

**UNIVERSIDADE FEDERAL DE SANTA CATARINA  
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA  
CURSO DE CIÊNCIAS DA COMPUTAÇÃO**

**Efeito da Mobilidade em Métricas de Qualidade de  
Serviço para Redes Bluetooth**

**Michel Ribas Lobato  
Rafael Krüger Tavares**

Trabalho de conclusão de curso submetido à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de bacharel em Ciências da Computação

Florianópolis, 2003

# **Efeito da Mobilidade em Métricas de Qualidade de Serviço para Redes Bluetooth**

**Michel Ribas Lobato**  
**Rafael Krüger Tavares**

Este trabalho de Conclusão de Curso foi julgado adequado para obtenção de Graduação em Ciência da Computação e aprovado em sua forma final junto a Universidade Federal de Santa Catarina

---

Prof<sup>o</sup> Dr. Carlos Becker Westphall  
Orientador

Apresentada à Banca Examinadora, integrada pelos Professores:

---

Prof<sup>o</sup> Dr. Carlos Becker Westphall

---

Prof<sup>a</sup> Dr<sup>a</sup>. Carla Merckle Westphall

---

Prof<sup>o</sup> Dr. Mario Antônio Ribeiro Dantas

## **AGRADECIMENTOS**

Nossos sinceros agradecimentos aos nossos pais e familiares que, de uma forma ou de outra contribuíram para o surgimento deste trabalho. Ao nosso orientador, Professor Carlos Becker Westphall, pela oportunidade de sua orientação no desenvolvimento de nossas atividades. Aos colegas e amigos do LRG – Laboratório de Redes e Gerência da UFSC Marcos, Breno, Mateus, Tadeu, Jean, Valério, Kleber, Rafael, Parra e Wesley, por todo apoio a nós dispensado, e a Professora Clarice Fortkamp Caldin, que nos ajudou na metodologia científica do trabalho. A todos o nosso MUITO OBRIGADO!

## RESUMO

O desenvolvimento da tecnologia da informação cresce cada vez mais nos dias de hoje. O advento das redes de computadores sem fio já é uma realidade. A crescente demanda do mercado por dispositivos sem fio como laptops, telefones celulares e *handhelds*, faz com que a busca por garantia da qualidade de serviço(QoS) se torne o foco de grandes centros de pesquisa mundiais. O esquema de filas FIFO não possui garantia de QoS. Este trabalho analisa parâmetros de QoS para redes *Bluetooth* em esquemas de fila alternativos à FIFO; como o SFQ, o FQ e o DRR, relacionando o desempenho dos parâmetros de QoS com a mobilidade dos nós na rede.

Palavras-chave: Qualidade de Serviço, *Bluetooth*, Simulações

## **ABSTRACT**

The development of information technology increases everyday. The wireless networks technologies became real. The crescent market demand for wireless devices such as laptops, cell phones and handhelds, makes the search for quality of service be the focus of research to great research centers over the world. This work presents a QoS analysis of Bluetooth networks through queue schemas alternative to FIFO, such as SFQ, FQ and DRR, comparing the performance of QoS parameters with the mobility of the network nodes.

Keywords: Quality of service, Bluetooth, Simulations

## LISTA DE FIGURAS

Figura 1 - Diferenças entre redes sem fio e computação móvel.....	19
Figura 2 - O espectro eletromagnético.....	22
Figura 3 - Rede sem fio infra-estruturada.....	25
Figura 4 - Rede sem fio <i>ad hoc</i> .....	26
Figura 5 - Topologia sem fio <i>ad hoc</i> .....	26
Figura 6 - Comunicação entre os nós A e D em uma rede <i>ad hoc</i> .....	28
Figura 7 - Padrão IEEE 802.11.....	30
Figura 8 - Transmissão por FHSS.....	32
Figura 9 - Quadro comparativo entre as tecnologias FHSS e DSSS.....	33
Figura 10 - Características técnicas das tecnologias FHSS e DSSS.....	34
Figura 11 - Problema da estação oculta.....	39
Figura 12 - Problema da estação exposta.....	40
Figura 13 - Dimensões do microchip <i>bluetooth</i> .....	42
Figura 14 - Sincronização automática.....	43
Figura 15 - Conferência interativa.....	44
Figura 16 - <i>Gateway</i> para a internet.....	44
Figura 17 - Dispositivos multifuncionais.....	45
Figura 18 - Maior mobilidade.....	45
Figura 19 - Comparação ente as arquiteturas OSI e <i>Bluetooth</i> .....	49
Figura 20 - Pilha de protocolos <i>Bluetooth</i> .....	49
Figura 21 - Arquivo trace importado.....	54
Figura 22 - Arquitetura da <i>scatternet</i> .....	56

## **LISTA DE TABELAS**

Tabela 1 - Resultados da simulação FIFO. Limite: 30 pacotes .....	60
Tabela 2 - Resultados da simulação SFQ. Limite: 30 pacotes .....	61
Tabela 3 - Resultados da simulação FQ. Limite: 50 pacotes.....	62
Tabela 4 - Resultados da simulação DRR. Limite: 30 pacotes .....	63

## LISTA DE SIGLAS

ACL - *Asynchronous Connection Less*  
AP - *Access Point*  
ATM - *Assynchronous Transfer Mode*  
BSA - *Basic Service Area*  
BSS - *Basic Service Set*  
CTS - *Clear to Send*  
CSMA/CA - *Carrier Sense Medium Access with Colision Avoidance*  
DRR - *Deficit Round-Robin*  
DSSS - *Direct Sequence Spread Spectrum*  
EM - *Estações Móveis*  
ESA - *Extended Service Area*  
ESS - *Extended Service Set*  
FCC - *Federal Communications Commission*  
FIFO - *First In First Out*  
FHSS - *Frequency Hopping Spread Spectrum*  
FOMA - *Freedom of Mobile Multimedia Access*  
GPS - *Global Positioning System*  
HF - *High Frequency*  
HCI - *Host Controler Interface*  
IBSS - *Independent Basic Service Set*  
ICS - *Information Sciences Institute*  
IEEE - *Institute of Electrical and Eletronics Engineers*  
IR - *Infrared*  
ISDN - *Integrated Services Digital Network*  
ISM - *Industrial Scientific and Medical*  
ITU - *International Telecommunication Union*  
L2CAP - *Logical Link Control and Adaptation Protocol*  
LAN - *Local Area Network*



LLC - *Logic Link Control*  
LC - *Link Control*  
LMP - *Link Manager Protocol*  
MAC - *Medium Access Control*  
MACA - *Multiple Access with Collision Avoidance*  
MANET - *Mobile Ad hoc Network*  
MH - *Mobile Hosts*  
NAM - *Network Animator*  
NS-2 - *Network Simulator v.2*  
QoS - *Quality of Service*  
Otc1 - *Object Tool Command Language*  
PA - *Ponto de Acesso*  
PDA - *Personal Digital Assistant*  
PHY - *Physic Layer*  
PSTN - *Public Switched Telephone Network*  
RED - *Random Early Detection*  
RSSF - *Redes de Sensores Sem Fio*  
RSVP - *Resource Reservation Protocol*  
RTS - *Request to Send*  
SCO - *Synchronous Connection Oriented*  
SIG - *The Bluetooth Special Interest Group*  
SFQ - *Stochastic Fair Queuing*  
UA - *User Asynchronous*  
UI - *User Isosynchronous*  
US - *User Synchronous*  
VINT - *Virtual Internetwork Testbed*  
WAN - *Wide Area Network*  
WAP - *Wireless Application Protocol*  
WLAN - *Wireless Local Area Network*  
WRED - *Weighted Random Early Detection*  
WWAN - *Wireless Wide Area Network*

## SUMÁRIO

1 INTRODUÇÃO .....	11
1.1 Objetivos .....	15
1.2 Justificativa .....	15
1.3 Trabalhos Correlatos .....	16
1.4 Organização do Trabalho .....	17
2 REDES SEM FIO ( <i>WIRELESS NETWORKS</i> ) .....	19
2.1 Introdução .....	19
2.2 Técnicas de Transmissão de Dados: A Transmissão sem Fio .....	21
2.3 O Padrão IEEE 802.11 .....	23
2.3.1 Topologias .....	24
2.3.2 A Arquitetura do Padrão IEEE 802.11 .....	29
2.4 Qualidade de Serviço (QoS).....	34
2.5 Problemas Relacionados às Redes Sem Fio .....	38
2.5.1 Handoff.....	38
2.5.2 Problema da Estação Oculta e Problema da Estação Exposta.....	38
3 A TECNOLOGIA DE INTEGRAÇÃO <i>BLUETOOTH</i> .....	41
3.1 Introdução .....	41
3.2 Breve Histórico.....	42
3.3 Principais Usos .....	43
3.4 Características Técnicas do <i>Bluetooth</i> .....	45
3.4.1 Tipos de rede <i>Bluetooth</i> .....	45
3.4.2 Canais de comunicação .....	46
3.4.3 O Pacote de Dados <i>Bluetooth</i> .....	47
3.5 A Arquitetura da Tecnologia <i>Bluetooth</i> .....	48
3.5.1 <i>Bluetooth</i> Radio .....	50
3.5.2 Baseband.....	50
3.5.3 Link Manager Protocol ( <i>LMP</i> ) .....	51
3.5.4 Host Controller Interface ( <i>HCI</i> ) .....	51
3.5.5 Logical Link Control and Adaptation Protocol ( <i>L2CAP</i> ).....	52
3.5.6 <i>RFCOMM</i> .....	52
4 AMBIENTE DE DESENVOLVIMENTO.....	53
4.1 Sobre o NS.....	53
4.2 Modelo de Simulação .....	55
5 ANÁLISE DOS RESULTADOS .....	58
5.1 Perda de Pacotes .....	58
5.1.1 First in First out ( <i>FIFO</i> ).....	59
5.1.2 Stochastic Fair Queueing ( <i>SFQ</i> ) .....	60
5.1.3 Fair Queueing ( <i>FQ</i> ).....	61
5.1.4 Deficit Round Robin ( <i>DRR</i> ) .....	62
5.2 Latência .....	63
5.3 Jitter.....	65
6 CONCLUSÕES E TRABALHOS FUTUROS .....	67
REFERÊNCIAS .....	70
ANEXOS .....	72

# 1 INTRODUÇÃO

Os meios de comunicação se desenvolvem rapidamente, impulsionados pela necessidade constante do ser humano de se comunicar. Inovações como o grande poder de processamento, alta capacidade de armazenamento e outras novas tecnologias acabam contribuindo para a popularização de computadores e incentivam a criação de inúmeras redes de computadores.

A evolução é tão grande, que já passou do uso do antigo e nada prático de aparelhos como o telégrafo para os modernos dispositivos móveis e compactos sem fio.

Devido ao atual crescimento do segmento de computadores pessoais portáteis, estima-se que em 2005, o mercado para dispositivos portáteis sem fio crescerá cerca de 630% (BEAL, 2003). Portanto, será bastante comum as pessoas possuírem algum tipo de dispositivo portátil como celulares, *palmtops*, *notebooks* e PDAs com capacidade de se comunicar com a parte fixa da rede e até com outros computadores móveis.

Estes tipos de ambientes onde os usuários móveis podem realizar comunicações sem fio para acessar recursos distribuídos faz parte da linha de pesquisa da tecnologia de Redes Móveis sem fio ou *Wireless Networks*.

As redes sem fio/móveis, surgem como a quarta revolução na computação, antecedida pelos centros de processamento de dados da década de sessenta, o surgimento dos terminais nos anos setenta e as redes de computadores na década dos oitenta (LIMA, 2002).

O aparecimento deste tipo de rede propiciou uma série de facilidades no dia a dia das pessoas. O telefone celular, por exemplo, permite a comunicação interpessoal em qualquer lugar dentro da área de cobertura que um usuário esteja, podendo o mesmo através deste marcar reuniões, encontros, e até mesmo resolver problemas da empresa onde trabalha.

Com o surgimento de tecnologias cada vez mais inovadoras como o WAP, todas as facilidades da Internet foram trazidas a estes pequenos aparelhos, ou seja, andando em um lobby de hotel, ou em um shopping center, no parque da cidade, ou em qualquer outro lugar público já não é mais difícil ver as pessoas usando celulares. Pesquisas confirmam o que as pessoas já sabem por experiência própria: o uso de telefones celulares cresce no mundo inteiro e já se tornou uma coisa típica na vida das pessoas (BEAL, 2003).

Em países mais avançados como o Japão já existe, desde 2001, uma tecnologia que permite o envio de tráfego multimídia através de telefones celulares e terminais conhecida como FOMA (*Freedom of Mobile Multimedia Access*). Esta tecnologia permite o tráfego de pacotes com velocidades de 64 Kbps para envio e 384 Kbps para recepção (FOMA P2402, 2003).

As redes sem fio são evidentes no cotidiano da sociedade mais do que se pode imaginar. As emissoras de televisão se utilizam bastante esta tecnologia em suas transmissões via satélite. Navegadores, pilotos e aventureiros já se orientam de forma global através do uso de dispositivos GPS (Global Positioning System). Países como o Canadá fazem largo uso da comunicação através da transmissão por microondas, a Indonésia possui seu próprio satélite para o tráfego telefônico doméstico.

Todos esses serviços tiveram um fator muito importante para sua popularização: o apoio comercial, visto que os ambientes locais sem fio (*wireless*) são configurações interessantes para agregar valor às redes locais das corporações (DANTAS, 2002). As grandes empresas de telecomunicações perceberam o grande potencial de mercado que a comunicação sem fio poderia trazer e começaram a investir nessa nova tecnologia.

Os esforços foram tão intensos a IEEE (*Institute of Electrical and Eletronics Engineers*) constituiu um grupo de pesquisa para criar padrões abertos que pudessem tornar a tecnologia sem fio cada vez mais real (PEREIRA, 2003), assim, em 1990 surgiu o grupo de trabalho IEEE 802.11 que tinha como objetivo principal a elaboração de um padrão para a comunicação sem fio entre estações de uma área local.

Devido a alguns fatores como a baixa taxa de transferência de dados o IEEE 802.11 ficou parado por cerca de sete anos e, em meados de 1997, foi publicado o Padrão IEEE 802.11, que tinha como características a definição das regras relativas a subcamada de Controle de Acesso ao Meio (MAC) e camada física (PHY) para redes LAN sem fio, e definição de topologias para as mesmas.

Ao longo dos anos, os avanços tecnológicos correspondentes a miniaturização de componentes e a elevação das taxas de transmissão das redes sem fio propiciaram a explosão das mesmas. Atraídas por esta explosão tecnológica as empresas investiram ainda mais em tecnologia e em aparelhos que dessem suporte a estas tecnologias.

As grandes corporações começaram a fazer uso das redes sem fio em seus ambientes de trabalho, devido a sua principal característica: a mobilidade. Assim um executivo de uma

determinada empresa, ao utilizar o seu *notebook*, não precisaria mais ficar atrelado a estrutura física da rede, bastando apenas inserir uma interface de rede sem fio em seu *notebook*, para ter conectividade com a LAN da empresa, em qualquer lugar dentro da mesma.

Com relação a área científica, a maioria das pesquisas em redes de computadores sem fio se concentram na parte das redes locais de computadores (WLAN – *Wireless Local Area Network*) como as redes corporativas citadas anteriormente. Além disso, muitas tecnologias sem fio proprietárias têm sido usadas para possibilitar a comunicação entre dispositivos sem fio (PEREIRA, 2003).

Tais tecnologias procuram acabar com o problema da conexão física (através de cabos) tanto em redes de computadores quanto em outros dispositivos como periféricos de computador (mouse, teclado, impressoras, etc.), dispositivos móveis (telefones celulares, *notebooks*, *palmtops*, PDAs, etc.) entre outros. A tecnologia *Bluetooth* e a *HomeRF* são grandes exemplos destas soluções tecnológicas.

O *Bluetooth* está incluído no conjunto de soluções tecnológicas que visam não apenas a velocidade e a qualidade de transmissão dos dados, mas também a comodidade do usuário. A idéia do *Bluetooth* surgiu em 1994, quando a Ericsson, grande empresa de telecomunicações, iniciou um projeto procurando eliminar todo o incômodo causado pela quantidade de cabos e fios existentes em praticamente todas as instalações de computadores. Alguns inconvenientes com a utilização de cabos podem ser relacionados (HORSTMANN, 2002):

- O emaranhado de cabos;
- Vários padrões de cabos e conectores;
- Conexões não confiáveis;
- Necessidade de manter cabos e conectores para reposição;
- Dificuldade na movimentação de unidades computadorizadas para diferentes locais devido à restrição dos tamanhos dos cabos;
- Necessidade de intervenção manual quando o número de portas físicas não é suficiente para novas conexões;
- Necessidade de reconfigurar as unidades no sistema operacional quando há movimentação das unidades.

Hoje dia, o *Bluetooth* conta com o apoio de mais de 2500 associados que fazem parte do Grupo de Interesse Especial *Bluetooth*, ou SIG (*The Bluetooth Special Interest Group*).

Como principais características, o *Bluetooth* apresenta:

- Alta conectividade;
- Mobilidade;
- Comunicação através da criação de uma rede sem fio;
- Tamanho reduzido (toda a tecnologia é implementada em um chip);
- Transferência de voz e dados;
- Capacidade de interconexão entre diferentes tipos de dispositivos com baixa potência e baixo custo.

Todas essas características fazem do *Bluetooth* uma tecnologia bastante promissora e passível de uso em larga escala.

Apesar de todas as suas vantagens, as redes sem fio, assim como toda a tecnologia em ascensão deve fornecer a seus usuários uma comunicação rápida, eficaz e segura. Para que essas características sejam asseguradas certas regras devem ser obedecidas, ou seja deve existir a garantia da Qualidade de Serviço (QoS – *Quality of Service*) nas redes sem fio.

Qualidade de serviço nada mais é do que a garantia de certos parâmetros que as aplicações exigem como por exemplo largura de banda suficiente para tráfego multimídia. A qualidade de serviço pode ser medida através de várias métricas como vazão, atraso, *jitter* e taxa de perda de pacotes. Um dos grandes desafios da atualidade esta em conseguir esta garantia da qualidade de serviço, nas redes sem fio como um todo, tanto nas LAN'S sem fio como na tecnologia *Bluetooth*.

## 1.1 Objetivos

Já existe uma grande aceitação da tecnologia *Bluetooth* no mercado, refletida tanto em clientes tradicionais, quanto àqueles mais exigentes. Por ser uma tecnologia nova ainda há muito o que descobrir sobre a mesma. Sendo assim, o objetivo geral deste trabalho é analisar, através de simulações, o comportamento de uma rede *Bluetooth* mediante certas métricas de Qualidade de Serviço, levando em consideração o impacto que a mobilidade causa nestas métricas.

Para isso serão necessários os seguintes objetivos específicos:

- Estudar as principais características das redes de computadores sem fio, incluindo suas técnicas de transmissão;
- Avaliar alguns aspectos da tecnologia *Bluetooth*;
- Criar um cenário *Bluetooth* com uma certa quantidade de nós para fazer a simulação do mesmo;
- Utilizar, estudar e compreender o funcionamento de uma ferramenta de simulação, mais especificamente a ferramenta *Network Simulator (NS-2)*;
- Avaliar os resultados da simulação se atendo as métricas de vazão, atraso ou latência, *jitter* e perda de pacotes;
- Considerar o impacto causado pela mobilidade no cenário de simulação e realizar observações sobre este processo;
- Ajudar através da contribuição do presente trabalho nas pesquisas referentes a área das redes de computadores sem fio;

## 1.2 Justificativa

Atualmente, na chamada “era da informação”, estão surgindo usuários que têm a necessidade de estar constantemente online. Para estes usuários, a infra-estrutura fixa de rede, além de deixar os usuários dependentes dela, não está mais suprimindo esta crescente necessidade de acesso constante a rede, pois em situações como por exemplo, uma viagem de avião, não existe tal infra-estrutura. As redes sem fio surgiram na tentativa amenizar este problema.

Outra grande exigência dos usuários, é a garantia de qualidade na conexão, ou seja um usuário que está fazendo um download ou checando seus *emails*, não pode perder a conexão com a sua rede no meio destes processos.

Com o intuito de satisfazer ainda mais seus usuários, as grandes empresas de telecomunicações investem pesado nas pesquisas em Qualidade de Serviço, além de concorrerem umas com as outras para conseguir cada vez mais clientes, através do oferecimento de uma gama variada de serviços. O *Bluetooth*, por exemplo já oferece integração entre os mais variados tipos de dispositivos.

Como toda nova tecnologia, as redes *Bluetooth*, apesar de serem uma tecnologia muito promissora, ainda tem muitos aspectos a melhorar, sendo assim, as pesquisas com redes sem fio, onde está inserido o *Bluetooth*, procuram desenvolver soluções para superar desafios como mobilidade, inconstância de rotas e Qualidade de Serviço (HORSTMANN, 2002).

Devido a importância do escopo das redes de computadores sem fio na comunidade acadêmica, presenciada nas diversas publicações em artigos e revistas desta área, escolheram-se as mesmas representadas através da tecnologia *Bluetooth*, como tema deste trabalho.

### **1.3 Trabalhos Correlatos**

Diversos trabalhos científicos foram publicados no tema de redes de computadores sem fio e Qualidade de Serviço (QoS). Entre os trabalhos pesquisados destacam-se os mais relevantes:

- *Proposta e Validação de um Mecanismo para garantir QoS em Redes Sem Fio de Topologia Ad Hoc* (LIMA, 2002): Descreve um mecanismo de garantia de Qualidade de Serviço para redes sem fio do tipo *ad hoc* através do gerenciamento eficiente da largura de banda;
- *Análise de Desempenho em Redes Wireless Ad Hoc e Estabelecimento de um Acordo de Nível de Serviço Pró-Ativo* (PEREIRA, 2003): Apresenta um estudo detalhado das principais métricas de Qualidade de Serviço em redes sem fio do tipo *ad hoc*, através de simulações. Neste trabalho é montado um SLA (*Service*



*Level Agreement*) para redes sem fio *ad hoc* baseado na análise das métricas de Qualidade de Serviço;

- *Avaliação de Mecanismos para Gerenciamento da Fila da Interface do Host Controller Bluetooth* (HORSTMANN, 2002): Simula diferentes tipos de fila suportados pelo *host controller Bluetooth*, com o objetivo de agregar Qualidade de Serviço em dispositivos que utilizam essa tecnologia;
- *Avaliação de Desempenho em Redes Móveis sem Fio Ad hoc* (BELLÉ, 2003): Trata da comparação entre o cenário de análise da rede *Bluetooth* de (HORSTMANN, 2002) e o cenário de análise de uma rede sem fio *ad hoc* comum através de vários gráficos e tabelas. Também é avaliado o desempenho de uma rede *ad hoc* em seu âmbito geral;
- *A Method for Self-optimisation in Tree-pattern Bluetooth Scatternet* (LUO, 2003): Propõe um meio de otimização de uma *scatternet Bluetooth* através da reorganização dos seus nodos. A proposta é validada através de simulações;
- *Qualidade de Serviço em Redes IP com Diffserv: Avaliação Através de Medições* (MELO, 2001): Avalia a Qualidade de Serviço para aplicações de voz e multimídia nos mais diferentes ambientes, utilizando para isso a arquitetura de Serviços Diferenciados (*DiffServ - Differentiated Services*).

Esses e outros trabalhos foram fontes de pesquisa fundamentais para o desenvolvimento do presente trabalho.

#### **1.4 Organização do Trabalho**

A fim de propiciar uma disposição mais organizada dos assuntos apresentados, o trabalho foi dividido em sete partes, sendo elas as seguintes:

O primeiro capítulo procurou introduzir o assunto das redes de computadores sem fio com ênfase na tecnologia *Bluetooth*, através de um breve comentário sobre estas redes.

Em seguida, no segundo capítulo, será mostrado mais a fundo o paradigma das redes de computadores sem fio. Será dada ênfase nos conceitos da topologia destas redes e suas principais características. A estrutura do protocolo sem fio e a técnica de transmissão de dados por radiofrequência também serão abordados, assim como alguns conceitos da qualidade de serviço.

A tecnologia *Bluetooth* será objeto de estudo do terceiro capítulo. É feita menção a todos os aspectos da tecnologia, suas configurações como a *piconet* e a *scatternet*. Para introduzir o assunto um pouco da história da tecnologia é discutida.

O quarto capítulo trata da ferramenta utilizada para fazer as simulações do trabalho, o *Network Simulator* (NS-2). Esta ferramenta é descrita em detalhes, desde o seu funcionamento até a forma de interpretação dos resultados.

Os testes realizados junto com os seus respectivos resultados são assunto do quinto capítulo. Nesta parte, são apresentados todos os gráficos das simulações, e feitos comentários sobre estes gráficos. Os scripts de configuração do cenário *Bluetooth* simulado aparecem para facilitar a compreensão das simulações como um todo.

Por fim, no capítulo 6 são feitas as conclusões do trabalho, sugestões de trabalhos futuros, algumas considerações finais e descritas as dificuldades encontradas na realização do mesmo.

## 2 REDES SEM FIO (*WIRELESS NETWORKS*)

Este capítulo irá abordar os conceitos mais importantes das redes de computadores sem fio, desde o seu início até os principais requisitos atuais exigidos para o bom funcionamento deste tipo de rede.

### 2.1 Introdução

Antes do aparecimento das redes sem fio existia uma certa “dependência” dos usuários das redes de computadores à infra-estrutura da rede. Mesmo com a existência dos computadores móveis como *notebooks e laptops*, para se ter acesso a rede era necessário estar fisicamente ligado por um fio a alguma infra-estrutura local, seja na tomada telefônica de casa ou conectado a LAN do escritório.

Muitos usuários de computadores móveis, embora tenham o suporte da infra-estrutura de rede local, precisam se conectar aos dados que mantêm em casa mesmo à distância. Como é impossível fazer uma conexão por fio a partir de carros e aeronaves, começaram a existir inúmeras redes sem fio muito interessantes (TANEMBAUM, 1997).

Neste contexto, é importante fazer a distinção entre as redes sem fio e a computação móvel. Embora sejam parecidas, existem entre estes conceitos algumas diferenças como, por exemplo, o fato da existência de um *palmtop*, que embora possa ser levado para qualquer lugar, sugerindo a idéia de mobilidade, precisa de uma conexão com fio para a transferência de dados e acesso a Internet. A figura 1 mostra as principais diferenças entre as redes sem fio e a computação móvel.

Sem Fio	Móvel	Aplicações
Não	Não	Estações de trabalho fixas em escritórios
Não	Sim	Utilização de um portátil em um Hotel; manutenção de trem
Sim	Não	LANS em prédios mais antigos, sem fiação
Sim	Sim	Escritório portátil; PDA para estoque de loja

**Figura1** Diferenças entre redes sem fio e computação móvel (TANEMBAUM, 1997)

Um das principais vantagens das redes sem fio é a sua modularidade, ou seja, a capacidade de ser instalada em qualquer lugar, mesmo em ambientes mais desfavoráveis como, por exemplo, uma ilha.

