Nelson Abu Samra Rahal Junior

Modelagem e Implementação de Conexão a Banco de Dados Relacional para o DicomEditor

Florianópolis – SC

2000

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

Nelson Abu Samra Rahal Junior

Modelagem e Implementação de Conexão a Banco de Dados Relacional para o DicomEditor

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

Prof. Dr. rer. nat. Aldo von Wangenheim

Florianópolis, Novembro de 2000

Modelagem e Implementação de Conexão a Banco de Dados Relacional para o DicomEditor

Nelson Abu Samra Rahal Junior

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

Prof. Dr. rer. nat. Aldo von Wangenheim

Prof. Dr. rer. nat. Aldo von Wangenheim
Av. Trompowski, 475 - Florianópolis
awangenh@inf.ufsc.br

Prof. Fernando A. O. Gauthier, Dr. – Coordenador do Curso

Banca Examinadora

Prof. João Bosco Mangueira Sobral, Dr.

Prof. Roberto Willrich, Dr.

Euclides de Morais Barros Jr., MSc

Dayna Maria Bortoluzzi, MSc

"Preocupa-me saber que certos jovens dizem que não querem ingressar na faculdade porque eu não me formei. Em primeiro lugar, recebi uma educação muito boa, ainda que não tenha ficado o suficiente para tirar o meu diploma. Além disso, o mundo está se tornando mais competitivo, especializado e complexo a cada ano, fazendo com que a educação universitária seja, hoje em dia, tão crucial como um dia o foi a educação secundária."

Bill Gates

Ofereço este trabalho a todos os integrantes do projeto de pesquisa Cyclops, pelas nossas dedicações na procura de soluções na área de informatização medica, nossos projetos de pesquisas são uma pequena semente para um futuro melhor.

Gostaria de agradecer a todos que de uma forma ou de outra vem auxiliando em meus estudos e desbravamentos pessoais que estou fazendo para seguir dentro da pesquisa.

A minha esposa, obrigado pela sua compreensão pelas noites que venho dedicando a esses estudos e pelo apoio concedido.

A minha filha Marjury, é também por você que estou fazendo isso.

Ao prof. Aldo, obrigado pelo seu empenho e confiança depositados sobre a minha pessoa, espero continuar sempre retribuindo.

Ao Silvio, pelas correções feitas sempre que possível.

Ao Eros, sem seu apoio não teria vindo a Florianópolis fazer o mestrado.

Ao Junior (Peixe) pela sua ajuda na motivação de conclusão deste trabalho, mostrando sua continuidade em outros projetos.

E, outros que não foram citados mas que contribuíram ou estão contribuindo nesta pesquisa, o meu muito obrigado.

SUMÁRIO

RESUMO

Este trabalho de foi realizado no escopo do Projeto Cyclops, que é um Projeto de Cooperação Brasil-Alemanha direcionado ao desenvolvimento inteligente para suporte de diagnósticos médicos.

Desta maneira, o software DicomEditor tem a liberdade de optar pelo uso de um software gratuito, que acompanha o protocolo DICOM que é o CTN ou o uso do banco de dados ORACLE.

Este trabalho acrescenta a conexão do banco de dados ORACLE ao software DicomEditor, implementado em Smalltalk. Este software tem a funcionalidade de ser um cliente de rede DICOM. DICOM é um protocolo de armazenamento e recuperação de imagens médicas utilizadas em larga escala por hospitais e grandes clinicas médicas. Através deste trabalho apresentaremos os procedimento de operação das classes de conexão do DicomEditor ao banco de dados ORACLE, juntamente com o modelo relacional criado no banco de dados para armazenar as informações.

ABSTRACT

This work was a part of the CyclopsProject, which is a German-Brazilian Cooperation Program that develops softwares for medical diagnosis support.

It will be presented the conexion methods between the DicomEditor and Oracle ® as well the relational model created on the database to save the information. Thus, the DicomEditor software will be given the option to interface with a free software known as CTN or a Oracle ® Database System.

This work is based on the implementation of a interface between Oracle ® database system and the DicomEditor software using Smalltalk language. The DicomEditor software has been succesfully used as a DICOM netork client. DICOM is a protocol used to perfom the storage and recovery of medical images and has been extensively adopted by hospitals and medical centers.

1.0 - INTRODUÇÃO

Com os avanços tecnológicos na área médica, estão sendo produzidos cada vez mais equipamentos que utilizam recursos computacionais na aquisição de imagens médicas. Estas imagens, hoje, já possuem uma padronização em termos do protocolo DICOM. Através da aquisição destas imagens o projeto de pesquisa Cyclops pode realizar operações utilizando técnicas de visão computacionais.

As imagens originais e as novas imagens geradas pelo tratamento de técnicas de visão computacional devem ser armazenadas para sua futura recuperação.

O armazenamento destas imagens através do padrão DICOM é realizada utilizando-se um software CTN (Central Test Node), que permite gravar e recuperar os dados referente aos pacientes, estudos destes pacientes e a série de imagens que cada estudo possui, juntamente com as informações necessárias para estes dados estarem de conformidade com o padrão DICOM.

Para se ter uma autonomia tecnológica, o projeto Cyclops criou um software chamado DicomEditor que trabalha com padrão do protocolo DICOM, permitindo desta maneira através do software CTN armazenar e recuperar imagens médicas.

O software CTN grava os dados em banco de dados miniSQL, um banco de dados de ambiente Unix e que possui limitações de gerenciamento dos dados. As imagens gravadas não são armazenadas dentro de uma estrutura de tabela, sendo armazenadas em diretórios no disco rígido.

Com a utilização do software DicomEditor junto com o aplicativo CTN o processo de instalação e configuração é uma tarefa árdua e com a exigência de profissionais capacitados, sendo a necessidade destes profissionais um problema para a colocação dos produtos do projeto Cyclops no mercado corporativo.

Hoje so podemos utilizar o software DicomEditor com o aplicativo CTN, sendo que o ideal seria a opção por outras ferramentas que possibilitassem o armazenamento e recuperação dos dados.

2.0 - OBJETIVO

Este trabalho possui o objetivo de se conectar o aplicativo DicomEditor a um banco de dados ORACLE.

Para esta conexão se faz necessário à execução das seguintes tarefas:

[obj.1] - Possibilitar que o usuário realize a seleção no DicomEditor do módulo de armazenamento de dados desejado (CTN ou ORACLE);

[obj.2] - Criar uma modelagem de dados que possa armazenar as informações de configurações e dados existentes no padrão DICOM;

[obj.3] - Desenvolver um conjunto de classes que possa receber os dados das classes do DicomEditor e converte-los a instruções em SQL para serem armazenados e recuperados;

[obj.4] - Modelar as classes de conexão ao banco de dados ORACLE para que possa ser utilizada por futuros módulos de conexão em ODBC;

[obj.5] - Validar o módulo de conexão com banco de dados ORACLE, tendo certeza que as informações armazenadas podem ser transferidas para o software CTN e retornadas ao banco de dados ORACLE sem perda de nenhum dado.

[obj.6] – Teste de performance para se ter uma visão se o novo modulo de armazenamento em banco de dados no ORACLE é mais rápido que o modelo existente ou se existe uma queda de velocidade com este novo mecanismo.

[obj.7] – Criação de ícones para uma visualização rápida das imagens. Sendo estas imagens carregadas no tamanho correto se o usuário desejar.

3.0 - JUSTIFICATIVA

Com a necessidade de se criar um mecanismo que possibilite os produtos gerados pelo projeto Cyclops serem de fácil instalação e manutenção, bem como a necessidade de se trabalhar com ferramentas que possuam profissionais no mercado com experiência na sua manutenção, o projeto Cyclops iniciou-se uma remodelagem de seus aplicativos.

Nesta remodelagem se fez necessário um estudo do padrão DICOM, padrão este que possui carência de profissionais no mercado, mais que é um padrão aceito pela comunidade científica e privada.

Estando o DicomEditor em conformidade com o padrão DICOM e trabalhando em perfeito estado com o software CTN, o desafio deste projeto foi criar um mecanismo de portabilidade dos dados manipulados pelo DicomEditor a uma nova ferramenta, substituindo o aplicativo CTN, mais tendo o funcionamento do DicomEditor idêntico ao existente, tanto para o usuário como para o mínimo possível de alterações em seu código fonte.

4.0 - OBJETIVOS ESPECÍFICOS

Para que se possamos atingir os objetivos apresentados deve ser feito:

[obj.1] – Na interface do aplicativo DicomEditor deve ter a opção de escolha entre o software CTN e o banco de dados. Onde a opção de banco de dados deve abrir uma tela perguntado se é acesso nativo ao ORACLE ou acesso via driver ODBC a outros bancos de dados. No aplicativo DicomEditor as opções de "Load" e "Save" dos dados deverão sempre checar o tipo de conexão ativa antes de efetuar a operação.

[obs.2] – Através do estudo das classes existentes no aplicativo DicomEditor, devera ser criado um modelo de entidade de dados (DER) para comportar o armazenamento dos dados, conforme a padronização DICOM. Para que se possa mapear as informações, se faz necessário uma simulação colocando os dados em tabelas de banco de dados e recuperando no aplicativo DicomEditor, observando desta maneira os resultados. Sendo estes resultados satisfatórios iremos realizar o processo inverso que é a gravação.

[obj.3] – Para que possamos armazenar e recuperar os dados, é necessário a criação de um conjunto de classes e métodos que possam converter as informações armazenadas no Smalltalk em instruções SQL , bem como converter as informações em SQL para o formato do Smalltalk.

