GIL EDUARDO DE ANDRADE

VELOSENT: PROTOCOLO DE ROTEAMENTO SENSÍVEL AO CONTEXTO: POSIÇÃO, VELOCIDADE E SENTIDO, PARA REDES TOLERANTES A ATRASOS E DESCONEXÕES

CURITIBA

AGOSTO 2012

GIL EDUARDO DE ANDRADE

VELOSENT: PROTOCOLO DE ROTEAMENTO SENSÍVEL AO CONTEXTO: POSIÇÃO, VELOCIDADE E SENTIDO, PARA REDES TOLERANTES A ATRASOS E DESCONEXÕES

Dissertação apresentada como requisito parcial à obtenção do título de Mestre em Informática, pelo Programa de Pós-Graduação da Pontifícia Universidade Católica do Paraná

Área de Concentração: Ciência da Computação

Orientador: Prof. Dr. Luis Augusto de Paula Lima

CURITIBA

AGOSTO 2012

Termo de Aprovação

GIL EDUARDO DE ANDRADE

VELOSENT: PROTOCOLO DE ROTEAMENTO SENSÍVEL AO CONTEXTO: POSIÇÃO, VELOCIDADE E SENTIDO, PARA REDES TOLERANTES A ATRASOS E DESCONEXÕES

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre em Informática, pelo Programa de Pós-Graduação da Pontifícia Universidade Católica do Paraná, pela seguinte banca examinadora:

Prof. Dr. Luiz Augusto de Paula Lima
Pontifícia Universidade Católica do Paraná

Prof. Dr. Alcides Calsavara
Pontifícia Universidade Católica do Paraná

Prof. Dr. Carlos Alberto Maziero
Universidade Tecnológica Federal do Paraná

Dedicatória

A minha família,

Que me ensinou a importância de lutar pelos meus ideais,
podendo assim realizar todos os meus objetivos.

Agradecimentos

A todas as pessoas que, direta ou indiretamente, contribuíram para a realização deste trabalho.

Epígrafe

"...o comprometimento é a base das vitórias, quando nos comprometemos com o que acreditamos ser certo deixamos de lado as dúvidas, os por ques, e assim caminhamos, passo a passo, rumo ao sucesso, construímos a nossa felicidade..."
(Gil Eduardo de Andrade)

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

DTN	Redes Tolerantes a Falhas e Desconexões
GPS	Sistema de Posicionamento Global
API	Interface de Programação de Aplicativos
IR	Infravermelho
QoC	Qualidade de Contexto
QoS	Qualidade de Serviço
XML	Linguagem Extensível de Marcação
HTTP	Protocolo de Transferência de Hipertexto
CEP	Código de Endereçamento Postal
SMS	Serviço de Mensagens Curtas
MMS	Serviço de Mensagens Multimídia
GPRS	Serviços Gerais de Pacote por Rádio
URL	Localizador Padrão de Recursos
IP	Protocolo de Internet
NATs	Tradução de Endereços de Rede
UDP	Protocolo de Datagramas do Usuário
TCP/IP	Protocolo de Controle de Transmissão - Protocolo de Internet
TC	Transferência de Custódia
ACK	Mensagem de Reconhecimento
TCUCE	Tabela de Contexto dos Último Contatos Efetuados
TTL	Tempo de Vida
MRU	Movimento Retilíneo Uniforme
IC	Idade do Contato
RWP	Rota Aleatória

RW	Caminhada Aleatória
EASE	Algoritmo de Roteamento da Idade Exponencial de Pesquisa
LER	Roteamento do Último Encontro
ONE	Ambiente de Simulação para Redes Oportunistas

Lista de Símbolos

T	Momento de Encontro entre dois nós
P_x	Posição no eixo x
P_y	Posição no eixo y
V_x	Velocidade no eixo x
V_y	Velocidade no eixo y
V_m	Velocidade Média
Δx	Diferença entre posição inicial e final
Δt	Diferença entre tempo final e inicial
α	Coeficiente Angular
δ	Coeficiente Linear
X_{max}	Posição Máxima em x
Y_{max}	Posição Máxima em y
V_{min}	Velocidade Mínima
V_{max}	Velocidade Máxima
T_{pausa}	Tempo de Pausa

Resumo

O VeloSent, Protocolo de Roteamento Sensível ao Contexto, tem por objetivo obter e processar os dados do ambiente: velocidade, direção e sentido dos nós, durante os contatos realizados em uma rede Tolerante a Atrasos e Desconexões. Essas informações de contexto são armazenadas e utilizadas pelo protocolo juntamente com os dados de posição e momento dos últimos encontros efetuados, que permitem estimar a posição do nó destino para uma determinada mensagem. A metodologia implementada pelo VeloSent permite, além da estimativa da posição do destinatário, identificar qual dos nós vizinhos se move em sua direção e com qual velocidade o faz, sendo essas informações de contexto peças chave para determinar a escolha do próximo nó responsável a dar continuidade ao roteamento da mensagem.

Palavras-chave: Contexto, Sensibilidade, Adaptação, Roteamento, Redes, DTN, Protocolo, Simulador, The ONE.

Abstract

The VeloSent, Context-Sensitive Routing Protocol aims at obtaining and processing speed and direction data during contacts established by nodes in Disruption-Tolerant Networks (DTNs). Such context informations is used by the protocol along with the position and time of past encounters enabling a better estimate of the of a particular message. VeloSent uses this strategy in order to decide which of the neighboring nodes will more probably reach the target node (assuming the nodes move in straight lines with a constant speed).

Key-words: Context, Context-aware, Routing, Network, DTN, Protocol, Simulator, The ONE.

1 Introdução

Os avanços da tecnologia na fabricação de componentes eletrônicos têm aumentado a capacidade de armazenamento, processamento e comunicação dos dispositivos móveis, afetando de forma positiva a computação ubíqua, definida, pela primeira vez, pelo professor Mark Wiser, cientista chefe do Centro de pesquisa Xerox.

No seu artigo ele aborda a questão de que no futuro, o usuário estará centralizado na tarefa e não mais relacionado prioritariamente com a ferramenta utilizada, sendo a tecnologia enraizada implicitamente no contexto. Para Weiser, "as tecnologias mais profundas são aquelas que desaparecem. Elas se entrelaçam com o cotidiano até que se tornem indistinguíveis dele".

Dentro da computação ubíqua destaca-se a computação ciente ou sensível ao contexto. Esta computação define uma área de pesquisa relativamente recente e prevê a disponibilização de computação e comunicação sem fio o tempo todo e em todo lugar, permitindo que tais capacidades sejam incorporadas a elementos do dia-a-dia e utilizadas de maneira transparente (GREENFIELD, 2011).

A computação sensível ao contexto tem por objetivo, de uma forma geral, elaborar técnicas para coletar informações para dispositivos computacionais: entradas que reflitam as condições atuais do usuário, do ambiente no qual o mesmo se encontra e do próprio dispositivo computacional utilizado, considerando suas características de hardware, software e comunicação. Essas entradas são os chamados contextos, definição formalizada como sendo "Qualquer informação que possa ser utilizada para caracterizar a situação de entidades (pessoa, lugar ou objeto) que sejam consideradas relevantes para interação entre um usuário e uma aplicação (incluindo o usuário e a aplicação)" (DEY, 2001).

A sensibilidade ao contexto vem sendo utilizada como recurso para diversas aplicações, nos mais variados campos de pesquisa, dentre os quais destaca-se a área de Redes Tolerantes a Falhas e Desconexões, ou apenas DTN (Delay and Disruption Tolerant Networks - DTNs). Quando pensamos na arquitetura da Internet, constatamos que é uma solução tecnológica de sucesso, utilizada no mundo todo para interconectar os mais variados tipos de dispositivos de

comunicação, em diferentes cenários e dando suporte a diversas aplicações.

Contudo, algumas premissas necessárias ao bom funcionamento dessa arquitetura não são encontradas em determinados ambientes, tornando o perfil dos protocolos da Internet inadequado e pouco robusto. Exemplos de tais ambientes são: comunicações sem fio, comunicações entre dispositivos móveis, comunicações entre dispositivos com restrições de energia, comunicações rurais, comunicações submarinas, comunicações interplanetárias etc. Estes ambientes, considerados desafiadores, possuem em comum a dificuldade de manter uma comunicação fim-a-fim com baixa latência e baixa perda de pacotes. Devido a estas características, as redes que consideram estes aspectos foram denominadas Redes Tolerantes a Atrasos e Desconexões (FALL, 2003).

Para aplicações DTN, a garantia de entrega da mensagem é a mais importante métrica de desempenho, superando até mesmo o atraso. Sendo assim, um desafio, comum em redes DTN, é o roteamento das mensagens, havendo a necessidade de projetarem-se novos protocolos que sejam capazes de superar os problemas relacionados aos longos atrasos e frequentes desconexões, já que os protocolos atuais não são eficientes para transmissão de dados em uma rede com essas características.

Dentro dessa ótica, propõem-se a modelagem e o desenvolvimento de um protocolo sensível ao contexto, com intuito de rotear mensagens em redes tolerantes a atrasos e desconexões. O protocolo tem por objetivo obter, interpretar e analisar dados de contexto característicos das DTNs, mais especificamente a posição, a velocidade e o sentido dos nós que a compõem. Os dados obtidos possibilitam que o protocolo se adapte as restrições do ambiente, e assim utilize de forma otimizada os recursos nele disponível, que são limitados, e ao mesmo tempo consiga obter um bom desempenho com relação à entrega das mensagens, produzindo um baixo custo (menor consumo) e alto benefício (maior entrega).

O restante da dissertação está assim organizado. O Capítulo 2 aborda todos os conceitos sobre computação sensível ao contexto necessários para o desenvolvimento da proposta, são eles: contexto, sensibilidade ao contexto e adaptação ao contexto. O Capítulo 3 aborda todos os conceitos sobre a arquitetura DTN e suas principais características, como: o roteamento de mensagens em DTN e os principais protocolos de roteamento utilizados. O Capítulo 4 apresenta toda a modelagem matemática utilizada como metodologia para implementação e funcionamento do protocolo de roteamento proposto: VeloSent. No Capítulo 5 são apresentados os modelos de mobilidade utilizados durante os processos de simulação, bem como o trabalho na literatura mais alinhado com a proposta. O Capítulo 6 descreve o ambiente de simulação utilizado bem como todos os resultados obtidos em comparação com outros 3 protocolos.

2 Computação Sensível ao Contexto

Neste capítulo são apresentados os conceitos relacionados às tecnologias utilizadas para o desenvolvimento do trabalho proposto. Primeiramente são apresentadas as definições que envolvem a computação sensível ao contexto, tais como contexto e sensibilidade ao contexto; Também são descritos os mecanismos necessários para obtenção e processamento das informações relacionadas ao ambiente onde o usuário se encontra. Em um segundo momento, uma visão geral sobre os aplicativos mais relevantes, desenvolvidos utilizando-se do recurso de sensibilidade ao contexto é apresentada.

Por fim são apresentados os conceitos relativos às Redes Tolerantes a Atrasos e Desconexões - DTNs, descrevendo suas principais características, o funcionamento de sua arquitetura e os protocolos de roteamento que permitem a transmissão e entrega de mensagens neste tipo de rede.

2.1 Contexto e classificação de contexto

Ao longo dos anos vários pesquisadores tem se dedicado a tarefa de definir os conceitos de Contexto e Sensibilidade ao Contexto, para tal, muitos deles têm utilizado sinônimos, referindo-se a Contexto como localização, identidades de pessoas, objetos próximos e as alterações desses objetos (SCHILIT; ADAMS; WANT, 1994). Brown define contexto como a localização, as identidades de pessoas ao redor do usuário, a hora do dia, estação e temperatura (BROWN; BOVEY; CHEN, 1997). Ryan define como contexto o local do usuário, ambiente, identidade e tempo (RYAN; PASCOE; MORSE, 1998). Para Dey o contexto pode ser definido como o estado emocional do usuário, seu foco de atenção, localização e orientação, data e hora, objetos e pessoas no seu ambiente (DEY, 1998). Para Schmidt o contexto é definido como "conhecimento sobre o estado do usuário e o estado do dispositivo, incluindo o seu redor, situação, e localização".

Por outro lado, alguns pesquisadores possuem uma visão mais abstrata sobre o con-

ceito do que é Contexto. Definindo-o de forma mais detalhada, Schilit (SCHILIT; ADAMS; WANT, 1994), Dey (DEY; ABOWD; WOOD, 1998) e Pascoe (PASCOE, 1998) afirmam que os aspectos importantes do contexto são: onde você está, com quem está, e quais são os recursos nas proximidades. Eles definem o contexto como sendo o ambiente de execução em constante mudança. Mais detalhadamente temos:

- *Contexto de Computação*: processadores disponíveis, dispositivos acessíveis para entradas do usuário e exibição, capacidade de rede, conectividade e custos de computação;
- *Contexto do Usuário*: localização, conjunto de pessoas próximas, e situação social;
- *Contexto Físico*: Iluminação, nível de ruído, condições de tráfego e temperatura;
- *Contexto de Época*: como o tempo de um dia, semana, mês e estação do ano;

Embora várias definições tenham sido apresentadas, podemos pensar em contexto como sendo qualquer fragmento de informação que possa ser utilizado para descrever uma situação e que possa influenciar diretamente o comportamento do sistema de computação que conseguiu identificá-lo. As pesquisas nessa área tem tido maior foco em informações implícitas, que não são apresentadas diretamente pelo usuário, mas são detectadas pelos diversos tipos de sensores presentes atualmente. Isso ocorre devido à crença na evolução das tecnologias e o conseqüente aumento do número de técnicas para captação e processamento das informações de contexto.

Ao observarmos como a utilização do contexto pode influenciar o comportamento das aplicações que são sensíveis a ele, identificamos os aspectos: *quem são*, *onde estão*, *quando estão* e quais são como requisitos fundamentais para determinar a situação atual do usuário. Dentro dessa condição é possível identificar certos tipos de contexto, destacando sua importância e ainda classificando-os: *localização*, *identidade*, *atividade* e *tempo*.

As categorias apresentadas por Schilit: *onde você está*, *quem você é*, *e que objetos estão ao seu redor*, consideram apenas a localização e a identidade de uma pessoa. Porém para identificar a condição (situação) do usuário não podemos deixar de lado informações como tempo e atividade. Como exemplo poderíamos considerar o *login* do usuário em um sistema qualquer; essa ação torna possível identificarmos a pessoa, a partir disso outras informações, ou pedaços desta, tornam-se presentes, como endereço de e-mail, lista de contatos, data de nascimento, entre outros.

Isso permite que fragmentos primários de contexto, relativos à identidade do usuário possam ser utilizados como índices para determinar um contexto secundário; esta caracterização

permite aos desenvolvedores e projetistas de aplicações sensíveis ao contexto escolher o contexto a ser usado para estruturar seus aplicativos.

2.2 Sensibilidade ao Contexto

Atualmente, várias definições sobre sensibilidade ao contexto podem ser encontradas, porém, nem todas apresentam o mesmo conceito sobre o assunto, tornando sua compreensão pouco exata.

Alguns autores como Schilit e Theimer definem sensibilidade ao contexto como aplicações que simplesmente recebem informações sobre o seu contexto atual e, a partir e de acordo com elas, se adaptam (SCHILIT; THEIMER, 1994).

No entanto para Hull (HULL; NEAVES; BEDFORD-ROBERTS, 1997) e Pascoe (PASCOE, 1998) (RYAN; PASCOE; MORSE, 1998) a definição de sensibilidade ao contexto está centrada na capacidade dos dispositivos de computação em sentir, interpretar e responder as variações do ambiente onde o usuário se encontra, bem como os dispositivos de computação presentes neste ambiente.

Na visão de Salber, sensibilidade ao contexto é a capacidade de fornecer o máximo da flexibilidade de um serviço de computação de acordo com a detecção e interpretação do contexto em tempo real (SALBER; DEY; ABOWD, 1998).

De acordo como Ryan, as aplicações sensíveis ao contexto são definidas como aplicações que monitoram a entrada de sensores ambientais e permitem aos usuários escolher entre uma grande variedade de contextos físicos e lógicos, de acordo com os seus interesses e atividades atuais (DEY; ABOWD, 1999). Definições mais peculiares (ABOWD et al., 1997b) (DAVIES et al., 1998) (DEY; ABOWD, 1997) (KORTUEM; SEGALL; BAUER, 1998) (WARD; JONES; HOPPER, 1997) caracterizam as aplicações sensíveis ao contexto como sendo aplicações que dinamicamente alteram ou adaptam o seu comportamento com base no seu contexto e no contexto do usuário.

Pela ótica de Brown, sensibilidade ao contexto pode ser definida como aplicações que automaticamente fornecem informações, ou efetuam ações de acordo com o contexto corrente do usuário detectado por sensores (BROWN, 1998). Fickas define aplicações de ambiente dirigido como aplicações que monitoram a mudanças no ambiente e adaptam o seu funcionamento de acordo com diretrizes pré-definidas ou definidas pelo usuário (FICKAS; KORTUEM; SEGALL, 1997).

A sensibilidade ao contexto também é definida através de sinônimos, como: adaptativa (BROWN, 1996); reativa (COOPERSTOCK et al., 1995); responsável (ELROD et al., 1993); situada (HULL; NEAVES; BEDFORD-ROBERTS, 1997); ciente do contexto (REKIMOTO; AYATSUKA; HAYASHI, 1998) e direcionada ao ambiente (FICKAS; KORTUEM; SEGALL, 1997).

2.2.1 Mecanismos para detecção de contexto - sensibilidade ao contexto

A detecção, manipulação e interpretação do contexto requerem técnicas especializadas para que aplicações possam utilizar o contexto como recurso para customização da interação com o usuário. Sendo assim, mecanismos precisam ser desenvolvidos para detectá-lo e entregá-lo a aplicação; nas próximas seções são apresentados os mecanismos que têm sido pesquisados, desenvolvidos e utilizados.

Sensibilidade à localização

A localização é um importante contexto; ela muda sempre que o usuário se move. Um sistema de rastreamento de localização confiável é fundamental para muitas aplicações sensíveis ao contexto. Quando pensamos na localização aberta (outdoor), ou seja, o usuário encontra-se em lugares como ruas e avenidas a escolha mais adequada é a utilização do GPS (Sistema de Posicionamento Global). Porém, o GPS não funciona em locais fechados, dentro de casa e apartamentos, visto que nestes lugares a força do seu sinal é baixa, não penetrando a maioria dos edifícios.

A construção de um sensor de localização ideal para interiores torna-se um problema desafiador. Pesquisadores da Olivetti desenvolveram um sistema de rastreamento baseado em sinais ultra-sônicos e de rádio, visando uma granularidade de localização de 15 centímetros (WARD; JONES; HOPPER, 1997). O sistema de suporte a localização Cricket do Laboratório do MIT (PRIYANTHA; CHAKRABORTY; BALAKRISHNAN, 2000) também tira proveito de ambos os sinais de ultra-som e rádio. Ao invés do sistema de rastreamento de localização do usuário, no entanto, cada dispositivo portátil determina a sua própria localização.

Existem outras abordagens (fora a voltada a celulares), como o sistema RADAR da Microsoft Research (BAHL; PADMANABHAN; BALACHANDRAN, 2000) que usa a força do sinal de RF na rede de comunicações como um indicador de distância entre um transmissor e um receptor, sem a criação de um sistema de rastreamento de localização adicional. A localização é determinada por uma consulta a um banco de dados central de intensidade do

sinal RF com um conjunto de receptores fixos, para as posições do transmissor conhecidas.

Sensibilidade à contextos de baixo nível

Existem outros tipos de contexto que não os locais. Eles têm sido amplamente estudados, dentre os mais importantes e suas abordagens de detecção, destacam-se:

- *Tempo*: pode ser obtido a partir do relógio interno do computador; como exemplo de utilização de tempo podemos citar o Active Badge (WANT et al., 1992), ParcTab (WANT et al., 1995) e Cyberguide (ABOWD et al., 1997a).

- *Objetos próximos*: leva em consideração os objetos ao redor e suas características, como recursos de memória e processamento; como exemplo podemos citar o sistema de teletransporte (Teleporting) (BENNETT et al., 1994) e o Pager (CHEN; KOTZ, 2000) sensível ao contexto.

- *Largura de banda da rede*: identificar os tipos de conexões disponíveis e por conseqüência as larguras de banda para comunicação, permite adaptarmos a execução de aplicativos em conformidade com ela; como exemplo podemos citar o Odyssey (NOBLE et al., 1997) que oferece chamadas a API através das quais aplicações podem ser notificadas quando a largura de banda rede sofrer mudanças. Os trabalhos mais recentes incluem o Gerente de Congestionamento (NOBLE et al., 1997), funcionando como um módulo interno do kernel que mede a largura de banda e notifica as aplicações através de upcalls.

