, sem depender de uma infraestrutura fixa de transmissão, são interessantes para diferentes aplicações. Encaminhamento de mensagens, acesso remoto e compartilhamento de informações são exemplos de aplicações que podem beneficiar-se dessa forma de transmissão de dados.

Essa forma de comunicação porém apresenta diversos desafios do ponto de vista técnico. Por exemplo, além de questões de estabelecimento de um canal de comunicação e da criação de caminhos para propagação de pacotes, atrasos e conectividade intermitente são problemas a serem tratados.

Outra forma de comunicação que vem ganhando espaço com o aumento do uso dos dispositivos móveis são as Redes Oportunísticas (*opportunistic network*). Uma rede oportunística é uma rede normalmente desconexa, organizada de forma dinâmica, à medida que os encontros entre os participantes vão acontecendo.

Na comunicação oportunística, não é assumida a existência de um caminho completo entre os nós que desejam se comunicar. Emissor e destinatário podem nunca estar na mesma rede, ao mesmo tempo (Pelusi *et al*,

2006) e os encaminhamentos de mensagens podem ocorrer de forma indireta e não instantânea.

Dispositivos móveis são tipicamente os grandes beneficiários desse tipo de comunicação, uma vez que a mobilidade dos usuários associa-se implicitamente às diversas características que essa forma de comunicação apresenta. Problemas de cobertura e custo de acesso à rede infraestruturada costumam ser grandes empecilhos aos seus usuários.

Por esse motivo, a formação de redes sem a necessidade de infraestrutura, com dispositivos móveis, vem sendo estudada e implementada ao longo dos anos. Essas redes são tipicamente chamadas de MANETS.

MANETs e VANETs são exemplos de redes que consideram os problemas dessa forma de comunicação. Alguns projetos encontrados na literatura tratam da criação de ambientes de transmissão com conectividade intermitente e há propostas de protocolos para padronizar essa forma de comunicação.

MANETs, *Mobile Ad hoc Networks*, são caracterizadas como redes móveis de comunicação, que podem ser formadas sem a necessidade de infraestrutura pré-existente (Loyal 2004). Nessas redes, dispositivos com interfaces de comunicação sem fio interagem e formam redes circunstanciais com outros nós ativos na mesma região no mesmo período.

As características das MANETs, em especial a sua capacidade de operar de maneira independente fizeram com que a sua utilização em cenários militares e locais atingidos por catástrofes fosse bastante estudada e desenvolvida.

Além desses cenários, com a popularização dos dispositivos móveis, a criação de redes entre usuários com interesses comuns vem ganhando força, não só pela possibilidade de operar sem a utilização de infraestrutura, mas também pela possibilidade de ter acesso a serviços que seriam inviáveis nas arquiteturas de comunicação convencionais.

Uma variação desse conceito de rede são as VANETs (*Vehicular Ad hoc Networks*), que são redes de concepção próxima às MANETs, sendo porém executadas sobre dispositivos ligados a veículos. Apesar de pouca diferença

na definição dos termos, essa modificação traz implicações importantes para essa categoria de rede.

Por estarem ligados a veículos, nas VANETs os dispositivos normalmente possuem limitações menores de energia e tamanho físico. Por isso, costumam ser dispositivos com maior poder computacional e maior capacidade de comunicação. Além disso, veículos podem apresentar um padrão de movimento cíclico, o que fornece informações importantes à arquitetura de comunicação. Ônibus de linha ou veículos que são usados apenas durante parte do dia (para ir e voltar ao trabalho, permanecendo parado nos demais horários) são exemplos desse padrão de movimento.

No projeto DieselNet, (Burgess *et al.* 2006) por exemplo, um conjunto de cerca de 40 ônibus é utilizado para formar uma VANET e, sobre essa rede, uma série de experimentos é feita com o objetivo de comparar algoritmos de encaminhamento de pacotes. Diversas técnicas são utilizadas na construção dos protocolos analisados, tornando o protocolo proposto bastante robusto.

Nesse trabalho, o cálculo da probabilidade de entrega é o fator principal na ordenação dos pacotes a serem entregues a um nó na rede. Esse dado, porém, é ponderado levando em consideração outros fatores, como proximidade (se um pacote é para o nó em questão, entre outros fatores) e distância em “saltos” já percorrida (pacotes que foram pouco transmitidos são favorecidos), que são utilizados para diminuir o tempo médio de entrega e evitar que a entrega de um pacote seja postergada por períodos possivelmente indefinidos.

Projetos como o Data MULEs (Shah *et al.*, 2003) também trazem considerações e formalizações importantes sobre o tempo necessário para que um nó percorra uma região e encontre um ponto para a entrega de pacotes.

Nesse trabalho são feitas considerações importantes sobre as comprovações teóricas da capacidade de entrega de dados através da mobilidade humana em MANETs.

2.2 Projetos de padronização em redes oportunísticas

A RFC 4838, definida pela IRTF (*Internet Research Task Force*), apresenta uma arquitetura de rede tolerante a atraso que opera no sistema de "armazena e encaminha". Detalhes sobre a interconexão das diversas sub-redes e do sistema de endereçamento dos nós dessa rede também são apresentados. Para operar nessa arquitetura, os dados são unidos em blocos de dados, chamados Bundles (RFC 5050).

O desenvolvimento da Arquitetura DTN (RFC 4838) traz diversos pontos a serem analisados quando se discute uma arquitetura de comunicação oportunística, como, por exemplo, o uso de mensagens de comprimento variável ao invés de um fluxo de mensagens menores, comum em outros protocolos. Em redes oportunísticas esse tipo de abordagem é interessante, uma vez que muitas vezes o canal de transmissão tem uma capacidade considerável (sendo capaz de transferir alguns megabytes em segundos), principalmente se os dados trafegados precisam estar completos para serem aproveitados (um programa, por exemplo).

A capacidade de armazenar dados para depois encaminhar para um próximo nó é a base de funcionamento da comunicação oportunística.

Outros pontos da arquitetura DTN buscam implementar melhorias no funcionamento da rede. Para isso, buscam minimizar as transferências entre os nós (diminuindo o *overhead* da comunicação), informar a rede sobre aspectos relevantes dos dados trafegados e manter a rede em bom funcionamento após falhas.

A arquitetura DTN é uma boa base de análise para sistemas que exploram a comunicação tolerante a atrasos ou a comunicação oportunística.

De maneira simplificada, os dados são transportados em *bundles,* que são unidades de dados de comprimento variável contendo informações completas (um e-mail ou um arquivo, por exemplo) ou partes de um arquivo maior (de um vídeo), ao invés de pacotes de rede tradicionais como no conjunto TCP/IP.

Essa abordagem facilita o armazenamento e encaminhamento utilizado, uma vez que cada *bundle* contém as informações necessárias sobre a mensagem a ser entregue.

2.3 Protocolo Bundle

Todo o trabalho que envolve o gerenciamento dos *bundles* é descrito na RFC 5050, que define o "*Bundle Protocol*".

Os *bundles* são processados por uma camada abaixo das aplicações. Essa camada, chamada de camada *bundle* (*Bundle Layer*) faz o tratamento dos pacotes e o seu encaminhamento quando necessário. Na Figura 2 podemos ver a localização da camada *bundle* em relação às demais camadas e ao modelo TCP/IP.

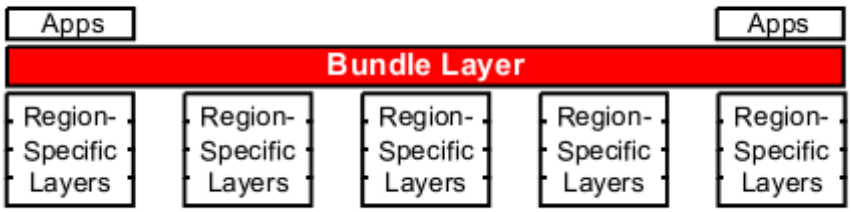

Figura 2 - Disposição da Camada Bundle

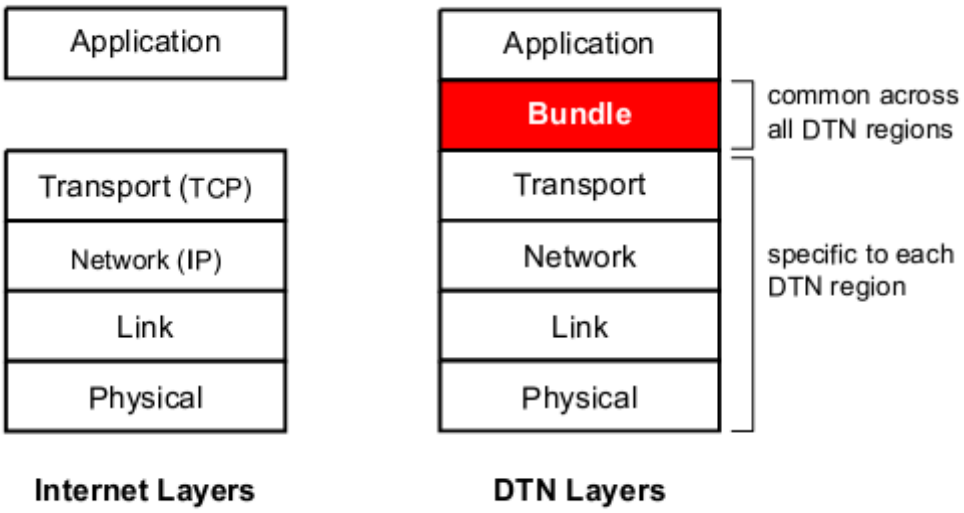

Figura 3 - Posição da Camada Bundle em relação ao TCP/IP

Como pode ser observado na Figura 2, a Camada Bundle opera como uma rede *overlay*, sobre a infraestrutura de rede existente em cada região. Na Figura 3 podemos observar a sua posição em relação à estrutura de camadas da Internet, onde o seu caráter de sobreposição é demonstrado, estando situada abaixo apenas da camada de aplicação.

Os *bundles* são compostos por três partes: os dados da aplicação de origem, informações que permitem à aplicação de destino decidir sobre como armazenar, tratar e usar os dados enviados, e um cabeçalho do protocolo *bundle,* utilizado pela Camada Bundle. Assim como os dados dos programas, *bundles* não têm um limite máximo de tamanho.

2.3.1 Transferência de Custódia:

Por conta da utilização de diversos meios de transmissão e protocolos distintos, a confiabilidade da entrega dos pacotes só pode ser garantida pela Camada Bundle, sendo que para isso é utilizado um sistema de custódia de *bundles.*

De maneira simplificada, a custódia do pacote garante que até o nó onde sem encontra, o pacote foi transferido de maneira correta e, caso algum problema na transmissão ocorra, ele pode ser corrigido daquele ponto.

Um nó da Camada Bundle tem de armazenar um pacote até que outro nó aceite a custódia do pacote, ou até que o tempo de vida (TTL) do pacote expire.

A transferência de custódia sozinha não é capaz de prover confiabilidade na entrega fim-a-fim. Para isso, é necessário a transferência de custódia e um Aviso de Recebimento. Nesse caso, o emissor deve manter uma cópia do pacote até que o Aviso de Recebimento chegue.

2.3.2 Classes de Serviço (Class of Service, CoS)

A camada Bundle permite seis Classes de Serviços para um *bundle*, que podem ser definidas como propriedades de um bundle. As seis classes de serviços especificadas pelo Protocolo Bundle são:

1. Transferência de custódia: após a transferência do *bundle* para um nó que tenha aceitado a sua custódia, o nó que encaminha o pode liberar os recursos que o *bundle* ocupa. O nó que aceitou a custódia envia um “Ack” para o emissor, avisando que o pacote foi entregue com sucesso;
2. Aviso de Recebimento: há o envio de uma notificação ao nó fonte (ou outro nó indicado) que o *bundle* foi entregue ao seu destino de maneira correta;
3. Notificação de transferência de custódia: há uma notificação ao nó fonte (ou a outro indicado) quando outro nó aceita a custódia de um *bundle*;
4. Notificação de encaminhamento: há uma notificação ao nó fonte (ou a outro indicado) sempre que o *bundle* é encaminhado a outro nó;
5. Prioridade de entrega: Baixa, Média e Alta;
6. Autenticação: algum método (Assinatura digital, por exemplo) é usado para verificar a identidade do emissor e a integridade do *bundle*;

2.4 O projeto Haggle

A arquitetura de comunicação especificada pelo projeto Haggle tem como objetivo permitir a interação entre dispositivos sem a necessidade de infraestrutura de transmissão. Para isso, a arquitetura proposta pelo projeto apresenta um sistema que difere radicalmente da organização em camadas do TCP/IP.

A arquitetura proposta é organizada em torno de seis gerenciadores (*managers*) que se comunicam entre si. Esses gerenciadores têm acesso às redes e a parte dos dados armazenados no dispositivo, permitindo que eles sejam transmitidos sem a necessidade de interação com o usuário.

Os gerenciadores (*Data*, *Name*, *Connectivity*, *Protocol*, *Forwarding*, *Resource*) são dipostos da seguinte maneira:

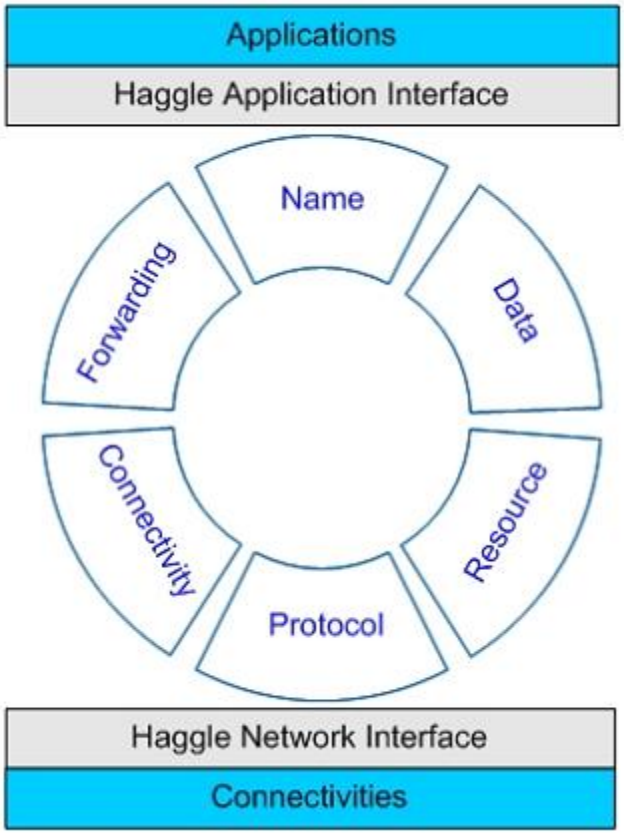

Figura 4 - Arquitetura Haggle

Na Figura 4 é possível analisar a distribuição dos gerenciadores e a sua interação com as camadas superiores e inferiores. Os seis podem se

comunicar através de um conjunto de chamadas definidas na especificação do projeto.

2.4.1 Data Manager

Ao invés de utilizar um sistema de arquivos separado, o próprio Haggle armazena os dados de forma persistente. Esses dados são mantidos numa estrutura que os torna, além de estruturados (como em um sistema de arquivos), facilmente localizáveis.

Existem relacionamentos entre as unidades de dados (por exemplo, um e-mail e uma foto anexa) e estes relacionamentos são representados nas estruturas. Os dados em cada um dos nós precisam ser acessíveis aos demais nós, inclusive para buscas.

No projeto Haggle os dados estão divididos em unidades chamadas Objetos de Dados (*Data Object*, DO). Os DO são compostos por uma série de campos e seus valores, que podem ser strings ou dados binários. Os DOs são ligados de duas maneiras diferentes.

A primeira é a ligação que existe entre os diversos DO que compõem uma estrutura maior. Por exemplo, uma página web e seus objetos ou um e-mail e seus anexos.

A segunda é a ligação entre um programa e seus dados. Nesse caso, a ligação indica “propriedade” de um DO por um programa específico. Várias aplicações podem ser “donas” de um mesmo DO ao mesmo tempo. Por exemplo, um programa de imagens pode ser ligado a uma foto que é anexa de um e-mail.

Como vários programas podem estar ligados a um mesmo DO, Haggle não permite a exclusão dos DO pelos programas, oferecendo apenas um comando que permite se desassociar (*unlink*) de um DO. Caso nenhum programa esteja associado a um DO, ele pode ser marcado para exclusão futura, por uma rotina de serviço.

Além da capacidade de recuperar um DO através de um identificador único, o *Data Manager* permite buscas num sistema parecido com o utilizado

no SQL. O usuário (ou o programa) passa as informações que deseja e o sistema retorna para ele um conjunto de DOs que se enquadram nos requisitos. Ex. “mime-type” EQUALS “text/HTML” AND “keywords” INCLUDES “news”.