A idéia da tecnologia de comunicação sem fio surgiu nas ilhas havaianas com o sistema ALOHA em meados da década de 70. A comunicação foi realizada através da instalação, em cada estação, de um pequeno transmissor/receptor de rádio FM, com um alcance suficiente para se comunicar com o transmissor/receptor central. Foram usadas duas faixas de frequência e a transmissão foi feita a 9600 bps (LIMA, 2002).

No princípio, problemas como a baixa taxa de transmissão e a pequena largura de banda não propiciaram a popularização da tecnologia, porém, a medida em que evoluía a miniaturização dos componentes e as comunicações pessoais sem fio como o telefone celular, as redes sem fio passaram a surgir como uma boa alternativa a mobilidade, e as primeiras redes sem fio comerciais começaram a aparecer no início dos anos 90.

Nas redes cabeadas, existem as redes LAN (*Local Area Network*) e WAN (*Wide Area Network*). Esse conceito também foi estendido para as redes de computadores sem fio. As redes WLAN (*Wireless Local Area Network*), são aquelas redes destinadas a atuar em uma área não muito abrangente, interligando diferentes dispositivos e periféricos.

As redes WWAN (*Wireless Wide Area Network*) possuem uma abrangência muito maior e têm suporte na telefonia, inicialmente desenvolvida para a comunicação de voz e depois adaptadas para transmissão de dados. Baseia-se, fundamentalmente, no sistema telefônico celular (LIMA, 2002).

Entre as diversas aplicações das redes sem fio estão as redes de sensores sem fio ou RSSFs (LOUREIRO, 2001). Esta tecnologia consiste de um tipo especial de rede sem fio *ad hoc* composta de vários tipos de nodos que são equipados com uma variedade de sensores tais como acústico, sísmico, infravermelho, vídeo câmera, calor, temperatura, pressão, etc.

Ao contrário de uma rede convencional, onde um computador não pode se comunicar com outro computador que está ao seu lado sem passar pela estrutura da rede, os sensores sem fio se comunicam diretamente uns com os outros, além de permitirem a execução de várias aplicações como clusters e computação distribuída. Entre as principais áreas de aplicação das RSSFs estão o controle, ambiental, tráfego, segurança, medicina/biologia e militar.

Essa infinidade de aplicações das redes sem fio demonstra a grande popularização desta tecnologia. Para atender a diferentes necessidades, novas tecnologias como a de redes de sensores sem fio tem surgido como por exemplo o *Bluetooth*, *HomeRF*, *WAP*, *HiperLan* e *WLAN* (DIAS, 2001). Cada uma dessas tecnologias tem suas particularidades e características como taxa de transmissão, modo de transmissão, alcance de transmissão e largura de banda. Pela grande aceitação de mercado e pela larga quantidade de trabalhos científicos publicados, decidiu-se dar ênfase a tecnologia *Bluetooth*.

## 2.2 Técnicas de Transmissão de Dados: A Transmissão sem Fio

A principal característica de uma rede sem fio, e que proporciona a maioria das aplicações é o modo como as mesmas transmitem seus dados. A técnica de transmissão de dados sem fio é utilizada há muito tempo, desde a época na qual foi inventado o telégrafo, considerado por muitos como o princípio da área de comunicação de dados.

Todo esse avanço nas telecomunicações sem fio não seria possível sem a existência do fenômeno físico das ondas. As ondas são originadas pelos elétrons, que têm a característica de se comportarem ora como partícula ora como onda. Quando se movem, os elétrons produzem ondas eletromagnéticas que podem se propagar no espaço livre e no vácuo.

As ondas possuem três importantes características:

- **Frequência:** Definida como o número de oscilações por segundo. É medida em Hertz (Hz) e representada pela letra  $f$ .
- **Amplitude ou crista da onda:** É a medida da altura da onda para voltagem positiva ou para voltagem negativa.
- **Comprimento de onda:** É a distância entre dois pontos máximos ou mínimos consecutivos de uma onda. É representada pela letra grega  $\lambda$  (lambda).

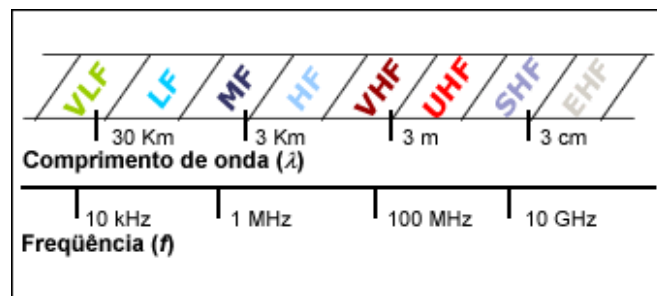
Qualquer tipo de comunicação sem fio é baseado no fato de que instalando-se uma antena com uma tamanho apropriado em um circuito elétrico pode-se transmitir e receber uma onda eletromagnética com eficiência por um receptor localizado a uma distância bastante razoável.

Para se ter uma noção da dimensão de uma onda, basta saber que no vácuo ela se propaga na velocidade da luz,  $c$ , que é de cerca de  $3 \times 10^8$  m/s e fazer uma relação entre  $f$ ,  $\lambda$  e  $c$ , que no vácuo é:

$$\lambda f = c$$

A velocidade da luz é uma constante, portanto conhecendo  $\lambda$  se chega a  $f$  vice-versa, então uma onda de 2 MHz por exemplo, tem aproximadamente 600 metros de comprimento e uma onda de 2 cm de comprimento tem uma frequência de 60 GHz.

Dá-se o nome de espectro eletromagnético, à faixa de frequência de operação utilizada pelos mais variados tipos de comunicação. O ITU, *International Telecommunication Union*, é o órgão responsável pela alocação de bandas para várias classes de serviços de acordo com as diferentes regiões do mundo (HORSTMANN, 2002). A figura 2 procura dar uma idéia do espectro eletromagnético.



**Figura 2 O Espectro Eletromagnético**

As frequências definidas no espectro eletromagnético se baseiam no comprimento de onda e a nomenclatura é dada oficialmente pelo ITU, assim a banda HF, por exemplo, se chama *High Frequency*. As redes sem fio operam em várias faixas de frequência, o *Bluetooth*, por exemplo, transmite na faixa de 2,402 a 2,480 GHz.

No espectro eletromagnético se encontram as mais variadas formas de transmissão, desde a transmissão por radiofrequência, passando por microonda e infravermelho até a luz visível. Outras opções como a luz ultravioleta, o raio X e o raio gama também são possíveis, além de possuírem frequências maiores, porém, são mais difíceis de se produzir e modular e são nocivos aos seres humanos.

O volume de informações que uma onda eletromagnética é capaz de transportar está diretamente relacionado a sua largura de banda (TANEMBAUM, 1997). Assim, quanto

maior a largura de banda, maior será a taxa de dados, porém ao mesmo tempo em que a largura de banda é proporcional a taxa de dados, o comprimento da onda é inversamente proporcional à largura de banda, sendo assim, quanto maior a largura de banda, menor será o alcance da transmissão.

Existem várias formas de se fazer uma transmissão no espectro eletromagnético. A maioria das transmissões é feita utilizando uma banda de frequência estreita, mas outras, mais complexas transmitem pulando de frequência em frequência em um padrão regular ou são dispersadas intencionalmente através de uma banda de frequência larga. Essa técnica é chamada de espalhamento espectral ou *spread spectrum*.

As duas formas mais utilizadas de se fazer o espalhamento espectral são através de salto de frequência (FHSS - *Frequency Hopping Spread Spectrum*) e através de uma sequência direta (DSSS - *Direct Sequence Spread Spectrum*). Estas formas serão discutidas na próxima seção.

Ainda há muito que pesquisar com relação ao aproveitamento do espectro eletromagnético. A medida em que o tempo passa surgem novos padrões de alta frequência, impulsionados pela grande competitividade de mercado e com o objetivo de tornar cada vez melhores as taxas de transmissão das redes de computadores sem fio, independente da tecnologia.

### **2.3 O Padrão IEEE 802.11**

O padrão IEEE 802.11 define uma série de especificações para redes sem fio. Teve seu início quando surgiu, em 1990, o grupo de trabalho IEEE 802.11 que tinha como objetivo principal a elaboração de um padrão para a comunicação sem fio entre estações de uma área local. Alguns problemas como a baixa taxa de transferência de dados, freou o desenvolvimento do padrão por alguns anos, até que em 1997 o padrão foi finalmente publicado.

Nessa época, a tecnologia sem fio estava se popularizando, e conseqüentemente as empresas de telecomunicações começaram a fabricar suas próprias interfaces *wireless*, o que causava muita incompatibilidade entre os dispositivos que utilizavam essas interfaces. O padrão veio como uma tentativa para unificar o uso das redes sem fio, garantindo que,

independente do fabricante da interface, as características dos dispositivos sem fio seriam as mesmas.

### 2.3.1 Topologias

A arquitetura do padrão IEEE 802.11 descreve dois tipos de redes sem fio: infra-estruturadas e *ad hoc*.

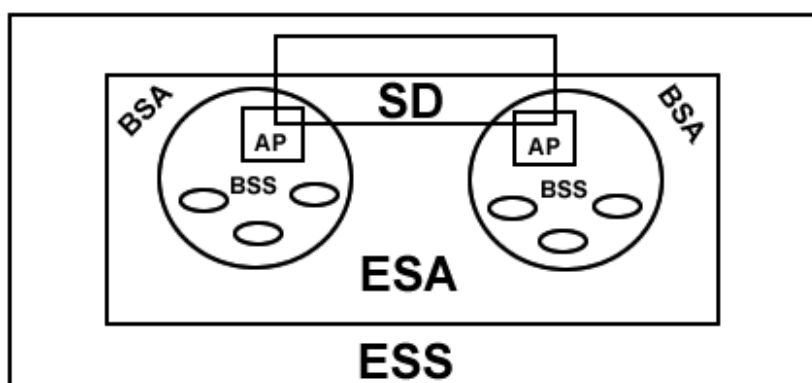
As redes infra-estruturadas são aquelas que necessitam de uma base de apoio à comunicação sem fio entre os seus componentes. Neste tipo de rede, embora os seus participantes estejam um ao lado do outro, não podem comunicar-se diretamente, precisando recorrer a infra-estrutura de rede para fazer a comunicação. Um exemplo deste tipo de rede é a rede de telefonia celular.

Em uma rede infra-estruturada, o processo de cobertura da área de operação baseia-se na divisão desta área em células. Essas células são chamadas BSA (*Basic Service Area*), um conjunto de estações comunicando-se umas com as outras constitui um BSS (*Basic Service Set*). O tamanho da área de cobertura ou célula depende das características do ambiente onde a rede foi instalada e da qualidade dos dispositivos de transmissão e recepção utilizados nas estações.

Na eventualidade de se querer cobrir uma região maior que uma célula, pode-se unir várias BSAs através de um sistema de distribuição que pode ser uma rede baseada em um meio físico de transmissão como o cabo coaxial, o par trançado ou até mesmo a fibra ótica via pontos de acesso ou APs (*Access Points*). Os pontos de acesso são estações especiais que captam as transmissões feitas pelas estações de sua BSA destinadas as estações localizadas em outras BSAs e as transmitem pelo sistema de distribuição. Os vários BSAs interligados pelo sistema de distribuição através de seus pontos de acesso constituem uma ESA (*Extended Service Area*).

O conjunto de estações formado pela união de vários BSSs conectados por um sistema de distribuição constitui um ESS (*Extended Service Set*). Cada ESS é identificado por um ESS-ID e cada BSS que está dentro de um ESS é identificado por um BSS-ID, esses dois identificadores formam o *Network-ID* de uma rede sem fio IEEE 802.11. Um ESS constitui uma rede local sem fio infra-estruturada, que é mostrada na figura 3.





**Figura 3 Rede sem fio infra-estruturada**

A infra-estrutura consiste nos pontos de acesso e no sistema de distribuição que, além de interligar os pontos de acesso, pode fornecer os requisitos necessários para interligar a rede sem fio a outras redes. As funções básicas dos pontos de acesso são (SOARES, 1995):

- **Autenticação, Associação e Re-associação:** Permitem que estações continuem conectadas à infra-estrutura mesmo quando se movimentam de uma BSA para outra. As estações utilizam procedimentos de varredura para determinar qual é o melhor ponto de acesso (a potência do sinal e a qualidade da recepção dos quadros enviados pelos APs são consideradas na classificação) e associam-se a ele, passando a acessar o sistema de distribuição através do AP escolhido.
- **Gerenciamento de Potência:** permite que as estações operem economizando energia. Para tal é necessário que o AP armazene temporariamente quadros endereçados a estações que estão poupando energia (operando com a função de recepção desabilitada ou modo *power save*). O AP e as estações operam com relógios sincronizados, periodicamente as estações ligam seus receptores e o AP transmite quadros anunciando tráfego, para que as estações possam se preparar para receber os quadros a elas endereçados que estão armazenados no AP.
- **Sincronização:** esta função deve garantir que as estações associadas a um AP estão sincronizadas por um relógio comum. A função de sincronização é implementada através de envio periódico de quadros (*beacons*) carregando o valor do relógio do AP. Esses quadros são usados pelas estações para atualizar seus relógios com base no valor neles transportado. A sincronização é usada, por exemplo, para programar o momento que uma estação deve ligar seu receptor

(*power up*) para receber as mensagens enviadas periodicamente pelo AP anunciando tráfego.

As redes sem fio *ad hoc*, também chamadas de MANET (*Mobile Ad hoc Network*) não necessitam de uma infra-estrutura de comunicação preexistente, ou seja, os dispositivos e computadores, podem comunicar-se uns com os outros diretamente, sem precisar sequer da presença de infra-estrutura. A figura 4 mostra uma rede sem fio do tipo *ad hoc*.

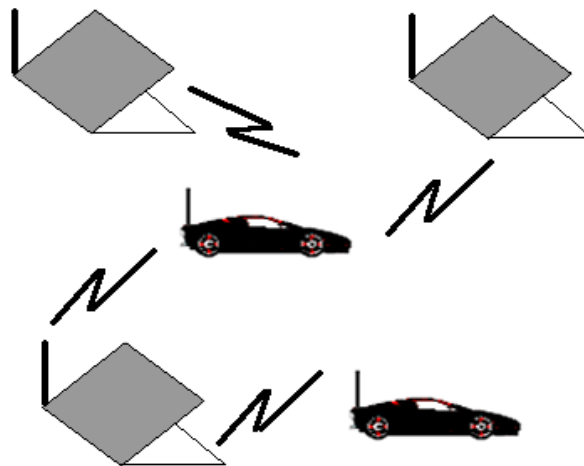


Figura 4 Rede sem fio *ad Hoc*

A topologia básica dessa rede IBSS (*Independent Basic Service Set*), estrutura a rede da seguinte maneira: Existência de dois ou mais nós ou estações STA (*Stations*) que estando na mesma área de cobertura de sinal, podem se reconhecer e estabelecer uma comunicação, formando a rede espontaneamente. Estes *hosts* também são chamados de nodos móveis ou MH (*Mobile Hosts*). Os NM podem estar em movimento e conectando-se com diferentes *hosts* ao longo do caminho. A topologia de uma rede sem fio *ad hoc* é mostrada na figura 5.

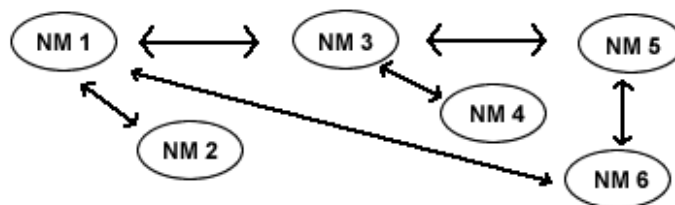


Figura 5 Topologia sem fio *ad Hoc*

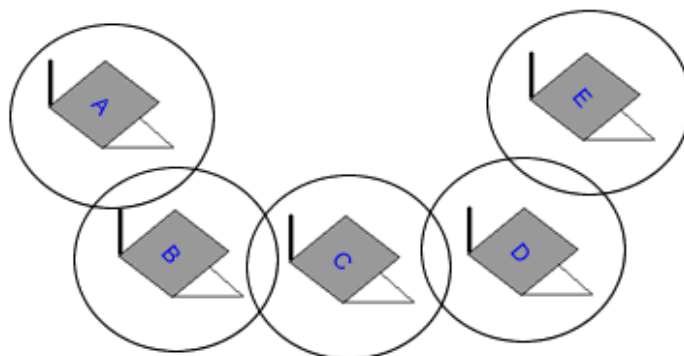
A principal aplicação das redes sem fio *ad hoc* é o seu uso em lugares onde a instalação de uma infra-estrutura de rede não é possível, como por exemplo, uma floresta. Seu uso também é feito de forma extensiva pelos militares, pois além destas redes serem mais práticas em campos de batalha, em caso de destruição da base de comunicações em uma guerra, a comunicação sem fio ainda permaneceria ativa.

Outro caso da aplicação das redes sem fio *ad hoc* consiste em sua utilização em casos de catástrofes naturais (furacões, enchentes, terremotos, etc.) onde a comunicação entre os grupos de resgate, médicos e pessoal da segurança pública se torna essencial.

A comunicação entre os nós de uma rede sem fio *ad hoc* é feita de forma muito interessante. A transmissão da informação é feita através de pacotes de rádio, sendo assim, um nó origem pode se comunicar com um certo nó destino diretamente, bastando o nó origem fazer uma transmissão do sinal e o nó destino efetuar a recepção deste sinal, considerando que o nó origem e o nó destino estejam na mesma área de alcance.

Pode ocorrer uma situação em que os nós que querem se comunicar não estejam em uma mesma faixa de alcance, nesta situação, pode ser formada uma rota entre os dois computadores através de vários *hops* (saltos) entre um ou mais computadores na rede.

A figura 6 ilustra o esquema de comunicação em uma rede *ad hoc*, o nó A quer se comunicar com o nó D, e os círculos indicam o alcance da comunicação das unidades móveis. Para isso, a transmissão do nó A passa para o nó B, e do nó B para o nó C que está no alcance do nó D, que recebe a informação. O caminho inverso da transmissão é feito quando o nó D quer se comunicar com o nó A.



**Figura 6 Comunicação entre os nós A e D em uma rede *ad Hoc***

Embora sejam muito práticas, as redes *ad hoc*, têm vantagens e desvantagens, quando se comparam com redes infra-estruturadas e fixas (LIMA, 2002). Como vantagens podem ser citadas:

- Rápida instalação: As redes *ad hoc* podem ser instaladas rapidamente em locais sem nenhuma infra-estrutura prévia;
- Tolerância a falhas: o mal funcionamento ou o desligamento de uma estação, pode ser facilmente sanado com a reconfiguração dinâmica da rede. Em uma rede fixa, quando ocorre uma falha em um roteador, o redirecionamento de tráfego é uma operação complexa;
- Conectividade: se duas estações estão dentro da área de alcance das ondas de rádio, elas têm um canal de comunicação entre elas. Em uma rede fixa, mesmo que duas estações móveis (EM) estejam uma ao lado da outra, é necessário que as estações se comuniquem com o ponto de acesso (PA) para que troquem informações;
- Mobilidade: em contraposição, à falta de mobilidade dos computadores fixos.

Como desvantagens podem ser citadas:

- Largura de Banda limitada: canais de comunicação sem fio normalmente possuem menor largura de banda que links em meios cabeados. Em ambientes internos (*indoor*), a velocidade para redes sem fio tipicamente varia de um a dois Mbps e em redes fixas esse valor já chega a Gbps;
- Erros no enlace sem fio: a taxa de erros em um link sem fio é tipicamente de um bit errado a cada  $10^5$  ou  $10^6$  bits transmitidos, enquanto que em uma fibra ótica esta taxa é tipicamente de um a cada  $10^{12}$  a  $10^{15}$  bits transmitidos;
- Localização: existe o problema de se conhecer a localização física da EM e enviar, para esse ponto, suas mensagens. Em redes fixas este problema não ocorre, pois o endereço IP indica implicitamente a localização do nodo. Para redes infra-estruturadas esse problema foi resolvido através do protocolo IP móvel, que se encarrega de localizar o nodo. Nas redes *ad hoc*, localizar o

usuário é um problema, pois não se tem nenhuma informação geográfica e o endereço da máquina não tem necessariamente mais nenhuma relação com sua posição;

- Roteamento: em uma rede fixa a topologia dificilmente se altera, os nodos ficam normalmente nas mesmas posições da rede. Em uma rede *ad hoc* os nodos movem-se de um lado para outro de forma não determinística. Se em um momento o nodo A pode se comunicar com um nodo C passando por um nodo B, nada garante que em um próximo momento isto continue ocorrendo. Tanto o nodo A quanto o nodo B ou C podem ter se movido de forma a ficarem fora da área de alcance uns dos outros. Neste caso é necessário encontrar, de alguma forma, outra rota de A para C.

### 2.3.2 A Arquitetura do Padrão IEEE 802.11

O Padrão IEEE 802.11 estipulou regras para a camada MAC (*Medium Access Control*), ou subcamada de acesso ao meio e a camada PHY (*Physic*) ou camada física para as redes LAN sem fio, porém não se ateve a camada de controle lógico de enlace LLC (*Logic Link Control*).

A camada MAC tem como protocolo de acesso ao meio o CSMA/CA (*Carrier Sense Medium Access with Colision Avoidance*) que é um protocolo com detecção de portadora e prevenção da colisão. Um dos primeiros protocolos criados para redes LAN sem fio foi o MACA (*Multiple Access with Colision Avoidance*) (TANEMBAUM, 1997).

A idéia deste protocolo é bastante simples: o transmissor deve estimular o receptor a liberar um quadro curto como saída, afim de que esta transmissão seja detectada pelas estações vizinhas, evitando o envio de dados enquanto o quadro de dados (que é um quadro grande) estiver sendo recebido.

Existem dois tipos especiais de quadros, o RTS e o CTS. Suas características são descritas a seguir:

- RTS (*Request to Send*): é um quadro curto de 30 bytes e contém o tamanho do quadro de dados que será enviado em seguida. As estações enviam este quadro quando querem fazer uma transmissão.

- CTS (*Clear to Send*): tem o mesmo tamanho do RTS e também contém o tamanho do quadro de dados a ser recebido (copiado de um quadro RTS), para fins de confirmação e aviso a outras estações. Este quadro é enviado pelas estações que estão prontas para receber uma transmissão.

Este sistema funcionava muito bem, pois dava uma certa “disciplina” à emissão e recepção de dados. Entretanto, ainda ocorriam colisões, pois duas estações poderiam enviar ao mesmo tempo um quadro RTS para uma estação em comum.

O problema das colisões estimulou várias pesquisas em cima deste protocolo e, em 1994, surgiu uma adaptação ao protocolo MACA, visando melhorar o seu desempenho. O novo protocolo passou a se chamar de MACAW e tinha, em relação ao protocolo anterior, algumas inovações:

- Aparecimento do quadro ACK: foi observado que ocorria um problema entre a camada de enlace de dados e a camada de transporte, pois sem as confirmações da camada de enlace os quadros perdidos não eram retransmitidos até que a camada de transporte percebesse sua ausência. O problema foi resolvido com a adição de um quadro ACK após cada quadro de dados transmitido com êxito.
- Uso do CSMA: para evitar que uma estação transmitisse um RTS ao mesmo tempo em que uma outra estação também estivesse transmitindo ao mesmo destino, passou-se a utilizar a detecção de portadora.

O escopo do padrão IEEE 802.11 é ilustrado na figura 7.



**Figura 7 Padrão IEEE 802.11**

A camada física é dividida em três partes, correspondentes as três tecnologias de transmissão de uma rede sem fio: radiação infravermelha difusa ou simplesmente infravermelho (IR), espalhamento espectral por salto de frequência (FHSS) e espalhamento espectral através de uma sequência direta (DSSS), essas tecnologias não são interoperáveis.

O FCC (*Federal Communications Commission*), órgão que regulamenta as telecomunicações nos EUA, liberou para uso uma faixa de frequência para fins industriais médicos e científicos chamada de ISM (*Industrial Scientific and Medical*). A maioria das redes sem fio faz suas transmissões através de sinais de rádio, portanto a banda ISM é largamente utilizada. A seguir serão detalhadas as três tecnologias de transmissão das redes sem fio:

**a) Infravermelho (IR):** A tecnologia de infravermelho consiste na transmissão de dados através de ondas infravermelhas, de frequências entre  $10^{12}$  e  $10^{14}$  Hz.

Este tipo de luz tem a característica de ser relativamente direcional, barata e fácil de se construir (um exemplo é o controle remoto de eletroeletrônicos). O grande inconveniente deste meio de transmissão é o fato de o mesmo não atravessar objetos sólidos.

As estações podem receber dados através de suas linhas de visada e por transmissões refletidas. O transmissor e um ou mais receptores se comunicam através de um plano de reflexão, que normalmente é o teto. Não é necessário que as estações estejam alinhadas entre si para se comunicarem.

No processo de comunicação, o transmissor envia seus quadros, iluminando o teto. Todos os nós devem monitorar o teto e não deve haver obstáculos entre os nós móveis.

Embora a transmissão por infravermelho não ultrapasse objetos sólidos, esta característica é útil para se montar uma rede sem fio de baixo custo e com relativa segurança em escritórios e empresas, pois um sistema infravermelho instalado em um ambiente fechado não interfere em um sistema semelhante instalado nas salas adjacentes.

A comunicação por infravermelha não pode ser usada em ambientes abertos, pois o sol brilha tanto no infravermelho como no espectro visível (TANEMBAUM, 1997).

**b) FHSS (*Frequency Hopping Spread Spectrum*):** É a tecnologia de espalhamento do espectro através de saltos de frequência.

A banda de frequência é dividida em vários canais de frequência de mesma largura, sendo assim, as estações que se comunicam utilizando esta técnica como os dispositivos *Bluetooth* por exemplo, precisam estar sincronizados para utilizar o mesmo padrão pseudo-aleatório de transmissão/recepção. Os dispositivos “pulam” de frequência em frequência, recebendo e transmitindo os dados.