- *Orientação*: como exemplo tem-se o MessagePad, que permite aos aplicativos ajustar a exibição quando alterações na orientação do dispositivo ocorrem.

- *Outros contextos de baixo nível*: temos a utilização de fotodiodos para detecção do nível de luz, acelerômetros para fornecer medições de inclinação e vibração, sensores de IR (infravermelho) para detecção da proximidade de seres humanos, microfone para detectar o som, e outros sensores para temperatura, pressão e gás CO.

Sensibilidade a contexto de alto nível

As informações de contexto de alto nível englobam situações como atividade atual do usuário. A detecção de contextos sociais complexos é um grande desafio; abordagens que contemplam essa possibilidade incluem a utilização da tecnologia de câmera e processamento de imagem - a consulta ao “calendário do usuário” para descoberta do que o usuário deve fazer

em determinada hora - técnicas de Inteligência Artificial para reconhecer contextos complexos através da combinação de vários sensores simples de baixo nível (GAEDKE et al., 1998).

Sensibilidade a mudanças no contexto

Monitores de fontes de contexto são utilizados, normalmente, com o objetivo de pesquisar e detectar o contexto atual, enviando as alterações para serviços de contexto, que, possuem uma interface de notificação do tipo publicação-assinatura; esta interface permite que clientes especifiquem seus interesses nas informações relativas às mudanças que acontecem no contexto, sendo o serviço de contexto responsável por entregar as mudanças para os clientes que também contribuem para que as mudanças ocorram.

2.2.2 Outras Considerações

Como visto anteriormente, a detecção, a manipulação e a interpretação do contexto requerem técnicas especializadas, para que aplicações possam utilizar o contexto como recurso para customização da interação com o usuário e com outros dispositivos. Sendo assim, metodologias precisam ser desenvolvidas para detectá-lo e interpretá-lo eficientemente.

2.3 Recursos para aplicações sensíveis ao contexto

As aplicações que recuperam informações manualmente para os usuários baseadas no contexto disponível são classificadas como *aplicações de seleção por proximidade*. Esta técnica de interação tem como objetivo apresentar uma lista de objetos ou locais, onde os itens mais relevantes no contexto do usuário são destacados, ou seja, mais fáceis de escolher (DEY; ABOWD, 1999).

As aplicações que recuperaram informações automaticamente para os usuários baseadas no contexto disponível são classificadas como *reconfiguração de contexto automática*. Esta técnica cria uma ligação automática com um recurso disponível baseada no contexto atual (DEY; ABOWD, 1999).

As aplicações que executam comandos manualmente para os usuários com base no contexto disponível são classificadas como *aplicações de comando do contexto*. Estes aplicativos são serviços executáveis disponibilizados de acordo com o contexto do usuário, sendo sua execução modificada com base neste contexto (DEY; ABOWD, 1999).

As aplicações que executam comandos para o usuário automaticamente com base no

contexto disponível usam *ações disparadas por contexto*. Estes aplicativos são serviços executados automaticamente quando uma transição específica de contexto acontece, e são baseados em simples regras SE-ENTÃO (IF-THEN) (DEY; ABOWD, 1999).

2.4 Desafios da computação sensível ao contexto

Mesmo com o grande de número de artigos publicados a partir do início da década de 90, a área de pesquisa sobre computação sensível ao contexto continua muito ativa, principalmente pelo fato de que muitas barreiras e desafios dela ainda precisam ser transpostos. Dentre eles, destacam-se:

- *Sensoriamento*: escolha e inclusão dinâmica dos contextos mais apropriados a cada aplicação; Técnicas para coleta de contextos físicos, lógicos e virtuais; Identificação e escolha de fontes de contextos;
- *Modelagem*: modelo de arquitetura para sistemas cientes de contexto; Modelo para representação uniforme da sintaxe dos dados de contexto coletados; Modelo de armazenamento de dados contextuais; Modelo de comunicação adotado entre diversos usuários ou aplicações;
- *Qualidade*: qualidade de contexto (QoC); Qualidade de serviço (QoS); Qualidade das fontes de contexto;
- *Segurança*: segurança para troca de dados entre usuários e aplicações; Confiabilidade das fontes de contextos;

Aliado aos desafios inerentes ao contexto tem-se a restrição de recursos disponíveis para execução das aplicações, fator importante a ser levando em conta, como pouca capacidade computacional, de energia e memória, gerando a necessidade de que os sistemas desse tipo realizem um gerenciamento de recursos consistente. Usando contexto de forma adequada é possível manter as aplicações conscientes dos recursos disponíveis e demandados.

2.5 Aplicações sensíveis ao contexto

O crescimento da computação móvel e o conseqüente aumento de usuários utilizando dispositivos portáteis tornam a criação de aplicações sensíveis ao contexto uma área promissora, podendo ser aplicada para suprir diversas necessidades, como a redução do consumo de

recursos de energia e de processamento. Entretanto, as informações que definem o contexto tornam-se heterogêneas de aplicação para aplicação, onde dados de contexto relevantes para um determinado aplicativo podem ser irrelevantes para outro. Dentro dessa ótica são apresentadas nas seções 2.5.1 à 2.5.7 várias aplicações que apresentam diversas finalidades, onde cada qual utiliza informações de contextos específicas para produzir a adaptação que lhe é necessária.

2.5.1 *Call Forwarding* (Encaminhamento de Chamadas)

Desenvolvido pelo grupo *Olivetti Research Ltd - ORL*, utiliza como contexto a localização do usuário; o contexto local é apresentado para a recepcionista, que rotineiramente encaminha chamadas para o telefone mais próximo do usuário de destino; a localização ajuda a encaminhar automaticamente as chamadas (WANT et al., 1992).

2.5.2 *Teleporting* (Teletransporte)

Desenvolvido pelo grupo *Olivetti Research Ltd - ORL*, utiliza como contexto a localização do usuário e a localização das estações de trabalho; é também conhecido como computação "*follow-me*/siga-me"; pode rastrear a localização do usuário para que a aplicação siga-o quando ele se mover (BENNETT et al., 1994).

2.5.3 *Shopping Assistant* (Assistente de Shopping)

Desenvolvido pela *AT T Bell Laboratories*, utiliza como contexto a localização do cliente dentro da loja, guiando-o através dela, fornecendo detalhes de itens, ajudando a localizá-los, apontando os que estão à venda e fazendo uma análise comparativa de preços (ASTHANA; CRAVATTS; KRZYZANOWSKI, 1994).

2.5.4 *Cyberguide*

Desenvolvido pelo grupo *Future Computing Enviroment (FCE) at the Georgia Institute of Technology*, utiliza como contexto o local do turismo e o tempo. Presta serviços de informação a um turista sobre a sua localização atual. Um diário de viagem é automaticamente compilado usando um histórico sobre viagens anteriores do turista; este diário é usado pelo sistema para fazer sugestões sobre os locais que ele tenha possível interesse em visitar. A informação sobre a localização é coletada por GPS e por um sistema de posicionamento de infravermelho (IR) desenvolvido (ABOWD et al., 1997a).

2.5.5 *Conference Assistant* (Assistente de Conferência)

Desenvolvido pelo grupo *Future Computing Environments (FCE) at the Georgia Institute of Technology*, utiliza como contexto a atividade atual (apresentação), a localização dos participantes, hora atual e calendário de apresentações. O assistente examina a programação da conferência, os temas das apresentações e a localização e interesses de pesquisa do usuário para sugerir as apresentações que atendam a ele. Sempre que o usuário entra em uma sala de apresentação, o Assistente de Conferência exibe automaticamente o nome do apresentador, o título da apresentação, e outras informações relacionadas (DEY et al., 1999).

2.5.6 *Fieldwork* (Trabalho de Campo)

Desenvolvido pelo grupo da *University of Kent at Canterbury*, utiliza como contexto a localização do usuário e o tempo atual. Fornece ferramentas para auxiliar na observação do pesquisador de campo e nas atividades de coleta de dados, ajudando o usuário no registro das informações sobre seu ambiente, através da coleta automática de algumas informações contextuais, assim como local e hora (RYAN; PASCOE; MORSE, 1998).

2.5.7 *Adaptative GSM phone and PDA* (Telefone GSM e PDA Adaptativo)

Desenvolvido pelo grupo *TEA (Technology for Enabling Awareness) at Starlab*, utiliza como contexto a atividade do usuário, o nível de luz, a pressão e a proximidade de outras pessoas. Funciona como um bloco de notas adaptado: utiliza letra grande quando o usuário está caminhando e fonte pequena quando para, se adaptando também a condições ambientais, como nível de luz. O telefone escolhe o toque e ajusta o volume ou entra em modo silencioso, levando em consideração estar na mão do usuário, sobre uma mesa, em uma mala, ou fora dela (SCHMIDT et al., 1999).

2.5.8 *Location-aware Information Delivery* (Entrega de Informação Sensível a Localização)

Desenvolvido pelo grupo MIT Media Laboratory, utiliza como contexto a localização do usuário e o tempo atual. O aplicativo aproveita tanto a localização quanto o contexto de tempo - uma mensagem de lembrete é criada com uma posição e, quando o destinatário chega a esse local, a mensagem é entregue através da síntese de voz sem a necessidade dele pegar o dispositivo e ler a mensagem na tela (MARMASSE; SCHMANDT, 2000).

2.6 *Middlewares*, infra-estruturas, *frameworks*, *toolkits* e bibliotecas sensíveis ao contexto

Atualmente, existe uma grande dificuldade em encontrar um suporte uniforme para criação, desenvolvimento e execução de aplicativos sensíveis ao contexto, levando diversos pesquisadores a criação de *frameworks*, *toolkits* e bibliotecas, já que são uma forma de abordagem interessante como suporte aos desenvolvedores de aplicações sensíveis ao contexto. As bibliotecas, *frameworks*, *toolkits*, *middlewares* ou infra-estruturas não são abordagens mutuamente exclusivas: havendo casos em que é útil ter todas elas (HONG, 2002).

Uma biblioteca é um conjunto generalizado de algoritmos relacionados. Exemplos incluem código para manipulação de strings e cálculos matemáticos complexos, tem seu foco exclusivamente em reutilização de código.

Os *frameworks* concentram-se mais sobre a reutilização de modelos através do fornecimento de uma estrutura básica; assumem as responsabilidades centrais em um aplicativo, mas apresentam formas de serem personalizados para necessidades específicas.

Os *toolkits* são construídos sobre os *frameworks*; o evento de uma interface gráfica para usuário seria um exemplo de um *framework* e um *toolkit* correspondente proporcionaria botões, *checkboxes*, e campos de entrada de texto para este *framework*.

As infra-estruturas de serviço e as tecnologias de *middleware* podem ser acessadas através de uma rede; seus formatos de dados e protocolos são de extrema importância porque permitem que os computadores que não sabem nada uns dos outros possam inter-operar (comunicar-se). Uma infra-estrutura exemplo é a própria Internet, e um exemplo de serviço oferecido por alguns computadores conectados à Internet é o DNSSistema de Nomes de Domínios.

Os *middlewares* fornecem a infra-estrutura básica de serviços que as aplicações precisam para executar; estes serviços vão desde os protocolos de baixo até o nível de aplicações do usuário.

2.6.1 PARCTAB - Sistema de Computação Móvel Sensível ao Contexto

O PARCTAB é um computador de mão sem fio que utiliza uma rede de celular baseada em infravermelho para comunicação. É usado como protótipo para o desenvolvimento de um sistema que analisa e reage a uma mudança de contexto do usuário. Nesse sistema de computação sensível, três aspectos de contexto são importantes: *onde você está*, *com quem está*, *e quais recursos estão próximos*. Para o PARCTAB, contexto abrange iluminação, nível

de ruído, conectividade de rede, os custos de comunicação, largura de banda da comunicação, e até a situação social. O sistema informa mudanças na localização dos aplicativos, fornecendo informações sobre a localização de um serviço público que coleta e redistribui informações sobre objetos e suas localizações (SCHILIT; THEIMER, 1994). A Tabela 2.1 mostra os recursos sensíveis ao contexto utilizados.

Tabela 2.1: Recursos do Sistema Sensível ao Contexto.

	Manual	**Automático**
Informação	Seleção de Proximidade	Reconfiguração Automática de Contexto
Comando	Comandos de Contexto	Ações Disparadas de Contexto

Fonte: (SCHILIT; ADAMS; WANT, 1994)

Seleção por proximidade

A seleção por proximidade é utilizada como técnica de interface com usuário - nela objetos localizados próximos são destacados. Há pelo menos três tipos de objetos localizados: a entrada e saída dos dispositivos do computador, como impressoras, monitores, alto-falantes e câmeras de vídeo; o conjunto de objetos que você já está interagindo, e que precisam ser tratados por um processo de software, como pessoas em uma mesma sala - e o conjunto de lugares que se quer obter mais informações: restaurantes, boates, postos de gasolina e lojas, ou mais genericamente, saídas e entradas destes lugares. Informações de localização podem ser usadas, por exemplo, como peso na escolha de impressoras que estão mais próximas. Outro fator que as interfaces de seleção por proximidade devem levar em conta são os requisitos de largura de banda.

Reconfiguração automática do contexto

A reconfiguração automática do contexto é utilizada para adicionar componentes novos, remover os existentes, ou alterar as conexões entre eles. Componentes podem ser definidos como servidores e canais de comunicação para os clientes. A Reconfiguração pode ser baseada também, além da localização, nas pessoas presentes em uma sala. A Reconfiguração de Contexto pode ainda incluir funções do sistema operacional: por exemplo, deixar o *circuito de armazenamento ativo* quando o celular estiver conectado a energia elétrica, utilizar a memória dos computadores próximos ociosos para armazenamento de backup, ao invés de mover para um disco local ou remoto.

Comandos de contexto

Os comandos de contexto são utilizados para prever as ações do usuário em uma dada situação, com o objetivo de explorar esse fato. Consultas a informações do contexto podem

produzir resultados diferentes baseadas no contexto onde estão inseridas. Desta mesma forma, o contexto pode adaptar os *comandos de contexto*. Por exemplo, o comando de impressão pode, por padrão, ser enviado para a impressora mais próxima.

Ações disparadas de contexto

As ações disparadas de contexto são implementadas como simples regras *IF-THEN*, utilizadas para especificar como os sistemas sensíveis ao contexto, devem se adaptar. No PARCTAB duas aplicações de ação disparada de contexto, *Active badge* - um marcador que transmite periodicamente um identificador exclusivo com a finalidade de determinar a localização do utilizador - e o tab baseado em *Lembretes de Contexto* foram experimentados. O *bagde* e o *location* são strings que correspondem ao portador do crachá e ao local de observação. Quando um evento correspondente ocorre, invoca uma ação. Por exemplo:

Coffe Kitchen arriving play -v 50 /sounds/rooster.au

O exemplo monitora o *badge* "Coffe", que está ligado à máquina de café na cozinha e reproduz o som do galo sempre que alguém faz café.

Considerações Finais

O PARCTAB foi um dos primeiros sistemas de computação móvel construídos para explorar softwares sensíveis ao contexto. Para construção da sua arquitetura foram utilizados sensores de infravermelho que permitiram a obtenção das informações contidas no ambiente onde o usuário se encontra; essa estrutura possibilitou a criação de aplicações que utilizavam como recurso adicional a sensibilidade ao contexto. Embora a tecnologia utilizada, seja limitada, como por exemplo, em relação ao seu alcance, foi possível pela primeira vez constatar de forma prática os principais benefícios trazidos pela utilização de sensibilidade ao contexto no dia-a-dia dos usuários.

2.6.2 *Toolkit* de contexto

O *Toolkit* de Contexto (SALBER; DEY; ABOWD, 1999) foi desenvolvido com objetivo de adaptar adequadamente os variados serviços disponíveis a partir do aumento da utilização de aparelhos portáteis interagindo com ambientes de computação avançados, criando a necessidade de construir uma maneira de adaptar adequadamente estes serviços, que levam em consideração o contexto do usuário, com objetivo de dar um melhor suporte a interação homem-máquina.

O *toolkit* foi desenvolvido utilizando-se da tecnologia Java e é composto por três abstrações básicas: *Widgets*, *Agregattors* e *Interpreters*.

Widgets

Os *Widgets* de contexto têm como principal objetivo efetuar a mediação entre um usuário e uma aplicação - a mediação entre um usuário e o ambiente a sua volta (SALBER; DEY; ABOWD, 1999). Eles encapsulam as informações sobre uma única parte do contexto, tais como localização ou atividade. Eles fornecem uma interface para os componentes ou aplicativos que usam o contexto, escondendo os detalhes básicos dos mecanismos de sensibilidade.

Agregattors

Os *Agregattors* ou agregadores de contexto podem ser pensados como meta-widgets, assumindo todas as capacidades de widgets, além de fornecer a capacidade de agregar informações de contexto, como usuários ou lugares. Age como um *gateway* entre aplicações e *widgets* elementares, escondendo toda parte complexa sobre mecanismos de sensibilidade ao contexto.

Interpreters

Os *Interpreters* ou interpretadores de contexto são usados para abstrair ou interpretar a informação de contexto de baixo nível até informação de alto nível. Por exemplo, identidade, localização e informação de nível de som podem ser usadas para interpretar que uma reunião está ocorrendo.

O *toolkit* torna a natureza distribuída do contexto transparente para as aplicações sensíveis a ele, fazendo com elas não necessitem mais saber se os componentes do contexto estão sendo executado localmente ou remotamente. Todos os componentes compartilham um mecanismo de comunicações comum (XML e HTTP) que suporta a distribuição transparente. Estes componentes são executados independentemente de qualquer aplicação, permitindo que eles sejam utilizados por múltiplos aplicativos. A figura 2.1 mostra a configuração dos componentes do contexto.

O *toolkit* permite aos desenvolvedores adicionar contexto a aplicações que não são sensíveis a ele e aumentar seu uso em aplicações que já são sensíveis ao contexto.

2.6.3 Infra-estrutura de Computação Sensível ao Contexto

Uma abordagem genérica de infra-estrutura, podendo ser utilizada em diversas aplicações, permite simplificar as tarefas de criação e manutenção de sistemas sensíveis ao contexto. Com ela é possível retirar a responsabilidade de obtenção e processamento dos dados de contexto do

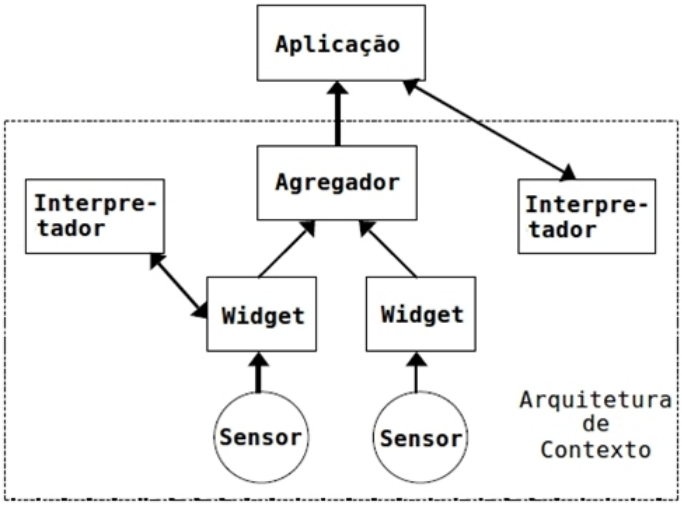

Figura 2.1: Configuração simples dos componentes do contexto. As setas indicam o fluxo de dados.

Fonte: (SALBER; DEY; ABOWD, 1999)

dispositivo e deixá-los sobre a responsabilidade de uma infra-estrutura de *middleware*, tornando mais fácil o incremento gradativo de novos sensores, novos dispositivos, e os novos serviços, possibilitando aos sensores e dispositivos compartilhar os dados brutos e de contexto, colocando a cargo da infra-estrutura a aquisição, processamento e interoperabilidade (HONG; LANDAY, 2001).

Vantagens de uma abordagem de infra-estrutura

Colocando o máximo de aquisição e processamento de contexto dentro de uma infra-estrutura, através de serviços que podem ser acessados por qualquer dispositivo e em qualquer aplicativo, é possível obter uma série de benefícios, dentre os mais interessantes podemos citar:

- *Independência de Hardware, Sistema Operacional e Linguagem de Programação*: uma infra-estrutura de serviço pode ser usada independentemente da plataforma de hardware, sistema operacional e linguagem de programação - com a utilização de um formato de dados e um protocolo de rede padrão é possível suportar um maior número de dispositivos e aplicações. Esta abordagem torna a infra-estrutura mais fácil para receber novos sensores, dispositivos, sistemas operacionais, e linguagens de programação.

