**UNIVERSIDADE FEDERAL DE SANTA CATARINA**
**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

Underléa Cabreira Corrêa

# Proposta de um *Framework* de Roteamento para Redes Móveis *Ad-hoc*

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Orientador:
Professor Vitório Bruno Mazzola, Dr.

Florianópolis, 18 de fevereiro de 2005.

# Proposta de um *Framework* de Roteamento para Redes Móveis *Ad-hoc*

Underléa Cabreira Corrêa

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração **Sistemas de Computação** e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

**Prof. Dr. Vitório Bruno Mazzola**
Orientador

**Prof. Dr. Raul Sidnei Wazlawick**
Coordenador do Programa de Pós-graduação em Ciência da Computação

Banca Examinadora

**Prof. Dr. Vitório Bruno Mazzola**
Presidente

**Prof. Dr. Antônio Augusto Fröhlich**
UFSC

**Prof. Dr. Carlos Barros Montez**
UFSC

**Prof. Dr. Ricardo Pereira e Silva**
UFSC

**Prof. Dr. Rômulo Silva de Oliveira**
UFSC

**Prof. Phd. Mario Antônio Ribeiro Dantas**
UFSC

*"...Pra essa terra onde nasci e amo tanto,*
*mas que volto qualquer dia a qualquer hora..."*
João Chagas Leite

# Agradecimentos

Agradeço a Deus e a todos que de uma forma ou outra colaboraram para concretização desta pesquisa. Em especial dedico a minha eterna gratidão e profundo respeito:

Ao meu querido e estimado orientador Professor Vitório Bruno Mazzola, agradeço sinceramente pela oportunidade, pela dedicação, compreensão, colaboração, preocupação, cuidado e confiança a mim concedida.

Ao meu estimado amigo e co-orientador professor Mario Antônio Ribeiro Dantas, peça fundamental no amadurecimento deste trabalho, agradeço também por toda atenção, preocupação, confiança, além da sincera amizade.

Ao Guto, agradeço pelo amor, confiança, apoio, zelo, amizade e principalmente por ser e se fazer todos os dias presente, como uma das pessoas mais importantes da minha vida.

Aos meus pais, irmãos e avós, e em especial a minha mãe Maria de Fátima Cabreira que sempre se preocupou com minha educação e me ensinou que responsabilidade, trabalho e dedicação são requisitos mínimos necessários para se obter bons resultados.

Ao José Augusto Strano Sala e Elaine Maria Fiori Sala, agradeço por todo apoio amizade e preocupação.

Aos meus amigos, pessoas que tornaram essa caminhada mais amena sempre dispostos a diminuírem tristezas e somarem alegrias. Entre esses, agradeço em especial a minha fiel “escudeira”, Fabricia Lemos de Faria, que durante esta empreitada sempre esteve ao meu lado, dividindo e compartilhando sua preciosa amizade, atenção, compreensão, dúvidas, alegrias e anseios, me ajudando a construir novos pensamentos e tornando meus dias mais alegres. Ao Guilherme Bertoni Machado por sua amizade e principalmente pelos momentos que parou para refletir, e junto comigo tentou descobrir o sentido desse trabalho. A Madalena Pereira da Silva e Fábio Schmiths Tani, agradeço por toda atenção a qual sempre gentilmente a mim dedicaram, compartilhando seus conhecimentos e inestimável amizade.

# Sumário

## LISTA DE ACRÔNIMOS

| | |
|---|---|
| MANET | *Mobile Ah Hoc Network* |
| FRAd-hoc | *Framework de Roteamento Ad-hoc* |
| UML | *Unified Mobile Language* |
| IETF | *Internet Engeneering Task Force* |
| WPAN | *Wireless Personal Network* |
| DoD | *Departamente of Defence* |
| DARPA | *Defence Advanced Research Projects Agency* |
| SURAN | *Survavible Adaptative Network* |
| ONR | *Office Naval Research* |
| SCN | *Survavible Communication Networks* |
| IP | *Internet Protocol* |
| QoS | *Quality of Service* |
| TTL | *Time to Live* |
| ISPs | *Internet Service Providers* |
| TCP | *Transfer Control Protocol* |
| LSP | *Link State Packet* |
| GPS | *Global Position System* |
| ZHLS | *Zone Hierarchical Link State* |
| DDR | *Distributed Dynamic Routing* |
| PN | *Preferred Neighbor* |
| ZID | *Zone Identified* |
| NID | *Nodo Identified* |
| Intra_ZT | *Intra Zone Table* |
| Inter_ZT | *Inter Zone Table* |
| ID | *Identified* |
| HARP | *Hybrid Ad hoc Routing Protocol* |
| PREQ | *Packet Request* |
| PREP | *Packet Response* |
| DPKT | *Data Packet* |
| GID | *Gateway Identifed* |
| Ant-AODV | *Ant – Ad hoc On-Demand Vector* |

| | |
|---|---|
| RERR | *Request Error* |
| RREQ | *Rote Request* |
| GNID | *Gateway Node Identified* |
| WN | *Wireless Network* |
| DSDV | *Destination-sequenced Distance Vector* |
| WRP | *Wireless Routing Protocol* |
| GSR | *Global State Routing* |
| FSR | *Fisheye State Routing* |
| STAR | *Source-tree Adaptative Routing* |
| DREAM | *Distance Routing Effect Algorithm for Mobillity* |
| HSR | *Hierachical State Routing* |
| OSLR | *Optimised Link State Routing* |
| TBRPF | *Topology Broadcast Reverse Path Forwarding* |
| DSR | *Dynamic Source Routing* |
| ROAM | *Routing On-Demand Acyclic Mult-path* |
| TORA | *Temporally Ordered Routing Algorithm* |
| RDMAR | *Relative Distance Micro-Discovery Ad-hoc Routing* |
| LAR | *Location-Aided Routing* |
| ARA | *Ant-Colony Based routing Algorithm* |
| ZRP | *Zone Routing Protocol* |
| ZHLS | *Zone-based Hierachical Link State* |
| DST | *Distributed Spanning Trees* |

# LISTA DE FIGURAS

# LISTA DE TABELAS E QUADROS

# RESUMO

A camada de redes numa arquitetura multicamadas tem como papel fundamental o roteamento de pacotes de uma origem a um destino. No entanto, a escolha dos algoritmos que definem as rotas e as estruturas de dados que utilizam, são elementos importantes a serem estabelecidos a nível dessa camada. De modo geral, os algoritmos de roteamento podem ser classificados em adaptativos e não adaptativos. Estes algoritmos, em geral, usam tecnologias como as de roteamento de trajeto mais curto, *flooding*, roteamento com vetor de distância, roteamento *broadcast*, roteamento *multicast*, e roteamento de trajeto de estado de enlace, buscando sempre atender características como, simplicidade, imparcialidade, escolha do melhor trajeto, robustez e outras qualidades desejáveis as quais um algoritmo de roteamento deve satisfazer.

No caso particular das redes sem fio móveis, ou MANETs (*Mobile Ad Hoc Networks)*, como também são conhecidas, os algoritmos de roteamento podem ser classificados em três categorias principais: pró-ativos, que se caracterizam por definir priori a rota entre destino e origem; reativos, que definem a rota sob demanda e híbridos, que combinam características dos dois primeiros.

Por mais que o ambiente MANET seja produtivo, a natureza dinâmica de sua topologia dificulta a realização do roteamento conforme realizado para redes com infra-estrutura fixa, o que se constitui num problema a ser estudado. Devido a esse fator, diversos trabalhos de pesquisa têm sido desenvolvidos com o intuito de oferecer entre outros, um algoritmo de roteamento que defina a topologia da rede atendendo a requisitos qualitativos e quantitativos.

Neste trabalho, é proposto um *framework* orientado a objeto denominado por FR*Ad-hoc* (*Framework* de Roteamento *Ad-hoc*) que agrega as classes com características genéricas ao domínio estudado, para que a partir desse possa-se customizar o desenvolvimento de um algoritmos de roteamento. Muito embora a generalização desse domínio ao qual se chegou até o presente momento, seja bastante reduzida, acredita-se que essa será suficiente para que se possa demonstrar a possibilidade de oferecer reuso de *software* por parte das aplicações que especializarão o FR*Ad-hoc – Framework* de roteamento *ad hoc*.



**Palavras reservadas:** Manet, Algoritmos, Roteamento, *Frameworks* Orientado a Objetos.

## *ABSTRACT*

The network layer in a multilayered architecture has as its main role the routing of packets from a source to a destination. However, the choice of algorithms that define routes and the data structures that they use, are important elements to be considered in this layer. In a general way, the routing algorithms can be classified in adaptive and non-adaptive. These algorithms, usually, use technologies like *shortest path routing*, *flooding*, *distance vector routing*, *broadcast routing*, *multicast routing* and state of link routing , always aiming to fulfill caracteristics such as, simplicity and imparciality.

In the particular case of the mobile wireless networks (MANETs - Mobile Ad hoc Networks), as they are known as well, the routing algorithms can be classified in three main categories: on-demand, that are characterized by definig a route between the destination and the source; Reactives, define a route on demand and hibrids, that combine characteristics from the first two.

As productive as a MANET environment can be productive, the dinamic nature of its topology dificults the accomplishment of the routing as it is done for networks with fixed infrastructure, this creates a problem to be studied. Because of this, many researches have been developed with the purpose of create amongst other, a routing algorithm that defines the network topology that defines a network topology that complies with quantitative and qualitative criteria.

In this work, it is proposed an object oriented framework called FRAd-hoc (Ad-hoc Routing Framework) that aggregates classes with generic characteristics to the studied domain, so that with it the routing algorithm development can be customized. Altough the generalization of this domain, established to the present moment, is quite reduced, it is believed that it will be enough to demonstrate the possibility to allow software reusability by the applications that will especialize the FRAd-hoc.


Key Words : Manet, Routing, Algorithms, Frameworks, Oriented-Objects

# CAPÍTULO I

## INTRODUÇÃO

O desejo em se comunicar à distância e sem fio é muito antigo e retrata as sociedades primitivas as quais se comunicavam através de sinais de fumaça e pombo correio marcando as primeiras iniciativas da comunicação sem fio que viria se tornar factível e cada vez mais eficiente com o auxilio da tecnologia. Desde então grandes cientistas como James Clerk Maxwell e Heinrich Hertz, bem como Guglielmo Marconi, deram sua valiosa contribuição através de suas descobertas, mas foi somente no início do século XXI que se marcou o desenvolvimento e a organização de duas tecnologias de comunicação que tornaria esse desejo factível: redes móveis e Internet (SVENSSON 1999 e KERAMANE 2000). A integração e o desenvolvimento dessas duas tecnologias bem como o notável crescimento da comunicação sem fio em conjunto com a alta tecnologia de *hardware* e *software*, contribuiu notoriamente para a criação de um novo paradigma computacional designado pela sigla MANETs – *Mobile Ad hoc Networks*, tornando a comunicação em qualquer lugar, e a qualquer momento realista.

As MANETs são relativamente novas, todavia esse é um nome que está sendo dado atualmente para uma tecnologia em desenvolvimento a mais de 20 anos através de pesquisas patrocinadas pelo governo norte americano. O conceito de redes de pacote de rádio móvel existe desde os anos 70, não muito antes de iniciar o desenvolvimento da tecnologia de comutação de pacotes, que cresceu dentro do que conhecemos agora como Internet. O Departamento de Defesa dos Estados Unidos (DoD) pesquisava como habilitar essa tecnologia para operar sem as restrições da infra-estrutura fixa (PERKINS 2001).

O projeto inicial contou com a participação da Agência de Pesquisas Avançadas DARPA, com o Exército dos Estados Unidos (*U.S.A Army*) e o escritório de Pesquisa Naval (ONR) com significantes programas de pacote de rádio incluindo entre outros programas de redes adaptativas (SURAN), programas de baixo custo de pacote de rádio e programas de comunicação de redes de salvamento (SCN). Atualmente, MANET é um grupo de pesquisa formado pela IETF, que tem se dedicado a padronizar tecnologia de roteamento móvel *peer-to-peer* em um domínio sem fio puramente móvel que ofereça suporte de serviços IP não orientados a conexão e operações efetivas aos diferentes contextos de uma rede móvel, mantendo o roteamento eficaz (CORSON

1999). Dentre trabalhos futuros desse grupo estão inseridos pesquisas em segurança, interação com protocolos das demais camadas e QoS (*Quality of Service*).

Devido o crescente interesse no desenvolvimento de aplicações para redes móveis *ad hoc* uma vasta quantidade de algoritmos de roteamento tem sido proposta para atender às necessidades do complexo sistema de roteamento MANET. Assim, através do estudo de um conjunto de algoritmos de roteamento verificou-se que dentre as novas propostas, algumas das quais tratadas neste trabalho, torna-se difícil apontar qual dos algoritmos seria suficientemente capaz de superar todas as desvantagens apresentadas por algoritmos anteriores e ainda oferecer um roteamento eficiente durante todo o período de formação de uma MANET. A dificuldade é justificada pela dinamicidade de um tal sistema, onde características como conectividade, quantidade de nodos, mobilidade, não são fatores constantes nessas redes. Por isso imaginamos a possibilidade de se obter mais de um algoritmo de roteamento executando num mesmo ambiente MANET. Para administrar esta dificuldade, será utilizada a tecnologia de *frameworks* orientado a objeto.

Com o estudo dos algoritmos de roteamento MANET de abordagem híbrida apresentado mais tarde no capítulo 3 pode-se verificar uma série de características comum a todos.

Por isso em busca da tentativa de proporcionar reuso e produtividade implementando algoritmos de roteamento a partir de um *framework,* é que este trabalho apresenta o desenvolvimento de uma estrutura denominada por FRAd-hoc – *Framework* de Roteamento Ad-hoc.

## 1.2 OBJETIVO GERAL

Com o intuito de criar uma estrutura que possibilite a implementação de algoritmos de roteamento para redes móveis *ad-hoc* que utilizam a abordagem híbrida, busca-se apresentar nesta pesquisa o desenvolvimento de uma estrutura denominada por FRAd-hoc – *Framework* de Roteamento *Ad-hoc*, responsável por agregar funcionalidades coletivas ao domínio estudado, de modo a produzir e disponibilizar artefatos de software reutilizáveis.

.

## 1.3 JUSTIFICATIVA

Devido à grande quantidade de algoritmos de roteamento que vêm sendo proposto com o intuito de apresentar uma solução apropriada ao roteamento em redes móveis *ad hoc*, verificou-se em (CORREA 2004 A e B) que os algoritmos buscam propor soluções diferentes utilizando técnicas similares as quais não são efetivamente eficientes aos diversos contextos apresentados durante todo o período de formação de uma MANET.

Pretende-se através desse trabalho desenvolver um *framework* orientado a objetos que comporte os mecanismos genéricos desses algoritmos fornecendo a partir de então a possibilidade do desenvolvimento de qualquer protocolo de roteamento que seja categorizado dentro da abordagem híbrida que incorpore a utilização de agentes móveis ou a divisão da rede em zonas, oferecendo artefatos de software reutilizáveis proporcionando futuramente a possibilidade de execução de um algoritmo somente durante o período que o mesmo apresente características apropriadas ao contexto da rede.

## 1.4 RESULTADOS ESPERADOS

Espera-se que esse trabalho estabeleça o inicio do desenvolvimento e da utilização de mais de um algoritmo de roteamento por parte da camada de redes em MANETs, efetivando a utilização de um protocolo de roteamento somente durante o período que a rede necessita das características apresentadas pelo mesmo.

## 1.5 METODOLOGIAS

Para que se possa efetivar o objetivo aqui definido, será necessário inicialmente:

- Efetuar análise comparativa e descrição do domínio das aplicações dos algoritmos de roteamento de rede móveis *ad hoc* que possuem abordagem híbrida;

- Identificar e definir quais classes e métodos que deverão generalizar o desenvolvimento do *framework*;

- Modelar as classes genéricas do domínio, as quais vão fazer parte do *framework*, bem como as classes dos algoritmos que fazem parte da especialização do mesmo, utilizando UML (*Unified Modelling Language*);

- Implementar e especializar o *framework* proposto utilizando a linguagem de programação Java, a fim de verificar a usabilidade desse. Justifica-se a utilização do Java por esse apresentar toda uma tecnologia voltada para o desenvolvimento multiplataforma.

## 1.6 LIMITAÇÕES

Sabe-se que para tornar esse trabalho efetivamente factível dever-se-ia implementar pelo menos três dos algoritmos de roteamento abordados nesta pesquisa, os quais consolidariam a proposta aqui apresentada. Além disso, seria necessário embutir o FR*Ad-hoc* num dispositivo móvel, e desenvolver uma ferramenta para efetuar o gerenciamento definindo a escolha do algoritmo mais apropriado para ser utilizado num determinado momento na rede. Contudo, devido a questões como complexidade do trabalho e indisponibilidade de tempo de recursos, este limitar-se-á apenas a análise, modelagem, desenvolvimento e teste de um pequeno *framework* caixa-branca, abrangendo apenas dois dos três algoritmos propostos ao desenvolvimento. É relevante esclarecer que também não serão tratados os aspectos relacionados à comunicação entre os nodos, já que os próprios autores das pesquisas utilizadas como exemplo para o desenvolvimento do *framework* proposto ignoram esse contexto.

## 1.7 TRABALHOS CORRELATOS

A intenção do grupo IETF de redes móveis *ad hoc* desde sua formação foi o de desenvolver a capacidade de roteamento móvel *peer-to-peer* em um domínio sem fio puramente móvel. Contudo abriu espaço para outros grupos de pesquisadores com interesse no desenvolvimento de pesquisas que tratassem de segurança, energia e interação com protocolos de camadas mais baixas e superiores. Desde então, dezenas de trabalhos têm sido desenvolvido para contribuir com a evolução das pesquisas no

âmbito de redes móveis *ad hoc*, dentre esses citamos abaixo alguns entre os quais consideramos mais relevantes.

Em seu trabalho (ABOLHASAN 2003) efetua uma classificação dos protocolos de roteamento considerados os mais atuais. Fornecendo uma visão geral de uma larga escala dos algoritmos de roteamento propostos na literatura. Como maior contribuição, julgamos que esse trabalho apresenta uma comparação de desempenho de todos os algoritmos de roteamento nele tratados, aconselhando qual dos protocolos abordados é capaz de oferecer melhor desempenho em redes de grande escala.

A pesquisa introduzida por (MIESO 2003), está entre as propostas de algoritmo de roteamento que incorporam a utilização de agentes móveis para o roteamento em redes móveis *ad hoc*. Através de uma arquitetura *clustering* usa-se agentes móveis para coletar e manter a informações de roteamento *intra-clustering* e *inter-clustering*. Esse trabalho é bastante similar a pesquisas que efetuam a divisão da rede em zonas que descrevemos mais detalhadamente no Capítulo 3. Contudo essa se diferencia pela utilização de agentes móveis.

Já o trabalho de pesquisa de (MICHAIL 2003) está entre os trabalhos que discutem os significativos benefícios que pode render uma eficiente aplicação da energia no projeto de algoritmos de roteamento em redes móveis *ad hoc*, propondo um conjunto de algoritmos que se referem a instruções de níveis do gasto de energia para transmissão de dados, descobertas de rota, e reserva de largura de banda. No entanto, esse trabalho não avalia os efeitos da mobilidade, pois alega que tais efeitos podem ser avaliados através do uso de mecanismos abordados por seus trabalhos anteriores.

Em (CASTAÑEDA 2002) são abordadas técnicas de localização dos nodos para algoritmos de roteamento sob demanda, buscando limitar a inundação dentro de uma pequena região na rede. Essa técnica demonstra por meio de simulação considerável redução de *overhead* e de atraso fim-a-fim.

## 1.8 ORGANIZAÇÃO DO TRABALHO

Este trabalho está estruturado da seguinte maneira: no capítulo 2 é dado uma explanação sobre redes móveis *ad hoc* abordando características, vantagens e dificuldades da utilização dessas redes.

No capítulo 3 tratamos, de modo genérico, alguns aspectos do roteamento, apontando algumas características que são essenciais para o desenvolvimento de tais

algoritmos. Logo em seguida, introduzimos a descrição dos algoritmos selecionados para fazer parte desse trabalho.

No capítulo 4 introduz-se a tecnologia de *framework* orientado a objetos e os aspectos que caracterizam um *framework*. Também são apresentadas a classificação dos *frameworks* e suas metodologias de desenvolvimento.

Os resultados de nossa pesquisa visando o desenvolvimento do *framework* orientado a objetos são apresentados no capítulo 5.

Finalmente, no capítulo 6, são expostas as conclusões provenientes deste trabalho de pesquisa, assim como são apresentadas algumas sugestões para trabalhos futuros no contexto do tema abordado.

# CAPÍTULO II

## REDES MÓVEIS *AD-HOC*

As redes móveis *ad hoc* (MANETs – *Mobile Ad hoc Networks*) ainda são ambientes em evolução dentro da comunicação sem fio (WU 2004). Isso ocorre devido a razões como a tendência na miniaturização dos dispositivos, tornando-os cada vez mais apropriados para essa área. O aumento de sua capacidade de processamento e armazenamento reunidos com aplicações capazes de oferecer boa interatividade entre hardware e software, são fatores que têm contribuído substancialmente com a popularidade dessas redes (PERKINS 2001).

Uma das motivações originais das MANETs foi encontrada, segundo (PERKINS 2001), na necessidade de sobrevivência dos militares em campos de batalha. Sendo que em tais regiões como desertos e florestas virgens não há infra-estrutura de comunicação territorial, bem como em situações que permitam a existência de uma, corre-se o risco de destruição da comunicação local. A arquitetura *ad hoc* com seus dispositivos móveis auto-organizáveis é também usada na assistência a desastres, conferências, redes de sensores, área de redes pessoais (PAN) e aplicações de computação embutida (WU 2004, MIGAS 2003).

### 2.1 CARACTERÍSTICAS

As redes *ad hoc* representam sistemas distribuídos complexos, que incluem nodos móveis sem fio que podem se organizar livre e dinamicamente de forma arbitrária dentro de uma topologia de rede temporária. Caracterizam-se por um sistema autônomo de nodos móveis independentes, que podem operar de modo isolado como mostrado na (Figura 2.1) ou por intermédio de um g*ateway* de interface com a rede fixa descrito pela (Figura 2.2). Nas MANETs, os nodos podem estar situados em aviões, barcos, caminhões, carros ou até mesmo em pessoas  e são equipados com transmissores e receptores usando antenas *omnidirecionais*, que captam sinais de todas as direções (*broadcast*), altamente direcional (*peer-to-peer*), ou uma combinação desses (CAMPBELL 2003). Por possuírem tecnologia de comunicação sem fio, os dispositivos computacionais móveis são capazes de trocar informações diretamente entre si ou

através de *multi-hop,* sem a necessidade de infra-estrutura de comunicação e devem estar fisicamente habilitados para se comunicarem mutuamente mesmo quando roteadores, estações base ou provedores de serviços de Internet (ISPs), não podem ser encontrados (CAMPBELL 2003).

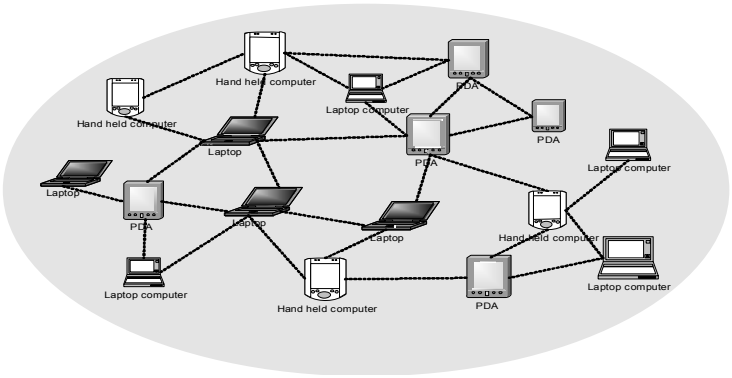



**Figura 2 1 - Topologia de uma rede móvel *ad hoc***

As MANETs empregam a estrutura tradicional TCP/IP para fornecer uma comunicação fim-a-fim entre os nodos (PERKINS 2001). Contudo, devido a sua mobilidade e aos recursos limitados em redes sem fio, cada camada do modelo TCP/IP requer definição e modificações para funcionar eficientemente.

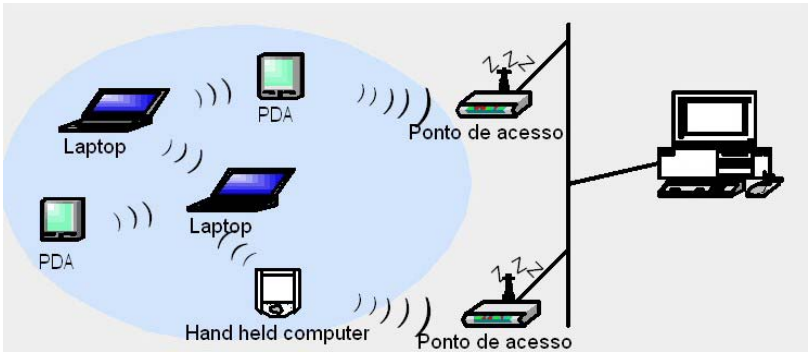



**Figura 2 2 – Topologia de rede móvel *ad hoc* comunicando-se com a rede fixa**

Conforme tratado em (CORSON 1999) temos definido como características mais salientes:

- **Topologia dinâmica**: nodos são livres para moverem-se arbitrariamente, fazendo com que a topologia da rede mude randômica e rapidamente em

momentos não esperados de modo imprevisível, e pode consistir de ligações bidirecionais e unidirecionais;

- **Segurança física limitada**: essas redes são geralmente mais propensas ao aumento das possibilidades de escuta e invasões;

- **Rápidas instalações**: são instaladas rapidamente sem a necessidade de infra-estrutura fixa;

- **Conectividade**: se dá através de um canal de comunicação entre dois ou mais *hosts*, dentro de uma mesma área geográfica de alcance de ondas de rádio;

- **Mobilidade**: os dispositivos móveis podem mover-se durante todo seu período de conexão;

- **Largura de banda limitada**: os enlaces sem fio continuam tendo capacidade significantemente mais baixa do que as redes fixas;

- **Energia limitada**: os nodos contam com baterias para funcionar; para tanto um dos critérios a ser levado em consideração no projeto desses sistemas é o consumo de energia;

- **Localização dos hosts**: nas redes moveis *ad hoc* encontramos a dificuldade de localização dos *hosts* móveis para estabelecimento de comunicação, visto que essas redes não utilizam endereço IP;

- **Roteamento**: a possibilidade dos nodos móveis se deslocarem de uma região para outra modificando constantemente sua topologia, traz a dificuldade do estabelecimento e determinação de uma rota válida.

Essas características criam um conjunto básico fundamental de hipóteses e interesses de desempenho para projetos de protocolos nos quais diferem daquela direção

do projeto de roteamento e outros protocolos de controle da rede dentro de alta velocidade e topologia estática da Internet fixa.