[obj.4] – Para a não existência de redundância de código fonte em Smalltalk, o projeto deverá ser implementado com a biblioteca EXDI que possui comandos genéricos para acesso nativo ao banco de dados ORACLE e acesso a outros bancos de dados através de conexão com driver ODBC.

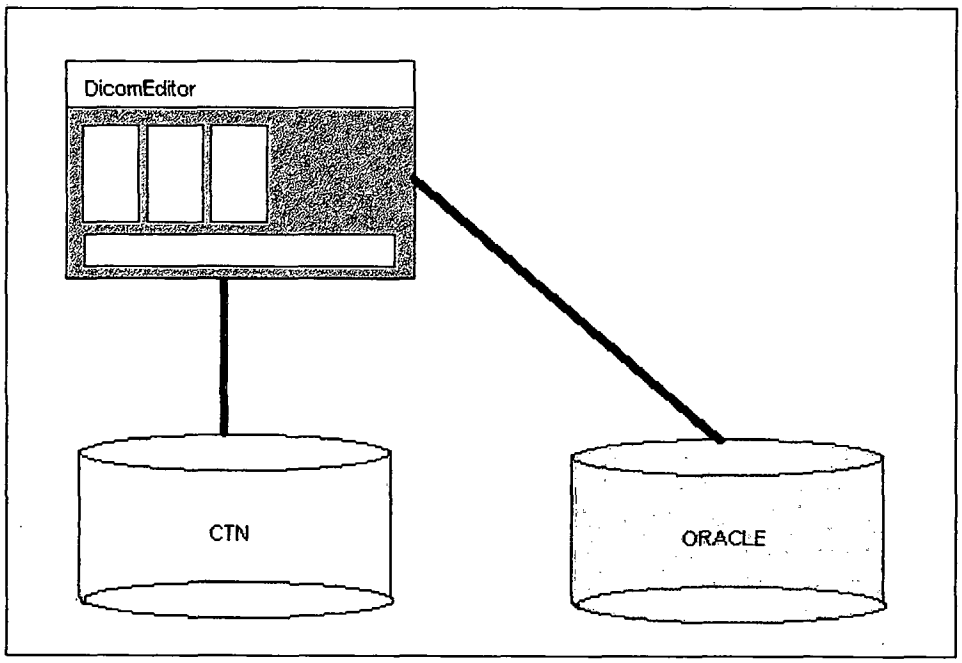
[obj.5] – Realizando a carga das imagens do software CTN e gravar em seguida no banco de dados ORACLE, para termos uma visão se os dados estão sendo armazenados corretamente, e através da recuperação dos dados do banco de dados ORACLE e testes no aplicativo DicomEditor com estes dados teremos um parecer se as informações estão consistentes ou não.

[obj.6] – Uma vês o projeto estando pronto, poderemos ter um controle de tempo de carga de dados tanto do aplicativo CTN como do banco de dados ORACLE. Para que

possamos verificar quais os gargalos de demora para uma reestruturação dos algoritmos existentes.

[obj.7] – Hoje o software CTN realiza apenas a gravação das imagens em seu tamanho original, para que o usuário possa recuperar estas imagens de maneira mais rápida o software DicomEditor com o banco de dados ORACLE deverá implementar um mecanismo de gravação das imagens no tamanho 64x64, sendo estas imagens denominadas de ícones, se o usuário tiver interesse pelas imagens após a carga dos ícones, será disponibilizado uma opção de recuperação das imagens no seu tamanho original.

Figura 1 – Aplicativo DicomEditor se conectando ao banco de dados ORACLE e ao software CTN.



5.0 – ANÁLISE DE REQUISITOS

Este capitulo, descreve a análise de requisitos necessária para o desenvolvimento do trabalho. Esta análise de requisitos está em conformidade com os objetivos específicos, já descritos. No que se refere a estes objetivos, o sistema deverá:

[req.1] – Para a realização do objetivo 1, se faz necessário uma remodelagem da interface, sendo adicionados componentes visuais de escolha de opções para o usuário, bem como a remodelagem dos menus prevendo o novo modulo a ser conectado ao DicomEditor, também se faz necessário a implementação de uma tela de "login" para o banco de dados, pedindo usuário, senha, ambiente de conexão (environment) e o tipo de conexão: ORACLE nativo ou driver ODBC. Nesta nova interface será apresentado ao usuário qual o "login" e "environment" ativos. Nesta apresentação esta inclusa a apresentação da porta de conexão e nome do servidor para acesso a base de dados via CTN.

[req.2] – No objetivo 2, se faz necessário o estudo de todas as classes existentes no DicomEditor. Mapeando as variáveis de classe, para que estas possam ser transformadas em modelo relacional de dados (DER). Para que se possa mapear estes variáveis de classe são necessários através das ferramentas de "debug" e "inspect" ficar analisando as informações existentes nas classes e suas variáveis. Podemos verificar estas informações por meio de carga dos dados do CTN e através de simulações de carga de dados do banco de dados ORACLE através de tabelas de testes.

[req3.] – Conforme a necessidade do objeto 3, se faz necessário a criação de pequenos softwares que utilizem as bibliotecas de acesso ao banco de dados em Smalltalk para se ter um domínio sobre a ferramenta. Com estes pequenos programas poderemos reutilizar o código para a implementação definitiva no DicomEditor. Se faz necessária a criação de códigos que comportem os comandos em SQL de INSERT, UPDATE e SELECT.

[re1.4] – No objetivo 4, tratamos a necessidade da não redundância do código fonte para acesso nativo a ORACLE e via driver ODBC, embora o acesso ao banco de dados mediante ao driver ODBC não faça parte deste trabalho, ele deve ser tratado para os trabalhos futuros do projeto Cyclops. Para que este requisito seja realizado foi

utilizada a tecnologia de acesso a base de dados mediando a biblioteca EXDI, e não pela biblioteca "Lens" que não realiza acesso a drive ODBC.

[req.5] – Os testes no aplicativo DicomEditor é o requisito necessário para o objetivo 5. Para estes testes se faz necessário uma amostra de imagens no software CTN para que o DicomEditor possa carregar e gravar no banco de dados ORACLE. Uma vez gravado no banco de dados o caminha reverso deve ser feito.

[req.6] – Na solução do objetivo 6, teremos que colocar no código fonte controle de tempo, para que seja registrado o tempo de carga de uma imagem e o tempo de término desta carga, desta maneira teremos as informações necessárias para sabermos qual modulo é mais rápido, o CTN ou banco de dados ORACLE.

[req.7] – Para a conclusão do objetivo 7, se faz necessário a implementação de uma base de dados específica para armazenamento de uma copia de cada imagem em forma de ícone, sendo este ícone de tamanho 64x64. Quando for realizado o processo de gravação de uma série de imagens, esta série deverá executar um algoritmo que possibilite a sua copia em tamanho reduzido. Ficando desta maneira a imagens armazenada em seu tamanho original e no tamanho reduzido. O processo de carga da imagem do banco de dados, devera ter uma opção de recuperação da imagem no seu tamanho original quando o usuário fizer a solicitação.

6.0 - REVISÃO DA LITERATURA

Para a realização de um projeto de software com o mínimo possível de erros de analise e implementação PRESSMAN (1995) apresenta o modelo aspiral de desenvolvimento de software.

REZENTE (1999), traz em seu livro de maneira superficial os aspectos necessários para uma visualização de custo de produção e mensuração de qualidade.

CHEN (1990), apresenta o modelo relacional de organização de dados, estas técnicas são trabalhadas e aprofundadas por COUGO (1997) e BARBIERI (1994) onde eles apresentam técnicas diferentes de desenhos para representação do modelo relacional, bem como um estudo aprofundado de como realizar as formas normais de normalização de dados.

Se um trabalho tem como pré-requisito o uso de banco de dados, não podemos deixar de fora DATE (1989), onde hoje ele é uma referencia mundial. Nesta obra DATE apresenta um esboço da utilização de comandos SQL. Para uma comparação e um estudo mais aprofundado também foi utilizado o livro FREEZE (1998), onde são apresentados os comandos SQL em pelo menos 5 bancos de dados Cliente/Servidor, para a comparação e visualização de suas características próprias.

O banco de dados utilizando neste projeto foi ORACLE. Sendo necessário o estudo desta ferramenta bem como a analise de sua filosofia de trabalho, para isso se deu à necessidade de um levantamento de livros disponíveis, onde BURLESON (1996) apresenta as características do banco de dados em sua versão 8, banco de dados distribuídos e ferramentas de ganho de produtividade na administração e gerenciamento. RAMALHO (1997) se torna uma leitura obrigatória, pois apresenta as características mínimas necessárias para se trabalhar com o ORACLE, esta obra é voltada ao publico leigo que nunca teve contato com a ferramenta. AULT (1997) apresenta realmente o que se evoluiu de uma versão 7.3 para a nova versão 8. MORAIS (1995) trazem a definição de maneira simples e clara de todos os conceitos referentes às partes internas do banco de dados, como tablespace, arquivos de redo, etc. HURSCH (1991) é um guia de referencias com os comandos necessários para a utilização do banco de dados através de comandos SQL. BODROWSKI (1995) sem duvida é a obra mais importante que existe

publicada no Brasil, ela traz uma visão superficial mais de abrangência total do produto. ABBEY (1997) é uma publicação semelhante à obra do RAMALHO (1997) tendo inclusive a mesma característica e o mesmo publico alvo. Para a realização deste projeto se fez necessário à utilização de todas estas obras referentes ao mesmo assunto, pois em nenhuma delas se encontra uma referencia completa e exemplos suficientes.