- *Fornecimento de Credenciais para Manutenção e Evolução*: sensores, serviços e dispositivos em uma infra-estrutura podem ser alterados de forma independente e dinâmica, sem afetar qualquer outra coisa. O resultado final é que todo o sistema pode ser gradualmente incrementado com novos sensores, serviços, dispositivos e aplicativos. Para adicionar um

novo sensor, basta um software que conecte o sensor com o restante do middleware. Para adicionar um novo serviço, basta um espaço no setor de middleware, onde os serviços podem ser enviados, descobertos, e depois executados quando necessário. Para adicionar um novo dispositivo, basta um software que entenda os protocolos e formatos de dados utilizados pela infra-estrutura.

- *Compartilhamento de Sensores, Processamento, Dados e Serviços*: dispositivos e aplicações sensíveis ao contexto tornam-se mais fáceis de serem desenvolvidas e implantadas, visto que os sensores, o processamento, os dados e serviços dentro da infra-estrutura podem ser compartilhados. Dispositivos individuais não precisam realizar a concepção de todos os tipos de sensores para adquirirem informações, ao invés disso, esta função pode ser delegada a infra-estrutura, que encontrará os sensores adequados nas proximidades. Pelo compartilhamento do poder de processamento, não há mais a necessidade de termos dispositivos poderosos, caros e que possuam microprocessadores que consumam muita energia. Compartilhando serviços, os aplicativos podem ser menores, o que torna seu armazenamento mais fácil em dispositivos portáteis, ao invés de aplicações monolíticas e auto-suficientes - temos um conjunto de funcionalidades de uma aplicação na forma de muitos pequenos serviços existentes na infra-estrutura onde as aplicações podem simplesmente invocá-los.

Formato de dados e protocolo padrão

A criação dos formatos de dados e protocolos utilizados em uma infra-estrutura consiste no primeiro desafio para sua construção. Esses padrões permitem que todas as partes que compõem a infra-estrutura possam se comunicar; eles precisam ser simples o suficiente para que possam ser aplicados em praticamente todos os dispositivos e possam ser utilizados por qualquer aplicação.

Os formatos de dados precisam lidar com o fato dos dados do sensor serem, em muitos casos, parciais e pouco confiáveis, o que leva à ambigüidade na forma como o contexto é interpretado. Existe a necessidade da distinção entre dados brutos, vindos do sensor, e os dados interpretados ou de contexto. Dados do Sensor possuem vários atributos, incluindo precisão, granularidade e exatidão, estes atributos afetam o modo como são interpretados nos contextos de maior nível.

Criação automática de caminho

A criação automática de caminho pode ser adaptada para sensibilidade ao contexto com intuito de simplificar a tarefa de aprimorar e transformar os dados brutos do sensor em

dados do contexto de maior nível, visto que pesquisas recentes sobre essa técnica (KICIMAN; FOX, 2000) (MAO; KATZ; BREWER, 2001) possuem seu foco centrado no protocolo de rede e no formato de dados como forma de interoperabilidade.

A criação automática de caminho depende de operadores que são um subconjunto especial de serviços. A figura 2.2 apresenta quatro tipos de operadores, o primeiro operador: *GPS* à *CEP*, o segundo: *localização do celular* à *GPS*, o terceiro: *localização do celular* à *CEP* e o quarto: *CEP* às *condições meteorológicas atuais para essa área*.

Se utilizados de forma individual, nenhuns destes serviços são muito interessantes, no entanto, aplicando o conceito de *pipes* do UNIX, eles podem ser encadeados para formar serviços mais robustos. Um exemplo, seria utilizar o sitema de posicionamento global para obter a localização geográfica de um usuário e através da mesma descobrir as condições meteorológicas locais, encadeando para tal os operadores *GPS* à *CEP* e *CEP* às *condições meteorológicas atuais*.

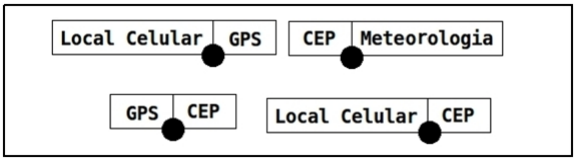

Figura 2.2: Operadores, tipo especial de serviço que reside na infra-estrutura, oferecendo conversões de dados. Seu benefício encontra-se no fato de poderem ser compostos em serviços mais poderosos.

Fonte: (HONG; LANDAY, 2001)

A Criação Automática de Caminho retira dos desenvolvedores de aplicativos a responsabilidade de saber sobre sensores e serviços específicos, em vez disso, eles precisam apenas se preocupar com a formulação da consulta de contexto correta.

Contudo, alguns desafios para a construção de um serviço de Criação Automática de Caminho são: o desenvolvimento de *tipos de dados padronizados*, caso contrário, os operadores não podem ser conectados uns com os outros; *a construção de bons caminhos* - o sistema precisa de uma maneira para selecionar um caminho, caso haja vários caminhos válidos; a representação da *consulta de contexto* (*query*) - deve ser rica o suficiente para obter dados importantes sobre o contexto e ao mesmo tempo ser simples para que possam ser compreendidas e conseqüentemente interpretadas.

Descoberta por proximidades

O serviço de *Descoberta Baseada na Proximidade* tem como objetivo encontrar todos

os sensores na proximidade. Supondo um serviço de captura que deseja saber se há uma reunião acontecendo em uma determinada sala, em vez de fixar serviços para usar sensores específicos naquela sala, ele pode pedir a infra-estrutura para localizar os sensores e em seguida, usar a *Criação Automática de Caminho* para ligar os dados de baixo nível dos sensores com as questões de alto nível: "*Há uma reunião agora?*". A combinação da Descoberta Baseada na Proximidade com a *Criação Automática de Caminho* facilita o projeto e a implementação de aplicações sensíveis ao contexto.

Contudo ainda existem desafios a serem superados para construção de um serviço de *Descoberta por Proximidade*, dentre eles: a *logística*, visto que nem todos os sensores são capazes de identificar sua localização, tendo que ser configurados manualmente; *a representação da localização* deve ser flexível o suficiente para que as consultas de alto nível possam ser construídas.

Considerações Finais

Existem várias vantagens na utilização de uma abordagem de infra-estrutura de serviços: a infra-estrutura poder ser neutra em relação à plataforma de hardware, sistema operacional e linguagem de programação, permitindo que uma maior variedade de dispositivos e aplicativos possa acessá-la; a camada de *middleware* separa as partes individuais da infra-estrutura de outras partes, permitindo que os sensores e serviços sejam atualizados de forma independente e dinamicamente um do outro enquanto o sistema ainda está em execução; os dispositivos podem ser mais simples, visto que eles podem ser baseados no uso de sensores, processamento de energia, serviços e dados contidos na infra-estrutura.

Alguns desafios ainda precisam ser encarados: projetar os formatos de dados e protocolos de rede; construir os serviços básicos de infra-estrutura, que incluem a *criação automática de caminho* e a *descoberta baseada na proximidade*; encontrar um meio termo entre dispositivos inteligentes e infra-estruturas inteligentes; delimitar o escopo dos dados do sensor e dos dados do contexto para garantir segurança e privacidade; construir uma infra-estrutura que seja ampliada para um grande número de sensores, serviços, equipamentos e pessoas.

2.6.4 *ContextPhone* - Plataforma para aplicações móveis sensíveis ao contexto

O *ContextPhone* teve seu desenvolvimento baseado na abordagem iterativa, com foco na interação humano-computador, o que beneficia os desenvolvedores na criação de aplicativos que integram-se com as tecnologias existentes e também com as vidas diárias dos usuários. O

ContextPhone é uma plataforma de software que consiste em módulos interligados fornecidos como um conjunto de bibliotecas C++, é executado em telefones móveis (celulares) usando Symbian OS e a plataforma de *smarthPhones* Nokia Series 60 (RAENTO et al., 2005).

O *ContextPhone* tem como principais objetivos: o *fornecimento do contexto como um recurso*, representando o contexto detectado, de forma que os seres humanos possam entendê-lo tornando-se com isso um recurso para a interação social e não apenas para a adaptação da máquina - a incorporação das aplicações existentes, suas interfaces integram-se com as aplicações já existentes do *smartphone*: mensagens e funções de chamadas - *o fornecimento de rápida interação e discrição*, permitindo interação rápida com execução em segundo plano, não interferindo na utilização de outros aplicativos - *a Garantia de robustez*, recuperando automaticamente a conectividade perdida - *o fornecimento de aberturas para controle pelos usuários*, responsabilizar os usuários para compensar e controlar emendas em muitas aplicações.

Como objetivo final tem-se a *possibilidade do rápido desenvolvimento* de aplicações sensíveis ao contexto, os desenvolvedores tornam-se capazes de adicionar fontes de dados de contexto, sensores, e camadas externas, construindo novas aplicações sem a necessidade de reconstruir todo o sistema.

Componentes do ContextPhone

A plataforma pode ser divida em quatro módulos: os sensores, que são responsáveis pela obtenção dos dados do contexto de diferentes fontes; as comunicações, que conectam os serviços externos através de protocolos padrões de Internet; *as aplicações customizáveis*, que podem aumentar ou substituir aplicativos embarcados mais facilmente e os serviços do sistema, que automaticamente lançam serviços em background.

- *Sensores*: são suportados quatro tipos de sensores, para *localização*, é utilizado um receptor GPS - para *interação* com o *usuário*, são utilizados, o estado ocioso/ativo, o perfil de alarme do telefone, o estado do carregador e captura de mídia para o *comportamento da comunicação*, são utilizadas chamadas, gravações de chamada, SMS enviado e recebido e conteúdo do SMS - para o *ambiente físico*, são utilizados dispositivos *bluetooth* ao redor, redes *bluetooth* disponíveis e o reconhecimento óptico (usando câmera embutida).

- *Comunicações*: para coletar os dados tem-se o *ContextoLoger* e para o compartilhamento de mídia o *ContextMedia*. O *ContextPhone* pode, ainda, enviar e receber SMS e MMS, os desenvolvedores podem incorporar qualquer serviço móvel que possa ser usado através de mensagens de texto, como serviços de localização de rede.

- *Aplicações Customizáveis*: os desenvolvedores podem utilizar as versões customizáveis

de aplicações embarcadas do ContextPhone, elas também suportam amplo registro de interação com o usuário, permitindo que desenvolvedores e pesquisadores estudem padrões de uso.

- *Serviços do Sistema*: a Série 60 da Nokia não oferece inicialização automática de aplicativos, serviços do *ContextPhone* adicionam esse recurso.

ContextLogger e ContextMedia

O *ContextLogger* tem como objetivo coletar e gravar os dados de mobilidade. Ele recebe dos sensores e aplicações customizáveis, notificações relativas às mudanças de contexto, gravando esses dados em um arquivo local e, periodicamente, transferindo esses arquivos para o servidor dos pesquisadores, tudo isso em background.

O *ContextLogger* tem sido usado para descobrir correlações entre os dados do contexto e a disponibilidade de um usuário. O Centro de Pesquisa Técnica da Finlândia (VTT) o tem usando para coletar dados sobre as proximidades de dispositivos *Bluetooth* com intuito de modelar os padrões recorrentes de interações entre pessoas em um grupo de trabalho. O projeto de Mineração da Realidade do Laboratório do MIT coleta dados de comunicação e proximidade dos estudantes para modelar a dinâmica da rede social.

O *ContextMedia* utiliza a idéia de mídia locativa, ou seja, anexar uma mídia a uma localização física. Para isso são utilizados vários sensores, que anotam a mídia e noticiam a captura, além de arquivos para transferência em segundo plano e compartilhamento da mídia anotada. Os usuários podem usar o canal de mensagens *Jabber* para notificar outros usuários sobre o compartilhamento em tempo real.

2.6.5 *Middleware* para reconfiguração automática e dinâmica de serviços dirigidos a contexto

Atualmente as soluções de *middleware* são enriquecidas com recursos dedicados explorados por serviços e usuários, no entanto o aumento do número de funcionalidades e a conseqüente necessidade de interagirem uns com os outros aumenta a complexidade do *middleware*, tornando esta abordagem inadequada. Uma forma viável para lidar e tratar a crescente complexidade, seria simplificar o projeto do *middleware*, mantendo nele apenas o núcleo de gerenciamento e coordenação das funções, movendo para fora da camada de middleware a lógica característica da ubiquidade (BOARI et al., 2008).

Usuários

Os usuários acessam serviços e conteúdo através de diferentes dispositivos hardware e interfaces de software. Pensando em uma plataforma que seja verdadeiramente ubíqua, é preciso englobar e servir todo tipo de dispositivo e/ou interface do usuário.

Serviços

Os serviços apresentam-se como qualquer tipo de ação efetuada sobre um conteúdo: da geração a adaptação dele, até o enriquecimento para a entrega.

Middleware

O *middleware* tem por objetivo fornecer componentes que modelem e percebam as complexas interações do usuário, vindas da ubiquidade e da mobilidade, dando suporte a eles para compor e ativar serviços dinamicamente.

Arquitetura do Middleware

A arquitetura permite o fornecimento de serviços através da sua descrição e publicação. Estes serviços são compostos em entidades de fluxo de trabalho que mapeiam as necessidades dos utilizadores, utilizando, completamente, à idéia de uma arquitetura orientada a serviços (BOARI et al., 2008).

Os fluxos de trabalho (*workflows*) são objetos autônomos criados com base no perfil do usuário, responsáveis pela composição de serviços. Um fluxo de trabalho consiste tanto nos serviços que executam operações em dados, quanto na lógica de controle para coordená-los. Sua composição inclui os metadados que descrevem o comportamento dos serviços, bem como seus formatos de entrada e de saída, a composição dinâmica que ajusta a composição que melhor atenda às necessidades do usuário sobre os serviços de sua preferência, o dispositivo que está sendo usado, o status da rede, entre outros, e a compartilhabilidade, permitindo que fluxos de trabalho possam ser compartilhados entre os usuários com necessidades semelhantes.

Os *proxies* mantém a sessão do usuário e informações do contexto. Invocando recursos do middleware, são entidades utilizadas para lidar com o contexto do usuário e informações de sessão. É composto pela lógica de gerenciamento de sessão, um aplicativo de streaming de multimídia direcionado ao *proxy*, que em seguida, redireciona esse conteúdo para o usuário real, pela lógica de gerenciamento do contexto, que manuseia o contexto do usuário, tanto pelo seu estado atual quanto pela reação imediata a mudanças bruscas e pela lógica de (re)configuração do fluxo de trabalho, onde o *proxy* determina os fluxos de trabalho dos serviços requisitados pelos usuários de acordo com a especificação feita pelo usuário através de um serviço de assinatura (registro de eventos).

O *kernel*, como é comum em algumas plataformas de *middleware*, fornece todos os serviços de propósito geral. Para tal, é composto por um repositório de serviço, onde implementações de serviços são disponibilizadas pela sua submissão para o repositório de serviços; um *Broker*, onde as solicitações do usuário são dissociadas das suas respectivas respostas, uma engine de construção de *workflow*, que processa as composições de serviço recebidas a partir do *proxy* do usuário, retornando suas implementações reais, ou seja, os fluxos de trabalho correspondentes e uma engine de execução do *workflow*, que tem como função inicializar o processo de *workflow*, para monitorá-lo e ser notificado quando erros de condição e execução ocorrerem.

Caso de Uso - Middleware

Os avanços da tecnologia permitem ao usuário acessar a Internet por meio de seu *smartphone* pessoal, explorando uma conexão GPRS lenta ou uma *Wi-Fi* rápida. Considerando um aluno lendo as páginas do portal do campus que oferece a ele, além disso, um serviço de notícias. O *middleware* de reconfiguração automática fornece serviços de configuração para usuários através da Web, permitindo que eles expressem suas preferências, como por exemplo:

- receber o título das notícias através de mensagens SMS em seu telefone assim que a notícia tornar-se disponível;
- ter relatórios diários por e-mail de todo o conteúdo das notícias do dia;
- quanto à navegação na web: ter imagens removidas e páginas redimensionadas quando a navegação for através de uma conexão GPRS; ter páginas redimensionadas quando navegam pela infra-estrutura sem fio do campus.

Comportamento do Middleware - Proxy

Ao receber qualquer nova ou alterada requisição do usuário, o proxy deve calcular uma descrição adequada das composições de serviço que implantam a requisição. Para tal são necessários os seguintes passos:

- *Resolução de Serviço*. O *proxy* obtém as instâncias dos serviços que são necessários para ajustar a composição do usuário.
- *Criação de fluxo de trabalho*. Ao lançar o motor (*engine*) de construção de fluxo de trabalho, o *proxy* organiza os serviços em conjunto e inicializa-os. Os Fluxos de trabalho que satisfazem as necessidades do usuário corrente são ativados e configurados imediatamente. O middleware de reconfiguração não reserva recursos até que eles sejam

necessários. Para isso, o *proxy* do usuário escuta e reage às mudanças no contexto do usuário. Composições de serviço são instanciadas quando o contexto do usuário mostra que há uma pré-necessidade.

Adaptação do Conteúdo Web

A solicitação HTTP é interceptada, coletando os pedidos que chegam do usuário. Cada pedido é, então, enviado para o gerente de interceptação para pré-processamento. Isso extrai a identidade do usuário, e finalmente, associa a identidade do mesmo com o comando, para tornar as informações de contexto de usuário e de sessão disponíveis.

Para lidar com as preferências do usuário, o comando implica a execução do fluxo de trabalho: um servidor *proxy* web consulta o servidor web real indicado na URL do pedido e recebe a página HTML correspondente; no final redimensiona a página para se adaptar a exibição no *smartphone*, cuja resolução é parte do atual contexto do usuário.

Reconfiguração Dinâmica do Fluxo de Trabalho

O estado da conexão do usuário fornece informação sobre a largura de banda disponível, assim como a taxa de transferência média, o seu endereço IP corrente, a presença de firewalls ou NATs e o mais importante, a tecnologia de rede que está sendo explorada. O *middleware* interpreta cada pacote UDP e assim consegue manter as informações sobre o tipo de conexão do contexto atual do usuário atualizadas. Quando o usuário entra em uma zona Wi-Fi, seu proxy garante a manutenção da sessão e o contexto conhecido para o *middleware* é alterado em conformidade. Isso altera o fluxo de trabalho, satisfazendo as novas preferências válidas (ou seja, sem a remoção da imagem), isso é feito de forma pró-ativa: o *proxy* não espera por um novo pedido para reorganizar a composição do serviço.

3 Redes Tolerantes a Atrasos e Desconexões

As duas principais características das DTNs são os atrasos e às desconexões. O atraso fim-a-fim é determinado pelo tempo de espera de cada nó, pelo atraso nas filas e pelo atraso nas transmissões das mensagens. Com relação às desconexões, elas podem ocorrer pela alta mobilidade que provoca constantes mudanças na topologia da rede, por condições adversas de comunicação e por economia de recursos como em sensores sem fio que dormem para poupar energia.

Para contornar os problemas de atrasos e desconexões as DTNs se servem das técnicas de comutação de mensagens e armazenamento persistente dos dados (OLIVEIRA et al., 2008). Quando uma mensagem precisa ser enviada ela é armazenada e encaminhada nó a nó desde a origem até o destino; por utilizar esse conceito, as DTNs são conhecidas como redes do tipo armazena-encaminha (*store-and-forward*) , ou seja, em primeiro momento a mensagem é recebida integralmente e armazenada para, num segundo momento, ser enviada ao próximo nó, podendo este ser ou não o destino da mensagem. Não há a necessidade de o destino estar ativo quando a origem enviar a mensagem.

3.1 A arquitetura DTN

Sendo necessária dentro de DNTs a utilização de comutação de mensagens e armazenamento persistente, foi adotada como solução, pelo grupo de pesquisa em DTN (*DTN Research Group* - DTNRG) a criação de uma camada de agregação (overlay) situada acima da camada TCP e abaixo da camada de Aplicação, que também permite a interoperabilidade entre redes convencionais e DTNs (FALL, 2003).

A arquitetura em *overlay* permite tornar a DTN totalmente independente das diversas redes regionais, permitindo que as aplicações se comuniquem através de múltiplas regiões. Para garantir interoperabilidade com qualquer tipo de rede, a camada de agregação é situada acima da camada de transporte das redes que se servem do perfil de protocolos TCP/IP. As camadas

abaixo da camada de agregação são definidas de acordo com a conveniência do ambiente e comunicação de cada região, podendo ser específicas para cada região englobada pela DTN. A figura 2.3 apresenta o formato da arquitetura DTN.

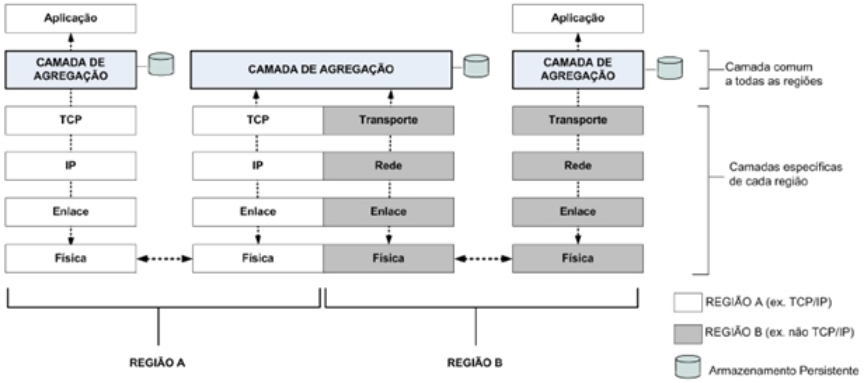

Figura 3.1: Arquiteutra DTN.