## 2.2 APLICAÇÕES

O crescente interesse pelas redes móveis *ad hoc* é justificado pelas diversas vantagens que essas redes podem proporcionar para certos tipos de aplicações. Uma vez que redes como essas podem ser construídas rapidamente sem a necessidade de infra-estrutura fixa, tornando-as adequadas a situações e locais os quais são impróprios para redes cabeadas. Outro aspecto que torna as redes móveis *ad hoc* atraentes é o fato de não dependerem de pontos que determinem sua organização e controle, evitando assim que o desempenho da rede seja afetado, caso um nodo em particular venha falhar, podendo assim ser adicionados facilmente novos nodos à rede. Dentro das aplicações para as quais as redes móveis *ad hoc* podem ser utilizadas temos:

**Militar:** MANETs torna possível a comunicação entre soldados em campo de batalha já que tais ambientes impossibilitam a existência de comunicação por meio de uma infra-estrutura fixa;

**Redes de sensores:** o aumento da popularidade do uso de sensores no meio ambiente permite operar automaticamente e coletar processos, e transmitir informações sobre seu ambiente sem o suporte de uma infra-estrutura;

**Assistência a desastres:** já que as redes móveis *ad hoc* não necessitam de uma infra-estrutura pré-existente, podem oferecer um método extremamente flexível em operações de resgates ou qualquer outro cenário que requeira estabelecimento de comunicação que seja rapidamente organizada, possibilitando a troca de informações com sobreviventes ou com quem presta auxilio aos sobreviventes, bem como auxiliando em situações de emergência, como policiamento e combate ao fogo;

**WPAN – *Wireless Personal Area Network***: é uma rede de área pessoal que permite interconectar os dispositivos centrados em torno de um espaço de trabalho individual de pessoas, cujas conexões são sem fio. Uma rede de área pessoal sem fio tipicamente usa

alguma tecnologia que permite uma comunicação dentro de aproximadamente 10 metros, ou seja, uma escala ainda muito curta.

## 2.3 DIFICULDADES

As MANETs introduzem um número de desafios técnicos significantes para a comunidade de pesquisa pois, quando comparadas com o atual momento da Internet, percebemos que essas redes contam ainda com características, que de certo modo, degradam sua utilização. Entre elas podemos citar, energia e largura de banda limitada, pois tratamos com dispositivos que contam com a utilização de baterias para desempenhar entre outras, as funções de transmissão, recepção, armazenamento e processamento de informações (MICHAIL 2003).

Outro aspecto a ser considerado é a necessidade de mecanismos de segurança eficientes, uma vez que a natureza operacional destas redes as torna, naturalmente, mais propensas à escuta, invasão e ataques. Ainda, existe a questão da interoperabilidade devido à falta de padronização dos produtos que dificultam a implementação e configuração de uma rede móvel *ad hoc* forçando a utilização de dispositivos de um único fabricante. Além disso, essas redes necessitam ser projetadas para operar em um ambiente largamente variado entre as redes militares com centenas de nodos, a redes com milhares de sensores de alta capacidade.

Entre os diversos problemas que podemos citar nas MANETs temos também o problema da localização dos *hosts* causado pelo fator da mobilidade, diversos trabalhos de pesquisa como (MARWAHA 2002 A e B, ONISHI 2001, JOA-NG 1999, NIKAEIN 2001, NIKAEIN 2000, ROY 2000, DOL 2002, MIGAS 2003) têm sido desenvolvidos com o intuito de oferecer, entre outros, um algoritmo de roteamento que definam a topologia da rede e estejam constantemente informando o estado e a localização de cada *host*. Alcançar a eficácia de um protocolo de roteamento para uma rede móvel *ad hoc* é uma tarefa instigante, já que esse deve executar sobre uma larga escala de contextos de rede.

## 2.4 CONCLUSÃO DO CAPÍTULO

Esse capítulo apresentou um breve histórico das redes móveis *ad hoc*, descrevendo suas principais características, bem como algumas vantagens que justificam sua aplicação e desvantagens que revelam pontos ainda em fase de evolução dentro de tal pesquisa.

É no conhecimento de algumas destas dificuldades que está baseada nossa pesquisa, particularmente no que diz respeito ao problema do roteamento nestas redes. O próximo capítulo vai discorrer sobre este problema, de modo mais específico.

# CAPÍTULO III

## PROTOCOLOS DE ROTEAMENTO

Por mais que um ambiente MANET seja produtivo, a natureza dinâmica de sua topologia dificulta a realização do roteamento *multi-hop* (ONISHI 2001). A principal tarefa da camada de rede é realizar o roteamento de pacotes de uma origem a um destino. No entanto, a escolha dos algoritmos que definem as rotas e as estruturas de dados que utilizam são elementos importantes a serem estabelecidos a nível dessa camada.

Em (TANENBAUM 2003) é definido que o algoritmo de roteamento é parte do software da camada de rede responsável pela decisão de como os pacotes de entrada serão transmitidos. Para julgar o mérito de todo e qualquer algoritmo de roteamento, (TANENBAUM 2003) enumera algumas características desejáveis as quais os algoritmos de roteamento devem satisfazer. Dentre as principais podemos citar, simplicidade, imparcialidade, escolha do melhor trajeto, escalabilidade, atender a parâmetros de qualidade de serviço (QoS – *Quality of Service*), ser independente da topologia, efetuar rápida convergência para o caminho ótimo, e robustez uma vez que se espera que uma rede funcione sem interrupções por longos períodos de tempo.

## 3.1 CLASSIFICAÇÃO DOS PROTOCOLOS DE ROTEAMENTO

Os algoritmos de roteamento em geral, podem ser categorizados em algoritmos adaptativos ou não adaptativos em que aplicam metodologia de roteamento das quais pode-se destacar (TANENBAUM 2003):

- **Roteamento do caminho mais curto**

  Esse conceito baseia-se na forma de medir o comprimento do trajeto que pode ser determinada de modo genérico pelo número de saltos ou pela distância física entre um nodo fonte e um nodo destino. Como exemplo temos o BGP (RFCs 1654 e 1771) (REKHTER 1995).

- **Inundação**

O algoritmo *flooding* impõe que cada pacote de entrada seja enviado para toda linha de saída, exceto para aquela em que chegou. Esse algoritmo pode gerar infinitos números de pacotes caso não sejam tomadas algumas medidas para minimizar o processo. Como exemplo de utilidade, esse algoritmo pode ser empregado em aplicações militares e aplicações de bancos de dados distribuídos.

- **Roteamento baseado no fluxo de dados**

A metodologia de roteamento baseado no fluxo preocupa-se tanto com a topologia quanto com a carga para o roteamento, isto é, caso haja sempre um grande volume de tráfego entre um determinado nodo fonte e nodo destino, essa abordagem define que talvez seja melhor encaminhar os dados por um caminho que seja mais longo, porém possua menor volume de informações sendo distribuídas.

- **Roteamento com vetor de distância**

Os algoritmos que utilizam essa abordagem operam fazendo com que cada nodo roteador mantenha uma tabela que fornece a melhor distância conhecida a cada destino determinando qual o trajeto que deve ser utilizado para alcançar esse destino. Essas tabelas são atualizadas através da troca de informações com vizinhos. Como exemplo temos o RIP (HEDRICK 1988) (RFC 1058), RIP (MALKIN 93) (RFCs 1387, 1388, 1389), IGRP e EIGRP (CISCO 1980).

- **Roteamento por estado de enlace**

De modo genérico esse método de roteamento estabelece que cada nodo roteador deve: descobrir seus vizinhos e aprender seus endereços, medir o atraso para cada um de seus vizinhos, criar uma tabela que guarde tudo o que acaba de ser aprendido, enviar essa tabela a todos os seus vizinhos e calcular o caminho

mais curto para cada um dos nodos roteadores. Como exemplo temos o OSPF (MOY 1997) (RFC 2178).

- **Roteamento hierárquico**

Os algoritmos que utilizam essa idéia dividem a rede em regiões fazendo com que os nodos conheçam todos os detalhes de roteamento apenas dentro de sua região sem conhecer nenhuma particularidade sobre a estrutura interna de outras regiões onde essas regiões estabelecem a comunicação por intermédio de nodos *gateways.*

- **Roteamento *broadcasting***

Esse método pode ser utilizado por aplicações que precisam enviar pacotes a todos os outros destinos simultaneamente.

- **Roteamento *multicast***

Roteamento *multicast* é o nome dado aos algoritmos de roteamento que utilizam o envio de mensagens para aplicações separadas que funcionam em grupo. Esse método pode ser exemplificado por um grupo de processos que implementam um sistema de bancos de dados distribuídos. Com freqüência é necessário que um processo envie uma mensagem a todos os outros membros do grupo (TANENBAUN 2003). O envio de uma mensagem a um desses grupos é chamado de *multicast*.

## 3.2 CLASSIFICAÇÃO DOS PROTOCOLOS DE ROTEAMENTO PARA MANET

Já os protocolos de roteamento para MANETs, segundo (ABOLHASAN 2003) podem ser classificados genericamente em três grupos diferentes: *Global*/Pro-ativo, *On-demand*/Reativo e Híbrido.

- **ABORDAGEM PRÓ-ATIVA**

Nos protocolos de roteamento pró-ativos, é definido por (ABOLHASAN 2003) que as rotas para todo destino são determinadas no inicio e mantidas usando um processo periódico de atualização da rota. A informação do roteamento é periodicamente transmitida por toda rede a fim de manter a tabela de roteamento consistente. Os protocolos pró-ativos diminuem o atraso da determinação das rotas para um destino, mas eles desperdiçam uma quantidade significativa dos escassos recursos sem fio a fim de manter a atualização da tabela de roteamento. Tais protocolos são escaláveis em relação à freqüência da conexão fim-a-fim. Embora os protocolos pró-ativos não sejam escaláveis em relação ao número total de nodos, eles podem ser feitos escaláveis se uma arquitetura hierárquica é usada. Em conclusão, essa abordagem de protocolo não é escalável em relação á freqüente mudança da topologia.

Como exemplo de protocolos pró-ativo temos dentre os mais conhecidos: DSDV (PERKINS 1994), WRP (MURTHY 1995), GSR (CHEN 1998), FSR (GERLA 2002), STAR (ACEVES 1999), DREAM (BASAGNI 1998), HSR (GEI 1999), OSLR (JAC 2001), TBRPF (BELLUR 2003).

- **ABORDAGEM REATIVA**

Nos protocolos reativos, um nodo inicia a descoberta da rota somente quando ele deseja se comunicar com seu destino (NIKAEIN 2001). Uma vez que a rota é estabelecida, ela é mantida por um processo de manutenção de rotas até o destino tornar-se inacessível ou até a rota não ser mais apropriada. Os protocolos reativos diminuem o *overhead* de comunicação para a determinação da rota; mas eles não são os melhores em termos de utilização da largura de banda. Por causa da natureza *flooding* ficam escaláveis em relação a freqüente mudança da topologia. Tais protocolos não são escaláveis em relação ao número total de nodos. Contudo, podem tornar-se escaláveis caso uma arquitetura hierárquica seja adotada. Em conclusão protocolos reativos não são escaláveis em relação a freqüência de conexão fim-a-fim.

Como exemplo de protocolos reativo temos dentre os mais conhecidos: AODV (DAS 2003), DSR (JHONSON 2002), ROAM (RAJU 1999), TORA (PARK 1997), RDMAR (AGGELOU 1999), LAR (KO 1998), ARA (GUNES 2002).

- **ABORDAGEM HÍBRIDA**

O protocolo de roteamento híbrido combina as propriedades básicas das duas primeiras classes de protocolos, possuindo assim natureza pró-ativa e reativa. São considerados por (ABOLHASAN 2003) a nova geração dos protocolos de roteamento para MANETs, pois são projetados para aumentar a escalabilidade, permitindo que os nodos com proximidade trabalhem juntos para formar uma espécie de *backbone* a fim de reduzir o *overhead* de descoberta da rota. Essa abordagem de protocolo pode fornecer o melhor *trade-off* entre *overhead* de comunicação e atraso, mas esses *trade-off* são subjetivos ao tamanho da zona e à dinâmica de uma zona. Além disso, a abordagem híbrida estabelece um compromisso com a emissão de escalabilidade em relação à freqüência de conexão fim-a-fim, ao número total de nodos e à freqüência da mudança da topologia.

Como exemplo de protocolos híbridos temos, dentre os mais conhecidos: ZRP (HAS 1999), ZHLS (JOA-NG 1999), DST (RADHAKRISHNAN 1999), DDR (NIKAEIN 2000), HARP (NIKAEIN 2001), Ant-AODV (MARWAHA 2002).

## 3.3 CARACTERÍSTICAS DE DESEMPENHO DOS PROTOCOLOS DE ROTEAMENTO DE UMA MANET

Quando tratamos de MANETs devemos sempre considerar que estamos lidando com um ambiente móvel onde a comunicação ocorre por meio de ondas de rádio, sem a existência de uma entidade central para uma eventual coordenação da rede, cuja topologia pode sofrer rápidas e constantes mudanças.

Para que a comunidade científica pudesse vir a contribuir com novas propostas de algoritmos de roteamento, (CORSON 1999) estabeleceu métricas qualitativas e quantitativas que devem ser atendidas como requisitos mínimos de eficiência que um protocolo de roteamento MANET deve atender.

Como definição de métricas qualitativas desejáveis que um protocolo de roteamento MANET deve atender temos (CORSON 1999):

1. **Operação distribuída** – Essa é uma propriedade essencial para o roteamento nas redes móveis *ad hoc* a ser declarada, a fim de evitar a centralização da informação;

2. **Livre de *Loop*** – Essa característica é desejável pois pode evitar que os pacotes trafeguem ao redor da rede por períodos de tempo arbitrários; como solução pode se usar uma variável do tipo TTL (t*ime to live* );

3. **Operação baseada na demanda** – É importante que o algoritmo de roteamento se adapte as condições de tráfego ao invés de supor uma distribuição uniforme do tráfego dentro da rede distribuindo todas as informações para todos os nodos. Caso isso seja feito de modo inteligente, pode-se utilizar a energia da rede e os recursos de largura de banda de forma mais eficiente;

4. **Operações pró-ativas** – Em alguns contextos, a latência adicional que incorre em operações baseadas na demanda podem ser inaceitáveis, então se os recursos de largura de banda e energia permitirem, a operação pró-ativa é desejável nesse momento;

5. **Segurança** – Sem algumas configurações de segurança no nível de rede ou de enlace um protocolo de roteamento de uma rede móvel *ad hoc* é altamente vulnerável a muitos ataques; para isso é desejado que mecanismos adicionais sejam acrescentados a fim de evitar a modificação da operação do protocolo;

6. **Operação de período de *Stand-by*** – A fim de poupar energia, ou de alguma outra necessidade de inatividade, os nodos de uma MANET podem parar de transmitir /receber informações por arbitrários períodos de tempo. Um protocolo de roteamento deve estar habilitado a lidar com tais períodos de ociosidade sem que isso cause conseqüências desfavoráveis.

Os pontos quantitativos que devem ser observados para analisar o desempenho de um protocolo de roteamento em uma rede móvel *ad hoc* segundo (CORSON 1999) são:

1. ***Throughput* e atraso de dados fim-a-fim** – Medidas estatísticas como variância, média e distribuição são muito importantes para avaliação da eficácia do desempenho de uma política de roteamento;

2. **Tempo de aquisição das rotas** - Essa é uma forma particular de medir atraso de um pacote fim-a-fim de interesse particular dos algoritmos de roteamento sob demanda; é o tempo requerido para estabelecer as rotas quando requisitadas;

3. **Porcentagem de entrega fora de ordem** - Medida externa do desempenho de algoritmos de roteamento sem conexões, de interesse particular aos protocolos da camada de transporte, tais como TCP, que entregam os pacotes na ordem correta;

4. **Eficiência** – Se a eficácia do roteamento dos dados é uma medida externa de desempenho de uma política, a eficiência é a medida interna de sua efetividade. Se o controle de tráfego de dados deve compartilhar o mesmo canal, e a capacidade do canal for limitada, então o controle de tráfego excessivo causará impacto no desempenho do roteamento.

Para alcançar a eficiência de um protocolo de roteamento, deve-se considerar também os diversos fatores relacionados ao contexto da rede tais como, o tamanho da rede em número de nodos, a quantidade de vizinhos que cada nodo possui, a velocidade com que a topologia da rede muda, fração das ligações unidirecionais, mobilidade dos nodos, e, entre outros, a freqüência com que os nodos entram e saem do período de sonolência. Observando-se então esses fatores peculiares às redes móveis *ad hoc,* pode-se medir a eficiência de um protocolo de roteamento tendo como parâmetros os seguintes valores (CORSON 1999):

- **Número médio de bits de dados transmitidos / Bits de Dados entregues** – esse parâmetro pode ser visto como uma medida da eficiência de bits de dados entregues dentro da rede, oferecendo indiretamente a contagem média dos saltos feitos por pacotes de dados;

- **Número médio dos bits de controle transmitidos / Dados de bits entregues** – mede a eficiência do bit do protocolo no gasto de overhead para entrega de dados. Observe que isso deve incluir não apenas os bits nos pacotes de controle de roteamento, mas também os bits no cabeçalho dos pacotes de dados, ou seja, qualquer coisa que não é dado, é controle, e deve ser contado no algoritmo;

- **Número médio do controle e pacotes de dados transmitidos / Pacotes de dados entregues –** essa medida tenta mensurar a eficiência de acesso ao canal do protocolo, ao invés de medir a eficiência em número de bits do protocolo de roteamento.

Um protocolo MANET deve funcionar eficazmente sobre uma larga escala de contextos de redes pequenas ou corporativas (CORSON 1999). No entanto devido a características próprias dessas redes surge a necessidade de que novos protocolos de roteamento sejam desenvolvidos, assim novos métodos para avaliar a eficiência de um protocolo de roteamento continuarão sendo propostas pelo grupo MANET baseado na descrição precedente das características e métricas de avaliação.

## 3.4 DESCRIÇÃO DO DOMÍNIO

Nessa seção abordaremos as características dos protocolos de roteamento para o ambiente de redes móveis *ad hoc* que serão utilizados neste trabalho de pesquisa. Primeiramente relataremos três trabalhos (JOA-NG 1999, NIKAEIN 2000, NIKAEIN 2001) bem conhecidos e categorizados como protocolos de roteamento híbrido. Na seqüência, trataremos de outros dois trabalhos de pesquisa (MARWAHA 02 A e B e MIGAS 03) que utilizam agentes móveis onde pode ser caracterizado como híbrido e caracteriza-se pela utilização da abordagem pró-ativa. Dentre os estudados temos:

### 3.4.1 *ZHLS – Zone-based hierarchical link state*

O ZHLS (JOA-NG 1999) é um protocolo de roteamento *peer-to-peer,* característica essa que evita gargalo de tráfego, previne único ponto de falha e simplifica o gerenciamento da mobilidade. O ZHLS (JOA-NG 1999) incorpora informação da posição dentro de uma abordagem de roteamento hierárquico. A rede é dividida dentro de zonas não sobrepostas e o roteamento é feito pela definição do seu zone_ID e do nodo_ID do destino. Inicialmente, cada nodo conhece sua própria posição e conseqüentemente o seu zone_ID através do sistema de posição global (GPS – *Global Position System*). O tamanho da zona depende de fatores como mobilidade do nodo, densidade da rede, força de transmissão e características de propagação. Já a divisão da rede pode ser baseada sobre um simples particionamento geográfico ou sobre uma divisão de propagação de rádio.

O nível da topologia do nodo (Figura 3.1) oferece a informação de como os nodos estão conectados junto a seu enlace físico. Por exemplo, na (Figura 3.1), se o nodo *a* quer enviar um pacote de dados ao nodo *f,* os dados terão que passar por *a – b – e – f*. Já o nível da topologia da zona (Figura 3.2) demonstra como as zonas são conectadas por seus enlaces virtuais. Como exemplos, na (Figura 3.2) vemos que os enlaces virtuais entre zona 4 e zona 3 são 4 – 1 – 3.

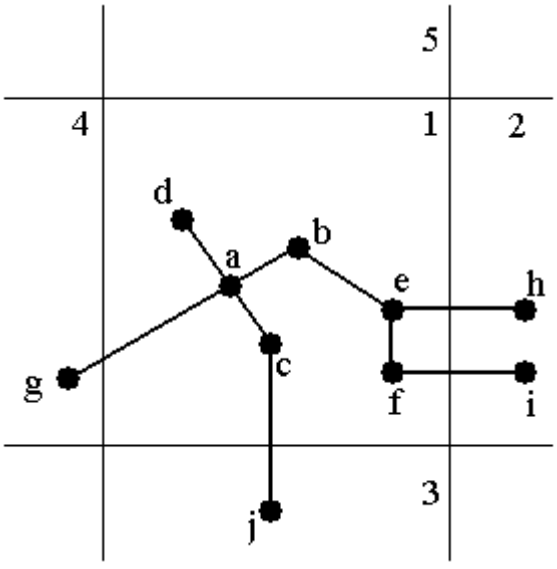



**Figura 3 1 – Topologia de nível do nodo.**
(JOA-NG 1999)

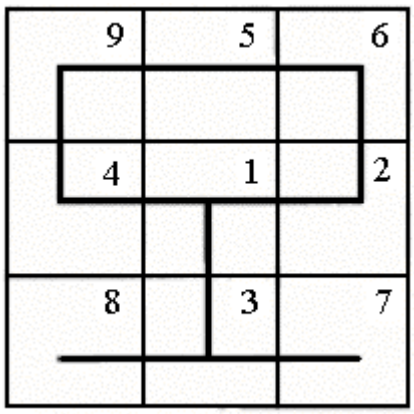



**Figura 3 2 – Topologia de nível da zona**
(JOA-NG 1999)

#### 3.4.1.1 Mapa da zona

Como mencionado anteriormente, no ZHLS, a rede está dividida dentro de zonas. O tamanho da zona depende de fatores tais como mobilidade do nodo, densidade da rede, força de transmissão, e características de propagação. O particionamento pode ser baseado sobre uma divisão geográfica ou sobre a divisão da propagação de rádio. A divisão geográfica é considerada simples e não requer qualquer medição de características de propagação de rádio, considerando o particionamento de propagação de rádio é mais correta para o reuso de freqüência. A divisão por propagação de rádio é preferível se a propagação medida puder ser feita no estágio de projeto. Entretanto algumas aplicações, tais como operação de emergência, de resgate de desastres, táticas de comunicações militares, e coação da lei, não permitem tais medidas. Em tais casos, um particionamento geográfico tem sido usado.

#### 3.4.1.2 Clustering

Equipado com seu zone_ID o nodo pode iniciar o *clustering* intra-zona e então os procedimentos de *clustering* inter-zona para construir suas tabelas de roteamento como veremos a seguir.

#### 3.4.1.3 *Clustering* Intra-zona

Cada nodo assincronamente espalha uma requisição de enlace. Os nodos dentro de sua faixa de comunicação respondem por sua vez com mensagens contendo suas ligações (zona_ID e nodo_ID). Após serem recebidos todos os enlaces de respostas, os nodos geram seus pacotes de estado de enlace denominados LSP (*link state packet*) contendo o nodo_ID de seus vizinhos de zonas diferentes. Como exemplo, na (Figura 3.3), os nodos *b*, *c*, e *d* são nodos vizinhos do nodo *a*, e a zona 4 é sua zona vizinha. O nodo *a* então propaga seus LSP de nodo localmente por toda sua zona via nodos intermediários. Desde que cada nodo execute esses procedimentos, uma lista de LSP, tal como mostrado na Tabela 3.1, pode ser armazenada em cada nodo. Entretanto, o LSP dos nodos de outras zonas não serão armazenados porque os LSP dos nodos são somente propagados dentro de suas zonas. Os procedimentos de *clustering* intra-zona é descrito na Figura 3.3 (a)-(d).

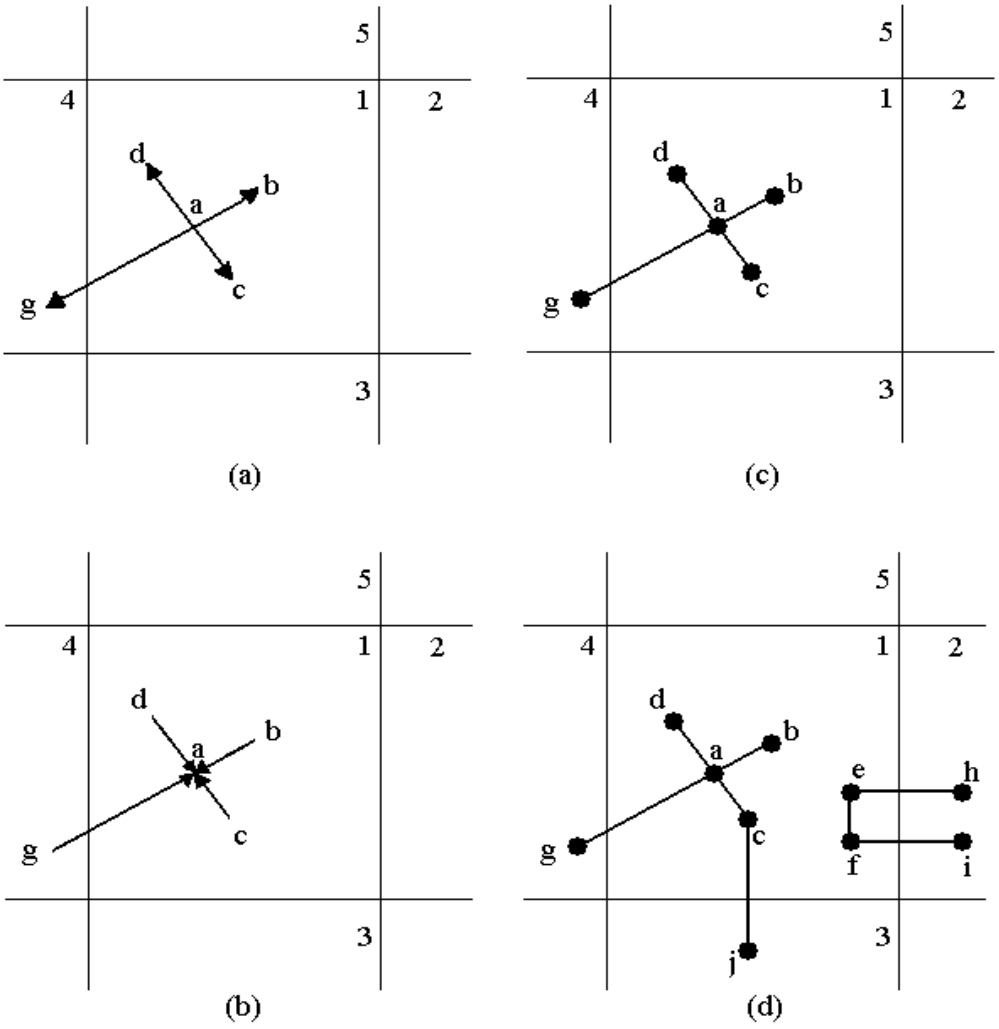



**Figura 3 3 – Procedimento *clustering* intra-zona**

(a) Nodo *a* encaminha através de *broadcast* um enlace de requisição para seus vizinhos. (b) Nodo *a* recebe o enlace de resposta dos seus vizinhos. (c) Nodo *a* gera seus próprios LSP de nodo e encaminha-os através das zonas. (d) Todos os nodos desempenham os passos anteriores assincronamente . (JOA-NG 1999)

**Tabela 3 1 - LSP do nodo na zona**

TABELA 1

Nó LSP na Zona 1

Após receber todos os LSP dos nodos da mesma zona, cada nodo conhecerá a topologia do nível do nodo dessa zona. O algoritmo de caminho mais curto é usado para construir suas tabelas de roteamento. A Tabela 3.2 mostra um exemplo da tabela de roteamento intra-zona do nodo *a*. Devido a mobilidade do nodo e o desaparecimento do canal, o procedimento anterior tem que ser periodicamente realizado para detectar e atualizar qualquer mudança no enlace físico. Se um nodo se move para outra, zona seu LSP de nodo deve ser deixado em sua zona anterior. Assim um temporizador é ajustado a cada LSP de nodo recebido, onde ao expirar será apagado.