Para o desenvolvimento dos módulos necessários neste projeto, foram feitos através das técnicas de programação orientada a objetos, com a literatura do TAYLOR (1995) realizei um dos primeiros contatos com esta técnica, este livro apresenta em forma de cartilha as informações de Classe, Objeto, Herança, Polimorfismo, Banco de dados Relacional e Banco de dados Orientado a Objetos.

A implementação dos códigos fontes forma realizados com a linguagem de programação Smalltalk, linguagem esta que tem toda uma técnica e filosofia diferente das linguagens clássicas como Pascal e C. Para seu aprendizado se fez necessário à leitura de vários livros, entre eles podemos citar: PLETZKE (1997) que é uma das obras mais importantes trazendo vários exemplos de programas de maneira avançada. OBJECTSHARE (1999) manual de referencial mostrando a conexão do banco de dados a linguagem de programação. PARCPLACE (1995), um tutorial de conexão do banco de dados à linguagem de programação. PINSON (1988) traz muitos códigos de programação, mais para uma outra versão de Smalltalk, servindo para este trabalho apenas de referencia de sintaxe. BECK (1997) traz uma visão da aplicabilidade dos conceitos da programação orientada a objetos utilizando Smalltalk, trazendo exemplos completos para a consulta e teste. LEWIS (1995) apresenta como o Smalltalk/Visualworks pensa, sendo esta obro de leitura obrigatória para a programação nesta ferramenta.

A analise orientada a objetos foi realizada através dos conceitos da UML, embora o resultado final apresentado neste projeto tenha sido apenas de um diagrama de classes, os conceitos da UML permite que se adquira uma visão de como realizar a programação. Meu primeiro contato com esta técnica foi em sala de aula na pós-graduação, onde o professor fez a indicação do livro do ERIKSSON (1961). A primeira obra nacional é a do FURLAN (1988) que apresenta de maneira superficial a UML. LARMAN (2000) e BOOCH (2000) trazem os conceitos atualizados e exemplos mais

propícios ao desenvolvimento de software na área comercial, sendo Booch uns dos criados da UML.

WANGENHEIM (1997) é o ponto de inicio deste projeto, este paper traz as informações sobro o projeto de pesquisa Cyclops, sobre o aplicativo DicomEditor e o modelo de classes.

RAHAL JUNIOR (2000) traz o esboço prévio deste projeto de pesquisa apresentando o caminho a ser seguido para a sua conclusão, sendo este caminho hoje diferente ao do modelo proposto inicialmente neste trabalho.

7.0 - MATERIAL E MÉTODOS

7.1 - Banco de Dados Oracle

No ano de 1969, um cientista da IBM chamado E.F.Codd, iniciou uma revolução na área de informações de sistemas com a proposta de uma nova abordagem no gerenciamento de dados. Em sua proposta Codd idealizou o conceito de banco de dados relacional, que pode ser definido como um banco de dados que aparece ao usuário como uma coleção de tabelas relacionadas - e nada além de tabelas.

No ano 1979 a Relational Software Inc (que passou a se chamar Oracle Corparation) criou o primeiro banco de dados do mercado, o ORACLE.
O ORACLE foi projetado para receber grandes quantidades de muitos tipos de dados diferentes. Portanto, seu uso é apropriado para grandes empresas ou instituições, ou outras empresas onde é necessário manter e trabalhar com grandes quantidades de informações sobre conjuntos volumosos de dados.

Características técnicas do Banco de Dados Oracle

- Suporta grandes bancos de dados, potencialmente da ordem de centenas gigabytes de tamanho, e controla o espaço para esses enormes bancos de dados;
- Suporta um grande número de usuários concorrentes, executando uma variedade de operações simultaneamente com os mesmos dados;
- Possui uma alta performance independente da quantidade de informações gerenciadas;
- Trabalha 24 horas por dia, interruptamente, e mesmo as cópias de segurança podem ser feitas com o banco de dados sendo utilizado pelos usuários;

- Controla seletivamente a disponibilidade dos dados para os usuários sem afetar todas as aplicações;
- Possui meios de limitar e monitorar o acesso aos dados, protegendo-os de usuários não autorizados;
- Reforça a integridade das informações através de regras que ditam os padrões aceitos para os dados;
- Permite a utilização da arquitetura cliente-servidor com processamento distribuído, sendo que as responsabilidades de compartilhamento dos dados é da porção back-end, enquanto a porção cliente (front-end) apenas se concentra na interpretação e apresentação dos dados;
- Permite sistemas de banco de dados distribuídos com a combinação dos dados localizados em computadores diferentes, transparente a todos os usuários dos equipamentos ligados de uma rede;
- software é portável entre diferentes sistemas operacionais e é o mesmo em todos os sistemas;
- Permite diferentes tipos de computadores e sistemas operacionais compartilhar as mesmas informações através de uma rede.

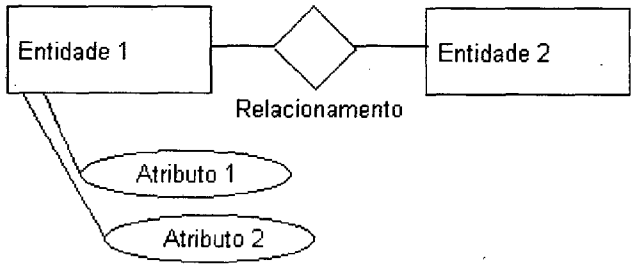
O banco de dados ORACLE foi selecionado para trabalho pela sua portabilidade à praticamente todos os sistemas operacionais e computadores existentes no mercado. Ele traz uma desvantagem com relação a seus concorrentes na parte de custo do produto, sendo na média um pouco mais caro.

Este produto também teve como critério de sua escolha a vantagem de ter acesso nativo a ferramenta utilizada para o desenvolvimento do software deste projeto. Sendo que os demais produtos só poderiam ser acessados mediante a uma conexão em drive ODBC.

7.2 - DER – Diagrama de Entidade-Relacionamento

O Diagrama de Entidade-Relacionamento (Fig.2) é uma modelagem de dados utilizada amplamente nas técnicas de análise estruturada de dados para a representação do modelo de entidades do banco de dados relacional. Uma entidade é uma Tabela, que existe dentro do banco de dados. Através do DER podemos visualizar todos os relacionamentos existentes no banco de dados. No processo de modelagem também é definido dentro de cada entidade os campos, seus tipos e tamanhos, onde um campo é um dado que deve ser armazenado e o tipo referesse a características dos dados, isto é: numérico, dinheiro, caracter, lógico,etc., e o tamanho é espaço necessário para poder se armazenar os dados.

Figura 2 - Diagrama de Entidade-Relacionamento



Fonte: Cougo, Paulo Sérgio. Modelagem conceitual e projeto de banco de dados. São Paulo: Editora Campus, 1997.

Em março de 1976, Peter P. Chen publicou um trabalho intitulado "The Entity-Relationship Model: Toward the unified view of data", no qual definia uma possível abordagem para o processo de modelagem dos dados. Esse trabalho, após sua divulgação e ampla aceitação, passou a ser considerado como um referencial definitivo para o processo de modelagem de dados. A abordagem entidade-relacionamento é composta de uma técnica de diagramação e de um conjunto de conceitos que devem ser entendidos e respeitados.

A técnica de diagramação é bastante simples e serve como meio de representação dos próprios conceitos por ela manipulados. Utiliza, basicamente, um retângulo para representar as entidades, um losango para representar os relacionamentos e balões para indicar e alocar os atributos.

A proposta original de Chen, que se estabeleceu e mantém-se extremamente atualizada até os dias de hoje, baseia-se em um princípio que a torna ao mesmo tempo completa e inquestionável: a formalização do óbvio. Aparentemente, esse pode parecer um meio um tanto simplista de se definir todo o embasamento conceitual no qual se baseia o modelo de entidade-relacionamento. Entretanto, é justamente por apresentar essa simplicidade e objetividade que a abordagem entidade-relacionamento possui tanta flexibilidade e adaptabilidade. Ao contrário do que possa parecer, essa é sua maior qualidade.

A modelagem de dados tem a sua apresentação criada através da ferramenta ERWin, que possibilita uma documentação das entidades e sua exportação para qualquer tipo de banco de dados relacional. Desta maneira a documentação é realizada uma única vez e portável para todos os bancos de dados que o projeto se fizer necessário.

Existem várias notações gráficas para se representar um DER, tais como: IDEF1X, Bachman, Setas, "Pé-de-Galinha", Martin e Peter Chen, este trabalho utiliza a notação "IDEF1X".

7.3 - SQL - Linguagem Estruturada de Pesquisa

SQL "Structured Query language" (Linguagem Estruturada de Pesquisa) é composta por um grupo de facilidades para definição, manipulação e controle de dados em um banco de dados relacional.

Criada na década de 70 o SQL tem como objetivo efetuar, em alguma forma sintática ou concreta, alguma ou todas as características do modelo relacional abstrato.

Definida por D.D.chamberlin e outros (1974) no laboratório de Pesquisa da IBM

em São José, Califórnia, e inicialmente implementada em um protótipo da IBM chamado SEQUEL-SRM (1974-75).

O SQL se tornou padrão oficial em 1986, o American National Standards Institute (ANSI) encarregou seu Comitê de banco de dados (chamado X3H2) de desenvolver uma proposta para uma linguagem relacional padrão (1982).