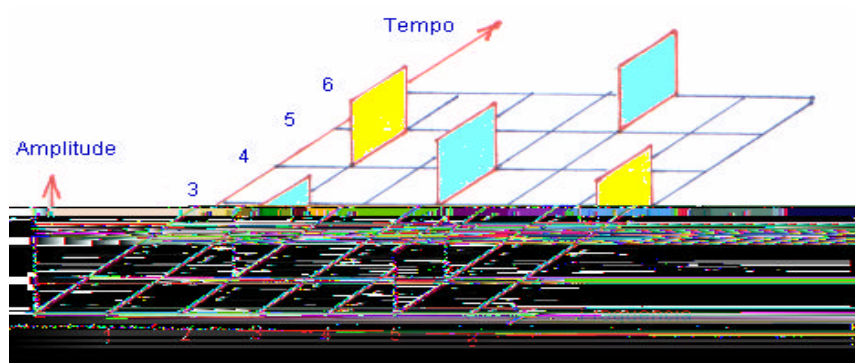
Esta característica de ocupar temporariamente pequenos canais de largura de banda fixa permite que se possam instalar várias redes sem fio que utilizam o FHSS em um mesmo ambiente sem ocasionar interferência entre as mesmas. Para isto, basta separar estas redes através de diferentes padrões de saltos pseudo-aleatórios.

As redes que utilizam FHSS são menos suscetíveis a ruídos e interferências. Na eventualidade de ocorrer qualquer tipo de interferência ou ruído, o problema só afetará a transmissão enquanto os dispositivos estiverem transmitindo naquela frequência específica. Após um curto período de tempo, os dispositivos passam a transmitir em uma frequência completamente diferente da anterior, eliminando a interferência ou ruído.

Esta característica também é útil para a segurança das informações transmitidas pela rede, além de evitar eventuais ataques através da espionagem eletrônica, visto que é bastante difícil “adivinhar” o padrão pseudo-aleatório dos saltos de frequência.

A figura 8 mostra um exemplo de transmissão/recepção utilizando o FHSS.

A cor amarela representa uma transmissão, enquanto que a cor azul representa uma recepção. A cada unidade de tempo que passa, os dispositivos mudam para uma faixa de frequência diferente, definida no padrão pseudo-aleatório de saltos de frequências consistindo em 6 saltos de frequências em um intervalo de 6 unidades de tempo.



**Figura 8** Transmissão por FHSS (ROSA, 2003)



Devido ao tempo necessário para pular de uma frequência para outra, as taxas de transmissão do FHSS são relativamente menores que as taxas de transmissão de outros sistemas, porém estes sistemas consistem em uma boa opção de instalação em ambientes abertos onde existem interferências e outros sistemas FHSS.

**c) DSSS (*Direct Sequence Spread Spectrum*):** Consiste no espalhamento espectral por seqüência direta. Neste sistema o sinal é transmitido através da extensão do mesmo em uma vasta faixa de frequência. O DSSS opera na banda ISM de 2,4 GHz e transmite em uma faixa de frequência fixa.

O sinal transmitido em um sistema DSSS é distribuído igualmente pela faixa de frequência, consistindo em um sinal com característica de baixa potência porém constante em todo o intervalo de frequência.

Devido às características de seu sinal, os sistemas DSSS também são resistentes a ruídos e interferências, pois a ocorrência destes fenômenos só afeta uma parte do sinal, restando ainda as outras partes do sinal transmitidas em frequências próximas. Caso exista um outro sistema de rádio transmitindo em uma faixa de frequência que está sendo utilizada pelo DSSS, o mesmo será pouco afetado, pois seu sinal é fraco.

O DSSS tem taxas de transmissão maiores, porém se em um mesmo ambiente forem instalados dois sistemas DSSS de mesma frequência, eles acabarão por ocasionar uma interferência muito forte, a ponto de influenciar a troca dos dados entre as estações. Sendo assim, os sistemas DSSS podem ser melhor aplicados em ambientes fechados onde não ocorram interferências externas. A figura 9 faz uma comparação entre FHSS e DSSS.

FHSS	DSSS
Complicada	Fácil e simples
Alta potência	Baixa potência
Alto período de latência	Curto período de latência
Conexão de entrada lenta	Conexão de entrada rápida
Alto alcance interno	Curto alcance interno
Altas taxas de transferências	Baixas taxa de transferência

**Figura 9** Quadro comparativo entre as tecnologias FHSS e DSSS

De acordo com a figura 9, ambas as tecnologias apresentam características suficientes para implementação em redes de computadores sem fio. A figura 10 mostra as características técnicas dos dois sistemas de transmissão.

CARACTERÍSTICAS	FHSS	DSSS
Banda	2.4 GHz	2.4 GHz
Padrão	IEEE 802.11	IEEE 802.11b
Técnica de modulação	Modulação de frequência (FM)	Modulação de amplitude (AM)
Canal da portadora	Envia dados sobre canais 1 MHz	Fixado em canal de 17 MHz
Serviços suportados	Dado, vídeo e voz.	Dado
Máx. de canais independentes	15	3
Tecnologia da indústria	HomeRF, Bluetooth	802.11b

**Figura 10 Características técnicas das tecnologias FHSS e DSSS**

## 2.4 Qualidade de Serviço (QoS)

Existem várias definições sobre o que é a qualidade de serviço. Este termo era inicialmente aplicado para algumas tecnologias de rede como a tecnologia ATM. Com o passar do tempo, o termo QoS passou a ser aplicado em diversas áreas da computação como sistemas de armazenamento de disco, sistemas operacionais entre outros.

Embora não exista uma única expressão que a defina, pode-se dizer que para a área de redes qualidade de serviço ou QoS (*Quality of Service*) diz respeito à habilidade que uma rede tem de prover melhor serviço para um determinado tráfego em tecnologias como: IP, *Frame-Relay*, ATM, *Ethernet* com o padrão IEEE 802.11 e outras (MELO, 2001).

No escopo das redes de computadores sem fio, os esforços para se garantir a QoS são bem mais difíceis que em redes cabeadas, devido a fatores como largura de banda, erros no enlace físico, localização e roteamento destas redes.

Em redes *Bluetooth*, é utilizada a radiofrequência como forma de transmissão dos dados. Um dos grandes inconvenientes deste tipo de transmissão é o fato de que, quando as estações estão fora de alcance, a comunicação entre elas é perdida.

Assim, um dos desafios principais no projeto destes sistemas móveis é o adequado fornecimento da qualidade de serviço (QoS) (LIMA, 2002), que é representada através de parâmetros da rede, tais como: *throughput*, retardo de trânsito, taxa de erros residuais, proteção, prioridade, latência, vazão, *jitter* entre outros.

Pode-se direcionar a QoS em dois sentidos: QoS no nível de pacote e QoS no nível de conexão (LIMA, 2002). A QoS no nível de pacote refere-se a análise de requisitos referentes as características dos pacotes de dados como demora de entrega do pacote, o processamento, e o desempenho do erro. A QoS no nível de conexão trata de parâmetros relacionados a conexão das redes de computadores como a inicialização e a gerência da conexão.

Vários mecanismos para garantir QoS em redes sem fio foram propostos em ambas as direções (LIMA, 2002):

- **QoS no nível de pacotes:** *Mobiware*, *MASCARA* e *INSIGNIA*;
- **QoS no nível de conexão:** Gerência da Largura de Banda para Garantir QoS, Mecanismo de Adaptação Rede/Aplicação e Predição da Mobilidade para Garantir QoS Adaptável em Redes sem Fio.

Com o intuito de atender a crescente necessidade dos usuários de redes de computadores de garantia de conexão, velocidade da rede, pouco atraso, e outras características de desempenho, surgiram vários mecanismos de garantia de QoS, tanto em redes corporativas quanto domésticas.

Estes mecanismos se dividem em dois tipos de estratégias, a da reserva de recursos e a da priorização (MELO, 2001):

- **Reserva de Recursos:** os recursos da rede são divididos de acordo com os requisitos de QoS da aplicação, e sujeitos à política de administração de largura de banda. O RSVP (*Resource Reservation Protocol*), por exemplo, fornece os mecanismos para implementação de serviços integrados (*IntServ*) baseado na reserva de recursos.
- **Priorização:** o tráfego da rede é classificado e os recursos de rede são divididos de acordo com critérios de políticas de administração de largura de banda. Para habilitar QoS os mecanismos de classificação fornecem tratamento preferencial a aplicações identificadas como tendo requisitos mais exigentes. A arquitetura de serviços diferenciados (*DiffServ*) provê este tipo de serviço.

Os parâmetros de QoS podem ser solicitados tanto por usuários quanto por aplicações. Aplicações e usuários geram diferentes tipos de tráfego e requisitam diferentes parâmetros de QoS, assim sendo, uma rede deve estar preparada para atender a todos estes requisitos

através da alocação consciente de seus recursos. Os principais requisitos de aplicações e usuários em uma rede são:

- **Vazão:** Requisito fundamental para a maioria das aplicações em uma rede de computadores, a vazão reflete o fluxo de dados da rede. A vazão pode ser medida em Kbps ou Mbps e indica a capacidade máxima de transmissão teórica de uma conexão. Este parâmetro de QoS é fundamental para o correto funcionamento de qualquer aplicação;
- **Latência:** Tempo que um pacote leva para ir e voltar de um ponto a outro da rede. Quanto menor for a latência, mais rápida é a rede. A latência ou atraso pode ser medida em milissegundos. Uma das formas de se observar a latência de uma rede é através da utilização do programa “ping”, disponível na maioria dos sistemas operacionais. Os principais responsáveis pela latência são o atraso de transmissão, codificação e decodificação, empacotamento e desempacotamento, e a demora de processamento dos dispositivos de transmissão (computadores, roteadores e *hubs*);
- **Jitter:** O *jitter* nada mais é do que uma variação na latência, ou seja, é a diferença entre o atraso máximo e mínimo. Devido ao enfileiramento de pacotes nas interfaces de rede dos dispositivos, ocasionado pela variação do tráfego da rede, o atraso também varia, gerando o *jitter*;
- **Perda de pacotes:** É uma medida exibida como uma porcentagem, que mostra o montante de pacotes perdidos em uma transmissão. A perda de pacotes é calculada dividindo-se a quantidade de pacotes perdidos pela quantidade de pacotes transmitidos, em cada intervalo de tempo. Quanto maior a perda de pacotes mais prejudicada será a rede;
- **Erros:** Este requisito mede a integridade de uma transmissão. A taxa de erros é expressa como uma porcentagem do total da transmissão dentro de um determinado período. Este problema pode ser causado por falhas na transmissão dos pacotes, ocasionando em erros no conteúdo dos mesmos. Outros fatores causadores de erros são: falha nos meios de transmissão, problemas na codificação dos dados e sobrecarga em alguns dos equipamentos da rede.

Este último é um bom parâmetro para caracterizar o estado de um enlace sem fio (PEREIRA, 2003). A taxa de erros em um ambiente sem fio com comunicação *full-duplex* e

sem interferências deve ser inferior a 6%. São admitidas taxas de erros de 10%, porém, o enlace é classificado como degradado. Para taxas superiores a 10% o *link* torna-se inadequado para operação (RIBAS, 2002).

Para prover QoS de forma apropriada, é necessário também que se tenha um certo controle sobre o tráfego da rede. Como o tráfego é variável, existem mecanismos implementados nas interfaces de redes que tentam controlar da melhor maneira possível este tráfego. Este controle é feito através do uso de algoritmos de filas e do descarte de pacotes.

Se o tráfego de entrada é maior do que o tráfego de saída da interface, os pacotes que entram são enfileirados de acordo com o algoritmo de enfileiramento. Com relação ao descarte de pacotes, pode-se avaliar o requisito de perda de pacotes, e, no enfileiramento de pacotes, é avaliado o requisito da latência. Os algoritmos de enfileiramento utilizados no trabalho são descritos a seguir:

- ***First In First Out (FIFO)***: Este tipo de fila é o mais popular, sendo utilizado tanto no escopo das redes de computadores quanto dos sistemas operacionais, através do escalonamento de processos. Neste esquema, os pacotes vão sendo atendidos a medida em que chegam na fila, ou seja, o FIFO não altera a ordem dos pacotes enfileirados. Embora esse mecanismo possa privilegiar uma aplicação em detrimento da outra ele não garante o controle do serviço sobre o fluxo de tráfego.
- ***Stochastic Fair Queuing (SFQ)***: Esta disciplina de filas utiliza um esquema de *hash* junto com várias filas para colocar os pacotes que entram em suas filas correspondentes. O número de filas é relativamente menor que o número possível de fluxos. Os fluxos vão sendo direcionados para as filas, sendo que os que estiverem na mesma fila tem tratamento equivalente. A desvantagem deste esquema é que se um fluxo mais importante será desfavorecido se cair na mesma fila de um fluxo de menor prioridade.
- ***Deficit Round Robin (DRR)***: O DRR funciona utilizando um “*token*” para controlar o fluxo de pacotes. Através de um ciclo, o *token* vai passando de fila em fila. Em cada ciclo as filas recebem um número fixo de créditos, assim, elas só podem liberar os pacotes se o tamanho do pacote for menor ou igual ao número de créditos da fila. Quando um pacote é liberado, o crédito da fila é

reduzido de forma equivalente. Essas operações são executadas na medida em que o *token* vai passando pelas filas e ocorrem de forma ordenada. Para organizar os fluxos nas filas, é utilizado o esquema SFQ. Esse mecanismo provê, para alguns fluxos, um controle mais eficiente com relação a garantias de envio e latência.

- **Fair Queuing (FQ):** Neste esquema, é atribuído para cada pacote um tempo teórico de finalização. O algoritmo retira pacotes da fila quando houver capacidade no canal. Pacotes com tempo de finalização menores são enviados antes de pacotes com tempo de finalização maiores.

## 2.5 Problemas Relacionados às Redes Sem Fio

### 2.5.1 Handoff

As redes sem fio, como já foi dito, utilizam como forma de transmissão dos seus dados a radiofrequência. As redes *ad hoc*, assim como as infra-estruturadas operam dentro do mesmo alcance, caso contrário uma estação não poderia se comunicar com a outra. Quando uma estação móvel muda de sua área de alcance para outra área de alcance, ou seja, muda de uma célula para outra, acontece o fenômeno do *handoff*.

O *handoff* é uma funcionalidade que permite que um dispositivo continue a sua comunicação quando este passa de uma célula para outra. Um exemplo bastante comum de *handoff* é o da rede de telefonia celular, quando um usuário muda de uma área de serviço para outra.

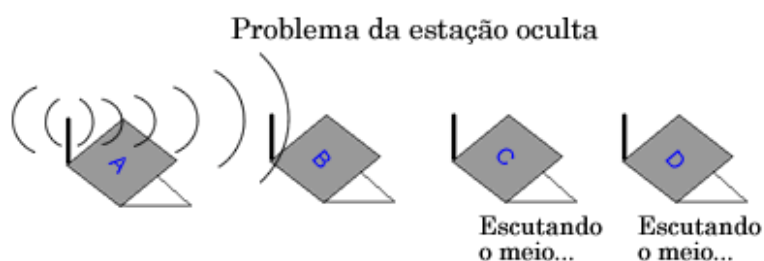
Outros fatores como efeitos de sombreamento (*shadowing*), reflexão ou atenuação do sinal, interferências ou obstáculos também podem provocar o *handoff*, dentro da mesma célula. Este tipo de *handoff* é chamado de intracelular.

### 2.5.2 Problema da Estação Oculta e Problema da Estação Exposta

Para garantir um acesso ordenado ao meio as redes sem fio usam o protocolo CSMA/CA (*Carrier Sense Medium Access with Collision Avoidance*) que é um protocolo com detecção de portadora e prevenção da colisão. Esse protocolo é bastante útil para a dinâmica da rede, ele pode causar alguns inconvenientes.

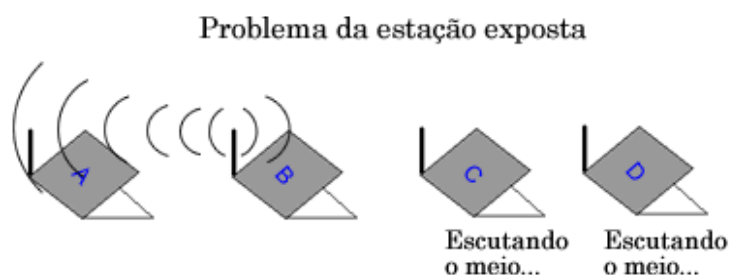
Sejam A B C e D nós móveis. A e B estão na mesma área de alcance, podendo haver interferência entre eles, o nó C também pode interferir em B e D, mas não pode interferir em A.

Quando A transmite algo para B, o nó C, ao detectar o meio físico, não perceberá o nó A, pois está fora de alcance do mesmo e concluirá de forma errada que pode iniciar a transmissão. Se o nó C não começar a transmitir, ela irá interferir em B, removendo o quadro de A. O problema de uma estação não conseguir detectar outra estação concorrente para o meio físico é chamado de problema da estação oculta (*hidden station problem*). A figura 11 mostra um cenário onde este problema pode ocorrer.



**Figura 11 Problema da estação oculta**

Por outro lado, pode ocorrer a situação inversa, ou seja, o nó B transmitindo para o nó A. Se o nó C detectar o meio físico, ele perceberá a transmissão e concluirá erroneamente que não pode transmitir para D, quando esta transmissão apenas geraria uma recepção de má qualidade na área entre B e C, onde nenhum destes nós estão localizados. Este problema é chamado de problema da estação exposta (*exposed station problem*), e é exposto na figura 12.



**Figura 12 Problema da estação exposta**



## 3 A TECNOLOGIA DE INTEGRAÇÃO *BLUETOOTH*

Este capítulo trata especificamente da tecnologia *Bluetooth*. Todos os aspectos relevantes para o desenvolvimento do trabalho como as principais características da tecnologia, detalhes de implementação, bem como sua arquitetura, serão mostrados em detalhe, com a intenção de proporcionar uma visão geral da tecnologia.

### 3.1 Introdução

Com a popularização das redes de computadores, usuários comuns passaram a dispor de benefícios que só as grandes corporações e universidades possuíam. Esta popularização, aliada ao interesse comercial dos fabricantes de rede e a necessidade dos usuários de estarem permanentemente conectados à Internet, estimulou o desenvolvimento das mais variadas tecnologias de rede.

Deste modo, apareceram as tecnologias atuais como a telefonia celular, o GPS, as redes sem fio, entre outras. Estas tecnologias já existem há algum tempo e estão acessíveis a qualquer tipo de usuário, porém, a interconexão entre diferentes sistemas tem sido um desafio para os projetistas de hardware.

Outro problema é a quantidade enorme de fios necessários à conexão de diferentes periféricos como impressoras e scanners, monitores e outros. A existência de diferentes padrões de conexão como as portas Serial e Paralela, e as conexões USB e *Firewire*, faz com que o sonho de ligar a impressora no microcomputador de um usuário comum se transforme em um pesadelo.

Neste cenário aparentemente “caótico” da modernidade, surgiu a tecnologia *Bluetooth*. O que era no início uma tentativa de eliminar as conexões cabeadas entre diferentes periféricos tornou-se uma tecnologia de conexão sem fio relativamente veloz e de baixo custo, que interliga desde periféricos comuns até dispositivos completamente diferentes como uma geladeira e um telefone celular.

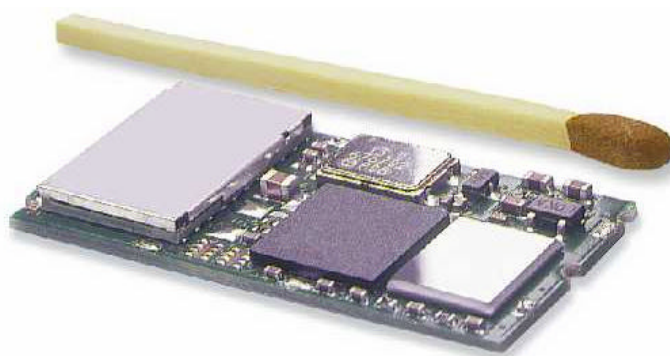
### 3.2 Breve Histórico

No ano de 1994, a empresa sueca Ericsson iniciou o desenvolvimento de uma tecnologia de interconexão sem fio entre diferentes dispositivos chamada inicialmente de *Multi-Communicator Link – MC Link*. Como os engenheiros do projeto optaram por utilizar uma banda de baixa frequência de rádio não licenciada, a tecnologia se tornou de baixo custo e a partir disto, descobriram-se outras aplicações que poderiam utilizar essa tecnologia.

Com o objetivo de difundir a nova tecnologia, em 1997 a Ericsson convidou outras empresas de tecnologia para participar de seu projeto e em 1998 foi criado o Grupo de Interesse Especial *Bluetooth SIG (The Bluetooth Special Interest Group)*.

Inicialmente, o grupo era composto por 5 empresas de telecomunicações e de informática: Ericsson, IBM, Intel, Nokia e Toshiba. Em 1999 outras empresas aderiram ao SIG: Microsoft, Motorola, 3Com e Lucent Technologies. Atualmente, o Bluetooth SIG conta com mais de 2500 associados e constitui o órgão responsável pela tecnologia *Bluetooth*.

O *Bluetooth* é um microchip de padrão aberto e global para transmissões via rádio que provê conexões simples e sem fios entre PCs, telefones móveis, impressoras e muitos outros dispositivos, permitindo também que esses dispositivos tenham acesso a recursos da rede. A figura 13 ilustra as dimensões do microchip *Bluetooth*.



**Figura 13** Dimensões do microchip *Bluetooth* (*The Official Bluetooth Website*)

Através do uso da transmissão de rádio, a transferência de voz e dos dados realiza-se em tempo real. A modalidade de transmissão sofisticada adotada na especificação do *Bluetooth* assegura a proteção contra interferências e a segurança dos dados. O sistema de rádio do *Bluetooth* é construído em um microchip pequeno e opera em uma faixa de frequência de disponibilidade global assegurando a compatibilidade de uma comunicação em nível mundial.

A especificação do *Bluetooth* tem três níveis de potência definidos: um nível mais baixo que pode cobrir uma área menor como por exemplo dentro de um quarto, um nível médio que pode cobrir, uma casa e um nível mais alto que pode cobrir uma área mais ampla. Os controles de software e a codificação da identidade embutidos em cada microchip garantem que somente as unidades pré-ajustadas por seus proprietários possam se comunicar. O *Bluetooth* suporta conexões ponto-a-ponto e multiponto.

### 3.3. Principais Usos

Para facilitar a vida do usuário o *Bluetooth* visa exclusivamente a eliminação de qualquer cabo ou fio, razão pela qual várias empresas de telefonia celular e de computadores estão juntas neste projeto, tendo como principal objetivo fazer a ligação entre todos os seus produtos e periféricos. Entre os vários usos e vantagens do *Bluetooth* destacam-se:

- **Sincronização Automática** : Sincronização automática do *desktop* para o *notebook* (PC-PDA e PC-HPC), ou para um telefone móvel, integrando diferentes dispositivos (Figura 14).

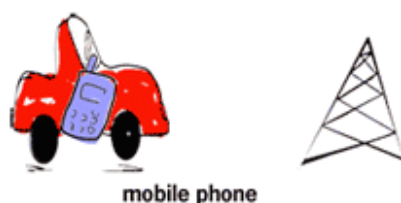


Figura 14 Sincronização automática (*The Official Bluetooth Website*)

- **Conferência Interativa:** Em reuniões e conferências poderão ser transferidos documentos selecionados imediatamente com os participantes escolhidos, além de trocas de cartões empresariais eletrônicos automaticamente, sem qualquer uso de fio (Figura 15).



Figura 15 Conferência interativa (*The Official Bluetooth Website*)

- **Gateway para a Internet:** Uso do *notebook* para navegar na Internet em qualquer lugar, independentemente do tipo de conexão seja ela feita por celular ou PSTN,ISDN, LAN ou xDSL. (Figura 16).



Figura 16 Gateway para a Internet (*The Official Bluetooth Website*)

- **Dispositivos multifuncionais:** Em uma casa, um celular pode ter funções como as de um telefone de linha fixa. Quando em movimento, funciona normalmente como um telefone móvel. E quando este mesmo telefone chega dentro de alcance de outro telefone móvel que também possui o chip *Bluetooth* embutido ou mesmo aparelho funciona como um *walkie talkie* (Figura 17).

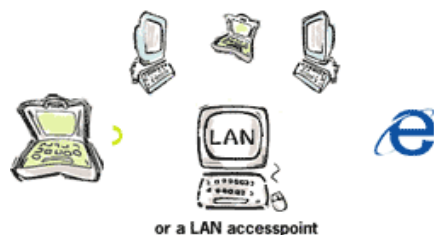


Figura 17 Dispositivos multifuncionais (*The Official Bluetooth Website*)

- **Maior Mobilidade:** A conexão de um fone sem fio a um celular ou *notebook* para manter as mãos livres para outras tarefas mais importantes (Figura 18).



Figura 18 Maior mobilidade (*The Official Bluetooth Website*)

### 3.4 Características Técnicas do *Bluetooth*

A tecnologia Bluetooth dispõe de vários mecanismos de conexão, assim como formatos de configuração de rede e formato de pacotes. Estes mecanismos garantem a interconexão dos equipamentos de forma segura e eficiente.

#### 3.4.1 Tipos de rede Bluetooth

Existem dois tipos de redes ou configurações possíveis em uma rede *Bluetooth*. Para que os nós da rede se comuniquem, é preciso que o dispositivo *Bluetooth* exerça dois papéis diferentes: o papel de *master* e o papel de *slave*. Um dispositivo *master* é aquele que inicia a comunicação em um certo instante, e um dispositivo *slave* é aquele dispositivo com o qual o *master* se comunica.

Os dispositivos se comunicam de forma seqüencial, ora na condição de *master*, ora na condição de *slave*. Os dispositivos *Bluetooth* podem estar inseridos em uma *piconet* ou em uma *scatternet*:

- ***Piconet***: É o agrupamento de dois ou mais dispositivos *Bluetooth*. Uma *piconet* é formada de no mínimo dois e no máximo nove equipamentos *Bluetooth*, onde um dispositivo é o *master* e os demais são *slaves*.
- ***Scatternet***: Uma *scatternet* é o conjunto de várias *piconets*. Uma *scatternet* pode ser formada por várias *piconets* e a comunicação entre elas é feita pelos dispositivos *master* de cada *piconet*. Cada *piconet* é identificada por uma seqüência diferente de salto (HORSTMANN, 2002).

#### 3.4.2 Canais de comunicação

O *Bluetooth* conta com dois tipos diferentes de canais de comunicação. Estes canais são utilizados para o estabelecimento de conexões e transmissão de dados entre os equipamentos da rede. São eles:

- ***Synchronous Connection Oriented (SCO)***: Este canal é do tipo simétrico, e é utilizado principalmente para o transporte de voz. Um dispositivo *master* pode suportar até três canais simultâneos e um dispositivo *slave* pode suportar dois ou três canais.
- ***Asynchronous Connection Less (ACL)***: Este canal suporta tanto conexões assimétricas quanto conexões simétricas. Somente pode existir um canal ACL entre um dispositivo *master* e um *slave* em uma *piconet*. O canal ACL é mais utilizado para o transporte de dados.