**Tabela 3 2 – Tabela de roteamento intra-zona do nodo *a.***

| Destino | Próximo Nó |
|---|---|
| b | b |
| c | c |
| d | d |
| e | b |
| f | b |
| 2 | b |
| 3 | e |
| 4 | g |

(JOA-NG 1999)

### 3.4.1.4 *Clustering* Inter-zona

Os nodos podem receber respostas de nodos de suas zonas vizinhas. Esses nodos são chamados de nodos *gateway* como mostrado na (Figura 3.3), os nodos *a*, *c*, *e*, e *f* são nodos *gateway* da zona 1. Desde que cada LSP do nodo contenha o ID de zona da zona em que está conectado, cada nodo conhecerá à qual zona está conectada sua zona. Por exemplo, baseado no LSP do nodo na Tabela 3.1, as zonas 2, 3 e 4 são zonas conectadas na zona 1. No estágio de iniciação, após ter-se certificado que todos os LSP de nodos foram recebidos, cada nodo de mesma zona gera o mesmo LSP da zona. Os nodos *gateway* então espalham esse LSP da zona por toda a rede. Desde que cada zona execute esse procedimento, uma lista de LSP da zona, idêntica a descrita na Tabela 3.3, é armazenada por cada nodo. Assim cada nodo conhecerá a topologia do nível da zona da rede.

O procedimento *clustering* inter-zona é descrito na (Figura 3.4 (a) – (b)). Após cada nodo receber todos os LSP da zona, o algoritmo de trajeto mais curto é usado para encontrar o menor trajeto em termos de saltos, e construir a tabela de roteamento inter-zona do nodo, mostrada na Tabela 3.4.

Os procedimentos anteriores repetem-se periodicamente. Contudo, os nodos *gateway* não distribuirão seus LSP de zona se seu valor permanecer inalterado. Essa vantagem é obtida da mudança infreqüente das ligações virtuais e reduz conseqüentemente a quantidade de tráfego. Além disso, diferente do LSP nodo, nenhum temporizador é apontado para o LSP da zona, sendo esse atualizado somente quando qualquer enlace virtual for quebrado ou criado.

**Tabela 3 3 – LSP DA ZONA.**

LSP das Zonas

| **Origem** | **LSP** |
|---|---|
| 1 | 2,3,4 |
| 2 | 1,6 |
| 3 | 1,7,8 |
| 4 | 1,9 |
| 5 | 6,9 |
| 6 | 2,5 |
| 7 | 3 |
| 8 | 3 |
| 9 | 4,5 |

(JOA-NG 1999)

**Tabela 3 4 – Tabela de roteamento inter-zona do nodo *a***

| Destino | Próximo Nó |
|---|---|
| b | b |
| c | c |
| d | d |
| e | b |
| f | b |
| 2 | b |
| 3 | e |
| 4 | g |

(JOA-NG 1999)

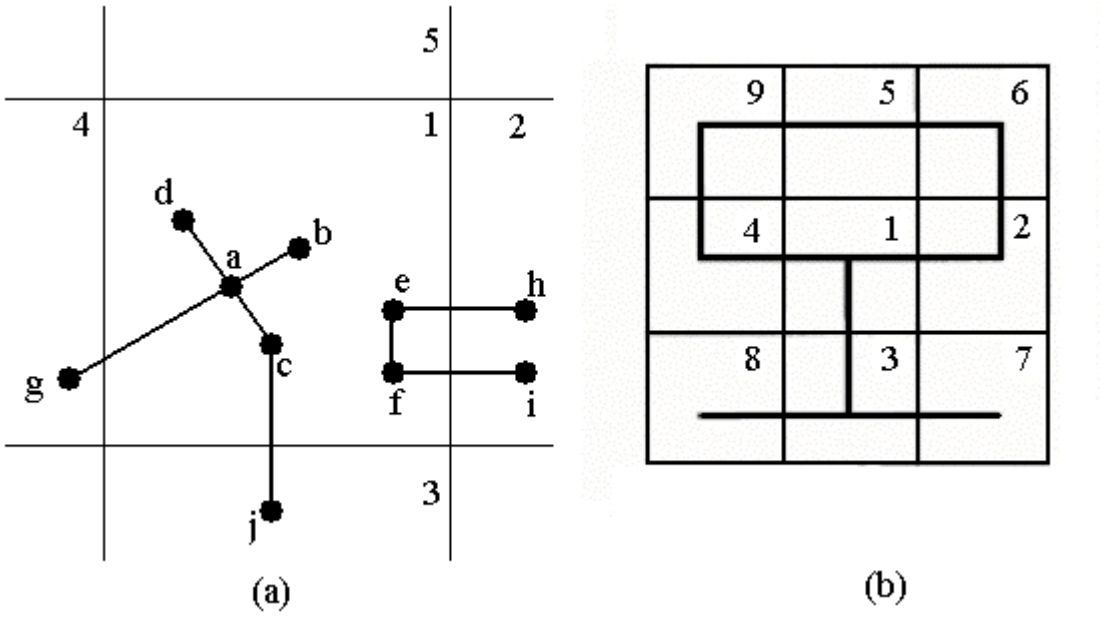



**Figura 3 4 – Procedimento *clustering* inter-zona.**

(a) os nodos *gateway* encaminham os LSP da zona através da rede. (b) Enlaces virtuais entre as zonas adjacentes são estabelecidos

(JOA-NG 1999)

Por exemplo, um nodo recebe dois LSP da zona originado de diferentes nodos *gateway* de uma mesma zona. Após encaminhar o primeiro deles, o nodo não encaminhará o segundo, já que ele é idêntico ao primeiro. Por essa razão, mesmo que possa haver mais de um nodo *gateway* em uma zona, somente um LSP de zona é gerado dessa zona. A sincronização local está prontamente disponível se um sistema de posicionamento global (GPS – *Global Position System*) for usado.

### 3.4.1.5 Busca de posição e mecanismo de roteamento

Com o protocolo IP, o roteamento é projetado para ser hierárquico. A rede é dividida em sub-redes diferentes. Desde que os nodos na rede IP sejam fixos, cada nodo é associado com um endereço IP hierárquico, o qual contém um ID da sub-rede fixa. Similarmente, no ZHLS, a rede é dividida em zonas. Entretanto, a mobilidade dos nodos proíbe-nos de associá-los com ID da zona fixa. No entanto, uma fonte precisa encontrar um ID da zona de um nodo destino antes que qualquer transmissão de dados possa iniciar.

Por exemplo, o nodo *a* quer enviar dados ao nodo *z* (Figura 3.5). Antes de enviar dados ao nodo *z*, o nodo *a* verificará se o nodo *z* existe em sua tabela de roteamento intra-zona. Se existir, o nodo *a* encaminhará os dados para o nodo *z* de acordo com sua tabela de roteamento intra-zona. Caso contrário, se o nodo *z* estiver numa zona diferente o nodo *a* então enviará uma requisição de posição para cada outra zona *x*. Cada nodo intermediário encaminhará o pedido de requisição de posição destinado para a zona *x* para outra zona *x* de acordo com sua tabela de roteamento inter-zona. O caminho do nodo *a* para zona *x* é adaptável a mudanças da topologia. Um nodo *gateway* de cada zona receberá a requisição da posição e checará sua tabela de roteamento intra-zona para ver se o nodo *z* existe em sua zona. Como já mencionado o ZHLS não limita um nodo *gateway* por zona. Isso evita um único ponto de falha. Um nodo *gateway* na mesma zona do nodo *z* replicará com uma resposta da posição.

O ID da zona 5 e o ID do nodo *z,* por exemplo, são então especificados no cabeçalho de dados. O nodo *a* distribuirá seus dados via nodo *g* para zona 5 de acordo com sua tabela de roteamento inter-zona (Tabela 3.4). Todos os nodos intermediários, esses na zona 5, distribuirão os dados para essa zona de acordo com suas tabelas de roteamento. Quando os dados alcançarem a zona 5, os nodos intermediários usarão em lugar suas tabelas de roteamento intra-zona para distribuir os dados para o nodo *z*.

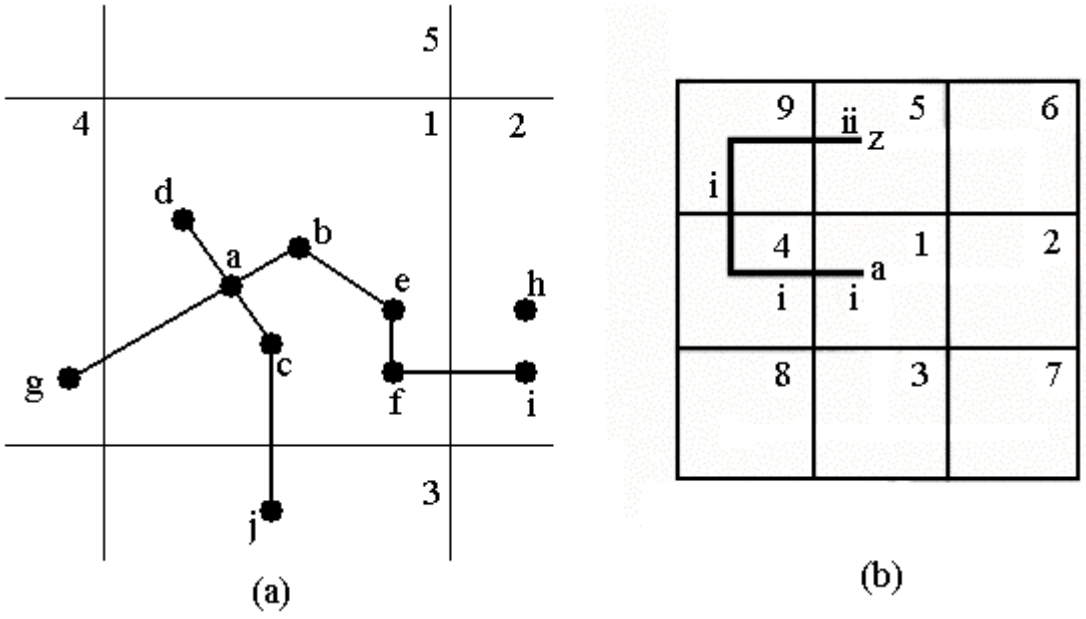



**Figura 3 5 – Trajeto de roteamento.**

Legenda : *i* usa a tabela de roteamento inter-zona, e o trajeto *ii* usa a tabela de roteamento intra-zona

(JOA-NG 1999)

Mesmo se o nível do nodo ou a topologia do nível da zona mudar durante a transmissão de dados o roteamento pode ainda ser particularmente feito. Por exemplo, as topologias de nível de zona no tempo *t1* e *t2* mostradas na (Figura 3.6 (a) e (b)), respectivamente, os nodos na zona *x* podem ainda rotear dados para nodo *d* mesmo através de um caminho virtual entre zona *x* e zona *d* (ID da zona do nodo *d*) é quebrada num momento de transmissão correta mesmo se o nodo *s* possui a informação inter-zona levemente obsoleta porque somente o ID da zona e do nodo de um destino são necessários para o roteamento, a rota é adaptável a topologia dinâmica.

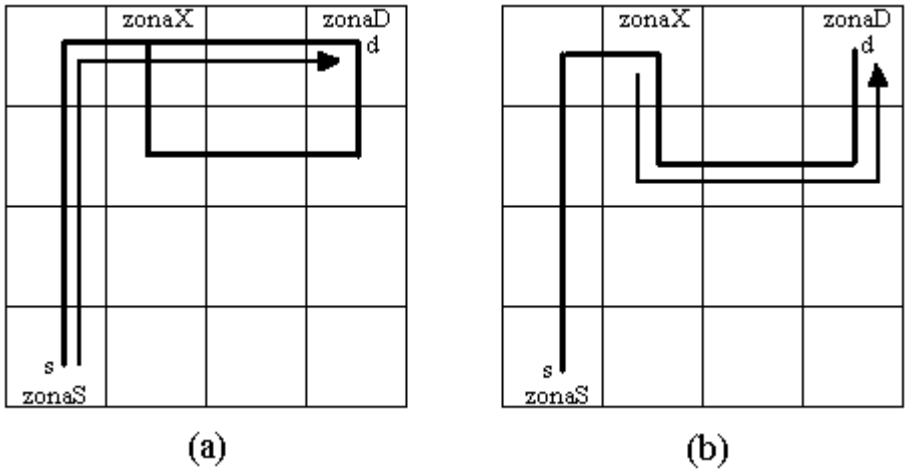



**Figura 3 6 – Mudanças no *backbone* virtual.**

Legenda : (a) tempo t1. (b) tempo t2.
(JOA-NG 1999)

### 3.4.2 *DDR – Distributed dynamic routing algorithm for mobile ad hoc networks*

Como o ZHLS, o DDR (NIKAEIN 2000) é uma abordagem híbrida que se baseia no conceito de zonas não sobrepostas. No DDR, cada nodo precisa conhecer somente o próximo salto para todos os nodos dentro de sua zona, reduzindo dessa forma o roteamento de informação e utilização da largura de banda.

O protocolo proposto não requer conhecimento de sua posição física para o roteamento, evita *broadcasting* enviando somente a informação necessária embutida em *beacons* (sinais de alarme) para os vizinhos, e o tamanho de suas zonas aumentam e diminuem dinamicamente de acordo com algumas características como densidade do nodo, taxa de conexão e desconexão da rede, força de transmissão e mobilidade do

nodo. A idéia principal do DDR é construir uma floresta de uma topologia da rede de modo distribuído usando somente a troca periódica de mensagens entre os nodos e seus vizinhos.

### 3.4.2.1 Definições preliminares do algoritmo DDR

Uma topologia MANET é representada por um gráfico *G = (V,E)* arbitrário, onde *V* é o conjunto de nodos móveis, e *E* é o conjunto de margens. Uma margem existe se e somente se a distância entre os nodos móveis é menor ou igual a um raio *r*. Esse *r* representa a faixa de transmissão de rádio no qual depende das características do canal sem fio incluindo força de transmissão. Conseqüentemente, a vizinhança de um nodo *x* é definida pelo conjunto de nodos que estão dentro do circulo com centro em *x* e raio *r*, onde *x'* é um nodo arbitrário no gráfico G. O grau do nodo *x* em *G* é o número de margens que estão conectados a *x*. O gráfico *G= (V, E)* é chamado de árvore *T* se e somente se *G* está conectado e não contém ciclos. Um nodo é chamado de *beco sem saída,* se ele for um nodo folha na árvore *T* (isso é, seu grau for igual a 1).

Uma floresta *F* é um gráfico no qual os componentes conectados são árvores. É assumido pelo autor desse trabalho que cada nodo móvel *x* gera uma mensagem periodicamente, conhecida como *beacon* B, para os nodos móveis vizinhos que estão dentro de uma faixa de rádio de transmissão direta. O *beacon* é usado para construir uma floresta num modo distribuído. Os intervalos de tempo entre dois *beacons* devem depender de algumas características da rede, como a mobilidade do nodo e a taxa de conexão e desconexão da rede. Existem cinco campos em um *beacon* como mostrado na (Figura 3.7) e descritos logo abaixo.


**Figura 3 7 – Campos de um *beacon***

(NIKAEIN 2000)

1. Número ID da zona (ZID – *Zone ID* ) – Identifica cada árvore de outras árvores. Cada nodo inicia suas ZIDs para seus NIDs. Um ZID é determinado dependendo dos IDs de alguns nodos selecionados pertencentes ao mesmo que

x. O ZID é também usado para distinguir os nodos que não são usados na árvore mas eles estão na faixa de transmissão direta um do outro. Esses nodos são chamados de nodos gateway e a margem que conecta os dois nodos é chamada de *bridge*;

2. Número ID do nodo (NID – *Number ID*) – faz a identificação dos nodos móveis;

3. Grau do NID – (NID_*Deg*) – É o grau associado ao NID no gráfico G;

4. Meu vizinho preferido – (*My_PN*) – é um *flag* que distingue dois diferentes modos : o **modo de eleição do nodo** – que indica o determinado vizinho preferido de *x*, onde nesse caso a *flag* é apontada para 1. E o **modo de encaminhamento de PNs**, que indica que o nodo *x* notifica aos nodos pertencentes a sua árvore sobre os novos membros ou sobre os membros removidos, nesse caso a *flag* é apontada para zero;

5. Vizinho preferido – (PN – *Preferred Neighbor*) – o vizinho preferido de um nodo é o nodo que possui o grau máximo de vizinhança entre os nodos vizinhos. Então a floresta é construída pela conexão. O vizinho preferido de *x* é o nodo que possui características preferidas entre os nodos vizinhos de *x*.

Cada nodo na rede mantém duas tabelas: tabela intra-zona e tabela inter-zona. A tabela intra-zona mantém a informação a respeito dos nodos de uma árvore. Ela contém dois campos denominados: número ID do nodo (NID), que representam o número ID de cada nodo que suporta as árvores da extremidade diretamente com o nodo x. Assim, o número de entrada na tabela intra-zona dá o grau atual de um nodo numa árvore, e o campo com os nodos preferidos aprendidos denominado (*Learned_PN*), que representa os nodos que são alcançáveis indiretamente pelos seus NIDs associados na tabela intra-zona. A tabela intra-zona do nodo é mostrada na (Figura 3.8), ela é simbolizada por Intra_$ZT_x$. O Intra_$ZT_x$ dá a visão atual do nodo *x* relativo à sua árvore, e é atualizado no recebimento dos *beacons*.

Já a tabela inter-zona mantém a informação relativa às zonas vizinhas da zona a que o nodo *x* pertence. A tabela inter-zona do nodo *x* mostrada na (Figura 3.9), e é simbolizada por Inter_$ZT_x$ .


**Figura 3 8 – Tabela Intra-zona**
(NIKAEIN 2000)


**Figura 3 9 – Tabela inter-zona**
(NIKAEIN 2000)

Cada entrada na Inter_$ZT_x$ contém o número *ID* de um nodo *gateway* denominado (*GNID*), o *ID* da zona de seu nodo *gateway* (*NZID*), isto é, o *ID* da zona vizinha e a estabilidade dessa zona vizinha com relação ao nodo *x* conhecida como (*Z_stability*).

### 3.4.2.2 Descrição do algoritmo de roteamento DDR

- **Eleição do vizinho preferido**

Consideremos *x* e *y* como sendo qualquer nodo de um gráfico *G=(V, E)*. O autor assume que inicialmente cada nodo *x* conheça o número ID e o Grau de seus nodos vizinhos. Baseados nessas duas informações, o nodo *x* pode determinar seus PN. O nodo *x* procura um conjunto de nodos cujo grau é igual ao grau máximo da vizinhança. Esse conjunto é simbolizado por $PN=\{y/deg(y)=max(deg(N_x))\}$. Diferenciam-se três casos:

1) Se o conjunto é vazio, então o nodo *x* não possui PN o que significa que não possui vizinhos. Na (Figura 3.10), o nodo *n* não possui vizinho conseqüentemente não possui PN;

2) Se o $PN_x$ possui somente um membro, então esse membro é o *PN* eleito. Por exemplo, na Figura 3.10, o nodo *k* tem 4 vizinhos: *f, c, d, y,* mas o conjunto de $PN_k$ tem apenas um membro que é o nodo *f*;
3) O conjunto de $PN_x$ pode ter mais que um membro que é o caso para o nodo *d*, desde que $PN_d = \{k, c\}$. Isso significa que há mais de um vizinho com o grau máximo da vizinhança. Nesse caso, nós assumimos que o nodo *x* elege o nodo de número *ID* maior. Assim, nodo *d* elege o nodo *k* já que seu número *ID* maior que nodo *c* (com relação à ordem alfabética).

Cada nodo *x* sempre quer criar uma margem entre si e um de seus nodos em que o grau é igual ao grau máximo da vizinhança. Assim, o modo em que o nodo é eleito segue funções crescentes monótonas dependendo de seu grau e seu número ID. Uma floresta é construída depois de conectar cada nodo a sua PN (veja o teorema em DDR (NIKAEIN 2000)).

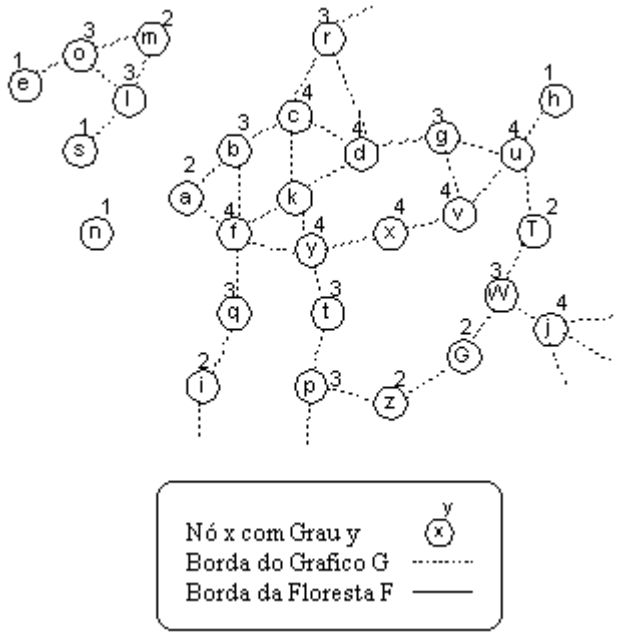



**Figura 3 10 – Gráfico G.**

Legenda : no gráfico G cada nodo é caracterizado pelo seu grau e a letra ao qual representa seu numero ID. Assume que cada nodo conhece o numero ID e o grau de seus nodos vizinhos

(NIKAEIN 2000)

- **Intra-tree clustering**

Assim que o nodo *x* determinar suas $PN_y$ , deve notificar seus nodos vizinhos, especialmente y de sua decisão. No entanto, nodo *x* aponta seus *beacons* para $B_x$ = *(ZID, x, deg(x),1,y)*. Então o nodo *x* atualiza sua tabela intra-zona com relação a *y*. Ao receber os *beacons* de *x*, cada nodo atualiza suas informações em relação a *x* e verifica se eles tem sido escolhidos como *PN* de *x*. Entre os nodos vizinhos de *x*, o $PN_y$ encaminha as decisões de *x* para os nodos que asseguram uma árvore de extremidade com *y* (esses nodos residem na primeira coluna da tabela intra-zona do nodo *y*, isto é, Intra_ZT.NID) pelo apontamento de seus *beacons* para $B_y$ = *(ZID, y, deg(y), 0, x)*. Se o nodo *y* é escolhido como *PN*, ele encaminha suas decisões encapsuladas no campo *PN* em um *beacon*, que é $B_y$ = *(ZID, y, deg(y),0, x : x' : x" : ...)* note que a *flag My_PN* distingue dois modos diferentes: *modo de eleição PN* e *modo de encaminhamento PN(s)*. Outros nodos vizinhos de *x*, adicionam *y* a sua própria tabela de intra-zona. Nesse modo, dizemos que *y* é aprendido (*learned*) para ser *PN* de *x*. Note que o nodo *x* é também aprendido por outros nodos vizinhos de *y*.

O nodo *x* não precisa recalcular seus *PN* a menos que o $\text{Intra\_ZT}_x$ mude. Se a *PN* de *x* permanecer imutável, o campo *PN* de *beacons* de *x* somente conterá o conjunto de *PN*s aprendidas por *x*, esse conjunto é simbolizado por $Learned\_PN_x$.

O nodo *x* encaminha a $Learned\_PN_x$, se esse não for um nodo de beco se saída. Conseqüentemente, cada nodo não adiciona um nodo para sua tabela intra-zona sem conhecê-lo ou aprendê-lo. Nesse modo, todos os nodos pertencentes à mesma árvore são informados sobre a existência de outros nodos. Um outro cenário possível ocorre quando o nodo *y* não pode dar-se ao luxo de ser o nodo preferido de *x*, já que esse não possui energia suficiente ou ele já aceitou ser um *PN* de muitos nodos. Entretanto, o nodo *y* pode decrementar seus graus em relação a seus vizinhos com *PN*. Por exemplo, na Figura 3.11, nodo *k* elege nodo *f* como seu *PN*. Então *k* gera $B_k$ = *(ZID, k, 4, 1, f)* e atualiza sua tabela intra-zona. Note que o nodo *k* seta o campo *My_PN* para 1. Assim, o nodo *f* pode ser aprendido (*learned*) pelos nodos *c, d*.

Tendo um nodo recebido um *beacons* de *k*, nodo *f* atualiza a informação a respeito do nodo *k* na $Intra\_ZT_f$; supõe-se que o *PN* de *f* permanece imutável, aquele é o nodo *y*. Uma vez que os nodos *b, a, q, y, k* escolham nodo *f* como seu *PN* (com mais alto grau de prioridade), o nodo *f* deve encaminhar suas decisões encapsuladas no campo *PN*

ajustando seus *beacons* para $B_f$ *= (ZID, f, 5, 0, b : a : q : y : k)*. Entretanto, os nodos *b, a, q, y* são aprendidos (*learned)* pelo nodo *k*, isto é, $Learned\_PN_k$ *= b, a, q, y*. Conseqüentemente, se o *PN* do nodo k permanecer inalterável, o campo *PN* de *beacons k* somente conterá $Learned\_PN_k$. Desse modo, o conjunto de $Learned\_PN_k$ será aprendido bem como pelos nodos *c, d*. Porém, nodos *c, d* não encaminharão seus *learned PN*, já que eles são nodos de beco sem saída.

Também, os nodos *a* e *b* estão no modo sem roteamento (*non-router*), porque são nodos de *beco sem saída* (sem um nodo *gateway*). A Figura 3.11 mostra a floresta construída. A Tabela 3.4.5 ilustra as tabelas intra-zona dos nodos *f* e *k*, quando suas árvores são estabelecidas.

Como mostrado na Tabela 3.5, a visão do nodo *x* sobre suas árvores consiste de dois níveis: *NID*, *Learned_PN*. O nível *NID* contém os nodos mantidos nas extremidades da árvore com nodo *x*, isto é, o nodo *x* pode alcançá-lo diretamente. O segundo nível *Learned_PN* contém os nodos que são aprendidos (*learned*) pelo nível *NID*, como tratado anteriormente. De fato, o nodo *x* pode alcançá-los via seus *NID* associados em suas tabelas intra-zona. Entretanto, o nodo *x* somente conhece o próximo salto para seus segundos nodos do nível. Assim cada nodo obtém uma visão parcial de suas árvores no sentido que não conhece sua estrutura detalhada da árvore. Por exemplo, na (Figura 3.11), considere o cenário onde o nodo *k* quer comunicar para um de seus nodos pertencentes a sua árvore. De acordo com sua tabela intra-zona (Tabela 3.5 (a)), o nodo *k* pode alcançar os nodos *a, b, q, y, t, x* através do nodo *f* e não *c, d*. Embora os nodos *c, d* possam receber os pacotes de *k*, eles podem simplesmente deixá-los cair , já que *c* e *d* não são definidos como o próximo salto. A (Figura 3.12) mostra a visão do nodo *k* em sua árvore.