O SQL é um padrão mundial de linguagem de consulta a banco de dados relacional, neste trabalho foi utilizado para se consultar o banco de dados ORACLE, e o resultado da consulta atribuindo ao modelo de objetos existentes no software deste projeto. Bem como o mecanismo de transformar informações existentes nos objetos do software do projeto em instruções de SQL para inclusão, alteração ou exclusão dos registros existentes no banco de dados ORACLE.

Os métodos existentes no projeto para manutenção ao banco de dados ORACLE pode ser utilizado para demais bancos de dados relacionais existentes no mercado, sem a necessidade de adaptação do código.

7.4 - UML – Linguagem Unificada de Modelagem

BOOCH (2000) diz:

> A UML, Linguagem Unificada de Modelagem, é uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas complexos de software. A UML proporciona uma forma-padrão para a preparação de planos de arquitetura de sistemas, incluindo aspectos conceituais tais como processos de negócios e funções do sistema, além de itens concretos como as classes escritas em determinada linguagem de programação, esquemas de banco de dados e componentes de software reutilizáveis.

Os esforços para a criação da UML se iniciaram oficialmente em outubro de 1994. A UML é a unificação dos métodos já existentes da análise orientada a objetos, métodos como o de Booch e OMT.

Atualmente a UML se encontra na sua versão 1.3.

Para a sua criação, foi estabelecido três objetivos:

1. Fazer a modelagem de sistemas, do conceito ao artefato executável, com a utilização de técnicas orientadas a objetos.
2. Incluir questões de escala, inerentes a sistemas complexos e de tarefas críticas.
3. Criar uma linguagem de modelagem a ser utilizada por seres humanos e por máquinas.

A UML foi selecionada para trabalho em virtude de sua vasta documentação existente na internet e em livros, bem como por utilizar características de técnicas mais antigas e que possui também vasta literatura disponível.

De suas várias etapas de documentação, estaremos apenas utilizando o Diagrama de Classes, que possibilita termos uma visão da modelagem de classes a nível de herança e associação.

7.5 - SMALLTALK

Smalltalk é uma linguagem de programação orientada a objetos. Smalltalk é um modelo puro de orientação a objetos.

Idealizada inicialmente para ser e foi o sistema operacional no Xerox Dynabook mais tarde conhecido como ALTO. Era o grande projeto dos sonhos da Xerox: um computador leve, portátil e fácil de usar. Criada por Alan Kay na década de 70 no PARC - Polo Alto Research Center.

Muito do que Alan idealizou só foi inicialmente conquistado quando esse deixou a Xerox e foi para a Apple Computer.

A versão do Smalltalk utilizada neste projeto é Visualworks 3.0 não comercial. Nesta versão se encontra várias classes idealizadas pelas equipes de desenvolvimento no Brasil e Alemanha.

Mediante a este montante de bibliotecas já codificadas a linguagem Smalltalk foi escolhida para o projeto Cyclops bem como a sua facilidade em programação e alta produtividade.

Esta versão do Smalltalk roda nas principais plataformas de sistemas operacionais existentes no mercado.
Sua portabilidade possibilita através do banco de dados ORACLE a customização dos softwares do projeto Cyclops para qualquer tipo de plataforma de trabalho que os usuários desejarem.

7.6 - DICOM

DICOM é um padrão que permite aos diversos equipamentos de imagens médicas digitais (Tomógrafos Computadorizados, Raio-X, Ecógrafos, etc.) se comunicarem entre si.

O Colégio Americano de Radiologia (ACR) e a Associação Nacional dos Fabricantes de Equipamentos Elétricos (NEMA) formaram um comitê em 1983. O objetivo deste grupo, nomeado como ARC-NEMA Digital Imaging and Communications Standards Comity, era definir e desenvolver uma interface entre equipamentos de tipos diferentes. Além da especificação da conexão física, o desenvolvimento do padrão inclui ainda um dicionário dos elementos necessários na codificação e interpretação das imagens.

Sua primeira versão foi criada na década de 80.

O modelo de DICOM utilizado neste projeto é a versão 3.0, pela sua abrangência e robustez.

7.7 – DicomEditor Hoje

7.7.1 - Visão Geral do Software

DICOM Editor é um software cliente de Teleradiologia em conformidade com o padrão Digital Image Communications in Medicine, versão 3.0, capaz de acessar equipamentos de aquisição de imagens radilógicas como tomógrafos computadorizados e ressonâncias magnéticas e também servidores de bancos de imagens radiológicas que estejam em conformidade com este padrão.

Para isso, **DICOM Editor** modela e implementa o Padrão DICOM 3.0, implementando tanto a) as definições de objetos de imagem radilógica nas diversas modalidades e também definições de objetos do tipo laudo e descrição de exames, como b) também implementa os serviços de busca e armazenamento de imagens definidos por DICOM 3.0, como c) também ainda implementa os protocolos de comunicação definidos por DICOM 3.0.

DICOM Editor permite que se acesse, visualize e edite imagens radiológicas e dados de paciente e laudos a partir de um equipamento radiológico ou de um servidor de banco de imagens que esteja em conformidade com o Padrão DICOM 3.0.

DICOM Editor permite também que se grave imagens radiológicas e dados de paciente e laudos em um equipamento radiológico ou em um servidor de banco de imagens em conformidade com o Padrão DICOM 3.0. Estes dados podem ter sido adquiridos de outra fonte ou serem dados da mesma fonte, os quais foram modificados pelo usuário.

DICOM Editor permite a visualização das imagens radiológicas e outros dados de exames e a aplicação de diversas ferramentas de inspeção destes dados. Estas ferramentas fazem parte do **DICOM Editor**.

Para permitir várias modalidades de visualização e edição de imagens radiológicas e também a visualização de outras informações associadas às imagens, como dados de paciente e laudos, e também o cálculo de informações a prtir de dados da imagem, como histogramas de densidades radiológicas, **DICOM Editor** possui vários módulos adicionais específicos para estas funções, mas que são componentes integrantes do **DICOM Editor**.

7.7.2 - Objetivos do DICOM Editor

O objetivo do software **DICOM Editor** é o de servir como software para estação de trabalho radiológica durante o processo de análise e laudo de imagens radiológicas por parte de um corpo médico. Pode ser usado também para fins educativos como ferramenta de acesso e análise de coleções de imagens radiológicas-exemplo.

7.7.3 - Estrutura do DICOM Editor

7.7.3.1 - O Modelo de Informação do DICOM Editor

Explicitaremos a seguir algumas características do Padrão DICOM 3.0, cujo conhecimento é necessário à compreensão do modelo de dados descrito a seguir.

O Padrão DICOM 3.0 baseia-se em modelos explícitos e detalhados de como "Objetos do Mundo Real", tais como pacientes, imagens, laudos, etc, comumente envolvidos em operações radiológicas, são descritos e como se relacionam entre si. Estes modelos podem ser utilizados por fabricantes de software e claramente descrevem as estruturas básicas do Padrão DICOM e também os itens de dados requeridos pelo padrão e como estes itens de dados interagem. O padrão nada fala sobre formas de implementação do modelo, ficando isto a cargo do fabricante.

O Modelo de Informação DICOM 3.0 está descrito no Capítulo 3 do Manual do Padrão DICOM como um modelo entidade-relacionamento e deve ser seguido por fabricantes de software interessados em desenvolverem software de acordo com o padrão. A modelagem do DICOM Editor seguiu este modelo. O Padrão descreve os objetos do mundo real como entidades neste modelo entidade-relacionamento. As entidades do modelo são objetos do mundo real como um paciente, um estudo (exame), uma imagem, as quais são descritas formalmente como Information Object Definitions - IODs (Definições de Objetos de Informação). A utilização do termo IOD é uma nomenclatura específica do DICOM e será utilizada neste documento a partir daqui.
Há no Padrão DICOM dois tipos de IODS: IODs normalizados e IODs compostos. IODs normalizados representam objetos atômicos do mundo real e modelam apenas uma Information Entity - IE (Entidade de Informação). IEs são todos os objetos do mundo real do modelo que podem ser descritos através de um IOD normalizado.

Objetos do mundo real que são complexos são descritos por IODs compostos. IODs compostos modelam objetos que possuem mais de uma IE, como por exemplo Uma imagem de ressonância magnética ou uma imagem de tomografia computadorizada. IODs compostos forma modelados e implementados no DICOM Editor como Classes de Objetos.

Conjuntos de características de objetos compostos do mundo real que são inter-relacionadas são descritas pelo Padrão DICOM como Information Object Modules - IOMs (Módulos de Objetos de Informação). IOMs são tipos especiais de IODs normalizados que sempre representam conjuntos dados do tipo numérico ou texto e que estão relacionados a um IOD complexo. Na implementação do DICOM Editor os IOMs também foram modelados e implementados como Classes de Objetos.

Modelagem na Implementação dos IODs no DICOM Editor

Na modelagem do Padrão DICOM 3.0 realizadas na implementação do DICOM Editor, foram implementadas as seguintes classes de objetos na linguagem de programação Smalltalk, as quais modelam e implementam as IODs com as quais a tabela as relaciona.