### 3.4.3 O Pacote de Dados Bluetooth

Um pacote *Bluetooth* é formado por três partes:

- **Access Code:** Neste campo se encontram as informações de identificação dos pacotes *Bluetooth*, além de outras informações como sincronização temporal e correção de erros. Seu tamanho fica entre 68 ou 72 bits. O *access code* também é usado para informações relacionadas ao processo de conexão dos dispositivos *Bluetooth*. Existem três tipos de códigos de acesso (HORSTMANN, 2002): *Channel Access Code* (CAC), que é utilizado para identificar uma *piconet* e *Device Access Code* (DAC) e *Inquiry Access Code* (IAC), usados nas conexões *Bluetooth*.
- **Packet Reader:** O campo *packet reader* tem um tamanho de 18 bits, mas devido ao esquema de proteção contra erros de transmissão utilizado neste campo, seu tamanho totaliza 54 bits. No esquema de proteção contra erros chamado de codificação 1/3 FEC (*Forward Error Correction*), o mesmo bit é transmitido três vezes seguidas. Este campo contém as informações relativas ao endereço do dispositivo destino do pacote e tipo de dado ou pacote de controle. Dentro do *packet reader* existem ainda campos para controle de fluxo, sequenciamento e reconhecimento de pacotes e um cabeçalho do tipo CRC de 8 bits.
- **Packet Payload:** O *packet payload* é responsável por carregar os dados das transmissões. O tamanho do campo *payload* depende do tipo de canal. Para canais ACL por exemplo, o *payload* carrega um cabeçalho de 8 ou 16 bits que indica o tamanho do pacote de dados, além de fornecer campos para canais lógicos e fluxo de controle. O *payload* também suporta fragmentação de pacotes de dados.

Com relação aos canais lógicos, o *Bluetooth* emprega cinco diferentes tipos de canais para controle da informação (HORSTMANN, 2002):

Controle da conexão:

- Gerenciamento do canal (LC – *Control Channel*)
- Gerenciamento da conexão (LM – *Link Manager*)

Informação do usuário:

- Informação assíncrona – (UA – *User Asynchronous*)
- Informação isosíncrona – (UI – *User Isosynchronous*)
- Informação síncrona – (US – *User Synchronous*)

### 3.5 A Arquitetura da Tecnologia *Bluetooth*

A arquitetura do *Bluetooth* é baseada no padrão IEEE 802.11 que define uma rede sem fio do tipo *ad hoc*, e é implementada em seu núcleo, o “*Bluetooth Core*”. Essa arquitetura permite a interconexão entre os módulos *Bluetooth* independente da necessidade de uma estação de ponto de acesso, ou seja, os dispositivos podem conversar diretamente entre si e com outros simultaneamente.

O modelo do *Bluetooth* foi especificado de forma que as suas camadas fossem dispostas em uma hierarquia bastante funcional. As camadas inferiores são responsáveis pela interconexão física dos dispositivos e pelo controle da conexão entre as unidades *Bluetooth*, enquanto que as camadas superiores são responsáveis pelos protocolos de aplicação e interoperabilidade entre diferentes protocolos, ou seja, o *Bluetooth* garante a interação com protocolos que não estejam implementados na sua especificação.

Em comparação com o modelo OSI, o *Bluetooth* implementa as camadas 1 e 2 do modelo. As outras camadas são implementadas em software. A figura 19 faz uma comparação entre a arquitetura *Bluetooth* e a arquitetura OSI.

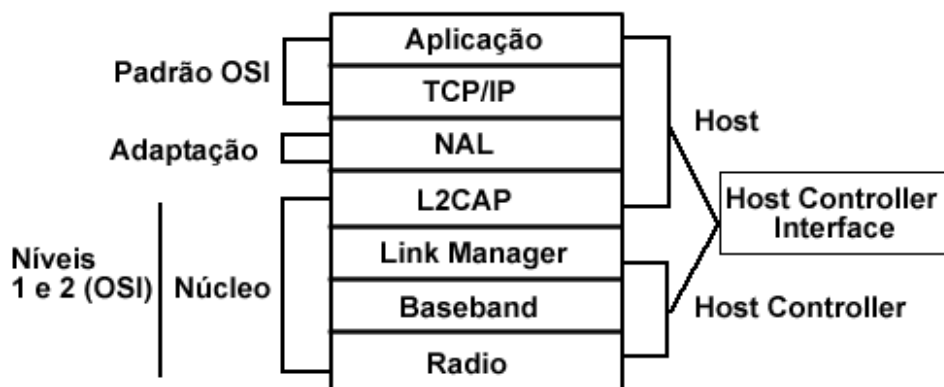




Figura 19 Comparação entre as arquiteturas OSI e Bluetooth

O *Bluetooth Core* ou núcleo *Bluetooth*, engloba as camadas de *Radio*, *Baseband*, *Link Manager*, *L2CAP* e a interface controladora do *host* que é chamada de *Host Controller Interface*. Com relação as camadas superiores, temos a camada *NAL (Network Adaptation Layer)*, ou camada de adaptação, que junto com as camadas do modelo OSI, são implementadas em software no *host*.

A interface de controlador de *host* só é necessária quando o *L2CAP* estiver embutido no software (HORSTMANN, 2002). Dependendo do hardware, o *L2CAP* já pode estar incorporado no mesmo. Assim, o *L2CAP*, o *Link Manager* e o *Baseband* comunicam-se diretamente, o que elimina a necessidade do *Host Controller*.

A figura 20 mostra uma visão mais detalhada da arquitetura *Bluetooth*, que por abranger vários tipos de protocolos também é chamada de pilha de protocolos *Bluetooth*.

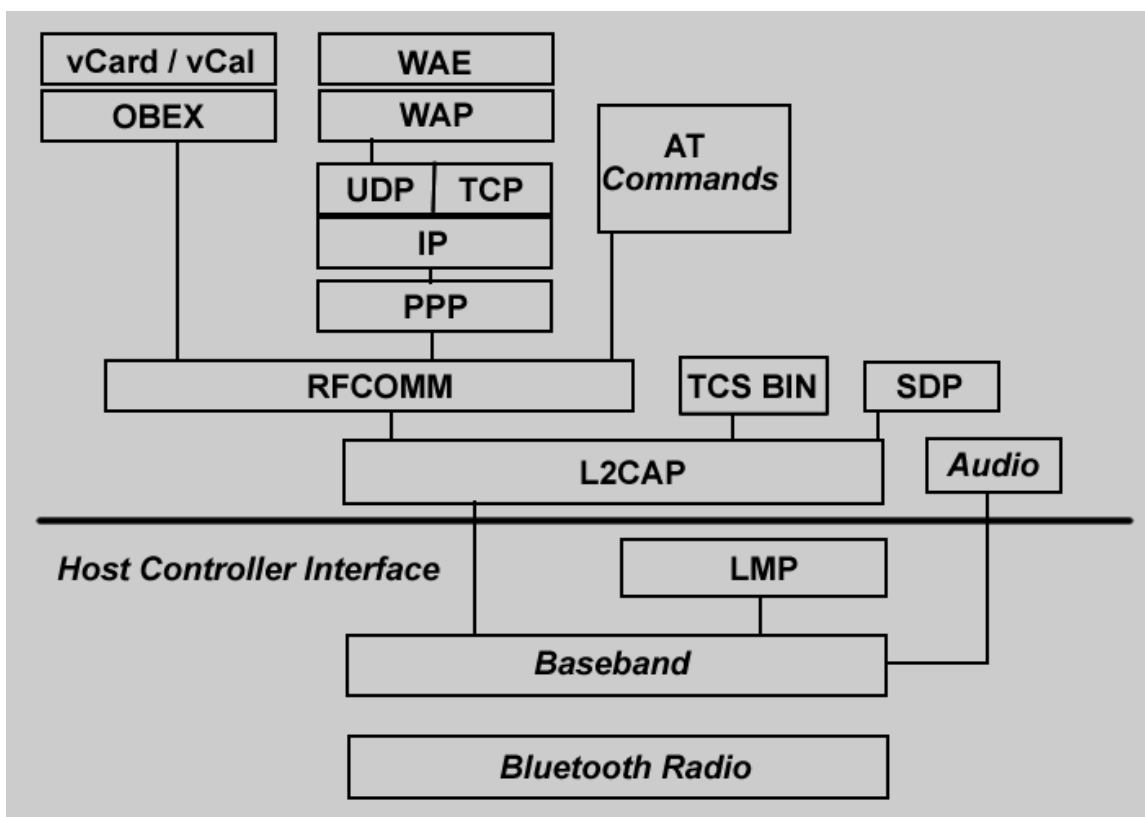


Figura 20 Pilha de protocolos Bluetooth

De acordo com a figura 20, a pilha de protocolos *Bluetooth* é composta pelas seguintes camadas:

### 3.5.1 *Bluetooth Radio*

É a base da pilha. Esta camada é responsável pela interface de conexão física da tecnologia. Nela estão implementados os mecanismos de transmissão por radiofrequência dos dispositivos *Bluetooth*.

O *Bluetooth Radio* opera na faixa de frequência ISM de 2,4 a 2,4835 GHz, e utiliza a tecnologia de transmissão FHSS (*Frequency Hopping Spread Spectrum*). Neste tipo de transmissão a faixa de frequência de operação do *Bluetooth* é dividida em 79 canais (ou 23 canais em países onde a banda ISM é menor) de 1 MHz onde os dispositivos *Bluetooth* fazem a transmissão.

Para fazer a transmissão, os dispositivos mudam a faixa de frequência de acordo com um padrão pseudo-aleatório (formado de acordo com um algoritmo) a uma taxa de até 1600 saltos por segundo ("1600 *hops/s*") a frequência de salto é dada de acordo com a seguinte fórmula:

$$f = 2402 + k \text{ MHz}$$

Onde  $f$  é a frequência a ser transmitida e  $k$  são os intervalos de frequência de 1MHz variando de 0 a 78. Cada transmissão tem duração de 625 ms, tempo chamado de "time slot" do sistema.

### 3.5.2 *Baseband*

Esta camada é responsável pelo controle do *Bluetooth Radio*, provendo os saltos de frequência utilizados na transmissão FHSS. Ela é responsável pela interface física do canal de frequência do *Bluetooth*. Esses canais são gerenciados pela camada independentemente de outros serviços como, por exemplo, a criptografia dos dados. É nesta camada que são estabelecidas as conexões de voz e de dados dos dispositivos *Bluetooth*.

### 3.5.3 Link Manager Protocol (LMP)

Este nível atende pelo processo de conexão entre os dispositivos *Bluetooth*, através de funcionalidades para anexar e desanexar dispositivos operando como *slaves* e regras de chaveamento entre um *master* e um *slave* para estabelecimento de conexões.

Nas transmissões de dados, a LMP opera o controle e a negociação de tamanhos de pacotes. A LMP também gerencia o consumo e os modos de energia dos dispositivos. É nesta camada que se encontram os mecanismos de criptografia da tecnologia *Bluetooth*.

Em resumo, a LMP possui as seguintes características (HORSTMANN, 2002):

1. Gerenciamento *piconet*.
2. Configuração de conexão.
3. Funções de segurança.

### 3.5.4 Host Controller Interface (HCI)

Com o objetivo de interconectar diferentes padrões de conexão de dispositivos, o *Bluetooth* pode ser conectado em outros periféricos através de conexões, como por exemplo, um cartão PCMCIA para *notebooks*. Estas conexões, geralmente implementam os níveis mais baixos da especificação *Bluetooth*. Para que ocorram essas conexões, são necessários um driver que precisa ser instalado no *host*, e um *host controller interface* para o *Bluetooth* aceitar dados sobre o meio físico.

Assim, quando os níveis inferiores da arquitetura estão implementados no hardware, são necessários 2 níveis adicionais:

- *HCI Driver*: É o software necessário para a interface do *host controller*, que é instalado no *host*. Sua função é a de formatar os dados que irão passar pelo *host controller* no hardware *Bluetooth*.
- *Host Controller Interface*: Faz parte do hardware *Bluetooth* e é o dispositivo pelo qual passam as comunicações sobre o meio físico.

### 3.5.5. Logical Link Control and Adaptation Protocol (L2CAP)

Depois que a conexão entre dois ou mais dispositivos *Bluetooth* foi estabelecida, esta camada procura servir às aplicações dos níveis de protocolos superiores através de algumas funcionalidades:

- Multiplexação: Múltiplas aplicações podem utilizar o canal entre dois dispositivos simultaneamente.
- Segmentação e Reagrupamento: O nível L2CAP aceita pacotes com tamanhos de até 64 kb, porém o nível *baseband* só aceita pacotes de 2.745 bits, por isso, o L2CAP reduz o tamanho dos pacotes que as aplicações enviam para serem recebidos pela camada inferior. Quando os pacotes são recebidos, o processo de reagrupamento é aplicado.
- Qualidade de Serviço (QoS): Certas aplicações que necessitam de parâmetros de QoS como latência, perda de pacotes e largura de banda, são atendidas pelo L2CAP, que verifica se o canal pode prover os requisitos de QoS solicitados pelas aplicações.

É importante salientar que essas funções só não são desempenhadas pelo L2CAP se houver um *host controller* em uso.

### 3.5.6 RFCOMM

Nesta camada é feita uma emulação do protocolo serial de comunicação de dados RS-232, o que provê recursos de transporte para as camadas superiores. O RFCOMM provê a capacidade de transporte para protocolos de mais alto nível que usam um mecanismo de transferência baseado em linha serial (DANTAS, 2002).

## 4 AMBIENTE DE DESENVOLVIMENTO

Para avaliar os parâmetros de QoS, será usado um simulador de redes. Foi decidido usar o simulador, pois através dele, pode ser simulada uma rede maior, com mais recursos e mais parâmetros podem ser analisados, sendo impossível fazer o mesmo trabalho a partir de uma rede real, sendo que o primeiro empecilho seria a ausência total de dispositivos wireless.

Sendo assim, o simulador escolhido foi o NS-2 (*Network Simulator* versão 2.26)(NS2), é parte do projeto VINT (*Virtual Internetwork Testbed*), desenvolvido pelo ICS (*Information Sciences Institute*) da *University of Southern Califórnia*.

### 4.1 Sobre o NS

O NS-2 é um simulador de eventos discreto, com foco voltado na pesquisa em redes de computadores, fornecendo suporte substancial para simulação do TCP, roteamento, e protocolos de *multicast* sobre redes com fio e *wireless*. Foi totalmente programado na linguagem C++ seguindo o paradigma de orientação a objeto. Foi desenvolvido para ambiente Linux, mas atualmente existe uma versão mais pobre para Windows. Como é um software *open source* (ie. seu código pode ser visto e modificado por qualquer pessoa), permite adaptações em C++ e Java. Além disso, possui uma linguagem própria, Otcl (*Object tool command language*) que é a linguagem onde são escritos os scripts contendo os dados das redes a serem simuladas, como a topologia, número de nós, tempo, etc.

O simulador é acompanhado de uma ferramenta chamada de *Network Animator* (NAM). Com esta ferramenta é possível visualizar o diagrama da rede durante as animações, acrescentar comentários, avançar ou retroceder na visualização do comportamento do modelo.(Horstman, 2002).

O NS-2 é bem conceituado na comunidade científica, sendo reconhecido por entidades como IEEE e a IBM, que recomendam o uso do mesmo em pesquisas na área de gerência de redes.

Depois de escrito o script tcl, ele é interpretado e se não contiver erros, gera um arquivo de trace, com todos os eventos que ocorreram na simulação. Esse arquivo de trace pode ser importado para uma planilha do Excel. Após importar o arquivo e selecionar as colunas que

são relevantes á essa análise, a planilha contendo os dados da planilha de trace é como a mostrada na figura 21.

Evento	Tempo	Nó	Identificador do pacote		
S	205.000.000.000	3	1510	cbr	54
S	205.000.000.000	4	1511	cbr	108
S	205.000.000.000	5	1512	cbr	216
S	205.000.000.000	6	1513	cbr	432
S	205.000.000.000	7	1514	cbr	865
S	205.000.000.000	8	1515	cbr	1730
R	205.004.701.377	9	1510	cbr	74
R	205.007.263.308	10	1511	cbr	128
R	205.012.436.352	12	1513	cbr	452
R	205.023.531.221	13	1514	cbr	885
R	205.044.042.385	14	1515	cbr	1750
R	205.047.628.231	11	1512	cbr	236
S	206.000.000.000	3	1516	cbr	54
S	206.000.000.000	4	1517	cbr	108
S	206.000.000.000	5	1518	cbr	216
S	206.000.000.000	6	1519	cbr	432
S	206.000.000.000	7	1520	cbr	865
S	206.000.000.000	8	1521	cbr	1730
R	206.002.692.053	9	1516	cbr	74

**Figura21 Arquivo trace importado**

A primeira coluna identifica o tipo do evento. Um evento s representa um pacote enviando e um pacote r um pacote recebido. Há ainda o evento d, que representa pacote perdido. A coluna tempo mostra o tempo no relógio da simulação em que tal evento ocorreu, do mesmo modo, a coluna nodo, mostra a qual nodo tal evento está relacionado. Por ex: um s representa que o nodo x está enviando um pacote, e um r, um nodo y está recebendo um pacote. A última coluna identifica o pacote. São três dados: o primeiro é o identificador único do pacote, o segundo é o tipo de dado do pacote e o terceiro é o tamanho do pacote.

(Horstman, 2002), validou o processo de análise dos dados de uma simulação no software NS-2 usando um arquivo de trace. Ele validou os resultados obtidos através dos dados do arquivo trace usando o software Arena. Com base nos resultados da validação do protótipo usado, ficam garantidos os procedimentos para cálculo de latência e *jitter*.

Assim, para saber a latência de um pacote, subtrai-se para o mesmo identificador, o tempo  $r$  de  $s$ , e para obter o *jitter*, subtrai-se a latência do pacote atual pela latência do pacote anterior. A perda de pacotes é obtida contando-se as ações “d”.

O processo de análise dos dados é totalmente determinístico. Algumas variáveis do processo de geração de pacotes foram arbitradas, como tempo de transferência e taxa de geração de pacotes, instantes iniciais e finais da geração dos fluxos. Desta forma torna-se mais fácil mapear o processo ao longo do tempo. O mapeamento permite prever qual é a situação teórica do sistema em qualquer instante de tempo. (Horstman, 2002).

## 4.2 Modelo de Simulação

A arquitetura do modelo simulado é composta de duas *piconets* interligadas, formando uma *scatternet*. As filas são formadas em dois pontos, no dispositivo *slave*, gerador de tráfego, e no dispositivo *master*, que recebe os dados. O *master*, não sendo o destino final dos pacotes, deve enviá-los pelo caminho que une as duas *piconets*. (Horstman, 2002).

Cada *piconet* é formada por seis dispositivos *slaves* e um *master*. O sétimo *slave* é membro das duas *piconets* agindo como *gateway* e tornando as duas *piconets* uma *scatternet*.

Dessa forma, estabeleceu-se uma relação unívoca de gerador de pacotes, na *piconet* A, e de receptor de pacotes, *piconet* B. Todos os *slaves* da *piconet* A devem enviar os pacotes para o seu dispositivo *master*, e este redirecionar os pacotes para o dispositivo *slave* comum as duas *piconets*. A figura 22 mostra a topologia da rede.

Os seis *slaves* de cada *piconet* estão em constante movimento. Os outros três nodos não se movem para manter a topologia da rede. Se os dispositivos *masters* estivessem em movimento, eles poderiam deixar de ser *masters*, e nesse momento toda a topologia da rede seria modificada, podendo ser formada até mais *piconets*, e os pacotes estabelecerem uma rota alternativa, prejudicando assim o processo de análise. Com os três nós principais parados, é possível estabelecer a rota forçando os pacotes a passarem sempre por esses nós, independente da localização dos outros *slaves*.

Esse ambiente onde os nós estão localizados, pode ser considerado um local, como uma sala com dimensões 200x150m. Cada nó *slave* irá se deslocar para um determinado local e em uma determinada velocidade, de forma, que esse local esteja dentro do alcance do nó *master* deste *slave*, para que a topologia da rede seja mantida. Assim que chegar á esse

destino, o nó irá parar por uma quantidade de tempo aleatória, e depois irá seguir para outro destino, e assim sucessivamente, até terminar o tempo da simulação.

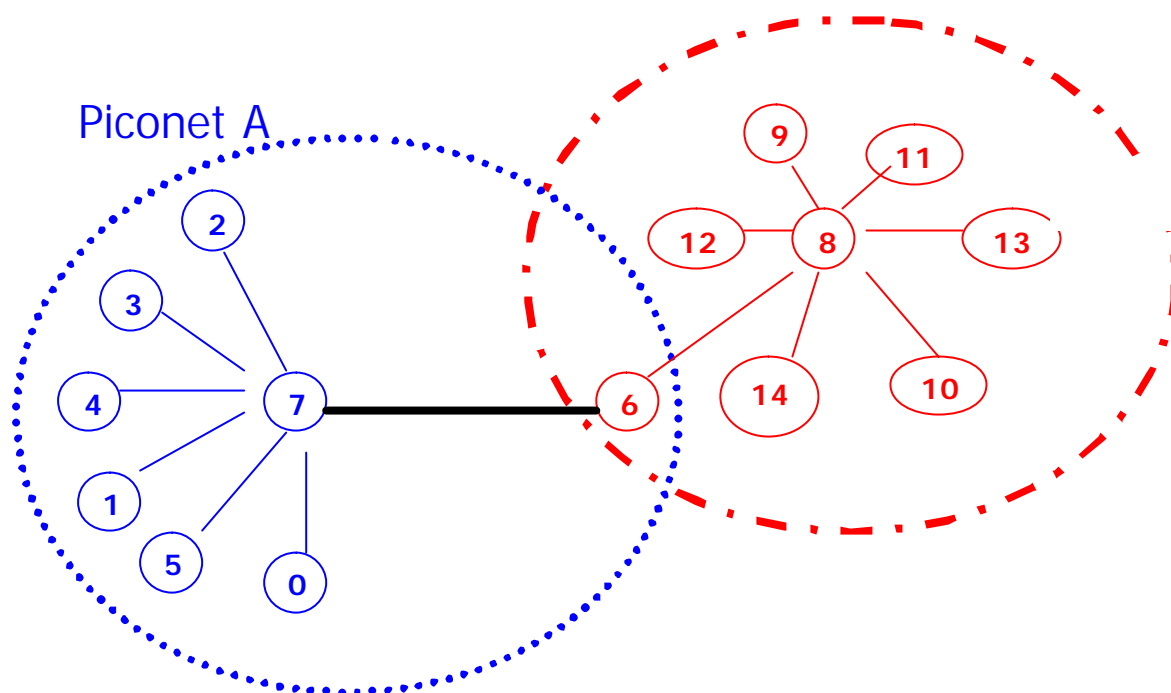


Figura 22 Arquitetura da *scatternet* (BELLÉ, 2002)

Dessa forma, estabeleceu-se uma relação unívoca de gerador de pacotes, na *piconet A*, e de receptor de pacotes, *piconet B*. Todos os *slaves* da *piconet A* devem enviar os pacotes para o seu dispositivo *master*, e este redirecionar os pacotes para o dispositivo *slave* comum as duas *piconets*. A figura 1 mostra a topologia da rede.

Foram adotados diferentes tamanhos de pacotes durante a simulação, com o objetivo de analisar o efeito do tamanho do pacote na latência e *jitter*, além de observar qual tamanho de pacote tem mais chance de ser perdido. Assim foram analisados pacotes de 54, 108, 216, 432, 864 e 1728 Kbytes.

Os pacotes serão enviados um a cada segundo, forçando um *delay* no envio dos pacotes. Um atraso de um segundo pode não representar a realidade num processo de transmissão de curta distância, porém, facilita o processo de análise, melhorando a visualização do comportamento do modelo.



A taxa de transmissão dos pacotes na rede foi a taxa de transmissão do padrão IEEE 802.11b, ou seja, de 2Mb/s, isso porque em simulações de redes *wireless* o NS-2 é configurado para esse padrão de rede, o qual o *Bluetooth* faz parte. ´

Para definir o intervalo de tempo da simulação, foi considerado que todos os pacotes tivessem terminado o processamento, isto é, que todos os pacotes ou tivessem sido recebidos pelos nós *slaves* receptores de tráfego correspondentes, ou então, que o pacote fosse considerado perdido. Assim, optou-se como tempo total da simulação 200 segundos, sendo que o esquema de fila que levou mais tempo para terminar o processamento dos pacotes foi o FQ com 174 segundos e o mais rápido foi o SFQ com demora de apenas 73 segundos. O tempo médio ficou em 134,75 segundos.

Cada nó *slave* enviou durante os 30 segundos iniciais da simulação, 29 pacotes, obedecendo ao sistema totalmente determinístico. No total, foram enviados 174 pacotes, sendo que alguns foram perdidos e outros ficaram na fila durante boa parte do resto do tempo da simulação, resultado das políticas de enfileiramento das filas e do tamanho dos pacotes.

Foi usado para todos os esquemas de fila, o mesmo processo de análise, foram gerados gráficos para visualizar melhor o impacto da latência e do *jitter*, e tabelas para mostrar a perda de pacotes. Nos gráficos, são considerados a latência e o *jitter* de cada pacote, em ordem de recebimento pelo nó *slave* correspondente, ao invés de considerar a variação em relação ao tempo. Assim é possível verificar que cada pacote interfere na latência do pacote que será recebido posteriormente, de uma forma muito acentuada em alguns casos.

## 5 ANÁLISE DOS RESULTADOS

Como dito anteriormente foram feitas simulações envolvendo 4 tipos de filas. O esquema FIFO (*first in first out*), que é o esquema implementado pelo *Host Controller* atualmente (HORSTMAN, 2003), o esquema FQ, o SFQ e o DRR.

O esquema FIFO não implementa nenhum mecanismo de seleção de tráfego ou controle de QoS, assim sendo, durante a simulação, nenhum recurso é utilizado para a seleção de pacotes. A simulação apenas determinou que o tamanho máximo de pacotes na fila é de 30 pacotes. Após a fila atingir esse tamanho, os pacotes são descartados.

No esquema SFQ, foi adotado o valor de enfileiramento padrão do NS-2, que é de no máximo, 30 pacotes. Os esquemas DRR e FQ são configurados com um limite máximo de 50 pacotes, mesmo assim, após a análise, os esquemas DRR e SFQ possuíam resultados muito próximos.

A seguir, são mostrados os resultados das simulações a partir das métricas escolhidas para análise. Começando pela perda de pacotes, seguido pela latência, e por final, o *jitter*. Para cada métrica é analisado o comportamento de cada esquema de fila. Os gráficos e tabelas são mostrados ao longo do texto, pois são necessários para uma melhor compreensão dos resultados.

### 5.1 Perda de Pacotes

A perda de pacotes nesses tipos de filas pode se dar de dois modos.

Os esquemas FIFO e FQ possuem a característica de rejeição de pacotes, em que quando o limite máximo da fila é alcançado, os pacotes que chegam são descartados até que um pacote seja liberado. Quando um pacote é liberado, então uma vaga é aberta e os pacotes que chegam, entram sucessivamente na fila, até que o limite máximo seja alcançado e os pacotes sejam descartados novamente.