2.4.2 Name Manager

No projeto Haggle, a identificação dos nós não é feita da maneira comum, através de diversos cabeçalhos aninhados. Como ele se propõe a utilizar qualquer possibilidade de conexão *ad hoc* e alguns serviços de identificação necessitam de uma infraestrutura centralizada (DNS, por exemplo), um mecanismo de identificação mais sofisticado se faz necessário.

Normalmente, um mesmo destinatário possui diversas identificações, dependendo do modo que a entrega será feito. Por exemplo, o endereço IP, o e-mail, o endereço MAC ou alguma outra forma de identificação. No Haggle, todos esses “nomes” são armazenados em uma estrutura em forma de grafo, que identifica o destinatário. Qualquer nome do grafo pode ser usado como endereço, desde que haja um protocolo que faça uso daquela forma de endereçamento. Para enviar algum dado a outro usuário, a aplicação apenas indica um dos nomes do grafo (podendo inclusive ser o nome da pessoa). Quando o Haggle encontra uma forma de transmitir o dado, escolhe de forma automática o “endereço” a ser utilizado.

Quando encontra outros nós da rede, o gerenciador pode trocar informações a fim de completar ou adicionar grafos da sua estrutura de nomes.

2.4.3 Connectivity Manager

Como o projeto Haggle pretende dar suporte a diversas tecnologias de rede ao mesmo tempo é necessário um gerenciador capaz de identificar e utilizar corretamente as características de casa tecnologia. Além disso, é necessário fornecer uma interface para que os outros gerenciadores utilizem os serviços da rede. Esse serviço é feito pelo *Connectivity Manager*.

2.4.4 Protocol Manager

A função do *Protocol Manager* é encapsular um conjunto de protocolos e inicializá-los de forma correta. Nesse caso, protocolo é toda forma pela qual os DO podem ser transmitidos a outros nós da rede através de um nome. Por exemplo, há suporte para SMTP, HTTP, etc.

Cada protocolo monitora a presença de vizinhos através das conexões e a partir dessas informações é decidido qual DO enviar para cada nó.

2.4.5 Forwarding Manager

Forwarding Manager é responsável por encapsular informações sobre os DO a serem transferidos, incluindo os respectivos grafos de nomes dos destinatários. Esse encapsulamento cria um objeto chamado de *Forwarding Object* (FO). Um FO contém, além dos dados a serem enviados e dos destinatários, metadados sobre a transmissão. Esses metadados podem incluir, por exemplo, informações sobre expiração, dicas de rotas e uma lista dos nós que já receberam esse FO.

Após a criação do FO, para cada protocolo disponível é gerada uma tupla {Protocolo, NO, vizinho) com cada vizinho para o qual esse dado pode ser entregue. Além disso, para cada tupla é calculado um número que representa a probabilidade estimada de que com esse vizinho e esse protocolo seja realizada a entrega fim-a-fim do NO.

Após isso são criadas as “Tarefas de encaminhamento” (*Forwarding Tasks*) para serem executadas quando o *Resource Manager* decidir.

2.4.6 Resource Manager

Todas as tarefas de acesso à rede do Haggle são executadas pelo *Resource Manager*. As tarefas são compostas dos dados a serem transmitidos, do método pelo qual a transferência será feita, dos custos e dos benefícios associados a essa tarefa.

Os custos medem o quanto de recurso (bateria, tempo, dinheiro) será gasto para a execução da tarefa, enquanto os benefícios representam a utilidade estimada da execução da tarefa para o usuário final. Ambos devem ser recalculados sempre que o gerenciador considerar a execução da tarefa, uma vez que variam de acordo com o tempo e outras condições.

As tarefas podem ser assíncronas ou imediatas. Tarefas assíncronas são as mais comuns, podendo ser atrasadas (por tempo indeterminado) em preferência de outras tarefas. Tarefas imediatas são aquelas que precisam ser executadas no momento, ou nunca mais poderão ser executadas. Entre elas estão, por exemplo, pedidos de conexão que precisam ser aceitos ou rejeitados.

2.5 Saratoga

Saratoga é um protocolo de disseminação de conteúdo desenvolvido para uso em satélites de órbita baixa. Sua utilização, porém, mostrou-se útil também para outros cenários, onde desconexões podem ser frequentes ou existe uma assimetria de conexões grande demais para o protocolo TCP operar. (Wood *et al*, 2007b)

Saratoga é baseado em UDP (RFC 768) e UDP-lite (RFC 3828), gerando pouca sobrecarga de transmissão. De maneira geral, é utilizado em ambientes com conexões esporádicas ou intermitentes.

A recuperação de pacotes perdidos é feita através de um mecanismo simples de ACK negativo. Em sua operação, pode atuar sobre conexões altamente assimétricas, sendo capaz inclusive de operar em modo unidirecional.

Quando utiliza conexões dedicadas, o Saratoga procura maximizar as transmissões enquanto o *link* está disponível. Protocolos de congestionamento podem ser utilizados em casos de conexões compartilhadas.

Em uso desde 2004, o Saratoga é empregado nas transmissões principalmente em satélites do DMC (*Disaster Monitoring Constellation*). Até

setembro de 2007, cinco satélites estavam em órbita, com mais três em construção, todos usando IP como *payload* (Wood *et al*, 2007a).

Através da escolha do tamanho do *offset* do arquivo, podem-se transferir de forma eficiente arquivos com poucos megabytes até vários de Gigabytes. Para isso, são suportados descritores com 16 e 32 bits. Versões futuras suportarão até 128 bits, o que deve evitar problemas futuros com o tamanho dos arquivos.

Tanto Ipv4 quanto Ipv6 são suportados para transmissões com Saratoga e o seu uso em vários links que carregam tráfego IP já foi testado e aprovado.

2.5.1 Composição

Os nós do Saratoga são simples servidores de arquivos, suportando diversas operações com arquivos, incluindo *download* (*pull*)*, upload* (*push*), listagem de diretórios e exclusão de arquivos. Cada uma dessas operações é considerada uma transação independente pelo protocolo.

Uma transação pode começar com um pedido de “_get_”, “_put_”, “_getdir_”, ou “_delete_”.

A transação mais comum é a “_get_”, que começa com o envio de um pacote “REQUEST” que é enviado pelo nó que deseja receber o arquivo paro o nó que contém o arquivo. Caso a transação seja rejeitada, um pacote de metadados, que informa a rejeição, é enviado.

Caso a transação seja aceita, um pacote de metadados é enviado de volta com informações úteis sobre a transação e o arquivo, seguido de uma sequência de pacotes com o arquivo em requisitado.

Essa sequência de pacotes inclui (e termina com) alguns pacotes de dados que contém um *flag* que pede ao receptor que envie ao emissor um relatório de recepção, no formato do pacote “HOLESTOFILL”, que é um NACK.

Após o recebimento do pacote “HOLESTOFILL”, o servido começa uma série de retransmissões seletivas até que o “HOLESTOFILL” informe que todos os pacotes foram entregues de maneira correta.

Exemplo

No exemplo a seguir, o pacote de número 2 é perdido durante a transmissão.

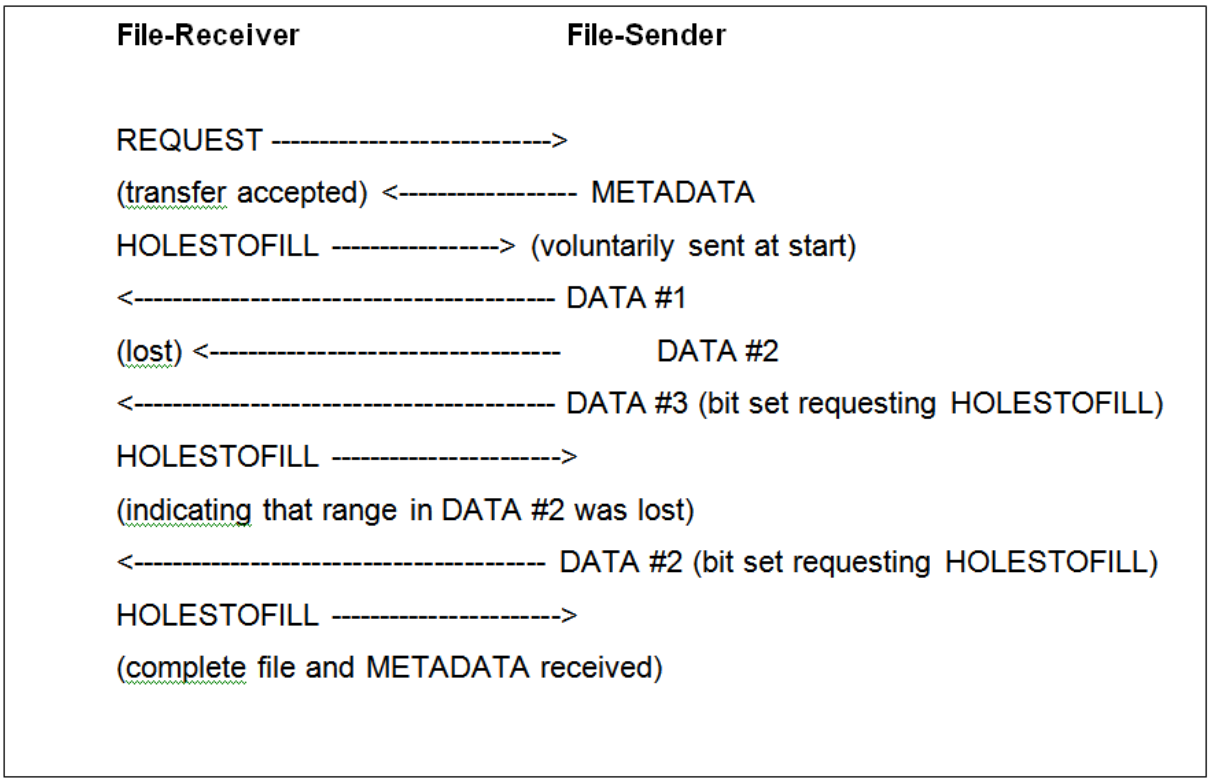

Figura 5 - Exemplo de comunicação no protocolo Saratoga

Nó trecho de comunicação apresentado na Figura 2 podemos observar as requisições e as respostas, bem como o comportamento do protocolo quando um pacote é perdido.

2.6 Protocolos de encaminhamento e roteamento

A escassez de recursos e a conectividade circunstancial nas MANETs faz com que o desenvolvimento de um algoritmo de encaminhamento seja um desafio constante. Além disso, uma estratégia de roteamento inteligente é necessária para utilizar de maneira eficiente os recursos limitados, enquanto se adapta às mudanças nas condições da rede (Abolhasan et al. 2004).

Desde a criação pela DARPA (*Defense Advanced Research Projects Agency*) das redes de transmissão de pacotes por rádio (Jubin e Tornow, 1987)

diversos trabalhos apresentam estudos e propostas de algoritmos e estratégias de roteamento para MANETs.

Os algoritmos podem ser divididos em três grandes classes: proativos, reativos e híbridos.

Os algoritmos proativos mantêm informações de roteamento de pelo menos parte da rede, normalmente armazenadas em uma ou mais tabelas. Essas informações são atualizadas periodicamente e/ou quando ocorrem mudanças na topologia da rede. Essa característica, contudo, pode trazer problemas de uso elevado de memória, de processamento e de comunicação, dependendo das características do algoritmo e da rede.

Os algoritmos reativos foram projetados para minimizar o *overhead* de descoberta e manutenção de rotas existente nos algoritmos proativos. Para isso, apenas as informações a respeito das conexões existentes são mantidas. Quando necessária, a descoberta de rotas é feita normalmente pelo processo de inundação, que pode trazer problemas de sobrecarga de mensagens no caso de redes muito grandes.

Algoritmos híbridos são considerados como a nova geração de protocolos de roteamento (Abolhasan *et al*. 2004) e são proativos e reativos em essência. Eles têm como princípio básico a divisão da rede em partes, permitindo assim aumentar a escalabilidade, utilizando métodos proativos para partes da rede mais importantes e reativos para os nós mais distantes ou menos utilizados.

Diversos trabalhos analisam e comparam os algoritmos sob diferentes pontos de vista. Por exemplo, no trabalho de (Zhang, 2006), é apresentada uma vasta comparação entre diversos algoritmos; Parte dessas comparações é mostrada na Tabela 1.

Protocolo	Gerenciamento de buffer	Estimativa de probabilidade de entrega	Utiliza localização ou informações futuras	Complexidade	Ano publicação
Epidemic	Infinito	Não	Não	Simples	2000
SWIM	Infinito	Não	Não	Simples	2003
MRP	Remove o mais recente (envia aleatoriamente)	Não	Não	Simples, integrada com protocolos existentes	2003
Spraying	Infinito	Não	Localização	Simples, rastreamento central	2001
DLE	DOA, DRA, DLE, DLR	Sim	Não	Trocas e computacional	2001
PROPHET	Infinito	Sim	Não	Trocas e computacional	2003
MV	Sim, elimina aqueles que outros nós tem maior probabilidade de entrega	Sim, calcula a probabilidade de entrega do bundle em h saltos para o destino	Não	Trocas e computacional	2005
SEPR	Sim, elimina aqueles que outros nós tem maior probabilidade de entrega	Sim, através de medições	Não	Computacional	2003
CAR	Infinito	Sim, filtro Kilman	Não	Computacional	2005
MBR	Infinito	perfil do usuário é assumido	Sim, ambas	Trocas e computacional, rastreamento central	2001
IBRR	Infinito	Sim, necessida de duas trocas entre nós	Sim, ambas	Trocas e computacional, rastreamento central, mobilidade	2002
Erasure coding	Infinito	Não	Não	Decodificação mais complexa computacionalmente	2005

Tabela 1 - Algoritmos de encaminhamento para MANETs e DTNs

Apesar das características da Tabela 1 mostrar que os diversos algoritmos utilizam variados modos de operação, diversos aspectos que devem ser levados em conta nos algoritmos são comuns a grande parte deles.

Esses aspectos cobrem principalmente as informações relativas ao estado da rede, encontros entre os nós, quantidade de mensagens e estado dos recursos do dispositivo (energia, memória e processamento).

2.7 Problemas pendentes

Diversas soluções existem, ou foram propostas, para a comunicação oportunística. Caracterizada pela presença intermitente de um ou mais meios

de conexão e pela utilização de intermediários para o encaminhamento de dados, esse tipo de rede vem ganhando importância nos últimos anos.

Diversas arquiteturas de comunicação com suporte à comunicação oportunística foram desenvolvidas ao longo dos anos, porém, a maioria delas está intimamente ligada aos requisitos do cenário para o qual foram desenvolvidas. Assim, não estão preparadas para adaptar-se a novas utilizações sem um grande esforço e um número significativo de alterações.

Por outro lado, as arquiteturas que se propõem a serem de uso geral, como o Haggle, esbarram nas dificuldades para suas implementações. Esses investimentos são necessários porque essas arquiteturas alteram muito o modo como os aplicativos interagem com a rede, requerendo a substituição de protocolos de comunicação usuais.

Apesar de as propostas dos grupos de trabalho da IRTF definirem arquiteturas de comunicação avançadas e trazerem diversos avanços na solução de alguns problemas da área, o seu desenvolvimento não contempla alguns aspectos necessários para a sua utilização no ambiente de MANETs.

Entre esses aspectos, podemos citar a necessidade de codificação diferente da usual, uma estrutura de identificação baseada em conjuntos de dispositivos.

Além disso, tanto a arquitetura DTN quanto o protocolo *Bundle* ainda apresentam problemas de confiabilidade, de detecção de erros, de segurança e de desempenho (Wood *et al*, 2009).

Apesar de ainda não existir uma solução para MANETs capaz de se adaptar a diversos contextos, diversos trabalhos demonstram a utilidade e a viabilidade de tal solução, conforme demonstrado em (Chaintreau *et al*. 2007), onde um conjunto de análises matemáticas demonstra sobre quais circunstâncias uma determinada classe de algoritmos de encaminhamento pode garantir que um pacote será entregue num tempo finito. Apesar dessas análises, a implementação de um sistema que consiga cumprir esses requisitos se mostra irreal na atualidade.

Embora a transferência de arquivos seja a principal funcionalidade prevista pelos sistemas atuais, ela pode servir de base para serviços mais complexos, como redes de monitoramento pessoal e interconexão de

dispositivos, entre outros. Esses serviços necessitam de protocolos bem definidos e meios homogêneos de transmissão e administração para que possam operar de forma satisfatória.