Nome	**IOD do Padrão DICOM Implementado**	**Tipo de IOD**
CyclopsDBObject	Classe de Objeto Abstrata. Representa qualquer objeto DICOM acessado de uma base de dados DICOM. Superclasse de todos as classes aqui descrita.	-
CyclopsPatient	Patient IE	Normalizado
CyclopsPatientModule	Patient IOM	Normalizado
CyclopsPatientStudyModule	Patient Study IOM	Normalizado
CyclopsStudy	Study IE	Normalizado
CyclopsGeneralStudyModule	General Study IOM	Normalizado
CyclopsSeries	Series IE	Normalizado
CyclopsGeneralSeriesModule	General Series IOM	Normalizado
CyclopsContrastBolusModule	Contrast Bolus IOM	Normalizado
CyclopsOverlayPlaneModule	Overlay Plane IOM	Normalizado
CyclopsSOPCommonModule	SOP Common IOM	Normalizado
CyclopsDicomImage	Image IE Utilizada no DICOM Editor como Classe de Objeto Abstrata. Representa qualquer objeto Imagem DICOM e implementa todas as características comuns a imagens no DICOM Editor. Superclasse de todos os tipos de imagens.	-
CyclopsDicomCTImage	CT Image IOD	Composto

CyclopsDicomMRImage	MR Image IOD	Composto
CyclopsDicomSCImage	SC Image IOD	Composto
CyclopsDicomUSImage	US Image IOD	Composto
CyclopsDicomXRAYImage	XRAY Image IOD	Composto
CyclopsGeneralImageModule	General Image IOM	Normalizado
CyclopsImagePixelModule	Image Pixel IOM	Normalizado
CyclopsImagePlaneModule	Image Plane IOM	Normalizado
CyclopsCTImageModule	CT Image IOM	Normalizado
CyclopsMRImageModule	MR Image IOM	Normalizado
CyclopsSCEquipmentModule	SC Equipment IOM	Normalizado
CyclopsUSImageModule	US Image IOM	Normalizado
CyclopsXRAYImageModule	XRAY Image IOM	Normalizado

Classes de Serviços DICOM 3.0 Implementadas no DICOM Editor

7.7.3.2 - Visão Geral das Classes de Serviços Definidas por DICOM 3.0

O padrão DICOM 3.0 define um protocolo contendo uma série de operações genéricas, como armazenar, recuperar, procurar por, mover, etc; as quais podem ser aplicadas sobre um objeto DICOM, como por exemplo uma Imagem de Tomografia Computadorizada. O Padrão DICOM chama estas operações de DICOM Message Service Elements - DIMSE (Elementos de Serviço de Mensagens DICOM). Os serviços providos pelo protocolo DIMSE são subdivididos em serviços DIMSE-C, aplicados sobre IODs compostos, e serviços DIMSE-N, aplicados a IODs normalizados.

A descrição da informação (dados) sobre os quais um serviço vai atuar (um objeto DICOM) e a operação que será aplicada sobre este objeto (serviço DIMSE) são combinadas em uma classe denominada Service Object Pair Class - SOP Class (Classe de Par Objeto-Serviço). As classes SOP representam a unidade funcional elementar definida pelo padrão DICOM. Para uma determinada classe SOP, um dispositivo fisico radiológico ou processo computacional pode representar um de dois papéis: a) assumir o papel de Service Class Provider - SCP (Provedor de Classe de Serviços), caso em que atua como servidor para esta classe de serviços e b) como Service Class User - SCU (Usuário de Classe de Serviços), caso em que atua como cliente dessa classe de serviços. Cada classe de serviço é identificada através de um código identificador único - UID, que precede uma mensagem entre um cliente e um servidor DICOM,

especificando a semântica da operação a ser realizada a seguir e indicando qual a estrutura do resto da mensagem.

7.7.3.3 - Como Classes de Serviços e Papéis DICOM são Implementados no DICOM Editor

No DICOM Editor foi implementado o papel de SCU. O DICOM Editor comunica com servidores de imagens e outros objetos DICOM via rede de computadores TCP/IP e, para realizar suas tarefas de recuperação, manipulação e armazenamento de imagens, dados de paciente e laudos, implementa a classess de serviços padrão DICOM pertencentes ao grupo de SOPs denominado Query/Retrieve Service Classes (Classes de Serviços de Busca e Recuperação). Para isso o DICOM Editor implementa as Classes de Serviços do padrão DICOM denominadas C-STORE, C-FIND e C-GET para todas as IEs do padrão implementadas.

O Serviço C-STORE é implementado de tal forma que é invocado pelo DICOM Editor para requisitar ao servidor de imagens DICOM com o qual está conectado que seja realizado o armazenamento de uma instância de um SOP composto.

O Serviço C-FIND é implementado de tal forma que é invocado pelo DICOM Editor para requisitar ao servidor de imagens DICOM com o qual está conectado para que aceite um conjunto de cadeias de caracteres a ele enviadas e encontre dentre as instâncias SOP por este gerenciadas aquelas que casam com estas cadeias. O serviço C-FIND retorna, para cada acerto, uma lista de atributos e seus valores.

O Serviço C-GET é implementado de tal forma que é invocado pelo DICOM Editor para requisitar ao servidor de imagens DICOM com o qual está conectado para que envie ao DICOM Editor os dados de uma ou mais Instâncias de SOP Compostos. Esta requisição é construída com base em pares atributo-valor anteriormente supridos pelo servidor DICOM em questão ou obtidos do usuário e é enviada de tal forma ao servidor que força o SCP deste a enviar dados associados, incluindo imagens, dados de paciente e laudos.

Na Tabela abaixo estão descritos os pares objeto-serviço do Padrão DICOM implementados no DICOM Editor, juntamente com seu código identificador único de classe associado - Class UID, tal qual definido na documentação DICOM.

Classes.

Nome da Classe SOP	Modalidade	UID de Classe
CT Image Storage	Tomografia Computadorizada	1.2.840.10008.5.1.4.1.1.2
MR Image Storage	Ressonância Magnética	1.2.840.10008.5.1.4.1.1.4
Ultrasound Image Storage	Ultrassonografia	1.2.840.10008.5.1.4.1.1.6.1
X-Ray Image Storage	Radiografia Convencional	1.2.840.10008.5.1.4.1.1.12.1
Patient Root Query/Retrieve Information Model - FIND	Todas as Modalidades - Verifica existência de Paciente	1.2.840.10008.5.1.4.1.2.1.1
Patient Root Query/Retrieve Information Model - GET	Todas as Modalidades - Recupera dados de Paciente	1.2.840.10008.5.1.4.1.2.1.3
Study Root Query/Retrieve Information Model - FIND	Todas as Moadlidades - Verifica existência de Exame/Estudo	1.2.840.10008.5.1.4.1.2.2.1
Study Root Query/Retrieve Information Model - GET	Todas as Modalidades - Recupera dados de Exame/Estudo	1.2.840.10008.5.1.4.1.2.2.3

7.7.3.4 - Interfaces Gráficas de Usuário e Ambientes de Interação

O DICOM Editor possue vários ambientes de interação com o usuário. Cada um deles é um módulo independente e possue uma ou mais interfaces de usuário próprias. Cada módulo é implementado como uma subclasse da Classe Smalltalk **ApplicationModel**.

A seguir serão descritos os módulos do DICOM Editor e sus respectivas interfaces gráficas.

7.7.3.5 - Módulo Básico: DICOM Editor

O ambiente básico de interação com o sistema é denominado DICOM Editor e implementado através de uma Classe Smalltalk denominada DICOMEditor.

O ambiente DICOMEditor é o ambiente inicial de interação com o software e permite acesso a uma base de dados DICOM. A interface de usuário do DICOMEditor está representada na figura 2. As funcionalidades deste ambiente estão descritas adiante.

7.7.3.6 - Acesso Inicial à Base de Imagens DICOM (Servidor DICOM)

O DICOMEditor inicializa-se automaticamente toda vez que uma instância de sua classe é criada e aberta. Esta inicialização inclue a leitura de um arquivo denominado **connection.conf** do mesmo diretório onde se encontra o arquivo imagem Smalltalk contendo o DICOMEditor. No arquivo **connection.conf** estão descritos na sintaxe da linguagem XML os dados dos servidores DICOM aos quais o DICOMEditor terá acesso. Um exemplo do arquivo está listado abaixo:
Para poder ter acesso a um desses servidores, o usuário deve selecionar qual o servidor "atual" e para isso, escolher o ponto de menu "Configure DICOM Connection". Selecionando este ponto de menu, o sistema apresenta uma janela de diálogo onde o usuário pode escolher um dentre os servidores descritos no arquivo de configuração ou pode solicitar a edição deste arquivo para a inclusão de dados de um novo servidor. Caso solicite a edição, um editor de texto é aberto sobre o arquivo de configuração.

Uma vez selecionado o servidor DICOM 3.0 que se deseja acessar, o usuário tem a possibilidade de acessar a lista de todos os pacientes armazenados neste servidor através do ponto de menu " Get Patients". Após selecionado este ponto, o DICOMEditor buscará a lista de todos os dados de paciente armazenados neste servidor, criando uma base de dados interna que conterá instâncias da IE CyclopsPatient e seus módulos, uma para cada paciente carregado da base. Depois da carga dos dados de paciente, o DICOMEditor passa a mostrar, na lista do lado esquerdo da interface, todos os nomes dos pacientes carregados da base. Isto é realizado através da execução das SOPs "Patient Root Query/Retrieve Information Model - FIND" e "Patient Root Query/Retrieve Information Model - GET".

Caso se deseje carregar os dados de paciente de outro servidor, isso pode ser feito a seguir, selecionando-se outro servidor da lista. O DICOMEditor passará a mostrar os pacientes de ambos os servidores em uma única lista.

