UNIVERSIDADE FEDERAL FLUMINENSE
INSTITUTO DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Mauricio Manoel Ribeiro

César de Souza Bouças

Utilização de Redes Mesh para Visualização de Exames Médicos

Niterói
2007

Mauricio Manoel Ribeiro

César de Souza Bouças

Utilização de Redes Mesh para Visualização de Exames Médicos

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense como parte dos requisitos para obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: Célio Vinicius Neves de Albuquerque
Co-orientador: Robson Hilario da Silva

Niterói
2007

Mauricio Manoel Ribeiro

César de Souza Bouças

Utilização de Redes Mesh para Visualização de Exames Médicos

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense como parte dos requisitos para obtenção do Grau de Bacharel em Ciência da Computação.

Aprovado em julho de 2007.

BANCA EXAMINADORA

Prof. CÉLIO VINICIUS NEVES DE ALBUQUERQUE, Ph.D.
Orientador
UFF

Prof. Robson Hilario da Silva, M.Sc.
UFF

Profa. Anna Dolejsi Santos, D.Sc.
UFF

Niterói

2007

RESUMO

Telemedicina, ou teleconsulta, consiste em uma comunicação em tempo real entre dois profissionais da área médica, em que um assiste ao outro no diagnóstico de um paciente.

Nosso trabalho consiste na apresentação de uma aplicação direcionada à telemedicina através da utilização de uma comunicação, via rede sem fio, entre um dispositivo móvel e uma aplicação servidora. Nesta comunicação, o Cliente requisitará ao Servidor a visualização de um determinado arquivo que contenha dados sobre algum exame realizado em um determinado paciente. O Servidor abre o arquivo em formato DICOM e transfere os dados relativos a este para o Cliente, que por sua vez visualiza estes dados.

Para efeitos de implementação, foi utilizada a ferramenta Microsoft Visual Studio .NET, para a programação, em linguagem C#, da aplicação. Também foi utilizada uma classe para manipulação de arquivos DICOM.

Palavras Chave:
PDA, *mesh*, telemedicina, DICOM

LISTA DE ACRÔNIMOS

ACR:	ASSOCIATION COLLEGE OF RADIOLOGY
AVI:	AUDIO VIDEO INTERLEAVE
CSV:	COMMA SEPARATED VALUES
DHCP:	DYNAMIC HOST CONFIGURATION PROTOCOL
DICOM:	DIGITAL IMAGING COMMUNICATIONS IN MEDICINE
GIF:	GRAPHICS INTERCHANGE FORMAT
HTML:	HYPERTEXT MARKUP LANGUAGE
IAM:	INFARTO AGUDO DO MIOCÁRDIO
IEEE:	THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS
IP:	INTERNET PROTOCOL
JPEG:	JOINT PHOTOGRAPHIC EXPERTS GROUP
NEMA:	NATIONAL ELECTRICAL MANUFACTURES ASSOCIATION
OLSR:	OPTIMIZED LINK STATE PROTOCOL
PDA:	PERSONAL DIGITAL ASSISTANT
PNG:	PORTABLE NETWORK GRAPHICS
RFID:	RADIO FREQUENCY IDENTIFICATION
TIFF:	TAGGED IMAGE FILE FORMAT
WI-FI:	WIRELESS FIDELITY
WMF:	WINDOWS METAFILE FORMAT

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1 - INTRODUÇÃO

1.1 MOTIVAÇÃO

A saúde é um bem fundamental para toda a população, independentemente de cor, credo, raça ou condição social. Todos os cidadãos têm direito a um serviço de saúde eficiente e objetivo. Todos os esforços e tecnologias que venham melhorar o desempenho do nosso sistema de saúde são bem vindos.

A *telemedicina* [6] vem se apresentando como um conjunto de ferramentas eficiente e simples em que um especialista consulta outro, através de um sistema cliente-servidor, quando se sente em dificuldades de elaborar um diagnóstico e necessita de ajuda para tanto. Através dela, um sistema médico pode comunicar-se com outro, através de comunicações em rede, passando dúvidas e inclusive enviando-lhe prontuários com exames feitos no paciente para ilustrar melhor o seu problema. A resposta do outro especialista, através do sistema que estiver utilizando, vem em tempo real e assim o problema é resolvido com eficiência, uma vez que não é necessário deslocar o paciente para outro consultório e ter que marcar uma consulta com outro médico.

Uma solução proposta para um uso eficiente da telemedicina foi apresentada pelo projeto MARFIM (Medicina Assistida por Redes Sem Fio Multimídia) [12]. Neste projeto, é proposta uma aplicação de telemedicina voltada para profissionais do Sistema Único de Saúde (SUS). A princípio, o projeto MARFIM consistiria em um projeto piloto chamado AToMS (AMI Teleconsultation & Monitoring System)[12]. Este projeto seria voltado ao tratamento emergencial de pacientes com Infarto Agudo do Miocárdio (IAM). Através do sistema proposto, um paramédico poderia enviar a um cardiologista, em tempo real, um prontuário eletrônico com dados de exames cardiológicos feitos no paciente. Assim, obteria uma resposta em tempo real e um acompanhamento especializado durante o andamento de toda a operação de emergência.

Soluções móveis de suporte à telemedicina apresentam baixo custo de instalação e implementação. Através de uma rede em malha comunitária (mais detalhes sobre redes em malha no Capítulo 2) um hospital poderia ter vários pontos de acesso que se intercomunicassem, e profissionais dotados de dispositivos sem fio poderiam, através destes, comunicarem com especialistas de outras áreas para assim prover um atendimento de boa qualidade e eficiente.

Uma maneira de uniformizar e aperfeiçoar esta comunicação entre aplicações médicas veio através do padrão DICOM (mais detalhes na Seção 3.3 “Padrão DICOM”do Capítulo 3 “Ferramentas do projeto”). O padrão DICOM é um padrão de envio e recebimento de imagens médicas digitalizadas a ser seguido por qualquer aplicação de telemedicina. Nos arquivos DICOM, como será visto mais adiante, não só a imagem do exame é mostrada, mas também dados como nome do médico, paciente, data e hora da realização do exame.

Em nossa aplicação, utilizamos as vantagens dadas pelo padrão DICOM para o formato de imagens médicas, além do conceito de redes em malha comunitárias para prover a comunicação e a transferência de dados entre o dispositivo móvel cliente e o servidor remoto.

Assim, temos uma oportunidade de gerar uma aplicação que utiliza e implementa os conceitos da telemedicina, fazendo com que se torne real a possibilidade da mesma ser implantada. Além disso, provar ser possível ter uma aplicação a baixo custo que possa servir como motivação para propostas futuras da área médica. Estas propostas visariam a concretização de um projeto de software que ajudasse nos serviços de saúde nos hospitais públicos brasileiros.

1.2 OBJETIVOS

O objetivo deste projeto é apresentar um software funcional com suporte à telemedicina para ambientes sem fio (PDA), que depois poderá ser utilizado em projetos futuros que visem melhorar os serviços em um hospital. O grande foco do projeto é demonstrar que é possível realizar a transferência de imagens médicas em formato DICOM de um servidor a um dispositivo móvel. Como exemplo de aplicabilidade, podemos citar o caso de um paramédico em uma emergência de um hospital que possa comunicar-se em tempo real a um cardiologista enquanto estiver tratando um paciente que tenha sofrido um infarto. Em casos de transferência, o fato de estar usando uma rede móvel permite que o especialista possa acompanhar todo o trabalho em tempo real durante a ocorrência da mesma.

Aqui estaremos dando um passo inicial rumo ao desenvolvimento de aplicativos de suporte médico em dispositivos pequenos. Todos os conceitos e métricas utilizados no seu desenvolvimento podem ser reaproveitados mais tarde, com isso gerando novas oportunidades e tornando possível o serviço eficiente de telemedicina ao alcance do serviço público de saúde.

Esperamos, com este projeto, incentivar cada vez mais as pesquisas na área de telemedicina, uma área que ainda se encontra dando seus passos iniciais, mas que tem tudo para crescer, oferecendo vantagens para todos que nela estiverem envolvidos. Esperamos também demonstrar a importância que uma aplicação que ofereça a comodidade e eficiência de uma rede móvel através de um dispositivo sem fio pode ter em situações em que respostas imediatas são necessárias.

1.3 ORGANIZAÇÃO

O trabalho está organizado em cinco capítulos, como descrito a seguir. No Capítulo 2, apresentamos o conceito de redes em malha e o protocolo utilizado para roteamento e encaminhamento de pacotes utilizado na aplicação. O Capítulo 3 apresenta o Visual Studio, a ferramenta utilizada na geração da aplicação; a ferramenta MySQL, utilizada para gerenciamento de Banco de Dados, utilizada aqui para armazenamento da base de dados do sistema; os conceitos do padrão DICOM, segundo o qual será feita a armazenagem e leitura de dados; e a ferramenta DicomWorks, utilizada para manipulação de imagens DICOM. O Capítulo 4 descreve a aplicação em si, seus detalhes em relação à entrada, à saída, ao processamento e à comunicação entre o cliente móvel e o hospedeiro remoto. Em seguida são apresentados dados de análise de desempenho do software. No Capítulo 5 é apresentada uma breve conclusão do trabalho feito, com perspectivas para projetos futuros.

CAPÍTULO 2 - COMUNICAÇÃO EM REDES EM MALHA

2.1 INTRODUÇÃO

Para que um dispositivo sem fio possa comunicar-se com um servidor remoto, é necessária a utilização de uma rede sem fio. Atualmente, um dos maiores alvos de estudos nas áreas de Ciência da Computação e de Engenharia de Telecomunicações tem sido as redes sem fio.

No mundo atual a comodidade tem recebido muita importância. Oferecer uma forma das pessoas poderem acessar uma rede com informações em hospedeiros remotos sem a necessidade de uma conexão utilizando cabos é a melhor forma de oferecer comodidade, pois assim a pessoa pode mover-se livremente em um ambiente e acessar em qualquer ponto as informações desejadas.

Existem várias maneiras de se oferecer uma comunicação utilizando redes móveis. Como exemplos, podemos citar a Internet por celular, a Internet por ondas de rádio e as redes em malha (*mesh*) [20]. Neste capítulo, damos uma abordagem especial a esta última, utilizada na nossa aplicação. Em seguida abordamos um protocolo de rede especialmente voltado para redes móveis utilizado em nossa aplicação, chamado protocolo OLSR. Por último apresentamos exemplos de algumas aplicações que utilizam redes *mesh*.