Fonte: (FALL, 2003)

3.1.1 Desafios - Redes DTN

O sistema de entrega de mensagens em DTNs é do tipo assíncrono para suportar atrasos e desconexões. Para as aplicações DTNs, a entrega de mensagens é mais importante que qualquer outra métrica de desempenho, inclusive o atraso. Sendo assim, o primeiro grande desafio dentro de redes DTNs está no roteamento das mensagens, visto a necessidade do projeto de protocolos capazes de superar os problemas de freqüentes desconexões. Outros desafios também encontrados relacionados ao ambiente de redes DTNs são os longos atrasos (variam de horas a dias), conectividade intermitente, recursos limitados de comunicação e alta taxa de erros.

3.1.2 Transmissão de Mensagens

Quando uma mensagem precisa ser enviada ela é armazenada e encaminhada nó a nó desde a origem até seu destino, por esse motivo redes DTNs também são conhecidas como redes do tipo armazena-encaminha. Uma mensagem recebida, primeiramente é armazenada para em seguida ser enviada ao próximo nó.

As aplicações DTNs enviam mensagens de tamanhos variáveis chamadas de unidades de dados da aplicação (Application Data Units - ADUs). As mensagens são transformadas pela camada de agregação em uma ou mais unidades de dados de protocolo denominados agregados,

que são armazenados e encaminhados pelos nós DTN.

Contatos - tipos de contatos

Um contato corresponde a uma ocasião favorável para os nós trocarem dados. A arquitetura DTN classifica os contatos em: persistentes, sob demanda, programado, oportunista e previsível. Para qualquer um desses tipos de contatos sempre existem probabilidades de falhas.

Os *contatos persistentes* são aqueles que se encontram sempre disponíveis. Os contatos sob demanda requerem alguma ação para que sejam instanciados, uma vez acionados, funcionam como contatos persistentes até serem encerrados.

Os *contatos programados* funcionam através de um agendamento, pré-estabelecido, entre dois ou mais, antes que ocorra a troca de informações. O horário de duração também é estabelecido previamente. Uma exigência das redes de contatos programados é a sincronização do tempo entre os nós que compõem esta rede para que a troca de informações seja realizada com sucesso. Contatos programados podem ser vistos em uma rede de sensores onde determinados nós "acordam" em horários pré-estabelecidos, voltando a "dormir" para poupar energia, fora dos horários pré-estabelecidos.

Os *contatos previsíveis* são aqueles onde é possível fazer previsões sobre o horário e duração dos contatos com base em históricos de contatos previamente realizados. Em contrapartida aos contatos programados, os previsíveis possuem certo grau de incerteza sobre sua ocorrência.

Os *contatos oportunistas* tiram proveito de encontros que não foram previamente programados entre nós, tendo como objetivo obter vantagem de contatos realizados totalmente ao acaso para realizar a comunicação com qualquer nó fora do alcance de um nó fonte.

Transferência de Dados entre nós DTNs

As redes DTNs utilizam-se do protocolo de agregação em conjunto com os protocolos que operam nas camadas abaixo da camada de agregação para retransmissão nó a nó em casos de perdas ou dados corrompidos. Entretanto, como os protocolos que operam abaixo da camada de agregação não são executados de modo fim-a-fim nas redes tolerantes a atraso e desconexão, os mecanismos que provêem confiabilidade fim-a-fim só podem ser implementados na camada de agregação (WARTHMAN, 2003).

A camada de agregação suporta a retransmissão nó a nó através do mecanismo denominado Transferência de Custódia (TC), que tem como objetivo passar a responsabilidade da entrega de uma mensagem de um nó para outro nó. Para realização da transferência de custódia,

um temporizador é utilizado pela camada de agregação em conjunto com retransmissões na implementação de um mecanismo de reconhecimento custódia-a-custódia.

Quando um nó DTN envia uma mensagem ao próximo nó, ele efetua uma solicitação de transferência de custódia e inicia um temporizador de retransmissão. Se a camada de agregação do próximo salto aceitar essa solicitação é retornado um reconhecimento (mensagem *ACK*) para o nó que enviou a mensagem com o pedido de transferência de custódia. Contudo se o reconhecimento (*ACK*) não for retornado antes de o temporizador expirar, o nó emissor reenvia o agregado. A figura 2.4 apresenta o mecanismo de transferência de custódia.

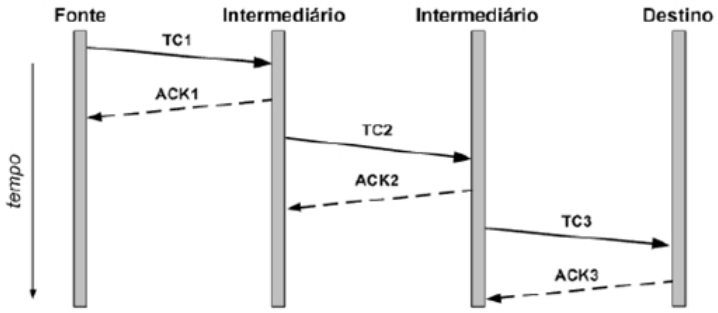

Figura 3.2: Transferência de Custódia (TC).

Fonte: (OLIVEIRA, 2008a)

A arquitetura DTN não exige que todos os nós DTN aceitem a transferência de custódia (OLIVEIRA, 2008a), pode ser possível que um nó tenha capacidade de armazenamento suficiente para agir como custódio, mas escolha não aceitar um pedido de transferência caso sua capacidade de bateria esteja abaixo de um determinado limiar.

Em DTNs, um dos recursos mais disputados é o acesso ao armazenamento em cada nó, ao contrário de muitas outras redes onde as mensagens são simplesmente descartadas no momento em que a memória acaba. Um custódio só pode apagar um agregado em duas situações: quando transfere o agregado a outro custódio ou se o tempo de vida do agregado expirar.

Classes de Prioridades

A arquitetura DTN define três classes de prioridades: baixa (*bulk*), normal (*normal*) e expressa (*expedited*). A classe baixa possui a menor prioridade; nenhum agregado desta classe é transportado até que todos os agregados das outras duas classes tenham sido transmitidos. Os agregados da classe normal são transportados antes dos agregados da classe baixa, e os agregados da classe expressa são transmitidos com prioridade sobre as outras duas classes.

Opções de entrega

A arquitetura DTN define oito opções de entrega dos agregados, determinadas pela aplicação. No momento do envio de uma unidade de dados a aplicação pode requisitar qualquer combinação das opções de entrega disponíveis. A informação sobre as opções requisitadas pela aplicação é levada juntamente com cada agregado produzido pela camada de agregação. Abaixo são apresentadas as oitos opções de entrega disponíveis na arquitetura DTN:

- *pedido de transferência de custódia*: solicitação para que um agregado seja entregue utilizando procedimentos de transferência de custódia;
- *pedido de aceitação de custódia pelo nó fonte*: provê uma forma da aplicação exigir que o nó DTN fonte aceite a custódia dos agregados enviados (por exemplo, armazenamento de forma persistente dos agregados);
- *notificação de entrega dos agregados*: conhecida como aviso de recebimento (return receipt). Um aviso único enviado pelo nó destino para os nós que participam do encaminhamento do agregado, podendo chegar até o nó fonte (WARTHMAN, 2003);
- *notificação de reconhecimento positivo do agregado pela aplicação*: solicitação de um relatório sobre o estado do reconhecimento, similar ao relatório do estado de entrega do agregado, porém, é gerado pela camada de aplicação do destino e não pela camada de agregação do destino;
- *notificação de recepção do agregado*: é gerado sempre que um agregado é recebido por um nó DTN;
- *notificação de aceitação de custódia*: é gerado para os nós que solicitam a custódia do agregado, inclusive para o nó DTN fonte;
- *notificação de encaminhamento do agregado*: é gerado para os nós que encaminharam o agregado, inclusive para o nó DTN fonte;
- *notificação de apagamento do agregado*: é enviada quando um agregado é apagado do buffer de um nó DTN. O objetivo é informar o motivo pelo qual o descarte se deu.

As notificações/solicitações de relatórios sobre o estado agregado pode resultar no aumento inaceitável do tráfego na rede, por isso a arquitetura DTN define que a geração de relatórios seja obrigatória somente no caso de um agregado aceito sob custódia ser apagado.

3.1.3 Protocolos de Roteamento

Como citado anteriormente, um dos grandes desafios em uma arquitetura DTN é o roteamento das mensagens desde seu envio efetuado pelo nó fonte até sua entrega recebida pelos nós destino. Este protocolos devem ser capazes de superar os problemas de atrasos extremamente longos e das freqüentes desconexões, já que os protocolos convencionais não estão aptos a manipular eficientemente a transmissão de dados em DTN (OLIVEIRA, 2010).

Existe uma grande variedade de protocolos de roteamento DTN que determinam como o envio da mensagem de nó-a-nó deve acontecer. Esses protocolos têm variações na política utilizada para escolher para quais nós vão repassar a mensagem para que essa chegue ao destino, de acordo com as características da aplicação e o modelo de mobilidade envolvido. Devido ao grande número de tipos de DTN, várias soluções tem sido propostas, logo abaixo são apresentados os protocolos de roteamento mais reconhecidos para essas redes segundo (OLIVEIRA, 2010).

Protocolo Directy Delivery

Em geral, os protocolos de transferência de mensagens, roteiam pacotes a outros nós com intuito de alcançar o destinatário final. No entanto o protocolo *Direct Delivery* não transfere mensagens para outro nó, a não ser que este nó seja o destino final da mensagem. O *Direct Delivery* pode ser considerado o algoritmo mais simples possível, visto que um nó envia uma mensagem para outro nó apenas se este for o destino da mensagem. Sendo assim, esse esquema não possui limite para o atraso da entrega, sendo sua vantagem possuir apenas uma única transmissão.

Protocolo Spray and Wait

O protocolo Spray and Wait (SPYROPOULOS; PSOUNIS; RAGHAVENDRA, 2005) produz apenas um determinado número de cópias de cada mensagem a serem transmitidas a outros nós da rede. O algoritmo tem como principal característica limitar o número de cópias de mensagens e transmissões na rede com intuito de não comprometer o seu desempenho.

Na primeira fase, é distribuído um número L de cópias da mensagem para L destinos. Caso o destino não seja encontrado nessa fase, cada um dos L nós que contém a mensagem produzirá uma tentativa de entrega direta (Direct Delivery), transmitindo a mensagem apenas se houver um encontro com o nó destino. Por ser um esquema simples, gera um número limitado de cópias de mensagens obtendo uma taxa de entrega satisfatória.

Protocolo Epidemic

O Protocolo *Epidemic* (VAHDAT; BECKER, 2000) ou roteamento epidêmico é um protocolo de roteamento estocástico, pelo fato de suportar entrega de mensagens a destinos arbitrários através suposições mínimas sobre o conhecimento da topologia de rede. O protocolo pressupõe que um nó fonte não possui conhecimento sobre a localização de um nó de destino, não conhecendo assim, a melhor rota para alcançá-lo.

Sua idéia principal é que a movimentação dos nós da rede crie a possibilidade de que entrem no alcance de transmissão uns dos outros periodicamente e, o mais importante, de maneira aleatória, ou seja, a mobilidade dos nós é usada como solução para que as mensagens cheguem ao seu destino.

Portanto, apenas o contato periódico par-a-par é preciso para garantir a entrega de eventuais mensagens. Durante o contato entre dois nós, listas com as mensagens armazenadas em cada nó são trocadas para que o nó possa determinar quais mensagens armazenadas no buffer do nó vizinho ele ainda não possui. Logo após a identificação das mensagens, cada nó efetua uma solicitação para que as cópias das mensagens que ainda não possui sejam enviadas. Essa troca é efetuada toda vez que um contato ocorre, permitindo que as mensagens sejam distribuídas rapidamente pelos nós que compõem a rede. Portanto, aumentando o número de cópias de uma mesma mensagem, maior é a probabilidade dela ser entregue em um menor tempo (menor atraso).

As principais desvantagens são alto custo, relativo ao grande número de retransmissões e conseqüente consumo dos recursos dos nós. Além disso, o tamanho limite do *buffer*, afeta diretamente a taxa de entrega.

Protocolo PROPHET

O *PROPHET* (LINDGREN; DORIA; SCHELÉN, 2003) é um protocolo de roteamento probabilístico que utiliza históricos de encontros e transitividade (*Probabilistic Routing Protocol using History of Encounters and Transitivity - PROPHET*). Assim como acontece no roteamento epidêmico, quando dois nós iniciam um contato são trocadas as listas com informações que identificam as mensagens armazenadas em cada nó. A diferença é que agora existe uma informação extra para cada mensagem indicada na lista. Essa informação corresponde à probabilidade de cada nó a entregar mensagens para um destino conhecido b, onde $P(a, b) \in R \mid 0 \leq P(a, b) \leq 1$.

O valor de P(a, b) aumenta sempre que a e b se encontram. Se a e b deixam de se encontrar freqüentemente, $P(a, b)$ diminui à proporção que o tempo transcorre. Esse tempo é controlado por uma constante k denominada constante de envelhecimento, que corresponde ao

número de unidades de tempo transcorridas desde a última vez que a métrica foi atualizada. A probabilidade de entrega também possui uma propriedade transitiva, que se baseia na seguinte observação: se um nó *a* encontra um nó *b* freqüentemente, e o nó *b* encontra freqüentemente um nó *c*, logo o nó *c* provavelmente é um bom nó para encaminhar mensagens destinadas para *a*.

Uma constante β, onde $\beta \in \mathrm{R} \mid 0 \leq \beta \leq 1$, é utilizada para definir o impacto da transitividade na entrega. Quando um nó recebe a lista do vizinho, ele calcula a probabilidade de entrega para cada uma das mensagens que ainda não possui armazenada. Em seguida, para cada mensagem, o nó compara a probabilidade indicada na sua lista com a probabilidade indicada na lista recebida do vizinho. Essa comparação é realizada para verificar qual dos dois nós possui a maior probabilidade de entrega. Feita essa comparação, devem ser realizados três procedimentos. Primeiro, o nó deve enviar um pedido das mensagens não armazenadas que possuem uma maior probabilidade de serem entregues através dele. Segundo, recebe o pedido de mensagens do vizinho e as envia.

Terceiro, apaga do *buffer* todas as mensagens que o vizinho tem maior probabilidade de entregar. No final, cada nó só ficará com as mensagens cujas probabilidades de entrega sejam maiores através dele. Os resultados das simulações demonstram que o *PROPHET* é capaz de entregar mais mensagens do que o roteamento epidêmico e com uma sobrecarga de comunicação menor, especialmente quando o alcance de comunicação dos nós é pequeno. Isso se deve ao fato do *PROPHET* enviar mensagens somente para os melhores nós, enquanto o epidêmico envia mensagens para todos os nós que encontra. Estratégias de descarte de mensagens baseadas na estimativa da probabilidade de entrega também podem ser utilizadas para aumentar o desempenho da rede quando os nós possuem *buffers* limitados.

3.2 Considerações Finais

Nos últimos anos muitos pesquisadores têm investigado e desenvolvido algoritmos de roteamento para redes DTN, contudo esta área continua sendo um dos maiores desafios, devido à especificidade deste ambiente.

Ainda é preciso evoluir muito a partir das soluções propostas, para que resultados satisfatórios sejam obtidos; é necessário encontrar um equilíbrio entre a sobrecarga produzida na rede pela grande quantidade de mensagens geradas pelos algoritmos (consumo da capacidade da rede) e o tempo médio de atraso de entrega das mensagens. Métodos efetivos devem ser construídos para que a entrega das mensagens seja garantida; esquemas inteligentes capazes de

tomar as melhores decisões precisam ser aprimorados atendendo assim aos requisitos impostos pelos longos atrasos e constantes desconexões.

4 O Protocolo VeloSent

Nos últimos anos muitos protocolos de roteamento têm sido propostos como trabalho de pesquisa dentro da área de redes tolerantes a atrasos e desconexões (DTN), o que comprova sua importância para obtenção de um desempenho satisfatório, para as métricas de taxa média de entrega das mensagens, tempo médio de atraso ou latência, número médio de saltos necessários para o roteamento da mensagem e quantidade de recursos de rede utilizados.

Diversos trabalhos foram publicados nos últimos anos apresentando o protocolo de roteamento como ponto chave em cenários que apresentam restrições, comprovando que a escolha do protocolo mais adequado depende das características do ambiente analisado.

Considerando que as DTNs apresentam cenários restritos, caracterizados pelo número limitado de contatos entre os nós que a compõem, pela baixa capacidade de transmissão e por nós com recursos escassos, tem-se a necessidade de definir uma política de encaminhamento de mensagens onde cada nó possa definir como e quando essas mensagens devem ser transmitidas, bem como qual nó vizinho deve ser escolhido para recebê-la.

Os protocolos propostos atualmente podem ser divididos, de um modo geral, em dois grandes grupos, de acordo com a forma como ocorre a disseminação das mensagens. Os protocolos que utilizam-se da disseminação epidêmica caracterizam-se pelo fato de que todos os nós da rede encaminham todas as mensagens recebidas para todos os seus vizinhos, fazendo com que as mensagens sejam replicadas rapidamente, isso aumenta, consideravelmente, as chances de que as mensagens sejam entregues, mas por outro lado essa qualidade está atrelada as características da rede, do contexto onde os nós estão inseridos, ou seja, da capacidade de armazenamento dos nós e duração do contato, o que em redes com restrições, caso das DTNs é limitado.

Contudo os protocolos que se utilizam da disseminação probabilística efetuam a análise sobre o encaminhamento ou não de uma determinada mensagem aos nós vizinhos. Eles são baseados em dados estatísticos, obtidos através de cálculos matemáticos usando informações como: duração dos contatos, periodicidade dos contatos, tamanho da mensagem, taxa de mensagens entregues ao nó destino, entre outras. Essa característica permite que os recursos da

rede sejam utilizados eficientemente, não gerando um volume grande de mensagens, através da propagação indiscriminada destas, entretanto esse modo depende de dados estatísticos, que se ausentes ou defasados podem gerar a perda de oportunidades de transmissão.

Entretanto, uma avaliação mais profunda desses protocolos, utilizando para tal, ambientes distintos daqueles para os quais foram projetados, mostra que os mesmos apresentam uma variação de desempenho expressiva. Sendo assim, apesar da grande quantidade de protocolos propostos, fica perceptível, dentro da literatura presente, que poucos ou nenhum deles é capaz de apresentar, independente do cenário, um desempenho competitivo, analisando de forma precisa o contexto onde está inserido e reagindo adequadamente as informações externas capturadas. A partir disso o trabalho apresentado busca demonstrar que a utilização de sensibilidade ao contexto em DTNs pode ser eficiente e tornar regular o desempenho dos protocolos, mesmo quando o contexto ou cenário onde estão sendo executados é modificado.

4.1 O Contexto e o Histórico de Contatos em DTN

Os cenários característicos de DTNs apresentam, em sua maioria, peculiaridades contrastantes que provocam uma oscilação considerável no desempenho dos principais protocolos de roteamento. Adicionado a isso, existe um conjunto de contextos que se formam ao longo da existência de cada cenário e desafiam a capacidade dos protocolos de roteamento em manter um padrão de desempenho em relação a métricas fundamentais para toda e qualquer aplicação, tais como a taxa média de entrega das mensagens (OLIVEIRA, 2008b).

A instabilidade no desempenho desses protocolos de roteamento para DTNs, quando são expostos a ambientes com características oscilatórias, pode ser contornada (a proposta em questão não tem por objetivo resolver o problema por completo, mas sim evidenciar a possibilidade da redução na oscilação do desempenho) através da utilização de métodos que possibilitem a detecção e adaptação dos protocolos de acordo com as mudanças ocorridas no contexto, tornando-os sensíveis ao mesmo.

Nas seções 4.1.1 e 4.1.2 são apresentados os conceitos utilizados ao longo do desenvolvimento do protocolo VeloSent, para que fosse possível utilizar-se do contexto como metodologia para roteamento de mensagens em DTNs.

4.1.1 Detecção do Contexto

A detecção de contexto, em um ambiente DTN, não é uma tarefa trivial, o contexto abrange diversos aspectos, podendo ser qualquer fragmento de informação que possa ser utilizado para descrever uma situação e com isso influenciar diretamente o comportamento de um sistema de computação ou nesse caso um protocolo de roteamento.