Os esquemas SFQ e DRR no entanto, possuem a característica de seleção e eliminação de pacotes, ou seja, quando o limite máximo da fila é alcançado, uma política de exclusão de pacotes seleciona o pacote a ser excluído.

Com esses comportamentos fica garantido que a mobilidade não foi o fator determinante na perda de pacotes, sendo que não houve situações de *handoff* na simulação

e quase todas as perdas de pacotes podem ser explicados por um dos 2 comportamentos acima. Isso não quer dizer que a mobilidade não afeta na perda de pacotes. O que acontece aqui, é que os pacotes maiores (de 1700Kb) tiram praticamente todos os recursos da fila, e provocam situações de congestionamento que contribuem para que os pacotes sejam descartados. A situação da mobilidade apenas pode fazer um pacote chegar um pouco mais rápido ou mais devagar ao seu próximo *hop*, mas decididamente, se houver uma fila congestionada no nó que deve receber o pacote, o fator mobilidade não parece fazer diferença.

### 5.1.1 First in First out (FIFO)

Na simulação FIFO pode se observar que os pacotes vão sendo recebidos corretamente até o instante de tempo 11. A partir desse momento a fila lota e os pacotes que chegam vão sendo descartados. Esse período de descarte dura até o instante 30 da simulação. Como a velocidade de transmissão é grande, os pacotes chegam muito rápido ao nó *master* da primeira *piconet*. Com os pacotes de 1730 Kb contribuindo para o congestionamento, e sendo emitido um pacote de cada tipo por segundo, o limite da fila é alcançado rapidamente. Durante esses 19 segundos ocorrem todas as perdas de pacotes. Os pacotes são perdidos durante o período em que os nós *slaves* estão enviando pacotes. Logo após o término do período de envio, não há mais perda de pacotes.

O principal motivo para isso ocorrer é que não há mais tráfego indo dos nós *slaves* para o *master* da primeira *piconet*. Era nesse nó onde todos os pacotes foram descartados, pois ele recebia pacotes de 6 nós e tinha que rotear todos para o mesmo canal. Na medida que não existe mais pacotes chegando nesse nó, ele pode encaminhar todos os pacotes que estão na fila para o próximo nó. E como todos os pacotes menores chegam ao nó *master* mais rápido, então eles são descartados. Os pacotes de 1730Kb só começam a chegar depois do instante 20, logo, depois que os nós terminam de enviar os pacotes eles continuam chegando. Ao contrário dos pacotes de 54Kb. Na tabela 1 pode ser visualizado o comportamento da simulação.

<i>Perda de Pacotes – First in First out (FIFO)</i>			
Tamanho	Enviados	Perdidos	Recebidos
54	29	14 (48%)	15 (52%)
108	29	12 (41%)	17 (59%)
216	29	13 (45%)	16 (55%)
432	29	14 (48%)	15 (52%)
865	29	9(31%)	20 (69%)
1.730	29	3 (10%)	26 (90%)
	174	65 (37%)	109 (63%)

**Tabela 1 Resultados da simulação FIFO. Limite: 30 pacotes**

### 5.1.2 Stochastic Fair Queueing (SFQ)

O esquema de enfileiramento SFQ funciona criando fluxos para cada tipo de pacote, com base no limite de pacotes de fila. Para implementar esta forma de funcionamento, o esquema SFQ forma filas separadas para cada tipo de fluxo de entrada e no esquema de *round-robin* transfere um pacote de cada fluxo por vez. Os pacotes de cada fluxo são retirados considerando um esquema FIFO (Horstmann 2003). Como o limite máximo é de 30 pacotes e são seis tipos de dados diferentes, são criados então, 6 fluxos de dados dentro das filas. Esses fluxos de dados são filas do esquema FIFO apenas para cada tipo de dado. Assim são criados 6 filas, cada uma com limite máximo de 5 pacotes.

Os pacotes menores chegam no nó *master* mais rapidamente, mas também são encaminhados para a outra *piconet* rapidamente. Os pacotes maiores no instante 9 segundos ainda estão chegando no nó *master* da primeira *piconet*. Logo as filas dos pacotes de 216Kb e 432Kb é a que tem seu limite máximo alcançado mais rapidamente, e começa a descartar os pacotes no instante 9. Logo após, no instante 10, os limites máximos para os fluxos de 54Kb e de 108Kb são alcançados, e os pacotes desses tamanhos também começam a ser descartados. No instante 11 é a vez da fila dos pacotes de 865Kb ter seu limite máximo alcançado. Para o fluxo de dados de 1730Kb isso só acontece aos 15 segundos.

Novamente, a fila permanece cheia somente durante o período de envio dos pacotes dos nós *slaves*, sendo que os últimos descartes ocorrem aos 30 segundos para os fluxos de 54, 108, 216 e 432 Kb. Sempre que um pacote de um dos fluxos é recebido, ele abre vaga para um novo pacote pertencente ao mesmo fluxo. A partir do instante 30, a fila passa

somente a entregar os pacotes e a taxa de entrada/saída na fila se estabiliza. Isso dura até o instante 35. A partir de então, a taxa de entrada praticamente zero e a fila passa somente a encaminhar os pacotes para o *gateway*, que encaminhará para os nós de destino. A tabela 2 mostra o comportamento da simulação.

<i>Perda de Pacotes – Stochastic Fair Queueing (SFQ)</i>			
<b>Tamanho</b>	<b>Enviados</b>	<b>Perdidos</b>	<b>Recebidos</b>
54	29	14 (48%)	15 (52%)
108	29	17 (58%)	12 (42%)
216	29	17 (58%)	12 (42%)
32	29	17 (58%)	12 (42%)
865	29	12 (42%)	17 (58%)
1.730	29	5 (17%)	24 (83%)
	174	82 (47%)	92 (53%)

**Tabela 2 Resultados da simulação SFQ. Limite: 30 pacotes**

### 5.1.3 Fair Queueing (FQ)

Como para a FQ foi definido o tamanho máximo de pacotes na fila do modo padrão do NS-2 (50 pacotes), contra 30 da FIFO, era de se esperar que se passasse mais tempo antes de começar o descarte. Assim, menos pacotes seriam descartados. Isso é comprovado, pois o primeiro descarte ocorre após 17 segundos, 5 segundos depois. Considerando que fila deve estar recebendo 6 pacotes por segundo, são 30 pacotes de diferença, mas ainda deve ser levado em consideração os pacotes que estão saindo da fila durante esse tempo, e que os pacotes de tamanho de 1700Kb só entram na fila do *master* da primeira *piconet* 10 segundos depois de enviados.

Novamente, os pacotes são perdidos somente durante o tempo em que eles estão sendo enviados (17s a 30s). Com isso pode-se concluir que os pacotes estão chegando tão rápido ao *master* da primeira *piconet*, que assim que o fluxo pára, os pacotes param de chegar no *master*, e a fila não enche mais. A tabela 3 mostra os descartes para a simulação da FQ.

<i>Perda de Pacotes – Fair Queueing (FQ)</i>			
Tamanho	Enviados	Perdidos	Recebidos
54	29	10 (34%)	19 (66%)
108	29	10 (34%)	19 (66%)
216	29	11 (38%)	18 (62%)
432	29	9 (31%)	12 (69%)
865	29	5 (17%)	24 (83%)
1.730	29	2 (7%)	27 (93%)
	174	47 (27%)	127 (73%)

**Tabela 3 Resultados da simulação FQ. Limite: 50 pacotes**

#### 5.1.4 Deficit Round Robin (DRR)

O esquema DRR é o que apresenta o maior número de diferenças em relação aos outros esquemas de filas. Os pacotes de tamanho menor não são descartados. Mesmo assim, é o menor índice de perda de pacotes, e as perdas demoram a acontecer e ocorrem também, fora do período de emissão de pacotes.

Como dito acima, o esquema DRR possui a característica de seleção e eliminação de pacotes, ao invés da rejeição a pacotes quando a fila estiver cheia. E a política para selecionar o pacote a ser eliminado consiste em eliminar pelo menos 5% do tamanho da fila em bytes, selecionando o menor número de pacotes possível. Assim chega-se á conclusão de que sempre os pacotes maiores serão descartados para tentar alcançar a janela de 5% rapidamente.

Como o esquema DRR vai alterando em ciclos iguais o controle para recebimento de pacotes, os pacotes menores chegam primeiro, pois demoram menos para serem encaminhados para o *master* da segunda *piconet*. Os pacotes maiores demoram mais que nos outros esquemas para chegarem, pois o DRR demora muito para fazer todos os ciclos para encaminhá-lo. Assim, a quantidade de pacotes de 1700Kb no *master* da primeira *piconet* é sempre grande, e conseqüentemente a probabilidade de um pacote ser selecionado para descarte também.

Assim, o limite em bytes da fila é alcançado aos 21 segundos (o que mais demora a começar a descartar) e um pacote de 1700Kb é descartado. Esse limite continua sendo

alcançado durante vários pontos após o período de emissão de pacotes, sendo que a última vez em que um pacote é descartado é no instante 52, quando um pacote de 860Kb é descartado. A tabela 4 mostra os resultados para o DRR.

<i>Perda de Pacotes – Deficit Round Robin (DRR)</i>			
<b>Tamanho</b>	<b>Enviados</b>	<b>Perdidos</b>	<b>Recebidos</b>
54	29	0 (0%)	29 (100%)
108	29	0 (0%)	29 (100%)
216	29	0 (0%)	29 (100%)
432	29	5 (17%)	24 (83%)
865	29	11 (38%)	18 (62%)
1.730	29	13 (45%)	16 (55%)
	174	29 (17%)	145 (83%)

**Tabela 4 Resultados da simulação DRR. Limite: 30 pacotes**

## 5.2 Latência

A medida da latência teve resultados bastante similares entre os esquemas FIFO e FQ. Os esquemas SFQ e DRR tiveram medidas bem diferentes.

Os gráficos foram feitos considerando os pacotes por ordem de chegada, ou seja, o valor do pacote 5 do gráfico FIFO é diferente do valor 5 do gráfico FQ, a menos que a ordem de recebimento dos pacotes seja idêntica.

Na medida dos esquemas FQ e FIFO os pacotes maiores determinaram a taxa de aumento da latência. No começo, o fluxo de pacotes menores ainda é maior, a latência mantém-se pequena, mas a partir do pacote 17 no FQ e do pacote 19 na FIFO a latência dos pacotes começa a aumentar, com média de 20 segundos.

Algumas vezes, em ambos os esquemas, latência de um pacote aumentar consideravelmente em relação ao anterior, e depois cair. Isso se deve ao recebimento de um pacote de 1700Kb, que permaneceu longo tempo na fila, os pacotes anteriores demoram menos tempo, e os posteriores, mesmo sofrendo os efeitos de estarem atrás de um pacote enorme, possuem uma latência menor. Os gráficos do Anexo A visualizam esses detalhes.

Após todos os pacotes menores terem sido recebidos, ficam no sistema apenas os pacotes de 860 e 1700 Kbytes. Eles podem ser visualizados facilmente a partir do pacote 75 no FIFO e 90 no FQ chegando á 90 e 105 segundos respectivamente. Nessa etapa, a

latência está assustadoramente grande, aumentando muito o tempo em relação ao período em que os pacotes menores estavam no sistema e para cada pacote que chega, sua latência é sempre maior que a do pacote anterior.

Com isso chega-se à conclusão, que quanto maior o pacote, maior a latência que os pacotes sofrerão ao percorrer a rede, afetando inclusive os pacotes menores, pois mesmo antes de os pacotes maiores começarem a serem recebidos, o simples fato de já estarem no sistema contribui para o aumento da latência de todos os pacotes no sistema.

Os dois outros esquemas também sofrem o impacto dos pacotes de 1700 e 860Kb na latência, mas são ajudados pela política de descartes dos esquemas. O esquema SFQ através do controle de fluxos, sendo que quando uma fila de um determinado fluxo alcança o limite, ele é descartado. Isso gera uma taxa de descarte maior que a dos outros esquemas - SFQ é a que possui o maior número de pacotes descartados - o que conseqüentemente irá acarretar na diminuição da latência.

O esquema DRR, através de seu mecanismo de gerenciamento que privilegia os pacotes menores. Assim eles são colocados e retirados da fila antes dos pacotes maiores, o que contribui para que eles não sofram os efeitos causados na latência pelos pacotes maiores.

Para se ter uma idéia de como o descarte de pacotes do esquema SFQ contribui para diminuir a latência basta olhar no gráfico do Anexo A. O último pacote a ser recebido possui latência de 45 segundos, sendo que nos esquemas FQ e FIFO são observados pacotes com latência maior que 100 segundos.

Durante os pacotes 41 a 77 a latência dos pacotes varia muito, aumentando e diminuindo constantemente, ou por um pacote maior que é recebido, ou por pacotes que são descartados. Mesmo assim, a média da latência continua constante. Isso é diferente do que ocorre com os esquemas FIFO e FQ, que mesmo quando um pacote menor é recebido depois de um maior, sua latência em relação aos outros de seu tamanho aumenta. Na medida em que novos fluxos chegam ao sistema, há mudanças nos tempos, mas sempre com a tendência de equilíbrio.

Já no esquema DRR, apesar alguns pacotes também possuírem latência maior que 100 segundos, é interessante observar que este esquema possui o maior número de pacotes com latência abaixo de 40 segundos: 100 pacotes, contra 65 do esquema FIFO, 65 do FQ e 84 do SFQ. Isso sendo que o esquema DRR foi o que menos descartou pacotes e o SFQ foi o que possui a taxa de descartes mais alta.



Essa vantagem do DRR se deve principalmente ao fato de priorizar os pacotes pequenos. Dos 40 primeiros pacotes recebidos, nenhum possui tamanho de 1700Kb. Assim, os pacotes menores são enviados depois, chegam antes e recebem pouca influência de pacotes maiores, que são os últimos a serem recebidos. Quando os pacotes maiores são recebidos, a latência aumenta, mas como quando os pacotes maiores começam a chegar ela ainda está baixa, ela não se torna tão grande quanto à dos esquemas FIFO e FQ.

### 5.3 Jitter

O *jitter*, como citado anteriormente é a diferença da latência de do pacote atual pela latência do pacote anterior, em ordem de recebimento. Assim, para valores de latência que aumentam seu valor tende a ser positivo, porém quando a latência do pacote atual é menor, o valor do *jitter* é negativo. Os gráficos feitos para analisar o *jitter* mostram dessa forma os resultados.

Várias vezes ocorreram valores nulos para o *jitter*, evidenciando que a latência de um pacote em relação ao anterior é nula. O *jitter* nulo é o valor considerado ideal, principalmente para transmissões de dados em tempo real como som e áudio, pois se o *jitter* nessas transmissões variar demais, a transmissão poderá sofrer “lag” (para *jitter* positivo) e um aceleração da transmissão (no caso de repetidos pacotes com *jitter* negativo). Em casos como esse é melhor ter uma latência total um pouco maior, mas com um *jitter* nulo, assim a transmissão vai chegar com um pequeno atraso, mas sem travar ou acelerar.

Com base nas afirmações acima pode-se considerar os resultados como um desastre. Os esquemas FIFO e FQ tiveram um *jitter* extremamente semelhante, sendo que o principal culpado para os resultados negativos é a política de descarte, que descarta pacotes sem nenhum controle, e os pacotes grandes, que possuem uma latência muito alta.

Suponha o seguinte exemplo: um pacote de 108Kb acabou de ser recebido, logo em seguida é recebido um pacote de 1700Kb, que possui uma latência muito maior que a do pacote de 108Kb. Em seguida, um pacote de 54Kb, chega, e o *jitter* cai estrondosamente. Na próxima vez que uma situação do exemplo acontecer, o *jitter* novamente irá explodir, só que como a latência do último pacote de 1700 K é maior que a do primeiro, o valor do *jitter* irá aumentar ainda mais. Mas como a latência do próximo pacote também vai ser maior a queda não vai ser proporcional á primeira. Mesmo assim, isso é um dos fatores que mais contribui para a absurda variação do *jitter*, que começa pequeno, com os pacotes

menores chegando e as suas latências parecidas, mas que da metade dos pacotes recebidos em diante parece um sismógrafo detectando um terremoto de 8.0 graus na escala Richter.

O esquema DRR possui a pior variação de *jitter*, isto é, é o esquema em que o *jitter* varia mais. Essa variação ocorre pelo sistema de descartes do esquema, que seleciona para descarte os pacotes que estão a mais tempo na fila, e que conseqüentemente são os maiores. Assim, livres de alguns pacotes enormes que são descartados, vários pacotes tem sua latência diminuída, o que contribui para variar o *jitter*. Outra curiosidade, é que o *jitter* no esquema DRR possui uma variação maior no instante em que os pacotes estão sendo descartados, entre os recebimentos dos pacotes 55 e 90. Assim, para os pacotes recebidos por último (os maiores), a variação do *jitter* não é grande, diferentemente dos esquemas FIFO e FQ. Assim, a maior variação do *jitter* nos esquemas FIFO e FQ é nos pacotes finais, onde são recebidos alternadamente os pacotes de 1700Kb e 860Kb.

Assim, a melhor variação de *jitter* ficou com o esquema SFQ, o que não quer dizer que ela tenha sido boa. A política de descarte de pacotes foi um fator que influenciou diretamente no valor do *jitter* dos pacotes. Como muitos pacotes são descartados, e o descarte é sempre voltado para tentar manter uma média de latência – *jitter* = 0 – a metade inicial dos pacotes recebidos possui um *jitter* bem baixo. Já a metade final possui variações, mas ainda bem menores que as verificadas nos outros esquemas, com casos máximos de 13 segundos para o esquema SFQ e mais de 20 nos outros esquemas.

Assim pode-se dizer que para equilibrar um *jitter* ideal, seria necessário, banir as diferenças de tamanhos de pacotes e tentar ao máximo manter os fluxos de dados constantes, sem um período onde todos os nós estão enviando e um período onde os dados estão trafegando na rede. Infelizmente, uma simulação nesse molde não faz parte do escopo do trabalho.

## 6 CONCLUSÕES E TRABALHOS FUTUROS

Depois de analisar os resultados, a primeira conclusão que se chega é que usando pacotes com uma diferença de tamanhos não muito grande, todas as métricas seriam melhoradas para todos os esquemas de filas. Porém, o trabalho visa uma simulação de esquemas de filas em uma situação de esgotamento de recursos, com os nós *slaves* da *piconet* em constante movimento, analisado a formação de filas em um ponto de convergência específico do sistema, e como a mobilidade dos nós influencia nesse esgotamento de recursos.

O esquema SFQ por exemplo, teria um desempenho muito melhor com um maior limite de pacotes na fila, ou com uma taxa de chegada de pacotes menor, pois ele atinge o limite de seus fluxos muito rápido, e quando esse limite é atingido um pacote desse fluxo é descartado.

Para o esquema FIFO, um tamanho uniforme dos pacotes traria uma latência média com desvio padrão baixo, o que geraria um *jitter* mais estável, principalmente no final da simulação, onde os pacotes de 1700Kb e 800Kb possuem latências muito diferentes.

O esquema DRR possui uma menor taxa de perda de pacotes. Como muitos pacotes tem que ser enviados, a latência é maior. O esquema SFQ possui uma latência mais adequada, mas em razão de ocorrer um descarte de pacotes muito elevado, o que deixa menos pacotes para serem transmitidos. No total, o esquema SFQ envia 53 pacotes a menos que o esquema DRR.

Vendo como a mobilidade influencia nessa situação de esgotamento de recursos, é necessário ressaltar dois pontos principais. O primeiro é que o software *Network Simulator* não gera nenhuma informação sobre os estados dos nós. Ele acompanha os pacotes, assim, cada vez que um pacote realiza uma ação, como entrar na rede, chegar em algum nó, ou ser descartado, o *Network Simulator* informa, mas não informa onde os nós estão, ou qual o estado dele quando um evento desses ocorre. Mesmo assim, foi tentado verificar a influencia da mobilidade. O segundo é que a mobilidade interfere muito pouco em casos de esgotamento de recursos. Como os nós continuam durante toda a simulação alcançáveis ao *master*, e esse ao *slave* que atua como *gateway* para a outra *scatternet*, não se configura uma situação de possível esgotamento de recursos do mesmo modo que na fila em nenhum

momento da simulação, deixando como fatores principais o tamanho dos pacotes e a política de descarte do esquemas de filas.

Os esquemas FIFO e FQ possuíram valores de latência e *jitter* muito próximos, mesmo tendo uma diferença em relação á perda de pacotes. Considerando as três métricas, o esquema FQ possui uma pequena vantagem, em razão de seu limite máximo da fila ser maior do que o do esquema FIFO. Mesmo assim, nenhum dos dois esquemas consegue um resultado marcante em alguma das três métricas analisadas.

A política de descarte de pacotes influencia diretamente na taxa de perda de pacotes. O esquema DRR foi o que apresentou a menor perda de pacotes e o SFQ a maior. FIFO e FQ tiveram um desempenho mediano. Considerando a perda de pacotes, o resultado mais surpreendente foi do esquema SFQ que teve um descarte muito acentuado. Isso foi devido á taxa de transferência, que fazia os pacotes chegarem muito rápido no nó *master*. Assim os fluxos individuais ficavam lotados muito rapidamente, e quanto mais pacotes são descartados, novos pacotes entram na rede, e se o *gateway* não der conta do trabalho, esses pacotes que entram também são descartados. Isso fez a latência média do esquema SFQ ser a melhor. Sendo que quando o esquema SFQ descarta pacotes, ele também tenta manter a latência constante, o *jitter* do esquema SFQ foi o que menos variou.

O esquema DRR, como já citado, possui a menor taxa de perda de pacotes, mas sua latência é bem mais alta e o *jitter* varia bem mais do que o esquema SFQ. Assim, podemos considerar que se para você é mais importante receber o maior número de pacotes possível, e principalmente o *jitter* não é importante, o esquema DRR é o que possui o melhor resultado nessa situação de falta de recursos. Mas se você considerar mais importante manter uma latência baixa, uma transmissão de pacotes mais rápida e um *jitter* que não varia muito, o esquema SFQ é o melhor.

Assim, pode-se dizer, que o desempenho dos esquemas de filas possui influência direta da taxa de entrada de pacotes no sistema e o tipo de tráfego. Já a mobilidade, não tem tanta influencia no esgotamento de recursos da fila, pois o efeito dos tipos de tráfego nessa simulação possui uma característica de esgotamento de recursos muito forte e que sobrepõe a influencia da mobilidade.

Vale dizer que essas conclusões são válidas somente para essa simulação, pois pode ocorrer de você estar uma outra rede, com o esquema SFQ, e precisar de latência baixa. Mas se não houver descarte de pacotes de uma forma como houve nesta simulação, não é

assegurado que a latência irá se manter baixa em relação aos outros esquemas como se manteve neste trabalho.

A continuidade do trabalho poderia ser realizada avaliando outros esquemas de filas, já citados em (Horstman, 2002), como *Random Early Detection* (RED), *Weighted Random Early Detection* (WRED), e as variações do esquema FQ, *Start time Fair Queuing*, *Self-clocked Fair Queuing*, *Eligible Start time Fair Queuing*, *Smallest Eligible Fair Finishing Time First*. Poderia ser feito também, uma análise de outros tipos de tráfego, com pacotes de tamanhos diferentes dos usados, e condizentes com o tráfego usado na simulação. O mais indicado para esse trabalho seria simular tráfego IP.

Para o problema da mobilidade, seria interessante causar uma situação de esgotamento de recursos para a mobilidade. Dessa forma poderia ser conhecido o impacto que a falta de mobilidade – *handoff* e fora da área de serviço – ou o excesso dela – novas configurações de *piconets* e *scaternets* – causam nas métricas analisadas. O NS-2 não oferece uma visualização geral do sistema durante a simulação, de forma que se houver algum *handoff*, ou houver modificação na arquitetura da rede, o usuário não poderá saber o que aconteceu. Seria necessário então, um outro simulador, ou conseguir equipamentos para fazer um teste real.

Uma outra alternativa seria tentar acrescentar políticas de QoS implementando mecanismos *DiffServs* (*Differentiated Services*) e depois de analisar o impacto desses mecanismos, elaborar um SLA padrão para redes *Bluetooth* já incorporando medidas de QoS e métricas que garantam a QoS para redes *Bluetooth*.

## REFERÊNCIAS

- BEAL, A. et al. *The Future of Wireless: Different than You Think, Bolder than You Imagine*. Disponível em <[http://www.accenture.com/xd/xd.asp?it=enweb&xd=\\_isc/iscworkingpaperabstract\\_134.xml](http://www.accenture.com/xd/xd.asp?it=enweb&xd=_isc/iscworkingpaperabstract_134.xml)> Acesso em: 13 ago. 2003
- BELLÉ, E. *Avaliação de Desempenho em Redes Móveis sem Fio Ad hoc*. Dissertação (Mestrado em Ciências da Computação) Universidade Federal de Santa Catarina, Florianópolis, 2003
- DANTAS, M. A. R. *Tecnologias de Redes de Comunicação e Computadores*. Rio de Janeiro: Editora Axcel Books, 2002
- DIAS, K., SADOK, D. *Internet Móvel: Tecnologias, Aplicações e QoS*. Simpósio Brasileiro de Redes de Computadores, 2001
- D'OLIVEIRA, S. *Análise de Tráfego de Dados em Redes Bluetooth*. Trabalho de Conclusão de Curso (Graduação em Ciências da Computação) Universidade Federal de Pernambuco, Pernambuco, 2001
- Ericsson AG. Disponível em <[http://geein.fclar.unesp.br/atividades/historicos/Ericsson\\_AG.pdf](http://geein.fclar.unesp.br/atividades/historicos/Ericsson_AG.pdf)> Acesso em: 23 nov. 2003
- FOMA P2402. Disponível em <[www.nttdocomo.co.jp/english/fun/fun\\_09.html](http://www.nttdocomo.co.jp/english/fun/fun_09.html)> Acesso em: 8 jun. 2003
- HORSTMANN, G. *Avaliação de Mecanismos para Gerenciamento da Fila da Interface do Host Controller Bluetooth*. Dissertação (Mestrado em Ciências da Computação) Universidade Federal de Santa Catarina, Florianópolis, 2002
- KAMIENSKI, C. et al. *Simulando a Internet: Aplicações na Pesquisa e no Ensino*. Sociedade Brasileira de Computação, 2001
- LIMA, A. *Proposta e Validação de um Mecanismo para Garantir QoS em Redes Sem Fio de Topologia Ad hoc*. Dissertação (Mestrado em Ciências da Computação) Universidade Federal de Santa Catarina, Florianópolis, 2002
- LOUREIRO, A. et al. *Redes de Sensores Sem Fio*. Sociedade Brasileira de Computação, 2001
- LUO, R. et al. *A Method for Self-optimization in Tree-pattern Bluetooth Scaternet*. IM, 2003
- MELO, E. *Qualidade de Serviço em Redes IP com Diffserv: Avaliação Através de Medições*. Dissertação (Mestrado em Ciências da Computação) Universidade Federal de Santa Catarina, Florianópolis, 2001

MILLER, Micheal. *Descobrimdo Bluetooth*. Tradução de Altair Dias Caldas de Moraes e Claudio Belleza Dias. Rio de Janeiro: Editora Campus, 2001

NS2. *Network Simulator*. Disponível em <[www.isi.edu/nsnam/ns](http://www.isi.edu/nsnam/ns)> Acesso em: 5 jun. 2003

PEREIRA, M. *Análise de Desempenho em Redes Wireless Ad-hoc e Estabelecimento de um Acordo de Nível de Serviço Pró-ativo*. Dissertação (Mestrado em Ciências da Computação) Universidade Federal de Santa Catarina, Florianópolis, 2003

SOARES, L. *Redes de Computadores: das LANs, MANs e WANs as Redes ATM*. Rio de Janeiro: Editora Campus, 1995

TANEMBAUM, A. *Redes de Computadores*. Rio de Janeiro: Editora Campus, 1997.