2.2 REDES EM MALHA

2.2.1 UMA ABORDAGEM SOBRE REDES MESH

Para prover comunicação em redes sem fio, foram criadas as redes denominadas *ad hoc*. Este tipo de rede consiste no fato de todos os seus participantes serem nós móveis que se comunicam sem fio, participando do roteamento de pacotes uns aos outros. Para prover esta comunicação

sem fio, o grupo IEEE resolveu estabelecer um protocolo para buscar uma padronização em sua implementação. Este protocolo é conhecido como IEEE 802.11, que além do modo *ad hoc* também possui o modo de operação infra-estruturado.

O modo infra-estruturado consiste de estações móveis entram em contato direto com pontos de acesso, que são responsáveis pela transmissão dos pacotes dentro da rede. Os nós móveis só podem entrar em contato com um ponto de acesso, mesmo que estejam muito próximos um do outro. Exemplos de redes infra-estruradas são as redes de telefonia celular e as redes Wi-Fi [20].

Uma nova maneira de implementar redes *ad hoc* que surgiu foi a rede em malha, que também pode ser chamada de rede *mesh*. Nas redes *mesh* há a diferenciação entre os nós denominados clientes e os nós denominados pontos de acesso (PA). Os PAs são responsáveis pelo acesso à grande rede, e os nós clientes participam do repasse de pacotes entre si. Ou seja, os PAs são responsáveis pelo roteamento e encaminhamento de pacotes. Uma outra característica das redes *mesh* que as fazem diferentes de uma rede *ad hoc* convencional é o fato de clientes estáticos (como um PC, por exemplo) poderem participar da rede, também utilizando-a para realizar acessos a uma rede de banda larga, através da configuração de uma rede sem fio. Com isso, este dispositivo, ainda que estático, pode participar do repasse de pacotes e da comunicação com dispositivos móveis. Na Figura 2.1, apresentamos a arquitetura de uma rede *mesh*. Na Figura 2.2 mostramos exemplos de dispositivos que podem servir como nós clientes neste tipo de rede e, na Figura 2.3, são apresentados exemplos de dispositivos que atuam como PA.

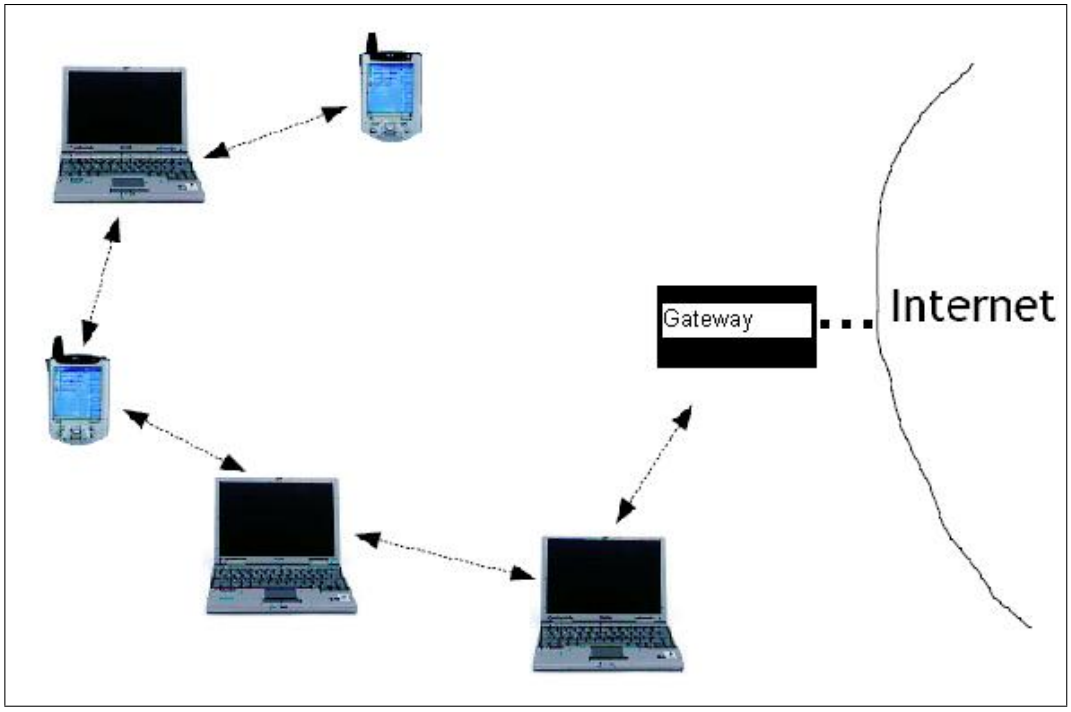

Figura 2.1: Arquitetura de uma rede *mesh*

A Figura 2.1 ilustra a arquitetura de uma rede *mesh*. Os nós clientes são representados pelos dois PDAs e pelos três notebooks. Aqui temos apenas um PA que também é responsável pela comunicação com a Internet, que na Figura 2.1 chamamos de Gateway. Se algum dos PDAs ou dos notebooks deseja comunicar-se com a Internet deve obter um endereço IP com o Gateway. Um pacote vindo da Internet passa pelo Gateway e em seguida é pasado ao PDA ou notebook ao qual foi endereçado. Essa transmissão pode ser feita diretamente pelo Gateway ou através de outro nó cliente, se essa for a melhor decisão, de acordo com o algoritmo de roteamento que estiver rodando na rede.

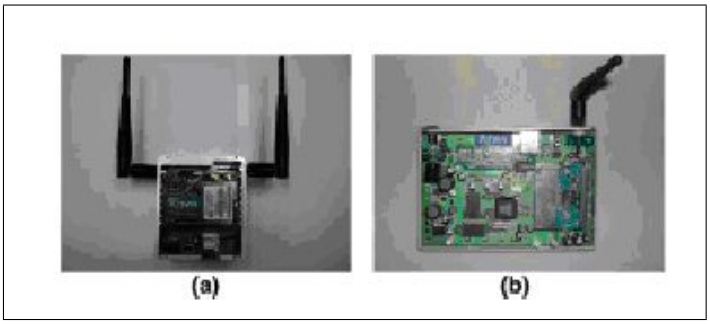

Figura 2.2: Exemplos de roteadores *mesh*: (a) Power PC (b) Advanced Risc Machines (ARM)

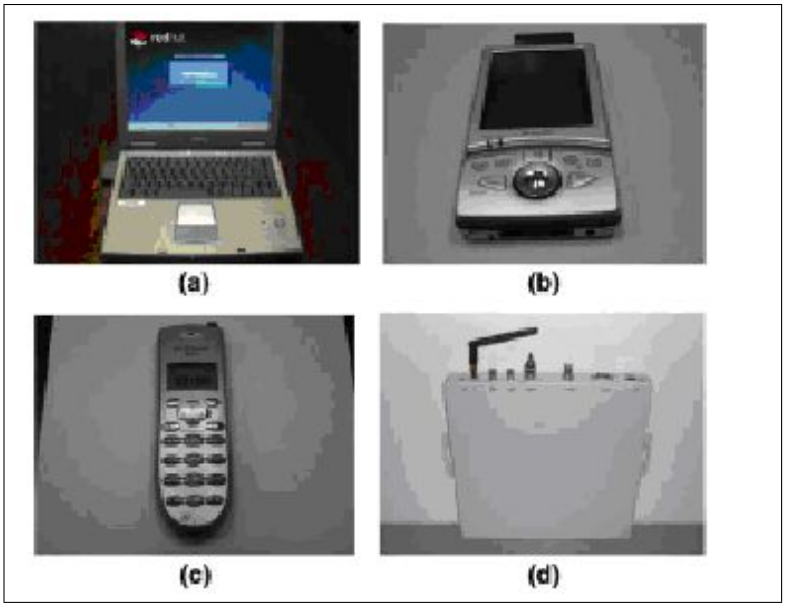

Figura 2.3: Exemplos de nós clientes de uma rede *mesh*: (a) Laptop (b)Personal Digital Assistant (PDA) (c) Celular (d) Wi-Fi Radio Frequency Identification (RFID) Reader

Nas Figuras 2.2 e 2.3, extraídas de [1], mostramos exemplos do que podem ser nós de uma rede *mesh*. Na Figura 2.2 temos exemplos de dispositivos especializados em exercer funções de roteamento de pacotes dentro da rede, com isso eles podem trabalhar como PAs dentro de uma rede *mesh*. Um nó móvel deve conectar-se à rede para fazer parte desta. Para isso, deve se registrar em um PA. Já na Figura 2.3, temos exemplos de dispositivos que podem conectar-se a uma rede *mesh*.

Assim como em redes *ad hoc*, uma rede *mesh* também oferece mobilidade, o que pode trazer problemas. Em uma rede móvel, as quedas de enlace ou a saída de um nó são mais frequentes, e a rede deve estar preparada para lidar com este tipo de evento indesejável. Quando um nó se move, ele pode sair do alcance de um enlace sem fio ou então ser alcançado por outro com maior eficiência. Para tratar esses problemas sem deixar de oferecer uma comunicação de boa qualidade aos componentes de rede, foram propostos alguns protocolos de roteamento especiais para redes do tipo *ad hoc*.

2.3 PROTOCOLO OLSR

Agora que falamos sobre redes móveis, chegamos a um ponto crítico do assunto em questão: o protocolo de roteamento de pacotes. No caso de uma rede sem fio com mobilidade, são muito

freqüentes os casos de queda no enlace e de perdas de pacotes. Várias têm sido as formas de buscar uma otimização das formas de repasse de mensagens em ambientes sem fio. No caso da rede *mesh* utilizada para trocar as mensagens entre o cliente PDA e o servidor remoto, o protocolo escolhido para descoberta de rotas foi o protocolo *Optimized Link State Protocol* (OLSR) [3].

A aplicação desenvolvida utiliza este protocolo sobre sua camada de rede, já que foi o protocolo escolhido para ser implementado na rede *mesh* da Universidade Federal Fluminense, utilizada para prover a comunicação entre o PDA e o Servidor.

O OLSR é um protocolo do tipo pró-ativo, ou seja, é mantida uma tabela de rotas que é periodicamente atualizada, mantendo em si as distâncias entre os nós e os enlaces disponíveis para o repasse de pacotes. Esta tabela é sempre consultada quando há a requisição de envio de alguma mensagem entre dois nós da rede. Cada nó contém tabelas com informações de roteamento, que podem ser atualizadas caso este perceba uma mudança na topologia da rede (queda de enlace, entrada ou saída de um nó, por exemplo). Ele nada mais é do que uma otimização do já conhecido protocolo de estado de enlace.

Para entender bem o protocolo OLSR, primeiro vamos fazer uma referência rápida ao conceito de inundação. A inundação é uma técnica na qual o nó que retém o pacote repassa-o para todos os seus vizinhos. Estes, por sua vez, verificam se a mensagem está destinada a eles. Se estiver, passa a mensagem às camadas superiores e não repassa o pacote a mais ninguém. Se não, repassam a todos os seus vizinhos. Esta técnica leva a um tráfego muito acentuado da rede, acarretando em queda de desempenho. Na Figura 2.5 temos um exemplo de inundação: aqui temos uma mensagem saindo por inundação do nó que está no meio da topologia sendo transmitida por inundação aos seus vizinhos imediatos e também por estes aos seus vizinhos.