Acesso aos Dados do Paciente

Para se visualizar os dados de um determinado paciente basta selecionar o paciente da lista e selecionar a página entitulada "Patient" no bloco de notas que se encontra na interface do DICOMEditor. Nesta página estão listados os dados referentes à Patient IE e ao Patient IOM que descrevem o paciente selecionado.

Acesso aos Exames do Paciente

Para acessar os dados dos exames do paciente, chamados na nomenclatura DICOM de "estudos", basta selecionar um paciente. Caso haja exames deste paciente carregados no DICOM Editor, estes serão listados na lista de estudos à direita da lista de pacientes. Caso se deseje carregar dados de exames para o paciente selecionado a partir da base dimagens do servidor DICOM atualmente configurado, pode se fazer isto clicando com o botão do meio do mouse sobre o nome do paciente selecionado e, no menu que surge, selecionar o ponto "Load Studies". Isto é realizado através da execução das SOPs "Study Root Query/Retrieve Information Model - FIND" e "Study Root Query/Retrieve Information Model - GET", implementadas no DICOM Editor.

Para se visualizar os dados de um determinado exame de um paciente basta selecionar o estudo da lista de estudos e selecionar a página entitulada "Study" no bloco de notas que se encontra na interface do DICOMEditor. Nesta página estão listados os dados referentes à Study IE , ao Study IOM e ao Study Patient IOM que descrevem dados do exame selecionado, como tipo, parte do corpo examinada, data, duração, médico requisitante, médico que forneceu laudo, etc.

Acesso às Imagens de um Exame

As imagens no padrão DICOM estão definidas como sendo agrupadas em "Séries". Uma série é uma IE e no DICOM Editor foi modelada como uma Classe Smallltalk, denominada CyclopsSeries. Uma série possui como elementos imagens de uma única modalidade, como Ultrassom ou Tomografia Computadorizada. Uma série pode conter uma ou mais imagens, que sempre são o resultado de uma sequência de aquisições realizada pelo aparelho radiológico que foi utilizado. Séries contendo imagens de raios-X contém tipicamente uma só imagem, séries de tomografias computadorizadas, ultrassons ou ressonâncias magnésticas contém muitas imagens.

Para acessar as imagens de um exame, basta selecionar este exame e, a seguir, selecionar uma das séries que aparecerão na lista de séries. A se selecionar uma série, as imagens desta série aparecerão como ícones, de forma reduzida, na parte de baixo do DICOMEditor. Feito isto, clica-se com o botão central do mouse sobre o nome da série selecionada e então, no ponto "Series Editor" do menu que surge. Com isto, será criada e aberta a interface de uma instância do módulo DICOMSeriesEditor sobre esta série. É possível abrir-se vários DICOMSeriesEditor sobre séries diferentes.

Caso não tenham sido carregadas do servidor as séries associadas a um exame, isto pode ser feito clicando-se com o botão central do mouse sobre um exame selecionado e selecionando-se o ponto de menu "Load Series". Neste caso serão carregados do servidro configurado os dados referentes às séries associadas ao exame selecionados e os dados das imagens de cada uma dessas séries. Para cada série é criado um processo independente, que carrega as séries e suas imagens em processamento de fundo, liberando a interface de usuário para que usuário possa realizar outras operações enquanto espera a carga dos dados.

Para a visualização de dados referentes à série e às imagens existem também paginas para estes fins no bloco de notas da interface DICOMEditor. Basta que uma série esteja selecionada para que estes dados, que vão desde o número e data da série até informações sobre parâmetros técnicos dos equipamentos utilizados para a aquisição da as imagens da série, possam ser visualizados.

7.8 - CTN – Central Test Node

O Central Test Node (CTN) é uma aplicação de demonstração que utiliza um banco de dados relacional (miniSQL) para armazenar informações de configuração dos dados existentes numa imagem, pacientes e outros objetos. A informação de configuração é armazenada em um conjunto de tabelas relacionadas. O CTN irá utilizar muitas tabelas relacionadas para armazenar informações de configurações e dados.

Desenvolvido pela Mallinckrodt Institute of Radiology em Washington University of St. Louis, USA. A última versão que se encontra é 2.8.6, que utiliza banco de dados Msql para armazenar as informações de Séries de Imagens e suporte a pacientes, estudos e séries em operações de armazenamento e recuperação de dados.

Neste trabalho utilizamos o CTN para a captura de imagens que se encontram no servidor, estas imagens uma vez disponibilizadas no DicomEditor podem ser manipuladas pelo banco de dados ORACLE.

Para se ter o domínio e independência tecnológica é que este trabalho tem a sua origem, numa próxima etapa este aplicativo cairá em desuso.

8.0 – MODELAGEM

8.1 – Desenvolvimento do Trabalho

A realização deste trabalho tem como ponto inicial o estudo da linguagem de programação Smalltalk. Através de leituras de manuais e implementações de pequenos software para a aplicabilidade da teoria. Smalltalk é uma linguagem orientada a objetos pura, desta maneira se fez necessário o estudo de técnicas de programação orientada a objetos para um melhor entendimento da linguagem.

A segunda etapa do trabalho foi o estudo do banco de dados ORACLE, instalação, configuração e filosofia da ferramenta. Nesta etapa o banco de dados foi testado com outras linguagens de programação.

Na terceira etapa do projeto se fez necessário o estudo de conexão entre o Smalltalk e o banco de dados ORACLE. Uma vez realizado a conexão iniciou-se um estudo aprofundado de como o Smalltalk trabalha com banco de dados relacional. Vários protótipos de software foram criados para uma solidez sobre a ferramenta. Esta parte do projeto foi caracterizada pela falta de literatura, existindo apenas dois livros de conteúdo superficial do fabricante da linguagem Smalltalk.

Tendo sido realizado estes seguimentos obrigatórios no desenvolvimento do trabalho pode-se realmente iniciar o objetivo do projeto de pesquisa, o estudo do DicomEditor para identificar as informações necessárias para o armazenamento e recuperação no banco de dados ORACLE.

Este estudo começou com a leitura dos códigos fontes existentes no DicomEditor, debugando classe por classe para identificação das informações. Após esta debugação iniciou-se a criação de classes e métodos que recebiam informações para testar se o caminho estava correto, através dos resultados visíveis no DicomEditor. Nesta etapa do trabalho foi possível realmente identificar as classes e variáveis necessárias.

Com as informações coletadas, foi realizado a criação do modelo relacional de entidade e sua aplicabilidade no banco de dados ORACLE. Esse modelo relacional

sofreu várias alterações até a identificação da melhor solução para ganho de performance no processo de carga e gravação das imagens.

Para uma melhor performance no processo de carga das imagens, foi criado um recurso de criação de Ícones, onde toda imagem tem uma cópia reduzida de tamanho 64x64. Desta maneira o DicomEditor carrega primeiro os Ícones e sendo necessário é realizado o processo de carga das imagens originais.

Através dos recursos existentes no módulo de conexão de banco de dados do Smalltalk, se deu origem a criação das classes e métodos necessárias para a leitura e gravação dos dados com o DicomEditor. O Smalltalk possibilitou a criação de maneira interativa de todos os métodos de consulta ao banco de dados, realizando o processo de conversão de informações contidas em objetos para a linguagem de consulta de dados SQL.

Após a estas etapas o protótipo do projeto já se encontrava pronto, tendo a necessidade de ser submetido a testes de validação.

Para estes testes foram realizadas as conexões do DicomEditor ao software CTN e executado os módulos de carga de imagens. Estas imagens uma vez dentro do DicomEditor foram gravadas no bando de dados ORACLE e carregadas para identificação de suas características. Essas identificações foram checadas com as imagens originais. As imagens carregadas do banco de dados ORACLE também foram testadas com outros aplicativos existentes no DicomEditor, tendo seu resultado equivalente as imagens originais.

Uma vez testado o código fonte criado, passou-se por uma reestruturação, sendo adicionados métodos mais abstratos para um melhor reaproveitamento de código fonte, código este que passa a ser utilizado em vários outros projetos do Cyclops.

Os métodos de carga e armazenamento de dados do DicomEditor com o banco de dados ORACLE receberam os recursos de "background", que possibilitou as instruções de código sejam executadas e o software DicomEditor fique liberado para a execução de outras tarefas sem a necessidade de aguardar o término de um processo que já esta em andamento.

Para a conclusão do trabalho foi modificado o layout do DicomEditor para ter disponibilizado em sua interface as opções de se trabalhar com o banco de dados ORACLE o com o software CTN.

Este projeto passou por duas implementações, sendo a primeira utilizando um conjunto de classes denominadas "Leans", onde este conjunto de classes transforma os comandos em SQL em instruções Smalltalk, deixando o código orientado a objetos. E a segunda implementação utilizando as classes denominadas "EXDI", que utilizam instruções SQL para execução dos comandos, sendo estas instruções escritas em formato texto e atribuídas a um comando em forma de parâmetro para a sua exeção.

Foi escolhida a segunda implementação pela a sua compatibilidade com driver ODBC, sendo esta compatibilidade de grande importância para não existir dois códigos com o mesmo objetivo no software DicomEditor.

8.2 – Modelagem Propriamente Dita

O modelo relacional das tabelas de banco de dados foi criado levando em consideração o armazenamento de todos os dados existentes nas classes do DicomEditor, bem como a concatenação dos campos no mínimo possível de arquivos para ser ter um numero mínimo de buscas em Query. Estes procedimentos foram realizados para ser ter ganho de performance.

Foram criados 5 tabelas, sendo elas: Patients, Study, Series, Images_Data, Images_Bits e Icons_bits.

Definição:

Patients (pacientes): é utilizada para armazenar as informações referentes aos pacientes.