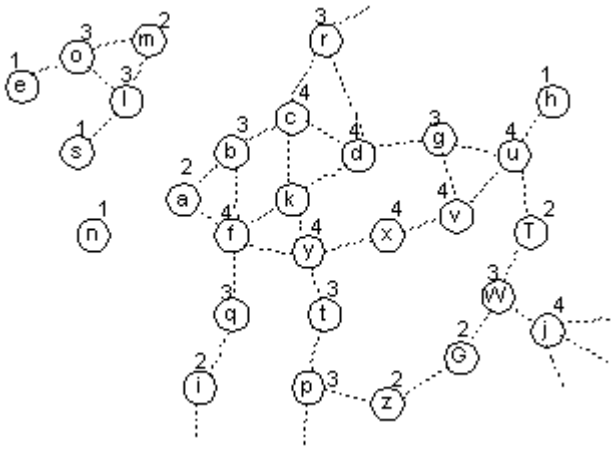



**Figura 3 11 – Floresta construída.**
(NIKAEIN 2000)

**Tabela 3 5 - Tabela de intra-zona dos nodos *k* e *f* relacionada com a Figura 3.4.11**

| NID | LEARNED_PN |
|---|---|
| f | a,b,q,y,t,z |
| c | - |
| d | - |

(a) IntraZTk

| NID | LEARNED_PN |
|---|---|
| y | x,t |
| k | c,d |
| b,a,q | - |

(b) IntraZTf

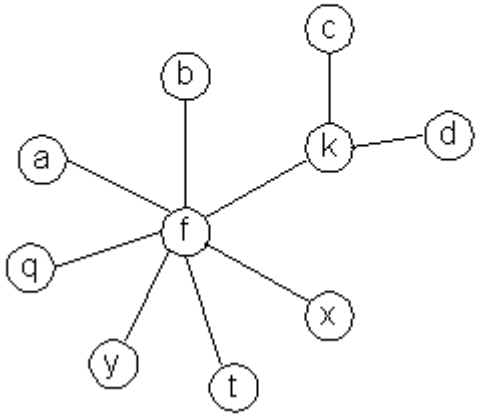



**Figura 3 12 – Visão do nodo *k* de sua árvore.**

(NIKAEIN 2000)

Uma vez que o nodo x determine suas novas $PN_{y'}$, ele deve atualizar sua tabela intra-zona e notificar aos membros permanecidos em sua árvore sobre os membros removidos. Para esse propósito o nodo *x* corta totalmente o galho correspondente ao seu antigo *PNy* na sua tabela intra-zona, que *y -> Learned_PNy*. Então os nodos *x* encaminham o conjunto de membros removidos somente para os nodos permanecidos

no campo *NID* de sua tabela intra-zona apontando seus *beacons* para *Bx= (-1, x, deg(x), 0, y -> Learned_PNy)*. Assim, os nodos permanecidos encaminham os nodos removidos se eles não forem nodos de fim inoperante (*beco sem saída*). *O ZID = -1* significa que cada nodo possui ou tem que remover os nodos especificados no campo *PN* do *beacon* e recalcular seus ZID de qualquer maneira aos membros removidos. Por exemplo, na (Figura 3.11), se o enlace entre os nodos *k* e *f* é quebrado então o nodo *k* remove o nodo *f* e seus *learned* PNs $f \rightarrow a : b : q : y : t : x$ sobre o galho cortado e assim por diante.

- **Inter-tree clustering**

Cada nodo *x* encontra dois casos durante a construção de sua árvore. Primeiramente, pode suceder adicionar alguns nodos à sua árvore e atualizar sua tabela de intra-zona. De outra maneira, o nodo *x* põe os nodos que permaneceram na sua tabela inter-zona. Esses nodos são considerados como nodos *gateways* e eles serão movidos de uma tabela inter-zona para tabela intra-zona sempre que puderem juntar a árvore de *x*. O nodo *x* incrementa o *Z_Stability* de um nodo *gateway g1* em sua tabela inter-zona se o *ZID* atual recebido do *g1* for similar ao *ZID* já existente na *Inter_ZTx*. A estabilidade da zona depende diretamente do número *ZID*. A seção III c5 em (NIKAEIN 2000) oferece explanação mais detalhada de como um nodo determina seu *ZID* e quando ele incrementa a *Z_Stability* de suas zonas vizinhas.

### 3.4.2.3 Construção da floresta:

A floresta é construída pela conexão de cada nodo a seus vizinhos preferidos. NIKAEIN 2000 prova que, qualquer que seja a topologia da rede, sua abordagem produzirá uma floresta. Há uma outra possibilidade para construir uma floresta usando o grau mínimo da vizinhança em vez do grau máximo. Embora garanta a construção de uma floresta, essa é construída numa área onde há menos conectividade dos nodos. De fato, os nodos com um alto grau de conectividade podem não se interceptar na floresta. Por exemplo, na (Figura 3.13) os nodos *f & k, y, c, d* pertencem a diferentes árvores.

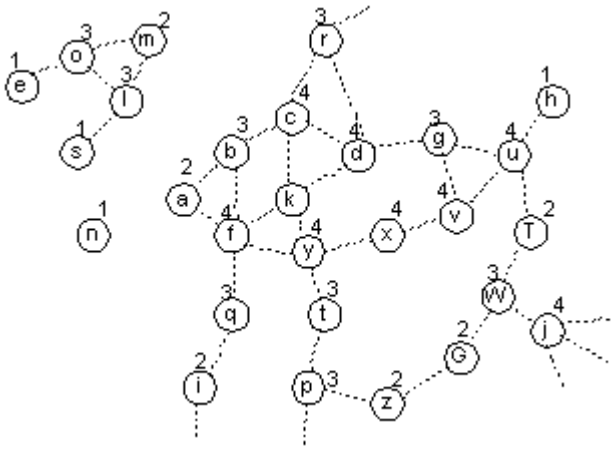



**Figura 3 13 – Floresta construída com o grau mínimo da vizinhança**
(NIKAEIN 2000)

### 3.4.2.4 Nome da zona

O objetivo aqui é demonstrar que cada nodo calcula seus nomes de zona independentemente, e todos os membros da zona alcançam quase o mesmo resultado. O nome da zona dependerá conforme as mesmas características de zona, tais como número ID, Grau do nodo, ou estabilidade de um nodo durante seu tempo de vida na zona. A escolha do número ID é simples e não requer manter outra informação na tabela intra-zona. Entretanto, o grau e a estabilidade do nodo são mais apropriados que o número *ID* com relação a características da zona. Além disso, o nome da zona não deve variar de modo tão freqüente com relação à taxa de conexão e desconexão dos nodos. Para essa finalidade, o nodo *x* seleciona uma sub-rede *s* do conjunto de nodo em sua tabela intra-zona para designar um nome ou mais precisamente um número ID para sua zona. Conseqüentemente, o nodo *x* seleciona o número *ID* mais elevado de *q* em sua tabela de intra-zona. Um modo de estimar a variável que pode ser $q = [n/d]$, onde *n* é o número de bits no número ID e *d* é o grau de compressão da função *hash*. Se a cardinalidade de uma zona é menor que *q* então o nodo selecionado reusa seus números ID respectivamente. Assim, o nodo *x* concatena todos os números ID *hasheds*.

O resultado dessas concatenações gera o *ZID*, por exemplo, na (Figura 3.14) se assumimos que $n=12$ e $d=13$ então $q=4$, assim o nodo *k* seleciona os quatro nodos *y, x, t, q*. Note que, o nodo *x* seleciona esses nodos num modo ascendente. O nodo *k* determina sua *ZID* calculando $h(y)| h(x)| h(t)| h(q)$, onde | denota concatenação. Desse modo, obtem-se o mesmo número de bits num *ZID* como no *NID*. Consequentemente,

cada zona $z_i$ mudará seus números ID com o passar do tempo. Se designarmos $ZID_{t1}$, o *ZID* no tempo *t1* e o $ZID_{t0}$, o *ZID* no tempo *t0* onde *t0<t1*, enquanto o nodo usa uma função similarmente baseada na distância euclidiana para atualizar o valor de *Z_Stability*. Essa função é chamada *Dsimilarity* onde *Dsimilarity* $(ZID_{t1}, ZID_{t0}) <= d_{critic}$. Se a distância entre dois *ZIDs* é a mais afastada distância crítica euclidiana, os nodos apontam o valor de *Z_Stability* para 1, de outro modo esse valor é incrementado. Em ambos os casos o nodo atualiza o *ZID*. Para um melhor entendimento verifique (NIKAEIN 2000).

Assim, concluímos que a estabilidade da zona é estritamente relacionada ao número ZID. De fato, a determinação do *ZID* é baseado em alguns *NIDs* escolhidos aleatoriamente numa árvore. Ele, portanto identifica a zona e pode simplesmente refletir a estabilidade da zona. A Figura 3.14 mostra a situação onde cada nodo aponta um nome para suas árvores.

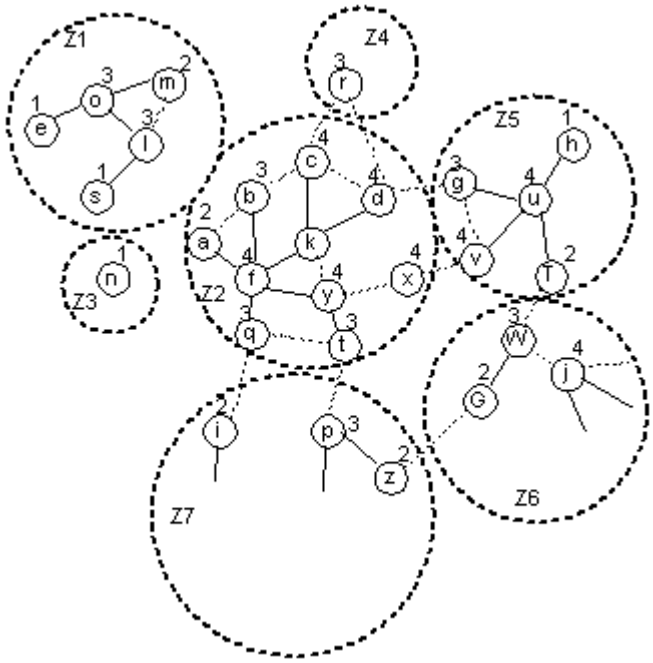



**Figura 3 14 - Divisão da zona**

(NIKAEIN 2000)

### 3.4.3 *HARP – Hybrid ad hoc routing protocol*

O HARP (NIKAEIN 2001) é um protocolo de roteamento de nível de zona hierárquico, onde o roteamento é executado em nível de intra-zona e inter-zona, como

nos dois algoritmos anteriormente explanados (JOA-NG 1999 e NIKAEIN 2000), sendo que o roteamento intra-zona confia num mecanismo pró-ativo (herdado do DDR), já a inter-zona confia num mecanismo reativo. No HARP (NIKAEIN 2001) cada nodo mantém somente a informação de roteamento daqueles nodos que estão dentro de sua zona, e de suas zonas vizinhas, e é responsável pela descoberta e manutenção dos trajetos para satisfazer os requerimentos da aplicação. Evita atraso extra causado por falha no trajeto durante a transmissão de dados, restaura o trajeto antes do período de instabilidade, além de gerar e selecionar trajetos conforme o nível de estabilidade da zona que é definida pela estabilidade da conexão de uma zona com relação a suas zonas vizinhas.

#### 3.4.3.1 Idéia Básica

O HARP visa o estabelecimento do trajeto mais estável de uma fonte a um destino a fim de melhorar atraso do desempenho devido à falha no trajeto. Ele aplica o mecanismo de descoberta do trajeto entre zonas que pretende limitar o *flooding* da rede, e que filtre os trajetos candidatos o mais cedo possível de acordo com os critérios de estabilidade. Como a estabilidade é o parâmetro mais desejado, HARP oferece diferentes mecanismos para antecipar a falha do trajeto utilizando um procedimento de manutenção do trajeto, cuja complexidade é reduzida pela natureza pró-ativa do algoritmo de roteamento dentro de uma zona.

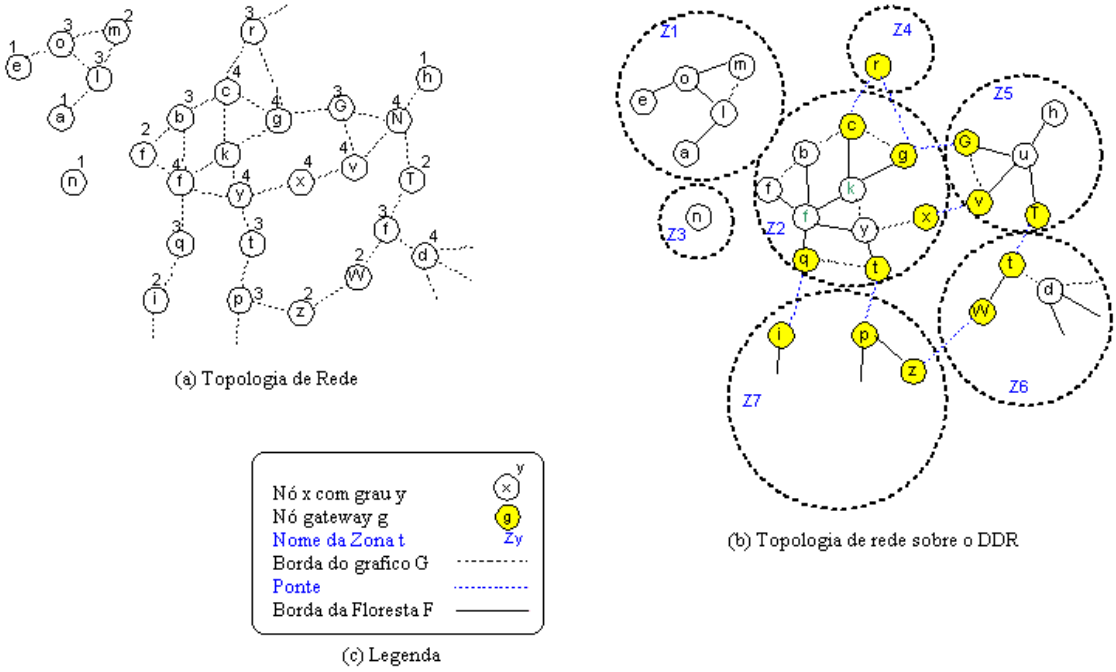
#### 3.4.3.2 Mecanismo de roteamento

O mecanismo de roteamento, como mencionado anteriormente, é executado sobre dois níveis: intra-zona e inter-zona, dependendo se o destino pertence na mesma zona como o nodo expedidor. O roteamento intra-zona envolve somente encaminhamento porque o HARP aplica abordagem pró-ativa dentro de uma zona, como já mencionado, o que significa que geração e seleção das rotas são desempenhadas durante a fase *clustering intra-tree* do DDR (NIKAEIN 2000). Por essa razão, somente a tarefa do nodo dentro de uma zona é para encaminhar os tráfegos de dados ao longo do trajeto pré-computado. O roteamento inter-zona implica, já que

HARP usa uma abordagem reativa entre zonas para gerar rotas e selecionar as mais estáveis de todas para o destino.

O roteamento inter-zona inclui fases de: descoberta do trajeto e manutenção de trajeto. A fase da descoberta do trajeto consiste em duas partes: requisição do trajeto (PREQ) e resposta do trajeto (PREP). O trajeto solicita propagar de zona-a-zona via nodos *gateway* graças tanto a tabela intra-zona quanto a tabela inter-zona, enquanto a resposta do trajeto é *unicasted* de volta do destino para fonte. Dentro de uma zona, o trajeto solicita seguindo a estrutura da árvore fornecida pelo DDR.

Como uma conseqüência, a propagação requisitada do trajeto é limitada a um subconjunto de nodos expedidores. Após esse *flooding* limitado, diversos trajetos podem candidatar-se para um dado destino. O destino escolhe o trajeto mais estável e envia um trajeto resposta de volta à fonte. A manutenção do trajeto fornece diferentes mecanismos para assegurar que pacotes possam ser transmitidos seguramente da fonte ao destino. Cada trajeto é associado com um tempo renovado após qualquer nova fase de descoberta de trajeto ser disparada. Isso é feito para evitar falha no trajeto como a topologia da rede pode mudar após um certo tempo. Os nodos podem ter também comportamento imprevisto que pode causar falha no trajeto. Nesse caso, um procedimento de recuperação reativa do trajeto é provocado além do mecanismo precedente, que pode ser visto como uma recuperação pró-ativa do trajeto.




(a) Topologia de Rede

(b) Topologia de rede sobre o DDR

(c) Legenda

**Figura 3 15 – Infra-estrutura do DDR.**

(NIKAEIN 2001)

**Tabela 3 6 - Tabela de intra-zona dos nodos *k* e f relacionada a Figura 3.4.15 (b)**

| NID | LEARNED_PN |
|---|---|
| f | f,b,q,y,t,x |
| c | - |
| g | - |

(a) Tabela de Intra-zona do nó k: IntraZTk

| NID | LEARNED_PN |
|---|---|
| y | x,f |
| k | c,g |
| b,f,q | - |

(b) Tabela de Intra-zona do nó s: IntraZTf

### 3.4.3.3 Algoritmo de descoberta do trajeto

Como já dito anteriormente, o objetivo principal do algoritmo da descoberta do trajeto é gerar e selecionar o trajeto mais estável entre fone e destino. Suponha que um nodo *s* informa que quer enviar dados a seu destino *d*. Antes de enviar o dado ao nodo *d* o nodo *s* verifica se o nodo *d* existe em sua tabela intra-zona. Se assim, o nodo *s* encaminha os dados para o nodo *d* de acordo com sua tabela intra-zona sem atraso

algum. Demonstra-se esse caso como roteamento intra-zona. Caso contrário, se o nodo *d* pertence a uma zona diferente do nodo *s*, esse encaminha uma requisição de trajeto (PREQ) para toda zona vizinha *z* via nodos *gateway g*. Denota-se esse caso como roteamento inter-zona.

A estabilidade é desconhecida no inicio, mas os nodos fora do *gateway* mudarão o valor da estabilidade quando propagarem a mensagem de requisição de trajeto (PREQ) de zona-a-zona. Cada nodo intermediário distribui a PREQ através de suas zonas para fora do nodo *gateway* de acordo com sua tabela intra-zona. Em contraste, cada nodo *gateway* encaminha PREQ de sua zona para as próximas zonas de acordo com sua tabela inter-zona e atualiza a estabilidade do trajeto.

No recebimento da PREQ por um nodo *gateway g* de dentro da sua zona, esse nodo verifica se o destino *d* pertence a sua tabela intra-zona ou não. E assim, o nodo *g* encaminha sua PREQ de acordo com sua tabela intra-zona; caso contrario *g* distribui essa PREQ por toda sua zona e possivelmente para todas outras zonas vizinhas. A fim de assegurar que a travessia da mensagem PREP de resposta do trajeto faça exatamente o mesmo trajeto de retorno à fonte, cada nodo *gateway* mantém alguma informação do estado do roteamento encontrado no PREQ e PREP para cada trajeto. Essa informação de estado de roteamento inclui: *gateway*-ID de que um pedido de trajeto é recebido (ou corresponde ao *gateway*) e estabilidade do trajeto. O nodo destino *d* eventualmente receberá uma PREQ se for acessível pela fonte. Então o nodo *d* escolhe o trajeto mais estável entre os pedidos recebidos do trajeto. Se mais de um trajeto com mesma estabilidade é encontrado, o nodo *d* escolhe um aleatoriamente. Posteriormente, o nodo *d* cria uma resposta do trajeto (PREP) correspondendo ao trajeto mais estável, e distribui esse PREP de volta ao *gateway* de que recebeu o (PREQ).

Respostas das rotas incluem a descoberta da estabilidade que ajuda o nodo fonte durante o procedimento da manutenção do trajeto. Cada nodo distribui de volta à PREP em direção ao nodo *gateway* correspondente de acordo com cada tabela da zona e a informação do estado do roteamento. Uma vez que a PREP alcance o nodo fonte *s* o mesmo cria um pacote de dados, e envia-os através do trajeto descoberto. Conseqüentemente, cada nodo encaminha o pacote de dados a próximas zonas via *gateway* correspondente em direção ao destino de acordo tanto com a tabela de intra-zona quando com a tabela inter-zona. Além disso, um nodo *gateway* filtra uma requisição de trajeto caso receba uma com um baixo grau de estabilidade. Essa filtragem

é chamada de *Selective filtrering* e garante a qualidade do par de nodos *gateway* mais estável.

#### 3.4.3.4 Manutenção do trajeto

A meta do algoritmo de manutenção do trajeto é a melhora do desempenho de atraso baseado na estabilidade do trajeto. O algoritmo de manutenção do trajeto inclui as fases: *path refreshment, path waiting time, path error,* e *new path discovery*. A fase de *path refreshment* constrói um novo trajeto antes de um período de tempo chamado *path discovery update time*, e então desvia o tráfego para esse novo trajeto no começo do tempo de atualização (*update time*). Esse tempo de atualização da descoberta do trajeto é estimado baseado na descoberta da estabilidade do mesmo. Contudo, embora a fonte possa se manter comunicando com o destino após o tempo de atualização, ele pode enfrentar alta probabilidade de falha do enlace. Para tal, o nodo fonte renova a fase de descoberta do trajeto ao se aproximar o tempo de atualização dessa descoberta. Se uma falha no enlace ocorrer nesse meio tempo, o nodo correspondente mantém o tráfego por uma duração de tempo de espera aguardando receber alguma informação de roteamento correspondente a seu nodo *gateway* alvo, e no mesmo tempo ele envia um *path error* de volta a fonte.

A análise racional para um tempo de espera é que o HARP aplica a abordagem pró-ativa dentro de uma zona, e há uma perspectiva de receber nova informação de roteamento embutida no *beacon* (sinal de alarme) periódico. Além disso, o nodo fonte beneficia-se da estabilidade da descoberta do trajeto para obter a estabilidade real de cada zona durante toda transmissão de dados. Note que a estabilidade é uma função côncava. Por esse propósito, o nodo fonte coloca a descoberta de estabilidade no pacote de dados assim que cada nodo *out gateway* pode verificar se sua estabilidade satisfaz a real estabilidade da próxima zona. Se não, esse nodo *out-gateway* envia de volta um *path error* com a real estabilidade para o nodo fonte de modo para que esse possa atualizar o tempo da descoberta do trajeto adequadamente. O *path error* é *unicasted* de volta para fonte se o trajeto retorno ainda existir, caso contrário ele é encaminhado por *broadcasted* para rede com um tempo de vida (TTL- t*ime to live* Uma vez que a fonte tenha recebido o *path error*, ela inicia o *new path discovery.*

Como exemplo, utilizou-se a (Figura 3.16) que descreve o formato das mensagens *path request* PREQ e do *path reply* PREP. Elas incluem um endereço fonte *src@,* um enderço destino dest@, um número de porta *port#,* um *ID* do nodo *gateway GID*, e uma estabilidade estimada *stability.*

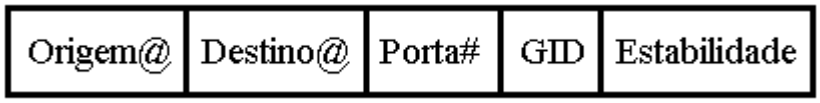



**Figura 3 16 – Campos do *path request* PREQ e *path reply* PREP**

(NIKAEIN 2001)

A (Figura 3.17) mostra os campos de um pacote de dados denominado DPKT. Os campos de um pacote de dados incluem um endereço fonte src@, um endereço destino dest@, um número de porta port#, um id do nodo *gateway* GID, a descoberta da estabilidade *stability*, e os dados *data.*

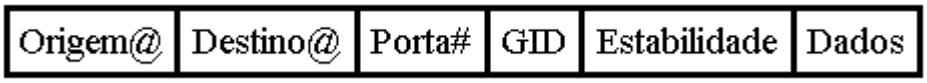



**Figura 3 17 – Campos do pacote de dados DPKT.**

(NIKAEIN 2001)

Por exemplo, na (Figura 3.18), considere o cenário do roteamento intra-zona onde o nodo *s* que se comunicar com um dos nodos dentro da zona $Z_2$, por exemplo. *F, b, q, y, k, c, g, x, t*. De acordo com sua tabela intra-zona (veja Tabela 3.4.7(b), mostrada mais acima), o nodo *s* pode alcançar os nodos *x, t* via *y*, e os nodos *c*, *g* através *k*, enquanto os outros nodos *f, b, k* são diretamente alcançáveis. Conseqüentemente, a tabela de roteamento intra-zona sempre indica o próximo salto para cada destino dentro da zona. Então, se o nodo *s* quer enviar algum dado para o nodo *c*, ele primeiramente encaminha um DPKT incluindo os dados ao nodo *k* e então *k* encaminha um DPKT ao nodo *c* (linhas pontilhadas na Figura 3.18).

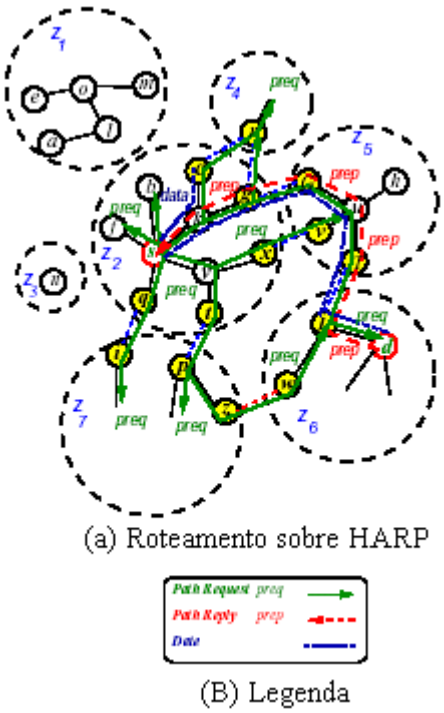



**Figura 3 18 – Roteamento e encaminhamento no HARP**

Legenda Roteamento e encaminhamento no HARP: *path Discovery, path maintenace,* e trafego de entrega de dados. (NIKAEIN 2001)

O roteamento inter-zona ocorre se o nodo destino pertencer a uma zona diferente daquela do nodo fonte. Por exemplo, na (Figura 3.18), o nodo fonte *s* é um membro da zona $Z_2$ enquanto o nodo destino *d* pertence a zona $Z_6$. Conseqüentemente, o nodo *s* enviará uma solicitação de trajeto PREQ através de sua zona. Essa PREQ cruza da zona $Z_2$ para as zonas $Z_4$ $Z_5$ ,$Z_7$ , via nodos *gateway c, g, x, q, t* respectivamente; como isso é ilustrado na Figura 3.18 (seguindo as linhas em azul de *s* até *d*). Esses nodo *gateway* atualizam o campo do *GID* com seu *ID* e armazenam o valor anterior do *GID* pelo meio do *gateway* correspondente. Esses nodos também encaminham o PREQ para suas zonas vizinhas de acordo com sua tabela de inter-zona, isso é, via *gateway r, G, v, i, p* respectivamente. Agora a PREQ atravessa as zonas $Z_5$ , $Z_7$ , $Z_2$ para as zonas $Z_6$ via *Z, T*. Note que o nodo *T* aplica filtragem seletiva para um dos trajetos solicitados recebidos de G e v. Assume-se que o nodo *T* escolhe o nodo *G* como o nodo *gateway* mais estável. Finalmente a PREQ entra na zona $Z_6$ onde o nodo *d* pertence à via *w, j*. Os nodos *w, j* encaminham essa PREQ para *d* de acordo com sua tabela de roteamento intra-zona. Ao receber essa PREQ pelo nodo destino *d*, o mesmo escolherá o mais estável de todos. Nesse exemplo, admite-se o trajeto via *d, j, T, G, g, s* como o trajeto mais estável, de

modo que o nodo *d* cria um trajeto de resposta correspondente ao trajeto PREP (*d@, s@, p#, j@, stability*) escolhido; onde *j@* indica o endereço do *gateway* em que o trajeto escolhido tem sido apenas recebido, e os pontos *stability* para a estabilidade total do trajeto na (Figura 3.18) seguindo as linhas de setas tracejadas em vermelho de *d* a *s*).O nodo *j* atualiza os campos de *src@* e GID no trajeto de resposta para PREP (*j@, s@, p#, T@, stability*), e salva o valor antigo do *src@* por meio do *gateway* correspondente = *@d*. A informação do estado de roteamento para cada nodo *gateway* aponta ao próximo *gateway* em direção ao destino no tempo de transmissão dos pacotes de dados. O nodo *T* também atualiza o trajeto de resposta e passa o PREP para o nodo *G* em vez do nodo *v,* já que esse tem sido filtrado para fora por causa da prioridade da estabilidade. Cada nodo internamente distribui as PREP para o próximo *gateway*, de acordo com sua tabela intra-zona. Uma vez a PREP tenha atingido o nodo fonte *s*, o mesmo cria o pacote de dados DPKT (*s@, d@, p#, g@, stability*, *data*); onde *g@* é o nodo *gateway* em que o trajeto resposta foi recebido e a *stability* indica a descoberta da estabilidade durante a descoberta do trajeto. Então, o nodo *s* envia o DPKT para o destino por uma duração do tempo de atualização da descoberta do trajeto (seguindo as linhas pontilhadas em azul na Figura 3.18). Esse trajeto é renovado se *s* ainda quiser enviar pacotes para *d*.