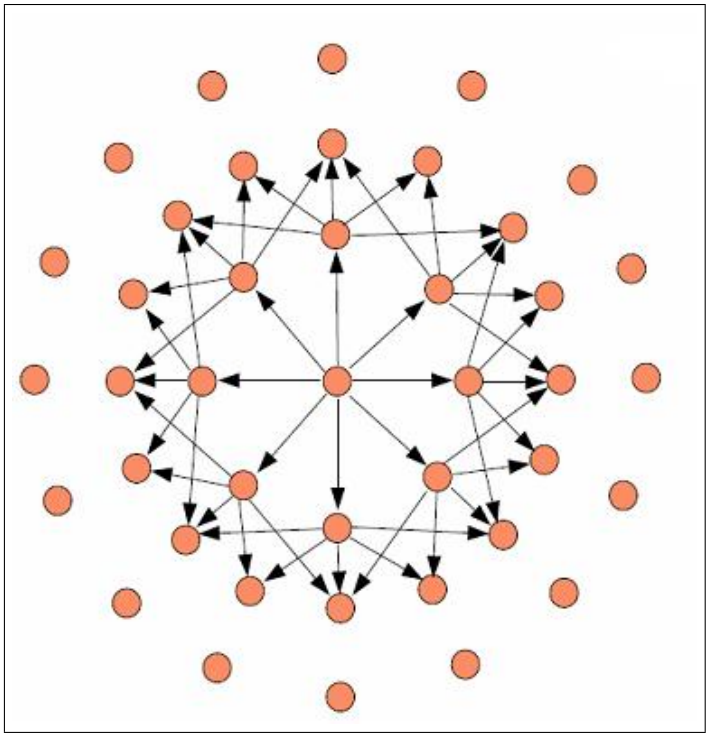

Figura 2.4: Exemplo de inundação

2.3.1 DESCOBERTA E MANUTENÇÃO DE ROTAS

O OLSR utiliza o processo de inundação controlada para descoberta da topologia da rede. Mensagens de *Hello* são enviadas periodicamente de um nó para todos os seus vizinhos. Com isso, torna-se possível a detecção destes nós vizinhos e a descoberta da topologia da rede. Caso um nó responda a esta mensagem de *Hello*, ele é inserido na tabela de rotas do nó que enviou esta mensagem. Ao longo do tempo, os nós mandam mensagens de *Hello* para verificar se os enlaces que os interligam a todos os seus vizinhos ainda estão ativos. Caso não estejam, a linha da tabela de rotas referente a eles é excluída.

Envio de mensagens

Para o envio de suas mensagens, o OLSR utiliza uma otimização da inundação, através de nós selecionados, denominados MPR (*Multipoint Relay*). As mensagens, quando repassadas, não são repassadas a todos os nós vizinhos, e sim aos nós MPR selecionados. A idéia da utilização destes MPRs é minimizar ao máximo a redundância de informações dentro de uma determinada região, além de diminuir o número de pacotes de controle de tráfego gerados.

Cada nó mantém uma tabela, que pode ser alterada periodicamente, dos seus nós MPR e o custo do enlace entre ele e cada MPR. Esta tabela é passada periodicamente a todos os outros nós por inundação. Assim, um nó que deseja mandar sua mensagem a outro determinado nó já saberá por qual MPR o destinatário poderá ser alcançado, já que poderá calcular, através do algoritmo de Dijkstra [11], o caminho de MPRs com o menor custo até ele. A Figura 2.6 ilustra a inundação controlada do OLSR, em que os nós azuis representam os nós MPR selecionados para passar a mensagem adiante.

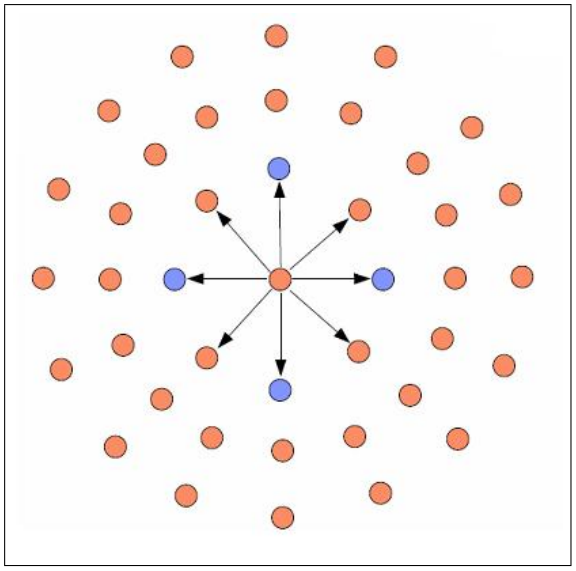

Figura 2.5: Exemplo OLSR com os nós MPR selecionados

2.3.2 MENSAGENS BROADCAST

No caso de envio de mensagens *broadcast*, os nós que recebem esta mensagem verificam se estão no grupo de MPR do nó que lhes tenha enviado a mensagem. Se estiverem, passam a mensagem adiante. Caso contrário, não repassam a mensagem. Com isso, o OLSR reduz significativamente o volume de tráfego na rede, não havendo uma imensa inundação de pacotes, reduzindo a possibilidade de congestionamento na rede.

2.4 EXEMPLOS DE APLICAÇÕES UTILIZANDO REDES MESH

Vários estudos e propostas de aplicações utilizando redes mesh têm surgido e sido postas em prática. Para ilustrar melhor o assunto, mostraremos exemplos de aplicações empregando esta tecnologia. Inicialmente mostraremos uma implementação acadêmica, feita na Universidade Federal Fluminense, e em seguida será abordada uma aplicação comercial, posta em prática pela Nortel.

2.4.1 GT-MESH

O GT-Mesh é uma proposta conjunta dos departamentos de Engenharia de Telecomunicações e de Ciência da Computação da Universidade Federal Fluminense. A idéia deste projeto é de realizar um teste de uma rede comunitária em malha envolvendo o campus da Praia Vermelha, em Niterói [20].

Com isso objetiva-se apresentar uma proposta de uma rede com acesso à internet de baixo custo e fácil implementação, e em seguida a expansão para comunidades residenciais de bairros próximos à universidade. Possuindo uma rede já estabelecida, subprojetos poderiam ser realizados objetivando a melhoria da utilização da rede e, mais posteriormente, o oferecimento de Qualidade de Serviço (QoS).

Neste experimento, foram utilizados 14 nós espalhados pelo campus da Praia Vermelha e comunidades residenciais vizinhas. Os roteadores funcionam com o sistema operacional OpenWRT [19]. Como protocolo de roteamento, foi escolhido o *Optimized Link State Protocol* (OLSR, em cujos detalhes entraremos mais adiante).

2.4.2 NORTEL WIRELESS MESH NETWORK (WMESH)

Na cidade de Taipei, em Taiwan, foi posta em prática uma instalação de uma imensa rede corporativa *mesh*, com pontos de acesso espalhados por toda a cidade. A idéia é distribuir aproximadamente 10 mil pontos de acesso em uma área 272 quilômetros quadrados onde 90% da população de Taipei reside (dados de 2005) [17].

Na topologia da solução proposta pela Nortel, vários pontos de acesso formam juntos uma rede comunitária (CAN), por onde podem entrar os dispositivos móveis da rede. As CANs, por sua vez, são conectadas a roteadores que provêem acesso à grande rede.

Para realizar a conexão entre a rede distribuída e o backbone, os pontos de acesso utilizam funções de comutação combinadas com outras funções de rede sem fio. Nos backbones estão localizados os chamados *Wireless Gateways* (WG), que são os responsáveis, entre outras funções, pela segurança das conexões.

Para conectar-se à rede, o dispositivo móvel recebe um endereço IP dinâmico, via *Dynamic Host Configuration Protocol* (DHCP). Para isso, utiliza a técnica do IP Móvel, com o WG trabalhando como *Home Agent* e o ponto de acesso trabalhando como *Foreign Agent*, e é feito o tunelamento entre eles. O dispositivo móvel então se registra ao ponto de acesso, recebendo o seu endereço de IP a ser utilizado naquela sessão à qual estará conectado.

CAPÍTULO 3 - FERRAMENTAS DO PROJETO

3.1 INTRODUÇÃO

Para oferecer uma melhor ilustração do software que é o assunto desta monografia, resolvemos fazer uma abordagem sobre as ferramentas utilizadas para o seu desenvolvimento. Melhorias sobre o produto podem ser objetivadas e inclusive implementadas. O conhecimento das ferramentas é de fundamental importância, pois assim quem quiser trabalhar ou mesmo fazer pesquisas sobre o software desenvolvido poderá ter uma noção completa sobre a sua concepção.

Apresentamos neste capítulo uma visão geral sobre o ambiente de desenvolvimento Microsoft Visual Studio .NET, um ótimo mecanismo para desenvolver uma aplicação voltada a pequenos dispositivos. Em seguida falamos sobre o sistema gerenciador de banco de dados MySQL, utilizado para guardar os dados dos pacientes que utilizamos na aplicação (mais detalhes sobre a aplicação serão descritos no Capítulo 4). A seguir, apresentamos o protocolo de envio de mensagens DICOM, especialmente desenvolvido para envio e recebimento de imagens médicas para efeitos de estudo e avaliação de prontuários médicos. Por último abordamos a ferramenta DICOMWorks, utilizada para visualização e edição de imagens DICOM.

3.2 MICROSOFT VISUAL STUDIO .NET

Ao se desenvolver um software uma das principais escolhas recai sobre o ambiente de desenvolvimento e a escolha da linguagem que julgamos a mais apropriada para o problema proposto. Em nosso caso, tivemos preferência pela ferramenta Microsoft Visual Studio .NET, devido ao fato de já termos conhecimento e experiência com a dita ferramenta, além das comodidades oferecidas pela mesma em relação a desenvolvimento para pequenos dispositivos.

Aqui apresentamos brevemente a ferramenta de desenvolvimento Microsoft Visual Studio .NET, mais detalhes sobre este aplicativo podem ser encontrados no sítio da MSDN [16].

3.2.1 PROGRAMAÇÃO VISUAL

O Visual Studio oferece recursos tanto para programação de aplicações de console como para aplicações com interface. Através dos *frameworks* 2.0 e, mais recentemente, 3.0, o Visual Studio também oferece a oportunidade de se trabalhar com dispositivos móveis de maneira simples e eficiente, daí vem a sua escolha para o desenvolvimento do nosso trabalho.