Study (estudo): traz as informações dos estudos referentes a cada paciente.

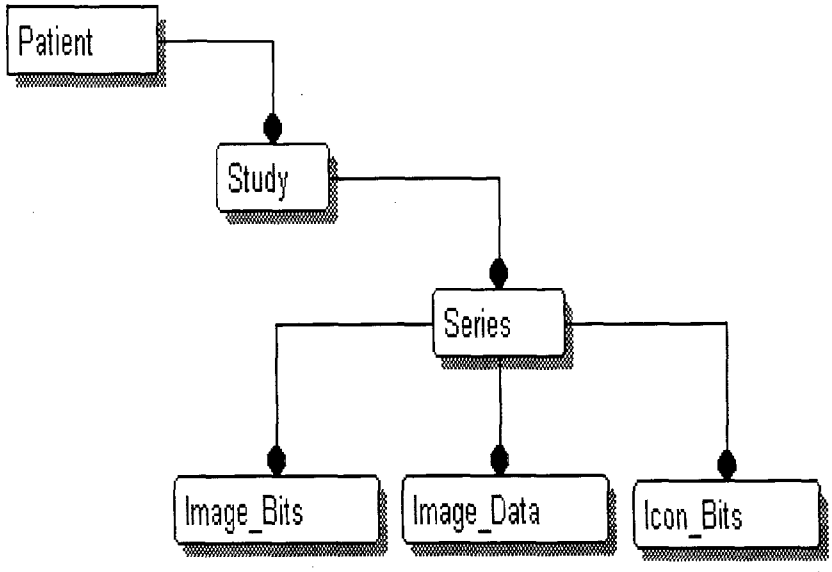
Series: cada estudo pode ter varias series de imagens, sendo esta tabela para armazenar as informações das series existentes no modelo DICOM. Estas informações são genéricas a serie inteira.

Images_Data: tem as informações referentes ao tipo de imagem (CT, MR, NM, SC, US e Raio-X) e também informações referentes ao padrão DICOM que tem repetir a cada imagem.

Images_Bits: Traz o armazenamento dos bits das imagens, em seu tamanho original.

Icons_Bits: Traz o armazenamento dos bits dos ícones, no tamanho 64x64.

Diagrama de Classes do projeto



Criando a Tabela de Patients

create table patients(
id char(10) not null,
name char(128),
birthdate char(32),
sex char(16),
primary key(id))

Criando a Tabela de Study

create table study (
patient char(10) not null,
studyid char(10) not null,
accessionnumber char(128),
referringphysiciansname char(128),

studydate char(32),
studydescription char(128),
studyinstanceuid char(128),
studytime char(32),
patientsage char(16),
patientssize char(16),
patientsweight char(16),
primary key(patient, studyid))

Criando a Tabela de Series

create table series(
study char(10) not null,
seriesnumber char(10) not null,
modality char(16),
seriesinstanceuid char(128),
seriesdate char(32),
seriestime char(32),
protocolname char(128),
bodypartexamine char(128),
smallespixelvalueinseries number(10,0),
largestpixelvalueinseries number(10,0),
operatorname char(128),
fra_frameofreferenceuid char(128),
fra_positionreferenceindicator char(128),
equ_dateoflastcalibration char(32),
equ_deviceserialnumber char(128),
equ_institutionaddress char(128),
equ_institutionaldepname char(128),
equ_institutionname char(128),
equ_manufacture char(128),
equ_manufacturersmodelname char(128),

equ_pixelpaddinggvalue number(10,0),
equ_softwaresions char(128),
equ_spatialresolution char(128),
equ_stationname char(128),
equ_timelastcalibration char(32),
img_width number(10,0),
img_height number(10,0),
img_depth number(10,0),
img_bitsperpixel number(10,0),
ico_width number(10,0),
ico_height number(10,0),
ico_depth number(10,0),
ico_bitsperpixel number(10,0),
primary key(seriesnumber, seriesnumber))

Criando a Tabela de Image_Date

create table images_data (

Study char(10) not null,
SeriesNumber char(10) not null,
ImageNumber char(10) not null,

IPL_ImageOrientationPatient char(128),
IPL_ImagePositionPatient char(128),
IPL_PixelSpacing char(128),
IPL_SliceLocation char(128),
IPL_SliceThickness char(128),

IPI_BitsAllocated char(128),
IPI_BitsStoped char(128),
IPI_Columns char(128),

IPI_HighBit char(128),
IPI_LargestImagePixelValue char(128),
IPI_PhotometricInterpretation char(128),
IPI_PixelRepresentation char(128),
IPI_Rows char(128),
IPI_SamplesPerPixel char(128),
IPI_SmallestImagePixelValue char(128),

CON_BolusAgent char(128),
CON_BolusRoute char(128),
CON_BolusStartTime char(128),
CON_BolusStopTime char(128),
CON_BolusTotalDose char(128),
CON_BolusVolume char(128),

OVE_BitPosition char(128),
OVE_BitsAllocated char(128),
OVE_Columns char(128),
OVE_Data char(128),
OVE_Origin char(128),
OVE_Rows char(128),
OVE_Type char(128),

VOI_LutData char(128),
VOI_LutDescriptor char(128),
VOI_LutExplanation char(128),
VOI_Sequence char(128),
VOI_WindowCenter char(128),
VOI_WindowCenterWidthExpl char(128),
VOI_WindowWidth char(128),

SOP_InstanceCreationDate char(128),
SOP_InstanceCreationTime char(128),
SOP_InstanceCreationUID char(128),
SOP_ClassUID char(128),
SOP_InstanceUID char(128),

GEN_PatientOrientation char(128),
GEN_ImageDate char(128),
GEN_ImageTime char(128),
GEN_ImageType char(128),
GEN_AcquisitionNumber char(128),
GEN_AcquisitionTime char(128),
GEN_AcquisitionDate char(128),
GEN_ReferencedImageSequence char(128),
GEN_SourceImageSequence char(128),
GEN_ImageComments char(128),

CT_Kup char(128),
CT_RescaleIntercept char(128),
CT_RescaleSlope char(128),

MR_EchoTime char(128),
MR_TrainLength char(128),
MR_InversionTime char(128),
MR_AcquisitionType char(128),
MR_RepetitionTime char(128),
MR_ScanningSequence char(128),
MR_ScanOptions char(128),
MR_SequenceVariant char(128),
MR_TriggerTime char(128),

NM_ActualFrameDuration char(128),
NM_CodeValue char(128),
NM_CodingSchemeDesignator char(128),
NM_LossyImageCompression char(128),
NM_ReferencedSopClassUID char(128),
NM_ReferencedSopInstanceUID char(128),
NM_EnergyWindowNumber char(128),
NM_AngularViewVector char(128),
NM_DetectorVector char(128),
NM_EnergyWindowVector char(128),
NM_FrameIncrementPointer char(128),
NM_NumberOfDetectors char(128),
NM_NumberOfEnergyWindows char(128),
NM_NumberOfPahses char(128),
NM_NumberOfRotations char(128),
NM_NumberOfRRIntervals char(128),
NM_NumberOdSlices char(128),
NM_NumberOfTimeSlots char(128),
NM_PhaseVector char(128),
NM_RotationVector char(128),
NM_RRIntervalVector char(128),
NM_SliceVector char(128),
NM_TimeSliceVector char(128),
NM_TimeSlotVector char(128),

SC_ConversionType char(128),

US_CodeValue char(128),
US_CodingShemeDesignator char(128),
US_FrameIncrementPointer char(128),
US_LossyImageCompression char(128),
US_PlanarConfiguration char(128),

US_ReferencedSopClassUID char(128),
US_ReferencedSopInstanceUID char(128),

XRAY_PixelIntensityRelShip char(128),

primary key(study, seriesnumber, imagenumber));

Criando a Tabela de Images_Bits

create table images_bits(
study char(10) not null,
seriesnumber char(10) not null,
imgid char(10) not null,
bits long Raw,
primary key(study, seriesnumber, imgid))

Criando a Tabela de Icons_Bits

create table icons_bits(
study char(10) not null,
seriesnumber char(10) not null,
imgid char(10) not null,
bits long Raw,
primary key(study, seriesnumber, imgid))

9.0 - RESULTADOS e DISCUSSÃO

9.1 – Discussão

Este projeto traz vários aspectos de ganhos na facilidade de operação e instalação do DicomEditor em relação ao seu modelo anterior que só trabalhava com o aplicativo CTN.

Podemos ressaltar com o maior ganho em facilidade de instalação o processo de implantação da base de dados, onde o banco de dados ORACLE pode ser instalado em plataformas de sistemas operacionais amigáveis como Windows 9x/NT.

Por ser uma base de dados ORACLE a realização de Backup's (Cópia de segurança dos dados existentes) pode ser feita com aplicativos que acompanham o próprio banco de dados de maneira interativa, sendo possível a realização por técnicos sem grandes conhecimentos da área de informática e o Restore (Restauração de uma cópia de segurança) segue as mesmas facilidades. O procedimento de cópia de segurança pode ser feito ate mesmo com o banco de dados sendo utilizado por vários usuários ao mesmo tempo.

O aplicativo DicomEditor traz facilidade no manuseio por meios de menus, desta maneira podemos escolher qual o tipo de servidor desejamos utilizar: CTN ou ORACLE.