Nesse contexto, o desenvolvimento de uma arquitetura que supra essas necessidades, enquanto mantém a compatibilidade com a atual arquitetura da Internet se torna necessário. A proposta para uma arquitetura dessa classe é apresentada na seção seguinte.

2.8 Pontos de interesse

Após a análise dos projetos da área e dos problemas que eles apresentam quando vistos como soluções de uso geral, a necessidade do desenvolvimento de uma arquitetura de comunicação modular, capaz de operar em diversos cenários e com a capacidade de se valer da comunicação oportunística para aumentar suas funcionalidades se tornou clara.

Para permitir o desenvolvimento e a implementação desejados, os diversos pontos que deveriam ser tratados pela arquitetura foram separados e descritos. Essa separação ocorre apenas para efeitos de documentação, uma vez que no código, diversos pontos (a transferência de dados e a contabilização, por exemplo) estão diretamente relacionados.

A lista abaixo, apesar de não ser exaustiva (outros itens sempre podem ser inseridos, dependendo do foco que se tem), é bastante completa, cobrindo os principais pontos da área. Ela não visa definir como certo algum dos procedimentos escolhidos para esse trabalho, apenas citar e lembrar dos pontos que podem influenciar qualquer trabalho na área.

2.8.1 Identificação dos nós

Para o correto funcionamento de praticamente qualquer sistema de comunicação, a capacidade de identificar unicamente cada um dos nós participantes é necessária. Essa identificação pode ocorrer de diversos modos,

como através do nome do usuário, do endereço físico da interface de rede, de chaves pública/privada, etc.

Prover suporte para esses diversos tipos de identificação, bem como a métodos de validação é uma funcionalidade que precisa ser oferecida pela arquitetura.

2.8.2 Contabilização

Em uma arquitetura oportunística os nós participam como consumidores e provedores de serviços para outros nós. Por esse motivo, alguma forma de compensação pelos serviços prestados pode estar envolvida. Recompensas financeiras, reputação e vantagens em usos futuros da rede estão entre as compensações mais comuns.

Para esse fim, é necessário que a arquitetura proveja algum mecanismo de contabilização dos dados trafegados por cada participante. Preferencialmente, esse mecanismo deve permitir a verificação dos dados, evitando possíveis fraudes.

2.8.3 Transferência de dados

Essa, que é a funcionalidade básica provida pela maioria das arquiteturas de comunicação em rede, se mostra uma das mais complicadas no caso de uma arquitetura genérica para redes tolerantes a atrasos. O problema reside no fato de diferentes dispositivos terem possivelmente diferentes sistemas de arquivos e programas para manipular os dados transferidos.

Outro ponto a ser tratado é a verificação dos dados. Dada a possibilidade de interferência ou alteração dos dados durante a sua transmissão, a comprovação da integridade dos dados deve ser provida pela arquitetura.

2.8.4 Troca de dados sobre os nós

A maior parte das arquiteturas de comunicação prevê que, a cada encontro entre dois nós participantes, um conjunto de dados sobre a situação atual da rede seja trocado. Tipicamente, esses dados contém informações sobre encontros com outros nós e sobre os pacotes (*bundles*) que o nó carrega e precisa encaminhar. Essas informações normalmente são utilizadas pelo protocolo de encaminhamento.

As informações sobre os encontros com outros nós da rede usualmente são utilizadas pelo protocolo de encaminhamento para tentar escolher as possíveis rotas que cada pacote que ele carrega irá seguir. Uma vez que a topologia da rede se altera constantemente, assim como a quantidade de participantes, essas informações são de extrema importância.

A troca de informações sobre os pacotes que cada nó carrega serve para que os nós decidam quais pacotes serão trocados entre eles. Informações sobre pacotes entregues no passado também podem ser trocadas, permitindo assim calcular a confiabilidade de um nó, aumentando as chances do protocolo escolher um caminho útil.

2.8.5 Segurança

Aspectos de segurança precisam ser tratados nos diversos pontos da arquitetura de comunicação. Parte desses aspectos já foi discutida anteriormente nesse documento, como na identificação dos nós. Outro ponto que merece atenção especial na segurança são os arquivos, ou objetos / *bundles* encaminhados. Eles precisam ser observados sobre três pontos de vista: o emissor do arquivo, os nós intermediários e o receptor do mesmo.

Para o emissor, existem duas grandes preocupações de segurança. A primeira é uma possível violação da privacidade daquele arquivo. Uma vez que os dados a serem trafegados podem ser de cunho sigiloso, é necessário garantir que apenas os usuários designados terão acesso ao conteúdo dos arquivos trafegados pela rede. A segunda é a possibilidade de algum nó

intermediário alterar o conteúdo do pacote de dados, causando assim possíveis problemas de integridade dos dados.

No segundo ponto de vista, os nós intermediários precisam basicamente preocupar-se apenas com a sua própria segurança. Como estão aceitando armazenar dados de terceiros, é necessário garantir que esses dados não possam prejudicar o sistema do usuário, seja com a presença de um vírus ou qualquer código malicioso.

As preocupações do receptor são muito parecidas com a do emissor. Tipicamente, ele necessita de algo que garanta que o pacote entregue a ele é exatamente o que foi criado pelo emissor e que o emissor é realmente aquele que está indicado.

Para garantir todos esses aspectos de segurança, uma infraestrutura de chave publica/privada é a opção natural. Apesar de solucionar os problemas acima indicados, é necessário verificar se essa solução é viável no cenário em que a arquitetura irá ser executada. Por esse motivo, suportar outros mecanismos de identificação de emissores e privacidade e integridade dos dados é necessário.

O uso certificados digitais assinados por uma Autoridade Certificadora (AC) confiável pelos nós comunicantes permite a realização de assinaturas digitais e encriptações dos objetos encaminhados. *Hashs* dos objetos gerados com as chaves privadas dos emissores permitem a verificação da autenticação dos objetos recebidos, usando as chaves públicas correspondentes. Envelopes digitais permitem o uso combinado de criptografia simétrica e assimétrica na codificação dos objetos transmitidos, provendo a confidencialidade de seus conteúdos, de forma que sejam acessíveis apenas ao emissor e receptor(es) apropriado(s).

2.8.6 Algoritmo de encaminhamento

O algoritmo de encaminhamento é possivelmente o ponto onde as diferenças entre as diferentes arquiteturas são mais visíveis. Um ponto de extrema importância para a funcionalidade dessa arquitetura de serviços é permitir que

diferentes algoritmos possam ser corretamente implementados, facilitando o aceso desses às informações que a arquitetura dispõe e permitindo a implementação de diferentes políticas

2.8.7 Algoritmo de alocação de espaço

Dispositivos móveis compõem o ambiente de execução mais comum para a comunicação oportunística. Sabendo que esses dispositivos sofrem de limitações de espaço de armazenamento, um bom algoritmo de gerenciamento de memória se faz necessário.

É função desse algoritmo decidir quais pacotes de dados (*bundles*) serão mantidos. As escolhas sobre quais pacotes eliminar e quando executar essa ação têm efeito direto sobre o funcionamento da rede. Em muitos casos, essa função é realizada pelo protocolo de encaminhamento.

2.8.8 Aviso sobre encaminhamento

Muitas redes utilizam mecanismos de aviso de entrega. Usualmente, esse mecanismo é uma mensagem contendo o identificador da mensagem que foi entregue e dados sobre o caminho percorrido.

Muitas vezes essas mensagens são enviadas por nós intermediários, permitindo um controle da posição de um pacote e o rastreio do seu encaminhamento.

Além da funcionalidade de avisar o emissor que os dados foram entregues corretamente ao destinatário, esse mecanismo permite que outras cópias do pacote que por ventura existam na rede sejam eliminadas. O caminho percorrido por um *bundle*, contudo, nem sempre pode ser refeito já que as interações entre os nós são circunstanciais.

2.8.9 Informações de contexto

Com o aumento do poder computacional e o incremento no número de funcionalidades dos dispositivos móveis, a quantidade e a qualidade das

informações sobre o contexto em que o usuário se encontra aumentou significativamente nos últimos anos.

Apesar de poucas arquiteturas de comunicação utilizarem de maneira significativa esse tipo de informação, fornecer suporte para o seu armazenamento e utilização é uma característica importante para serviços futuros.

Dentre as informações de contexto, a localização vem ganhando destaque com o aumento dos serviços baseados em localização (*Location-Based Services*). LBS é um conceito que denota aplicações que integram a localização geográfica com a noção geral de serviços (Schiller e Voisard, 2004). Entre os exemplos estão os sistema de navegação para carros, planejamento de viagens e localização de serviços de emergência ou de pontos de interesse do usuário.

Capítulo 3

UMA ARQUITETURA DE COMUNICAÇÃO OPORTUNÍSTICA

3.1 Projeto

A necessidade de suportar a comunicação entre dispositivos móveis em redes *ad hoc* com conectividade intermitente motivou o desenvolvimento da arquitetura de comunicação apresentada neste trabalho. Para tanto, a arquitetura fornece um ambiente seguro para identificação, transmissão e contabilização de objetos, denominados pacotes ou *bundles*.

O termo pacote, nesse caso, não está relacionado à estrutura de transmissão (PDU) típica da camada de rede na arquitetura TCP/IP. Os pacotes transmitidos por meio dessa arquitetura correspondem a arquivos associados a mensagens ou a diferentes tipos de dados, além das informações de controle necessárias. Transmissões ocorrem visando à entrega de pacotes diretamente aos seus destinatários, a intermediários potenciais para a entrega direta desses pacotes, ou ainda a servidores de *e-mail* responsáveis pelos domínios associados à identificação dos usuários.

3.2 Apresentação da proposta

Desenvolvida como uma arquitetura de serviços para redes oportunísticas, a arquitetura de comunicação aqui proposta apresenta os serviços necessários para o funcionamento de diversos sistemas de comunicação.

Implementada num sistema multicamada, diversos aspectos do seu funcionamento podem ser adaptados sem que sejam necessárias alterações nas demais funcionalidades. Por exemplo, o algoritmo de encaminhamento pode ser alterado sem que sejam necessárias outras alterações. Esse tipo de alteração melhor descrita na Seção 3.5.3.

A arquitetura desenvolvida provê um mecanismo de identificação e comunicação entre os nós participantes. Essa comunicação é utilizada para trocar conjuntos de dados (*bundles*) dos usuários sem a intervenção direta dos mesmos. Assim, um bundle enviado pelo usuário X trafega até o destinatário Y passando por zero ou mais nós intermediários.

Essa estrutura de comunicação é utilizada por um ou mais serviços, que são os “donos” dos conjuntos de dados, operando de forma análoga aos aplicativos (cliente de e-mail, programa de mensagens instantâneas, etc.) na infraestrutura de rede atual.

3.3 Olympia

A arquitetura de comunicação apresentada nesse trabalho é denominada Olympia1. Ela foi desenvolvida com o objetivo de servir de base em sistemas de comunicação em ambientes de dispositivos móveis, com suporte à comunicação oportunística.

1 O nome é uma homenagem ao pintor Édouard Manet e seu quadro Olympia.

Com base na análise dos trabalhos relacionados, durante o desenvolvimento desse trabalho buscou-se agrupar as questões de projeto envolvidas, a fim de facilitar o entendimento e a organização do sistema gerado.

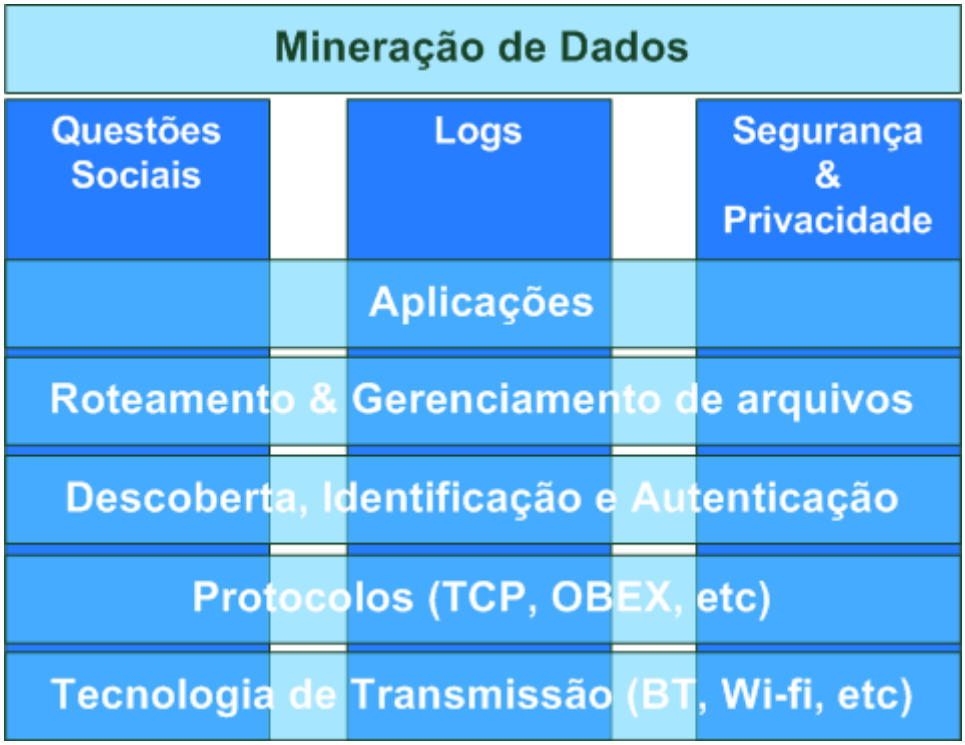

Figura 6 - Organização dos elementos a serem trabalhados

A Figura 6 apresenta a disposição dos elementos da arquitetura como foram organizados. Cinco conjuntos (camadas) tratam de questões que se "sobrepõem" durante o uso de uma arquitetura de comunicação oportunística. Assim, os protocolos de comunicação operam sobre uma determinada tecnologia de transmissão, e os algoritmos de Descoberta, Identificação e Autenticação só podem operar se houver um meio (meio físico + protocolos) que permita a comunicação com outros dispositivos. Isso ocorre sucessivamente, com cada camada operando com base nas anteriores.

Três temas são considerados transversais, atingindo todos os elementos da arquitetura.

O primeiro é Segurança e Privacidade, que precisa ser tratado nos diversos níveis para que os usuários e os dados trafegados não sejam expostos. A maior parte das características de segurança é provida por algum

tipo de criptografia, porém soluções diferentes podem ser utilizadas em determinadas camadas.

O segundo tema é a questão dos registros (*logs*), que precisam ser manipulados. Registrar as diversas operações e eventos permite ter um melhor conhecimento do ambiente no qual a arquitetura está operando, favorecendo um melhor acerto de configurações. Além disso, como normalmente as comunicações ocorrem num modelo em que os usuários podem operar como retransmissores, questões de contabilização (e eventuais recompensas) podem estar envolvidas.

O último tema transversal está relacionado à análise das diversas questões sociais que envolvem o funcionamento de sistemas oportunísticos. Descobrir os círculos de contato de cada usuário e como eles interagem entre si tem um papel fundamental na melhoria dos algoritmos de encaminhamento, além de permitir o desenvolvimento de novas aplicações e serviços.

Nesse tema são tratadas também muitas questões de inferência de contexto, uma vez que essas refletem diretamente na dinâmica social em que os usuários estão envolvidos. Por exemplo, quando um conjunto de colegas de trabalho está jogando futebol após o expediente, as relações podem ser diferentes do que durante o horário de trabalho.

Utilizando todas essas informações estão as questões relacionadas à mineração de dados, que fornece subsídios para que o sistema possa ser ajustado.

3.4 Funcionalidades

A arquitetura é composta por duas partes: a infraestrutura de comunicação, que dá suporte à comunicação oportunística, e o conjunto de serviços que operam sobre essa infraestrutura.

A infraestrutura é responsável por identificar e manipular os dados (*bundles*) que estão no dispositivo, pela a busca e identificação de outros nós, pela manutenção de dados com informações sobre os outros nós, pela

transferência dos *bundles,* além de operações de manutenção, como o gerenciamento de espaço de armazenamento no dispositivo. Além disso, é função da infraestrutura realizar as conexões e transferências de dados com outros nós.

Os *bundles* contêm as informações necessárias para o encaminhamento e o tratamento das informações nele contidas, além da sua carga útil (*payload*). O *payload* do *bundle* contém os dados que um serviço deseja enviar para o destinatário. Por exemplo, no caso do serviço de mensagens (descrito na seção de estudos de caso) o *payload* é utilizado para armazenar a mensagem que deve ser entregue ao destinatário. No caso de um serviço de compartilhamento de fotos, o *payload* seria usado para armazenar as fotos em questão.

As informações que compõem o *bundle* e a sua organização são descritas na seção 3.5.2.