3.2.2 LINGUAGEM C#

Para a escrita do código da aplicação, escolhemos a linguagem C#, que tem um ambiente de desenvolvimento integrado ao Visual Studio, o Microsoft Visual C#. Uma visão mais abrangente desta ferramenta pode ser encontrada no sítio do Visual C# [2]. Nossa escolha por essa linguagem se deu à sua fácil utilização e a experiências anteriores com a mesma.

A linguagem C# é uma linguagem concebida sob o paradigma da programação orientada a objeto. Com isso, podemos isolar a interface da parte de implementação e da comunicação com a base de dados.

3.2.3 DESENVOLVIMENTO DE DISPOSITIVOS NATIVOS

Outro aperfeiçoamento, segundo o sítio da Microsoft [5] é a possibilidade de desenvolvimento de aplicativos para dispositivos inteligentes e portáteis. Segundo o sítio, "O Visual C# .NET 2003 (cuja versão atualizada é o 2005) e o Microsoft .NET Framework 1.1 (atualizado nas versões 2.0 e 3.0) incluem agora suporte nativo para mais de 200 dispositivos móveis da Web,

incluindo telefones celulares, pagers e PDAs. A versão do Visual C# .NET para desenvolvedores profissionais, denominada Visual C# .NET 2003 Professional, permite o desenvolvimento para dispositivos inteligentes tais como o Pocket PC."

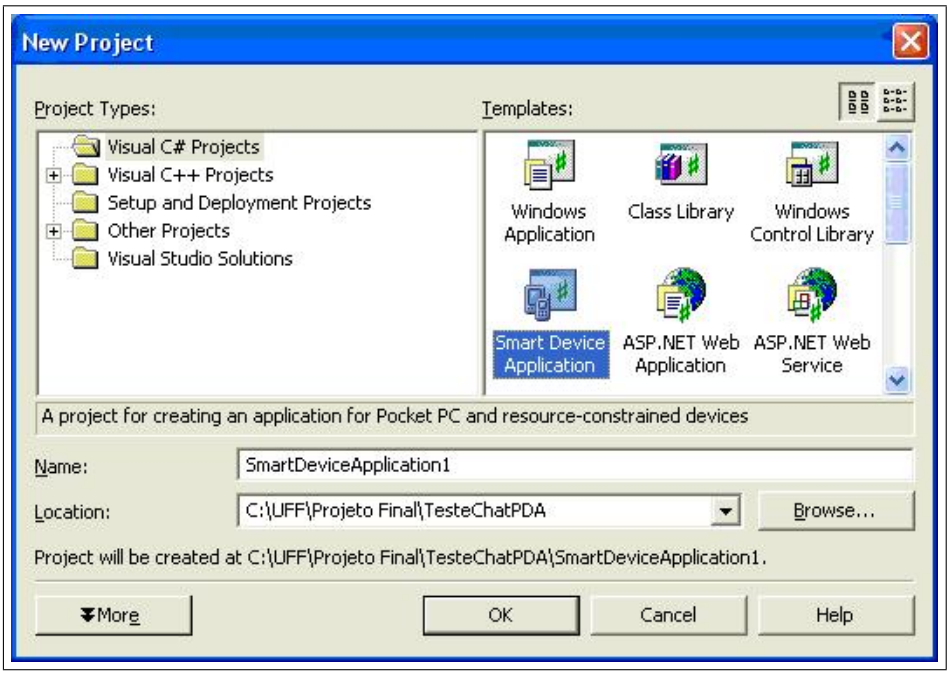

Figura 3.1: Tela para seleção de desenvolvimento em pequenos dispositivos - Visual Studio .NET 2003

Na Figura 3.1, vemos uma criação de um projeto em Visual C# em ambiente de programação para pequenos dispositivos. Essa é uma facilidade oferecida pelo Visual Studio para quem deseja implementar um software para Pocket PC e celulares, por exemplo.

3.2.4 EMULADOR

O Visual Studio também oferece uma opção de depuração através de um emulador de ambiente PDA. Através deste mecanismo, pode-se fazer uma depuração inicial antes de passar a aplicação para o ambiente definitivo. Com isso, podemos ter uma eficácia maior no desenvolvimento, pois não é necessário desperdiçar tempo passando a aplicação para o Pocket antes de descobrir alguns erros que o código possa ter.

3.3 MYSQL

Para efeitos de armazenamento dos dados relevantes para o sistema final, foi utilizado o sistema gerenciador de banco de dados (SGBD) MySQL [14]. Nossa escolha por esse SGBD se deu ao fato da sua facilidade de utilização, além de ser uma ferramenta gratuita.

O MySQL é um SGBD que utiliza a linguagem *Structured Query Language* (SQL) para manipular os dados nele armazenados.

3.3.1 ORIGEM E CONSUMIDORES

O MySQL foi criado na Suécia, pelos programadores David Axmark, Allan Larsson e Michael Widenius [8].

Hoje em dia o MySQL é o SGBD mais popular do mundo, sendo utilizado por 11 milhões de clientes, entre eles sítios de grande importância e grande popularidade, como o Google, o Wikipedia, o Yahoo!, o Ticketmaster, etc. [5]. Grandes corporações como a NASA, a Motorola e a Silicon Graphics também utilizam o MySQL como SGBD.

3.3.2 OBJETIVOS

Os objetivos a serem alcançados pelo servidor MySQL são [14]:

- possuir disponibilidade de recursos para qualquer tipo de aplicação;
- prover facilidade de uso;
- torná-lo cada vez mais rápido e seguro;
- torná-lo livre de *bugs*.

3.3.3 VANTAGENS DA UTILIZAÇÃO DO MYSQL

A escolha do MySQL como SGBD de uma aplicação pode trazer uma série de vantagens, como [14]:

- excelente desempenho e estabilidade;
- pouca exigência quanto a recursos de hardware: a economia em relação a gastos com hardware chega a 70% se comparada a outros SGBDs;
- facilidade de uso;
- redução de custos com administração e suporte por volta de 50%;
- custos de obtenção de licença 90% mais baratos;
- robustez;
- portabilidade: pode ser usado em sistemas operacionais diversos, como Windows, Linux, FreeBSD, Solaris, etc..

3.4 PADRÃO DICOM

Em nossa aplicação de visualização de exames médicos, o armazenamento e a recuperação de dados são feitos segundo o padrão DICOM (*Digital Imaging and Communications in Medicine*) [15]. O padrão DICOM propõe uma forma de armazenamento de imagens médicas cujo formato possibilitaria armazenar junto com a imagem outros dados relevantes do paciente e a visualização desta imagem local e remotamente.

O padrão DICOM foi proposto e desenvolvido por um comitê formado por membros da *American College of Radiology* (ACR) e da *National Electrical Manufacturers Association* (NEMA). A proposta do padrão é definir uma especificação para auxiliar a visualização e a transferência de imagens médicas.

3.4.1 OBJETIVOS DO PADRÃO

Os objetivos do padrão DICOM [15] são obter compatibilidade e promover uma maior eficiência na comunicação de imagens médicas, independentemente do aparelho que está participando da comunicação. A idéia fundamental é que o padrão DICOM seja utilizado por cada área médica que utilize o envio e estudo de imagens, como a cardiologia, a radiologia e a endoscopia, entre outras.

Outros objetivos do padrão são facilitar o desenvolvimento e a expansão dos sistemas de requerimento de interface entre um arquivo de imagem e redes de sistema de comunicação (PACS). Isso possibilitaria esses sistemas comunicarem-se com outros sistemas de informação hospitalar. O padrão também visa permitir a criação de uma base de dados de informações de diagnósticos que possam ser examinados por uma grande variedade de aparelhos distribuídos geograficamente. [13].

3.4.2 FORMATO DE IMAGENS

O padrão DICOM especifica um formato de compressão de imagens que o diferencia dos já conhecidos padrões JPEG, TIFF e GIF. Nas imagens que são comprimidas utilizando-se o padrão DICOM, não só a imagem, baseada no formato JPEG, mas também as informações relativas ao paciente são inseridas no arquivo.

Um arquivo DICOM contém um cabeçalho que armazena, entre outras informações, o nome do paciente, o tipo de escaneamento e as dimensões da imagem. Além do cabeçalho, um arquivo DICOM também possui campos com todos os dados da imagem. O arquivo DICOM também pode ser comprimido reduzindo, assim, o tamanho da imagem.

Um exemplo de uma imagem DICOM é mostrado a seguir. Neste exemplo, os primeiros 794 bytes para formar o cabeçalho.

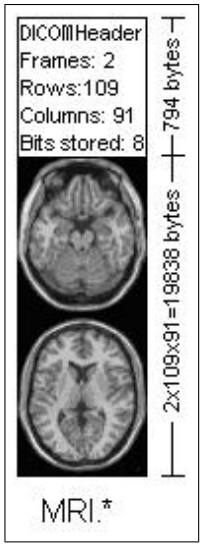

Figura 3.2: Exemplo de uma imagem DICOM

O tamanho do cabeçalho de uma imagem DICOM pode variar de acordo com as informações que são armazenadas nele. Na Figura 3.2, o cabeçalho define uma imagem com dimensões 109x91x2 voxels, com resolução de 1 byte por voxel (o voxel é o equivalente tridimensional de um pixel). Ou seja, que o tamanho total da imagem é de 19838 bytes. A imagem em si vem logo em seguida do cabeçalho.

3.4.3 VISÃO DA ARQUITETURA DO PADRÃO

O padrão DICOM contém uma arquitetura específica para troca de mensagens entre dois dispositivos remotos. A idéia do padrão é prover comunicação entre serviços independentemente do protocolo de rede que estiver sendo utilizado na comunicação entre os sistemas finais.

Na Figura 3.3, extraída de [13] podemos ver o modelo de protocolo de comunicações do DICOM. Acima da parte preta temos o protocolo a nível de aplicação e abaixo a níveis de apresentação, sessão, transporte, rede, enlace e físico. A idéia é, como dito anteriormente, prover comunicações entre sistemas que se comuniquem nos mais diversos protocolos de rede. À esquerda encontramos um ambiente ponto a ponto ("ppp", ou "ponto a ponto"). Ao meio, temos o protocolo TCP/IP. Podemos reparar que sobre a camada de transporte há uma camada especificada pelo padrão, denominada *Upper Layer Protocol* (ULP), que na Figura 3.3 pode ser encontrada acima do protocolo TCP de transporte. À direita temos o padrão ISO-OSI. Através dessa independência em camada de rede, o padrão DICOM nos assegura que quaisquer sistemas

que sejam compatíveis poderão comunicar-se independentemente do protocolo de rede utilizado em suas comunicações.