O processo de identificação do contexto requer a criação de regras que permitam, aos nós da rede, detectar e analisar o contexto local, que adicionado a um conhecimento prévio sobre a topologia da rede, obtido através do histórico de contatos entre nós e posterior estimativa da localização do destino, permite determinar a ação mais apropriada a ser tomada, com objetivo de obter-se um desempenho otimizado.

Analisando informações de contexto características de DTN, pode-se intercalar dados como: a quantidade de nós vizinhos que compõe possíveis contatos, sejam eles diretos ou indiretos, a velocidade com que os nós se movimentam, em qual direção e sentido seguem, a localização do nó, a capacidade de armazenamento, a duração de um contato e a taxa de transmissão, com o intuito de detectar o contexto, que pode ser utilizado como referência por um protocolo de roteamento.

4.1.2 Informações de Contexto Relevantes ao Protocolo VeloSent

Dentre as informações apresentadas anteriormente, três são consideradas passíveis de estudo, dentro dessa proposta, com o objetivo de prover detecção de contexto para a função de roteamento. Consideraram-se então neste trabalho, para efeito de contexto, as informações sobre a velocidade de deslocamento, sentido e posição geográfica dos nós da rede.

Torna-se importante salientar que várias outras informações podem ser tratadas em paralelo as citadas anteriormente, como por exemplo, o nível de bateria, já que um nó com essa consciência, ao detectar que sua energia está se esgotando poderia beneficiar-se do próximo contato para encaminhar todas as mensagens que contém com intuito de evitar que elas sejam perdidas em definitivo. Sendo assim, trabalhos futuros podem adicionar ao protocolo aqui proposto, mais informações de contexto, com intuito de melhorar o desempenho do algoritmo.

Entretanto, para que seja possível modelar um processo de adaptação dinâmica computacionalmente viável, de acordo com as informações detectadas, referentes às características peculiares encontradas em nós que compõem uma DTN, as informações de contexto ficaram restritas aos atributos de localização, velocidade e sentido dos nós.

4.2 Modelagem do Protocolo VeloSent

O protocolo VeloSent tem como base a utilização do histórico de encontros entre os nós da rede, o que possibilita a obtenção estimada da sua topologia. Essa capacidade evita que técnicas que criam um grande número de trocas de mensagens, como banco de dados compartilhados sejam utilizadas, garantindo um uso otimizado dos recursos disponíveis. Algoritmos como GREASE (GROSSGLAUSER; VETTERLI, 2006) utilizam-se desse histórico como ferramenta para determinar qual dos nós vizinhos deve receber a mensagem a ser entregue ao seu nó de destino.

Contudo o GREASE é um protocolo que não possui em sua concepção a ideia de contexto, não utilizando-se de outros dados relevantes para a escolha do próximo nó a receber uma mensagem. Sendo assim propõe-se a utilização das informações de contexto velocidade e sentido, que armazenadas juntamente com a localização dos nós, durante os contatos, permite estimar mais precisamente a localização do nós de destino, e ainda, analisadas durante o contato com os nós vizinhos permite identificar qual deles caminha na direção de entrega da mensagem e com qual velocidade o faz, produzindo também, como o GREASE, um refinamento da estimativa da trajetória do nó ao longo do roteamento.

4.2.1 Tabela de Contexto dos Últimos Contatos Efetuados - TCUCE

Durante os contatos entre os nós que compõem uma DTN, dados sobre o contexto, relevantes ao algoritmo de roteamento, são armazenados em um tabela denominada Tabela de Contexto dos Último Contatos Efetuados - TCUCE. Os dados armazenados nesta tabela são compostos por: identificador do nó contatado, momento em que o contato foi efetuado (relógio local), posição do nó, e velocidade do nó. Através desses dados é possível obter 3 informações chaves para o protocolo de roteamento proposto:

1. Utilizando-se do momento em que o contato foi efetuado é possível ter uma estimativa inicial da posição geográfica do nó, já que quanto menor for a idade do contato (tempo decorrido) mais alta é a possibilidade do nó ainda estar na região onde foi encontrado pela última vez;

2. Através da velocidade é possível calcular de forma estimada onde encontra-se o nó de destino no momento em que uma mensagem destinada a ele é roteada entre nós vizinhos, utilizando-se do tempo decorrido desde o último contato efetuado - idade do contato;

3. Através da velocidade e sentido dos nós vizinhos é possível descobrir qual deles segue na direção mais próxima a posição estimada do nó de destino, é possível também estimar o ponto e momento onde se encontrarão (caso isso vá ocorrer) e também estimar quão próximos estarão nesse momento.

São as informações extraídas dos dados armazenados na tabela que permitem obter de forma estimada: a topologia da rede, a localização geográfica dos nós, o sentido que seguem, com qual velocidade e para qual região se movimentam e quais nós se encontram em um futuro próximo. Todas essas informações são analisadas pelo protocolo proposto no momento em que é tomada a decisão sobre qual nó será o próximo a receber uma determinada mensagem durante o seu roteamento.

4.2.2 Estrutura da Tabela de Contexto dos Últimos Contatos Efetuados

Como mencionado anteriormente a TCUCE é composta pelos campos de *ID* - identificador do nó, T - momento em que o encontro entre dois nós ocorreu, P_x - posição dentro do eixo "x" em que o nó se encontra no momento do contato, P_y - posição dentro o eixo "y" em que o nó se encontra no momento do contato, V_x - velocidade dentro do eixo "x" em que o nó se encontra no momento do contato e V_y - velocidade dentro do eixo "y" em que o nó se encontra no momento do contato.

A figura 4.1 ilustra a estrutura da TCUCE mencionada anteriormente:

ID	T	Px	Py	Vx	Vy
4	63	4	6	-1	2
5	64	7	7	3	-1

Figura 4.1: Estrutura da TCUCE - Tabela de Contexto dos Ultimos Contatos Efetuados.

O protocolo VeloSent, em sua versão inicial, não possui, ainda, uma metodologia de gerenciamento de descarte das linhas da Tabela de Contexto dos Últimos Contatos Efetuados. Sendo assim, os dados armazenados na TCUCE não são removidos. Contudo esse descarte torna-se necessário por questões de armazenamento e também torna-se interesante se considerarmos que, quanto mais antigo for um contato, menos preciso serão os seus dados de contexto.

A definição de uma idade máxima para os contato, como metodologia para definir se seus dados de contexto ainda são relevantes, poderia ser utilizada como parâmetro para identificar as linhas da tabela que ainda são úteis e quais podem ser descartadas.

4.2.3 O Funcionamento do Protocolo VeloSent

O protocolo de roteamento VeloSent funciona em 3 fases: (a) em um primeiro momento uma análise do ambiente é efetuada para detectar nós vizinhos e seus respectivos contextos, em seguida obtêm-se do vizinho que encontrou o destino a menos tempo, o momento, a localização e a velocidade do nó de destino; (b) a partir do contexto do nó de destino, localização e velocidade, é possível estimar a sua nova localização desde o último encontro até o momento atual, considerando o quanto se deslocou desde então; (c) com a localização estimada do destino, os dados de contexto coletados dos vizinhos são analisados para estimar qual deles segue na direção onde está localizado o destino e com qual velocidade, estimando matematicamente o momento e ponto de um possível encontro entre eles. De acordo com essa informação um novo nó *âncora* é escolhido dentre os vizinhos analisados e então receberá a mensagem a ser entregue ao nó de destino. Importante observar, que uma réplica da mensagem é transmitida ao novo nó âncora, o que possibilita termos mais de um nó nessa mesma condição. O VeloSent não possui um mecanismo de "cura", como por exemplo o TTL (*time-to-life*) - tempo de vida, para descartar as mensagens já entregues.

Primeira Fase

A primeira fase do protocolo é caracterizada pela detecção e comunicação com os nós vizinhos, obtendo deles informações de suas TCUCE sobre um nó de destino para uma determinada mensagem a ser transmitida. O nó que tem por finalidade transmiti-la é denominado como nó âncora. Nessa fase a âncora armazena ou atualiza os dados de contexto (posição, momento e velocidade) sobre os contatos que estão sendo efetuados e obtém a posição e momento em que esses vizinhos tiveram contato com o nó de destino da mensagem em questão. A primeira fase pode ser caracterizada como a etapa onde ocorre comunicação para obtenção do contexto local e do último contexto conhecido sobre o nó de destino através da idade dos contatos. A figura 4.2 ilustra a primeira fase:

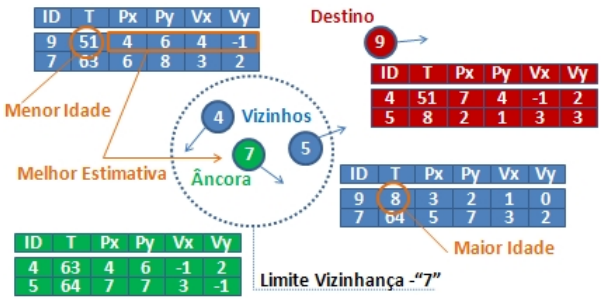

Figura 4.2: Primeira fase do protocolo - descoberta do contexto do nó de destino

No cenário apresentado na figura 10 o nó âncora (7) efetua contato com seus nós vizinhos (4, 5) na expectativa de obter informações sobre o nó de destino (9) de uma dada mensagem. Durante esse período a TCUCE dos nós vizinhos é consultada para buscar informações sobre o último contato que eles tiveram com o nó de destino, esses dados são passados ao nó âncora, que os utiliza para calcular a idade destes contatos e assim descobrir qual dos vizinhos possui a melhor estimativa sobre o destino.

Nesse momento, torna-se importante ressaltar que, o cálculo da idade é feito utilizando-se como referência o relógio do nó vizinho, obtendo deste o momento atual e o momento em que ele encontrou com o destino, isso possibilita a não necessidade de sincronismo de relógios. Entretanto, no exemplo acima, considera-se que todos os relógios avançam com uma mesma medida de tempo e partem juntos do momento 0, para que seja possível ter-se um melhor entendimento sobre o cálculo da idade do contato. Portanto, considerando que "51 $>$ 8", e supondo que os relógios dos nós estão no momento T=65 é possível descobrir que a idade do contato do nó "4" é menor, já que encontrou o destino em um momento posterior ao do nó "5", ou seja, a idade do contato do nó "4" é IC_4 = 65-51 = **6**, enquanto a idade do contato do nó "5" é IC_5 = 65-8 = **57**. Portanto os dados de contexto armazenados pelo vizinho "4" durante o seu contato com o nó de destino são escolhidos para serem utilizados nas próximas fases do protocolo: P_x=4; P_y=6; V_x=4 e V_y=-1.

Segunda Fase

A segunda fase inicia-se após os dados de contexto terem sidos levantados através dos contatos efetuados pelo nó âncora com seus vizinhos. Nesta etapa serão iniciadas as análises dos dados relativos ao nó de destino, que foram selecionados de acordo com a menor idade calculada na primeira fase. De posse desses dados é possível ter uma estimativa da atual posição do nó destino, considerando a sua posição anterior, sua velocidade, e o tempo decorrido desde o contato até o momento atual. Essa estimativa é a primeira informação crucial para determinar como a mensagem será roteada. A segunda fase pode ser caracterizada como a etapa em que a localização do de destino é estimada para o momento corrente. A figura 4.3 ilustra a segunda fase:

Figura 4.3: Segunda fase do protocolo - posição estimada do destino

No cenário apresentado na figura 4.3 é possível visualizar o momento em que os nós "4" e "9" se encontram. Se considerarmos as tabelas apresentadas na figura 3 verificamos que esse contato ocorre no momento "51" e posteriormente, no momento 63 um novo contato entre os nós "7" (âncora) e "4" ocorre, fazendo com que o nó "7" obtenha os dados de contexto do nó de destino e assim possa , através da posição, velocidade e idade, estimar onde encontra-se, atualmente o nó "9".

Sabendo a posição onde encontrava-se o nó de destino, adicionada da informação sobre sua velocidade nos eixos x e y, é possível estimar onde ele está passado o tempo do último contato que ele efetuou com o nó vizinho "4", considerando que esse tempo é exatamente a idade desse último contato. Como é apenas uma estimativa de localização, considera-se que as velocidades do nó "9", assim como a de todos os outros nós que compõem a rede, mantiveram-se constantes durante esse período, o que caracteriza um MRU - Movimento Retilíneo Uniforme. Sendo assim teríamos como dados: *Idade do Contato* (*IC*); *Posição do destino em x* (P_x); *Posição do destino em y* (P_y); *Velocidade do destino em x* (V_x); *Velocidade do destino em y* (V_y); Utilizando a fórmula matemática da velocidade média: V_m=Δx/Δt, considerando as informações de contexto do destino, obtidas pelo nó âncora em contato com seus vizinhos, teríamos como fórmula para obter a estimativa de sua nova localização:

$$\mathbf{P}e_x = (V_x \,.\, \mathrm{IC}) + P_x \qquad \mathbf{P}e_y = (V_y \,.\, \mathrm{IC}) + P_y$$

Ao final da segunda fase do protocolo, pela aplicação das fórmulas apresentadas para os dados de contexto obtidos, tem-se, calculada, a primeira importante informação: a posição estimada do destino, utilizada para determinar como a mensagem será roteada.

Terceira Fase

A terceira fase começa quando tem-se determinada a estimativa da localização do nó de destino, a partir dessa informação, agregada aos dados de contexto dos nós vizinhos obtidos

na primeira fase é possível identificar, de forma estimada, qual desses nós segue no sentido que chegará mais próximo da localização do destino obtida na segunda fase do protocolo. A terceira fase pode ser caracterizada pela etapa onde estima-se um possível ponto de encontro com o nó de destino, tendo assim a identificação e seleção do vizinho que segue no sentido que o levará mais próximo desse encontro.

O nó escolhido será o próximo a receber a mensagem durante o seu processo de roteamento até o destino. A figura 4.4 ilustra a terceira fase:

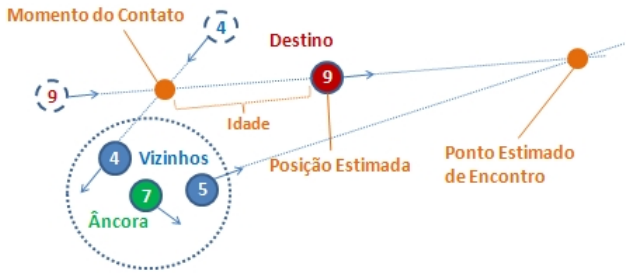

Figura 4.4: Terceira fase do protocolo - ponto estimado de encontro com o destino.

O cenário da figura 4.4 apresenta o ponto estimado onde um possível contato entre os nós "5", escolhido para receber a mensagem, e o nó "9" de destino ocorrem. Para tal estimativa são considerados que os nós mantêm constantes suas velocidades de acordo com o que foi capturado e armazenado nas tabelas de contexto dos últimos contatos efetuados - TCUCE.

Nestas condições, os nós seguem a trajetória de uma reta, permitindo que sejam utilizados recursos matemáticos conhecidos para prever o ponto de intersecção entre ambas. Essa é a fase mais complexa do protocolo, sendo assim ela é dividida em quatro etapas apresentadas e detalhadas a seguir:

1. *Encontrando a equação da reta que define a trajetória dos nós*: o primeiro passo da terceira fase do protocolo é encontrar a equação das retas que definem a trajetória dos nós vizinhos e também do nó de destino. Para isso utilizamos a equação da reta, $y=\alpha.x+\delta$, onde α é o coeficiente angular da reta, e pode ser calculado pela equação: $\alpha=(y-y0)/(x-x0)$ e δ é o coeficiente linear da reta e pode ser calculado pela equação: $\delta = y - \alpha.x$;

2. *Encontrando o ponto de intersecção entre os nós vizinho e destino*: o segundo passo da terceira fase é determinar em qual ponto os nós se encontram, ou seja, descobrir o ponto de intersecção entre as retas (equações) que representam a trajetória dos nós. Essa fase consiste em resolver o sistema de equações que contém as duas equações encontradas na etapa anterior, ou seja, a equação da reta de um nó vizinho com a equação da reta do nó

de destino. Ao final dessa etapa temos os pontos de intersecção entre os nós vizinhos e o nó de destino. A figura 4.5 ilustra a trajetória dos nós e o ponto de intersecção entre elas.

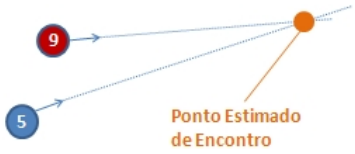

Figura 4.5: Segunda etapa da terceira fase do protocolo - ponto de intersecção estimado.

3. *Cálculo do tempo necessário para que o ponto de intersecção seja atingido*: após determinar o ponto de intersecção entre os nós vizinhos e o nó de destino, inicia-se a etapa do cálculo estimado do tempo necessário para que esse encontro ocorra. Nesta etapa é possível determinar qual dos nós vizinhos leva menos tempo para alcançar o ponto de encontro com o nó de destino. Para tal utiliza-se a fórmula da velocidade média, já que temos a velocidade dos nós, o ponto inicial de sua partida e o ponto final que seria a intersecção com destino, sendo assim a única variável a ser encontrada seria exatamente o tempo que leva: V_m=Δx/Δt. A figura 4.6 ilustra o tempo necessário para atingir o ponto de intersecção.

Figura 4.6: Terceira etapa da terceira fase do protocolo - tempo necessário.

4. *Estimativa da posição do destino no momento em que o nó vizinho estiver no ponto de intersecção*: após as etapas 3 e 4, ou seja, a identificação do momento em que os nós se encontram e quantidade de tempo que leva para que isso ocorra inicia-se a etapa de verificação da posição do nó de destino no momento em que os vizinhos atingirem o ponto de encontro estimado na etapa 2. Essa estimativa da posição do destino pode ser feita de acordo com o tempo calculado na etapa 3, para isso considera-se a velocidade do destino obtida na primeira fase do protocolo. A fórmula utilizada aqui, assim como na etapa anterior, é a da velocidade média: V_m=Δx/Δt. A figura 4.7 ilustra a estimativa da posição do destino no momento em que o nó vizinho estiver no ponto de interseção das duas trajetórias.

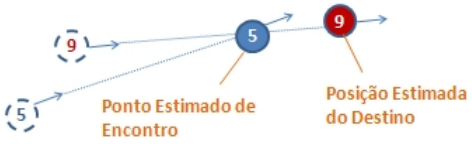

Figura 4.7: Quarta etapa da terceira fase do protocolo - estimativa da posição do destino quando o nó vizinho atingir o ponto de intersecção das retas que representam as suas trajetórias.

5. *Distância entre o ponto de intersecção e a posição do nó de destino no momento do encontro*: a última etapa da fase 3 é a verificação da distância entre o nó vizinho no momento em que o ocorre a intersecção das retas que representam suas trajetórias. Visto que não necessariamente os nós passarão pelo ponto de intersecção ao mesmo tempo, é calculado, de forma estimada, onde encontra-se o nó destino no momento em que o nó vizinho está no ponto de encontro, já que para o protocolo VeloSent esse é o momento onde os dois estarão mais próximos. Entretanto cálculos matemáticos mais apurados podem definir uma estimativa mais precisa sobre o momento em que essa menor distância ocorre. Efetuando esse cálculo é possível definir qual dos nós vizinhos chegará mais próximo do destino em um futuro próximo, podendo assim rotear até este uma mensagem. A figura 4.8 apresenta a distância entre o nó vizinho e o nó de destino.

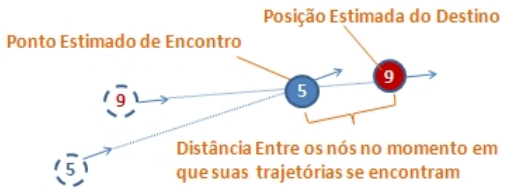

Figura 4.8: Quinta etapa da terceira fase do protocolo - distância entre o nó vizinho e o nó de destino no momento em que o vizinho está no ponto de intersecção das retas que representam as suas trajetórias.

Ao final de todas as fases do protocolo, já com todos os cálculos necessários efetuados, é possível, então, definir o novo nó âncora que receberá a mensagem a ser roteada até o seu nó de destino.

5 Modelos de Mobilidade e Trabalho Relacionado

Este capítulo destina-se a apresentar os principais modelos de mobilidade utilizados durante as simulações efetuadas para analisar o comportamento protocolo de roteamento proposto neste documento, além de mostrar o trabalho na literatura utilizado como referência no desenvolvimento do VeloSent.

5.1 A Mobilidade dos nós e as Redes *Ad hoc*

As redes *ad hoc* sem fio possuem como característica a não existência de uma infra-estrutura de rede, enquanto que os nós podem ser fixos ou móveis. Essa característica torna esse tipo de rede extremamente versátil, podendo ser utilizada em ambientes sem nenhuma infra-estrutura, tais como, monitoramento ambiental, eventos catastróficos ou campos de batalha (MORAES, 2009).