*The Official Bluetooth Website*. Disponível em <<http://www.bluetooth.com>> Acesso em: 3 ago. 2003

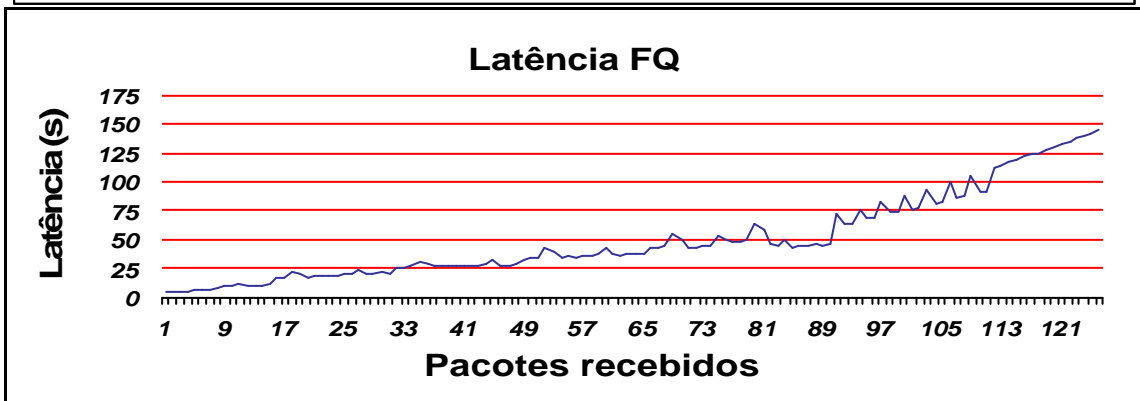
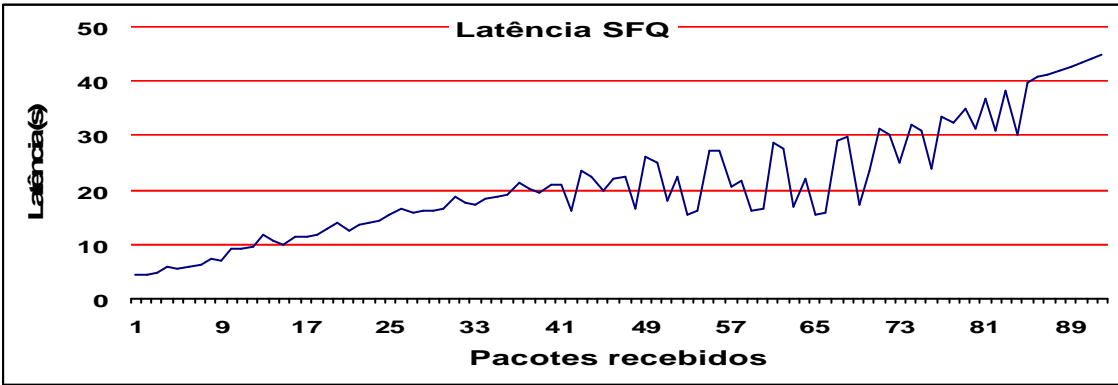
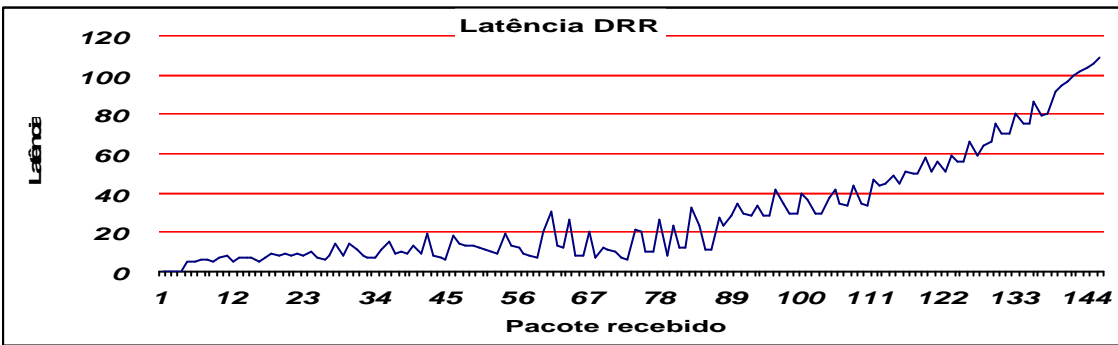
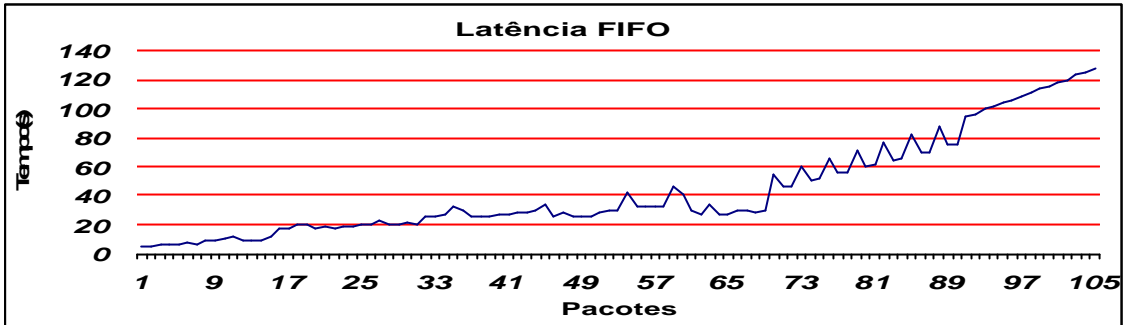
ROSA, H. *Bluetooth*. Disponível em <[http://geocities.yahoo.com.br/wirelessbrasil/bluetooth/bluetooth\\_01.html](http://geocities.yahoo.com.br/wirelessbrasil/bluetooth/bluetooth_01.html)> Acesso em: 18 maio 2003

RIBAS, J. *Perfil de Link Sem Fio em Ambiente Aberto: Avaliação Através de Medições*. Dissertação (Mestrado em Ciências da Computação) Universidade Federal de Santa Catarina, Florianópolis, 2002

# ANEXOS

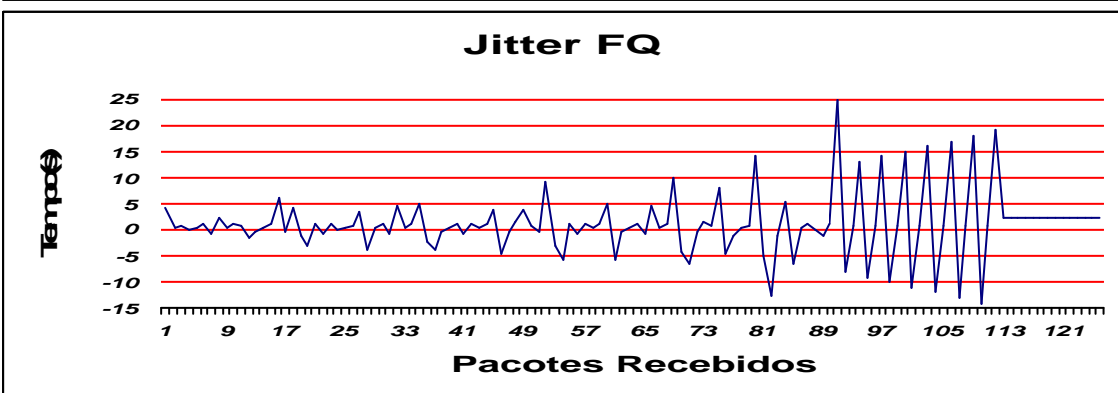
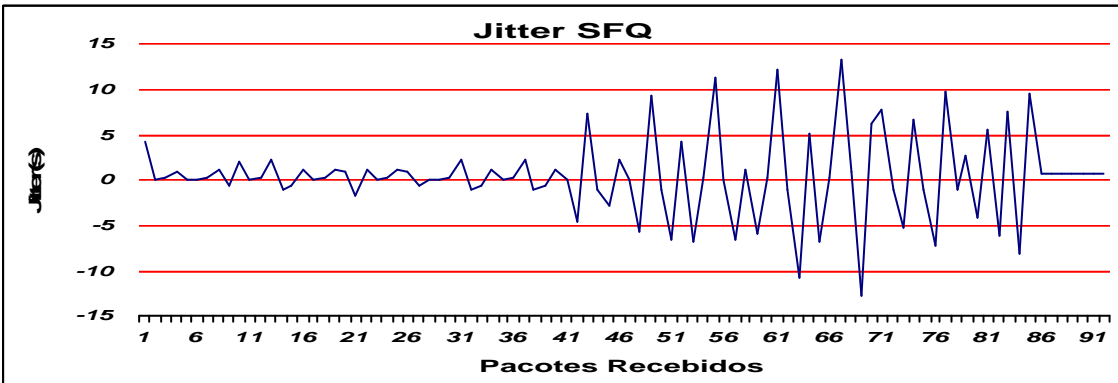
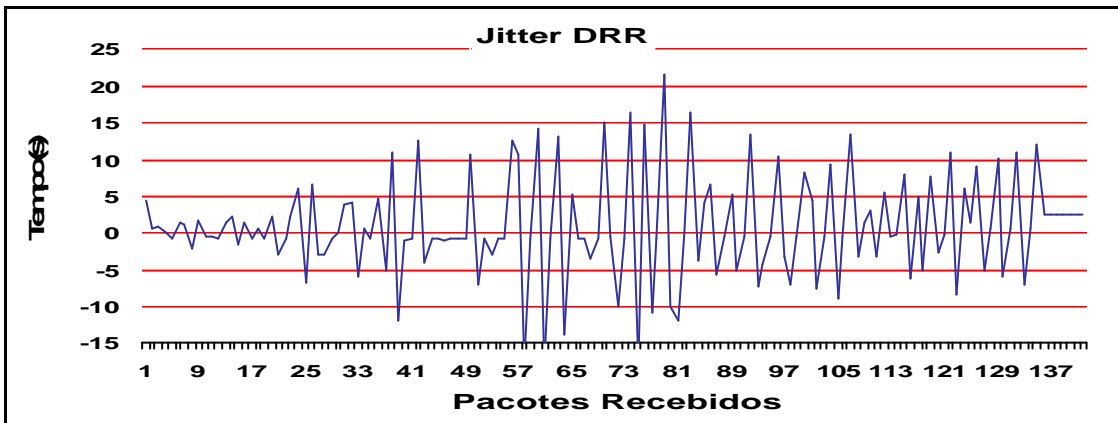
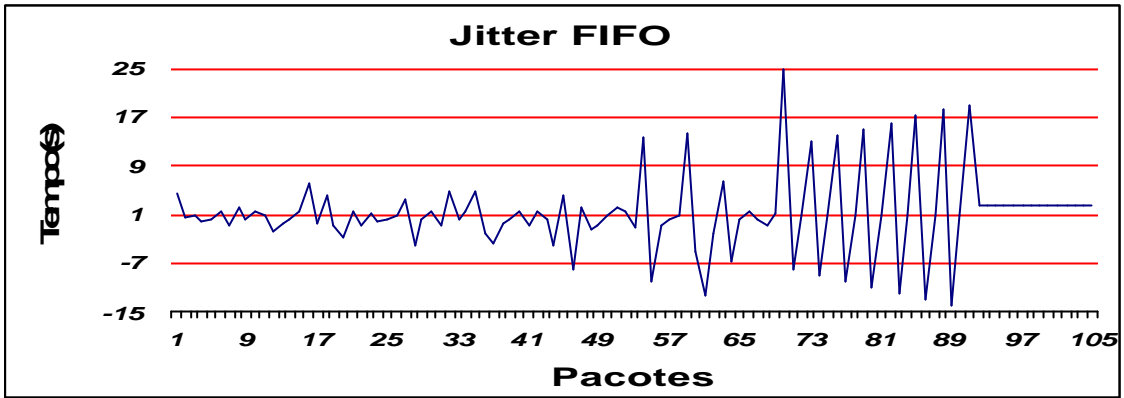
## Anexo A – Gráficos

### Gráficos Latência





## Gráficos Jitter



## Anexo B – Scripts TCL

```
#Simulação usando FIFO
global opt
    set opt(chan)    Channel/WirelessChannel
    set opt(prop)    Propagation/TwoRayGround
    set opt(netif)    Phy/WirelessPhy
    set opt(mac)      Mac/802_11
    set opt(ifq)      Queue/DropTail/PriQueue
    set opt(ll)       LL
    set opt(ant)      Antenna/OmniAntenna
    set opt(x)        200
    set opt(y)        150
    set opt(ifqlen)   30
    #set opt(tr)      wired-and-wireless.tr
    #set opt(namtr)   wired-and-wireless.nam
    set opt(nn)       12
    set opt(adhocRouting) DSDV
    set opt(stop)     50
    set num_wired_nodes 1
    set num_bs_nodes  2

set ns [new Simulator]
# set up for hierarchical routing
$ns node-config -addressType hierarchical
AddrParams set domain_num_ 3
lappend cluster_num 1 1 1
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 13 1
AddrParams set nodes_num_ $eilastlevel

# set tracefd [open $opt(tr) w]
# $ns_ trace-all $tracefd
# set namtracefd [open $opt(namtr) w]
# $ns_ namtrace-all $namtracefd

#
# set file_trace [open TCC_30_FIFO.tr w]
# set file_trace [open TCC_50_FQ.tr w]
# set file_trace [open TCC_30_SFQ.tr w]
# set file_trace [open TCC_50_DRR.tr w]
$ns trace-all $file_trace
#
# set nam_file [open TCC_30_FIFO.nam w]
# set nam_file [open TCC_50_FQ.nam w]
# set nam_file [open TCC_30_SFQ.nam w]
# set nam_file [open TCC_50_DRR.nam w]
```

```

    $ns namtrace-all $nam_file

set topo [new Topography]
$topo load_flatgrid $opt(x) $opt(y)
# god needs to know the number of all wireless interfaces
create-god [expr $opt(nn) + $num_bs_nodes]

#create wired nodes
set temp {0.0.0}
for {set i 0} {$i < $num_wired_nodes} {incr i} {
    set W($i) [$ns node [lindex $temp $i]]
}

$ns node-config -adhocRouting $opt(adhocRouting) \
    -llType $opt(ll) \
    -macType $opt(mac) \
    -ifqType $opt(ifq) \
    -ifqLen $opt(ifqlen) \
    -antType $opt(ant) \
    -propInstance [new $opt(prop)] \
    -phyType $opt(netif) \
    -channel [new $opt(chan)] \
    -topoInstance $topo \
    -wiredRouting ON \
    -agentTrace ON \
    -routerTrace OFF \
    -macTrace OFF

    set temp {1.0.0 1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10 1.0.11
1.0.12}
    set BS(0) [$ns node [lindex $temp 0]]
    set BS(1) [$ns node 2.0.0]
    $BS(0) random-motion 0
    $BS(1) random-motion 0

    $BS(0) set X_ 80.0
    $BS(0) set Y_ 75.0
    $BS(0) set Z_ 0.0

    $BS(1) set X_ 120.0
    $BS(1) set Y_ 75.0
    $BS(1) set Z_ 0.0

#configure for mobilenodes
$ns node-config -wiredRouting OFF

```

```

for {set j 0} {$j < $opt(nn)} {incr j} {
    set node_($j) [ $ns node [lindex $temp \
        [expr $j+1]] ]
    $node_($j) base-station [AddrParams addr2id [$BS(0) node-addr]]
}

$ns duplex-link $W(0) $BS(0) 5Mb 2ms DropTail
$ns duplex-link $W(0) $BS(1) 5Mb 2ms DropTail

set null6 [new Agent/Null]
set null7 [new Agent/Null]
set null8 [new Agent/Null]
set null9 [new Agent/Null]
set null10 [new Agent/Null]
set null11 [new Agent/Null]
$ns attach-agent $node_(6) $null6
$ns attach-agent $node_(7) $null7
$ns attach-agent $node_(8) $null8
$ns attach-agent $node_(9) $null9
$ns attach-agent $node_(10) $null10
$ns attach-agent $node_(11) $null11

set cbr0 [new Agent/CBR]
set cbr1 [new Agent/CBR]
set cbr2 [new Agent/CBR]
set cbr3 [new Agent/CBR]
set cbr4 [new Agent/CBR]
set cbr5 [new Agent/CBR]
#
$ns attach-agent $node_(0) $cbr0
$ns attach-agent $node_(1) $cbr1
$ns attach-agent $node_(2) $cbr2
$ns attach-agent $node_(3) $cbr3
$ns attach-agent $node_(4) $cbr4
$ns attach-agent $node_(5) $cbr5

$cbr0 set packetSize_ 54K
$cbr1 set packetSize_ 108K
$cbr2 set packetSize_ 216K
$cbr3 set packetSize_ 432K
$cbr4 set packetSize_ 865K
$cbr5 set packetSize_ 1730K
#
$cbr0 set interval_ 1s
$cbr1 set interval_ 1s
$cbr2 set interval_ 1s
$cbr3 set interval_ 1s

```

```

    $cbr4 set interval_ 1s
    $cbr5 set interval_ 1s
#
    $ns connect $cbr0 $null6
    $ns connect $cbr1 $null7
    $ns connect $cbr2 $null8
    $ns connect $cbr3 $null9
    $ns connect $cbr4 $null10
    $ns connect $cbr5 $null11
#
#for {set i 0} {$i < $opt(mn)} {incr i} {
#  $ns initial_node_pos $node_($i) 20
# }
    $ns at 1s "$cbr0 start"
    $ns at 1s "$cbr1 start"
    $ns at 1s "$cbr2 start"
    $ns at 1s "$cbr3 start"
    $ns at 1s "$cbr4 start"
    $ns at 1s "$cbr5 start"
#
    $ns at $opt(stop) "$cbr0 stop"
    $ns at $opt(stop) "$cbr1 stop"
    $ns at $opt(stop) "$cbr2 stop"
    $ns at $opt(stop) "$cbr3 stop"
    $ns at $opt(stop) "$cbr4 stop"
    $ns at $opt(stop) "$cbr5 stop"
#
for {set i } {$i < $opt(nn) } {incr i} {
    $ns at $opt(stop).0000010 "$node_($i) reset";
}

$ns at $opt(stop).0000010 "$BS(0) reset";
$ns at $opt(stop).1 "puts \"NS EXITING...\" ; $ns halt"
$ns at 500.0 "finish"
puts "Starting Simulation..."
$ns run

```

```

#Simulação FQ
global opt
set opt(chan) Channel/WirelessChannel
set opt(prop) Propagation/TwoRayGround
set opt(netif) Phy/WirelessPhy
set opt(mac) Mac/802_11

```

```

set opt(ifq)    Queue/FQ
set opt(ll)    LL
set opt(ant)   Antenna/OmniAntenna
set opt(x)     200
set opt(y)     150
set opt(ifqlen) 30
#set opt(tr)   wired-and-wireless.tr
#set opt(namtr) wired-and-wireless.nam
set opt(nn)    12
set opt(adhocRouting) DSDV
set opt(stop)  50
set num_wired_nodes 1
set num_bs_nodes  2

set ns [new Simulator]
# set up for hierarchical routing
$ns node-config -addressType hierarchical
AddrParams set domain_num_ 3
lappend cluster_num 1 1 1
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 13 1
AddrParams set nodes_num_ $eilastlevel

# set tracefd [open $opt(tr) w]
# $ns_ trace-all $tracefd
# set namtracefd [open $opt(namtr) w]
# $ns_ namtrace-all $namtracefd

# set file_trace [open TCC_30_FIFO.tr w]
# set file_trace [open TCC_50_FQ.tr w]
# set file_trace [open TCC_30_SFQ.tr w]
# set file_trace [open TCC_50_DRR.tr w]
$ns trace-all $file_trace

# set nam_file [open TCC_30_FIFO.nam w]
# set nam_file [open TCC_50_FQ.nam w]
# set nam_file [open TCC_30_SFQ.nam w]
# set nam_file [open TCC_50_DRR.nam w]
$ns namtrace-all $nam_file

set topo [new Topography]
$topo load_flatgrid $opt(x) $opt(y)
# god needs to know the number of all wireless interfaces
create-god [expr $opt(nn) + $num_bs_nodes]

#create wired nodes
set temp {0.0.0}
for {set i 0} {$i < $num_wired_nodes} {incr i} {

```

```

set W($i) [$ns node [lindex $temp $i]]
}

$ns node-config -adhocRouting $opt(adhocRouting) \
  -llType $opt(ll) \
  -macType $opt(mac) \
  -ifqType $opt(ifq) \
  -ifqLen $opt(ifqlen) \
  -antType $opt(ant) \
  -propInstance [new $opt(prop)] \
  -phyType $opt(netif) \
  -channel [new $opt(chan)] \
  -topoInstance $topo \
  -wiredRouting ON \
  -agentTrace ON \
  -routerTrace OFF \
  -macTrace OFF

set temp {1.0.0 1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10 1.0.11
1.0.12}
set BS(0) [$ns node [lindex $temp 0]]
set BS(1) [$ns node 2.0.0]
$BS(0) random-motion 0
$BS(1) random-motion 0

$BS(0) set X_ 80.0
$BS(0) set Y_ 75.0
$BS(0) set Z_ 0.0

$BS(1) set X_ 120.0
$BS(1) set Y_ 75.0
$BS(1) set Z_ 0.0

#configure for mobilenodes
$ns node-config -wiredRouting OFF

for {set j 0} {$j < $opt(nn)} {incr j} {
  set node_($j) [ $ns node [lindex $temp \
    [expr $j+1]] ]
  $node_($j) base-station [AddrParams addr2id [$BS(0) node-addr]]
}

$ns duplex-link $W(0) $BS(0) 5Mb 2ms FQ
$ns duplex-link $W(0) $BS(1) 5Mb 2ms FQ

set null6 [new Agent/Null]

```

```

set null7 [new Agent/Null]
set null8 [new Agent/Null]
set null9 [new Agent/Null]
set null10 [new Agent/Null]
set null11 [new Agent/Null]
$ns attach-agent $node_(6) $null6
$ns attach-agent $node_(7) $null7
$ns attach-agent $node_(8) $null8
$ns attach-agent $node_(9) $null9
$ns attach-agent $node_(10) $null10
$ns attach-agent $node_(11) $null11

set cbr0 [new Agent/CBR]
set cbr1 [new Agent/CBR]
set cbr2 [new Agent/CBR]
set cbr3 [new Agent/CBR]
set cbr4 [new Agent/CBR]
set cbr5 [new Agent/CBR]
#
$ns attach-agent $node_(0) $cbr0
$ns attach-agent $node_(1) $cbr1
$ns attach-agent $node_(2) $cbr2
$ns attach-agent $node_(3) $cbr3
$ns attach-agent $node_(4) $cbr4
$ns attach-agent $node_(5) $cbr5

$cbr0 set packetSize_ 54K
$cbr1 set packetSize_ 108K
$cbr2 set packetSize_ 216K
$cbr3 set packetSize_ 432K
$cbr4 set packetSize_ 865K
$cbr5 set packetSize_ 1730K
#
$cbr0 set interval_ 1s
$cbr1 set interval_ 1s
$cbr2 set interval_ 1s
$cbr3 set interval_ 1s
$cbr4 set interval_ 1s
$cbr5 set interval_ 1s
#
$ns connect $cbr0 $null6
$ns connect $cbr1 $null7
$ns connect $cbr2 $null8
$ns connect $cbr3 $null9
$ns connect $cbr4 $null10
$ns connect $cbr5 $null11
#

```



```

#for {set i 0} {$i < $opt(nn)} {incr i} {
#  $ns initial_node_pos $node_($i) 20
# }
    $ns at 1s "$cbr0 start"
    $ns at 1s "$cbr1 start"
    $ns at 1s "$cbr2 start"
    $ns at 1s "$cbr3 start"
    $ns at 1s "$cbr4 start"
    $ns at 1s "$cbr5 start"
#
    $ns at $opt(stop) "$cbr0 stop"
    $ns at $opt(stop) "$cbr1 stop"
    $ns at $opt(stop) "$cbr2 stop"
    $ns at $opt(stop) "$cbr3 stop"
    $ns at $opt(stop) "$cbr4 stop"
    $ns at $opt(stop) "$cbr5 stop"
#
for {set i } {$i < $opt(nn) } {incr i} {
    $ns at $opt(stop).0000010 "$node_($i) reset";
}

$ns at $opt(stop).0000010 "$BS(0) reset";
$ns at $opt(stop).1 "puts \"NS EXITING...\" ; $ns halt"
$ns at 500.0 "finish"
puts "Starting Simulation..."
$ns run

```

```

#Simulação SFQ
global opt
set opt(chan)    Channel/WirelessChannel
set opt(prop)    Propagation/TwoRayGround
set opt(netif)   Phy/WirelessPhy
set opt(mac)     Mac/802_11
set opt(ifq)     Queue/SFQ
set opt(ll)      LL
set opt(ant)     Antenna/OmniAntenna
set opt(x)       200
set opt(y)       150
set opt(ifqlen)  30
#set opt(tr)     wired-and-wireless.tr
#set opt(namtr)  wired-and-wireless.nam
set opt(nn)      12
set opt(adhocRouting) DSDV