### 3.4.4 *Mobile agents based routing protocol for mobile ad hoc networks (Ant-AODV)*

Esse protocolo (MARWAHA 2002 A e B) tenta superar desvantagens como redução da capacidade de aproveitamento da rede devido à atualização contínua das tabelas de roteamento dos nodos móveis causada pelo AODV (DAS 2003) e a dependência dos nodos móveis que aguardam a informação das rotas para os vários destinos que são fornecidos pelos agentes formigas no método baseado em formiga *Ant-based routing* (MINAR 1999). Devido a isto, o protocolo busca combinar o método de roteamento AODV (DAS 2003) com *Ant-based routing* (MINAR 1999) para dar origem a o método de roteamento híbrido *Ant-AODV,* hábil a reduzir o atraso fim-a-fim e a latência da descoberta da rota fornecendo alta conectividade quando comparado aos métodos de roteamento AODV e *Ant-based*. O protocolo de roteamento *Ant-AODV* usa troca de mensagens de erro de rota (RERR) para informar os nodos da falha do enlace como no AODV, e mensagens *hello* são espalhadas através de *broadcast* para manter a

tabela dos vizinhos atualizada, para uma escolha aleatória do próximo salto (evitando o nodo previamente visitado) pelos agentes *ants*.

No *Ant*-AODV, um agente *ant* trabalha independentemente e fornece rota aos nodos como mostra a (Figura 3.19). Os nodos também possuem capacidade de descobrir rotas de lançamento *on-demand* (Figura 3.19) para encontrar rotas aos destinos para os quais eles não possuem uma rota de entrada bastante atual. Mesmo se um nodo lança uma requisição de rota (RREQ), para um destino ao qual não possui uma rota bastante atual, a probabilidade de suas respostas de recepção rapidamente dos nodos vizinhos é alta (quando comparado com o AODV), já que um agente atualiza constantemente a tabela de rotas, possibilitando que um nodo fonte efetue a troca de uma rota considerada longa por uma mais atual e mais curta fornecida pelos *ants*. A tabela de roteamento no *Ant*-AODV (MARWAHA 2002 A e B) é comum tanto para os *ants* quanto para os nodos.

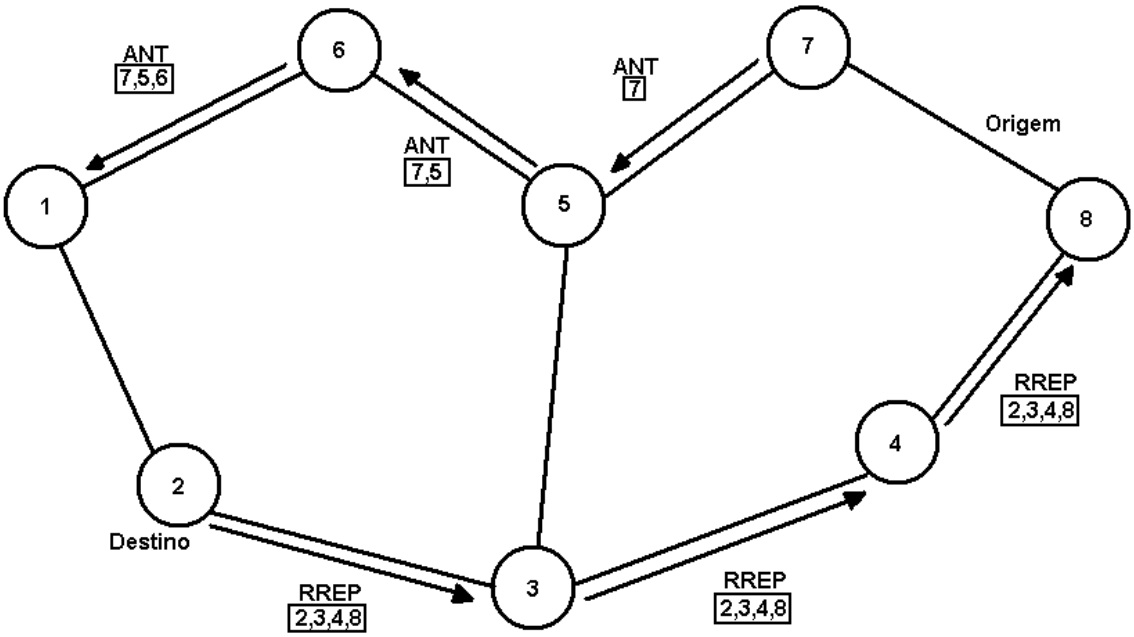



**Figura 3 19 - Propagação da resposta de rota de um pacote.**

(MARWAHA 2002)

### 3.4.5 *Mobile agents for routing topology discovery, and automatic network reconfiguration in ad hoc networks*

A pesquisa publicada em (MIGAS 2003) propõe acessar diferentes modelos de uso dos agentes móveis e estáticos para determinar a melhor rota através da rede *ad hoc* fornecendo benefícios como melhor desempenho, escalabilidade, comunicação fim a fim confiável, além de reduzir possíveis atrasos e minimizar perdas. A idéia é baseada no fato de que cada nodo terá um agente estático que executará monitorando a disponibilidade dos recursos como, por exemplo, disponibilidade de conexão, força de processamento, capacidade de memória e custo. Além dos agentes estáticos, existem os agentes móveis passeando independentemente nas redes *ad hoc* coletando informações dos agentes estático, usando essas informações arrecadadas para determinar o melhor trajeto para roteamento de tráfego na rede.

Para demonstrar a idéia do algoritmo, considere o caso ilustrado na Figura 3.20, onde *A, B* e *C* são nodos móveis e pertencem às redes sem fio $WN_A$, $WN_B$, *e* $WN_C$ respectivamente. Suponha que $WN_A$ intercepta (**.**) com $WN_B$ e que $WN_B$ intercepta (**.**) com $WN_C$. Entretanto *A* pode ver *B* como nodo da zona e vice-versa, e *B* pode ver *C* como nodo da zona e vice-versa. Contudo *A* não pode ver *C* e vice-versa, porque eles não estão na mesma faixa de transmissão, e não podem ver qualquer outro nodo que pertença a uma rede que não intercepta com eles próprios. A intenção do autor é usar uma mistura de agentes móveis e estáticos para descobrir a topologia da rede em qualquer rede móvel *ad hoc* e assim criar um mapa da posição que serão auto-organizadas com o movimento móvel dos terminais.

Uma expansão para o exemplo acima é apresentada aqui e ilustrada na figura 3.4.25. Considere o caso onde *A, B, C, D* e *E* são nodos móveis e pertencem a rede sem fio $WN_A$ , $WN_B$ , $WN_C$ , $WN_D$ , $WN_E$ *e* $WN_F$ . Suponha que $WN_A$ **.** $WN_B$ **,** $WN_B$ **.** $WN_C$, $WN_A$ **.** $WN_D$, $WN_D$ **.** $WN_E$ , $WN_E$ **.** $WN_F$ , $WN_F$ **.** $WN_C$ . No caso de que o nodo *B* queira se comunicar com *E*, ele pode escolher passar tráfego na rede através do nodo *A* ou *C* já que $WN_B$ que o nodo *B* pertence; eles interceptam com $WN_A$ **.** $WN_C$ . Uma decisão deve ser baseada em vários parâmetros tais como: disponibilidade e força de processamento do nodo *C* e *D*, capacidade de memória, e assim por diante. Infelizmente sem uma certa medida o nodo *B* decidiria passar tráfego para rede

aleatoriamente selecionada de um nodo adjacente. Isso pode diminuir a escalabilidade, a disponibilidade e a performance.

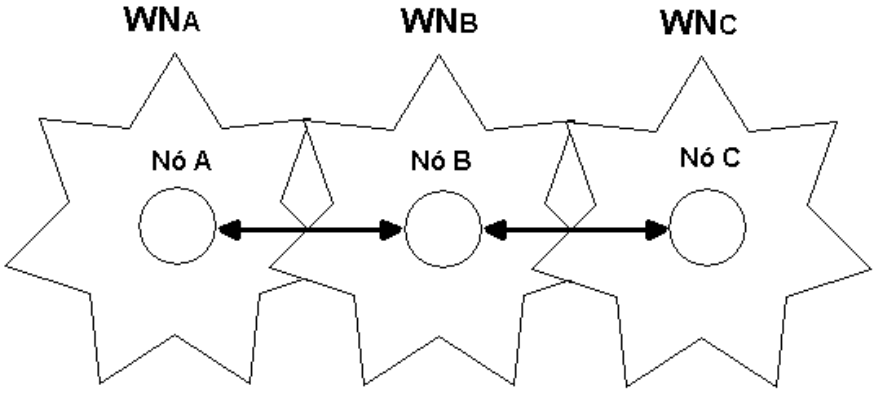



**Figura 3 20 – Descobrindo a vizinhança da rede.**
(MIGAS 2003)

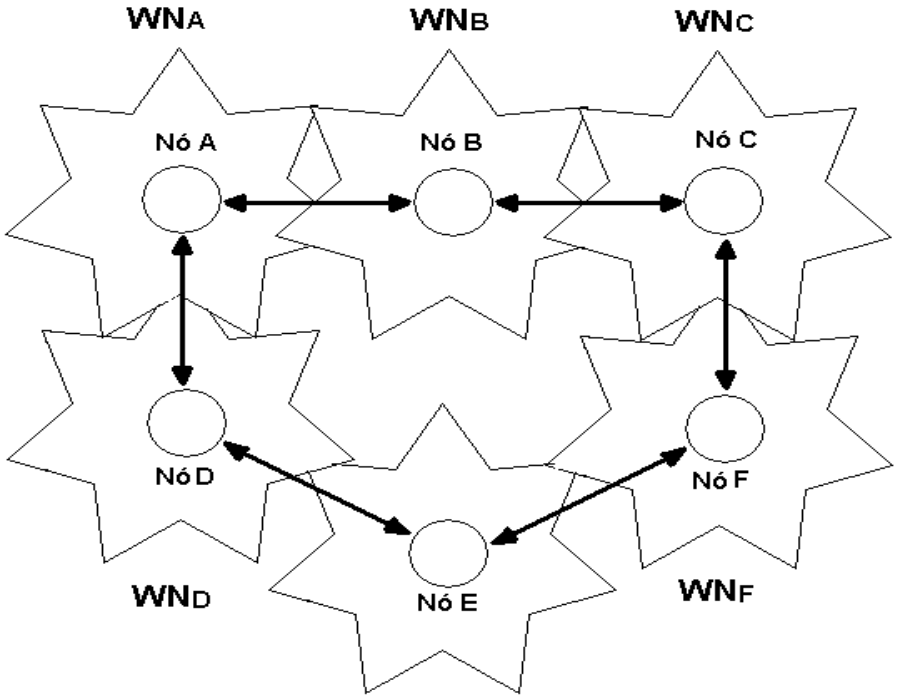



**Figura 3 21 – P assando tráfego na rede a nodos adjacentes.**
(MIGAS 2003)

Nas redes sem fio, devido à própria mobilidade dos nodos, problemas podem aparecer quando um nodo móvel decide se mover de uma rede para outra. Baseado no exemplo simplificado da Figura 3.21. Considere o caso onde o nodo *F* que pertence a

$WN_F$ é alcançável pelo nodo *C* e *E* e decide mover-se para perto do nodo *A*. Nessa nova situação o nodo *f* estará somente alcançado via nodo *A*. Por essa razão, nodo *C* e *E* não o conhecerá. Para isso podemos usar agentes móveis que distribuirão as informações de atualização relativas à posição geográfica de cada nodo móvel.

Cada nodo móvel rodará um agente servidor que fornecerá as funcionalidades básicas para os agentes estáticos e móveis, tais como migração, comunicação e segurança. Os agentes estáticos serão residentes em *hosts* móveis e estarão continuamente executando. Esses agentes serão principalmente responsáveis pelas seguintes operações: manter uma tabela de roteamento, decidir o melhor trajeto para encaminhar o tráfego na rede baseado na informação encontrada na tabela de roteamento; e monitoramento dos recursos de sistemas em termos de capacidade de memória, processamento apto, desempenho da rede, custo e assim por diante. Por outro lado, agentes móveis serão responsáveis pelas seguintes operações. Coleta de informações geradas pelos agentes estáticos, atualização da tabela de roteamento nos *hosts* móveis e descobertas de novas rotas. Eles devem também informar aos agentes estáticos e outros agentes móveis sobre as mudanças na rede.

Agentes estáticos manterão as tabelas de roteamento continuamente, decidindo o melhor trajeto dinamicamente, e monitorando os recursos do sistema periodicamente. Agentes móveis trocarão informações com outros agentes móveis periodicamente, atualizando a tabela de roteamento continuamente, comunicando-se com os agentes estáticos e com outros agentes móveis quando necessário.

## 3.5 CONCLUSÃO DO CAPÍTULO

Para um melhor entendimento desse trabalho de pesquisa, nesse capítulo foram abordadas características de roteamento relacionadas mais explicitamente aos algoritmos de roteamento para redes móveis *ad hoc,* já que este trabalho é voltado a esse domínio.

A complexidade dos algoritmos apresentados é conseqüência da dificuldade associada à dinâmica da topologia das redes *ad hoc*. Isto justifica o interesse que nosso trabalho de pesquisa vai ter para a área, uma vez que a definição de um *framework* associado a estes algoritmos pode auxiliar na definição do algoritmo mais adequado a ser adotado em função das características topológicas da rede considerada

# CAPÍTULO IV

## *FRAMEWORKS* ORIENTADO A OBJETOS

Os *Frameworks* são tipicamente importantes para o desenvolvimento de sistemas abertos, onde tanto funcionalidades quanto arquiteturas devem ser reusadas através de uma família de aplicações relacionadas. Um *framework* orientado a objetos é um conjunto de classes de objetos colaborando entre si, embutidas num projeto abstrato para fornecer soluções para uma família de problemas relatados, tipicamente consistindo de uma fusão de classes concretas e abstratas. As classes abstratas normalmente residem no *framework*, enquanto as classes concretas residem na aplicação. Pode-se, então, conceituar um *framework* como uma aplicação semi-completa de aplicações que contém certos aspectos fixos comuns a todas as aplicações no domínio de um dado problema.

Para (PARSONS 1999) a doutrina central da adoção da tecnologia de objetos, têm sido o reuso do *software* oferecendo potenciais benefícios em termos de produtividade e qualidade. Em geral, a tecnologia orientada a objetos tem intensificado o reuso do *software*, introduzindo a promissora tecnologia de *framework* orientado a objeto que promete a todos o fornecimento do reuso em grande escala.

Dentre as diversas definições de *framework* orientado a objetos encontradas na literatura, podemos destacar:

Para (JOHNSON 1997), *Um framework é um esqueleto de uma aplicação que pode ser personalizada pelo desenvolvimento da mesma*.

Já (GANGOPADHYAY 1995) afirma que *Um framework é projetado para cobrir uma família de aplicações ou subsistemas de um determinado domínio e é tipicamente proferido como sendo uma coleção de classes abstratas interdependentes, em conjunto com suas subclasses concretas*.

Em (KRAJNC 2003) é definido que: *Um Framework é um projeto reusável do todo ou de parte de um sistema que é representado por um conjunto de classes abstratas e ou concretas e a maneira de como seus exemplos interagem*.

E por fim para (PARSONS 1999), *Um framework consiste de frozen spots* (KLING 2004) *(pedaços de software já codificados para serem reusados) e hot spots* (SRINIVASAN 1999) *(elementos flexíveis, permitindo usuários ajustar o framework*

*para aplicação concreta da necessidade). Diferente da maioria das bibliotecas de classes, os frameworks encapsulam fluxo de controle assim como interfaces de objetos, modelando assim o comportamento dinâmico do sistema assim como sua infra-estrutura*.

## 4.1 CLASSIFICAÇÃO DE *FRAMEWORKS*

Na literatura encontra-se uma série de conceitos que classificam os *frameworks* quanto ao seu modo funcional e estrutural. Dentre essas se encontram: os *frameworks* classificados como dirigido à arquitetura ou a dados, os *frameworks* caixa-branca, caixa preta, caixa cinza e ainda os que se classificam como *frameworks* caixa de vidro, os que denominam-se por vertical e horizontal e ainda os categorizados quanto ao seu escopo, conforme trata-se a seguir.

De acordo com (JHONSON 1988) *frameworks* podem ser classificados como: *frameworks* caixa-branca (*white-box frameworks*) para os dirigidos a arquitetura (baseada por herança) e *frameworks* caixa-preta (*black-box frameworks*) para os dirigidos a dados (baseada por composição).

Os *frameworks* caixa-branca baseiam-se fortemente nas características da linguagem orientada a objetos, tais como herança e ligações dinâmicas para alcançar extensibilidade (KRAJNC 2003). De acordo com (FAYAD 1997) as funcionalidades existentes em um *framework* caixa-branca, são reusadas e estendidas por meio de herança das classes bases do *framework* ou anulando métodos *hook* pré-definidos usando padrões como métodos *template*. Um *framework* caixa-branca é considerado como sendo difícil de usar, pois exige esforço no desenvolvimento do código e a geração de subclasses requer um íntimo conhecimento da estrutura interna do mesmo.

Em conseqüência, os *frameworks* caixa-preta são mais fáceis de usar, porém mais difíceis de desenvolver, pois requer que o desenvolvedor defina as interfaces e métodos *hook* que antecipam a extensa faixa do potencial de casos uso (SILVA 2000). Esses *frameworks* suportam extensibilidade definindo interfaces para componente que podem ser ligados dentro de um *framework* através da composição de objeto.

Um modo de caracterizar as diferenças entre *framework* caixa-branca e caixa-preta é observar que no *framework* caixa-branca o estado de cada instância é explicitamente disponível a todos os métodos do *framework*, enquanto num *framework*

caixa-preta, qualquer informação passada aos membros do *framework* deve ser passada explicitamente. Assim, para superar desvantagens apresentadas tanto por parte dos *frameworks* caixa-branca quanto pelos *frameworks* caixa-preta, (FAYAD 1997) categoriza uma outra abordagem de *framework* denominando-a por caixa-cinza. Um bom *framework* caixa-cinza possui flexibilidade e extensibilidade suficiente, e também tem a habilidade de ocultar informações desnecessárias aos desenvolvedores aumentando a reusabilidade e a facilidade de uso (KRAJNE 2003).

Uma outra classificação encontrada em (KRAJNE 2003) denomina que alguns autores adicionam ainda a classificação de *frameworks* caixa de vidro, como sendo um *framework* que possibilita apenas o estudo da implementação, já que apenas permitem ser vistos, mas não alterados.

Têm-se também os *frameworks* classificados quanto a dependência de um domínio denominados por (KIM 2001) como *framework* horizontal e vertical, onde um *framework* dependente de um domínio especifico é categorizado como *framework* vertical, já os *frameworks* independentes de domínios são classificados como horizontais (KIM 2001).

Além dessas classificações, os *frameworks* podem ser categorizados também por seu escopo. Dentre estes, se destacam (FAYAD 1997):

- ***Frameworks* de infra-estrutura de sistemas** (*System infraestructure frameworks*) – são *frameworks* primariamente usados internamente dentro de uma organização de *software* e não são vendidos diretamente aos clientes. Prometem simplificar o desenvolvimento de sistemas de infra-estrutura eficientes e portáveis, como por exemplo, sistemas operacionais, *frameworks* de comunicação;
- ***Frameworks* de integração *middleware*** (*Middleware integration frameworks*) – esses *frameworks* são normalmente usados para integrar aplicações e componentes distribuídos e são projetados para melhorar a habilidade dos desenvolvedores de *software* de modularizar, reusar e extender suas infra-estruturas de *software* para trabalhar em um ambiente distribuído. Como exemplo temos os *frameworks* ORBs, de *middleware* orientados a mensagem e a bases de dados transacionais;

- ***Frameworks* de aplicações de empresas** (*Enterprise application frameworks*) – são *frameworks* que tratam de amplos domínios de aplicações, como exemplo, temos telecomunicações, indústria, engenharia financeira e são base para as atividades de negócios da empresa.

Com relação aos *frameworks* de infra-estrutura de sistemas e *frameworks* de integração *middleware*, os *frameworks* de aplicação de empresas, pode-se dizer que esses requerem maior investimento tanto para serem desenvolvido quanto para serem comprados. Entretanto podem fornecer um retorno substancial no investimento, já que suportam diretamente o desenvolvimento de aplicações de produtos e de usuário final. Em contrapartida, as duas primeiras abordagens de *frameworks* focam largamente sobre o desenvolvimento de interesses internos do software (SILVA 2000).

## 4.2 CICLO DE VIDA DE *FRAMEWORKS*

O ciclo de vida dos *frameworks* difere notoriamente daqueles componentes de programas em linguagens convencionais, pois podem ser reusáveis em muitas aplicações. Segundo (SILVA 2000), um *framework* nunca é um artefato de *software* isolado, mas sua existência está sempre relacionada à existência de outros artefatos, originadores do *framework*, que são gerados a partir dele ou que exercem alguma influência na definição de estruturas de classes do *framework*.

À medida que um *framework* caixa-branca vai adquirindo maturidade, poderá então ser convertido para um *framework* caixa-preta, sendo que esse processo pode ocorrer durante a própria implementação do *framework*. Entretanto (JOHNSON 1988), determina que é bastante difícil apontar de modo antecipado como isso acontecerá, visto que muitos *frameworks* nunca se tornam caixa-preta.

Dentre os fatores que podem exercer influência na definição da estrutura de um *framework* temos (SILVA 2000):

**Atuação do desenvolvedor** – toda abordagem de desenvolvimento de *framework* necessita a figura do desenvolvedor, pois

ele é o responsável por definir quais as classes que irão compor a estrutura do *framework*, suas responsabilidades e o nível de flexibilidade oferecidas aos usuários do *framework*. O desenvolvedor exerce responsabilidade na construção e na manutenção do *framework*;

**Aplicações do domínio tratado** – um *framework* é formado por um domínio de aplicações que atuam como fonte de informação desse domínio, sendo que o *framework* é constituído pela etapa de generalização do domínio dessas aplicações;

**Aplicações geradas sob o *framework*** – o objetivo fundamental de um *framework* é gerar reusabilidade na produção de diferentes aplicações, a fim de reduzir tempo e esforço por parte dos usuários / desenvolvedores de tais aplicações. No entanto, devido à complexidade do domínio, muitas vezes não se consegue abstrair completamente toda realidade abordada, tornando o *framework* apenas uma descrição aproximada do domínio construído a partir das informações até então disponíveis. Necessitando para tal o desenvolvimento de aplicações que completem os procedimentos e estrutura de dados existentes no *framework*. Sendo essa etapa muitas vezes esclarecedora, gerando conhecimento de informações não tratadas durante a construção do domínio.

## 4.3 DESENVOLVIMENTO DE *FRAMEWORKS*

Dentro do todo estudado até aqui, podemos perceber que o desenvolvimento do *framework* é mais complexo do que as aplicações específicas do mesmo domínio.

A criação do *framework* deve ser seguida por algumas etapas de desenvolvimento como sugere (BOSCH 1999). Essas etapas são seguidas por:

- **Análise do domínio** – é a fase de descrição do domínio em estudo que será abrangido pelo *framework*. Nessa etapa, devem-se levantar os requisitos e

identificar conceitos relacionados com o domínio, informações essas que podem ser obtidas por intermédio de consulta a aplicações anteriormente desenvolvidas, livros, relatórios ou por especialistas no domínio. Como resultado desse processo obter-se-á a análise do modelo de domínio;

- **Projeto da arquitetura** – essa etapa, o analista define um estilo de arquitetura adequado ao desenvolvimento do *framework*, ou seja, deve determinar se o sistema será, por exemplo, cliente-servidor, ou baseado em camadas, ou utilizará padrões como CORBA, etc. A partir desse ponto é que o projeto de mais alto nível do *framework* é elaborado;

- **Implementação do *framework*** – esse é o estágio da codificação das classes abstratas e concretas que foram definidas para o *framework*;

- **Teste do *framework*** – é a fase que avalia a usabilidade do *framework,* estabelecendo se o mesmo realmente oferece as funcionalidades que foram planejadas. Contudo, não é trivial decidir se uma entidade é utilizável ou não. A real avaliação do *framework* resume-se no desenvolvimento de aplicações que o utilizem;

- **Produção de aplicações de teste** – para avaliar a usabilidade de um *framework*, é interessante que se faça o desenvolvimento de uma aplicação de teste baseada no *framework*. Conforme o tipo de aplicação poder-se-á testar diferentes aspectos desse *framework*. A projeção de aplicações de teste permite verificar se o *framework* necessita ser re-projetado ou está suficientemente bom para ser utilizado.

## 4.4 METODOLOGIA DE DESENVOLVIMENTO DE *FRAMEWORKS*

A pesquisa de tecnologias de reuso tem sido estudada dentro da Engenharia de Software desde a década passada, incluindo biblioteca de classes, padrões de projeto, *frameworks*, e componentes. A produtividade do vasto desenvolvimento do software depende do grau de quanto os componentes desse software são efetivamente reusáveis.

A fim de fornecer reusabilidade de software, muitas metodologias sistemas de bibliotecas bem como a técnica de programação orienta a objetos tem sido intensivamente aplicada. (LEWIS 1995). Entretanto, alguns inconvenientes dessas metodologias exigem que os programadores estejam cientes de que bibliotecas de programação possuem limites em expandir o software para vários ambientes. Para tal inconveniente à aplicação de *framework* orientada a objetos foi proposta para assim superar esses problemas oferecendo um efetivo reuso (KRAJNC 2003).

Caracterizando a pouca existência de metodologias voltadas ao desenvolvimento de *frameworks,* (JOHNSON apud SILVA 2000) cita a metodologia projeto dirigido por exemplo (*example-driven desing*), cujas características são voltadas na análise do maior número possível de aplicações do domínio abordado. Sugere a seguinte seqüência de etapas para a produção de um *framework* (JOHNSON apud SILVA 2000):

- Análise do domínio – onde as aplicações existentes são a principal fonte de informação. Nessa etapa deve-se assimilar as abstrações já conhecidas; reunir exemplos de programas que poderiam ser desenvolvidos a partir do *framework* (no mínimo quatro); e avaliar a conformidade de cada exemplo;
- Definição de uma hierarquia de classes que possa ser especializada para abranger os exemplos (um *framework*) – para essa etapa é sugerido a utilização de *design patterns*;
- Testar o *framework* utilizando-o no desenvolvimento de exemplos de aplicações que abrangem o domínio – nessa fase, deve-se implementar, testar e avaliar cada exemplo usado na primeira etapa utilizando para isso, o *framework* desenvolvido.