As redes *ad hoc* devem possuir protocolos de comunicação descentralizados e que façam pouco uso de recursos computacionais e energia, dada a falta de uma infra-estrutura (MURTHY; MANOJ, 2004). Esses protocolos devem ser testados sob condições que procurem retratar os mais diversos cenários práticos que um usuário deste sistema possa encontrar, por este fato, o efeito de mobilidade em redes *ad hoc* tem sido estudado por diversos autores, interessados em investigar os sistemas e protocolos de comunicação sob condições de uso que reproduzam as situações de usuários em movimento que utilizam a rede (HONG et al., 1999), (POTNIS; MAHAJAN, 2006), (BETTSTETTER, 2001), (YOON; LIU; NOBLE, 2003).

Como consequência, o desempenho destes protocolos depende fortemente do modelo de mobilidade empregado e de suas características, visto que a dinâmica da mobilidade pode interferir efetivamente nos resultados de análise de desempenho dos protocolos.

5.2 Os Modelos de Mobilidade

Modelo de mobilidade, também conhecido por modelo de movimento, pode ser definido como um modelo matemático que descreve o padrão de movimento dos nós móveis (e.g., pessoas, veículos). Ele determina como os componentes do movimento (i.e., localização, velocidade, aceleração) dos nós variam ao longo do tempo. O principal objetivo é imitar o comportamento real da mobilidade dos nós. (CAVALCANTI, 2009)

Os principais modelos utilizados em redes *ad hoc* e empregados no desenvolvimento da proposta apresentada neste documento são descritos nas seções 5.1.1 e 5.1.2.

5.2.1 O Modelo de Mobilidade *Random Waypoint*

Dentre os vários modelos de mobilidade utilizados na literatura e nos simuladores para redes *ad hoc*, o *Random Waypoint* (RWP) é certamente o mais empregado (YOON; LIU; NOBLE, 2003), (CAMP; BOLENG; DAVIES, 2002), (BROCH et al., 1998). Suas principais características são a escolha aleatória da posição e velocidade e o *tempo de pausa* entre o intervalo de mudança de direção do movimento de cada nó.

Essas características, peculiares do modelo *Random Waypoint*, fazem com que ele se torne mais próximo da realidade e seja empregado largamente na literatura como metodologia para validar protocolos de comunicação em redes *ad hoc* móveis (MORAES, 2009).

O procedimento de funcionamento do RWP é determinado pela seguinte proposição, onde considera-se uma rede de área retangular com dimensões X_{max} x Y_{max}.

- **Proposição** - *O modelo de mobilidade RWP (BROCH et al., 1998), (CAMP; BOLENG; DAVIES, 2002): (i) Escolhe-se aleatoriamente uma posição inicial para cada nó dada pela coordenada (x, y) onde x e y são uniformemente distribuídos no intervlo [0, X_{max}][0, Y_{max}], respectivamente. (ii) Em seguida, cada nó seleciona uma ponto de destino (x', y') uniformemente distribuído na área da rede, e uma velocidade v uniformemente distribuída no intervalo [V_{min}, V_{max}], onde V_{min} e V_{max} são possíveis velocidades mínima e máxima, respectivamente, que um nó pode escolher, sendo $0 < V_{min} < V_{max}$. (iii) Antes de iniciar seu movimento o nó permanece parado por um tempo de pausa que pode ser fixo ou aleatório. (iv) Ao se expirar este tempo, o nó move-se em linha reta para o ponto de destino (x', y') com a velocidade escolhida v. (v) Ao atingir o destino, o nó repete o procedimento a partir de (ii), e assim sucessivamente até o final da simulação.*

A velocidade média instantânea dos nós na rede para um dado cenário de mobilidade é definida (YOON; LIU; NOBLE, 2003) por:

$$v(t) = \frac{\sum_{i=1}^{N} v_i(t)}{N}$$

onde N é o número total de nós e $v_i(t)$ é a velocidade do i-ésimo nó no tempo t. A figura 5.1 ilustra o movimento de um nó para o modelo RWP.

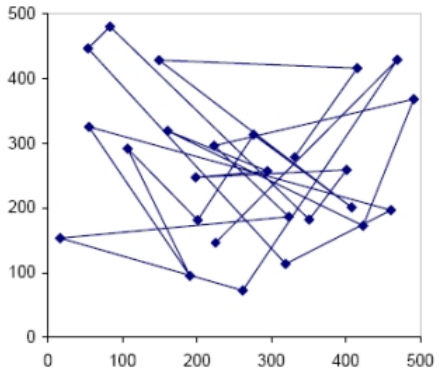

Figura 5.1: Movimento de um nó para o modelo RWP.
Fonte: (DAVIES, 2000).

5.2.2 O Modelo de Mobilidade *Random Walk*

Conforme (CAMP; BOLENG; DAVIES, 2002), o modelo de mobilidade *Random Walk* (RW) funciona da seguinte forma: em um intervalo de tempo constante, um nó move-se de sua posição atual para uma outra posição escolhendo aleatoriamente uma direção e uma velocidade. O *Random Walk* define para cada nó da rede um percurso que é gerado aleatoriamente, com variações, também aleatórias, nos módulos das velocidades. O RW é um modelo de simples implementação, e por essa característica tornou-se um dos modelos mais utilizados na análise de protocolos de roteamento em redes *ad hoc*.

A principal característica deste modelo é a sua independência temporal, visto que não há existência de memória, pois tanto o módulo da direção quanto o da velocidade de um nó, em um dado instante de tempo, independem dos valores anteriores. Neste modelo de movimento, ao contrário do *Random Waypoint*, o *tempo de pausa* é igual a zero ($T_{pausa} = 0$).

5.3 Trabalho Relacionado

As seções a seguir tratam do trabalho na literatura mais alinhado com a proposta apresentada neste documento.

5.3.1 Localização de nós móveis da rede através do "EASE": Descoberta de rotas apenas com a utilização do histórico de encontros

Em grandes redes adhoc e redes de sensores, alguns ou todos os nós podem estar se movendo. Sendo assim, a topologia da rede muda com o passar do tempo. Os algoritmos dentro desse cenário precisam basear suas decisões de roteamento sobre ao menos um conhecimento parcial da topologia da rede. A obtenção e troca de informações sobre a topologia (vetores de distância ou estados de ligação) consome muita largura de banda e energia (GROSSGLAUSER; VETTERLI, 2006).

Em geral, um nó só precisa saber a sua localização e a dos seus vizinhos para tomar uma decisão de roteamento para qualquer destino. Contudo, ainda necessita de roteamento geográfico de uma localização eficiente, ou seja, um serviço de banco de dados distribuído para registar à localização de cada nó de destino. Sendo assim, toda e qualquer mudança na topologia da rede precisa ser aplicada ao banco de dados distribuído, produzindo a troca de informações entre os nós, e portanto, criando um custo de transmissão. Uma maneira elegante de diminuir esse custo é explorar o efeito da distância, que basicamente é a observação da precisão com que a posição de um destino tem que ser avaliada para obter-se uma boa decisão de roteamento local.

Neste contexto, onde trocar as atualizações de localização de forma explícita acarreta em um alto custo, a única informação sobre a topologia que se tem disponível localmente para um nó da rede é o histórico de contatos com outros nós no passado. Mais especificamente, assumimos que cada nó se lembra da hora e local do seu último encontro com todos os outros nós, isto é, quando esses dois nós foram diretamente vizinhos, como mostra a figura 5.2.

O algoritmo em questão é chamado de "Roteamento do último encontro" (LER) pois em cada nó ao longo de uma rota de pacotes, a decisão sobre o próximo salto depende apenas do tempo e local do último encontro do nó com o destino, e uma informação auxiliar transportada por esse pacote.

O centro da investigação consiste, que por um lado, a mobilidade dos nós gera incerteza sobre sua localização, mas pelo outro, que algum nó $\boldsymbol{d}$ é o destino de um pacote e algum outro

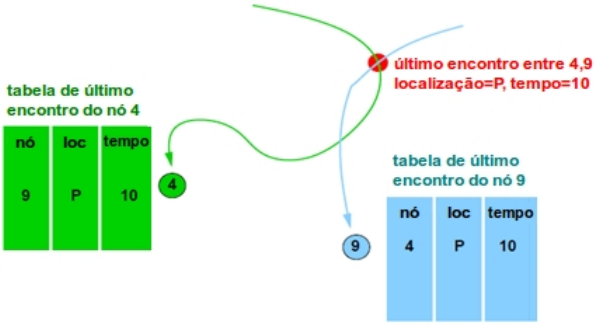

Figura 5.2: Tabela de último encontro - cada nó lembra o local e a hora da o último contato com todos os outros nós na rede.

Fonte: (GROSSGLAUSER; VETTERLI, 2006).

nó ***i*** encontrou o nó ***d*** no passado e lembra a localização desse último encontro.

Três observações explicam o motivo pelo qual o algoritmo LER pode dar origem a rotas eficientes:

1. o local do último encontro é ainda uma estimativa razoavelmente boa da localização do destino, depois de algum tempo;

2. o tempo decorrido desse encontro, ou a "idade" do estimador, é uma medida de precisão para essa estimativa;

3. a mobilidade própria do nó ***i*** significa que uma estimativa recente da posição de ***d*** está disponível a certa distância ***d***, uma vez que ***d*** encontrou outros nós o tempo todo, devido à mobilidade, isto essencialmente leva a um efeito de difusão das estimativas da posição em torno de ***d***.

EASE - Algoritmo de Roteamento da Idade Exponencial de Pesquisa

O algoritmo EASE funciona através de fases alternadas. Na primeira, quando há o recebimento de um pacote por um nó âncora - este realiza uma busca local em torno de si com intuito de identificar um possível próxima âncora. Na segunda fase, um algoritmo de roteamento existente, baseado em posição, é usado para rotear o pacote para o nova âncora.

A abordagem em duas fases é útil para a análise do desempenho, contudo, é possível identificar que algumas informações relevantes são ignoradas, uma vez que os bancos de dados LER locais dos nós não são consultados durante a segunda fase. Dentro desse cenário, é proposto um algoritmo modificado chamado GREASE (Greedy EASE) que verifica a idade do último encontro com o destino em cada salto. Quando um nó que possui uma estimativa mais

recente sobre a localização do destino que o âncora é encontrado o pacote é roteado para ele, e este nó torna-se assim o novo âncora.

5.3.2 Considerações Finais

A utilização do histórico de encontros entre os nós da rede como técnica para estimar a sua topologia, proporciona um uso mais otimizado dos recursos disponíveis, já que não necessita de um grande número de trocas de mensagens entre os nós, o que acontece em técnicas como banco de dados distribuídos.

A partir do histórico de encontros torna-se possível, também, determinar a idade desses contatos, ou seja, o tempo decorrido desde que eles aconteceram. Essa idade permite estimar a posição de um nó destino, e baseado nessa informação de posição, algoritmos de roteamento podem definir qual dos seus vizinhos deve ser o próximo nó a receber uma mensagem e continuar seu roteamento. O protocolo GREASE, por exemplo, escolhe, para receber uma mensagem, o vizinho que possui a menor idade de contato com o nó de destino dessa mensagem.

O protocolo VeloSent parte dessa mesma premissa, ou seja, também utiliza histórico de encontros e a informação de idade. Contudo, propõe uma análise ainda mais detalhada do contexto que envolve os contatos e os vizinhos que cercam o nó âncora no momento do roteamento. No VeloSent os nós armazenam em suas tabelas locais, durante os encontros, além da posição e do momento, a velocidade com que os nós se deslocam.

A utilização da velocidade permite identificar o sentido de deslocamento dos nós, possibilitando estimar de forma mais precisa a posição onde localiza-se o nó de destino após o tempo decorrido desde o seu último encontro com ele. O VeloSent permite, ainda, identificar qual nó vizinho, no momento do roteamento, desloca-se na direção de encontro com o destino.

6 Implementação, simulação e análise dos resultados obtidos pelo protocolo VeloSent

As seções a seguir apresentam a metodologia utilizada na implementação, simulação e nos testes efetuados com o protocolo VeloSent.

6.1 Simulação: ambiente de simulação

O software de simulação utilizado para verificar o comportamento do protocolo VeloSent baseado em contexto, proposto neste artigo, foi o Ambiente de Simulação para Redes Oportunistas The ONE (KERäNEN; OTT; KäRKKäINEN, 2009).

O The ONE é um simulador robusto e flexível, que permite a configuração e simulação de diversos tipos de modelos de mobilidade dos nós que compõem uma DTN, possibilitando também a utilização de vários tipos de protocolos de roteamento de mensagens.

O simulador dispõe de funcionalidades peculiares durante a simulação, como a possibilidade de visualização em tempo real de todas as transmissões de mensagens que estão ocorrendo, bem como todos os contatos efetuados entre os nós.

O The ONE também traz implementado, de forma nativa, a maioria dos protocolos de roteamento mais conhecidos da literatura, tantos os que possuem seu funcionamento baseado no modelo estocástico, caso do protocolo *Epidemic* e do protocolo *Spray And Wait*, quanto os baseados no modelo probabilístico, caso protocolo *Prophet* e do protocolo *MaxProp*. Essa característica permitiu que outros protocolos pudessem ser utilizados durante as simulações para efeito de comparações com o protocolo proposto.

Por fim, a estrutura, como foi implementado o The ONE, permite a codificação e agregação de novos protocolos sem a necessidade de um conhecimento profundo sobre todos os módulos que o compõem, o que facilitou o trabalho de implementação dos novos protocolos GREASE e VeloSent ao seu projeto.

6.2 Simulação: metodologia utilizada

Ao levarmos em consideração o fato de que o protocolo proposto nesse documento trata-se de um protocolo baseado em contexto: posição, velocidade e sentido, ou seja, ele obtém e analisa os dados do ambiente para decidir como responder aos mesmos, este tende a adaptar-se ao meio onde encontra-se. Sendo assim, diferentemente da maioria dos protocolos já conhecidos na literatura, que possuem seu melhor desempenho quando executados em determinados tipos de cenários, o protocolo VeloSent tem por objetivo, avaliar e responder de acordo com seu contexto. Por esse motivo, optou-se por testá-lo em três ambientes distintos, onde variaram-se os tipos de nós que compõem a rede, como: pessoas, carros e bondes elétricos, variaram-se os modelos de movimento destes: como movimentos baseados em mapa de cidade, movimentos com rotas aleatórias (*Random Waypoint*) e movimento de pessoas caminhando aleatoriamente (*Random Walk*). E por fim, para obtenção dos resultados, modificou-se também para cada simulação efetuada a densidade da rede, ou seja, a quantidade de nós utilizados dentro da simulação.

Sendo assim as simulações foram desenvolvidas para três cenários diferentes, levando em consideração principalmente o movimento dos nós, visto que o contexto avaliado tem seu foco no movimento, mais especificamente, nas trajetórias dos nós que compõem a rede. Com intuito de obter-se uma análise mais profunda e comparativa foram simulados, nos mesmos ambientes, mais 3 protocolos, são eles: o protocolo *Epimedic*, muito conhecido na literatura e representante do grupo de protocolos estocásticos, o protocolo *Prophet*, representante do grupo de protocolos probabilísticos e o protocolo GREASE, que assim como o protocolo VeloSent utiliza o histórico de encontro com outros nós para determinar a topologia da rede, mas ao contrário do mesmo não se preocupa em analisar outros possíveis dados de contexto disponíveis, como a velocidade e o sentido de deslocamento dos nós vizinhos.

Por fim, avaliaram-se os custos e benefícios dos protocolos, visto que o intuito desta pesquisa é demonstrar que a utilização da adaptação ao contexto, mesmo que apenas sob determinadas informações, pode ser eficiente, independentemente do cenário analisado, sendo possível obter um bom desempenho não apenas em uma determinada métrica, mas sim conseguir um equilíbrio entre elas.

6.3 Simulação: resultados obtidos

Como citado anteriormente, as simulações foram divididas em três etapas, caracterizadas pelos diferentes cenários utilizados em cada uma delas. Cada cenário escolhido se diferencia do outro, principalmente, pelo tipo de movimento dos nós que o compõem, essa metodologia utilizada foi escolhida pelo fato de que o contexto analisado pelo protocolo está centrado na forma como os nós da rede se movimentam, mais especificamente, na sua velocidade e sentido.

6.3.1 Primeiro Cenário: Movimento baseado em mapa

O primeiro cenário escolhido para o início da simulação e obtenção dos primeiros resultados foi o mapa da cidade de *Helsinki*, cenário *default* do simulador The ONE. Este cenário contém vários tipos de nós, cada qual com um tipo específico de movimento, diretamente relacionado ao local onde se locomove, a direção, a velocidade e o sentido com que o faz. Nesse cenário os nós foram divididos em grupos: de pessoas, carros e bondes elétricos, no intuito de reproduzir uma situação real do dia-a-dia. Com isso o tempo de simulação utilizado foi de 12 horas (de acordo como configurações do simulador e não o tempo real), considerando que seja este, aproximadamente, o período total de um dia normal de trabalho em uma grande cidade.

Neste mesmo ambiente foram feitas simulações com os 4 protocolos citados anteriormente, e a cada nova simulação foi aumentado o número de nós (densidade da rede) para que fosse possível obter uma quantidade de dados suficiente para visualização do comportamento de cada um dos protocolos. Configurações disponibilizadas pelo simulador, características de ambientes DTN, tais como: quantidade limitada de armazenamento e raio de alcance máximo para comunicação e transmissão de dados entre os nós também foram utilizados durante a simulação. A figura 6.1 apresenta o cenário utilizado quando executado pelo simulador The ONE.

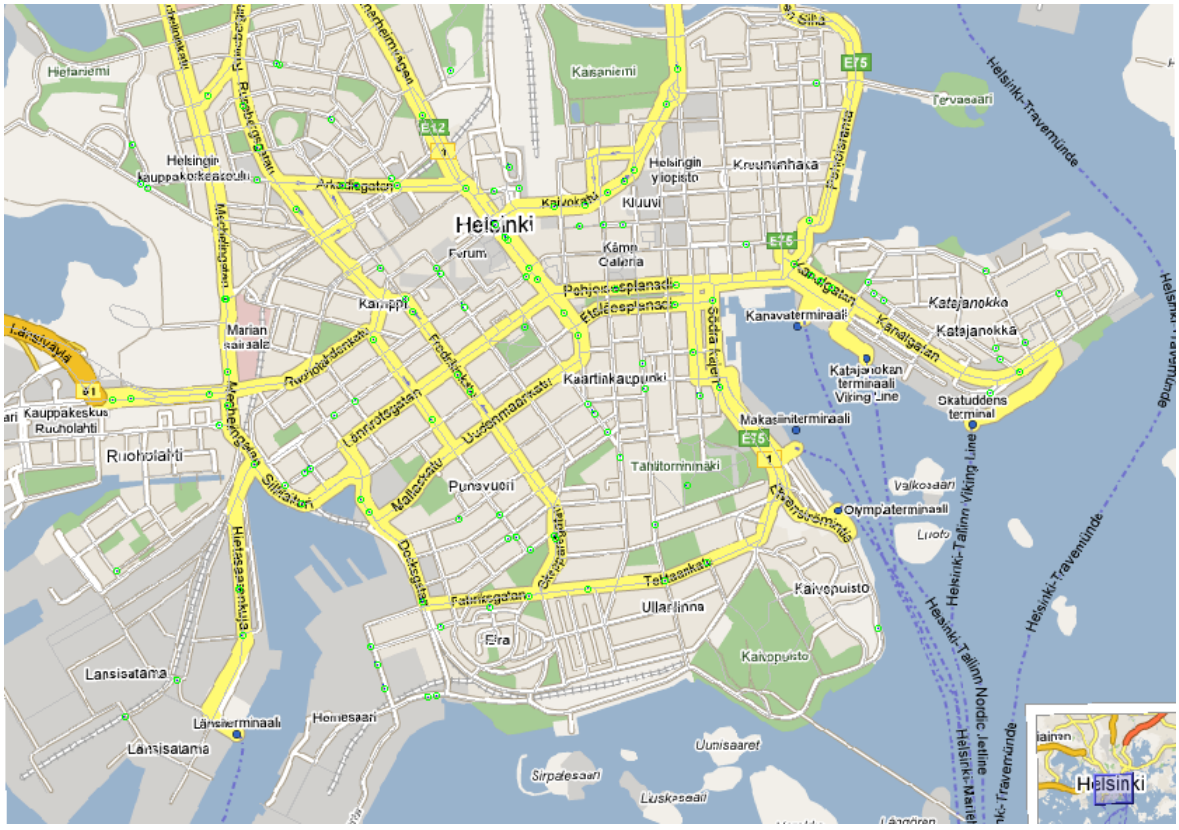

Figura 6.1: Movimento baseado em mapa - Cenário real de uma cidade - Simulador The ONE.