```

```

set opt(stop)      50
set num_wired_nodes 1
set num_bs_nodes   2

set ns [new Simulator]
# set up for hierarchical routing
$ns node-config -addressType hierarchical
AddrParams set domain_num_ 3
lappend cluster_num 1 1 1
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 13 1
AddrParams set nodes_num_ $eilastlevel

# set tracefd [open $opt(tr) w]
# $ns_ trace-all $tracefd
# set namtracefd [open $opt(namtr) w]
# $ns_ namtrace-all $namtracefd

# set file_trace [open TCC_30_FIFO.tr w]
# set file_trace [open TCC_50_FQ.tr w]
# set file_trace [open TCC_30_SFQ.tr w]
# set file_trace [open TCC_50_DRR.tr w]
$ns trace-all $file_trace

#
# set nam_file [open TCC_30_FIFO.nam w]
# set nam_file [open TCC_50_FQ.nam w]
# set nam_file [open TCC_30_SFQ.nam w]
# set nam_file [open TCC_50_DRR.nam w]
$ns namtrace-all $nam_file

set topo [new Topography]
$topo load_flatgrid $opt(x) $opt(y)
# god needs to know the number of all wireless interfaces
create-god [expr $opt(nn) + $num_bs_nodes]

#create wired nodes
set temp {0.0.0}
for {set i 0} {$i < $num_wired_nodes} {incr i} {
    set W($i) [$ns node [lindex $temp $i]]
}

$ns node-config -adhocRouting $opt(adhocRouting)\
    -llType $opt(ll) \
    -macType $opt(mac) \
    -ifqType $opt(ifq) \
    -ifqLen $opt(ifqlen) \
    -antType $opt(ant) \
    -propInstance [new $opt(prop)] \

```

```

-phyType $opt(netif) \
-channel [new $opt(chan)] \
-topoInstance $topo \
-wiredRouting ON \
-agentTrace ON \
-routerTrace OFF \
-macTrace OFF

set temp { 1.0.0 1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10 1.0.11
1.0.12}
set BS(0) [$ns node [lindex $temp 0]]
set BS(1) [$ns node 2.0.0]
$BS(0) random-motion 0
$BS(1) random-motion 0

$BS(0) set X_ 80.0
$BS(0) set Y_ 75.0
$BS(0) set Z_ 0.0

$BS(1) set X_ 120.0
$BS(1) set Y_ 75.0
$BS(1) set Z_ 0.0

#configure for mobilenodes
$ns node-config -wiredRouting OFF

for {set j 0} {$j < $opt(nn)} {incr j} {
    set node_($j) [ $ns node [lindex $temp \
        [expr $j+1]] ]
$node_($j) base-station [AddrParams addr2id [$BS(0) node-addr]]
}

$ns duplex-link $W(0) $BS(0) 5Mb 2ms SFQ
$ns duplex-link $W(0) $BS(1) 5Mb 2ms SFQ

set null6 [new Agent/Null]
set null7 [new Agent/Null]
set null8 [new Agent/Null]
set null9 [new Agent/Null]
set null10 [new Agent/Null]
set null11 [new Agent/Null]
$ns attach-agent $node_(6) $null6
$ns attach-agent $node_(7) $null7
$ns attach-agent $node_(8) $null8
$ns attach-agent $node_(9) $null9
$ns attach-agent $node_(10) $null10
$ns attach-agent $node_(11) $null11

```

```

set cbr0 [new Agent/CBR]
set cbr1 [new Agent/CBR]
set cbr2 [new Agent/CBR]
set cbr3 [new Agent/CBR]
set cbr4 [new Agent/CBR]
set cbr5 [new Agent/CBR]
#
$ns attach-agent $node_(0) $cbr0
$ns attach-agent $node_(1) $cbr1
$ns attach-agent $node_(2) $cbr2
$ns attach-agent $node_(3) $cbr3
$ns attach-agent $node_(4) $cbr4
$ns attach-agent $node_(5) $cbr5

$cbr0 set packetSize_ 54K
$cbr1 set packetSize_ 108K
$cbr2 set packetSize_ 216K
$cbr3 set packetSize_ 432K
$cbr4 set packetSize_ 865K
$cbr5 set packetSize_ 1730K
#
$cbr0 set interval_ 1s
$cbr1 set interval_ 1s
$cbr2 set interval_ 1s
$cbr3 set interval_ 1s
$cbr4 set interval_ 1s
$cbr5 set interval_ 1s
#
$ns connect $cbr0 $null6
$ns connect $cbr1 $null7
$ns connect $cbr2 $null8
$ns connect $cbr3 $null9
$ns connect $cbr4 $null10
$ns connect $cbr5 $null11
#
#for {set i 0} {$i < $opt(nn)} {incr i} {
#  $ns initial_node_pos $node_($i) 20
# }
$ns at 1s "$cbr0 start"
$ns at 1s "$cbr1 start"
$ns at 1s "$cbr2 start"
$ns at 1s "$cbr3 start"
$ns at 1s "$cbr4 start"
$ns at 1s "$cbr5 start"
#
$ns at $opt(stop) "$cbr0 stop"

```

```

    $ns at $opt(stop) "$cbr1 stop"
    $ns at $opt(stop) "$cbr2 stop"
    $ns at $opt(stop) "$cbr3 stop"
    $ns at $opt(stop) "$cbr4 stop"
    $ns at $opt(stop) "$cbr5 stop"
#
for {set i } {$i < $opt(m) } {incr i} {
    $ns at $opt(stop).0000010 "$node_($i) reset";
}

$ns at $opt(stop).0000010 "$BS(0) reset";
$ns at $opt(stop).1 "puts \"NS EXITING...\" ; $ns halt"
$ns at 500.0 "finish"
puts "Starting Simulation..."
$ns run

```

```

#Simulação DRR
global opt
set opt(chan) Channel/WirelessChannel
set opt(prop) Propagation/TwoRayGround
set opt(netif) Phy/WirelessPhy
set opt(mac) Mac/802_11
set opt(ifq) Queue/DRR
set opt(ll) LL
set opt(ant) Antenna/OmniAntenna
set opt(x) 200
set opt(y) 150
set opt(ifqlen) 30
#set opt(tr) wired-and-wireless.tr
#set opt(namtr) wired-and-wireless.nam
set opt(nn) 12
set opt(adhocRouting) DSDV
set opt(stop) 50
set num_wired_nodes 1
set num_bs_nodes 2

```

```

set ns [new Simulator]
# set up for hierarchical routing
$ns node-config -addressType hierarchical
AddrParams set domain_num_ 3
lappend cluster_num 1 1 1
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 13 1
AddrParams set nodes_num_ $eilastlevel

```

```

# set tracefd [open $opt(tr) w]
# $ns_ trace-all $tracefd
# set namtracefd [open $opt(namtr) w]
# $ns_ namtrace-all $namtracefd

# set file_trace [open TCC_30_FIFO.tr w]
# set file_trace [open TCC_50_FQ.tr w]
# set file_trace [open TCC_30_SFQ.tr w]
# set file_trace [open TCC_50_DRR.tr w]
$ns trace-all $file_trace

# set nam_file [open TCC_30_FIFO.nam w]
# set nam_file [open TCC_50_FQ.nam w]
# set nam_file [open TCC_30_SFQ.nam w]
# set nam_file [open TCC_50_DRR.nam w]
$ns namtrace-all $nam_file

set topo [new Topography]
$topo load_flatgrid $opt(x) $opt(y)
# god needs to know the number of all wireless interfaces
create-god [expr $opt(nn) + $num_bs_nodes]

#create wired nodes
set temp {0.0.0}
for {set i 0} {$i < $num_wired_nodes} {incr i} {
    set W($i) [$ns node [lindex $temp $i]]
}

$ns node-config -adhocRouting $opt(adhocRouting) \
    -llType $opt(ll) \
    -macType $opt(mac) \
    -ifqType $opt(ifq) \
    -ifqLen $opt(ifqlen) \
    -antType $opt(ant) \
    -propInstance [new $opt(prop)] \
    -phyType $opt(netif) \
    -channel [new $opt(chan)] \
    -topoInstance $topo \
    -wiredRouting ON \
    -agentTrace ON \
    -routerTrace OFF \
    -macTrace OFF

set temp {1.0.0 1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10 1.0.11
1.0.12}

```

```

set BS(0) [$ns node [lindex $temp 0]]
set BS(1) [$ns node 2.0.0]
$BS(0) random-motion 0
$BS(1) random-motion 0

$BS(0) set X_ 80.0
$BS(0) set Y_ 75.0
$BS(0) set Z_ 0.0

$BS(1) set X_ 120.0
$BS(1) set Y_ 75.0
$BS(1) set Z_ 0.0

#configure for mobilenodes
$ns node-config -wiredRouting OFF

for {set j 0} {$j < $opt(nn)} {incr j} {
    set node_($j) [ $ns node [lindex $temp \
        [expr $j+1]] ]
    $node_($j) base-station [AddrParams addr2id [$BS(0) node-addr]]
}

$ns duplex-link $W(0) $BS(0) 5Mb 2ms DRR
$ns duplex-link $W(0) $BS(1) 5Mb 2ms DRR

set null6 [new Agent/Null]
set null7 [new Agent/Null]
set null8 [new Agent/Null]
set null9 [new Agent/Null]
set null10 [new Agent/Null]
set null11 [new Agent/Null]
$ns attach-agent $node_(6) $null6
$ns attach-agent $node_(7) $null7
$ns attach-agent $node_(8) $null8
$ns attach-agent $node_(9) $null9
$ns attach-agent $node_(10) $null10
$ns attach-agent $node_(11) $null11

set cbr0 [new Agent/CBR]
set cbr1 [new Agent/CBR]
set cbr2 [new Agent/CBR]
set cbr3 [new Agent/CBR]
set cbr4 [new Agent/CBR]
set cbr5 [new Agent/CBR]

#
$ns attach-agent $node_(0) $cbr0
$ns attach-agent $node_(1) $cbr1

```

```

$ns attach-agent $node_(2) $cbr2
$ns attach-agent $node_(3) $cbr3
$ns attach-agent $node_(4) $cbr4
$ns attach-agent $node_(5) $cbr5

$cbr0 set packetSize_ 54K
$cbr1 set packetSize_ 108K
$cbr2 set packetSize_ 216K
$cbr3 set packetSize_ 432K
$cbr4 set packetSize_ 865K
$cbr5 set packetSize_ 1730K

#
$cbr0 set interval_ 1s
$cbr1 set interval_ 1s
$cbr2 set interval_ 1s
$cbr3 set interval_ 1s
$cbr4 set interval_ 1s
$cbr5 set interval_ 1s

#
$ns connect $cbr0 $null6
$ns connect $cbr1 $null7
$ns connect $cbr2 $null8
$ns connect $cbr3 $null9
$ns connect $cbr4 $null10
$ns connect $cbr5 $null11

#
#for {set i 0} {$i < $opt(nn)} {incr i} {
#  $ns initial_node_pos $node_($i) 20
# }
$ns at 1s "$cbr0 start"
$ns at 1s "$cbr1 start"
$ns at 1s "$cbr2 start"
$ns at 1s "$cbr3 start"
$ns at 1s "$cbr4 start"
$ns at 1s "$cbr5 start"

#
$ns at $opt(stop) "$cbr0 stop"
$ns at $opt(stop) "$cbr1 stop"
$ns at $opt(stop) "$cbr2 stop"
$ns at $opt(stop) "$cbr3 stop"
$ns at $opt(stop) "$cbr4 stop"
$ns at $opt(stop) "$cbr5 stop"

#
for {set i } {$i < $opt(nn) } {incr i} {
  $ns at $opt(stop).0000010 "$node_($i) reset";
}

```



```
$ns at $opt(stop).0000010 "$BS(0) reset";  
$ns at $opt(stop).1 "puts \"NS EXITING...\" ; $ns halt"  
$ns at 500.0 "finish"  
puts "Starting Simulation..."  
$ns run
```

## Anexo C – Artigo

# Análise de Parâmetros de QoS e Esquemas de Filas em um Ambiente Bluetooth

Rafael Krüger Tavares, Michel Ribas Lobato, Carlos Becker Westphall

Departamento de Informática e Estatística – Universidade Federal de Santa Catarina  
(UFSC)

Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brazil

{lobato,twu,westphal}@inf.ufsc.br

**Abstract.** *The development of information technology increases everyday. The wireless networks technologies are already a reality. The crescent market demand for wireless devices such as laptops, cell phones and handhelds, makes the search for guarantee of quality of service be the focus of research to great research centers over the world. This paper presents a QoS analysis of Bluetooth networks through queue schemas alternative to FIFO, comparing the performance of the QoS parameters with the mobility of the network nodes.*

**Resumo.** *O desenvolvimento da tecnologia da informação cresce cada vez mais nos dias de hoje. O advento das redes de computadores sem fio já é uma realidade. A crescente demanda do mercado por dispositivos sem fio como laptops, telefones celulares e handhelds, faz com que a busca por garantia da qualidade de serviço(QoS) se torne o foco de grandes centros de pesquisa mundiais. Este artigo analisa parâmetros de QoS para redes Bluetooth em esquemas de fila alternativos à FIFO, que não garante QoS, relacionando o desempenho dos parâmetros de QoS com a mobilidade dos nós na rede.*

## 1. Introdução

Devido ao atual crescimento do segmento de computadores pessoais portáteis, estima-se que em 2005, o mercado para dispositivos portáteis sem fio crescerá cerca de 630% [Beal, Beck, Lynch, 2001]. Portanto, será bastante comum as pessoas possuírem algum tipo de dispositivo portátil como celulares, *palmtops*, *notebooks* e PDA's com capacidade de se comunicar com a parte fixa da rede e até com outros computadores móveis.

O problema principal é que um dos principais recursos que podem ser usados para garantir QoS, que é uma fila para seleção e buferização de tráfego, não é usado corretamente, sendo que no *Bluetooth*, essa buferização é implementada em um esquema FIFO (*First In First Out*). O esquema FIFO, não oferece nenhuma garantia de QoS, sendo que quando a fila torna-se congestionada, os pacotes que tentam entrar na fila são descartados. Outro problema que pode ser analisado é quanto à mobilidade. Como o *Bluetooth* é uma tecnologia móvel, os recursos de QoS que são usados em redes cabeadas, podem ter funcionamento diferente em situações de mobilidade.

Assim, o trabalho se propõe a utilizar esquemas de filas que garantam QoS em redes cabeadas e compará-los com o esquema FIFO em situações que levam em consideração o efeito que a mobilidade causa nos parâmetros de qualidade de serviço.

Para avaliar os parâmetros de QoS, será usado um simulador de redes. Foi decidido usar o simulador, pois através dele, pode ser simulada uma rede maior, com mais recursos, e mais parâmetros podem ser analisados, sendo impossível fazer o mesmo trabalho a partir de uma rede real, sendo que o primeiro

empecilho seria a ausência total de dispositivos *wireless*. Sendo assim, o simulador escolhido foi o *Network Simulator 2*, que é parte do projeto VINT (*Virtual Internetwork Testbed*), desenvolvido pelo ICS (*Information Sciences Institute*) da *University of Southern California* (NS, 2003).

Este artigo apresenta na seção 2, uma menção a todos os aspectos da tecnologia *Bluetooth*, suas configurações como a *piconet* e a *scatternet*. Para introduzir o assunto um pouco da história da tecnologia é discutida. Na seção 3 são apresentados conceitos de QoS e dos esquemas de filas. A seção 4 apresenta o modelo e os resultados das simulações. Finalmente, a seção 5 apresenta a conclusão e os trabalhos futuros.

## 2. Bluetooth

O *Bluetooth* é um microchip de padrão aberto e global para transmissões via rádio, que provê conexões simples e sem fios entre PCs, telefones móveis, impressoras e muitos outros dispositivos, permitindo também que esses dispositivos tenham acesso a recursos da rede

Através do uso da transmissão de rádio, a transferência de voz e dos dados realiza-se em tempo real. A modalidade de transmissão sofisticada adotada na especificação do *Bluetooth* assegura a proteção contra interferências e a segurança dos dados. O sistema de rádio do *Bluetooth* é construído em um microchip pequeno e opera em uma faixa de frequência de disponibilidade global assegurando a compatibilidade de uma comunicação em nível mundial.

### 2.1. Arquitetura da Tecnologia Bluetooth

A arquitetura do *Bluetooth* é baseada no padrão IEEE 802.11 que define uma rede sem fio do tipo *ad hoc*, e é implementada em seu núcleo, o “*Bluetooth Core*”. Essa arquitetura permite a interconexão entre os módulos *Bluetooth* independente da necessidade de uma estação de ponto de acesso.

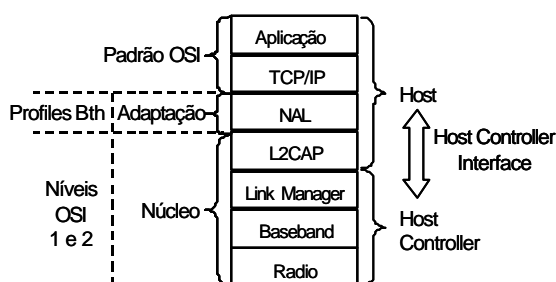
O modelo do *Bluetooth* foi especificado de forma que as suas camadas fossem dispostas em uma hierarquia bastante funcional. As camadas inferiores são responsáveis pela interconexão física dos dispositivos e pelo controle da conexão entre as unidades *Bluetooth*, enquanto que as camadas superiores são responsáveis pelos protocolos de aplicação e interoperabilidade entre diferentes protocolos, ou seja, o *Bluetooth* garante a interação com protocolos que não estejam implementados na sua especificação.

Em comparação com o modelo OSI, o *Bluetooth* implementa as camadas 1 e 2 do modelo. As outras camadas são implementadas em software.

O *Bluetooth Core* ou núcleo *Bluetooth*, engloba as camadas de Radio, *Baseband*, *Link Manager*, L2CAP e a interface controladora do *host* que é chamada de *Host Controller Interface*. Com relação às camadas superiores, temos a camada NAL (*Network Adaptation Layer*), ou camada de adaptação, que junto com as camadas do modelo OSI, são implementadas em software no *host*.

A interface de controlador de *host* só é necessária quando o L2CAP estiver embutido no software [Horstmann, 2002]. Dependendo do hardware, o L2CAP já pode estar incorporado no mesmo. Assim, o L2CAP, o *Link Manager* e o *Baseband* comunicam-se diretamente, o que elimina a necessidade do *Host Controller*.

Figura 1. As arquiteturas OSI e *Bluetooth* (Fonte: [Horstmann, 2002])



A figura 1 mostra uma visão mais detalhada da arquitetura *Bluetooth*, que por abranger vários tipos de protocolos também é chamada de pilha de protocolos *Bluetooth*. A pilha de protocolos *Bluetooth* é composta pelas seguintes camadas:

#### 2.1.1. Bluetooth Radio

É a base da pilha. Esta camada é responsável pela interface de conexão física da tecnologia. Nela estão implementados os mecanismos de transmissão por radiofrequência dos dispositivos Bluetooth.

### **2.1.2. Baseband**

Esta camada é responsável pelo controle do Bluetooth Radio, provendo os saltos de frequência utilizados na transmissão FHSS. Ela é responsável pela interface física do canal de frequência do Bluetooth. É nesta camada que são estabelecidas as conexões de voz e de dados dos dispositivos Bluetooth.

### **2.1.3. Link Manager Protocol (LMP)**

Este nível atende pelo processo de conexão entre os dispositivos Bluetooth, através de funcionalidades para anexar e desanexar dispositivos operando como *slaves* e regras de chaveamento entre um *master* e um *slave* para estabelecimento de conexões.

### **2.1.4. Logical Link Control and Adaptation Protocol (L2CAP)**

Depois que a conexão entre dois ou mais dispositivos Bluetooth foi estabelecida, esta camada procura servir às aplicações dos níveis de protocolos superiores através de algumas funcionalidades, como multiplexação, segmentação e QoS.

### **2.1.5. Host Controller Interface (HCI)**

Com o objetivo de interconectar diferentes padrões de conexão de dispositivos, o Bluetooth pode ser conectado em outros periféricos através de conexões, como por exemplo, um cartão PCMCIA para *notebooks*. Estas conexões, geralmente implementam os níveis mais baixos da especificação Bluetooth. Para que ocorram essas conexões, são necessários um *driver* que precisa ser instalado no Host, e um Host Controller Interface para o Bluetooth aceitar dados sobre o meio físico. Assim, o HCI acaba funcionando como uma interface entre o nível L2CAP e o *Baseband*, permitindo ao Host, pelo L2CAP, enviar comandos para o *Baseband*, e o *Baseband* enviar eventos para o Host.

## **3. QoS e Esquemas de Filas**

Embora não exista uma única expressão que a defina, pode-se dizer que para a área de redes Qualidade de Serviço ou QoS, diz respeito à habilidade que uma rede tem de prover melhor serviço para um determinado tráfego em tecnologias como: IP, *Frame-Relay*, ATM, *Ethernet* com o padrão IEEE 802.11 e outras [Melo, 2001].

Assim, um dos desafios principais no projeto destes sistemas móveis é o adequado fornecimento da qualidade de serviço (QoS) [Lima, 2001], que é representada através de parâmetros da rede, tais como: *throughput*, retardo de trânsito, taxa de erros residuais, proteção, prioridade, latência, vazão, *jitter* entre outros.

Os parâmetros de QoS podem ser solicitados tanto por usuários quanto por aplicações. Aplicações e usuários geram diferentes tipos de tráfego e requisitam diferentes parâmetros de QoS, assim sendo, uma rede deve estar preparada para atender a todos estes requisitos através da alocação consciente de seus recursos. Os principais requisitos de aplicações e usuários em uma rede são latência, *jitter*, perda de pacotes e vazão, destes, apenas este último não é considerado na análise.

### **3.1. Esquemas de Filas**

Filas e algoritmos de tratamento de filas são elementos críticos no gerenciamento de tráfego, essenciais na implantação da qualidade de serviço (QoS). Os algoritmos de tratamento de filas envolvem esquemas para medição de estado de utilização da fila, policiamento de tipos de tráfego e mecanismos de descarte de pacotes. Quando as filas são formadas, algoritmos de tratamento de filas determinam a ordem e a taxa em que o tráfego é retirado da fila para envio ao próximo dispositivo [Horstmann, 2002]. As filas que são usadas na simulação são a FIFO, que é implementada no Bluetooth, SFQ (*Stochastic Fair Queueing*), FQ (*Fair Queueing*) e DRR (*Deficit Round Robin*).

#### **3.1.1. FIFO (First In First Out)**

Este tipo de fila é o mais popular, sendo utilizado tanto no escopo das redes de computadores quanto dos sistemas operacionais, através do escalonamento de processos. Neste esquema, os pacotes vão sendo atendidos a medida em que chegam na fila, ou seja, o FIFO não altera a ordem dos pacotes enfileirados. Embora esse mecanismo possa privilegiar uma aplicação em detrimento da outra ele não garante o controle do serviço sobre o fluxo de tráfego.

### 3.1.2. Stochastic Fair Queueing (SFQ)

Esta disciplina de filas utiliza um esquema de *hash* junto com várias filas para colocar os pacotes que entram em suas filas correspondentes. O número de filas é relativamente menor que o número possível de fluxos. Os fluxos vão sendo direcionados para as filas, sendo que os que estiverem na mesma fila têm tratamento equivalente. A desvantagem deste esquema é que se um fluxo mais importante será desfavorecido se cair na mesma fila de um fluxo de menor prioridade.

### 3.1.3. Deficit Round Robin (DRR)

O DRR funciona utilizando um “*token*” para controlar o fluxo de pacotes. Através de um ciclo, o *token* vai passando de fila em fila. Em cada ciclo as filas recebem um número fixo de créditos, assim, elas só podem liberar os pacotes se o tamanho do pacote for menor ou igual ao número de créditos da fila. Quando um pacote é liberado, o crédito da fila é reduzido de forma equivalente. Essas operações são executadas na medida em que o *token* vai passando pelas filas e ocorrem de forma ordenada. Para organizar os fluxos nas filas, é utilizado o esquema SFQ. Esse mecanismo provê, para alguns fluxos, um controle mais eficiente com relação a garantias de envio e latência.

### 3.1.4. Fair Queueing (FQ)

Neste esquema, é atribuído para cada pacote um tempo teórico de finalização. O algoritmo retira pacotes da fila quando houver capacidade no canal. Pacotes com tempo de finalização menores são enviados antes de pacotes com tempo de finalização maiores.

## 4. Simulações e resultados

### 4.1. Cenário da Simulação

Cada piconet é formada por seis dispositivos *slaves* e um *master*. O sétimo *slave* é membro das duas *piconets* agindo como *gateway* e tornado as duas *piconets* uma *scatternet*.

Dessa forma, estabeleceu-se uma relação unívoca de gerador de pacotes, na *piconet* A, e de receptor de pacotes, *piconet* B. Todos os *slaves* da *piconet* A devem enviar os pacotes para o seu dispositivo *master*, e este redirecionar os pacotes para o dispositivo *slave* comum as duas *piconets*. Os seis *slaves* de cada *piconet* estão em constante movimento. Os outros três nodos não se movem para manter a topologia da rede. Se os dispositivos *master* estivessem em movimento, eles poderiam sair desta condição, e nesse momento toda a topologia da rede seria modificada, podendo ser formada até mais *piconets*, e os pacotes estabelecerem uma rota alternativa, prejudicando assim o processo de análise. Com os três nós principais parados, é possível estabelecer a rota forçando os pacotes a passarem sempre por esses nós, independente da localização dos outros *slaves*.

Foram adotados diferentes tamanhos de pacotes durante a simulação, com o objetivo de analisar o efeito do tamanho do pacote na latência e *jitter*, além de observar qual tamanho de pacote tem mais chance de ser perdido. Assim foram analisados pacotes de 54, 108, 216, 432, 864 e 1728 bytes. A taxa de transmissão dos pacotes na rede foi a taxa de transmissão do padrão IEE 802.11b, ou seja, de 2Mb/s, isso porque em simulações de redes *wireless* o NS-2 é configurado para esse padrão de rede, o qual o *Bluetooth* faz parte.

### 4.2. Resultados das Simulações

Foram feitas simulações envolvendo 4 tipos de filas. O esquema FIFO (*first in first out*), que é o esquema implementado pelo *Host Controller* atualmente [Horstmann, 2002], o esquema FQ, o SFQ e o DRR.

O esquema FIFO não implementa nenhum mecanismo de seleção de tráfego ou controle de QoS, assim sendo, durante a simulação, nenhum recurso é utilizado para a seleção de pacotes. A simulação apenas determinou que o tamanho máximo de pacotes na fila é de 30 pacotes. Após a fila atingir esse tamanho, os pacotes são descartados.

No esquema SFQ, foi adotado o valor de enfileiramento padrão do NS-2, que é de no máximo, 30 pacotes. Os esquemas DRR e FQ são configurados com um limite máximo de 50 pacotes, mesmo assim, após a análise, os esquemas DRR e SFQ possuíam resultados muito próximos.

#### 4.2.1 Perda de Pacotes

Os esquemas FIFO e FQ possuem a característica de rejeição de pacotes, em que quando o limite máximo da fila é alcançado, os pacotes que chegam são descartados até que um pacote seja liberado. Quando um pacote é

liberado, então uma vaga é aberta e os pacotes que chegam, entram sucessivamente na fila, até que o limite máximo seja alcançado e os pacotes sejam descartados novamente.