Para realizar essas operações, um conjunto de funções e algoritmos está disponível, sendo que as operações foram isoladas, permitindo, por exemplo, que o algoritmo de remoção de *bundles* seja alterado sem que isso impacte em outras operações.

Além disso, a infraestrutura de comunicação é capaz de se comunicar com pontos de conexão fixos, permitindo o envio e o recebimento de informações através de uma rede local ou da Internet.

Os serviços operam sobre a arquitetura de comunicação, utilizando-a para atingir outros nós ou o destino das informações encaminhadas. Cada serviço pode utilizar diferentes conjuntos de algoritmos, ajustando-os aos seus objetivos. Diversos serviços foram previstos, tais como a transmissão de mensagens, o compartilhamento de arquivos, o envio de mensagens ao Twitter [2]e o envio de e-mails entre outros.

Para a prova de conceito e avaliação da arquitetura, foram desenvolvidos e implementados dois desses serviços: troca de mensagens e

[2] Twitter é um serviço de microblog que permite a divulgação de mensagens curtas. Disponível em www.twitter.com.

envio de mensagens ao *Twitter.* No Capítulo 5 os dois estudos de caso são demonstrados.

3.5 Estrutura da Arquitetura

Disposta em camadas, a arquitetura de comunicação desenvolvida é organizada da seguinte forma:

Figura 7 - As camadas da arquitetura de comunicação

As cinco camadas da arquitetura de comunicação apresentada estão dispostas de forma que quatro delas se comunicam apenas com as camadas imediatamente acima e abaixo delas, como pode ser observado na Figura 7. A camada de contabilização é capaz de se comunicar com todas as outras diretamente.

A seguir é apresentada uma breve explicação sobre cada uma das camadas, juntamente com alguns exemplos de código dessas camadas.

3.5.1 Camada de Aplicação

A camada de aplicação é responsável por prover interfaces que permitem às aplicações de rede enviar e receber pacotes utilizando a arquitetura.

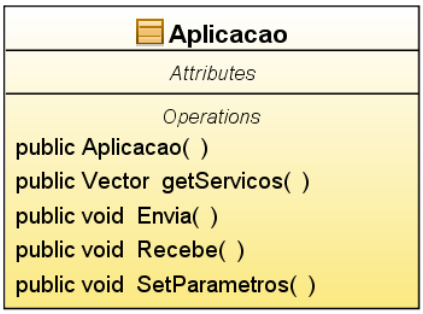

Figura 8 - Diagrama representativo de parte das funcionalidades da Camada de Aplicação

Nessa camada, os usuários têm acesso a métodos para determinar os serviços disponíveis, enviar e receber dados (bundles) através desses serviços e fazer alterações nas configurações da arquitetura.

Por exemplo, quando o serviço de mensagens está disponível, é possível enviar uma mensagem a algum usuário, checar as mensagens que foram recebidas e fazer alterações nos parâmetros do algoritmo de encaminhamento (aumentando o número de pacotes carregados para outros destinatários, por exemplo).

3.5.2 Camada de Gerenciamento de Objetos

Toda vez que um pacote é enviado por uma aplicação de rede utilizando as interfaces fornecidas pela camada de aplicação, a camada de gerenciamento de objetos é acionada para armazenar de forma persistente o pacote, a fim de que ele possa ficar registrado no dispositivo emissor até o momento em que uma conexão seja estabelecida com outro nó (o destinatário ou um nó intermediário) para o qual o pacote poderá ser encaminhado.

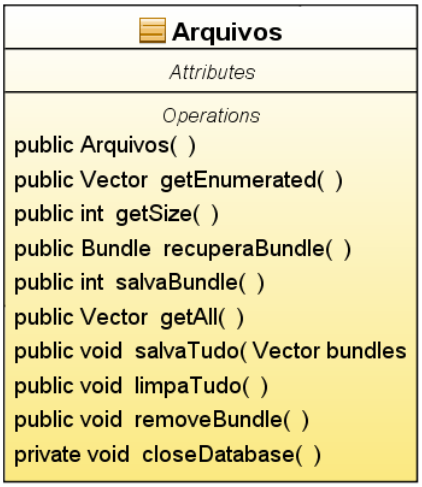

Figura 9 - Diagrama representativo de parte das funcionalidades da Camada de Arquivos

Como pode ser visto na Figura 9, a Camada de Arquivos contém os métodos utilizados na manipulação dos bundles e nos seus armazenamentos em memória secundária.

Quando uma conexão é estabelecida com algum dispositivo, a camada de gerenciamento é acionada para fornecer às camadas inferiores todos os pacotes que devem ser encaminhados. Ao receber uma requisição, esta camada carrega em memória principal informações sobre todos os pacotes armazenados e as passa para a camada de encaminhamento, a qual decidirá de que forma cada pacote será enviado.

Se uma conexão é estabelecida para o recebimento de pacotes vindos de um dispositivo, então a camada de gerenciamento é acionada para receber cada pacote transmitido e armazená-lo de maneira adequada, considerando sempre o seu destinatário.

Caso o pacote recebido seja destinado ao usuário do dispositivo que acabou de recebê-lo, então ele será armazenado em uma área destinada a pacotes do usuário, os quais poderão ser passados para a camada de aplicação quando esta fizer uso da interface de recebimento. Se o destinatário for outro usuário, então o pacote será armazenado em outra área destinada a pacotes que serão encaminhados posteriormente.

Pacotes destinados a outros usuários poderiam ser violados (lidos ou alterados) se fossem armazenados sem nenhum mecanismo de criptografia nos nós intermediários da rede. Por essa razão, a arquitetura apresentada

prevê o uso de uma infraestrutura de segurança para garantir a integridade e a confiabilidade dos dados transmitidos.

Por fim, considerando-se que é necessário gerenciar o tráfego de pacotes na rede, a arquitetura prevê o uso de um cabeçalho de informações associado a cada pacote, de modo que cada nó da rede possa saber exatamente o que fazer com os pacotes que possui.

A fim de fornecer uma estrutura bem definida para o gerenciamento dos pacotes, foi adotada a linguagem XML para a criação dos cabeçalhos dos pacotes. Os cabeçalhos XML têm os mesmo dados que o bundle, com exceção das informações sobre encaminhamentos anteriores (PATH), que são exclusivas do cabeçalho. Essa igualdade nos dados permite que as decisões que envolvem o bundle possam sem tomadas sem a necessidade de leitura do bundle completo.

A seguir temos a representação do XML de cabeçalho de um pacote e a descrição de cada um de seus campos.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com" elementFormDefault="qualified">
  <xs:complexType name="CABECALHO">
    <xs:sequence>
      <xs:element name="ID" type:"xs:string"/>
      <xs:element name="RMSID" type:"xs:integer"/>
      <xs:element name="SERVICO" type:"xs:string"/>
      <xs:element name="TAMANHO" type:"xs:integer"/>
      <xs:element name="CRIACAO" type:"xs:dateTime"/>
      <xs:element name="CRIADOR" type:"xs:string"/>
      <xs:element name="DESTINO" type:"xs:string"/>
      <xs:complexType name="PATH" type:"xs:string"/>
        <xs:element ref="HOP" minOccurs="1" maxOccurs="unbounded/>
      </xs:complexType>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="HOP">
    <xs:sequence>
      <xs:element name="NO" type:"xs:string"/>
      <xs:element name="TIMESTAMP" type:"xs:dateTime"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Figura 10 - *Schema* do cabeçalho de um pacote da arquitetura

Como é possível observar na Figura 10, o cabeçalho é composto de sete campos. O primeiro deles é o campo **ID**, que corresponde a um identificador do pacote transmitido que será utilizado para o tratamento de pacotes duplicados no destinatário. A identificação é feita tipicamente através de um número gerado aleatoriamente, concatenado com a identificação do nó emissor. A chegada de pacotes duplicados pode ocorrer em situações em que

a camada de encaminhamento da arquitetura considera mais prudente encaminhar um pacote através de mais de um nó da rede.

O campo **RMSID** indica a posição do *bundle* no armazenamento em memória secundária do dispositivo. Essa informação permite que um determinado *bundle* seja carregado em memória principal sem que seja necessário realizar uma busca ou o carregamento de outros *bundles*. Esse campo deve ser ajustado em cada armazenamento de um *bundle* num nó intermediário.

O campo **SERVICO** indica através de um código qual o serviço que é responsável pelo processamento deste *bundle.* Assim, cada *bundle* pode ser entregue a seu serviço responsável, que saberá como tratar as informações contidas no mesmo.

O campo **TAMANHO** contém o tamanho do *payload* em bytes, permitindo que a arquitetura saiba de antemão o tamanho do *bundle* que terá de tratar, levando essa informação em consideração na hora de decidir encaminhar ou aceitar um pacote.

O campo **CRIACAO** é utilizado para armazenar a data e a hora de criação de um pacote, podendo ser usado posteriormente em políticas de descarte e de análise do tempo de entrega.

Os campos **CRIADOR** e **DESTINO** são utilizados para identificar o emissor e o receptor do pacote. Essa identificação pode ser de diversos tipos, desde um nome em alto nível (“Danilo”, “Fulano”, etc.) ou conjuntos de dados que identifiquem o usuário, como o *hashs* da respectiva chave privada.

O campo **PATH** armazena uma lista de identificadores dos nós pelos quais um pacote já passou (**HOP**). Essa informação pode ser utilizada pelo protocolo de encaminhamento para decidir para quais nós o pacote será encaminhado a seguir, evitando eventuais laços de transmissão. Além disso, com base na lista de nós participantes do encaminhamento é possível aplicar juntamente com a camada de contabilização uma política de recompensa aos participantes. *Hashs* gerados por cada nó intermediário podem ser usados para garantir a autenticidade das informações de encaminhamento e contabilização.

Cada campo **HOP** existente no campo **PATH** do cabeçalho é composto de um campo **NO** (a identificação associada ao nó pelo qual o pacote passou)

e de um campo **TIMESTAMP**, que armazena a data e a hora na qual o pacote foi recebido pelo nó. Tais informações fornecem para a camada de encaminhamento uma estimativa de quanto tempo um pacote permaneceu em um nó, permitindo que esta ajuste suas políticas de encaminhamento a fim de garantir entregas mais rápidas para os destinatários.

3.5.3 Camada de Encaminhamento

A camada de encaminhamento é a responsável por decidir com qual dos nós vizinhos o dispositivo se conectará e quais pacotes serão transmitidos para um nó através de uma conexão previamente estabelecida. Após o estabelecimento de uma conexão pela camada de descoberta, é fornecido para a camada de encaminhamento um objeto de conexão.

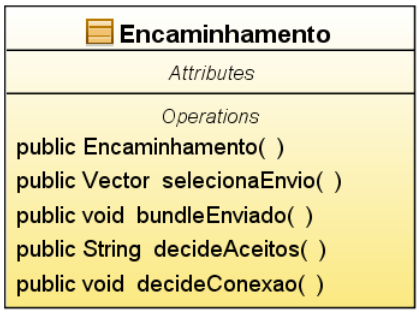

Figura 11 - Diagrama representativo de parte das funcionalidades da Camada de Encaminhamento

A Figura 11 demonstra parte dos métodos da Camada de Encaminhamento, responsável por tomar as decisões sobre a comunicação com outros nós.

O objeto de conexão contém informações sobre o tipo de conexão estabelecida e interfaces de comunicação genéricas que permitem à camada de encaminhamento tratar o envio e o recebimento de pacotes de forma independente da tecnologia usada nas transmissões. Além dessas informações sobre a conexão, este objeto contém também informações sobre um nó vizinho, como uma identificação. Todas essas informações podem ser utilizadas pelos algoritmos de encaminhamento para decidir, por exemplo, de acordo com o nó conectado quais *bundles* serão encaminhados.

Dessa forma, a camada necessita tratar apenas das decisões de encaminhamento, determinando qual o melhor caminho que um pacote deve seguir.

As decisões de encaminhamento podem utilizar três modos de operação: encaminhamento ***ad hoc***, encaminhamento **Baseado em Informações de Contexto** e encaminhamento **por E-mail**.

No primeiro modo de operação (*ad hoc*), um *bundle* transita de um nó para o outro através de uma conexão ponto a ponto, não fazendo uso de infraestrutura de rede para a transmissão. Como a conexão ocorre diretamente entre os nós, existe uma menor probabilidade de interferências externas no tráfego (*firewalls*, congestionamentos, etc.), permitindo que a conexão seja mais simples e rápida, mesmo com a relativa baixa capacidade de transmissão de algumas tecnologias. Algumas tecnologias que podem ser utilizadas com essa política de encaminhamento são a Bluetooth e a Wi-Fi, uma vez que ambas permitem conexões ad hoc e estão amplamente difundidas entre os dispositivos móveis atuais.

O modo de operação baseado em informações de contexto utiliza informações coletadas sobre os nós e os pontos de acesso utilizados por eles para determinar qual o melhor caminho a ser seguido por um *bundle*. Tais informações englobam dados como a localização de um dispositivo e frequência de encontros.

Sabendo-se, por exemplo, que um dispositivo se encontra em um determinado local que pode ser alcançado por meio de uma rede infraestruturada, a camada de encaminhamento pode encaminhar um *bundle* para seu destinatário através da rede. Além disso, caso se saiba de antemão que um determinado dispositivo transita em um local em uma determinada hora do dia, a camada de encaminhamento pode transmitir o *bundle* pela rede infraestruturada até um ponto de acesso naquele local. Lá, o *bundle* ficará armazenado até o momento de sua entrega.

No último modo de operação, a entrega é feita utilizando o mecanismo tradicional de envio de e-mail. Um usuário, quando conectado à uma infraestrutura de rede, é capaz de encaminhar *bundles* para os servidores de e-mail dos destinatários. Por sua vez, o destinatário, ao fazer o download de suas

mensagens, recebe o *bundle*. Essa política deve, preferencialmente, ser utilizada apenas em casos específicos, uma vez que requer que tanto o emissor quanto o destinatário tenham acesso eventual à infraestrutura de rede.

Algumas restrições de autenticação pré-envio também podem ser necessárias nessa forma de entrega de *bundles.* Muitas infraestruturas de rede não permitem o envio direto de e-mail pelos nós. Além disso, servidores de e-mail comumente não autorizam o recebimento de mensagens enviadas por nós que não são os responsáveis diretos pelo e-mail do domínio DNS usado nas transmissões.

Os protocolos de encaminhamento também estão alocados na camada de encaminhamento, podendo fazer uso das três políticas acima descritas. Para funcionarem de forma eficiente, estes protocolos podem usar informações de endereço dos dispositivos do usuário e de dados estatísticos sobre o encaminhamento de pacotes anteriores.

Para cada modo de operação existem diversos algoritmos disponíveis. É possível utilizar versões disponíveis na literatura (Direct Deliver (Spyropoulos *et al,* 2004), Epidemic (Mitchener e Vadhat, 2000) e PRoPHET (Lindgren et al, 2006), entre outros) ou utilizar as informações disponíveis para criar um algoritmo específico. Como a implementação desses algoritmos é isolada tanto dos *bundles* como da camada de transmissão, é possível que diferentes algoritmos sejam utilizados, de acordo com variáveis como taxa de ocupação da memória, a capacidade atual da bateria ou o nó com o qual se está conectado.

Dessa maneira, um algoritmo pode, por exemplo, calcular a probabilidade de entrega de um dado *bundle* através de um determinado dispositivo. Com base nesses cálculos, o protocolo pode decidir alterar o modo de encaminhamento, passando a utilizar, por exemplo, a entrega por *e-mail*, caso a probabilidade de entrega pela política de encaminhamento *ad hoc* não seja aceitável.

3.5.4 Camada de Descoberta

A camada de descoberta é responsável por gerenciar todas as conexões estabelecidas com os nós da rede. Para estabelecer conexões, a camada realiza periodicamente buscas por outros nós, utilizando as diversas interfaces de rede disponíveis no dispositivo. Essa tarefa é realizada utilizando-se mecanismos de difusão disponíveis nas tecnologias de transmissão (ex: *broadcast* em redes IP e *Inquiry* em redes Bluetooth) para localizar os dispositivos que estão próximos. Após a localização, uma conexão é estabelecida com o nó escolhido pela Camada de Descoberta.