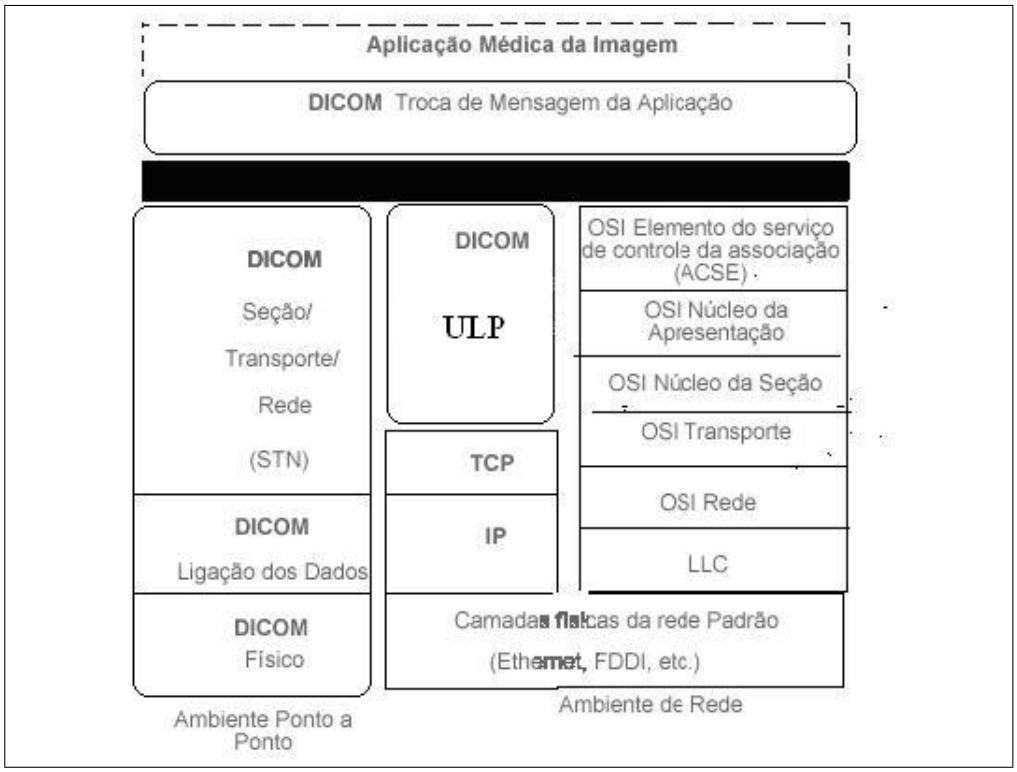

Figura 3.3: Modelo de protocolo de comunicações do DICOM

Na camada de aplicação, cinco áreas de funcionalidade são englobadas. São elas:

- transmissão e persistência de objetos completos, como imagens e documentos;
- requisição e obtenção dos objetos;
- desempenho eficiente de ações específicas, como apresentação de imagens em uma tela;
- gerenciamento de fluxo;
- qualidade e consistência na apresentação da imagem, tanto para visualização como para impressão.

3.4.4 VANTAGENS DO PADRÃO DICOM

Utilizar o padrão DICOM para o armazenamento e transferência de imagens médicas pode trazer uma série de vantagens. Uma delas é o fato do padrão definir que os dados do paciente e da imagem sejam guardados junto à esta. Com isto, ele possibilita a extração de imagens de forma direta, mantendo assim a integridade dos dados. Outra vantagem do DICOM é que ele possibilita melhorar o desempenho de sistemas de comunicação que troquem informações entre si através de imagens digitais médicas. Por último, podemos apontar sua independência da camada de rede, o que possibilita dispositivos com sistemas distintos, em redes privadas separadas, poderem se comunicar independentemente.

3.5 DICOMWORKS

O DICOMWorks [7] é uma ferramenta gratuita de visualização e edição de arquivos DICOM. Provê uma série de funcionalidades para quem deseja trabalhar com manipulação de imagens DICOM. Nossa escolha pela utilização desta ferramente deu-se ao fato de ela apresentar facilidade na edição dos arquivos DICOM e pela sua facilidade de uso.

Uma dessas vantagens é o reconhecimento de diretórios que contenham arquivos com imagens DICOM. Outra grande vantagem do DICOMWorks é a disponibilidade de painéis para visualização de até 4 imagens simultâneas (ver Figura 3.4). Suporta rolagem do mouse e zoom da imagem, entre outras vantagens [7].

A Figura 3.4 apresenta uma imagem DICOM aberta para edição no DICOMWorks. À direita, temos a imagem médica em 4 quadros e à esquerda a janela para edição dos dados do exame.

Figura 3.4: Exemplo de visualização de imagem DICOM no DICOMWorks

3.5.1 EDIÇÃO DE IMAGENS

Outro serviço oferecido pelo DICOMWorks é a edição das imagens DICOM. Através desta ferramenta, é possível visualizar e modificar todos os dados do arquivo DICOM, como nome do paciente, nome do médico, data do exame e imagem médica, por exemplo. O DICOMWorks também possibilita anonimato de algum dado. Ou seja, se não se deseja disponibilizar o nome do médico ou do paciente é possível ocultar estes dados.

3.5.2 EXPORTAÇÃO DAS IMAGENS EM OUTROS FORMATOS

Com o DICOMWorks também é possível realizar exportação das imagens para arquivos em formato Bitmap, JPEG, TIFF, WMF e GIF, entre outros. Também é possível exportar essas imagens em seqüência como um vídeo em formato AVI. Pode-se também exportar a imagem como uma página HTML, com algumas anotações inseridas pelo próprio usuário.

O DICOMWorks possibilita a criação de um CD com arquivos organizados pelos nomes dos pacientes em ordem alfabética, além disso ele lida com a impressão de um label para o CD com a lista dos nomes de todos os pacientes referentes aos arquivos DICOM a ser inseridos na mídia.

Esta ferramenta permite comprimir arquivos DICOM sem perdas em formatos DMZ (“DICOM Zip”) e torna possível a impressão de quaisquer grupos de imagens DICOM em uma impressora comum. Isto inclui impressão de imagens, de lista de pacientes e de labels para CDs, como descrito anteriormente.

CAPÍTULO 4 - IMPLEMENTAÇÃO

4.1 INTRODUÇÃO

Neste capítulo, chegamos ao objetivo maior da nossa monografia: a programação do software de visualização de exames médicos para ambiente PDA. Aqui descreveremos todas as funcionalidades da ferramenta desenvolvida, entre elas: entrada e saída de dados, seu processamento e resultados esperados.

Em seguida apresentaremos uma análise qualitativa do projeto, com estatísticas de atraso no envio de imagens DICOM e consumo de memória no dispositivo móvel. Com isso, poderemos ter a exata noção do comportamento do sistema, e o que pode e deve ser alterado para que, no caso de algum projeto futuro baseado nesta aplicação, ela possua um melhor desempenho.

4.2 APLICAÇÃO

4.2.1 BREVE DESCRIÇÃO DO PROBLEMA

A aplicação consiste em um Cliente móvel que envia a um Servidor remoto o nome do paciente cujos dados deseja-se visualizar. O Servidor por sua vez retorna uma lista com arquivos DICOM relacionados ao paciente selecionado. O Cliente seleciona o arquivo ao qual deseja ter acesso. O Servidor, então, envia todos os dados do arquivo ao Cliente.

O Cliente lê os dados enviados pelo Servidor e os escreve na tela para sua visualização. Para melhor ilustrar os conceitos envolvidos na elaboração do aplicativo, apresentamos os diagramas envolvidos na análise de requisitos do mesmo.

Diagrama de casos de uso

Em seguida, na Figura 4.1, é apresentado o diagrama de casos de uso do projeto em questão. Para sua elaboração foram escolhidos dois atores (Cliente e Servidor) e cinco casos de uso (Seleciona paciente, Envia lista de arquivos, Seleciona arquivo, Envia dados, Visualiza dados). A relação entre eles fica expressada como a seguir.

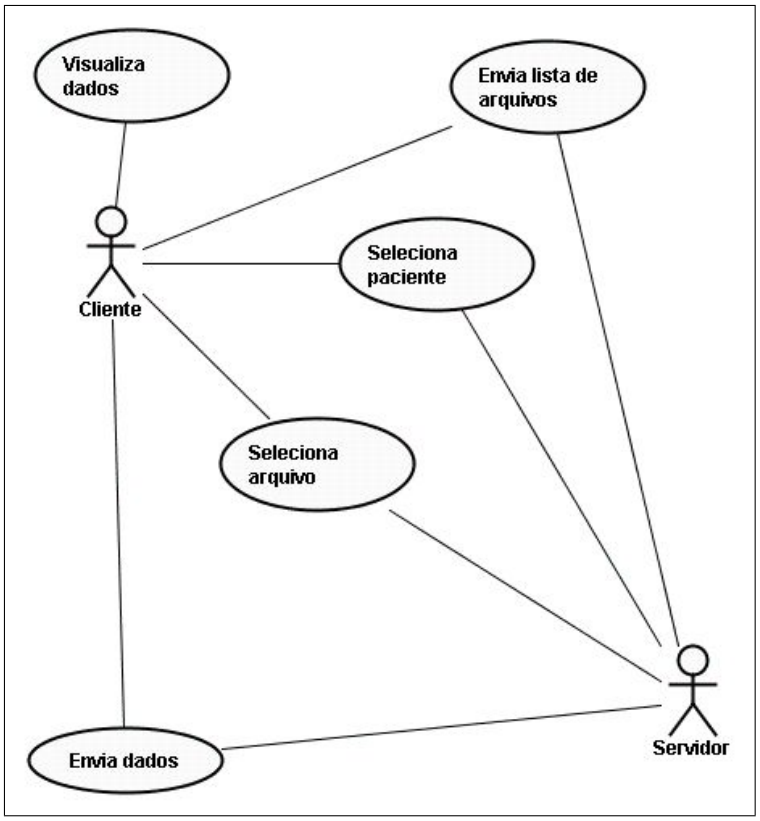

Figura 4.1: Diagrama de casos de uso do sistema

Descrição dos atores:

Cliente: é o aplicativo rodando no dispositivo móvel (PDA), que vai requisitar informações de exames sobre um determinado paciente.

Servidor: é um sistema de tratamento de requisições de arquivos DICOM que roda em uma máquina remota.

Chamamos de Sistema a aplicação que está sendo executada localmente.