Uma outra metodologia utilizada para o desenvolvimento de *frameworks* é a metodologia Projeto Dirigido por Pontos de Flexibilização (*hot spot driven design*) definida por (PREE apud SILVA 2000), que evidencia a busca de *hot spots* (partes da estrutura de classes do *framework* como classes e métodos, que devem ser mantidas flexíveis, para possibilitar sua adaptação a diferentes aplicações do domínio). Essa metodologia estabelece praticamente a mesma seqüência de etapas da metodologia

anteriormente citada. Entretanto, seu foco é na identificação das partes da estrutura que se diversifica de aplicação para aplicação, e a seleção de soluções de projeto adequado a esses casos (SILVA 1997). O projeto dirigido por *hot spot* estabelece algumas etapas abaixo descritas (PREE apud SILVA 2000):

- Na primeira etapa feita a identificação de classes. A partir de informações de especialistas do domínio, define-se uma estrutura de classes;
- Na segunda etapa são identificados, com o auxilio de especialistas do domínio, os *hot spots*, onde são destacados os aspectos diferentes de cada aplicação e se define o grau de flexibilidade que deve-se manter em cada caso;
- A terceira etapa trata da re-elaboração do projeto. Consta da modificação da estrutura de classes inicialmente definidas de modo a suportar a flexibilidade pretendida. Nessa etapa definir-se-á o que o usuário do *framework* deve fazer para gerar uma aplicação;
- A quarta etapa consiste num refinamento da estrutura do *framework* a partir de novas intervenções de especialistas do domínio. Se após isso o *framework* for avaliado como satisfatório, a conclusão do *framework* é definida.

Uma outra metodologia também citada em (TALIGENT apud SILVA 2000) é a metodologia de projeto da empresa *Taligent*. Nessa metodologia, a visão de desenvolver um *framework* que cubra as características e necessidades de um domínio é substituída pela visão de produzir um conjunto de *frameworks* estruturalmente menores e mais simples, que quando usados em conjunto, originarão as aplicações.

Um ponto importante no desenvolvimento do *framework* é tornar seu uso o mais simples possível, por meio da redução da quantidade de código que o usuário deve produzir devido à disponibilidade de implementações de classes concretas, bem como com a redução do número de classes que devem ser criadas e com a minimização do número de métodos que devem ser sobrepostos (TALIGENT apud SILVA 2000).

Ainda em (TALIGENT apud SILVA 2000) encontra-se a descrição da seqüência de passos que essa metodologia propõe. Dentre elas tem-se:

**Identificar e caracterizar o domínio do problema**

- Analisar o domínio e identificar os *frameworks* necessários (para o desenvolvimento de aplicações), o que pode incluir *frameworks* desenvolvidos e a desenvolver;
- Verificar soluções existentes;
- Apontar as principais abstrações;
- Estabelecer o limite de responsabilidades do *framework* que está se desenvolvendo;
- Validar esse conjunto de informações com especialistas do domínio.

**Deferir a arquitetura e o projeto**

- Aperfeiçoar a estrutura de classes obtida no passo anterior, centrando atenção em como os usuários interagem com o *framework* (por exemplo, quais as classes que o usuário interage e produz, bem como que métodos devem ser sobrepostos);
- Refinar o projeto com o uso de *design patterns;*
- Validar essas informações com especialistas do domínio.

**Implementar o *framework***

- Implementar as classes principais;
- Testar o *framework* gerando as aplicações;
- Solicitar a terceiros o procedimento de teste;
- Iterar para refinar o projeto.

**Desdobrar o *framework***

- Fornecer documentação na forma de diagramas, receitas de como usar o *framework* e exemplos de aplicações simples que complementem o *framework* incluindo o código da implementação;
- Manter e atualizar o *framework* conforme as regras a seguir:
    - Corrigir erros imediatamente;
    - Adicionar novas características ocasionalmente;
    - Alterar a interface do *framework* tão infreqüentemente quanto possível (é sempre melhor adicionar novas classes e métodos do que remover os existentes).

## 4.5 CONCLUSÃO DO CAPÍTULO

Dentre as metodologias abordadas nesse capitulo é importante ressaltar que nenhuma estabelece técnicas de modelagens as quais sejam capazes de conter a descrição de projeto de *frameworks* ou detalha o processo de desenvolvimento desse.

Visando o objetivo principal deste trabalho, iremos, no capítulo a seguir, estabelecer a proposta do *framework*, levando em conta o domínio da aplicação considerado, a saber, os algoritmos de roteamento para redes *ad-hoc*.

Faz-se importante mencionar sobre uma outra tecnologia de programação orientada a aspectos (AOP – *Aspects Oriented Programming*), a qual vem conquistando um importante e reconhecido espaço dentro da Engenharia de Software pelo fato de disponibilizar mecanismos para a decomposição dos elementos e das relações pertinentes ao problema como a tecnologia de programação orientada a objetos, e também por fornecer mecanismos para separar os interesses ortogonais ao problema (CHAVES 2003). Neste trabalho, porém, optamos pela tecnologia de objetos, justificando a larga usabilidade dessa dentro da comunidade científica, fornecendo potenciais benefícios em termos de produtividade e qualidade. Em geral, acredita-se que a tecnologia orientada a objetos juntamente com a tecnologia de *frameworks* seja capaz de atender os objetivos atuais dessa pesquisa conforme fundamentamos no discorrer deste capítulo.

# CAPÍTULO V

## MODELO DO FRAMEWORK

### 5.1 INTRODUÇÃO

Conforme já mencionado, vale lembrar que, por mais que o ambiente MANET seja produtivo, a natureza dinâmica de sua topologia dificulta a realização do roteamento *multi-hop* (ONISHI 2001). Devido a esse fator, diversos trabalhos de pesquisa, como (JOA-NG 1999, NIKAEIN 2001, NIKAEIN 2000, MARWAHA 2002 A e B e MINAR 1999) têm sido desenvolvidos com o intuito de oferecer, entre outros, um algoritmo de roteamento que defina a topologia da rede atendendo às melhores condições de características qualitativas e quantitativas exigidas. Assim, no capítulo 3, verificou-se que os algoritmos buscam propor soluções diferentes utilizando técnicas similares apresentando vantagens e desvantagens de acordo com situações específicas da rede. Portanto, com o intuito de oferecer uma solução, propomos a possibilidade de se obter uma ferramenta que além de agregar dois ou mais algoritmos de roteamento, possa mais adiante determinar a escolha do protocolo a ser executado no instante que a rede necessitar do perfil oferecido pelo mesmo. Para tal, introduzimos a proposta de uma estrutura denominada FRAd-hoc (*Framework* de Roteamento *Ad-hoc*) que implementa características da tecnologia de *frameworks* orientado a objetos conforme ilustra Figura 5.1.

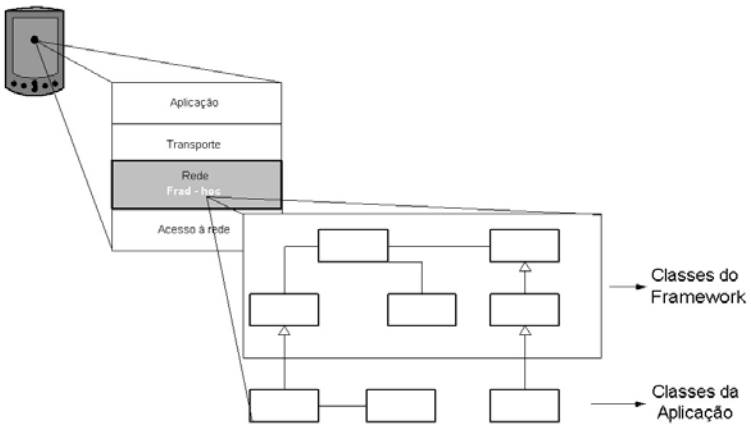



**Figura 5. 1 – *Framework* orientado a objetos**

Nosso objetivo é oferecer um *framework* que agregue as características genéricas ao domínio tratado na etapa de análise e projeto do *framework* proposto, servindo de

estrutura base ao desenvolvimento de qualquer outro algoritmo que implemente peculiaridades da abordagem híbrida. De modo a produzir e disponibilizar artefatos de software reutilizáveis.

## 5.2 ETAPA DE ANÁLISE E PROJETO DO *FRAMEWORK*

Conforme descrito no capítulo 4, existem algumas etapas as quais devem ser seguidas e que fazem parte de um modelo de desenvolvimento de um *framework* simplificado. A seguir discorre-se sobre cada uma dessas etapas no que se refere ao desenvolvimento deste trabalho de pesquisa.

A abordagem de desenvolvimento do *framework* proposto utilizou-se da metodologia de projeto dirigido por exemplo (*example-driven desing*) (JOHNSON 1988) citada no capítulo 4, pois a base desta pesquisa utiliza exemplos já desenvolvidos conforme sugere tal metodologia.

Essa abordagem divide-se em três fases:

- a primeira fase trata da análise do domínio onde as aplicações existentes são a principal fonte de informações;
- a segunda fase abrange a definição da hierarquia de classes que generalizam o domínio investigado;
- a terceira etapa, intitulada "teste do *framework*", consiste em utilizar o mesmo para o desenvolvimento de exemplos de aplicações dentro do domínio estudado.

### 5.2.1 ANÁLISE DE DOMÍNIO

De acordo com o que foi dito no início da seção, a análise de domínio refere-se à descrição do tema que se pretende abranger com o desenvolvimento do *framework*. Para tal, faz-se necessário identificar os requisitos, as características e conceitos relacionados com o domínio e isso pode ser feito, por exemplo, através de estudos bibliográficos, como é o caso demonstrado nesta subseção.

Demonstra-se por intermédio do Quadro 5.1, uma comparação entre os protocolos de roteamento estudados no capítulo 3, considerando algumas diferenciações e as principais características dos algoritmos em estudo. A fim de identificar aspectos comuns ao domínio consideraram-se os seguintes pontos:

1. Característica de abordagem pró-ativa e reativa;

2. Suporte a múltiplos trajetos;
3. Faz *flood* limitado para transmitir informações de roteamento ou solicitação de rotas;
4. Aplica estratégia de consumo de energia e questões de segurança;
5. Executa roteamento em dois níveis;
6. Utiliza agentes móveis e estáticos para efetuar comunicação;
7. Aponta sistema de roteamento orientado a curtos trajetos;
8. Os nodos utilizam mensagens de requisição de rotas ao destino;
9. Mensagens de erro de rota (RERR) são utilizadas para informar os nodos sobre uma falha no enlace local;
10. Mensagens *Hello* são utilizadas para manter a tabela dos vizinhos atualizada;
11. Utiliza sistema de posicionamento global (GPS –*Global Position System*);
12. Baseia-se no conceito de zonas;
13. Baseia-se na metáfora de formiga.

**Quadro 5.1 – Comparação das características de roteamento**

| Protocolo | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DDR | xx | x | - | x- | x | - | - | - | - | x | - | x | - |
| HARP | xx | x | x | x- | x | - | - | x | - | x | - | x | - |
| ZHLS | xx | x | - | - | x | - | x | - | - | x | x | x | - |
| Ant-AODV | xx | - | - | - | - | x | x | x | x | x | - | - | x |
| Mobile agents for routing topology discovery, and automatic network reconfiguration | - | x | x | x | - | xx | x | - | - | - | - | - | x |

**Quadro 5 1 - Comparação das características de roteamento**

Legenda: x = possui a característica

- = ausência da característica

Por meio dessa comparação, percebe-se, que além das características equivalentes adotadas pelos algoritmos em estudo (apresentadas no Quadro 5.1), cada protocolo de roteamento busca superar alguns problemas. Exemplos desses problemas são o desperdício dos recursos da rede sem fio, a falta de escalabilidade em relação à freqüente mudança topológica da rede e a conexão fim-a-fim apresentadas nos

protocolos de roteamento, e.g. (GERLA 2002 e MINAR 1999) que aplicam abordagem puramente pró-ativa ou reativa.

Dentre os protocolos de roteamento explanados na seção 3.4 de descrição do domínio, escolheu-se, para dar continuidade a esta pesquisa, os algoritmos de roteamento DDR (NIKAEIN 2000), HARP (NIKAEIN 2001) e Ant-AODV (MARWAHA 2002 A e B). Justifica-se a escolha do DDR (NIKAEIN 2000) por esse servir como base complementar ao HARP (NIKAEIN 2001) gerando a estrutura lógica do mesmo com relação às propriedades da rede (número de nodos na rede, número de comunicações, freqüência de mudança topológica). Além disso, esse oferece suporte a múltiplos trajetos, redução no custo de manutenção de overhead e o consumo dos recursos de rádio conduzindo a uma rede autônoma. Já o HARP (NIKAEIN 2001) é selecionado por ser apropriado a ambientes em que se deseja obter manutenção adiantada e estabilidade nos trajetos o que não é o caso do DDR (NIKAEIN 2000) e ZHLS (JOA-NG 1999). Outro algoritmo que se destaca é o Ant-AODV (MARWAHA 2002 A e B) por apresentar característica híbrida como o DDR (NIKAEIN 2000) e o HARP (NIKAEIN 2001) e ainda empregar o uso de agentes móveis sendo mais adequado a ambientes com pequenas topologias, bem como àqueles que necessitam da alta conectividade dos nodos.

### 5.2.2 DEFINIÇÃO DA ARQUITETURA DO FRAD-HOC

A arquitetura desenvolvida é classificada como um *framework* de infra-estrutura de sistema, caixa-branca, dirigido à arquitetura, pois deverá ser usado dentro de uma organização de *software* agregando classes e métodos abstratos, sendo que suas aplicações serão geradas com base na criação de subclasses das classes do *framework,* conforme ilustra a Figura 5.2.

Assim, seguindo a análise efetuada em seções anteriores definimos de modo genérico, que o domínio analisado engloba dois tipos de protocolos de roteamento: os algoritmos de roteamento de níveis de intra-zona e inter-zona, e os algoritmos que utilizam o auxilio de agentes móveis. No primeiro caso, o roteamento executado na intra-zona confia num mecanismo pró-ativo enquanto o executado na inter-zona confia num mecanismo reativo No segundo caso, o roteamento é confiado inicialmente ao auxilio de agentes móveis que são responsáveis tanto pela descoberta, quanto por manter as tabelas de roteamento dos nodos atualizadas, podendo esses também iniciar a

descoberta de uma rota a um determinado destino, quando necessário. Fundamentado nessa definição, determina-se que o FRAd-hoc tem como responsabilidade inicial compor e fornecer uma estrutura que servirá como base agregadora das características em comum dos algoritmos de roteamento estudados, fornecendo suporte necessário ao desenvolvimento de outros protocolos de roteamento.

Através da Figura 5.2 podemos vislumbrar o núcleo em comum dos conceitos e termos que são essenciais para executar o domínio do problema como um todo. Com essa modelagem generalizamos o conjunto de algoritmos de roteamento híbrido que tem em comum a divisão da rede em zonas, bem como os que fazem utilização de agentes móveis no estabelecimento de rotas para um destino.

O FRAd-hoc é composto de cinco classes que oferecem mecanismos básicos para o desenvolvimento de algoritmos de roteamento de abordagem híbrida. Cada classe dessa estrutura desempenha uma seqüência de métodos responsáveis pela correta comunicação e execução do roteamento. Dentre elas temos: Classe *Node*, responsável pelos mecanismos de comunicação através de troca de mensagens entre os nodos executado pelo método *sendMsg()*. Essa classe *Node* possui também métodos que indicam uma possível falha de rota, caso uma rota deixe de ser válida no momento de uma transmissão de dados, por exemplo. Além disso, o *framework* possui métodos responsáveis por atualizar a tabela de rotas e por estabelecerem a descoberta de um trajeto a um determinado destino, além de métodos que retornam resposta de uma rota válida quando solicitado por um nodo destino, como é o caso dos métodos *repRoute(), updateTabRoute(), updateIntraZT(), e pathDiscovery()*. Já as classes *IntraZT e InterZT*, são responsáveis por agregar métodos que executam funções de construção e atualização de zonas, como é o caso do método *buildIntraZT(), buildInterZT(), e updateIntraZT()*.

As classes *Agent* e *TabRoute* contém os métodos que serão utilizados por algoritmos que aplicam o paradigma de agentes móveis. Sendo que ambas agregam métodos que efetuam requisição e resposta de trajeto e manutenção e atualização do trajeto estabelecido (veja Figura 5.2).

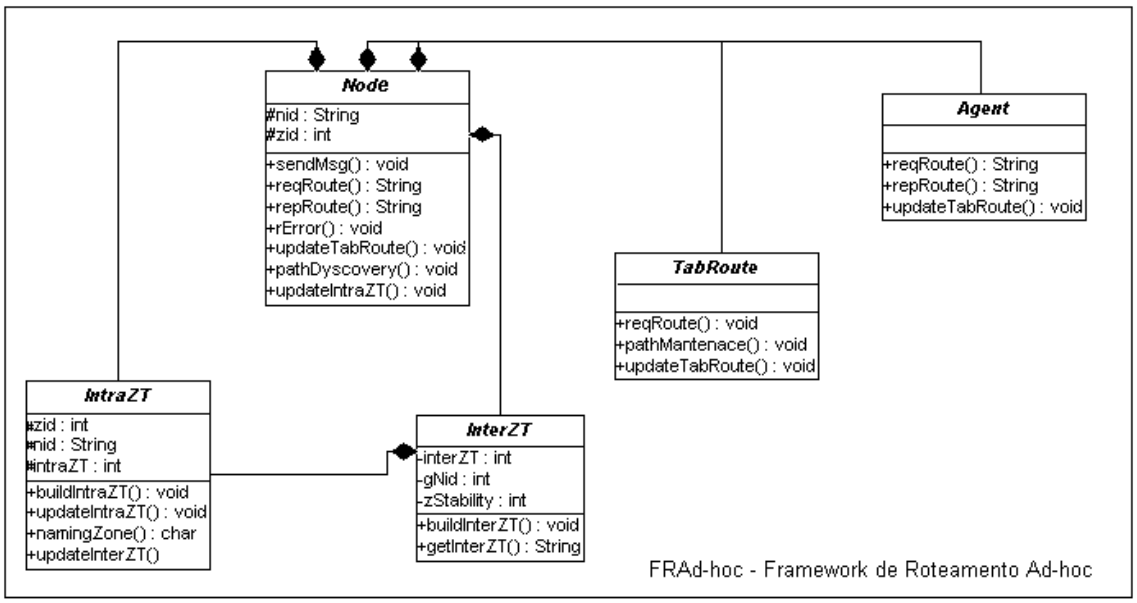



**Figura 5. 2 – Diagrama de classes do *framework* de roteamento *ad-hoc***

A dinâmica do *framework* pode se dar a partir da classe *Node*, através do encaminhamento de mensagens que possibilite que um nodo conheça outros nodos o qual compartilham do mesmo canal de freqüência, denominando esses como seus vizinhos diretos. No caso dos algoritmos que dividem a rede em zonas (veja Figura 5.3), as informações recebidas pelos nodos vizinhos, de modo geral, são armazenadas dentro de uma tabela denominada por *intraZT*. Isso ocorre após a execução do método de construção de zona, *buildIntraZT()*, que por sua vez, pode executar o método de construção de inter-zona denominado por *buildInterZT()*, caso um nodo *gateway* seja detectado. Tendo um nodo, conhecimento de seus vizinhos, ele pode executar o método de requisição de rotas, *reqRoute()* consultando inicialmente sua tabela de intra-zona. Caso a rota solicitada pelo nodo não seja conhecida por sua tabela de *IntraZT*, o mesmo pode disparar um método de requisição de rotas para sua tabela *InterZT*. Se a *InterZT* não obtém conhecimento do nodo ao qual deseja-se estabelecer comunicação o nodo poderá, então, iniciar um novo processo de envio de mensagens para assim verificar se houve alguma mudança na rede.

No caso dos algoritmos que utilizam agentes móveis, a Figura 5.4 demonstra que um nodo, ao conhecer seus vizinhos, pode receber constantes visitas de agentes móveis, os quais efetuam comparação de sua tabela de roteamento com a tabela de roteamento dos nodos visitados, oferecendo dessa forma, a atualização das rotas válidas a um destino. Contudo, caso o nodo necessite de uma rota que não esteja definida em sua

tabela de roteamento, o mesmo poderá encaminhar uma requisição de rota a seus nodos vizinhos, podendo abortar tal operação caso um agente móvel atualize-a antes de receber uma resposta válida ou se oferecer uma rota de trajeto mais curto.

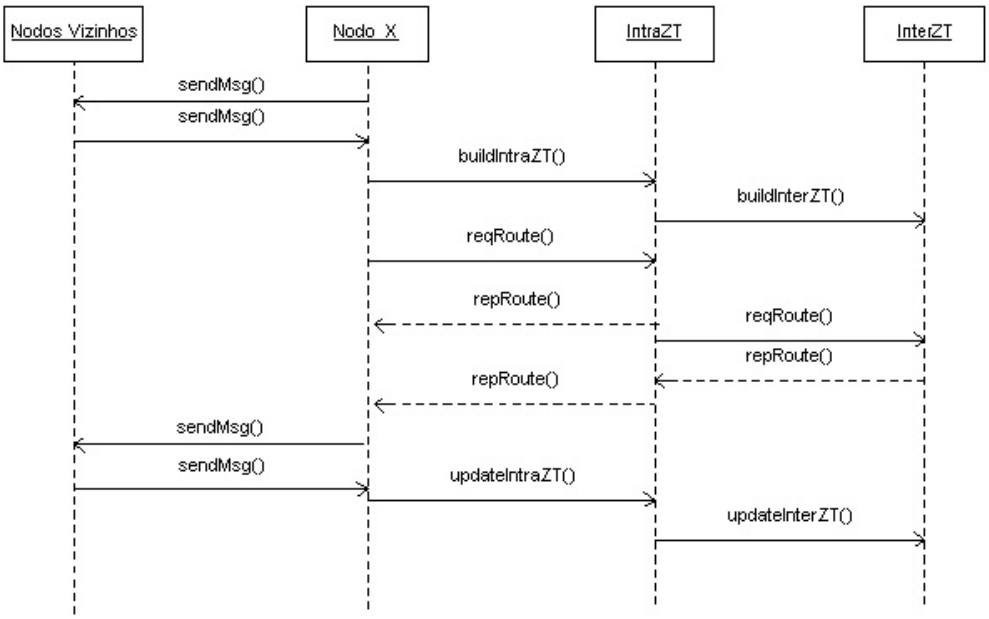



**Figura 5. 3 – Diagrama de seqüência do FRAd-hoc executando algoritmos que dividem a rede em níveis de zonas.**

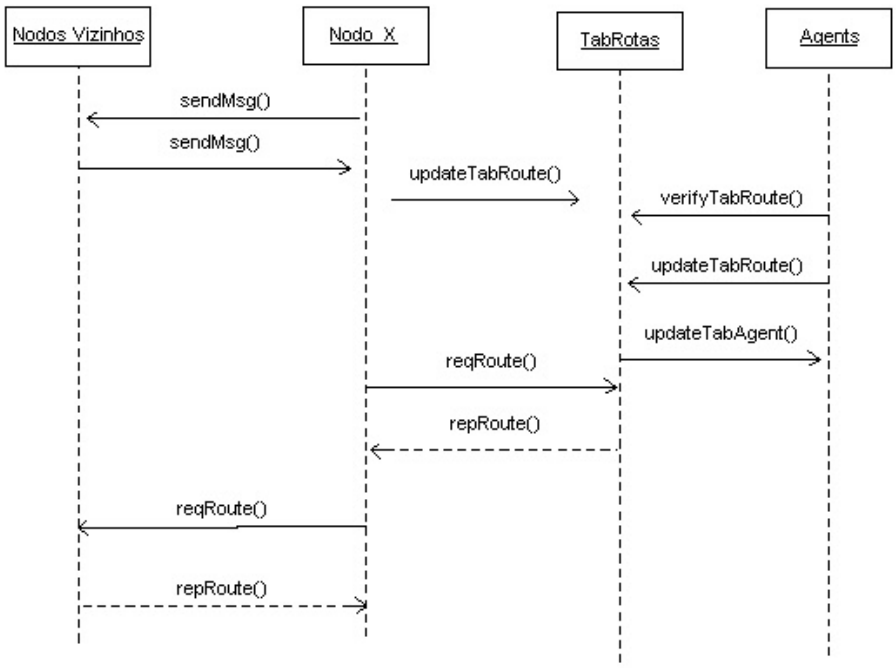



**Figura 5. 4 - Diagrama de seqüência do FRAd-hoc executando algoritmos que utilizam agentes móveis**

### 5.2.3 RESULTADOS DE ANÁLISE

Nesse estágio busca-se demonstrar a usabilidade do *framework* proposto, verificando através do desenvolvimento de algoritmos de roteamento, se a estrutura proposta oferece as funcionalidades planejadas. Como primeiro exemplo de aplicação desenvolvida, apresentamos o DDR – *Distributed Dynamic Routing Algorithm* (NIKAEN 2000). Logo então, apresenta-se a modelagem estática e dinâmica dos algoritmos HARP (NIKAEN 2001).

Devido a limitações enfrentadas, essa seção não abordará a implementação das classes do algoritmo HARP nem a modelagem e implementação das classes do algoritmo Ant-AODV conforme definido na seção de análise do domínio. Assim essa etapa será agregada aos trabalhos futuros.

- **Especializando o FRAd-hoc através do DDR.**

A Figura 5.5 apresenta a estrutura de classes do FRAd-hoc abordado na seção anterior e a estrutura de classes da aplicação desenvolvida sob o mesmo. Pode-se verificar de imediato a especialização de quatro classes concretas criadas pelo usuário

para obter as funcionalidades exigidas pelo algoritmo, sendo que três delas são herdadas do FRAd-hoc, demonstrando conforme desejado, clara evidência de reuso.

A seqüência de atividades dos métodos (veja Figura 5.6) implementados pelas classes do algoritmo DDR (NIKAEN 2000), se dá a partir da classe *NodeDdr*, onde a mesma inicia executando um método de troca de mensagens com seus nodos vizinhos denominado por *sendMsg()* responsável pela comunicação entre os nodos. Tendo um nodo o conhecimento de seus nodos vizinhos torna-se então habilitado a executar uma série de métodos iniciando pela execução do método responsável por determinar a escolha do nodo vizinho preferido denominado por *determinePn()* (veja Figura 5.7), conforme definido em (NIKAEN 2000). Logo então, é executado o método *createBeacon()*, responsável por gerar uma mensagem a qual será encaminhada para os nodos vizinhos contendo as informações de identificação da zona, identificação do nodo, grau do nodo, e nodo preferido (nodo que possui maior número de vizinhos) (NIKAEN 2000).