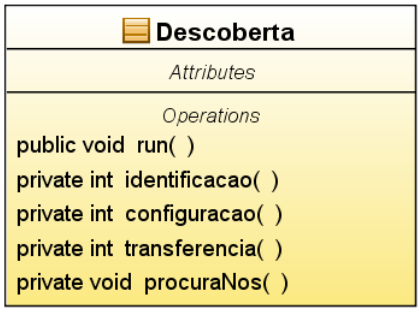

Figura 12 - Diagrama representativo de parte das funcionalidades da Camada de Descoberta

Uma vez estabelecida uma conexão, é gerado um objeto de conexão que será utilizado pela camada de encaminhamento. Esse objeto contém interfaces genéricas, que ocultam as especificidades de cada tecnologia de transmissão. Isso permite que, através de métodos únicos de envio e recebimento, seja possível transmitir *bundles* entre os nós da rede. Tal abstração se mostra importante se considerarmos que, ao utilizar tecnologias como a Bluetooth, pode ser necessário gerenciar manualmente o tamanho máximo de um quadro transmitido, o que obrigaria a aplicação a escolher com antecedência a tecnologia de transmissão e adequar o envio dos *bundles* às limitações da tecnologia.

Após localizar os nós disponíveis no raio de alcance, a Camada de Descoberta usa os métodos da Camada de Encaminhamento para escolher com qual nó uma conexão será estabelecida. Nesse momento, um protocolo de identificação e configuração deve ser usado para garantir a confiabilidade do nó e estabelecer os parâmetros da sessão que se inicia.

3.5.5 Camada de Contabilização

Uma vez que na arquitetura os nós participam como consumidores e provedores de serviço, alguma forma de compensação pelos serviços prestados pode ser utilizada. Recompensas financeiras, reputação e vantagens em usos futuros da rede estão entre as compensações mais comuns.

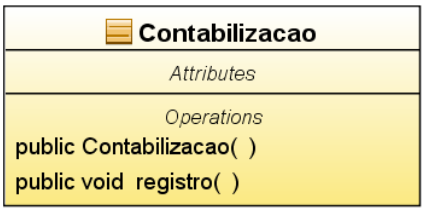

Figura 13 - Diagrama representativo de parte das funcionalidades da Camada de Contabilização

Para esse fim, é necessário que a arquitetura proveja um mecanismo de contabilização dos dados trafegados por cada participante. Além disso, os dados coletados podem ser utilizados pelos algoritmos de encaminhamento para melhorar a sua eficiência, uma vez que fornecem subsídios para determinar quais nós estão mais dispostos a entregar pacotes.

Informações incluídas nas estruturas de transmissão associadas aos *bundles* e mantidas localmente em cada nó podem ser utilizadas para esse fim. Mecanismos de coleta e validação dessas informações precisam ser estabelecidos, como o proposto em (Cruz, 2009).

3.5.6 Outras questões de projeto

Além da especificação das camadas, outras decisões de projeto têm influência direta sobre as funcionalidades e viabilidade de uma arquitetura de comunicação oportunística.

Gerenciamento de armazenamento

Como a arquitetura define que os *bundles* devem ser armazenados pelos nós intermediários até que estes tenham a possibilidade de entregá-los a outro nós, é necessário prover mecanismos que gerenciem esse armazenamento.

Levando-se em consideração as limitações dos dispositivos móveis e a simplicidade de seus sistemas de arquivos, um sistema de gerenciamento de

arquivos foi criado. Nesse sistema, para cada *bundle* é criado um arquivo XML de descrição que contém informações sobre os dados e sobre o trajeto percorrido pelo *bundle* até o momento (conforme descrito na seção 3.5.2).

Todas as informações dos *bundles* que são significativas para os protocolos da arquitetura ficam armazenadas nesses arquivos XML, os quais são gravados separadamente em memória secundária. Tal prática possibilita menor utilização da memória principal do dispositivo durante as tomadas de decisão na arquitetura (envio ou recebimento de um *bundle*, por exemplo), uma vez que o *bundle* é carregado em memória somente quando for utilizado.

Toda vez que alguma informação sobre um *bundle* é necessária, a arquitetura faz o carregamento desses arquivos de configuração ao invés dos *bundles*, trazendo economia de tempo, processamento e energia.

Segurança

A formação de redes entre nós “ao acaso” e o fato dos dados dos usuários trafegarem por esses nós traz preocupações sobre questões de segurança. Essas preocupações podem ser agrupadas em dois pontos principais: A segurança dos bundles e dos dados trafegados e a garantia na identificação dos nós.

A segurança dos *bundles* é um ponto crítico, pois a arquitetura deve garantir que sejam entregues aos destinatários de forma íntegra e provendo a confidencialidade de seus conteúdos. Além disso, é necessário que um *bundle* tenha o seu emissor identificado de maneira única, garantindo a sua autenticidade.

Para tratar de tais questões, foi escolhida uma abordagem mista, utilizando criptografia assimétrica e simétrica. Essa escolha se deu por questões de desempenho da estrutura de chaves assimétricas. Prevê-se que os participantes possuam pares de chaves de criptografia assimétrica, sendo que as chaves públicas são mantidas em certificados digitais assinadas por uma autoridade certificadora (AC) única para o conjunto de usuários que interagem. Isso permite que os nós possam validar as identidades dos outros nós com os quais interagem, sem a necessidade de contato com a AC (uma vez que carregam a chave pública da AC). A criptografia assimétrica é usada

para identificar os nós (o *hash* da chave privada é usado como identificação do usuário) e para proteger o *bundle.*

A proteção dos *bundles* é feita com base em uma sequência de passos semelhante à existente no OpenPGP (definido pela RFC 4880). Nesse modelo, os *bundles* são encriptados usando uma “chave de sessão” gerada de maneira aleatória; a chave de sessão é, então, encriptada usando a chave pública do destinatário, repetindo-se esse passo para cada destinatário. A chave encriptada é anexada ao *bundle,* permitindo que o destinatário acesse os dados.

Outra questão relacionada à segurança é a identificação dos nós. Apesar dos dados trafegarem de maneira protegida, garantir a identidade de cada um dos nós participante diminui as chances de um ataque contra a integridade do pacote, uma vez que nós potencialmente perigosos ou não confiáveis podem ser preteridos durante a escolha da rota. Além disso, a identificação é necessária para que a maior parte dos algoritmos de encaminhamento funcione corretamente.

Nesse trabalho, a identificação dos nós é prevista a partir do hash de uma chave privada associada a esse nó.

Capítulo 4

IMPLEMENTAÇÃO DE REFERÊNCIA

4.1 Implementação de Referência

Buscando validar a arquitetura proposta, foi desenvolvida uma implementação de referência de suas funcionalidades utilizando a plataforma JME (*Java Micro Edition*), comumente suportada por dispositivos móveis. Como tecnologia de comunicação foi escolhido o padrão Bluetooth, uma vez que é o mais comum nos aparelhos e tende, juntamente com a Wi-Fi, a continuar a crescer nos próximos anos. A implementação completa da arquitetura é composta de diversas classes que são responsáveis por fazer desde o tratamento de eventos do sistema até a estruturação de cada camada apresentada na seção 3. Algumas classes (cinco delas) estão diretamente ligadas à aplicação que utiliza a arquitetura (cada uma delas é a implementação de uma camada da arquitetura).

O diagrama de classes apresentado na Figura 14 ilustra os relacionamentos das classes da arquitetura.

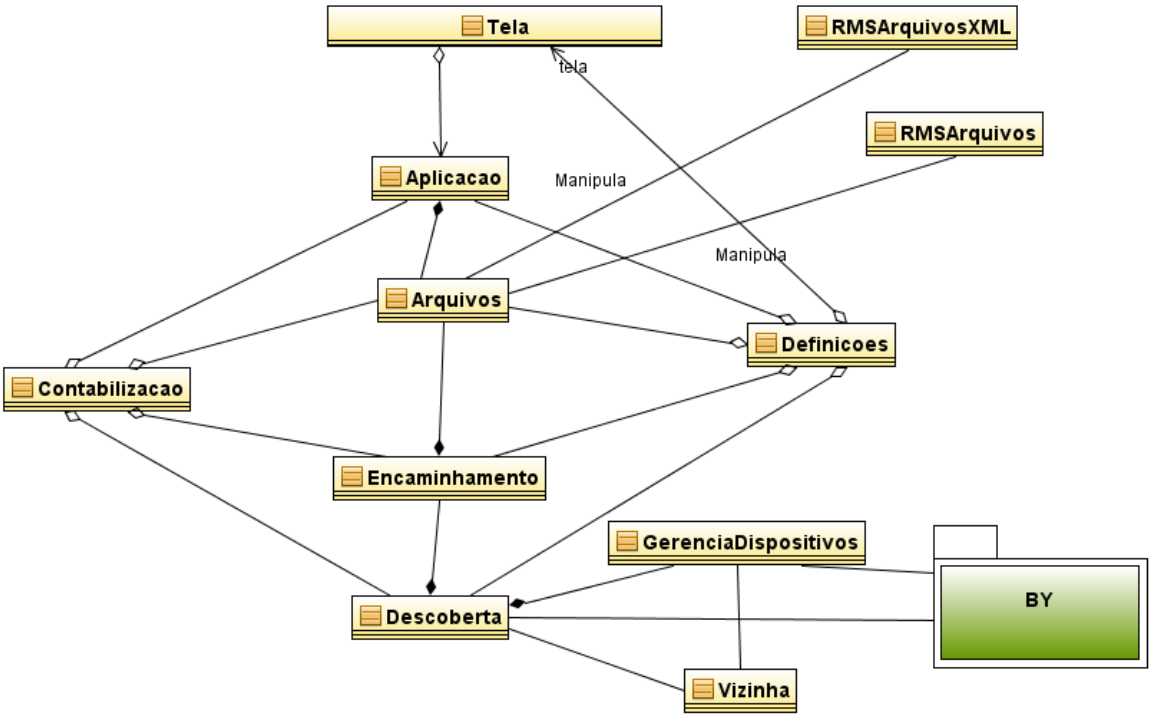

Figura 14 - Diagrama de classes da arquitetura. Classes auxiliares foram omitidas para facilitar a visualização

Para dar suporte à comunicação por Bluetooth, é utilizada uma API de comunicação referenciada no diagrama como BY. Essa API dá suporte à busca e à identificação de dispositivos, bem com possibilita a transferência de dados. As comunicações por Bluetooth podem ocorrer entre dispositivos ou entre dispositivos e pontos de acesso.

Os nós usuários da arquitetura de rede desenvolvida são capazes de se localizar realizando uma busca por um serviço Bluetooth com UUID (*Universally Unique Identifier*) predefinido. Todos os dispositivos na rede apresentam esse serviço ativo.

A classe Descoberta é a implementação da Camada de Descoberta e utiliza classes do pacote BY para realizar as operações de comunicação via Bluetooth. Atualmente, a implementação não conta com suporte a redes Wi-Fi, uma vez que a maior parte dos dispositivos testados não oferece suporte à configuração manual da rede (ficando restritos à configuração via DHCP obtida dos pontos de acesso ou em redes ad hoc). O padrão "Wi-Fi Direct"[3]deverá permitir que os mais diversos dispositivos criem conexões nos moldes dos

[3] Disponível em http://www.wi-fi.org/news_articles.php?f=media_news&news_id=909

atuais pontos de acesso. Com isso, as limitações impostas pelo software dos dispositivos devem ser mais facilmente contornadas.

A classe GerenciaDispositivos é utilizada pela Camada de Descoberta para gerenciar a lista de dispositivos encontrados, as conexões entre eles e demais detalhes de uma conexão.

Todas as configurações de funcionamento da arquitetura ficam registradas na classe Definições, que contém métodos e atributos estáticos acessíveis às camadas, permitindo que eventuais modificações sejam compartilhadas por toda a arquitetura.

Duas classes foram criadas para controlar o acesso aos arquivos de memória secundária dos Bundles e dos XMLs. Esses dados são armazenados na forma de arquivos RMS (*Record Management System*).

Atualmente, os serviços são implantados criando-se entradas nas funções que controlam as camadas. Uma vez que as chamadas são parametrizadas com o código do serviço (um número de identificação único no contexto de utilização), após a inserção do código fonte do serviço basta fazer uma chamada ao serviço através dos métodos da Camada de Aplicação. Uma nova versão com a implementação dos serviços através de classes separadas, com o uso de interfaces, está em desenvolvimento.

4.2 Testes e Resultados

A funcionalidade da arquitetura desenvolvida para o encaminhamento de *bundles* foi avaliada num cenário composto de alguns dispositivos móveis. As Figuras abaixo apresentam as informações do XML correspondente a um *bundle* durante sucessivos encaminhamentos até o seu destino.

```
<Olympia ID="504859462" DESTINO="helio"  TAMANHO="32" RMSID="1" SERVICO="4050">
    <CRIADOR>danilo</CRIADOR>
    <CRIACAO>Wed Jan 20 16:23:46 UTC 2010</CRIACAO>
    <PATH>
        <danilo>1264004626476</danilo>
    </PATH>
</Olympia>
```

Figura 15 - XML de identificação de um Bundle, logo após a criação.

A Figura 15 apresenta o XML com as informações do *bundle* logo após a sua criação. Nele podem ser observadas todas as informações sobre o *bundle* em questão. O elemento “PATH” contém apenas uma entrada, com a data de criação do *bundle* pelo emissor, uma vez que ainda não foi transferido.

Nos elementos dentro de “PATH” (“<Danilo>”, no caso) está a data/hora no formato que o Java retorna (milissegundos desde 1º de janeiro de 1970) quando o *bundle* foi criado.

```
<Olympia ID="504859462" DESTINO="helio"  TAMANHO="32" RMSID="14" SERVICO="4050">
    <CRIADOR>danilo</CRIADOR>
    <CRIACAO>Wed Jan 20 16:23:46 UTC 2010</CRIACAO>
    <PATH>
        <danilo>1264004626476</danilo>
        <ricardo>1264004658810</ricardo>
    </PATH>
</Olympia>
```

Figura 16 - XML de identificação de um Bundle, após o primeiro encaminhamento.

Na Figura 16 podemos ver as alterações que o arquivo de informação do *bundle* sofreu após a primeira transferência. Enquanto os dados de informação do *bundle* (criador, destinatário, tamanho, etc) continuam os mesmos, os dados sobre a localização do *bundle* em memória secundária (campo RMSID) foi alterado para refletir a localização do *bundle* na memória do dispositivo atual.

Além disso, dentro do elemento “PATH” foi adicionado um outro elemento, representando o nó que recebeu o pacote, bem como a data/hora em que isso ocorreu. Nesse caso, o nó que atualmente está com o pacote é identificado por “ricardo”.

```
<Olympia ID="504859462" DESTINO="helio" RMSID="1" SERVICO="4050">
    <CRIADOR>danilo</CRIADOR>
    <CRIACAO>Wed Jan 20 16:23:46 UTC 2010</CRIACAO>
    <PATH>
        <danilo>1264004626476</danilo>
        <ricardo>1264004658810</ricardo>
        <helio>1264004739400</helio>
    </PATH>
</Olympia>
```

Figura 17 - XML de identificação de um Bundle, em seu destinatário final após dois encaminhamentos.

Como pode ser observado na Figura 17, nesse ponto o bundle foi transferido para o seu destinatário final. Nesse ponto, a última entrada do

elemento “PATH” corresponde ao destinatário (no caso, “helio”). A data/hora no elemento “helio" corresponde ao momento em que o pacote foi recebido pelo dispositivo. Assim como nos demais pontos, o valor do RMSID é alterado para refletir o posição do *bundle* no dispositivo.

Nesse caso o bundle passou por apenas um nó intermediário, esse número, porém, pode ser bem maior, chegando a dezenas de nós intermediários.

Para verificar a viabilidade da utilização da arquitetura, principalmente em relação ao consumo de tempo para cada operação ou fase da arquitetura de comunicação, uma série de testes foi realizada. Para os testes, três dispositivos foram utilizados.

Esses dispositivos foram escolhidos por serem de um mesmo fabricante, diminuindo assim a chance de alguma diferença no sistema operacional ou em bibliotecas causar alguma alteração nos resultados.

Tempo de processamento do XML

Para descobrir o efeito que o aumento do arquivo XML de informações gera no processamento intermediário em um nó, um teste foi montado e executado em dispositivos.

No teste realizado, os dispositivos eram responsáveis por adicionar um determinado número de elementos ao XML e depois percorrer a lista toda, a procura de um elemento (que era o último). Como a busca é sequencial, tem-se sempre o pior caso. Esse processamento simula também um caso em que o algoritmo (de encaminhamento, por exemplo) precisa saber todos os nós pelos quais o *bundle* passou anteriormente para tomar a sua decisão.

Os testes foram realizados com um número crescente de elementos no campo PATH do XML de um *bundle*. Iniciando-se o teste com 32 elementos, a quantidade era dobrada até que o teste era realizado com 1024 elementos.

Os resultados estão demonstrados no gráfico abaixo.