Descrição dos casos de uso:

Selecionar paciente:

Cenário principal: Cliente requisita visualização de exame de um paciente

1. Cliente digita nome do paciente
2. Cliente envia ao Sistema nome do paciente
3. Sistema envia ao Servidor nome do paciente

Enviar lista de arquivos:

Cenário principal: Servidor envia ao Cliente lista com arquivos de exames relacionados ao paciente

1. Servidor recebe mensagem com o nome do paciente
2. Servidor busca em banco de dados nomes de arquivos DICOM ligados ao paciente
3. Servidor envia ao Sistema lista com nomes de arquivos
4. Sistema envia ao Cliente lista com nomes de arquivos

Selecionar arquivo:

Cenário principal: Cliente envia ao Servidor nome do arquivo com a imagem a ser visualizada

1. Cliente recebe lista de arquivos
2. Cliente seleciona arquivo desejado
3. Cliente envia ao Sistema nome do arquivo a ser visualizado
4. Sistema envia ao Servidor nome do arquivo

Enviar dados:

Cenário principal: Servidor recebe nome de arquivo e envia os dados referentes a ele ao Cliente

1. Servidor recebe mensagem com o nome do arquivo
2. Servidor abre o arquivo e passa seus dados ao Sistema
3. Sistema envia dados ao Cliente

Visualizar dados:

Cenário principal: Cliente recebe arquivo e visualiza seus dados

1. Cliente recebe dados
2. Cliente visualiza dados

Diagrama de estados

Na Figura 4.2 é mostrado o diagrama de estados do sistema.

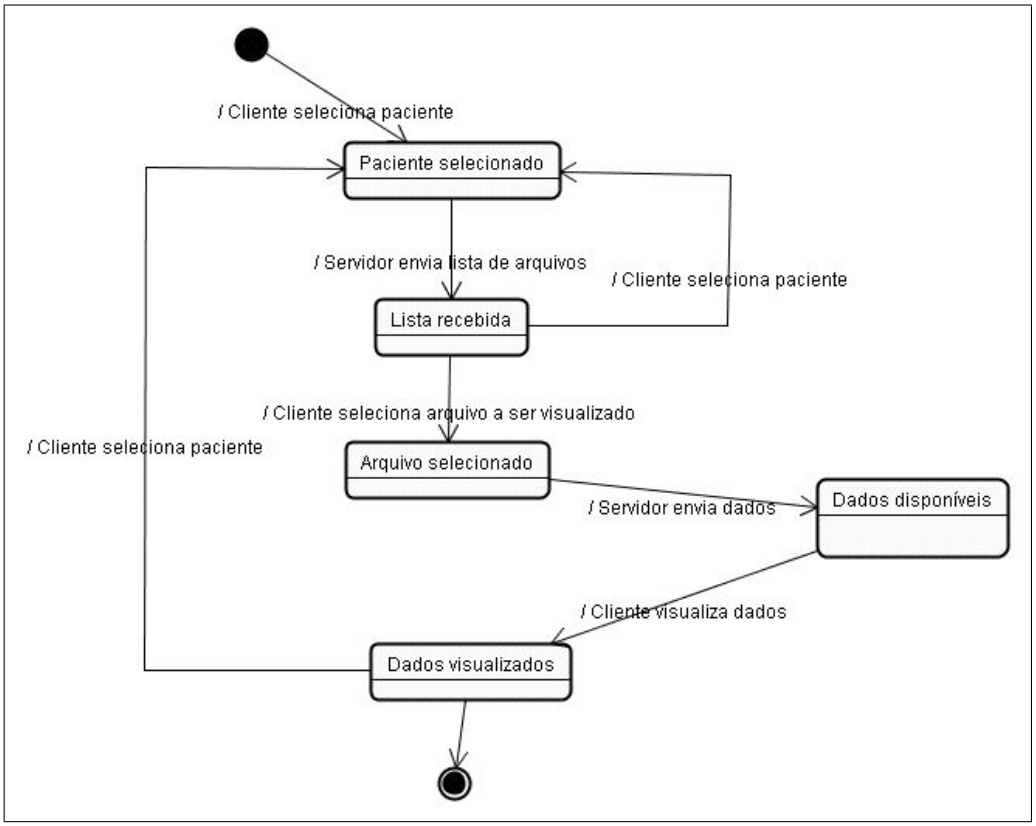

Figura 4.2: Diagrama de estados do sistema

Descrição dos estados do diagrama:

Paciente selecionado: o nome do paciente já foi selecionado pelo Cliente, que aguarda o recebimento da lista de arquivos DICOM associados a este paciente;

Lista recebida: Servidor já enviou a lista de arquivos DICOM referentes ao paciente selecionado pelo Cliente e fica no aguardo da decisão em torno do arquivo cujos dados o Cliente deseja visualizar;

Arquivo selecionado: Cliente já selecionou o arquivo da lista anteriormente passada pelo Servidor e fica no aguardo dos dados referentes ao arquivo selecionado;

Dados disponíveis: Cliente já tem todos os dados do arquivo selecionado para visualização;

Dados visualizados: Cliente tem os dados visualizados na sua tela.

4.2.2 FORMATO DO BANCO DE DADOS

Para armazenar os dados dos pacientes, utilizamos uma base de dados implementada no SGBD MySQL. A idéia desta estrutura é relacionar o nome de um paciente a todos os arquivos DICOM com exames realizados nele.

Tabela 4.1: Formato do banco de dados - tabela paciente-arquivo

Campo	Formato
Id	int
Arquivo	Varchar
Paciente	Varchar

A Tabela 4.1 está organizada da seguinte maneira: um identificador, o nome do arquivo e o nome do paciente examinado. O banco foi estruturado assim para otimizar buscas de arquivos por nome do paciente.

4.2.3 PROGRAMAÇÃO EM CAMADAS

O desenvolvimento da aplicação segue as regras do padrão DICOM de programação de softwares voltados à telemedicina. Como visto na Seção 3.4 do Capítulo 3, o padrão DICOM possui uma arquitetura de camadas que faz a aplicação ser independente do sistema operacional e do protocolo que está operando em sua camada de rede. Seguindo esta lógica foi proposta a arquitetura apresentada na Figura 4.3.

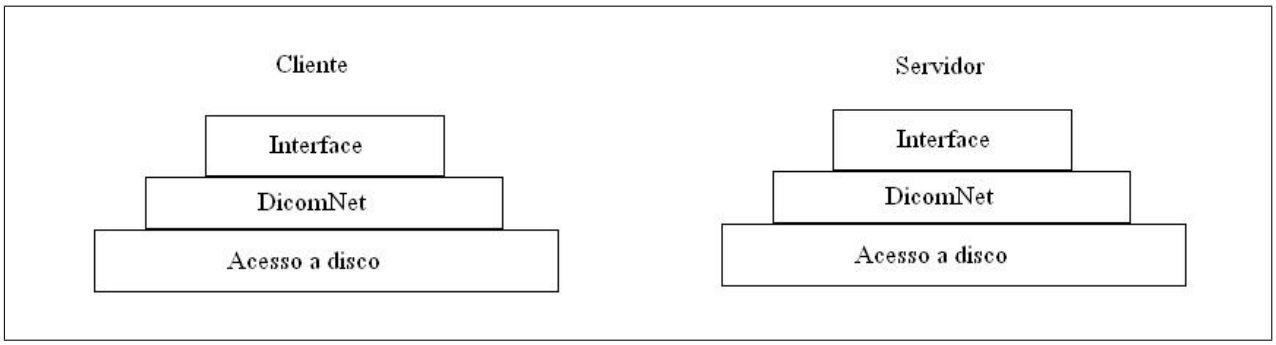

Figura 4.3: Arquitetura em camadas do sistema

Pode-se ver que tanto o Cliente quanto o Servidor possuem a mesma arquitetura.

- Camada de Interface

A camada de Interface refere-se à interação com o usuário. No Cliente, ela foi implementada procurando explorar os componentes suportados pelo sistema operacional do PDA de forma a oferecer facilidade de uso. No Servidor a interface é do tipo console, para mostrar dados relativos a requisições feitas pelo Cliente.

- Camada do protocolo de rede DICOM

Representada pela camada chamada DicomNet na Figura 4.3. Ela implementa as regras de comunicação entre o Cliente e o Servidor utilizando Sockets TCP. Além disso, a camada DicomNet dispara eventos para camadas superiores a ela. Estes eventos ocorrem quando uma conexão é feita e quando dados são recebidos.

- Acesso a disco

Esta camada interage com o Banco de dados e com arquivos no formato DICOM. Portanto podemos dividi-la em duas partes, representadas na Figura 4.4.

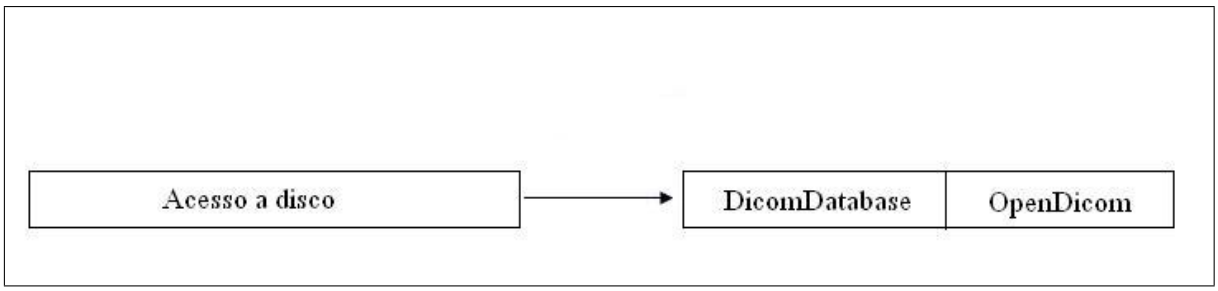

Figura 4.4: Camada de acesso a disco

A classe DicomDatabase representa uma classe responsável pela iteração com o Banco de Dados. Esta classe roda apenas no Servidor e interage com um conector MySQL. Ela recebe como entrada uma query SQL e retorna uma string com valores separados por vírgula.

Para manipular os arquivos DICOM, são usadas as funções das classes do projeto OpenDicom [18]. Este projeto tem código aberto e consiste em bibliotecas que oferecem diversas funcionalidades para abrir, extrair dados e gravar arquivos DICOM.

4.2.4 SERVIDOR

O Servidor executa em uma máquina remota que possui uma base de dados que liga os nomes dos pacientes aos respectivos arquivos DICOM com imagens e informações sobre seus exames.

A aplicação servidora será a responsável por gerenciar as requisições de arquivos DICOM feitas pelo Cliente. Ao receber um pedido de visualização de imagem médica de um Cliente, a aplicação servidora irá:

1. Fazer uma busca ao banco de dados pelos nomes de todos os arquivos que estejam relacionados ao nome do paciente recebido;
2. Preencher uma lista com os nomes dos arquivos retornados da busca e enviá-la à aplicação Cliente;

3. Receber o nome do arquivo cujos dados o Cliente deseja visualizar, e enviá-los ao Cliente.

Para descrever a aplicação servidora em maiores detalhes, abordaremos mais profundamente os passos acima.