Os esquemas SFQ e DRR no entanto, possuem a característica de seleção e eliminação de pacotes, ou seja, quando o limite máximo da fila é alcançado, uma política de exclusão de pacotes seleciona o pacote a ser excluído.

Com esses comportamentos fica garantido que a mobilidade não foi o fator determinante na perda de pacotes, sendo que não houve situações de *handoff* na simulação e quase todas as perdas de pacotes podem ser explicadas por um dos 2 comportamentos acima. Isso não quer dizer que a mobilidade não afeta na perda de pacotes. O que acontece aqui, é que os pacotes maiores (de 1700Kb) tiram praticamente todos os recursos da fila, e provocam situações de congestionamento que contribuem para que os pacotes sejam descartados. A situação da mobilidade apenas pode fazer um pacote chegar um pouco mais rápido ou mais devagar ao seu próximo *hop*, mas decididamente, se houver uma fila congestionada no nó que deve receber o pacote, o fator mobilidade não parece fazer diferença.

### ***First In First Out (FIFO)***

Na simulação FIFO pode se observar que os pacotes vão sendo recebidos corretamente até o instante de tempo 11. A partir desse momento a fila lota e os pacotes que chegam vão sendo descartados. Esse período de descarte dura até o instante 30 da simulação. Como a velocidade de transmissão é grande, os pacotes chegam muito rápido ao nó Master da primeira *piconet*. Com os pacotes de 1730 Kb contribuindo para o congestionamento, e sendo emitido um pacote de cada tipo por segundo, o limite da fila é alcançado rapidamente. Durante esses 19 segundos ocorrem todas as perdas de pacotes. Os pacotes são perdidos durante o período em que os nós *slaves* estão enviando pacotes. Logo após o término do período de envio, não há mais perda de pacotes.

**Tabela 1 – Resultados da simulação – FIFO – limite: 30 pacotes**

<b>Perda de Pacotes – <i>First in First out (FIFO)</i></b>			
<b>Tamanho</b>	<b>Enviados</b>	<b>Perdidos</b>	<b>Recebidos</b>
54	29	14 (48%)	15 (52%)
108	29	12 (41%)	17 (59%)
216	29	13 (45%)	16 (55%)
432	29	14 (48%)	15 (52%)
865	29	9(31%)	20 (69%)
1.730	29	3 (10%)	26 (90%)
	174	65 (37%)	109 (63%)

O principal motivo para isso ocorrer é que não há mais tráfego indo dos nós *slaves* para o Master da primeira *piconet*. Era nesse nó onde todos os pacotes foram descartados, pois ele recebia pacotes de 6 nós e tinha que rotear todos para o mesmo canal. Na medida que não existe mais pacotes chegando nesse nó, ele pode encaminhar todos os pacotes que estão na fila para o próximo nó. E como todos os pacotes menores chegam ao nó *master* mais rápido, então eles são descartados primeiro. Na tabela 1 pode ser visualizado o comportamento da simulação.

### ***Stochastic Fair Queueing – SFQ***

A forma de funcionamento do esquema SFQ é implementada formando filas separadas para cada tipo de fluxo de entrada e no esquema de *round-robin* transfere um pacote de cada fluxo por vez. Os pacotes de cada fluxo são retirados considerando um esquema FIFO [Horstmann 2002]. Como são seis tipos de dados diferentes, são criados então, 6 fluxos de dados dentro das filas. Esses fluxos de dados são filas do esquema FIFO apenas para cada tipo de dado. Assim são criados 6 filas, cada uma com limite máximo de 5 pacotes.

Os pacotes maiores no instante nove segundos ainda estão chegando no nó *master* da primeira *piconet*. Logo as filas dos pacotes de 216Kb e 432Kb é a que tem seu limite máximo alcançado mais rapidamente, e começa a descartar os pacotes no instante 9. Logo após, no instante 10, os limites máximos para os fluxos de 54Kb e de 108Kb são alcançados, e os pacotes desses tamanhos também começam a ser descartados. No instante 11 é a vez da fila dos pacotes de 865Kb ter seu limite máximo alcançado. Para o fluxo de dados de 1730Kb isso só acontece aos quinze segundos.

Novamente, a fila permanece cheia somente durante o período de envio dos pacotes dos nós *slaves*, sendo que os últimos descartes ocorrem aos 30 segundos para os fluxos de 54, 108, 216 e 432 Kb. Sempre que um pacote de um dos fluxos é recebido, ele abre vaga para um novo pacote pertencente ao mesmo fluxo. A partir do instante 30, a fila passa somente a entregar os pacotes e a taxa de entrada/saída na fila se estabiliza. A tabela 2 mostra o comportamento da simulação.

**Tabela 2 – Resultados da simulação – SFQ – limite: 30 pacotes**

<b>Perda de Pacotes – Stochastic Fair Queueing (SFQ)</b>			
<b>Tamanho</b>	<b>Enviados</b>	<b>Perdidos</b>	<b>Recebidos</b>
54	29	14 (48%)	15 (52%)
108	29	17 (58%)	12 (42%)
216	29	17 (58%)	12 (42%)
432	29	17 (58%)	12 (42%)
865	29	12 (42%)	17 (58%)
1.730	29	5 (17%)	24 (83%)
	174	82 (47%)	92 (53%)

### ***Fair Queueing – FQ***

Como para a FQ foi definido o tamanho máximo de pacotes na fila do modo padrão do ns-2 (50 pacotes), contra 30 da FIFO, era de se esperar que passasse mais tempo antes de começar o descarte, e menos pacotes são descartados. Isso é comprovado, pois o primeiro descarte ocorre após 17 segundos, 5 segundos depois. Considerando que fila deve estar recebendo 6 pacotes por segundo, são 30 pacotes de diferença, mas ainda devem ser levados em consideração os pacotes que estão saindo da fila durante esse tempo, e que os pacotes de tamanho de 1700Kb só entram na fila do *master* da primeira *piconet* 10 segundos depois de enviados. A tabela 3 mostra os descartes para a simulação da FQ.

Novamente, os pacotes são perdidos somente durante o tempo em que eles estão sendo enviados (17s a 30s). Com isso pode-se concluir que os pacotes estão chegando tão rápido ao *master* da primeira *piconet*, que assim que o fluxo pára, os pacotes param de chegar no *master*, e a fila enche mais

**Tabela 3 – Resultados da simulação – FQ – limite: 50 pacotes**

<b>Perda de Pacotes – Fair Queueing (FQ)</b>			
<b>Tamanho</b>	<b>Enviados</b>	<b>Perdidos</b>	<b>Recebidos</b>
54	29	10 (34%)	19 (66%)
108	29	10 (34%)	19 (66%)
216	29	11 (38%)	18 (62%)
432	29	9 (31%)	12 (69%)
865	29	5 (17%)	24 (83%)
1.730	29	2 (7%)	27 (93%)
	174	47 (27%)	127 (73%)

### ***Deficit Round Robin - (DRR)***

O esquema DRR é o que apresenta o maior número de diferenças em relação aos outros esquemas de filas. Os pacotes de tamanho menor não são descartados. É o menor índice de perda de pacotes, e as perdas demoram a acontecer.

Como dito acima, o esquema DRR possui a característica de seleção e eliminação de pacotes, ao invés da rejeição a pacotes quando a fila estiver cheia. E a política para selecionar o pacote a ser eliminado consiste em eliminar pelo menos 5% do tamanho da fila em bytes, selecionando o menor número de pacotes possível. Assim chega-se á conclusão de que sempre os pacotes maiores serão descartados para tentar alcançar a janela de 5% rapidamente. A tabela 4 mostra os resultados para o esquema DRR.

O principal motivo para isso ocorrer é que não há mais tráfego indo dos nós slaves para o Master da primeira piconet. Era nesse nó onde todos os pacotes foram descartados, pois ele recebia pacotes de 6 nós e tinha que rotear todos para o mesmo canal. Na medida que não existe mais pacotes chegando nesse nó, ele pode encaminhar todos os pacotes que estão na fila para o próximo nó. E como todos os pacotes menores chegam ao nó *master* mais rápido, então eles são descartados primeiro.

**Tabela 1.4 – Resultados da simulação – DRR – limite: 30 pacotes**

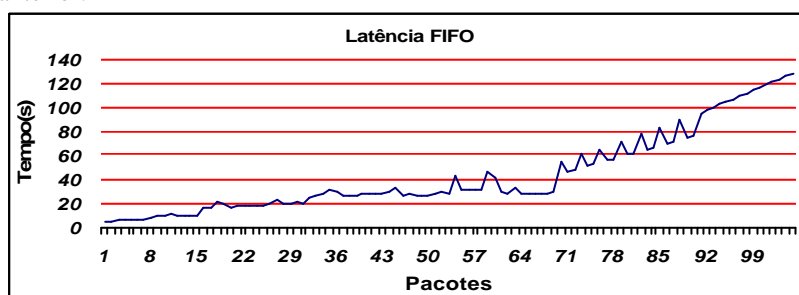
Perda de Pacotes – Deficit Round Robin (DRR)			
Tamanho	Enviados	Perdidos	Recebidos
54	29	0 (0%)	29 (100%)
108	29	0 (0%)	29 (100%)
216	29	0 (0%)	29 (100%)
432	29	5 (17%)	24 (83%)
865	29	11 (38%)	18 (62%)
1.730	29	13 (45%)	16 (55%)
	174	29 (17%)	145 (83%)

#### 4.2.2. Latência

A medida da latência teve resultados bastante similares entre os esquemas FIFO e FQ. Os esquemas SFQ e DRR tiveram medidas bem diferentes. Os gráficos foram feitos considerando os pacotes por ordem de chegada.

Na medida dos esquemas FQ e FIFO os pacotes maiores determinaram a taxa de aumento da latência. No começo, o fluxo de pacotes menores ainda é maior, a latência mantém-se pequena, mas a partir do pacote 17 no FQ e do pacote 19 na FIFO a latência dos pacotes começa a aumentar, com média de 20 segundos. A figura 2 mostra o gráfico da latência para o esquema FIFO e a figura 3 para o FQ.

Após todos os pacotes menores terem sido recebidos, ficam no sistema apenas os pacotes de 860 e 1700 Kbytes Nessa etapa, a latência está assustadoramente grande, aumentando muito o tempo em relação ao período em que os pacotes menores estavam no sistema e para cada pacote que chega, sua latência é sempre maior que a do anterior.

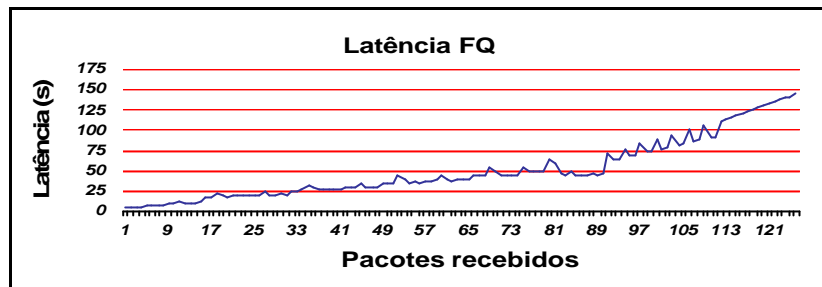


**Figura 2. Desempenho FIFO - Latência**

Os dois outros esquemas também sofrem o impacto dos pacotes de 1700 e 860Kb na latência, mas são ajudados pela política de descartes dos esquemas. O esquema SFQ através do controle de fluxos, sendo que quando uma fila de um determinado fluxo alcança o limite, ele é descartado. Isso gera uma taxa de descarte maior que a dos outros esquemas - SFQ é a que possui o maior número de pacotes descartados - o que consequentemente irá acarretar na diminuição da latência.

O esquema DRR, através de seu mecanismo de gerenciamento que privilegia os pacotes menores. Assim eles são colocados e retirados da fila antes dos pacotes maiores, o que contribui para que eles não sofram os efeitos causados na latência pelos pacotes maiores.



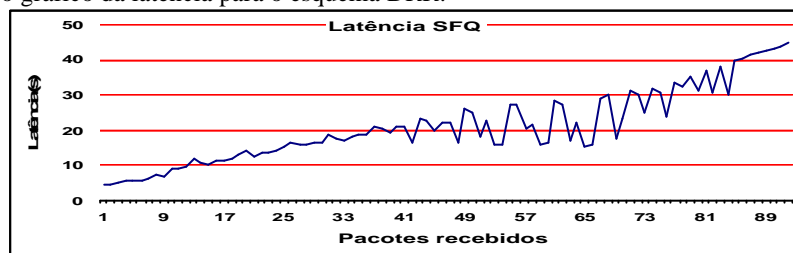


**Figura 3. Desempenho FQ - Latência**

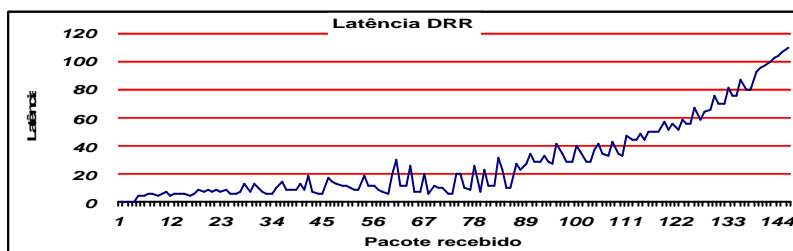
Para se ter uma idéia de como o descarte de pacotes do esquema SFQ contribui para diminuir a latência basta olhar no gráfico. O último pacote a ser recebido possui latência de 45 segundos, sendo que nos esquemas FQ e FIFO são observados pacotes com latência maior que 100 segundos. A figura 4 mostra o gráfico da latência para o esquema SFQ.

Já no esquema DRR, apesar alguns pacotes também possuem latência maior que 100 segundos, é interessante observar que este esquema possui o maior número de pacotes com latência abaixo de 40 segundos: 100 pacotes, contra 65 do esquema FIFO, 65 do FQ e 84 do SFQ. Isso sendo que o esquema DRR foi o que menos descartou pacotes e o SFQ foi o que possui a taxa de descartes mais alta.

Essa vantagem do DRR se deve principalmente ao fato de priorizar os pacotes pequenos. Dos 40 primeiros pacotes recebidos, nenhum possui tamanho de 1700Kb. Assim, os pacotes menores são enviados depois, chegam antes e recebem pouca influência de pacotes maiores, que são os últimos a serem recebidos. Quando os pacotes maiores são recebidos, a latência aumenta, mas como quando os pacotes maiores começam a chegar ela ainda está baixa, ela não se torna tão grande quanto a dos esquemas FIFO e FQ. A figura 5 mostra o gráfico da latência para o esquema DRR.



**Figura 4. Desempenho SFQ - Latência**



**Figura 5. Desempenho DRR - Latência**

### 4.2.3. Jitter

Para valores de latência que aumentam o valor do *jitter* tende a ser positivo, porém quando a latência do pacote atual é menor, seu valor é negativo. Várias vezes ocorreram valores nulos para o *jitter*, evidenciando que a latência de um pacote em relação ao anterior é nula. O *jitter* nulo é o valor considerado ideal, principalmente para transmissões de dados em tempo real como som e áudio, pois se o *jitter* nessas transmissões variar demais, a transmissão poderá sofrer “lag” (para *jitter* positivo) e um aceleração da transmissão (no caso de repetidos pacotes com *jitter* negativo). Em casos como esse é melhor ter uma latência total um pouco maior, mas com um *jitter* nulo, assim a transmissão vai chegar com um pequeno atraso, mas sem travar ou acelerar.

Com base nas afirmações acima podem-se considerar os resultados como um desastre. Os esquemas FIFO e FQ tiveram um *jitter* extremamente semelhante, sendo que o principal culpado para os resultados negativos é a política de descarte, que descarta pacotes sem nenhum controle, e os pacotes grandes, que possuem uma latência muito alta. A figura 6 mostra o *jitter* para o esquema FIFO e a figura 7 para o FQ.

O esquema DRR possui a pior variação de *jitter*, isto é, é o esquema em que o *jitter* varia mais. Essa variação ocorre pelo sistema de descartes do esquema, que seleciona para descarte os pacotes que estão a mais tempo na fila, e que conseqüentemente são os maiores. Assim, livres de alguns pacotes enormes que são descartados, vários pacotes tem sua latência diminuída, o que contribui para variar o *jitter*. Outra curiosidade, é que o *jitter* no esquema DRR possui uma variação maior no instante em que os pacotes estão sendo descartados, entre os recebimentos dos pacotes 55 e 90. Assim, para os pacotes recebidos por último (os maiores), a variação do *jitter* não é grande, diferentemente dos esquemas FIFO e FQ. A figura 8 mostra o gráfico do *jitter* para o esquema DRR.

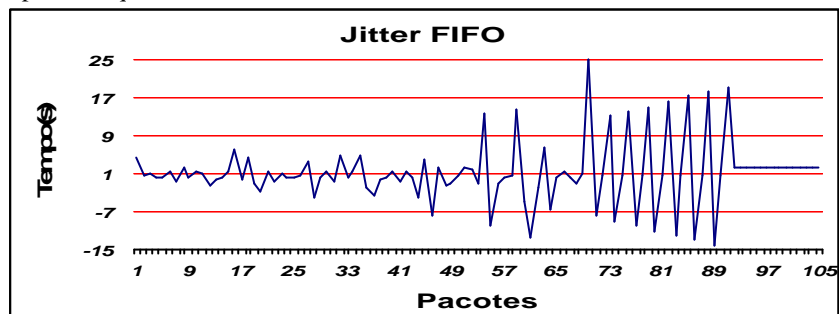


Figura 6. Desempenho FIFO – *Jitter*

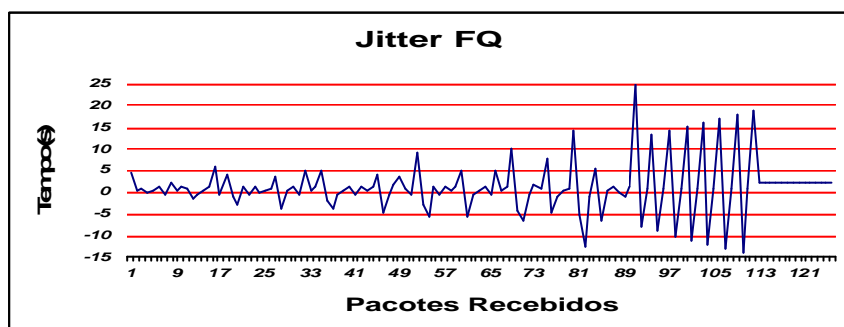


Figura 7. Desempenho FQ - *Jitter*

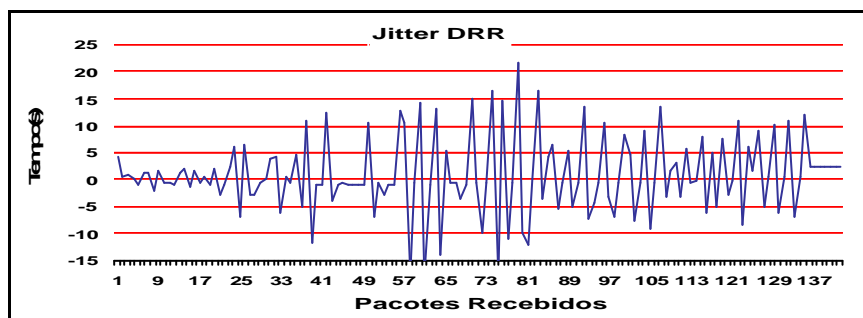
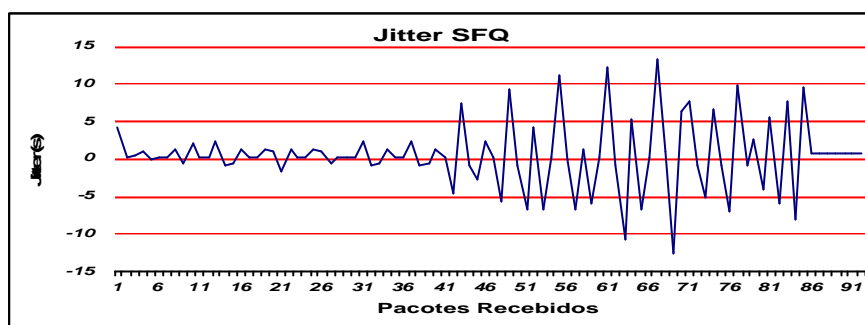


Figura 8. Desempenho DRR – *Jitter*



**Figura 9. Desempenho SFQ - Jitter**

A melhor variação de *jitter* ficou com o esquema SFQ, o que não quer dizer que ela tenha sido boa. A política de descarte de pacotes foi um fator que influenciou diretamente no valor do *jitter* dos pacotes. Com o descarte é sempre voltado para tentar manter uma média de latência a metade inicial dos pacotes recebidos possui um *jitter* bem baixo. Já a metade final possui algumas variações, mas bem menores que as verificadas nos outros esquemas.

## 5. Conclusão

No final, uma conclusão é certa: com pacotes com uma diferença de tamanhos não muito grande, todas as métricas seriam melhoradas para todos os esquemas de filas. Porém, o trabalho visa uma simulação de esquemas de filas em uma situação de esgotamento de recursos, com os nós slaves da piconet em constante movimento, analisado a formação de filas em um ponto de convergência específico do sistema, e como a mobilidade dos nós influencia nesse esgotamento de recursos.

Vendo como a mobilidade influencia nessa situação de esgotamento de recursos, é necessário ressaltar dois pontos principais. O primeiro é que o software Network Simulator não gera nenhuma informação sobre os estados dos nós. Ele acompanha os pacotes, assim, cada vez que um pacote realiza uma ação, como entrar na rede, chegar em algum nó, ou ser descartado, o Network Simulator informa, mas não informa onde os nós estão, ou qual o estado dele quando um evento desses ocorre. Mesmo assim, foi tentado verificar a influencia da mobilidade. O segundo é que a mobilidade interfere muito pouco em casos de esgotamento de recursos. Como os nós continuam durante toda a simulação alcançáveis ao master, e esse ao slave que atua como *gateway* para a outra *scatternet*, não se configura uma situação de possível esgotamento de recursos do mesmo modo que na fila em nenhum momento da simulação, deixando como fatores principais o tamanho dos pacotes e a política de descarte do esquemas de filas.

Os esquemas FIFO e FQ possuíam valores de latência e *jitter* muito próximos, mesmo tendo uma diferença em relação á perda de pacotes. Considerando as três métricas, o esquema FQ possui uma pequena vantagem, em razão de seu limite máximo da fila ser maior do que o do esquema FIFO. Mesmo assim, nenhum dos dois esquemas consegue um resultado marcante em alguma das três métricas analisadas.

A política de descarte de pacotes influencia diretamente na taxa de perda de pacotes. O esquema DRR foi o que apresentou a menor perda de pacotes e o SFQ a maior. FIFO e FQ tiveram um desempenho mediano. Considerando a perda de pacotes, o resultado mais surpreendente foi do esquema SFQ que teve um descarte muito acentuado. Isso foi devido á taxa de transferência, que fazia os pacotes chegarem muito rápido no nó master. Assim os fluxos individuais ficavam lotados muito rapidamente, e quanto mais pacotes são descartados, novos pacotes entram na rede, e se o *gateway* não der conta do trabalho, esses pacotes que entram também são descartados. Isso fez a latência média do esquema SFQ ser a melhor. Sendo que quando o esquema SFQ descarta pacotes, ele também tenta manter a latência constante, o *jitter* do esquema SFQ foi o que menos variou.

O esquema DRR, como já citado, possui a menor taxa de perda de pacotes, mas sua latência é bem mais alta e o *jitter* varia bem mais do que o esquema SFQ. Assim, podemos considerar que se para você é mais importante receber o maior número de pacotes possível, e principalmente o *jitter* não é importante, o esquema DRR é o que possui o melhor resultado nessa situação de falta de recursos. Mas se você considerar mais importante manter uma latência baixa, uma transmissão de pacotes mais rápida e um *jitter* que não varia muito, o esquema SFQ é o melhor.

Assim, pode-se dizer, que o desempenho dos esquemas de filas possui influência direta da taxa de entrada de pacotes no sistema e o tipo de tráfego. Já a mobilidade, não tem tanta influencia no esgotamento

de recursos da fila, pois o efeito dos tipos de tráfego nessa simulação possui uma característica de esgotamento de recursos muito forte e que sobrepõe a influencia da mobilidade.

### 5.1 Trabalhos Futuros

A continuidade do trabalho poderia ser realizada avaliando outros esquemas de filas, já citados em Horstman (2002), como *Random Early Detection* (RED), *Weighted Random Early Detection* (WRED), e as variações do esquema FQ, *Start time Fair Queuing*, *Self-clocked Fair Queuing*, *Eligible Start time Fair Queuing*, *Smallest Eligible Fair Finishing Time First*. Poderia ser feito também, uma análise de outros tipos de tráfego, com pacotes de tamanhos diferentes dos usados, e condizentes com o tráfego usado na simulação. O mais indicado para esse trabalho seria simular tráfego IP.

Para o problema da mobilidade, seria interessante causar uma situação de esgotamento de recursos para a mobilidade. Dessa forma poderia ser conhecido o impacto que a falta de mobilidade – *handoff* e fora da área de serviço – ou o excesso dela – novas configurações de *piconets* e *scaternets* – causam nas métricas analisadas. O NS-2 não oferece uma visualização geral do sistema durante a simulação, de forma que se houver algum *handoff*, ou houver modificação na arquitetura da rede, o usuário não poderá saber o que aconteceu. Seria necessário então, um outro simulador, ou conseguir equipamentos para fazer um teste real.

## 6. Referências

- Beal, A., Beck, J and Lynck P. (2001) “The Future of Wireless Different than you think, bolder than you imagine”, Disponível em [http://www.accenture.com/xd/xd.asp?it=enweb&xd=\\_isc/iscworkingpaperabstract\\_134.xml](http://www.accenture.com/xd/xd.asp?it=enweb&xd=_isc/iscworkingpaperabstract_134.xml), Agosto.
- Horstmann, G. (2002) “Avaliação de Mecanismos para Gerenciamento da Fila da Interface do Host Controller Bluetooth”, Dissertação de Mestrado em Ciências da Computação, UFSC, Florianópolis.
- Melo, E. (2001) “Qualidade de Serviço em Redes IP com *Diffserv*: Avaliação Através de Medições”, Dissertação de Mestrado em Ciências da Computação, UFSC, Florianópolis.
- Lima, A. (2001) “Proposta e Validação de um Mecanismo para Garantir QoS em Redes Sem Fio de Topologia *ad hoc*”, Dissertação de Mestrado em Ciências da Computação, UFSC, Florianópolis. *Network Simulator*, <http://www.isi.edu/nsnam/>, Agosto.