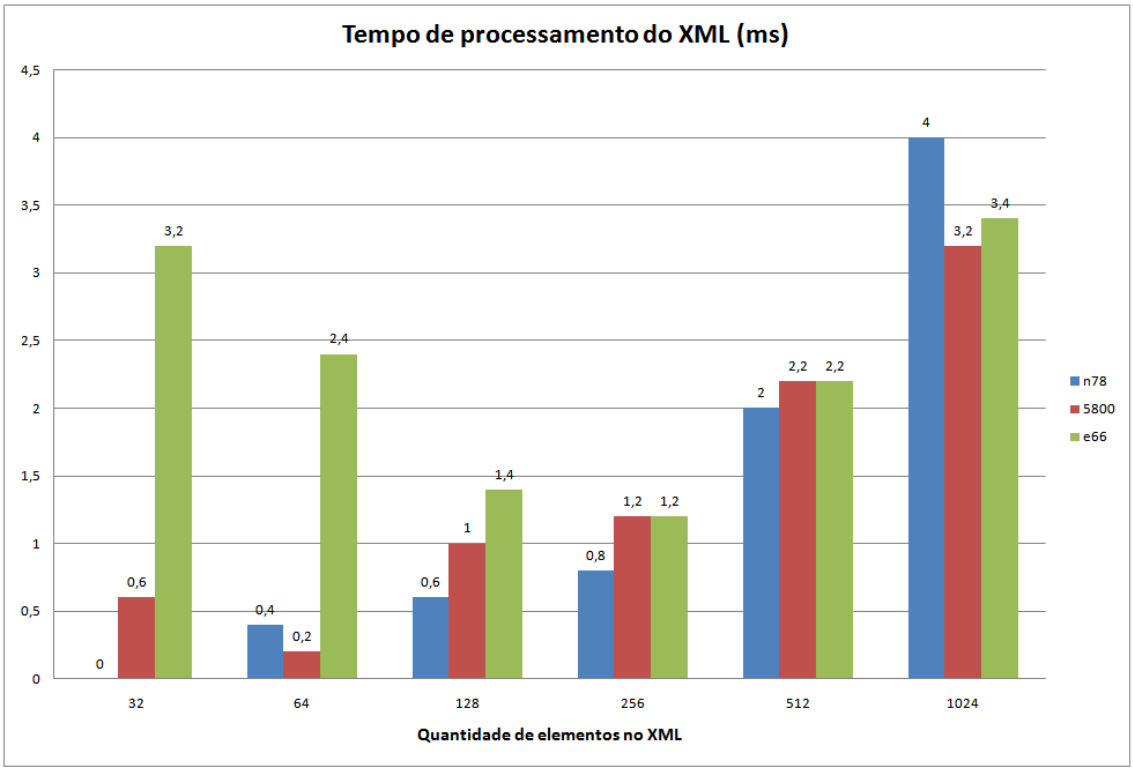

Figura 18 - Tempo de processamento (busca) do XML de encaminhamento em diferentes dispositivos

O tempo de criação é bem maior, ficando entre 300 e 600 milissegundos para o teste com 1024 elementos, pois leva em consideração a criação de cada elemento adicionado ao “PATH”, além da necessidade de adicionar esse elemento à lista no XML. Apesar de maior, esse tempo não tem impacto direto na utilização da arquitetura, já que se trata de um processamento que não ocorre durante o seu funcionamento. Esse tempo representa o tempo para criar o cenário do teste.

Nesse trabalho, porém, o maior interesse está na análise do tempo necessário para encontrar um elemento na lista, ou para percorrer ela toda. Esse tempo se mostra bastante baixo, mesmo com a lista tendo um tamanho improvável de ocorrer em situações práticas.

Uma questão interessante apareceu durante a análise dos dados. Por alguma razão que não foi possível determinar, o tempo necessário para a criação da lista com 64 nós em alguns dispositivos é maior que o tempo necessário para criar a lista com 128 nós. Testes complementares serão feitos para tentar descobrir se o comportamento se repete em outros pontos.

Tempo para manipular bundles

Como os dados que trafegam pelos nós ficam armazenados em memória secundária, a cada transferência esses dados precisam ser lidos pelo nó emissor e gravados pelo receptor.

Com o objetivo de avaliar o tempo necessário para cada uma dessas operações com variados tamanhos de blocos de dados, testes de gravação e leitura de dados foram executados. Os gráficos dos resultados se encontram a seguir.

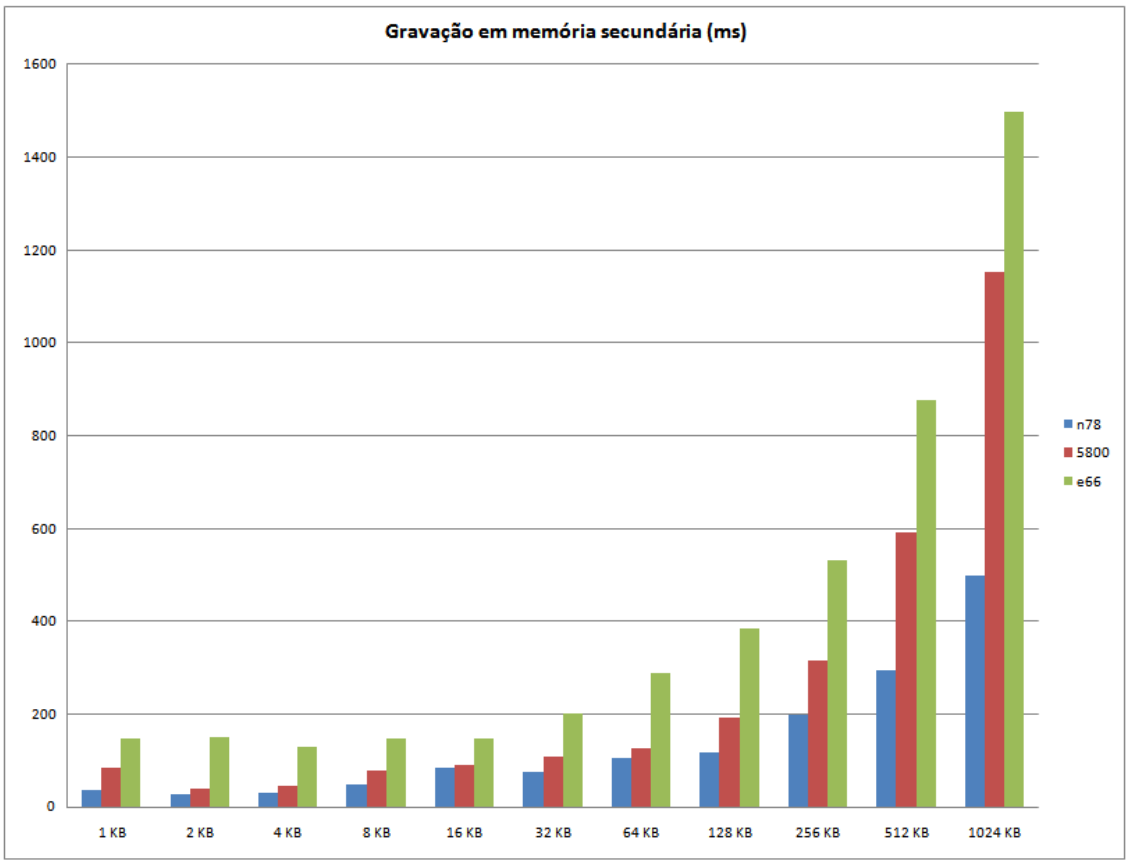

Figura 19 - Tempo de gravação (ms) de um bundle em memória secundária

Como pode ser observado no gráfico da Figura 19, os diferentes dispositivos têm comportamentos diferentes à medida que o volume de dados cresce. Apesar disso, o tempo de processamento ainda pode ser considerado aceitável na maior parte das aplicações (cerca de 1,5 segundo para gravar 1MB, no pior caso)

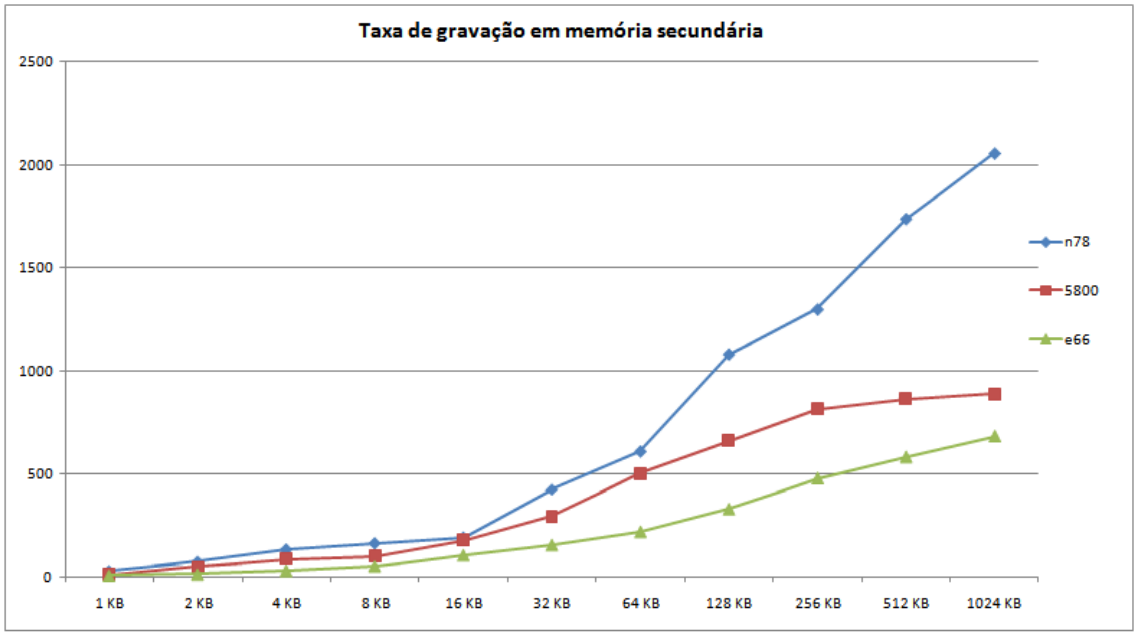

Figura 20 - Taxa de gravação (KB/s) em memória secundária

A Figura 20 ajuda a entender o comportamento dos tempos da Figura 19, demonstrando o comportamento de cada dispositivo à medida que o volume de dados cresce.

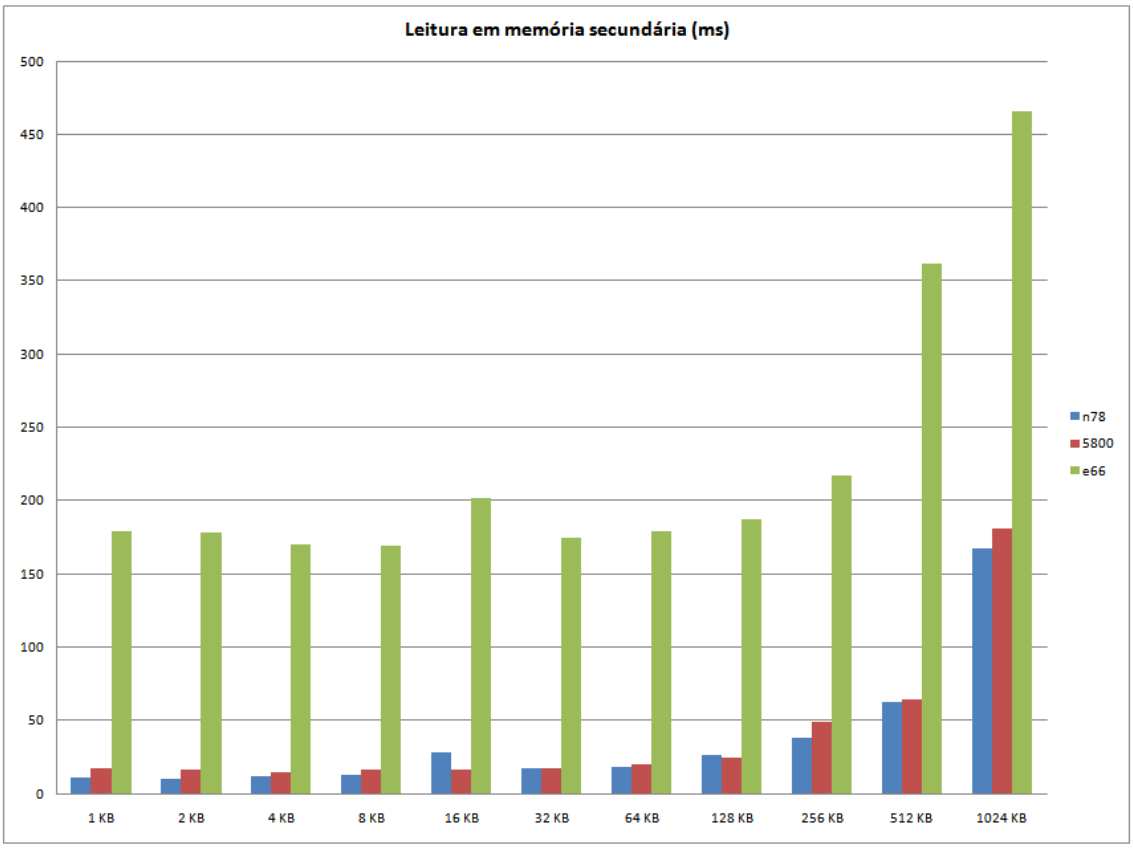

Figura 21 - Tempo de leitura (ms) de um bundle em memória secundária

Na leitura, os dispositivos voltaram a apresentar diferentes comportamentos, com dispositivos apresentando taxas parecidas (enquanto que na gravação os resultados diferiam consideravelmente).

Na Figura 21 o tempo necessário para que cada dispositivo pudesse recuperar um *bundle* de determinado tamanho é apresentado. A variação no gráfico sugere um crescimento significativo na taxa de leitura bastante significativo a medida que o tamanho do bloco de dados cresce.

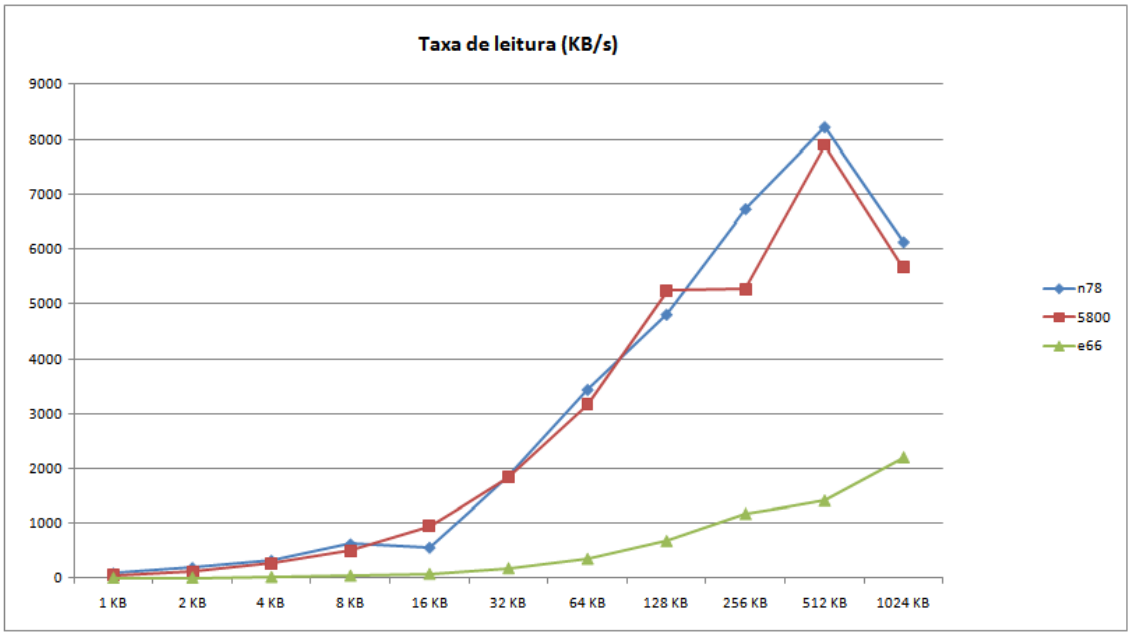

Figura 22 - Taxa de leitura (KB/s) em memória secundária

No gráfico apresentado pela Figura 22 podemos observar o crescimento na taxa de leitura. Assim como na gravação, blocos de dados “grandes” (em especial em blocos maiores que 16KB) a taxa de acesso (leitura ou gravação) cresce de maneira significativa.

Apesar de não haver uma justificativa fechada para a queda brusca de desempenho no teste com blocos de 1024KB, o fato do bloco estar próximo ao tamanho limite para a alocação de memória pode ter afetado negativamente o desempenho.