Passo 1: Fazer uma busca ao banco de dados pelos nomes de todos os arquivos que estejam relacionados ao nome do paciente recebido

A aplicação cria um socket TCP que roda em modo síncrono e que fica escutando em uma porta, aguardando o recebimento do nome do paciente a ser enviado pela aplicação Cliente. Para isso são utilizadas as funções Bind e Listen da classe Socket, da biblioteca System.Net.Sockets do Visual Studio .NET.

A chamada da função Listen faz com que o Servidor fique aguardando o Cliente entrar em contato para então estabelecer uma conexão TCP através da qual as duas aplicações passam a comunicar-se. A função Receive faz com que o socket comece a receber os dados vindos do Cliente (neste caso, o nome do paciente digitado no PDA). Quando os dados são recebidos, uma função é chamada para o processamento da resposta.

Ao receber os dados através do socket, a aplicação pode, então, ler o nome do paciente e fazer a busca pelos arquivos armazenados em banco com exames feitos no determinado paciente.

Como anteriormente dito, a base de dados está implementada em MySQL, e para manipulação dos dados por parte da aplicação utilizamos o pacote MySql.Data, na qual constam as classes MySqlConnection, utilizada para estabelecer a conexão com o banco; MySqlCommand, para realizar as consultas; e MySqlDataReader, para ler os dados retornados da consulta.

A classe DicomDatabase é a responsável pelo acesso e pelas consultas ao banco. Ela recebe a consulta SQL com o nome do paciente e retorna uma resposta no formato *Comma Separated Values* (CSV) [4], em que os nomes dos arquivos são separados por vírgula.

Passo 2: Preencher uma lista com os nomes dos arquivos retornados da busca, e enviá-la à aplicação Cliente

Enquanto lê os resultados retornados da consulta ao banco de dados, a aplicação preenche uma lista com os nomes dos arquivos. Essa lista é armazenada e enviada, via Socket TCP, à

aplicação Cliente. Para realizar o envio é criado um socket que, através da função Write, da classe NetworkStream da biblioteca System.Net.Sockets, enviará a lista de arquivos ao Cliente.

A função Write faz com que o socket envie a lista ao Cliente. Quando os dados são enviados, o socket é liberado para voltar a escutar na porta pré-definida para a comunicação com a aplicação do PDA, aguandando o arquivo a ser requisitado para visualização.

Passo 3: Receber o nome do arquivo cujos dados o Cliente deseja visualizar, e enviá-los ao Cliente

O Servidor cria um socket para recepção de dados, através da função Read da classe NetworkStream e, através dele, recebe o nome do arquivo selecionado pelo Cliente. Este arquivo estará armazenado no próprio Servidor. O Servidor abre o arquivo em formato DICOM e extrai os dados a serem enviados para a aplicação Cliente.

Para manipulação do arquivo DICOM, são utilizadas as classes da biblioteca OpenDicom [18]. Primeiro a aplicação abre o arquivo e retira deste todos os dados presentes no arquivo DICOM.

Depois é criado um socket TCP para envio de dados (como no passo 2) e, através dele, a aplicação servidora vai enviando, em pacotes de 1024 bytes, os dados do arquivo DICOM à aplicação Cliente. Primeiro são enviados dados como as dimensões do arquivo, nome do médico e a data do exame e logo depois é enviada a imagem para visualização.

4.2.5 CLIENTE

A aplicação Cliente rodará em um PDA e será a responsável por requisitar as informações do Servidor e visualizá-las em uma tela como a seguir.

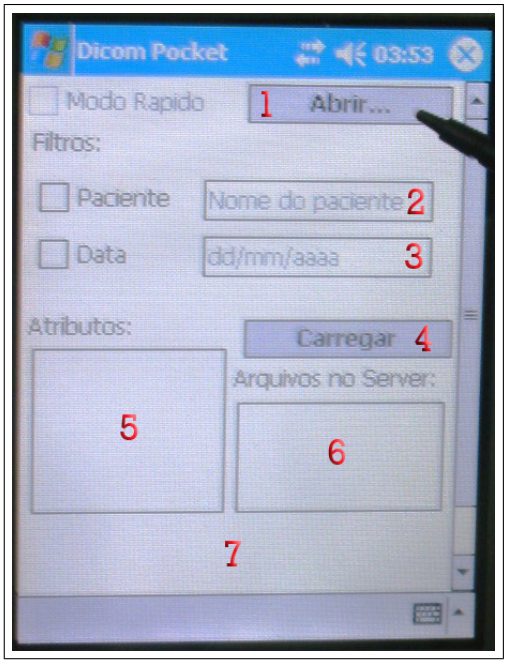

Figura 4.5: Tela inicial do aplicativo

Na Figura 4.5, temos a tela inicial do aplicativo. Os números representam:

1 - Botão para abrir arquivos locais que já estejam armazenados no PDA.

2 - Caixa onde é digitado o nome do paciente.

3 - Data da imagem a ser exibida.

4 - Botão de carregar, responsável por solicitar a lista de arquivos referentes ao paciente.

5 - Lista onde vão aparecer os atributos referentes ao arquivo DICOM que vai ser visualizado.

6 - Lista de arquivos DICOM referentes ao paciente a ser pesquisado.

7 - Painel onde será exibida a imagem do exame.

A aplicação Cliente fará a seguinte seqüência de passos:

1 - Estabelecer uma conexão com o Servidor;

2 - Digitar o nome do paciente e enviá-lo ao Servidor;

3 - Receber os nomes dos arquivos com exames feitos no paciente e visualizá-los;

4 - Selecionar o arquivo a ser visualizado e enviar seu nome ao Servidor;

5 - Receber os dados do arquivo vindos do Servidor e visualizá-los.

Para descrever a aplicação Cliente em maiores detalhes, abordaremos mais profundamente os passos acima.

Passo 1: Estabelecer uma conexão com o Servidor

Para conectar-se ao Servidor, a aplicação Cliente cria uma conexão TCP através da qual será possível estabelecer uma comunicação com a aplicação servidora. O processo de estabelecimento da conexão é feito de modo assíncrono através das funções BeginConnect e EndConnect, da classe Socket. A chamada da função BeginConnect faz com que o socket da aplicação Cliente envie uma requisição de conexão à aplicação servidora. Quando feita a conexão, é chamada uma função, onde é feita uma chamada para a função EndConnect, que termina o processo de estabelecimento da conexão.

Passo 2: Digitar o nome do paciente e enviá-lo ao Servidor

Com o estabelecimento da conexão, é possível enviar as requisições e receber os dados do Servidor. Para isso, o Cliente primeiro digita o nome do paciente no campo "Nome do paciente"e depois clica no botão "Carregar".

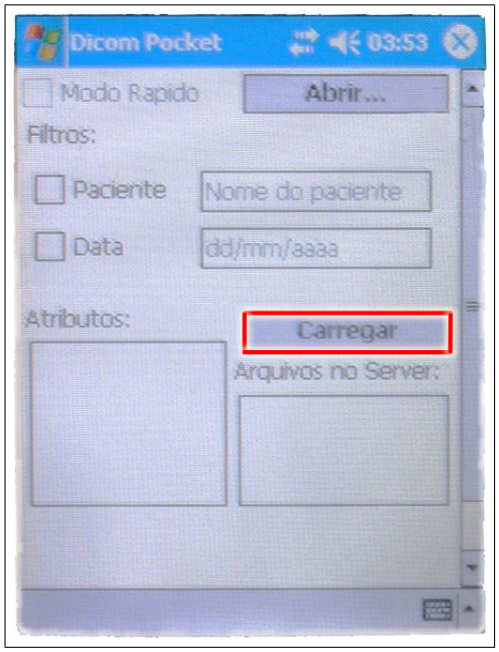

Figura 4.6: Botão carregar

A aplicação cria um socket TCP que será responsável pelo envio de informações à aplicação servidora. O processo de criação do socket é feito de forma assíncrona com chamadas às funções BeginSend e EndSend da classe Socket. O Cliente preenche um array de bytes com o nome do paciente e, através do socket TCP, envia este dado à aplicação servidora.

Passo 3: Receber os nomes dos arquivos com exames feitos no paciente e visualizá-los

Agora a aplicação cria um socket para recebimento de informações da aplicação servidora. O processo de criação deste socket é feito através da função Read da classe NetworkStream. Através deste socket, o Cliente vai recebendo os nomes dos arquivos do Servidor e colocando-os numa lista.

Após o preenchimento da lista, a mesma é varrida e os nomes dos arquivos vão sendo inseridos na lista "Arquivos no Server".

Passo 4: Selecionar o arquivo aser visualizado e enviar seu nome ao Servidor

Com a lista preenchida, agora é possível para o usuário definir qual exame deseja visualizar. Para isto, basta selecionar o arquivo correspondente na lista de arquivos.

Quando uma opção na lista é selecionada, é gerado um evento, dentro do qual o nome do arquivo é lido e o socket criado para envio de dados ao Servidor manda o nome do arquivo à aplicação servidora.

Passo 5: Receber os dados do arquivo vindos do Servidor e visualizá-los

Após comunicar ao Servidor o nome do arquivo a ser visualizado, o Cliente passa a esperar o envio dos dados por parte do Servidor através do socket criado para o recebimento dos dados.

Depois de receber todos os dados do arquivo DICOM enviados pelo Servidor, o Cliente armazena estes dados em memória, para depois acessá-los. Por último, os dados são carregados e visualizados nos campos correspondentes na interface.

Se preferir, o Cliente pode fazer uma nova busca digitando o nome de um novo paciente e fazendo a requisição dos dados. Outra alternativa é selecionar outro arquivo da lista para visualização de outro exame do determinado paciente.

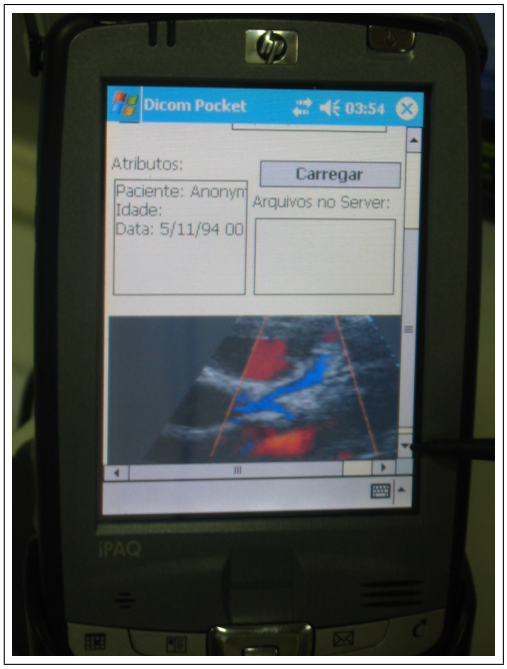

Figura 4.7: Arquivo DICOM visualizado na aplicação

Uma última opção é o Cliente abrir algum arquivo armazenado no próprio PDA. Isto é feito através do botão “Abrir”, conforme ilustrado na Figura 4.8.