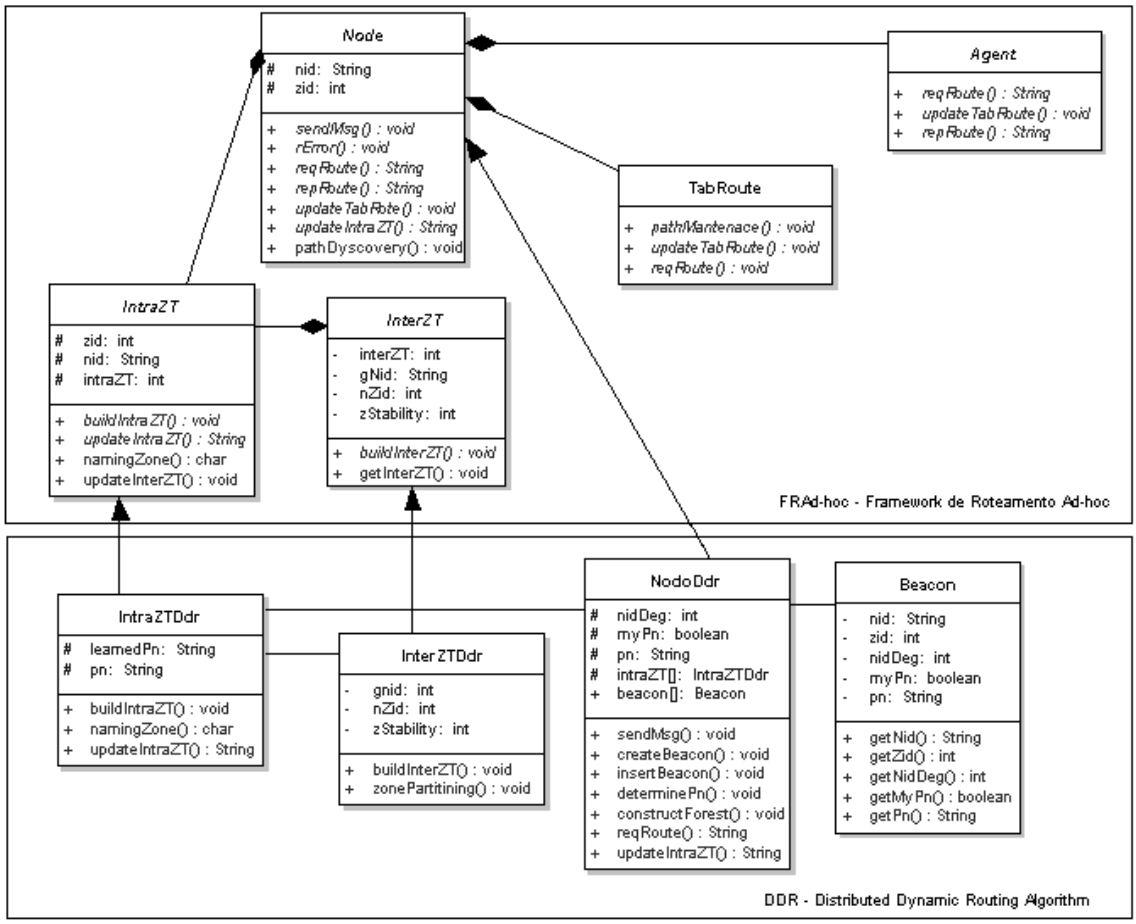



**Figura 5. 5 - Diagrama de classes da especialização do Frad-hoc através do algoritmo DDR**

As informações encaminhadas pelos nodos vizinhos através de mensagens *beacons* são armazenadas na tabela de intra-zona de cada nodo através do método *insertBeacons()* (veja Figura 5.9). Sob posse de tal informação com respeito a seus vizinhos um nodo pode então dar seqüência a construção da sua intra-zona através do método *buildIntraZT()* (veja Figura 5.8), caso o nodo já possua uma tabela de intra-zona válida, o mesmo executará apenas uma atualização adicionando ou removendo nodos que não pertencem mais a essa tabela de intra-zona. Após a construção de intra-zona, a classe IntraZT executa o método que gera o nome dessa zona através do método *namingZone()* (veja Figura 5.10) e conseqüentemente constrói a inter-zona através dos nodos que permaneceram na tabela de intra-zona os quais são denominados nodos *gateways*. Estes nodos podem ser movidos para tabela de intra-zona sempre que puderem se juntar à árvore de um nodo *x* (NIKAEN 2000).

Através da Figura 5.7, podemos verificar mais precisamente a dinâmica do método *determinePn()*, responsável por efetuar a eleição do nodo vizinho preferido. Para esse método são definidos três casos: o primeiro caso verifica se o conjunto dos nodos vizinhos (pnX) do nodo *x* é vazio, ou seja, mesmo não possui nodos vizinhos. Conseqüentemente não há nodo preferido.

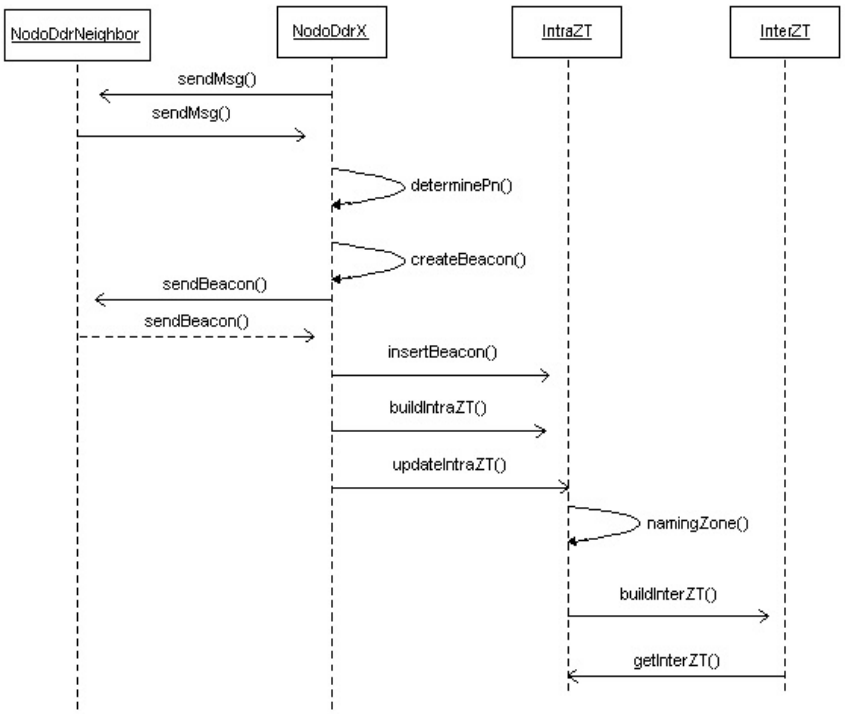



**Figura 5. 6 - Diagrama de seqüência do algoritmo DDR**

Contudo, o segundo avalia se o conjunto de vizinhos (pnX) é unitário. Em caso positivo, esse será definido como nodo preferido do nodo *x*. E por fim, se nenhuma das informações anteriores forem verdadeiras, temos o caso em que, se o conjunto de nodos vizinho obtiver mais de um elemento o nodo deve eleger o membro que possui o maior número identificador (NID – *identifier number* ) (NIKAEN 2000).

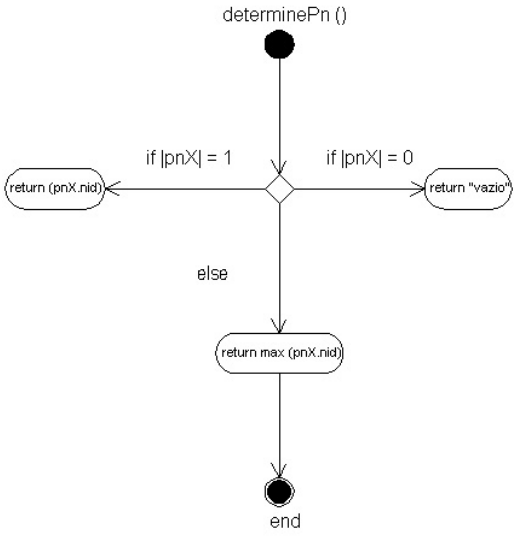



**Figura 5. 7 – Diagrama de atividades do método que elege o PN de um nodo**

A Figura 5.8 representa, de modo genérico, a atividade executada pelo método que estabelece a construção da zona, segundo estabelecido por (NIKAEN 2000). Para que seja possível essa construção, um nodo deve conhecer basicamente dois níveis: os NIDs (*node id*) dos seus vizinhos e dos vizinhos do nodo eleito como nodo preferido, segundo (NIKAEN 2000), por *learnedPn* de um nodo.

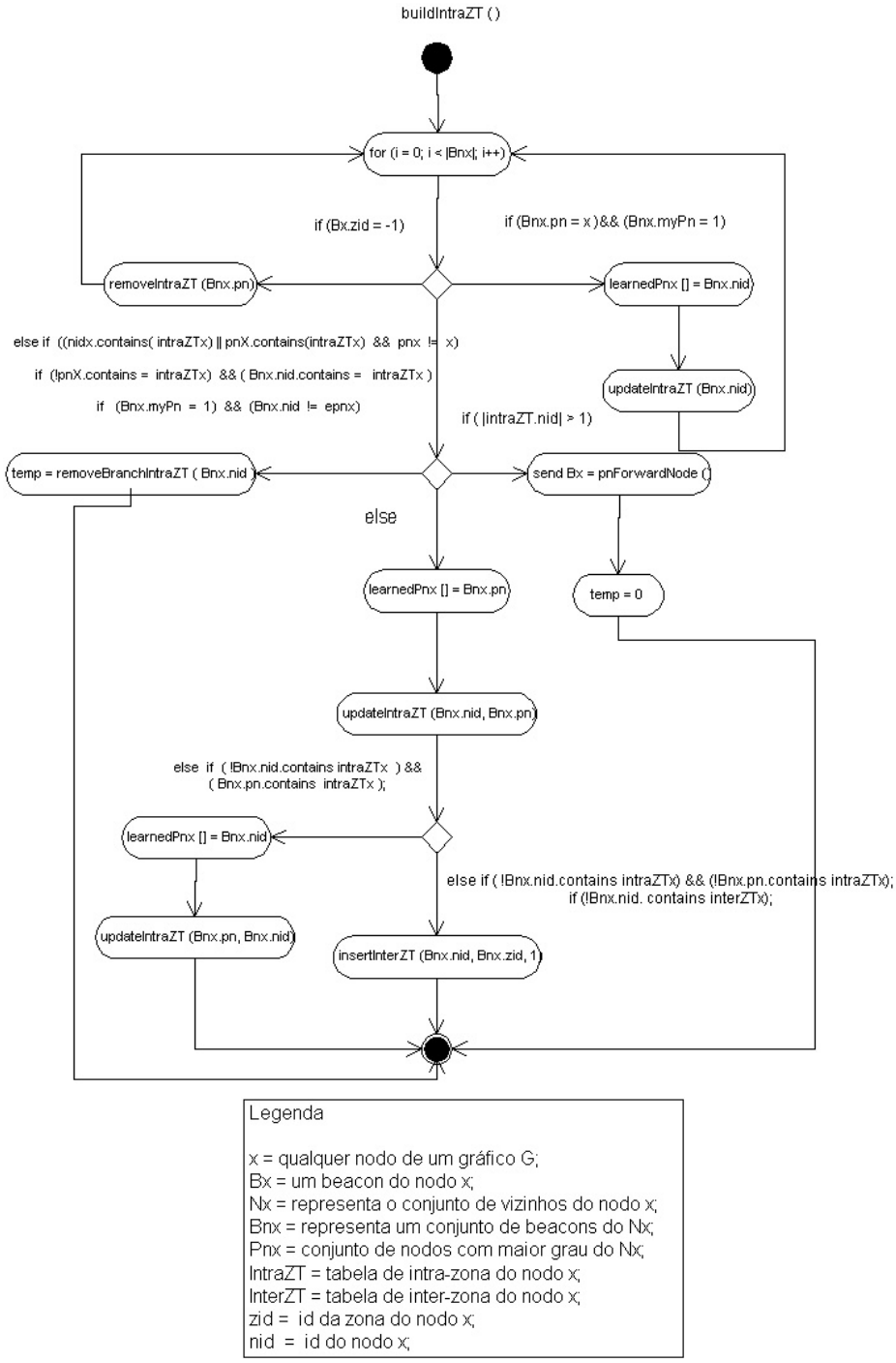



**Figura 5. 8 – Diagrama de atividades do método de construção da intra-zona**

Aqui a Figura 5.9 ilustra o método *insertBeacon()*, responsável por adicionar as informações recebidas através de mensagens encaminhadas pelos nodos vizinhos.

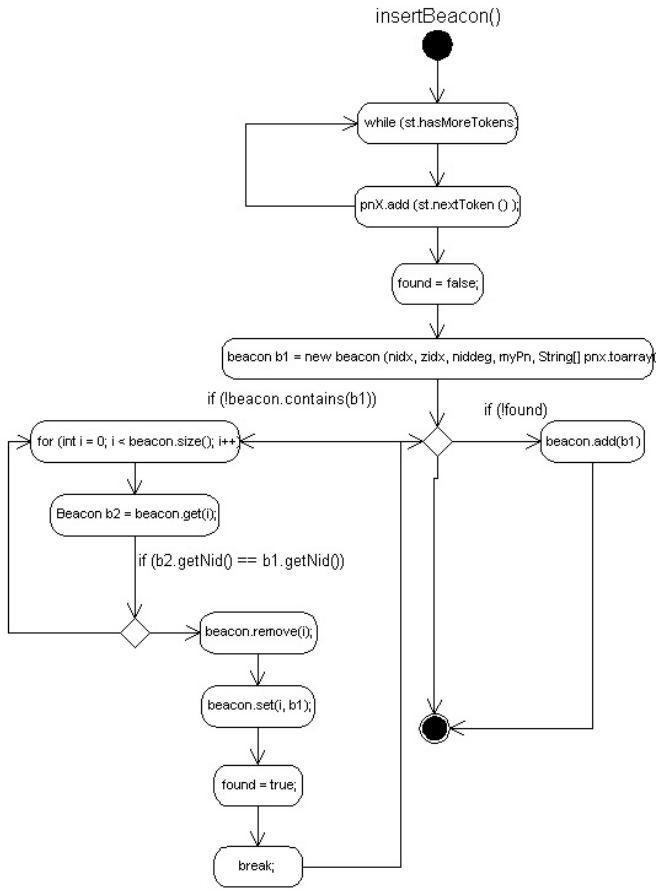



**Figura 5. 9 - Diagrama de atividades do método de inserção de *beacons***

A Figura 5.10 demonstra que cada nodo calcula o nome de sua zona independentemente, e esse dependerá de algumas características definidas em (NIKAEN 2000).

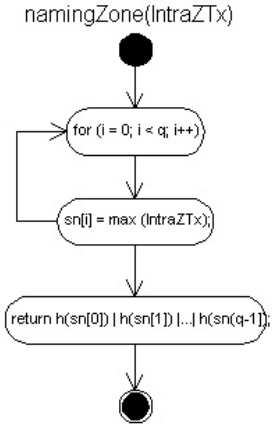



**Figura 5. 10 - Diagrama de atividades do método de nomeação da zona**

- **Especializando o FRAd-hoc através do HARP**

O HARP visa o estabelecimento do trajeto mais estável a fim de melhorar o atraso do desempenho devido alguma falha do trajeto. Nessa etapa, busca-se apresentar modelagem estática e dinâmica que representa os processos aos quais implementam o algoritmo de roteamento HARP abordado no capítulo 3.

Conforme já mencionado na seção 3.4, o HARP, como o DDR executa o mecanismo de roteamento sobre dois níveis denominados por intra-zona e inter-zona. Contudo, aqui daremos atenção apenas ao mecanismo realizado sobre o nível de inter-zona. As questões de roteamento sobre o nível de intra-zona são herdados do algoritmo DDR, já tratadas na etapa anterior. Por essa razão a tarefa do nodo dentro da zona é para encaminhar os tráfegos de dados ao longo do trajeto pré-computado. Já na inter-zona implica roteamento, pois o HARP utiliza uma abordagem reativa entre as zonas para gerar rotas, e selecionar as mais estáveis para o destino (NIKAEIN 2001). O roteamento inter-zona inclui fases de descoberta do trajeto e manutenção do trajeto. A fase de descoberta consiste na requisição e resposta do trajeto e a fase de manutenção preocupa-se em oferecer mecanismos que assegurem a validade de um trajeto entre o nodo fonte e o nodo destino durante o tempo de comunicação.

Através da Figura 5.11 pode-se constatar que esse algoritmo especializa três classes que agregam o *framework* proposto, sobrepondo alguns métodos oferecidos pelas classes herdadas da estrutura, denominados como genéricos e outros métodos e classes específicas do algoritmo. Dentre essas, pode-se mencionar a classe *Pkt* e *Beacon*

responsáveis por desempenharem funções as quais serão utilizadas na requisição e resposta de trajetos.

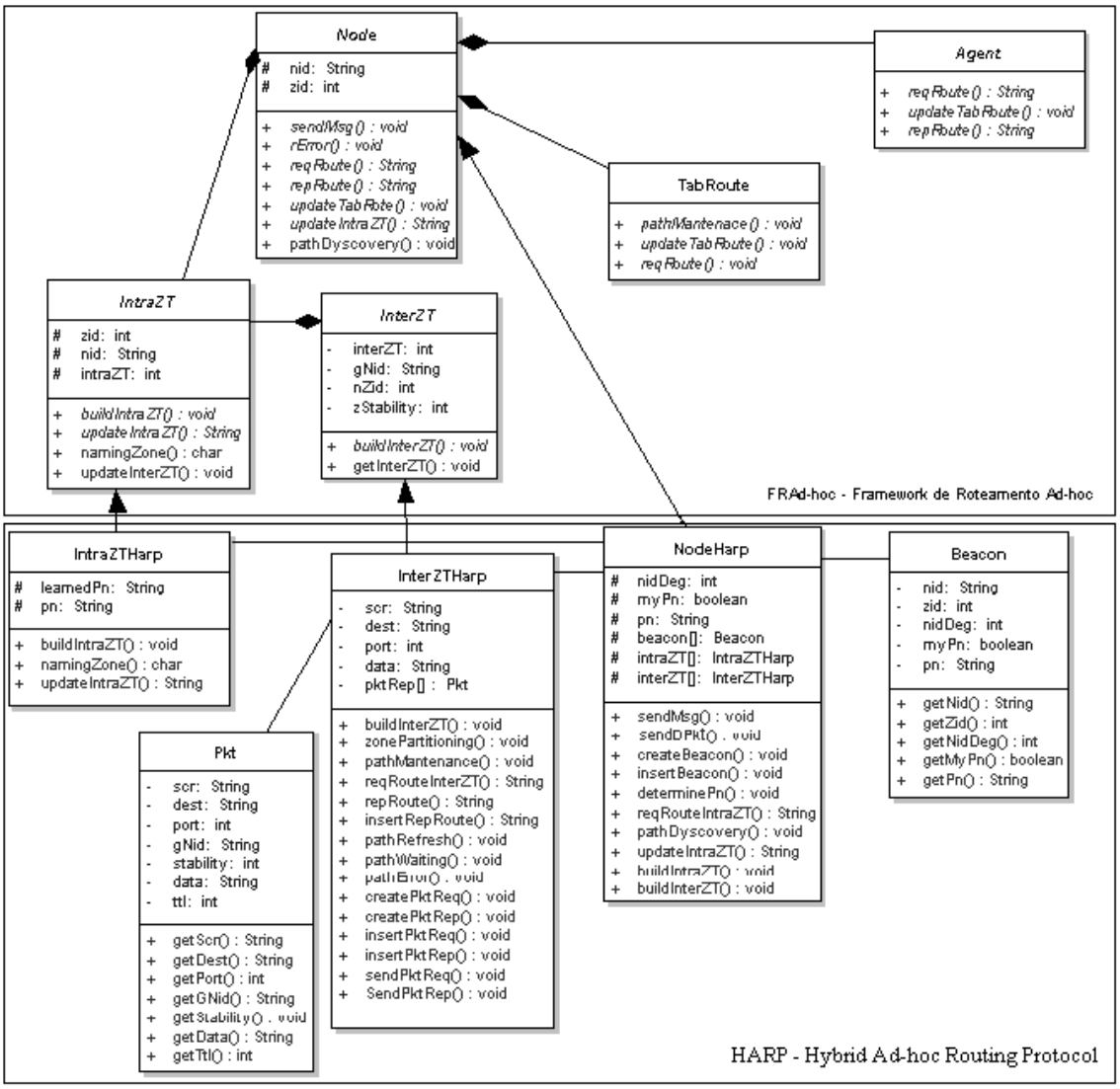



**Figura 5. 11 - Diagrama de classes da especialização do FRAd-hoc pelo algoritmo HARP**

O algoritmo de descoberta do trajeto consiste no encaminhamento de uma requisição de rota para os nodos *gateways* de uma zona e o recebimento de uma resposta de trajeto através desses nodos. O objetivo principal desse algoritmo é gerar e selecionar o trajeto mais estável entre a fonte e o destino (NIKAEIN 2001). Já, a manutenção do trajeto aplicada no HARP tem a função de fornecer mecanismos que asseguraram a transmissão dos pacotes da fonte a um destino. Esse mecanismo é iniciado à medida que um trajeto é descoberto e é mantido somente se o nodo fonte e destino desejam manter a comunicação.

A Figura 5.12 ilustra a seqüência dos métodos responsáveis por gerar e selecionar o trajeto mais estável entre a fonte e um destino que pertence a uma zona diferente do nodo fonte. Assim, é definido que o nodo fonte envia uma requisição de rota para sua tabela de inter-zona que, por sua vez, cria um pacote de requisição de rotas e o encaminha para suas zonas vizinhas através dos nodos *gateways* existentes em sua tabela de inter-zona. Logo, cada nodo *gateway* verifica se o nodo destino requisitado encontra-se em sua tabela de intra-zona. Caso essa hipótese seja verdadeira, o nodo *gateway* cria e encaminha um pacote de resposta de trajeto para a inter-zona do nodo fonte e assim, o nodo fonte criar e encaminha um pacote com os dados conforme trajeto estabelecido. Caso contrário, o nodo *gateway* deve encaminhar a requisição de trajeto recebida para os nodos *gateways* de suas zonas vizinhas.

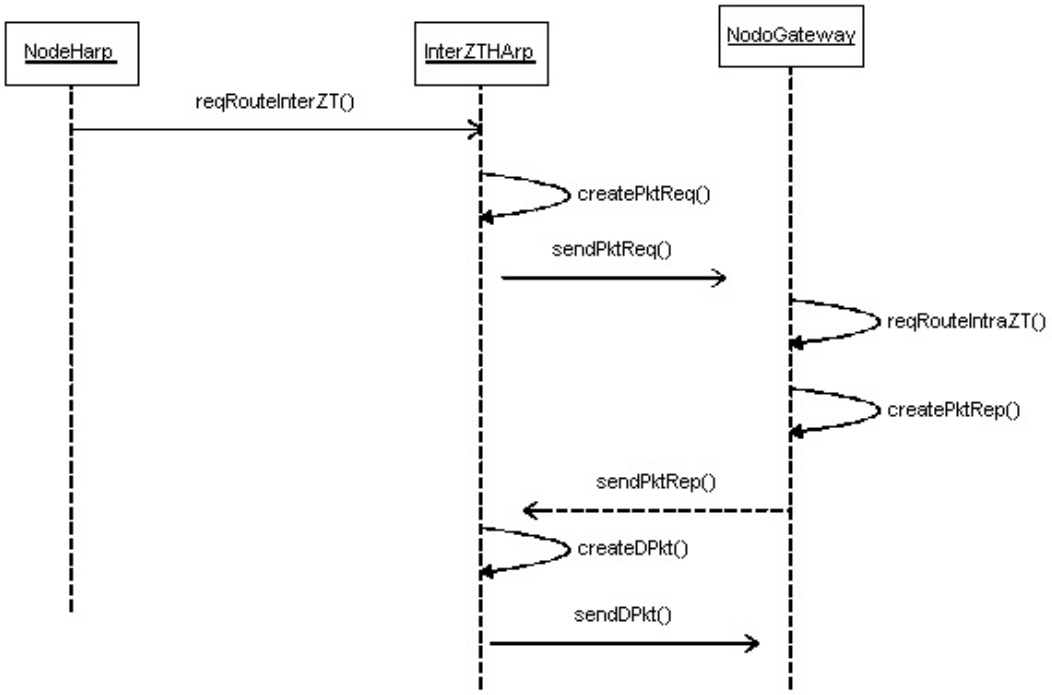



**Figura 5. 12 - Diagrama de seqüência do método pathDiscovery**

Tendo ocorrido o estabelecimento de um trajeto, o nodo fonte aplicará mecanismos que assegurem a conexão entre fonte e destino, enquanto haja a necessidade de troca de dados entre os mesmos. No HARP, esse mecanismo é denominado como manutenção do trajeto. Conforme ilustra a Figura 5.13, o método *pathMantenance()* executa um mecanismo, denominados por *pathRefresh* e *pathWaiting,* respectivamente. O primeiro

mecanismo, de atualização do trajeto, é responsável por oferecer um outro trajeto para um mesmo destino. O segundo mecanismo, de manutenção de rota, é responsável por manter a rota ativa durante o período de comunicação entre os nodos fonte e destino. Caso ocorra um erro de enlace ou expiração do tempo de espera, esse algoritmo define que um novo processo de descoberta de rota deve ser iniciado (NIKAEIN 2001).

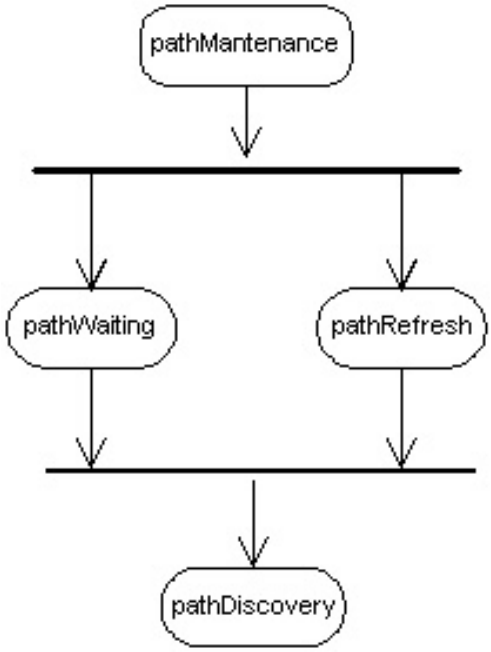



**Figura 5. 13 - Diagrama de seqüência do método pathMantenance**

Até aqui, buscou-se validar a possibilidade do desenvolvimento de um *framework* para o desenvolvimento de algoritmos de roteamento para redes móveis *ad-hoc* seguindo os passos sugeridos pela metodologia escolhida. Entende-se que a estrutura a qual denominamos como FRAd-hoc ainda agrega uma pequena quantidade de classes oferecendo por sua vez uma pequena quantidade de recursos a serem reusados quando comparado a *frameworks* orientados a objetos como por exemplo o *Hot Draw*. Mesmo diante dessa característica, encerramos aqui deixando a proposta do desenvolvimento inicial da estrutura de um *framework* caixa-branca que oferece suporte ao desenvolvimento de algoritmos de roteamento para redes móveis *ad-hoc,* conforme ilustra a Figura 5.14.