Esses resultados foram comprovados pela repetição dos testes, chegando a dados estatísticos bastante interessantes. Por exemplo, a variância no teste de leitura do dispositivo “n78” (que apresenta a queda brusca de

desempenho com blocos de 1024KB) foi de apenas 0,5. Com resultados variando entre 166 e 168 milissegundos nos testes.

Esse crescimento pode ter diversas razões, que não cabem serem discutidas aqui, mas que indicam um caminho na utilização de memória secundária nesses dispositivos, uma vez que unir diversos blocos de dados (*bundles*, por exemplo) pode trazer um ganho de desempenho bastante interessante. Por exemplo, a taxa de gravação em um dos dispositivos (n78) mais do que dobra ao se passar de um bloco de 16KB para um de 32KB (~193 KB/s para ~426 KB/s)

Ambos os testes, porém mostraram que o tratamento de grandes volumes de dados em dispositivos móveis ainda é uma questão delicada, pois apesar da evolução do hardware e do software, trabalhar com grandes quantidades de dados pode consumir um tempo significativo. Além disso, o comportamento dos dispositivos à medida que os blocos de dados aumentam não é completamente previsível, podendo melhorar ou piorar rapidamente.

Viabilidade da arquitetura - tempos

O desempenho do modelo de encaminhamento provido é dependente dos tempos de encontros dos usuários e pode depender também da política de encaminhamento adotada. Avaliações de desempenho, contudo, podem ser realizadas para analisar a eficiência dos diferentes algoritmos de encaminhamento.

Na questão do desempenho, o tempo que a arquitetura necessita para realizar as suas funcionalidades é uma questão de grande importância, uma vez que os encontros oportunísticos têm geralmente duração reduzida. Para analisar a viabilidade de sua utilização, os registros de utilização (logs) da arquitetura foram analisados para se obter o tempo necessário para que cada fase do funcionamento fosse realizada.

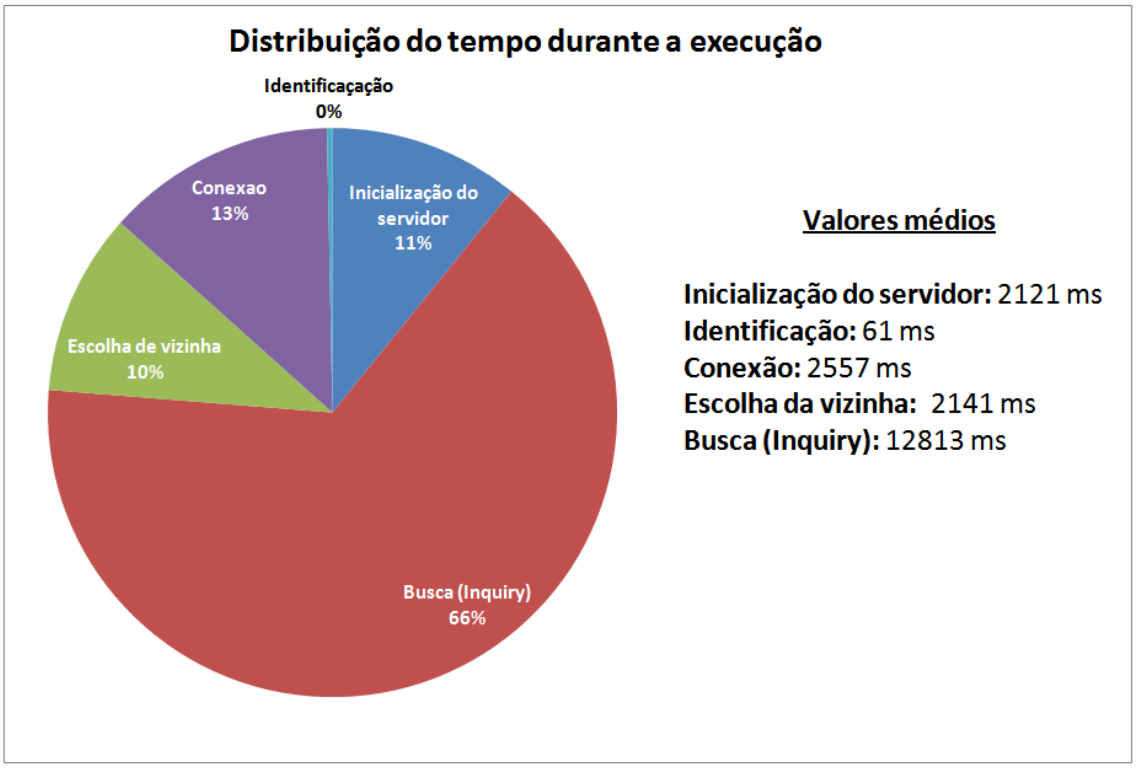

Figura 23 - Distribuição do tempo durante um encontro

Os resultados apresentados na Figura 23 indicam que apesar de bem estruturada, a arquitetura traz pouco impacto no tempo necessário para a formação da rede e a transmissão de dados entre os nós participantes. A maior parte do tempo utilizado pela arquitetura é gasto em funções que não podem ser alteradas. O *Inquiry* Bluetooth, por exemplo, tem a sua duração definida pela implementação da pilha de protocolos do dispositivo.

O tempo de transferência dos *bundles* foi omitido do cálculo de maneira proposital, uma vez que ele é proporcional ao volume de dados a ser transferido e a sua presença traria distorções ao gráfico.

Os tempos obtidos são altamente dependentes das tecnologias utilizadas. A comunicação Bluetooth, por exemplo, é muito suscetível à interferência. Nos testes realizados, a presença de um outro computador executando a operação de *Inquiry* no raio de alcance fez com que o tempo necessário para a operação saltasse de aproximadamente 12,8 segundos para cerca de 27 segundos.

Na Tabela 2 são apresentados valores médios da execução das funcionalidades da arquitetura num conjunto de celulares com transmissão pela tecnologia Bluetooth.

A utilização de criptografia foi suprimida do teste, uma vez que o impacto dessa funcionalidade é dependente da capacidade de processamento do dispositivo, não trazendo alterações significativas no desempenho da comunicação, aqui avaliada. Testes futuros serão realizados com essa funcionalidade específica.

Tabela 2 - Distribuição do tempo gasto durante os encontros (em segundos).

	Média	Mediana	Variância
Inicialização do serviço	2,056	2,121	0,96
Busca de dispositivos	14,95	12,813	9,082
Escolha de dispositivos	2,687	2,141	3,781
Conexao	2,552	2,557	0,446
Identificaçação	0,067	0,061	0

Conforme é possível observar na Tabela 2, o tempo de busca de dispositivos usando é alto (cerca 14,9 segundos em média). Esse tempo, porém, varia bastante de acordo com interferências externas.

O tempo de escolha de dispositivos representa o tempo necessário para descobrir quais dos nós encontrados possuem o serviço desejado operando no momento. Além disso, esse tempo engloba o tempo que o algoritmo de escolha leva para processar e decidir com qual nó será feita a conexão.

O tempo de conexão apresentado na tabela representa o tempo necessário para criar uma conexão Bluetooth com o nó escolhido no passo anterior. A identificação do nó corresponde ao período de troca de 3 mensagens entre os dispositivos, com o objetivo de realizar uma identificação mútua deles. Nesse ponto deve ser considerado o tempo gasto com operações de criptografia, caso ela esteja em uso.

O tempo de transferência compreende o tempo de negociação (quais *bundles* serão enviados) e o tempo de transferência (a transmissão efetiva desses *bundles*). A negociação envolve o envio dos arquivos XML de descrição dos *bundles* e a resposta do nó receptor com a informação de quais *bundles* ele aceita receber. Nos testes foram enviados 11 pacotes, com crescimento exponencial no tamanho (1KB, 2KB, 4KB, até 1024KB). A mediana

dos tempos necessários para completar essas transferências (dos 11 bundles) foi de 20070 milissegundos (20.07s).

A transferência de dados entre os dispositivos também apresenta um comportamento interessante, como pode ser observado a seguir.

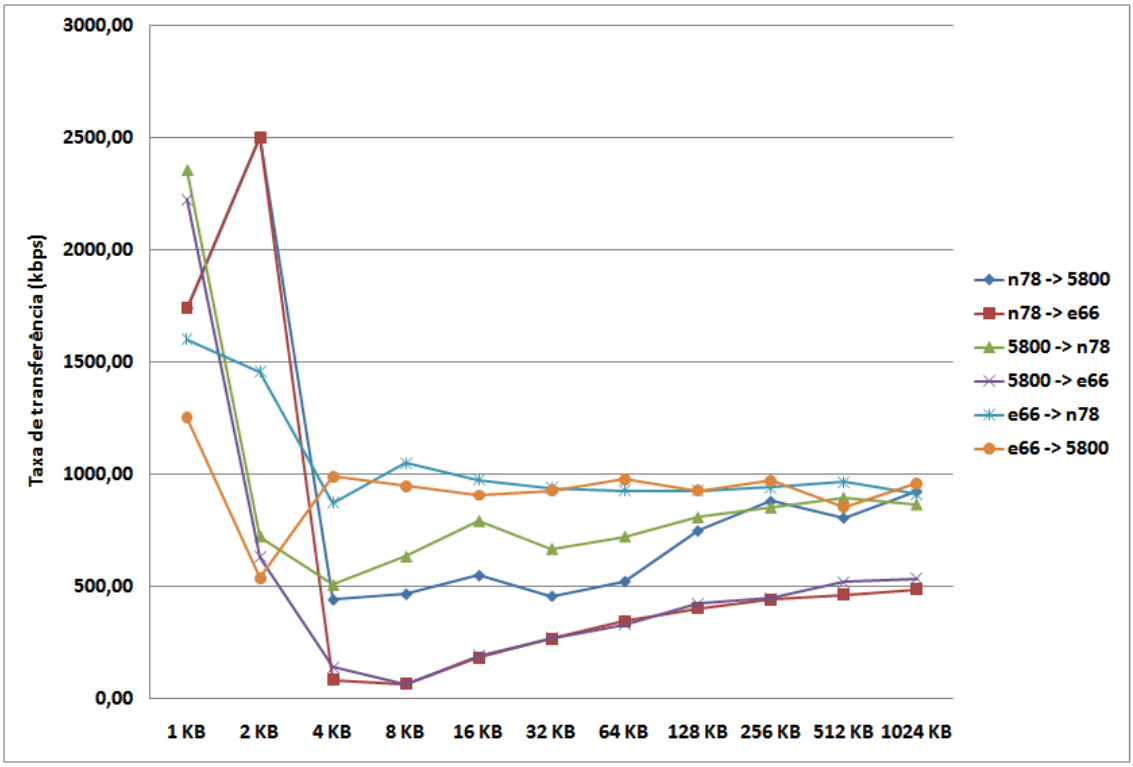

Figura 24 - Taxa de transferência (KB/s) via Bluetooth

Como pode ser observado na Figura 24, a taxa de transferência tem uma flutuação inicial bastante grande, que não deve ser considerada como válida para blocos de dados maiores, uma vez que é resultado de uma distorção que pacotes muito pequenos causam no calculo. Porém, à medida que o tamanho do bloco de dados cresce, a taxa de transferência se estabiliza rapidamente na maior parte dos casos.

Outro ponto importante de ser notado é que a inversão de papéis (emissor e receptor) causa um impacto significativo na taxa de transferência. Assim, por exemplo, o dispositivo “e66” é 50% mais lento para receber dados do dispositivo “5800” do que para enviá-los ao mesmo dispositivo.

Inferimos nos nossos testes que o dispositivo “e66” é mais lento na recepção (e não que os demais são mais lentos no envio), pois ao se utilizar um outro receptor, os resultados são significativamente superiores. Por

exemplo, o mesmo dispositivo “5800” tem taxas equivalentes aos demais dispositivos quando envia dados a outro dispositivo (no caso, o dispositivo “n78”).

Exemplos de uso

Para exemplificar o que as taxas de transferências obtidas significam, listamos alguns exemplos de arquivos comumente transferidos entre usuários. Esses exemplos podem ser vistos na Tabela 3.

Tabela 3 - Exemplos de arquivos e tempo necessário para transmissão

Tipo de arquivo	Tamanho aproximado	Tempo de transferência (aproximado)
E-mail	4KB	70ms
Toque celular	300KB	3s
Foto	1MB	9s
Música (.mp3)	4MB	40s

A Tabela 3 foi obtida a partir dos dados de taxa de transferência, pegando a taxa de transferência mais próxima do tamanho do arquivo e extrapolando-a quando necessário. Estes são apenas exemplos, para ilustrar melhor o que a taxa as taxas de transferência obtidas significam.

Como pode ser visto na Tabela 3, mesmo a tecnologia Bluetooth sendo mais voltada para a economia de energia do que para uma grande taxa de transmissão, a maior parte dos arquivos que os usuários costuma trocar pode ser transferida num tempo aceitável.

Segurança

Durante os testes realizados, a criptografia não foi utilizada uma vez que a questão da segurança dos *bundles* é de enorme complexidade. Por exemplo, diversos métodos e algoritmos podem ser usados, cada um com suas vantagens, desvantagens e características.

Apesar do modelo de segurança ter sido especificado anteriormente, a sua implementação ainda não foi incluída no código da arquitetura. Uma série

de testes está sendo realizada em um trabalho separado, a fim de determinar quais os melhores parâmetros a serem utilizados.

Balancear os algoritmos escolhidos, o tamanho das chaves e demais parâmetros e o grau de segurança desejado é um trabalho complexo e extenso demais para ser incluído nesse projeto. Por essa razão um projeto paralelo está sendo desenvolvido para investigar a capacidade dos dispositivos móveis em lidar com os mecanismos de criptografia e o impacto que esses mecanismos trazem ao desempenho dos softwares que fazem uso deles.

Entre os parâmetros estão, por exemplo, o tamanho do bloco de dados e o tamanho das chaves (tanto da simétrica quanto das assimétricas). Essas características têm impacto significativo tanto na segurança do *bundle* quanto no desempenho. Alguns testes preliminares demonstram as diferenças de desempenho dos diferentes algoritmos.

A implementação foi realizada utilizando-se a API Bouncy Castle[4]. Essa API permite o uso de mecanismos de criptografia simétrica e assimétrica em Java nas suas diferentes plataformas. No JME (dispositivos móveis), algumas restrições existem, porém a manipulação é possível e foi implementada com sucesso.

Um conjunto preliminar de testes foi realizado para permitir uma análise e a escolha dos algoritmos que serão utilizados na arquitetura. Por exemplo, na criptografia simétrica, o algoritmo AES demora cerca de 60 milissegundos para criptografar cada bloco de 8 KB, utilizando uma chave de 128 KB. Com uma chave de 256 KB o tempo necessário é de cerca de 85 milissegundo para o mesmo tamanho de bloco, ou seja, um tempo aproximadamente 30% maior.

4.3 Estudos de caso

Para demonstrar as funcionalidades da arquitetura de comunicação apresentada nesse trabalho, dois estudos de caso foram conduzidos. O

[4] Disponível em: http://www.bouncycastle.org/

primeiro caso foi a utilização da arquitetura com um serviço de mensagens pessoais. No segundo caso, um serviço de envio de mensagens ao *Twitter* foi desenvolvido. Ambos os estudos de caso se encontram descritos abaixo.

Envio de mensagens

O envio de mensagens constitui o serviço básico da arquitetura, permitindo que um nó envie uma mensagem de texto a outro nó. As mensagens não têm uma limitação de tamanho, sendo armazenadas como *payload* do *bundle*.

O serviço oferece a possibilidade de se enviar uma mensagem para um usuário, informando o nome do destinatário, o título e o conteúdo da mensagem.

Após o envio, a mensagem trafega juntamente com os demais *bundles* dos nós da arquitetura até (eventualmente) alcançar o seu destinatário. Nesse momento o usuário é avisado que ele recebeu uma nova mensagem, que está disponível para sua leitura quando desejar.

Figura 25 - Configuração dos dispositivos e envio de mensagem

A Figura 25 apresenta a tela de configuração do serviço de envio de mensagens (na parte esquerda) e a tela de envio de mensagens (na parte direita). O serviço, apesar de simples, cumpriu o objetivo de validar as funcionalidades da arquitetura, operando com diferentes algoritmos de encaminhamento entre os dispositivos. Além disso, os *logs* de utilização serviram para a análise do tempo necessário para cada fase da arquitetura, conforme descrito anteriormente.