Os resultados das simulações utilizados nesta pesquisa foram escolhidos com o intuito de compreender como os protocolos de roteamento se comportam de modo geral, ou seja, não apenas no que diz respeito a qual deles possui melhor desempenho na probabilidade de entrega das mensagens ao seu nó de destino, por exemplo, mas também a maneira como essa entrega é feita, avaliando-se métricas relativas ao custo produzido pela forma como os recursos de rede são utilizados.

Dentro deste contexto foram coletadas, primeiramente, as informações sobre o benefício do protocolo, caracterizado pela probabilidade média de entrega das mensagens geradas na rede, objetivo final de qualquer protocolo que tem a função de rotear mensagens em DTNs. Os valores médios foram obtidos a partir de um número inicial de nós que compunham a simulação, os quais foram sendo aumentados de forma gradativa. Com esses dados foi possível obter o gráfico da figura 6.2, para os 4 protocolos utilizados durante a simulação:

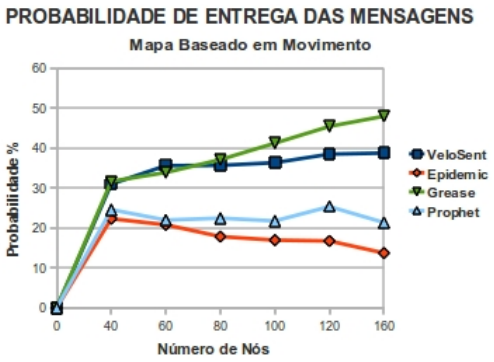

Figura 6.2: Movimento baseado em mapa - Probabilidade de entrega das mensagens

Analisando o gráfico gerado (6.2) observar-se que no quesito benefício, considerado nesse trabalho como a probabilidade média de entrega das mensagens, os protocolos baseados em histórico de encontros foram mais eficientes que os protocolos tradicionais, *Epidemic* e *Prophet*, sendo ainda superior o GREASE em relação ao protocolo VeloSent.

Após essa primeira etapa de simulação foram colhidas informações sobre o custo que cada protocolo produz na rede, representado, nesse trabalho, pela a quantidade de recursos de rede utilizados, como armazenamento e energia. Foi escolhido então, como custo, o número de transmissões efetuadas pelos nós, já que os dois parâmetros anteriores (energia e armazenamento) estão diretamente ligados a quantidade de transmissões, e são elas que permitem a troca de mensagens entre os nós. Quanto maior o número de transmissões, maior o consumo de energia dos nós, e quanto maior o número de mensagens transmitidas e armazenadas maior o consumo de armazenamento.

A partir dessa ótica foram analisados nas simulações, para efeito de custo, o total de transmissões necessárias para que cada protocolo conseguisse obter a probabilidade de entrega apresentada na figura 6.2. Com esses dados foi gerado o gráfico do custo de cada protocolo, apresentado na figura 6.3 e posteriormente através desses dois gráficos, o gráfico do custo pelo benefício de cada um, apresentado na figura 6.4.

Importante observar que, o gráfico do custo pelo benefício apresenta o número de transmissões iniciadas que são necessárias para 1% de probabilidade de entrega. Sendo assim, quanto maior for o custo pelo benefício do protocolo, maior será a utilização de recursos de rede em relação ao número médio de mensagens entregues, ou seja, pior será seu desempenho, uma vez que estará consumindo mais recursos para poder rotear mensagens até seu destino final.

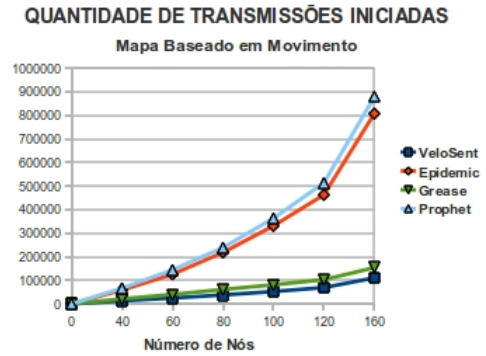

Figura 6.3: Movimento baseado em mapa - Quantidade de transmissões iniciadas.

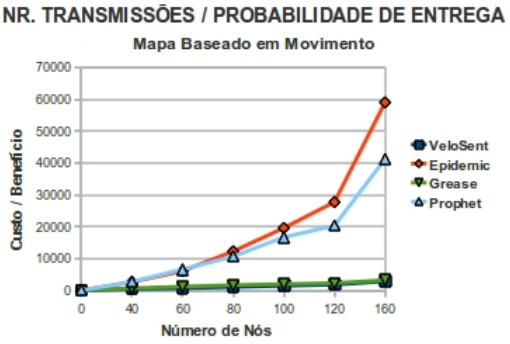

Figura 6.4: Movimento baseado em mapa - Custo / Benefício.

Pelos gráficos das figuras 6.3 e 6.4 é possível observar, primeiramente, que o custo produzido pelo protocolo VeloSent é o menor em comparação aos outros 3 protocolos, ou seja, a quantidade de recursos de rede gastos por ele são menores. Pelo segundo gráfico, que apresenta o custo/benefício dos protocolos, ou seja, o número de transmissões iniciadas pela probabilidade média de entrega das mensagens, é possível constatar que os protocolos GREASE e VeloSent possuem desempenho muito próximos e também são muito superiores ao protocolos *Epidemic* e *Prophet*. Devido essa semelhança de desempenho, foi produzido um quarto gráfico, apresentado na figura 6.5, que permitiu uma análise mais detalhada dos dois, para tal, utilizaram-se apenas os dados desses dois protocolos. Por fim, um quinto gráfico foi gerado, para análise do número médio de saltos que cada mensagem necessita até alcançar o seu destino, esse gráfico é apresentado na figura 6.6.

Figura 6.5: Movimento baseado em mapa - Total de transmissões / Probabilidade de entrega.

Figura 6.6: Movimento baseado em mapa - Número médio de saltos das mensagens.

Através dos gráficos das figuras 6.5 e 6.6, analisando inicialmente o de custo/benefício, é possível perceber que o protocolo VeloSent é o que possui melhor desempenho entre os 4 analisados, inclusive superior ao protocolo GREASE. Pode-se destacar também que o número médio de saltos de uma mensagem ao longo do seu roteamento é o menor entre todos, exigindo uma menor capacidade de armazenamento dos nós, assim como um menor gasto de energia com transmissões.

Após essa primeira fase da simulação foi possível obter resultados satisfatórios em comparação aos outros protocolos, contudo, como mencionado anteriormente, o objetivo do estudo, aqui proposto, não é a obtenção de um protocolo de roteamento adequado a um determinado cenário ou contexto, mas sim demonstrar que um protocolo capaz de analisar e se adaptar as condições na qual está inserido, ou seja, seu contexto pode apresentar bons resultados quando utilizado em ambientes tolerantes a atrasos e desconexões.

A partir desse pensamento, dois novos ambientes, com características diferentes desse primeiro, foram utilizados para verificar se o protocolo proposto seria capaz de continuar obtendo resultados satisfatórios. O novo cenário escolhido, ao contrário do anterior, não possui ruas, nem tão pouco movimentos padrões como os de linhas de trens e ônibus, mas sim rotas aleatórias. Nesse novo cenário, denominado de “Rotas Aleatórias” (*Random Waypoint*), os nós

trafegam em um ambiente aberto, sem obstáculos, como edifícios e casas utilizados no cenário anterior.

6.3.2 Segundo Cenário: Rotas Aleatórias (*Random WayPoint*)

Diferentemente do cenário anterior, os nós não foram divididos em vários grupos, visto que nesse ambiente todos os nós possuem padrões aleatórios de movimento, não obedecendo uma lógica como ruas ou linhas de trem e ônibus. O tempo de simulação utilizado foi, assim como anteriormente, de 12 horas, foram simulados nesse cenário os mesmos 4 protocolos utilizados anteriormente, e a cada nova simulação aumento-se o número de nós (aumentou-se a densidade da rede) para que fosse possível obter uma quantidade de dados suficiente para visualizar o comportamento de cada um dos protocolos. Na figura 6.7 é apresentado o cenário utilizado quando executado pelo simulador The ONE. Importante observar que as setas mostradas foram adicionadas a imagem com intuito de especificar como ocorrem os deslocamentos dos nós quando estes movem-se em rotas aleatórias.

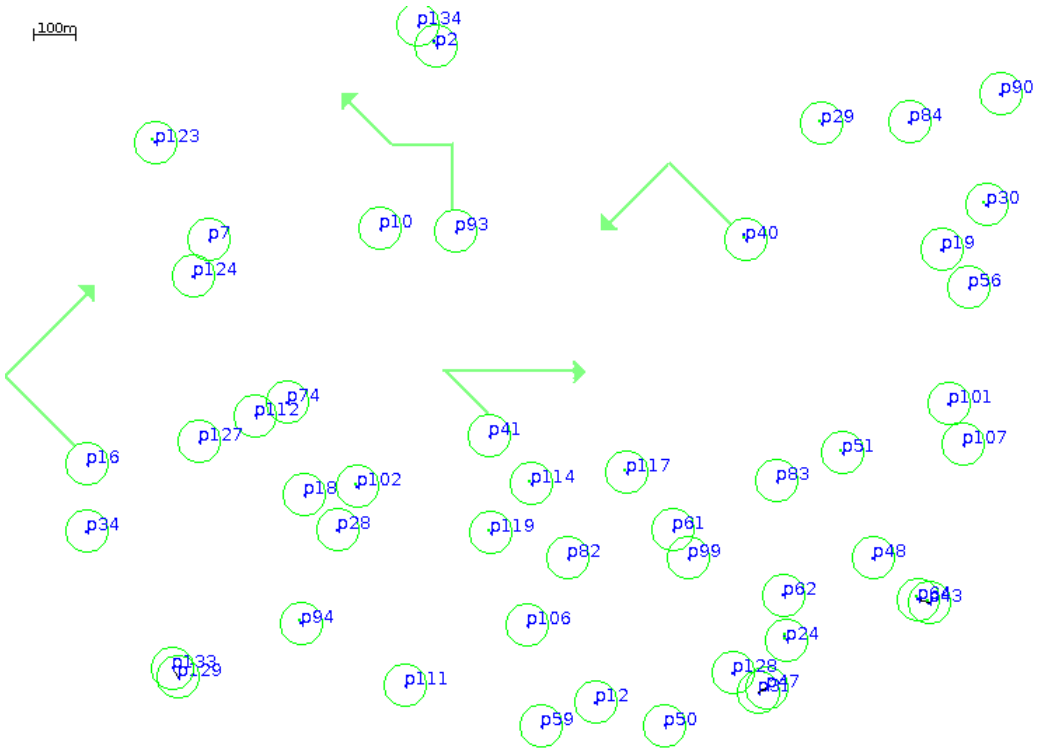

Figura 6.7: Rota Aleatória (*Random Waypoint*) - Movimento em um ambiente aberto - The ONE.

Assim como anteriormente efetuou-e uma análise inicial dos benefício de cada protocolo e dos seus custos para o ambiente proposto. Como já mencionado e explicado nesse

documento, o custo e o benefício são representados, respectivamente, pela probabilidade de entrega das mensagens e pelo o número total de transmissões produzidas para que as entregas ocorram. Portanto, os dois primeiros gráficos obtidos no ambiente de “Rotas Aleatórias” através dos dados coletados nas simulações são apresentados nas figuras 6.8 e 6.9.

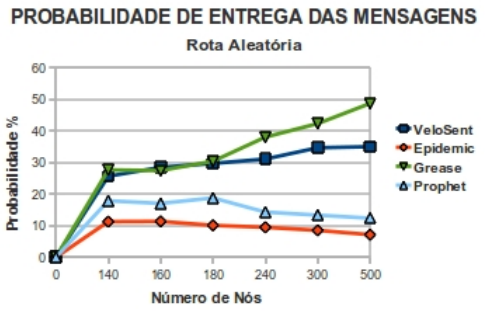

Figura 6.8: Rota Aleatória - Probabilidade de entrega das mensagens

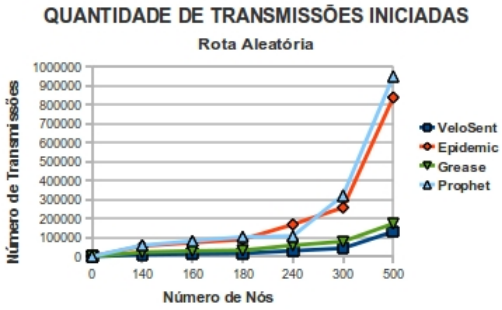

Figura 6.9: Rota Aleatória - Quantidade de transmissões iniciadas

Analisando os gráficos das figuras 6.8 e 6.9, percebe-se que apesar da variação do ambiente, o protocolo VeloSent conseguiu adaptar-se ao mesmo, não modificando completamente seu comportamento em relação aos outros protocolos utilizados, mantendo-se como o protocolo que utiliza a menor quantidade de recursos de rede. A partir dos valores do custo e do benefício dos protocolos foram produzidos, assim como no ambiente anterior, os gráficos das figuras 6.10, 6.11 e 6.12.

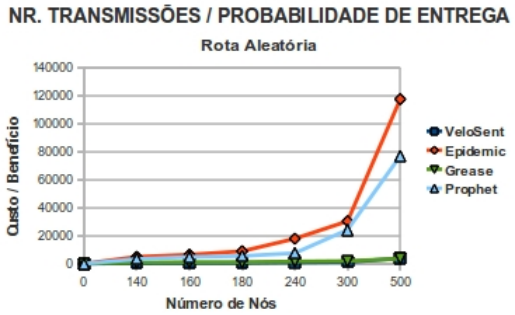

Figura 6.10: Rota Aleatória - Custo / Benefício.

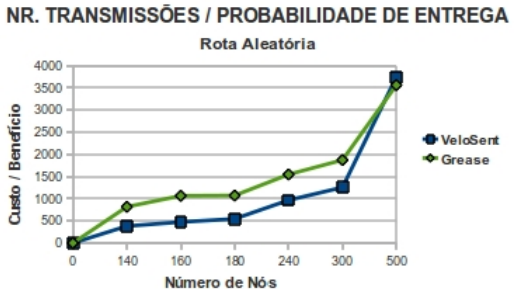

Figura 6.11: Rota Aleatória - Total de transmissões / Probabilidade de entrega.

Figura 6.12: Rota Aleatória - Número de médio de saltos das mensagens.

Semelhante ao cenário anterior, o protocolo VeloSent continuou a obter o melhor desempenho entre os 4 analisados no que diz respeito ao seu custo / benefício, pode-se salientar ainda que o número médio de saltos de uma mensagem ao longo do seu roteamento foi o menor, ou seja, o protocolo proposto continua necessitando de uma quantidade menor de réplicas de mensagens e consequente menor capacidade de armazenamento dos nós da rede.

Entretanto, também observa-se uma certa semelhança no desempenho comparativo entre os 4 protocolos. Por esse motivo foi utilizado um terceiro cenário, onde o movimento dos nós da rede é baseado em pessoas caminhando aleatoriamente (*Random Walk*). O objetivo dessa

terceira simulação foi encontrar um ambiente onde um dos protocolos, anteriormente com pior desempenho, demonstrasse melhora em uma das métricas analisadas: benefício, custo, custo benefício e número médio de saltos. O novo cenário escolhido foi denominado de "Pessoas Caminhando Aleatoriamente" (*Random Walk*).

6.3.3 Terceiro Cenário: Pessoas caminhando aleatoriamente (Random Walk)

Diferentemente do primeiro cenário, no que diz respeito ao padrão de movimento, e assim como no segundo, os nós não foram divididos em vários grupos, já que todos os nós possuem padrões aleatórios de movimento. Porém ao contrário do segundo ambiente, o de "Rotas Aleatórias", os nós não seguem um caminho em linha reta até um ponto aleatoriamente determinado ao iniciarem seus movimentos, mas sim o alternam aleatoriamente ao longo da simulação, como pessoas caminhando sem um destino pré-definido. O tempo de simulação utilizado foi, assim como nos dois cenários anteriores, de 12 horas, foram simulados nesse cenário os mesmos 4 protocolos já utilizados anteriormente, e a cada nova simulação aumento-se o número de nós (densidade da rede) para que fosse possível obter uma quantidade de dados suficiente para visualizar o comportamento de cada um dos protocolos.

Analogamente aos outros dois cenários efetuou-se inicialmente uma análise dos benefícios e dos custos de cada protocolo dentro do ambiente proposto. Os dois primeiros gráficos obtidos através dos dados coletados nas simulações para o ambiente de "Pessoas Caminhando Aleatoriamente" são apresentados nas figuras 6.13 e 6.14.

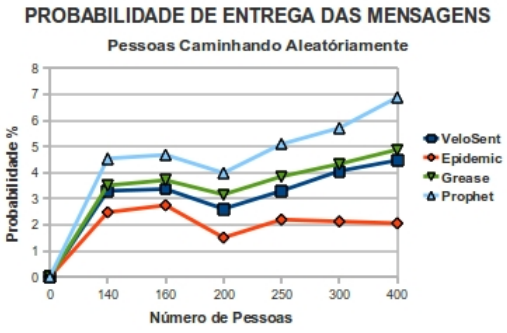

Figura 6.13: Pessoas caminhando aleatoriamente - Probabilidade de entrega das mensagens

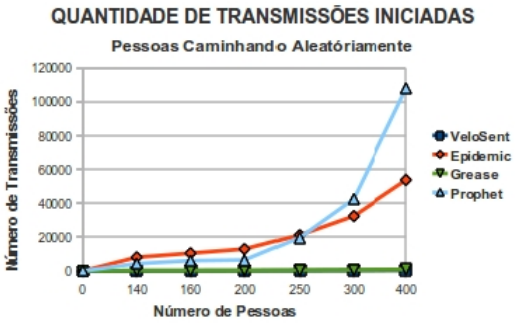

Figura 6.14: Pessoas caminhando aleatoriamente - Quantidade de transmissões iniciadas

Através do gráfico gerado, apresentado na figura 6.13, é possível observar uma diferença no desempenho do protocolo *Prophet* em relação ao seu comportamento nos dois outros cenários quando comparado aos outros protocolos, ou seja, nesse cenário o *Prophet* é mais eficiente no que diz respeito à métrica de probabilidade média de entrega de mensagens, o que prova que alguns protocolos têm melhores desempenhos em determinados ambientes para determinadas métricas.

Contudo se observarmos o gráfico da figura 6.14, fica evidente, que apesar do bom desempenho na entrega de mensagens, quando comparado aos outros protocolos, o custo para que tal fato ocorra é bem superior ao custo dos protocolos GREASE e VeloSent, evidenciando que os protocolos apresentam uma variação muito grande de comportamento, quando o ambiente a sua volta também varia, ou seja, a adaptação ao mesmo nem sempre ocorre.

Assim como efetuado anteriormente, com os valores do custo e benefício de cada protocolo, bem como com o número médio de saltos das mensagens ao longo do seu roteamento, foram produzidos mais três gráficos para análise, apresentados nas figuras 6.15, 6.16 e 6.17.

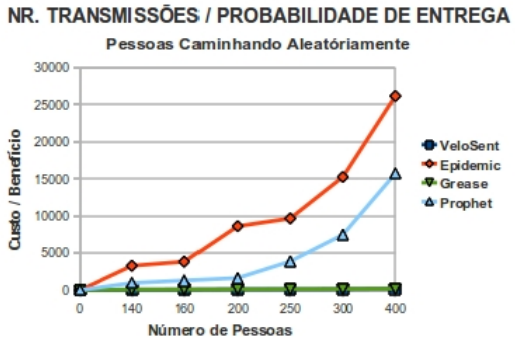

Figura 6.15: Caminhando aleatoriamente - Custo / Benefício.

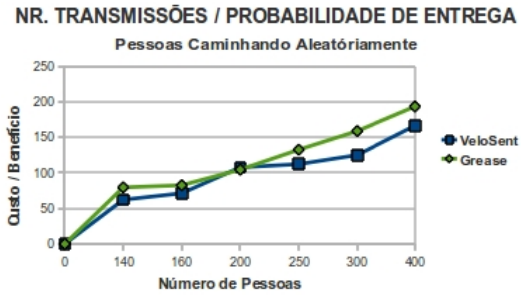

Figura 6.16: Caminhando aleatoriamente - Total de transmissões / Probabilidade de entrega.

Apesar do melhor desempenho do protocolo Prophet no quesito benefício - probabilidade média de entrega das mensagens, ao efetuar-se uma análise mais abrangente é possível constatar que o protocolo VeloSent, proposto nesse trabalho, continua a possuir o melhor desempenho em custo benefício entre os 4 apresentados. O protocolo VeloSent mantém-se melhor adaptado e utilizando de forma reduzida, quando comparado aos demais, os recursos da rede, além de necessitar de um número médio menor de saltos para que as mensagens sejam roteadas.