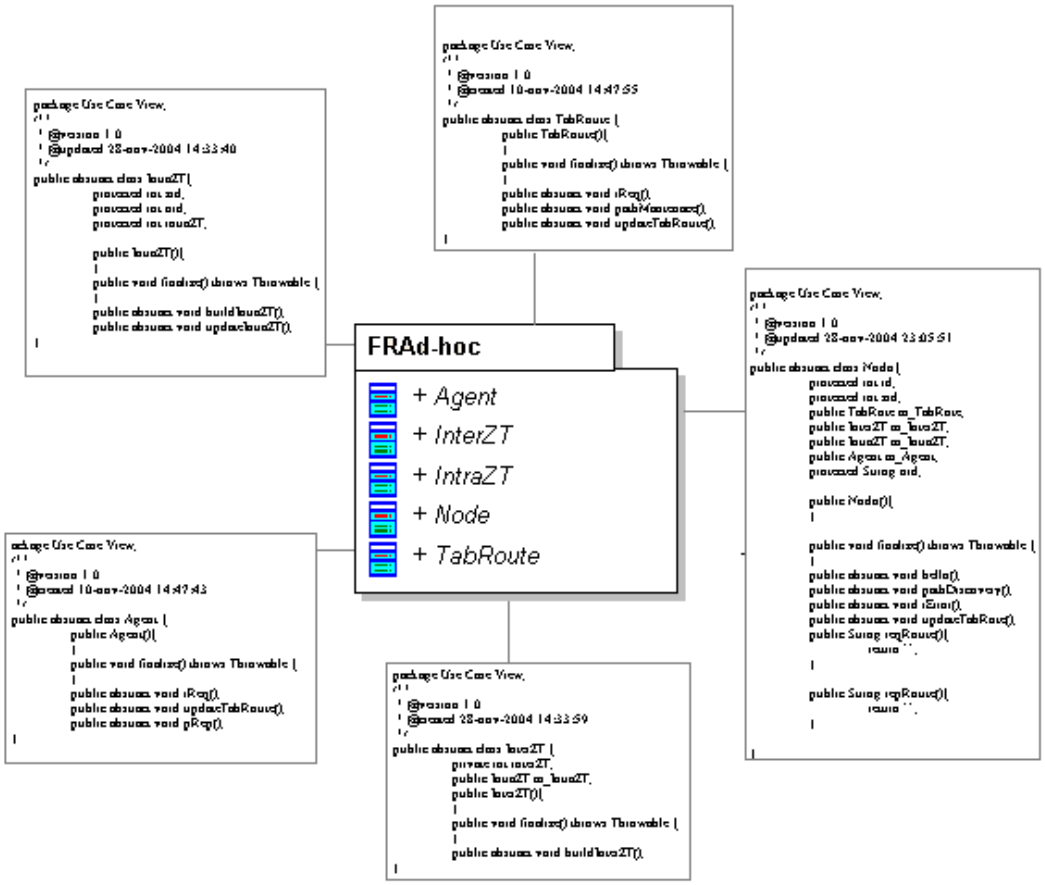



**Figura 5. 14 - *Framework* de roteamento *ad-hoc***

## 5.2 Conclusão

Este capítulo apresentou o desenvolvimento do protótipo da estrutura proposta, com o intuito de validar a idéia de se obter um *framework* para o roteamento em redes móveis *ad-hoc*. Conforme se pode perceber através dos resultados experimentais apresentados desta seção, muito embora sejam esses um tanto quanto simplificados devido às limitações impostas, acredita-se que a versão a qual se chegou incorpora características suficientes para indicar esse como um *framework* orientado a objetos, capaz de oferecer suporte ao desenvolvimento de algoritmos de roteamento híbrido para redes móveis *ad-hoc*.

# CAPÍTULO VI

## CONCLUSÃO E PROPOSTA PARA TRABALHOS FUTUROS

A tarefa de roteamento de mensagens da origem a um destino dado em qualquer rede de comunicação é uma tarefa fundamental ao bom andamento da transferência de informação. Mesmo em redes a infra-estrutura fixa, o roteamento é um processo que está longe de ser trivial.

Quando nos referimos a uma rede *ad hoc*, onde a topologia é modificada ao longo do tempo, devido ao deslocamento dos nodos, compostos de dispositivos computacionais móveis, o roteamento assume um papel ainda mais importante, sendo que o grau de complexidade da tarefa aumenta, devido a limitações impostas pelas características deste tipo de rede.

Como mostrado ao longo desta dissertação, as propostas de algoritmos de roteamento disponíveis na literatura são variadas, e classificadas em três grupos principais em termos da abordagem utilizada: os algoritmos pró-ativos, os algoritmos reativos e os híbridos, sendo que cada categoria ou, ainda, cada algoritmo particular apresenta suas vantagens e desvantagens, atendendo de forma mais acentuada uma ou outra limitação das redes *ad hoc*.

Apresentou-se aqui a proposta e o desenvolvimento de um *framework*, denominado FRAd-hoc, cujo objetivo foi gerar uma estrutura que oferecesse suporte à implementação dos algoritmos de roteamento híbridos para redes móveis *ad-hoc*.

A motivação para o desenvolvimento de tal elemento era a necessidade de se ter a possibilidade de manipular as diferentes alternativas em termos de protocolos de roteamento para redes *ad hoc*, aspecto de fundamental importância no desenvolvimento e evolução desta classe de redes.

Nossa preocupação no desenvolvimento desta pesquisa não foi o de esgotar o estudo do problema de roteamento, tampouco propor soluções inovadoras em termos de protocolo de roteamento, mas o de fazer uso de alguns dos protocolos publicados na literatura recente, de forma a possibilitar uma melhor adaptação da solução às características particulares do problema.

A proposta se mostrou inicialmente viável, pois os resultados do desenvolvimento indicam que, a partir desse *framework* de roteamento *ad-hoc*, pode-se

desenvolver outras aplicações de algoritmos de roteamento, pois o mesmo incorpora classes e métodos que generalizam o domínio pesquisado.

A grande vantagem do desenvolvimento de tal *framework* é poder contemplar as diversidades em termos de dinâmica de topologia das redes *ad hoc*, uma vez que o mesmo permitirá gerar implementações especializadas que mais se adequem às características da rede considerada.

Contudo, para dar continuidade a esse trabalho, outros algoritmos estarão sendo implementados sob a estrutura do *framework* proposto, a fim de melhor consolidar nossos objetivos.

Como proposta de pesquisa futura, acreditamos que uma ferramenta de gerência utilizando o *framework* proposto pode ser desenvolvida visando definir quais protocolos de roteamento estudados deverão executar durante os diferentes contextos de ambiente apresentado pelas redes móvel *ad hoc*. Além disso, através dos algoritmos implementados poder-se-á avaliar estatisticamente o perfil da rede ao qual se obtém a melhor performance do algoritmo executado.

## BIBLIOGRAFIA

ABOLHASAN, M., WYSOCKI, T., DUTKIEWICZ, Eryk., "A review of routing protocols for mobile ad hoc networks", Elservier Computer Science, www.elsevier.com/locate/adhoc, Aug, 2003.

ACEVES J.J., Garcia-Luna, SPOHN C., Marcelo, "Source-tree routing in wireless networks", in: Proceedings of the Seventh Annual International Conference on Network Protocols Toronto, Canada, October 1999, p. 273, 1999.

AGGELOU, G., TAFAZOLLI, R., (1999), "RDMAR: a bandwidth-e.cient routing protocol for mobile ad hoc networks", in: ACM International Workshop on Wireless Mobile Multimedia (WoWMoM), 1999, pp. 26–33, 1999.

BASAGNI S., I. CHLAMTAC, V.R. Syrotivk, WOODWARD, B.A., "A distance effect algorithm for mobility (DREAM)", in: Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom_98), Dallas, TX, (1998).

BELLUR, B., OGIER, R.G., TEMPLIN, F.L, "Topology broadcast based on reverse-path forwarding routing protocol (tbrpf)", in: Internet Draft, draft-ietf-manet-tbrpf-06.txt, work in progress, 2003.

BOSCH , J. , MOLIN, P., MATTSSON , M., BENGTSSON, PO, FAYAD, M., "Object-Oriented Frameworks: Problems & Experiences", in Building Application Frameworks, M. Fayad, D. Schmidt, R. Johnson (eds.), John Wiley, ISBN 0-471-24875-4, p.: 55-82, 1999.

CHAVES, C. V. F. e LUCENA, C. J., "Theory of Aspects for Aspect-oriented Software Development", The 17th Brazilian Symposium on Software Engineering, UFBA, PUC – RIO, Oct., 2003.

CISCO SYSTEMS, "IGRP – Interior Gateway Routing Protocol", Inc.Cisco Systems mid – 1980.

CORREA, U. C., MAZZOLA, V. B., DANTAS, M.A.R, "Analise Comparativa de Protocolos de Roteamento de Redes *Ad –Hoc",* Anais da 2a. Escola Regional de Redes de Computadores - ERRC, 149-155, 2004.

CORREA, U. C., MAZZOLA, V. B., DANTAS, M.A.R., "Analise de Algoritmos de Roteamento Wireless de Redes *Ad –Hoc",* Anais da 7a. Escola Regional de Informática - ERI, 100-110, 2004.

CORSON, S., MACKER, J., "Mobile Ad Hoc Networking (MANET)", IETF RFC 2501, http://www.ietf.org/rfc/rfc2501.txt, Jan. 1999.

CORSON, S., MACKER, J., "Mobile Ad Hoc Networking and IETF", ACM Mobile Computing and Communication Review, Vol. 2 and 3, Numbers 1,2,3,4, http://protean.itd.nrl.navy.mil/manet/manet_home.html, Oct., 1998.

CAMPBELL, A. T., CONTI M., GIORDANO S, "Mobile Ad Hoc Networks", Kluwer Academic Publishers. Manufactured in The Netherlands. Mobile Networks and Applications, p. 483–484, 2003.

CASTAÑEDA Robert, DAS R. Samir, MAHESH K. Marina, "Query localization techniques for on-demand routing protocols in ad hoc networks", ACM Computing Wireless Networks**,** Vol. 8 , Issue 2/3, p. : 137 – 151, 2002.

CHEN, T.-W., GERLA, M, "Global state routing: a new routing scheme for ad-hoc wireless networks", in: Proceedings of the IEEE ICC, 1998.

DAS, S., PERKINS, C. and ROYER, E., "Ad hoc on demand distance vector (AODV) routing", RFC 3561, http://www.ietf.org/rfc/rfc3561.txt, 2003.

FAYAD Mohamed, SCHMIDT, Douglas, "Object-Oriented Application Frameworks", Communications of ACM Vol. 40, No.10,1997.

GAMMA, Erich, HELM, Richard, JOHNSON, Ralph, e VLISSIDES John, "Design Patterns: Elements of Reusable Object-Oriented Software", Addison Wesley Professional,1994.

GANGOPADHYAY, D.; MITRA, S., "Understanding frameworks by exploration of exemplars", Computer-Aided Software Engineering, 1995, Seventh International Workshop, p.:90 – 99, 1995.

GERLA, M., "Fisheye state routing protocol (FSR) for ad hoc networks", Internet Draft, draft-ietf-manet-aodv-03.txt,work in progress, 2002.

GRAY, N.A.B**, "**Teaching object orientation: patterns and reuse"**,** Software Engineering Conference, 1996, p.:72 – 80, 1996.

GUANGYU*, Pei;* GERLA, M., Tsu-Wei Chen*,* "Fisheye state routing: a routing scheme for ad hoc wireless networks", Communications ICC. IEEE International Conference, Vol.1, Jun, p. 70-74, 2000.

GÜNES, M., SORGES, U., BOUAZIZI, I., "Ara—the ant-colony based routing algorithm for manets", in: ICPP workshop on Ad Hoc Networks (IWAHN 2002), August 2002, pp. 79–85, 2002.

HASS, Z.J., PEARLMAN, R., "Zone routing protocol for ad-hoc networks", Internet Draft, draft-ietf-manet-zrp-02.txt, work in progress, 1999.

HEDRICK, C., "RIP – Routing Information Protocol", IETF Working Group, RFC 1058, Rutgers University, June 1988, http://www.ietf.org/rfc/rfc1058.txt?number=1058, acessado em Jan., 2005.

IETF Working Group, "MANETs – Mobile Ad-hoc Networks", http://www.ietf.org/html.charters/manet-charter.html, acessado em Dez., 2004.

JACQUET, P., MUHLETHALER, P., CLAUSEN T., LAOUITI A., QAYYUM, A., VIENNOT, L., (2001), "Optimized link state routing protocol for ad hoc networks", IEEE INMIC, Pakistan, 2001.

JOA-NG, M. and I-TAI, L. "A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks", IEEE Journal on selected areas in communications, vol 17, nº 8, p. 1415-1425, 1999.

JOHNSON D., MALTZ D., JETCHEVA J., "The dynamic source routing protocol for mobile ad hoc networks", Internet Draft, draft-ietf-manet-dsr-07.txt, work in progress, 2002.

JOHNSON, Ralph E. & Foote, Brian, "Designing Reusable Classes". Journal of Object-Oriented Programming, Volume 1, Number 2, pages 22-30, 1988.

JOHNSON ,Ralfph, E., "Frameworks – Components and Patterns", Communications of the ACM, vol 40, Nº10, 1997.

KERAMANE C., "The Wireless World Web", Multimedia, IEEE , Volume: 7, Issue: 2, p. :10 – 14, 2000.

KIM, Dong-Kwan; YANG, Young-Jong; JUNG, Hyo-Taeg,, "Development of an object-oriented framework for intranet-based groupware systems", Systems, Man, and Cybernetics, IEEE International Conference, Vol.:3, p.:1982 – 1987,2001.

KLING, D., FLODSTRÖM, S., "Metapatterns – An overview", Malardalen University, Departament of Computer Science and Eletronics, http://www.idt.mdh.se/kurser/cd5130/msg/2002lp3/download/CD5130%20VT02%20Metapatterns.pdf, acessado em Nov. 2004.

KO, Y.-B., VAIDYA, N.H., "Location-aided routing (LAR) in mobile ad hoc networks", in: Proceedings of the Fourth Annual ACM/IEEE International

Conference on Mobile Computing and Networking (Mobicom_98), Dallas, TX, 1998.

KRAJNC, A., HERICKO, M., "Classification of Object-Oriented Frameworks." EUROCOM 2003.

LEWIS, Ted, "Object-Oriented Application Frameworks", ManningPUB, 1995.

MALKIN, G., "RIP Version 2 Protocol Analysis", IETF Working Group, RFC 1387, Jan. 1993, http://www.ietf.org/rfc/rfc1387.txt?number=1387, acessado em Jan. 2005.

MALKIN, G., "RIP Version 2 Carrying Additional Information", IETF Working Group, RFC 1388, Jan. 1993, http://www.ietf.org/rfc/rfc1388.txt?number=1388, acessado em Jan. 2005.

MALKIN, G., "RIP Version 2 MIB Extension", IETF Working Group, RFC 1389, Jan. 1993, http://www.ietf.org/rfc/rfc1388.txt?number=1388, acessado em Jan. 2005.

MARWAHA, S., THAM, C. K. and SRINIVASAN, D., "Mobile Agents based Routing Protocol for Mobile Ad Hoc Networks", Global Telecommunications Conference**.** GLOBECOM '02. IEEE, Vol. 1, p. 17-21, 2002.

MARWAHA, S., THAM, C. K. and SRINIVASAN, D., "A Novel Routing Protocol using Mobile Agents and Reactive Route Discovery for Ad Hoc Wireless Networks", Networks ICON, 10th IEEE International Conference, p. 311 – 316, 2002.

MICHAIL A, EPHREMIDES A, "Energy-efficient routing for connection-oriented traffic in wireless ad-hoc networks", ACM Computing Mobile Networks and Applications**,** Vol. 8 , Issue 5, p. : 517 – 533, 2003.

MIESO K. Denko, "The use of mobile agents for clustering in mobile ad-hoc networks", Proceedings of annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology, ACM International Conference Proceeding Series. p. 241 – 247, 2003.

MIGAS, N., BUCHANAn, W., J., WILLIAM, McARTNETY, K. A., "Mobile Agents for Routing, Topology Discovery, and Automatic Network Reconfiguration in AdHoc Networks", Enginnering of Computer-Based Systems, 10th IEEE International Conference and Workshop, p. 200-206, 2003.

MINAR, N., KRAMER, K.H., and MAES, P., "Cooperating mobile agents for dynamic Networking", Software Agents for Future Communications Systems, Chapter 12., Springer Verlag, 1999, http://llk.media.mit.edu/papers/archive/minar.pdf, acessado em Maio, 2004.

MOY, J., "OSPF Version 2", IETF Working Group, RFC 2178, Jul. 1997, http://www.ietf.org/rfc/rfc2178.txt?number=2178, acessado em Jan., 2005.

MURTHY, S. J.J. Garcia-Luna-Aceves, "A routing protocol for packet radio networks", in: Proceedings of the First Annual ACM International Conference on Mobile Computing and Networking, Berkeley, CA, 1995, pp. 86–95, 1995.

NIKAEIN, N. and BONNET, C., "HARP- Hybrid Ad Hoc Routing Protocol", In: IST2001 – International Symposium on Telecommunications http://www.eurecom.fr/~nikaeinn/harp.ps, 2001.

NIKAEIN, N., LABIOD, H. and BONNET, C., "DDR – Distributed dynamic routing algorithm for mobile ad hoc networks", Mobile and Ad Hoc IEEE Networking and Computing. MobiHOC 2000 First Annual Workshop, p19- 27, 2000.

NOBLE, J., "Classifying relationships between object-oriented design patterns", Software Engineering Conference, 1998. p.:98 – 107, 1998.

ONISHI, R., YAMAGUCHI, S., MORINO, H. and SAITO, T., "The Multi-agent System for Dinamic Network Routing", Autonomous Decentralized Systems, 5th IEEE International Symposium, p. 375-382, 2001.

PARK, V.D., CORSON, M.S., "A highly adaptive distributed routing algorithm for mobile wireless networks", in: Proceedings of INFOCOM, April, 1997.

PARSONS, D., RASHID, A., SPECK, A., Telea, A., "A framework for object oriented frameworks design", Technology of Object-Oriented Languages and Systems 1999, 7-10 p.:141 – 151, 1999.

PEI, G., M. Gerla, X. Hong, C. Chiang, "A wireless hierarchical routing protocol with group mobility", in: Proceedings of Wireless Communications and Networking, New Orleans, 1999.

PERKINS, C. E., "Ad Hoc Networking", 1st ed. United States of America, Addison-Wesley, p. 11- 27, 2001.

PERKINS, C.E. T.J. Watson, "Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers", in: ACM SIGCOMM_94 conference on Communications Architectures, London, UK, 1994.

RAJU J., J. Garcia-Luna-Aceves, "A new approach to ondemand loop-free multipath routing", in: Proceedings of the 8th Annual IEEE International Conference on Computer Communications and Networks (ICCCN), Boston, MA, October 1999, pp. 522–527, 1999.

RADHAKRISHNAN. S., RAO, N.S.V, RACHERLA, G., SEKHARAN, C.N., BATSELL, S.G., "DST—A routing protocol for ad hoc networks using distributed spanning trees", in: IEEE Wireless Communications and Networking Conference, New Orleans, 1999.

REKHTER, Y., "BGP – Border Gateway Protocol", IETF Working Group, RFC 1771, T.J. Watson Research Center, IBM Corp., Cisco Systems, July 1995, http://www.ietf.org/rfc/rfc1771.txt?number=1771, acessado em Jan., 2005.

ROYCLHOUDHURY, R., BANDYOPADHYAY, S., K., P., "A distributed mechanism for topology discovery in ad hoc wireless networks using mobile agents", Mobile and Ad Hoc Networking and Computing, MobiHOC IEEE 2000 First Annual Workshop, p. 145-146, (2000).

SILVA, Ricardo P. e, PRICE, R. T., "O uso de técnicas de modelagem no projeto de frameworks orientados a objetos", In: Proceedings of 26th International Conference of the Argentine Computer Science and Operational Research Society (26th JAIIO) / First Argentine Symposium on Object Orientation (ASOO'97), p.:87-94, 1997.

SILVA, Ricardo Pereira e, "Suporte ao desenvolvimento e uso de frameworks e componentes", Tese de Doutorado. Programa de Pós-Graduação em Computação, 2000. Universidade Federal do Rio Grande do Sul. www.inf.ufsc.br/~ricardo, acessado em Agosto, 2004.

SRINIVASAN, S., "Design patterns in object-oriented frameworks", Computer, Vol.:32, Issue: 2, p.:24 – 32 – JN jornal, 1999.

SVENSSON P. and C. Andersson., "Mobile Internet-An Industry-Wide Paradigm Shift", Ericsson Review No. 04, 1999.

TANENBAUN, Andrew S., "Computer Networks" – Third Edition, 2003.

TAIBI, T., LING, D.N.C, "Formal specification of design patterns: a comparison",Computer Systems and Applications, 2003. Book of Abstracts. ACS/IEEE International Conference, p.:77, 2003.

ZHANG L., Deering S., Estrin D., Shenker S., and Zappala D., "RSVP: A New Resource ReSerVation Protocol", In IEEE Networks Magazine, Vol. 31, No. 9, pp. 8-18, 1993.

WU, J. and STOJMENOVIC I., "Ad Hoc Networks", Computer, Volume: 37, Issue: 2, Feb.2004, p. 29-31, 2004.

# ANEXO I – RESPOSTA EMAILS TROCADOS COM AUTORES DOS ALGORITMOS EM ESTUDO

Dear Underléa,
I need some time to prepare the code for the public use, please let me know if your request is urgent?

Navid

>
> Mr. Nikaein,
>
> I´ve sent some emails for you about my work, but I'm not receive any
> answer. Please, if you don´t understand about my request, write to me
> that I´ll be more specific
>
> I´m student of Master Degree in Computer Science in University of
> Santa Catariana in Brazil ..I read your paper "HARP - Hybrid
> ad hoc routing protocol " and I´m interest in simulate them to compare
> with others routing protocols, so if is possible, I would like that
> you send to me the code of your algorithm "HARP - Hybrid ad
> hoc routing protocol" I´ll use the ns to simulate them.
>
> Thanks so much for your attention.
>
> Underléa Cabreira Corrêa
>
> Lab de Sistemas Distribuidos
> Depto Informática e Estatística
> Universidade Federal de Santa Catarina - UFSC
> Campus Universitário - Bairro Trindade
> CEP 88040-900 - Florianópolis - SC
> Tel.: (0xx48) 331.7558
>

Dear Underléa,
I didn't forget you, but I have got some unexpected stuff in the meantime and could not find some time to prepare the code as I promised to you. Now, I am working on it, and try my best to send you as soon as possible,

Sorry for the delay, Navid

>
>
> Hi Navid,
> Sorry if I was being improper, but I would like to get some return about the
> code that you were going to prepare for public use.
> Thanks so much att.
> Underléa
>

> Dear Underléa,
> I need some time to prepare the code for the public use, please let me
> know if your request is urgent?
>
> Navid
>
>
> Mr. Nikaein,
>
> I´ve sent some emails for you about my work, but I'm not receive any
> answer. Please, if you don´t understand about my request, write to me
> that I´ll be more specific
>I´m student of Master Degree in Computer Science in University of
> Santa Catariana in Brazil ..I read your paper "HARP - Hybrid
> ad hoc routing protocol " and I´m interest in simulate them to compare
> with others routing protocols, so if is possible, I would like that
> you send to me the code of your algorithm "HARP - Hybrid ad
>hoc routing protocol" I´ll use the ns to simulate them.
>
>
>
>>>> Thanks so much for your attention.
>>>>
>>>> Underléa Cabreira Corrêa
>>>>
>>>> Lab de Sistemas Distribuidos
>>>> Depto Informática e Estatística
>>>> Universidade Federal de Santa Catarina - UFSC
>>>> Campus Universitário - Bairro Trindade
>>>> CEP 88040-900 - Florianópolis - SC
>>>> Tel.: (0xx48) 331.7558
>>>>

Hi!
For that particular paper a simulation tool was not used, i simply used some heuristics and analytical studies.

Regards,
Mehran

On Tue, 22 Jun 2004 10:33 pm, you wrote:
> I´m student of Master Degree in Computer Science in University of
> Santa Catariana in Brazil ..I read your paper "A review of routing
> protocols for mobile ad hoc networks " and I´m interest to know what tool
> you used to compare the hybrid routing protocols? Did you use some
> simulator ? Thank so much.
> att.
>
> Underléa Cabreira Corrêa
>
> Lab de Sistemas Distribuidos
> Depto Informática e Estatística
> Universidade Federal de Santa Catarina - UFSC
> Campus Universitário - Bairro Trindade
> CEP 88040-900 - Florianópolis - SC

> Tel.: (0xx48) 331.7558

--
Mehran Abolhasan
B.E Computer, PhD
Smart Internet Technology CRC (SITCRC) Research Fellow
Telecommunications and IT Research Institute (TITR)
University of Wollongong
Northfields Avenue, NSW, 2522, Australia
Phone: +61 2 4221 5491
Fax: +61 2 4221 3277
Email 1: mehrana@uow.edu.au
Email 2: mehran@titr.uow.edu.au

Dear Underléa,

Sorry for not responding earlier but I am away from my office and only have
a limited access to email for next couple of weeks. We have been using GLOMOSIM
in our simulations. You can get more details from my coauthor, Mehran Abolhasan
email: Mehran Abolhasan <mehrana@uow.edu.au>

Best regards,

Tad Wysocki

Underléa Cabreira Corrêa wrote:

I´m student of Master Degree in Computer Science in University of
Santa Catariana in Brazil ..I read your paper "A review of routing protocols for mobile ad hoc networks " and I´m interest to know what tool you used to compare the hybrid routing protocols? Did you use some simulator ?
Thank so much.
att.

Underléa Cabreira Corrêa
Lab de Sistemas Distribuidos
Depto Informática e Estatística
Universidade Federal de Santa Catarina - UFSC
Campus Universitário - Bairro Trindade
CEP 88040-900 - Florianópolis - SC
Tel.: (0xx48) 331.7558

--
---

Associate Professor Tadeusz A Wysocki, MEngSc, PhD, DSc
School of Electrical, Computer, and Telecommunications Engineering
University of Wollongong
email: wysocki@uow.edu.au, tel: +61(0)2 42213413, fax: +61(0)2 42213236
http://www.elec.uow.edu.au/people/staff/wysocki/
---

Hola Underla Cabreira Corra,

Thank you for your letter and interest in the paper. I have set a web site
from which you may find more information regarding the simulation (with
the TCL code and simulation results.)

The http address is:
http://www.cs.bgu.ac.il/~schiller/elad/ns2/

My kind regards,
Elad Schiller

On Mon, 24 May 2004, [iso-8859-1] Underléa Cabreira Corrêa wrote:

>
>
>
> Hello,
>
> I´m student of Master Degree in Computer Sciencein University of Santa Catariana ..I read your paper "Random Walk for self-stabilizing group communication in ad-hoc networks" and I´m interest in simulate them to compare with others routing protocols, so if is possible, I would like that you send to me the code of your algorithm "Random Walk for self-stabilizing group communication in ad-hoc networks"I´ll use the ns to simulate them.
>
> Thanks so much for your attention.
>
> Underléa Cabreira Corrêa
>
> Lab de Sistemas Distribuidos
> Depto Informática e Estatística
> Universidade Federal de Santa Catarina - UFSC
> Campus Universitário - Bairro Trindade
> CEP 88040-900 - Florianópolis - SC
> Tel.: (0xx48) 331.7558
>

Hi Correa,

Glad to see you. Attached is the code and documents.

We appreciate your effort to try our theory on the ns.
It would be a great help for us if you send your code for the ns to me
in turn.

Enjoy and good work!

Ryokichi

p.s.

Now I graduated and joined a research institute of TOYOTA.
---
Ryokichi Onishi ( 大西 亮吉 )

TOYOTA InfoTechnology Center Co.,Ltd.
Strategic Planning GroupTel: +81-3-5561-8235 / Fax: +81-3-5561-8290
http://www.toyota-itc.com/