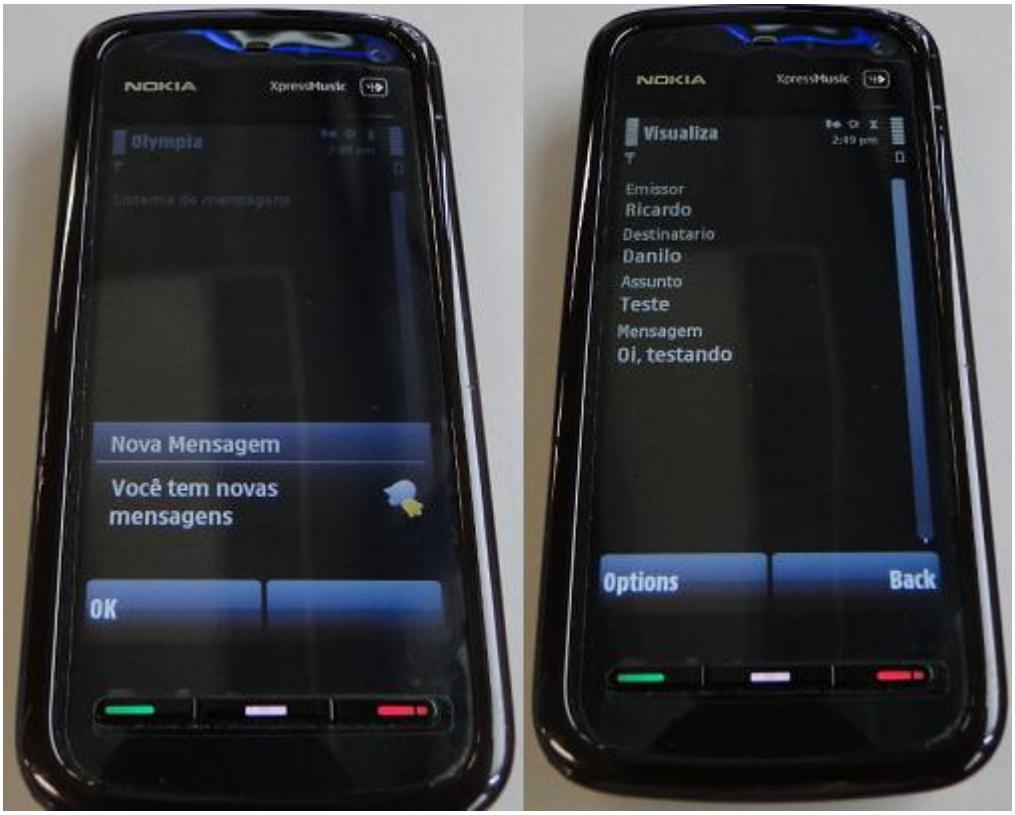

Figura 26 - Recebimento de uma mensagem e sua posterior visualização

Durante os testes, três algoritmos de encaminhamento foram utilizados: Entrega Direta, Encaminhamento Aleatório e Entrega Aleatória Sem Repetição.

Na entrega direta, o emissor retém a mensagem até que entre em contato com o destinatário da mensagem, fazendo a entrega da mensagem somente a ele.

Na Entrega Aleatória, o nó repassa as mensagens ao primeiro nó que as aceitar. esse comportamento pode criar “círculos” (*loops*) entre os nós. Quando um nó recebe uma mensagem que seja destinada a ele, ele não mais a repassa.

Na Entrega Aleatória Sem Repetição o nó também tenta entregar as mensagens que estão sob sua custódia para o primeiro nó que encontrar, este porém só aceita as mensagens que ainda não estiveram com ele. Assim, nenhum *loop* é criado. Para essa decisão, o XML de encaminhamento do *bundle* é analisado, assim o nó pode saber se já esteve de posse daquela mensagem.

O desempenho do encaminhamento, nesse caso, é dependente dos encontros ocorridos e está sujeito aos tempos avaliados anteriormente. A avaliação da eficiência de cada algoritmo, contudo, depende de um estudo de encontros entre os usuários envolvidos.

Envio de mensagens via Twitter

Esse serviço permite o envio de *tweets* (como são chamadas as mensagens do Twitter) através de um sistema que envolve a comunicação entre nós móveis da rede e o acesso à Internet através de uma infraestrutura de rede.

Figura 27- Tela inicial e de configuração do serviço de acesso ao Twitter

O usuário, antes de enviar uma mensagem, deve configurar o serviço, informando o seu nome de usuário e senha do Twitter. Essa informação é enviada juntamente com a mensagem, para que a mesma possa ser publicada.

Figura 28 - Envio de mensagem ao Twitter

Quando uma mensagem é enviada, como pode ser observado na Figura 28 (limitada a 140 caracteres por restrição do Twitter) esta é alocada no *payload* do *bundle*, juntamente com o nome e senha do usuário. A mensagem é encaminhada de maneira usual, como se fosse uma mensagem de texto do serviço de mensagens, com a diferença de que o serviço além de buscar por outros nós móveis também realiza regularmente a busca por um ponto de acesso da arquitetura que tenha o serviço de servidor desse serviço.

O destinatário de um pacote desse serviço sempre é um servidor do serviço, uma vez que eles são os pontos de conexão com o servidor do Twitter.

Ao encontrar um servidor capaz de lidar com mensagens desse serviço, o dispositivo envia o *bundle* em questão para ele, que faz a conexão com o servidor do Twitter via Internet e realiza a postagem da mensagem.

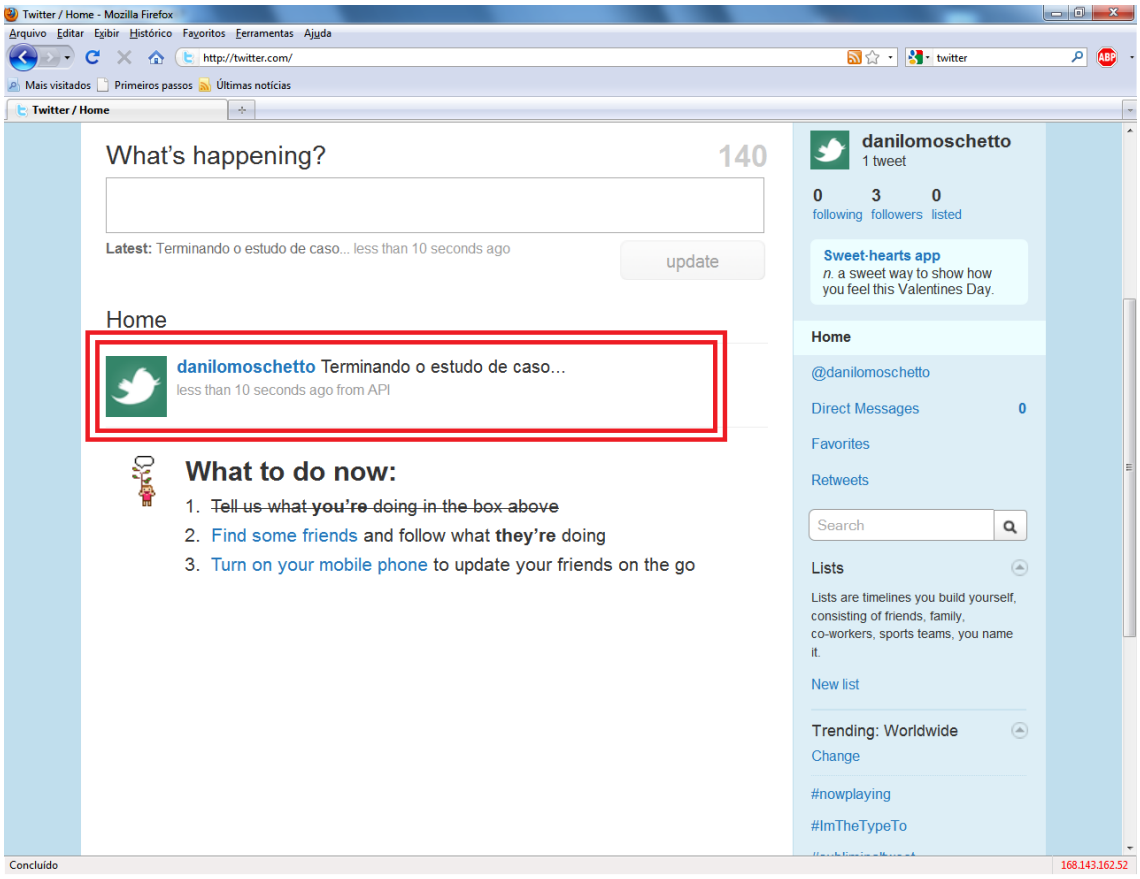

Figura 29 - Visualização da postagem na página do Twitter

Para a postagem, diversos métodos estão disponíveis. Nesse trabalho, foi escolhida a própria API disponibilizada pelo Twitter. Após o envio a mensagem pode ser acessada pela página do Twitter, como pode ser observado na Figura 29

CAPÍTULO 5

CONCLUSÕES E TRABALHOS FUTUROS

5.1 Conclusões

O crescente número de dispositivos móveis com elevada capacidade computacional e a melhora nas tecnologias de comunicação sem fio desses dispositivos criaram a expectativa de novas formas de comunicação entre os seus usuários.

Apesar de novos paradigmas de comunicação, como a comunicação oportunística, terem se desenvolvido, há ainda poucas funcionalidades práticas disponíveis de fato aos usuários finais.

Entre dificuldades para o desenvolvimento da área está a falta de arquiteturas de comunicação que permitam ao desenvolvedor se focar nas funcionalidades que interessam ao usuário.

Assim, esse trabalho apresentou um conjunto informações sobre o desenvolvimento de uma arquitetura de comunicação oportunística, levando em consideração os diversos interesses que podem estar envolvidos no seu desenvolvimento e na sua utilização. Além disso, cenários relevantes foram descritos e analisados para a utilização dessa arquitetura.

A arquitetura especificada e implementada segue os princípios teóricos desenvolvidos no trabalho, demonstrando o seu funcionamento. Apesar de se tratar de um projeto que ainda precisa de desenvolvimento e novas

funcionalidades, o seu funcionamento foi bastante satisfatório, demonstrando a viabilidade de uma solução genérica para a comunicação oportunística em dispositivos móveis.

Novas avaliações efetivas da arquitetura deverão ser realizadas, mas observa-se que a estruturação do protocolo definido para a identificação dos nós, para a negociação e transmissão dos pacotes e para a contabilização o desempenho mostrou-se viável, não ocupando uma parcela significativa do tempo disponível para o contato.

Dois serviços foram desenvolvidos e testados utilizando a arquitetura desenvolvida nesse trabalho. Suas funcionalidades foram testadas em diferentes celulares, demonstrando a portabilidade pretendida com a utilização do JME.

Os serviços foram utilizados com sucesso por usuários voluntários. Testes em maior escala e avaliações de desempenho e usabilidade ainda necessitam ser realizados. Porém, os testes realizados servem como prova de conceito, demonstrando o funcionamento da arquitetura de comunicação desenvolvida.

O avanço das tecnologias de transmissão sem fio, juntamente com o desenvolvimento do hardware e do software dos dispositivos móveis deve impulsionar novas aplicações e formas de comunicação. Nesse ponto a comunicação oportunística merece um lugar de destaque, já que pode oferecer solução para alguns problemas que de difícil solução para a comunicação infraestruturada, como zonas sem cobertura e a mobilidade.

O aumento no número e na complexidade das redes sociais tem impacto tanto na demanda, com as pessoas desejando formas de contato mais diretas e ágeis, quanto na capacidade dessas redes, uma vez que esse crescimento pode ser utilizado para melhorar a capacidade de encaminhamento desse tipo de rede.

A arquitetura desenvolvida provê um ambiente propício para o estudo de novas formas de comunicação e de serviços de aplicação para os usuários. Estudos específicos relacionados à forma de encaminhamento dos dados também podem ser realizados sobre a plataforma desenvolvida.

5.2 Trabalhos Futuros

Considerando a funcionalidade da arquitetura desenvolvida em seu estado atual, novos projetos de pesquisa devem ser baseados sobre sua utilização.

A determinação da relação entre durações de encontros casuais e tamanhos de bundles possíveis de serem transmitidos, bem como da fragmentação das transmissões são assuntos de interesse.

Considerando o desenvolvimento de novas tecnologias e melhorias nos padrões de rede existentes, como o Wi-Fi direct, devem favorecer a pesquisa do uso de novas tecnologias de transmissão.

Da mesma maneira, características do padrão Bluetooth 3 parecem adequadas para ampliar as taxas de transmissões de dados, mantendo reduzido o consumo de energia nas operações de consulta da rede e vizinhos ativos.

Novos serviços devem ser desenvolvidos sobre a arquitetura.

Além disso, a coleta de dados realizada com o uso da arquitetura deve prover informações relevantes para a pesquisa de padrões de acesso, encontros entre usuários e formações de redes sociais, que esperamos explorar para a criação de algoritmos de transmissão oportunística.

Capítulo 6

REFERÊNCIAS BIBLIOGRÁFICAS

ABOLHASAN, M.; Wysocki, T.; Dutkiewicz, E. (2004), “A review of routing protocols for mobile ad hoc networks”, Ad Hoc Networks, vol 2, no. 1, páginas 1-22

BANERJEE, N.; Corner, M. D.; Towsley, D.; Levine B. N. (2008) “Relays, Base Stations, and Meshes: Enhancing Mobile Networks with Infrastructure”, . In Proceedings of the 14th ACM international Conference on Mobile Computing and Networking. MobiCom '08. ACM, New York, NY, 81-91.

BURGESS, J.; Gallagher, B.; Jensen, D.; Levine, B. N. (2006), “Max-prop: Routing for vehicle-based delay-tolerant networks”, IEEE Infocom.

CHAINTREAU, A.; Hui, P.; Crowcroft, J.; Diot, C.; Gass, R.; Scott, J. (2007), "Impact of Human Mobility on Opportunistic Forwarding Algorithms," IEEE Transactions on Mobile Computing, vol. 6, no. 6, páginas 606-620.

CRUZ, Erlon Rodrigues. Arquitetura para controle de políticas de tarifação em redes WiMAX mesh. São Carlos: UFSCar, 2009. 105p. Dissertação (Mestrado) – Programa de Pós Graduação em Ciência da Computação. São Carlos, Universidade Federal de São Carlos, 2009

GUSTAFSSON, E.; Jonsson, A.; (2003), "Always best connected", Wireless Communications, IEEE, vol.10, no.1, páginas 49-55.

JUBIN, J.; Tornow, J. (1987), "The DARPA Packet Radio Network Protocols," Proc. IEEE, vol. 75, no. 1, páginas 21-32.

LINDGREN, A. e Doria, A. (2006) "Probabilistic Routing Protocol for Intermittently Connected Networks", Internet Draft draft-lindgren-dtnrgprophet-02.

LOYAL, T. (2004), "MANET: The future of the battlefield communications?", Intercom - Journal of the Air Force C41SR community.

MITCHENER, W.; Vadhat, A. (2000) "Epidemic Routing for Partially Connected Ad hoc Networks", Technical Report.

PELUSI, L.; Passarella, A.; Conti, M. (2006) "Opportunistic Networking: Data Forwarding in Disconnected Mobile Ad Hoc Networks," IEEE Commun. Mag., Dec. 2006.

SCHILLER, J.; Voisard, A. (2004), "Location-Based Services", 1ª edição, Morgan Kaufmann Publishers: San Francisco. ISBN 1-55860-929-6

SHAH, R.; Roy, S.; Jain, S.; Brunette, W. (2003), "Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks", IEEE SNPA Workshop.

SPYROPOULOS T., Psounis, K. e Raghavendra, C. S. (2004) "Single-copy routing in intermittently connected mobile networks", Sensor and Ad Hoc Communications and Networks - SECON.

SU, J.; Scott, J.; Hui, P.;Upton, E.; Lim, M. H.; Diot, C.; Crowcroft, J.; Goel, A.; Lara, E. de.; (2007) "Haggle: Clean-slate Networking for Mobile

Devices". Technical Report UCAM-CL-TR-680, University of Cambridge, Computer Laboratory, Jan. 2007.

WOOD, L.; Eddy, W. M.; Ivancic, W.; McKim, J.; Jackson, C. (2007a) "Saratoga: a delay-tolerant networking convergence layer with efficient link utilization," Proceeding of IWSSC'07, 2007.

WOOD, L.; Eddy, W.; Holliday, P; (2009) "A Bundle of Problems", IEEE Aerospace conference

WOOD, L.; McKim, J.; Ivancic, W.; ackson, C. (2007b) "Saratoga: Efficient Transport over Short-Lived Links". Apresentação Técnica. IETF 69 TSV area group meeting. Disponível em http://info.ee.surrey.ac.uk/Personal/

ZHANG Z. (2006), "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges," IEEE Communications Surveys & Tutorials, vol.8, no.1, páginas 24-37.