6.4 Análise dos Resultados

O Protocolo de Roteamento Sensível ao Contexto: posição, velocidade e sentido, proposto nesse documento, obteve resultados satisfatórios para os objetivos da pesquisa efetuada, visto que o objetivo não era encontrar um ambiente específico, nem mesmo uma métrica em especial para o qual o protocolo se sobressaísse perante aos outros, mas sim demonstrar através do processo de simulação que a utilização da sensibilidade e adaptação ao contexto é uma possibilidade considerável dentro de ambientes caracterizados pelos constantes atrasos e desconexões.

Sendo assim, observou-se, durante as simulações dos três cenários, que o comporta-

Figura 6.17: Caminhando aleatoriamente - Número de médio de saltos das mensagens.

mento do protocolo VeloSent obteve o melhor desempenho em custo benefício em comparação aos outros protocolos, ou seja, apesar da sua análise sobre o contexto ter sido reduzida apenas a posição, velocidade e sentido dos nós, se considerarmos todas as informações que englobam o contexto de uma DTN, já foi possível obter-se resultados consideráveis.

No primeiro cenário onde os movimentos dos nós foram baseados no mapa de uma cidade o protocolo de VeloSent obteve um desempenho bom com relação a probabilidade média de entrega das mensagens, superando os protocolos *Epidemic* e *Prophet*, ficando atrás apenas do protocolo GREASE. Considerando que a metodologia desse protocolo considera que as trajetórias dos nós são retas e analisa o sentido destas para determinar qual nó segue na direção do destino, porém ignora o fato, da existência de ruas finitas, ou seja, que em um determinado momento os nós trocarão seu sentido, os resultados obtidos foram muito bons. Ainda nesse cenário se avaliarmos os custos trazidos pelo protocolo VeloSent, em relação aos outros, observaremos que o mesmo possui o melhor desempenho, utilizando uma quantidade menor de transmissões, e consequentemente menos gasto de energia e armazenamento.

O segundo cenário apresenta resultados parecidos com o primeiro, ou seja, o protocolo proposto manteve seu desempenho em relação a probabilidade de entrega das mensagens, perdendo apenas para o GREASE, e continuou sendo o protocolo com melhor resultado no que diz respeito ao número de transmissões efetuadas, utilizando a menor quantidade de recursos da rede e precisando de uma menor quantidade média de saltos das mensagens ao longo do seu roteamento, demonstrando-se também o melhor protocolo em custo/benefício.

Contudo observando, de forma mais criteriosa, o gráfico que representa o custo/benefício dos protocolos de acordo com o aumento da densidade da rede, ou seja, o número de nós que a compõem, é possível constatar que a partir de uma determinada quantidade de nós, aproximadamente 480, o protocolo VeloSent tende a perder para o protocolo GREASE o posto de melhor no quesito custo/benefício, ou seja, pode-se constatar que a densidade de determinados

ambientes é uma informação de contexto que influencia diretamente no comportamento dos protocolos, tornando-se uma possibilidade, a adição da mesma ao protocolo aqui apresentado em um trabalho futuro, o que lhe tornaria adaptável também a esse parâmetro.

O terceiro e último cenário apresentou uma nova perspectiva sobre a avaliação dos 4 protocolos, ao contrário dos anteriores o protocolo *Prophet* superou todos os outros com relação a probabilidade de entrega das mensagens. Importante observar que o número de contatos nesse ambiente é muito reduzido se comparado aos anteriores, esse fato permite concluir que outros parâmetros de contexto, como número de conexões e vizinhos devem ser considerados também como informações de contexto, e assim, utilizados pela metodologia de roteamento dos protocolos. Contudo, nesse mesmo ambiente, o protocolo VeloSent obteve melhor desempenho, assim como nos cenários anteriores, nos quesitos de número de transmissões efetuadas, quantidade média de saltos das mensagens e no custo / benefício.

7 Conclusão

Pelos resultados obtidos, o protocolo VeloSent cumpriu o objetivo deste trabalho, introduzindo a utilização do contexto como metodologia eficiente no roteamento de mensagens em redes tolerantes a atrasos e desconexões. Esse fato pode ser verificado, principalmente, pelo bom desempenho apresentado durante todos os diferentes ambientes de simulação utilizados. Ao longo do desenvolvimento da pesquisa o protocolo utilizou-se, apenas, do contexto: posição, velocidade e sentido, e através destes parâmetros foi possível produzir a adaptação as características dos cenários simulados, sendo possível obter os melhores resultados de custo/benefício entre os 4 protocolos avaliados. Nesse sentido, constatamos que a aplicação da técnica de sensibilidade ao contexto no ambiente de redes tolerantes a atrasos e desconexões deva seguir três caminhos diferentes:

1. utilização da sensibilidade e adaptação ao contexto dentro dos protocolos já conhecidos na literatura, para que esses tornem-se mais eficientes e adaptáveis a vários cenários;
2. criação de uma camada de software capaz de obter e analisar o meio onde está inserida, e a partir disso selecionar um protocolo, dentre os já apresentados e avaliados na literatura, que melhor se adapte as condições percebidas do ambiente;
3. desenvolvimento de novos protocolos de roteamento, com seu funcionamento central baseado nas principais características de contexto das redes DTN, características estas que ainda precisam ser pesquisadas, detectadas e listadas.

Portanto, a ideia da utilização do contexto em redes tolerantes a atrasos e conexões ainda está em seu estágio inicial, e durante os próximos anos, através de trabalhos de pesquisa e publicação de artigos, ela tende a amadurecer e apresentar respostas sobre sua eficiência e sobre quais são as melhores formas da sua utilização. Entretanto, independentemente disso, é fato que a sensibilidade ao contexto é uma técnica com grande potencial quando utilizada como metodologia para roteamento de mensagens em DTNs.

O simulador The ONE utilizado nessa pesquisa, que foi acrescido dos protocolos VeloSent e GREASE pode ser obtido através do acesso ao link:

http://www.gileduardo.com.br/downloads/ONE.

7.1 Trabalhos Futuros

A utilização da sensibilidade ao contexto mostrou-se eficiente como metodologia para o roteamento de mensagens em redes tolerantes a atrasos e conexões. Contudo o contexto analisado pelo protocolo VeloSent limitou-se a três dados: posição, velocidade e sentido dos nós, deixando de lado, nesse primeiro momento, outras informações tão importantes quanto as analisadas.

Muitos outros dados de contexto podem ser agregados ao VeloSent em trabalhos futuros, dentre os quais destacam-se: a quantidade de energia restante (nível de bateria) dos nós da rede, visto que o protocolo efetua cálculos estimativos do tempo necessário para que os nós vizinhos encontrem-se com os destinos das mensagens a serem roteadas. Contudo o VeloSent não leva em consideração o tempo de bateria dos seus vizinhos, ou seja, se o próximo nó escolhido para o roteamento da mensagem terá energia o suficiente para chegar até o ponto de encontro com o destino.

Além do nível de energia, outro dado de contexto relevante, que foi possível identificar com as simulações, como fator a ser utilizado para melhor adaptação do VeloSent, foi a densidade da rede. O aumento do número de nós, em determinados modelos de movimento, tem influência significativa na forma como o VeloSent utiliza os recursos de rede e como encaminha as mensagens até seu destino final.

Outra possibilidade de trabalho futuro é explorar o dado de contexto referente ao raio de ação da antena de transmissão dos nós. Entre os cálculos matemáticos utilizados para definição do próximo nó vizinho a receber a mensagem, tem-se, de forma estimada, o momento em que cada vizinho estará mais próximos do destino, esta distância pode ser comparada ao raio de alcance da antena dos nós para definir se haverá a possibilidade de transmissão das mensagens ou não.

Por fim, ainda poderiam ser estudados e desenvolvidos futuramente:

- Análisar e considerar a velocidade, posição e sentido dos nós em três dimensões (3D);

- Analisar o movimento dos nós não apenas levando em consideração a derivada da velocidade e da direção (movimento retilíneo), mas sim encontrar outras funções que possam definir a trajetória dos nós com maior precisão;

- Desenvolver uma técnica de TTL (tempo de vida) para os dados de contexto armazenados nas tabelas locais, eliminando dados estimativos muito antigos sobre a posição dos nós da rede.

- Definir uma estimativa mais precisa sobre o momento em que os vizinhos estarão mais próximos do destino (fase 3 - etapa 5), visto que o VeloSent faz uma consideração aproximada, porém cálculos matemáticos mais apurados (derivada(t)=0) podem obter uma estimativa mais precisa.

- Analisar e mensurar os custos que as trocas de mensagens iniciais, efetuadas para obtenção dos dados de contexto e para o cálculo da idade dos contatos, trazem para rede.

Referências Bibliográficas

ABOWD, G. D. et al. Cyberguide: a mobile context-aware tour guide. *Wirel. Netw.*, Kluwer Academic Publishers, Hingham, MA, USA, v. 3, n. 5, p. 421–433, out. 1997.

ABOWD, G. D. et al. Context-awareness in wearable and ubiquitous computing. In: *ISWC*. [S.l.: s.n.], 1997. p. 179–180.

ASTHANA, A.; CRAVATTS, M.; KRZYZANOWSKI, P. An indoor wireless system for personalized shopping assistance. In: *In Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*. [S.l.]: IEEE Computer Society Press, 1994. p. 69–74.

BAHL, P.; PADMANABHAN, V. N.; BALACHANDRAN, A. *Enhancements to the RADAR User Location and Tracking System*. [S.l.], 2000.

BENNETT, F. et al. Teleporting - making applications mobile. In: *In Proceedings of 1994 Workshop on Mobile Computing Systems and Applications*. [S.l.]: IEEE Computer Society Press, 1994. p. 82–84.

BETTSTETTER, C. Smooth is better than sharp: a random mobility model for simulation of wireless networks. In: *Proceedings of the 4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*. New York, NY, USA: ACM, 2001. (MSWIM '01), p. 19–27.

BOARI, M. et al. Middleware for automatic dynamic reconfiguration of context-driven services. *Microprocess. Microsyst.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 32, n. 3, p. 145–158, maio 2008.

BROCH, J. et al. A performance comparison of multi-hop wireless ad hoc network routing protocols. In: *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*. New York, NY, USA: ACM, 1998. (MobiCom '98), p. 85–97.

BROWN, M. Supporting user mobility. In: *In Proceedings of IFIP World Conference on Mobile Communications*. [S.l.]: Chapman and Hall, 1996. p. 69–77.

BROWN, P.; BOVEY, J.; CHEN, X. Context-aware applications: from the laboratory to the marketplace. *Personal Communications, IEEE [see also IEEE Wireless Communications]*, v. 4, n. 5, p. 58–64, 1997.

BROWN, P. J. Triggering information by context. *Personal Technologies*, Springer-Verlag, v. 2, n. 1, p. 1–9, September 1998.

CAMP, T.; BOLENG, J.; DAVIES, V. A survey of mobility models for ad hoc network research. *Wireless Communications Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, v. 2, p. 483–502, 2002.

CAVALCANTI, E. R. *Avaliação de Modelos de Mobilidade em Redes Ad Hoc Sem Fio*. Dissertação (Mestrado) — Universidade Federal de Campina Grande, Coordenação de Pos-Graduação em Informática, Campina Grande, Paraíba, Brasil, 2009.

CHEN, G.; KOTZ, D. *A Survey of Context-Aware Mobile Computing Research*. Hanover, NH, USA, 2000.

COOPERSTOCK, J. R. et al. Evolution of a reactive environment. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1995. (CHI '95), p. 170–177.

DAVIES, N. et al. *Developing A Context Sensitive Tourist Guide*. 1998.

DAVIES, V. A. *Evaluating mobility models within an ad hoc network*. Dissertação (Mestrado) — Colorado School of Mines, 2000.

DEY, A. K. Context-aware computing: The cyberdesk project. In: *AAAI 1998 Spring Symposium on Intelligent Environments*. Palo Alto: AAAI Press., 1998. p. 51–54.

DEY, A. K. Understanding and using context. *Personal Ubiquitous Comput.*, Springer-Verlag, London, UK, UK, v. 5, n. 1, p. 4–7, jan. 2001. ISSN 1617-4909. Disponível em: <http://dx.doi.org/10.1007/s007790170019>.

DEY, A. K.; ABOWD, G. D. Cyberdesk: The use of perception in context-aware computing. In: *1st Workshop on Perceptual User Interfaces*. [S.l.: s.n.], 1997. p. 26–27.

DEY, A. K.; ABOWD, G. D. Towards a better understanding of context and context-awareness. In: *In HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*. [S.l.]: Springer-Verlag, 1999. p. 304–307.

DEY, A. K.; ABOWD, G. D.; WOOD, A. Cyberdesk: a framework for providing self-integrating context-aware services. In: *Proceedings of the 3rd international conference on Intelligent user interfaces*. New York, NY, USA: ACM, 1998. (IUI 98), p. 47–54.

DEY, A. K. et al. The conference assistant: Combining context-awareness with wearable computing. In: *Proceedings of the 3rd IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 1999. (ISWC '99), p. 21–.

ELROD, S. et al. Responsive office environments. *Commun. ACM*, ACM, New York, NY, USA, v. 36, n. 7, p. 84–85, jul. 1993.

FALL, K. A delay-tolerant network architecture for challenged internets. In: *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2003. (SIGCOMM '03), p. 27–34.

FICKAS, S.; KORTUEM, G.; SEGALL, Z. Software organization for dynamic and adaptable wearable systems. In: *Proceedings of the 1st IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 1997. (ISWC '97), p. 56–.

GAEDKE, M. et al. Web content delivery to heterogeneous mobile platforms. In: *ER Workshops*. [S.l.: s.n.], 1998. p. 205–217.

GREENFIELD, A. *Everyware: The Dawning Age of Ubiquitous Computing, New Riders Press*. 2011.

GROSSGLAUSER, M.; VETTERLI, M. Locating mobile nodes with ease: learning efficient routes from encounter histories alone. *IEEE/ACM Trans. Netw.*, IEEE Press, Piscataway, NJ, USA, v. 14, n. 3, p. 457–469, jun. 2006.

HONG, J. I. The context fabric: an infrastructure for context-aware computing. In: *CHI '02 extended abstracts on Human factors in computing systems*. New York, NY, USA: ACM, 2002. (CHI EA '02), p. 554–555.

HONG, J. I.; LANDAY, J. A. An infrastructure approach to context-aware computing. *Hum.-Comput. Interact.*, L. Erlbaum Associates Inc., Hillsdale, NJ, USA, v. 16, n. 2, p. 287–303, dez. 2001.

HONG, X. et al. A group mobility model for ad hoc wireless networks. In: *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*. New York, NY, USA: ACM, 1999. (MSWiM '99), p. 53–60.

HULL, R.; NEAVES, P.; BEDFORD-ROBERTS, J. Towards situated computing. In: *Proceedings of the 1st IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 1997. (ISWC '97), p. 146–.

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

KICIMAN, E.; FOX, A. Using dynamic mediation to integrate cots entities in a ubiquitous computing environment. In: *IN SECOND INTERNATIONAL SYMPOSIUM ON HANDHELD AND UBIQUITOUS COMPUTING 2000*. [S.l.]: Springer-Verlag, 2000. p. 248–254.

KORTUEM, G.; SEGALL, Z.; BAUER, M. Context-aware, adaptive wearable computers as remote interfaces to 'intelligent' environments. In: *Proceedings of the 2nd IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 1998. (ISWC '98), p. 58–65.

LINDGREN, A.; DORIA, A.; SCHELÉN, O. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 7, n. 3, p. 19–20, jul. 2003.

MAO, Z. M.; KATZ, R. H.; BREWER, E. A. *Fault-tolerant, Scalable, Wide-Area Internet Service Composition*. Berkeley, CA, USA, 2001.

MARMASSE, N.; SCHMANDT, C. Location-aware information delivery with commotion. In: . [S.l.]: Springer, 2000. p. 157–171.

MORAES, F. P. d. A. e. A. S. L. P. Renato M. de. Uma proposta para estabilizar o modelo de mobilidade random waypoint em redes ad hoc sem fio. In: *XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. Recife, PE, Brasil: SBRC, 2009. p. 147–160.

MURTHY, C. S. R.; MANOJ, B. *Ad Hoc Wireless Networks: Architectures and Protocols*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2004.

NOBLE, B. et al. Agile application-aware adaptation for mobility. In: *SOSP*. [S.l.: s.n.], 1997. p. 276–287.

OLIVEIRA, C. T. d. et al. Uma proposta de roteamento probabilístico para redes tolerantes a atrasos e desconexões. In: *XXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. Rio de Janeiro, RJ, Brasil: SBRC, 2008. p. 735–748.

OLIVEIRA, C. T. de. *Uma Proposta de Roteamento Probabilístico para Redes Tolerantes a Atrasos e Desconexões*. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro, Programa de Pos-Graduação em Engenharia, Rio de Janeiro, RJ, Brasil, 2008.

OLIVEIRA, E. C. R. de. *Roteamento Adaptativo a Contextos para Redes Tolerantes a Atrasos e Desconexões*. Tese (Doutorado) — Universidade Federal Fluminense, Programa de Pos-Graduação em Computação, Niterói, RJ, Brasil, 2008.

OLIVEIRA, T. R. de. *Um Modelo de Gerenciamento de Segurança Adaptativo para Redes de Emergência*. Dissertação (Mestrado) — Universidade Federal de Minas Gerais, Programa de Pos-Graduação em Ciência da Computação, Belo Horizonte, MG, Brasil, 2010.

PASCOE, M. J. Adding generic contextual capabilities to wearable computers. In: *Proceedings of the 2nd IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 1998. (ISWC '98), p. 92–.

POTNIS, N.; MAHAJAN, A. Mobility models for vehicular ad hoc network simulations. In: *Proceedings of the 44th annual Southeast regional conference*. New York, NY, USA: ACM, 2006. (ACM-SE 44), p. 746–747.

PRIYANTHA, N.; CHAKRABORTY, A.; BALAKRISHNAN, H. The cricket location-support system. In: *6th Annual International Conference on Mobile Computing and networking*. New York, NY, USA: [s.n.], 2000. p. 32–43.

RAENTO, M. et al. ContextPhone: a prototyping platform for context-aware mobile applications. *Pervasive Computing, IEEE*, v. 4, p. 51–59, 2005.

REKIMOTO, J.; AYATSUKA, Y.; HAYASHI, K. Augment-able reality: Situated communication through physical and digital spaces. In: *IN PROCEEDINGS OF THE 2 ND INTERNATIONAL SYMPOSIUM ON WEARABLE COMPUTERS*. [S.l.: s.n.], 1998. p. 68–75.

RYAN, N. S.; PASCOE, J.; MORSE, D. R. Enhanced Reality Fieldwork: the Context-aware Archaeological Assistant. In: GAFFNEY, V.; LEUSEN, M. van; EXXON, S. (Ed.). *Computer Applications in Archaeology 1997*. Oxford: Tempus Reparatum, 1998. (British Archaeological Reports).

SALBER, D.; DEY, A. K.; ABOWD, G. D. *Ubiquitous Computing: Defining an HCI Research - Agenda for an Emerging Interaction Paradigm*. [S.l.], 1998.

SALBER, D.; DEY, A. K.; ABOWD, G. D. The context toolkit: aiding the development of context-enabled applications. In: *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*. New York, NY, USA: ACM, 1999. (CHI '99), p. 434–441.

SCHILIT, B.; ADAMS, N.; WANT, R. Context-aware computing applications. In: *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*. Washington, DC, USA: IEEE Computer Society, 1994. (WMCSA '94), p. 85–90. ISBN 978-0-7695-3451-0. Disponível em: <http://dx.doi.org/10.1109/WMCSA.1994.16>.

SCHILIT, B. N.; THEIMER, M. M. Disseminating active map information to mobile hosts. *Network, IEEE*, IEEE, v. 8, n. 5, p. 22–32, set. 1994.

SCHMIDT, A. et al. Advanced interaction in context. In: *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*. London, UK, UK: Springer-Verlag, 1999. (HUC '99), p. 89–101.

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

VAHDAT, A.; BECKER, D. *Epidemic Routing for Partially-Connected Ad Hoc Networks*. [S.l.], 2000.

WANT, R. et al. The active badge location system. *ACM Trans. Inf. Syst.*, ACM, New York, NY, USA, v. 10, n. 1, p. 91–102, jan. 1992.

WANT, R. et al. The parctab ubiquitous computing experiment. *IEEE PERSONAL COMMUNICATIONS*, v. 2, p. 28–43, 1995.

WARD, A.; JONES, A.; HOPPER, A. A new location technique for the active office. *IEEE Personal Communications*, v. 4, p. 42–47, 1997.

WARTHMAN, F. *Delay-Tolerant Networks (DTNs) - A Tutorial*. 2003. Disponível em: <http://www.ipnsig.org/reports/DTN_Tutorial11.pdf>.

YOON, J.; LIU, M.; NOBLE, B. Random waypoint considered harmful. In: *INFOCOM*. [S.l.: s.n.], 2003.