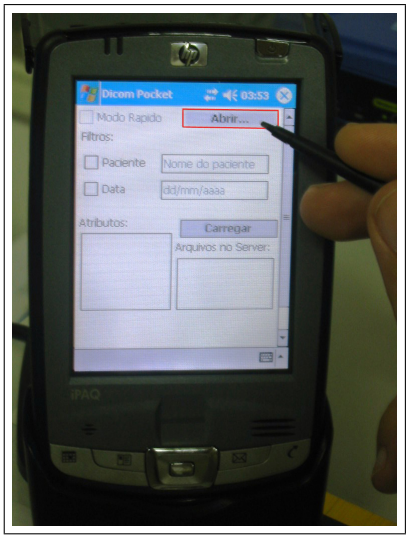

Figura 4.8: Botão abrir

4.3 ANÁLISES DE DESEMPENHO

4.3.1 CENÁRIO DE TESTES

Para realização dos testes foi utilizado um servidor dedicado com apenas um cliente acessando-o. Uma rede exclusiva para o projeto foi criada, conectando-se o Servidor a um roteador constituído por interfaces sem fio. Sendo assim, o tráfego gerado na rede foi apenas o tráfego da aplicação.

Foram feitas diversas requisições a arquivos com tamanho médio de 900KB aproximadamente.

4.3.2 RESULTADOS

Foram medidos o consumo de memória e o tempo médio relativos ao atendimento das requisições mencionadas na Sessão 4.3.1. Na Tabela 4.2 estão representados os resultados obtidos.

Tabela 4.2: Tabela de análises de consumo de memória e tempo de envio de dados

Tamanho da aplicação	90,5 KB
Consumo de memória (programa ocioso)	60 KB
Consumo de memória (programa carregando lista de arquivos)	60 KB
Consumo de memória (programa carregando imagem de 900 KB)	1,0 MB
Tempo de envio de dados do servdor ao Cliente	2,0 s

Como podemos ver, a aplicação é leve, portanto perfeitamente compatível a um ambiente PDA, que requer pouco consumo de memória. O arquivo executável ocupa apenas 90,5 KB de disco, com um consumo médio de 60 KB de memória quando ocioso. Quando carregando a lista de arquivos o consumo quase não se altera, exigindo ainda muito pouco do processamento do dispositivo. Quando carrega a imagem a memória apresenta maior uso, porém ainda a escalas muito baixas. Outro dado importante é que embora tenhamos testado em uma rede exclusiva com servidor dedicado, o tempo de resposta de 2s denota a possibilidade de implementação em maior escala.

CAPÍTULO 5 - CONCLUSÃO

5.1 SOBRE O USO DA TELEMEDICINA E DAS REDES EM MALHA

A telemedicina está em constante evolução, em que novos projetos e aplicações que englobam o seu uso vêm sendo gerados. O padrão DICOM [5] estabelece uma série de regras a ser seguidas por quaisquer aplicações que desenvolvam a telemedicina. Através dele, duas aplicações completamente distintas espalhadas em áreas geograficamente distantes, utilizando protocolos em camadas de rede diferentes, podem transferir imagens médicas de forma transparente. O principal objetivo da telemedicina, segundo a Organização Mundial de Saúde, é prover um atendimento médico de boa qualidade à distância [6].

Este trabalho mostra a visualização e transferência de imagens médicas com dados sobre exames feitos em pacientes. Outros exemplos de aplicações de telemedicina oferecidos atualmente no Brasil são [6]: videoconferência médica, segunda opinião, telediagnóstico por imagem e consulta on-line, entre outros.

Assim como a telemedicina, o uso de redes móveis sem fio também tem sido alvo de estudos e pesquisas nos últimos anos. Todos os anos, artigos e investigações são gerados apresentando propostas para aumentar a qualidade deste tipo de serviço. A mobilidade traz comodidade, o cliente que se conecta à rede não tem o seu espaço limitado pela extensão de um cabo, pode mover-se livremente e ainda assim permanecer conectado. Porém, ainda há dificuldades para escolha de um algoritmo de roteamento que seja o mais eficiente e que produza menos erros na rede móvel utilizada. Ainda falta, também, uma maneira de oferecer qualidade de serviço e segurança a um ambiente sem fio, muito suscetível à perda de enlaces e pacotes e a invasões indevidas.

Uma alternativa de implementação de uma rede móvel é a chamada rede em malha, ou *mesh* [1]. Em uma rede *mesh*, os pontos de acesso são os responsáveis pela escolha e processamento do protocolo de roteamento. Com isso, os nós clientes só participam do repasse, o que gera menos volume de processamento para eles. Através de redes *mesh*, eles também podem se comunicar com outras redes e trocar informações com elas. Como exemplo, podemos citar um

profissional que está em um hospital equipado com rede *mesh* e deseja comunicar-se com um especialista que está em outro lugar que utiliza outro tipo de rede, trocando informações com este.

Aqui neste trabalho podemos verificar a utilização de uma aplicação funcional que une os dois conceitos. É proposta uma solução para utilização da telemedicina em redes móveis. É extremamente vantajoso para um profissional da área de medicina poder ter o seu trabalho facilitado pela eficiência de um visualizador de exames médicos adicionado à comodidade de poder mover-se livremente sem deixar de receber seus dados em um dispositivo leve e fácil de ser carregado.

5.2 RESULTADOS ENCONTRADOS E LIMITAÇÕES

Com esta aplicação, já podemos ter uma perspectiva sobre a manipulação de imagens médicas em pequenos dispositivos, além de sua transferência desde um servidor remoto, através da utilização de uma rede *mesh*. Os dados são passados de forma eficiente e todas as suas informações são visualizadas de forma a facilitar a vida do usuário.

Os resultados encontrados foram satisfatórios, porém ainda há muito a melhorar. A aplicação, na forma como foi desenvolvida, oferece apenas a opção de visualização da imagem como um todo, em seu tamanho real. Ou seja, não é possível, na solução apresentada, fazer zoom de alguma parte da imagem, ampliando o grau de detalhamento da mesma. Para tanto, uma solução a ser proposta seria que, no caso do usuário selecionar uma parte da imagem a ser ampliada com zoom, o servidor enviasse apenas esta parte da imagem para ser vista em um zoom maior, redimensionando esta determinada parte para o tamanho da imagem completa. Isso possibilitaria uma visualização com maior grau de detalhamento do pedaço da imagem que o médico deseje analisar mais profundamente, aumentando a eficiência do diagnóstico.

Outra limitação encontrada foi na visualização de algumas imagens. Nossa solução teve seu foco mais voltado na parte de comunicação e transferência das imagens médicas, não tendo muito foco em relação à visualização das imagens. Algumas imagens DICOM recolhidas não puderam ser visualizadas com nitidez. Um projeto com foco maior na visualização de imagens poderia aproveitar este trabalho e implementar esta parte.

5.3 PERSPECTIVAS

As redes sem fio e suas variações são tendências fortes para o futuro das telecomunicações. Aplicações que explorem tais redes de maneira a trazer benefícios à sociedade merecem destaque.

No âmbito da medicina, este cenário pode ser explorado em hospitais ou em áreas abertas. Hospitais podem ter médicos dotados de PDAs podendo acessar informações de pacientes dentro da rede do hospital. Ou ainda, um paramédico dotado de um PDA em um local com um ponto de acesso poderia acessar bases de dados integradas ou consultar especialistas para obter informações relevantes.

A visualização e transferência de imagens médicas através de redes móveis foi o objeto deste trabalho. Trabalhos futuros poderão expandi-lo, oferecendo buscas mais robustas e completas em bases de dados. Outra melhora que pode ser vislumbrada é a comunicação constante e em tempo real entre o portador do PDA e outro indivíduo remotamente hospedado. Com isso, podemos ter uma maior eficiência no atendimento a casos emergenciais em ambiente hospitalar a baixo custo e com qualidade.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *Comput. Netw. ISDN Syst.*, 47(4):445–487, 2005.

[2] Visual C#. http://msdn.microsoft.com/vstudio/express/visualcsharp/, 2007. Acessado em 15-05-2007.

[3] T. Clausen and Optimized Link State Routing Protocol (OLSR) Internet Engineering Task Force RFC Experimental 3626 P. Jacquet, March 2003.

[4] Comma Separated Values CSV. http://en.wikipedia.org/wiki/comma-separated_values, 2007. Acessado em 30-06-2007.

[5] MySQL Customers. http://www.mysql.com/customers/, 2007. Acessado em 20-06-2007.

[6] Rede Nacional de Telemedicina. http://rute.rnp.br/sobre/telemedicina/, 2006. Acessado em 22-06-2007.

[7] DICOMWorks. http://dicom.online.fr/, June 2002. Acessado em 11-06-2007.

[8] MySQL em: http://pt.wikipedia.org/wiki/MySQL, 2007. Acessado em 12-06-2007.

[9] S.L Fritz. Dicom standardization. pages 311–321, 1999. Filmless Radiology, Spring-Verlag.

[10] DICOM Introduction. http://www.sph.sc.edu/comd/rorden/dicom.html. Acessado em 20-06-2007.

[11] J. Kurose and K. Ross. *Redes de Computadores e a Internet*. Pearson Addison Wesley, third edition, 2006.

[12] Projeto Marfim. http://mesh.ic.uff.br, 2006. Acessado em 12-06-2007.

[13] D. N. B. Monteiro. Estudo sobre a visualização de imagens médicas obtida por exames virtuais, 2006. Dissertação de Mestrado. Universidade Federal Fluminense. Niterói-RJ.

[14] MySQL. http://www.mysql.com, 2007. Acessado em 20-06-2007.

[15] National Electrical Manufactures Association (NEMA). http://medical.nema.org, 2007. Acessado em 15-05-2007.

[16] Visual Studio .NET. http://www.msdnbrasil.com.br/produtos/vstudio/, 2004. Acessado em 15-05-2007.

[17] Nortel. http://www.nortel.com/corporate/news/newsreleases/2004d/11_17_04_taipei_city.html, 2004. Acessado em 24-03-2007.

[18] OpenDicom.NET. http://opendicom.sourceforge.net/, 2007. Acessado em 22-06-2007.

[19] OpenWrt. http://openwrt.org/, March 2007. Acessado em 03-03-2007.

[20] Diego Passos, Douglas Vidal Teixeira, Débora C. Muchaluat-Saade, Luiz C. Schara Magalhães, and Célio Albuquerque. Mesh network performance measurements. pages 48–55, 2006.

[21] O. Ratib and H. Hoehn. Papyrus 3.0: the dicom compatible file format. 1993. Digital Imaging Unit, Center of Medical Informatics, University Hospital of Geneva, Geneva, Switzerland.