Por recurso de Ícones adicionado ao aplicativo, hoje o processo de visualizar uma séries de imagens pode ser feita de maneira mais rápida que a sua versão anterior, pois as imagens são mostradas no tamanho 64x64 e o usuário do aplicativo pode selecionar uma ou uma série inteira para carga da imagem no seu tamanho original. O usuário também pode selecionar uma ou uma serie de imagens para seu armazenamento.

Com a implementação deste projeto, hoje existe um conjunto de classes que podem ser reutilizadas em outros projetos do Cyclops. Estas classes e métodos foram criados já com esta intenção de reutilização pelos demais aplicativos.

Na parte de banco de dados, as estruturas existentes das classes e métodos de conexão são projetadas para trabalhar com qualquer outro banco de dados, sendo necessário apenas a criação de uma nova classe de acesso para cada tipo de banco de

dados. Todos os métodos de inserção, alteração, exclusão e consulta de dados são os mesmos, independentes do banco de dados utilizados (ORACLE ou ODBC). Isto foi possível graças ao modelo de classes e métodos do Smalltalk.

Este projeto foi um pequeno passo para a independência tecnológica que o Cyclops necessita. Hoje existe apenas a necessidade da carga de imagens mediante ao software CTN, mais este processo já esta sendo implementado e a dependência tecnológica que existia no armazenamento das imagens solucionado.

9.2 – Testes

Foram realizados testes de performance com o software CTN e o banco de dados ORACLE. Estes testes embora ainda sem muito aprofundamento já possibilitam uma visualização prévia dos caminhos a serem tomados em relação a configuração da ferramenta para se atingir a meta de tempo de resposta ideal.

Equipamentos do teste:

Servidor DICOM: Pentium K6/2-450 com 128 MB Ram e disco de 2 GB, se encontra o software CTN e o banco de imagens DICOM com sistema operacional Linux.

Servidor ORACLE: Pentium K6/2-500 com 128 MB Ram e disco de 3.6 GB, rodando Smalltalk e sistema operacional Windows/NT.

Rede TCP/IP.

Resultados:

Qtd. De Imagens	Carga de Imagens do CTN	Carga de Images do Oracle	Carga de Ícones do Oracle
105	4:25	2:37	0:05
19	0:25	0:23	0:01

1.0 - CONCLUSÕES

Este trabalho tem seu enfoque no armazenamento de dados no banco de dados ORACLE, e as ferramentas utilizadas na sua confecção irão dar uma pequena, mais importante contribuição aos demais projetos existentes no Cyclops.

Projetos como o de armazenamento de dados em banco de dados utilizando ODBC. Este projeto irá possibilitar uma redução de custo na implantação dos softwares de pesquisa Cyclops, pois permitira a utilização de banco de dados com custos mais acessíveis.

O Projeto do Portal, onde está sendo criado um portal médico, com acesso as informações de todas as clínicas conveniadas, o portal irá prover os dados dos pacientes e suas séries de imagens médicas.

A criação do Servidor DICOM, que possibilitará a independência total de tecnologia dos softwares Cyclops, não tendo mais a necessidade de utilização do programa CTN.

ANEXO 1

Manual do Usuário

Neste anexo é mostrado como utilizar o sistema DicomEditor: como executar, realizar a conexão com o banco de dados ORACLE, salvar um registro e recuperar os dados do banco de dados.

Como utilizar o Sistema DicomEditor

O sistema DicomEditor é um sistema que visa recuperar e armazenar dados de pacientes com as suas respectivas imagens capturadas pelo padrão DICOM. Sua janela pode ser observada na *Figura 1.*

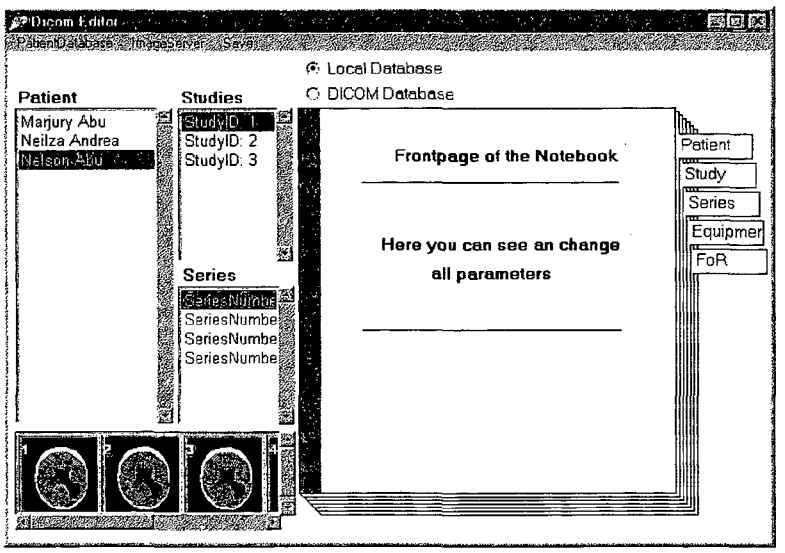


Figura 1: Janela Principal do Sistema DicomEditor

Inicializando o Software

Para a execução do software é necessário executar o comando existente na *Figura 2.*

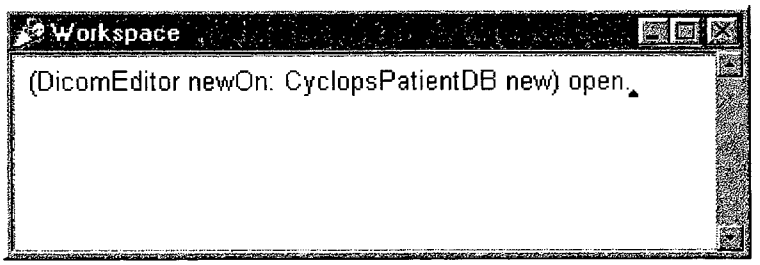


Figura 2: Comando de execução do Dicomditor

Menus de conexão do DicomEditor a base de dados

O DicomEditor pode se conectar a base de dados através do software CTN ou através do banco de dados ORACLE. Para isso o usuário deve selecionar a opção de menu desejada.

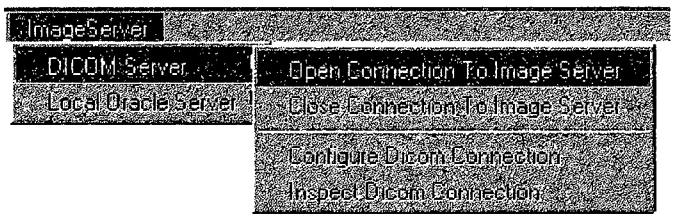


Figura 3: Menu de conexão utilizando o Software CTN

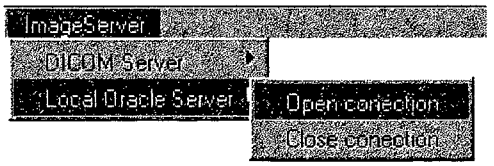


Figura 4: Menu de conexão utilizando o banco de dados ORACLE

Login no banco de dados ORACLE

Sendo executado a opção de conexão ao banco de dados ORACLE, é apresentada uma tela solicitando o nome do usuário e sua senha, *Figura 5.*

Após o preenchimento das informações e o pressionamento da tecla de confirmação (OK) o software DicomEditor já se encontra logado ao banco de dados.

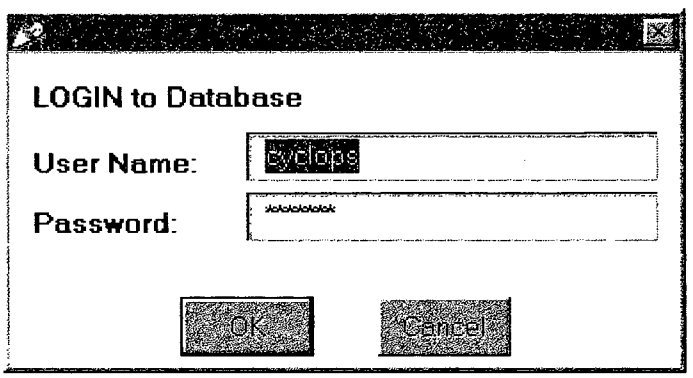


Figura 5: Tela de Login do DicomEditor ao banco de dados ORACLE

Seleção do modelo de conexão para operações de manutenção de dados

Os comandos de recuperação e gravação de registros são customizados conforme a opção de conexão ativa que se encontra selecionada, Local Database ou DICOM Database (*Figura 6*).

Local Database, permitirá que os comandos sejam executados no banco de dados ORACLE e DICOM Database realizará a execução dos comandos no software CTN.

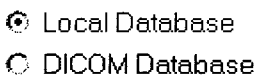


Figura 6: Caixa de opção da conexão ativa do DicomEditor

Carregando dados conforme o modelo de conexão

Para a recuperação de todos os pacientes existente, basta selecionar no menu a opção *PatientDatabase* e selecionar o item *Load All Patients*, conforme a *Figura 7.*

Para se recuperar todos os estudos de um paciente, basta posicionar o mouse em cima da caixa rotulada "Patients" selecionar o paciente desejado e pressionar o botão direito do mouse, sendo apresentado desta maneira um menu conforme a *Figura 8.*

A recuperação de uma série de imagens existentes em cada estudo do paciente é realizada através da seleção do estudo desejado e pressionado o botão direito do mouse em cima da caixa rotulada "Study", sendo apresentado desta maneira o menu conforme a *Figura 9.*

Caso a opção ativa na *Figura a6* seja *Local Database* o processo de recuperação de uma série de imagens trás na sua execução as imagens no formato de Ícones. Desta maneira para se recuperar uma imagem no seu tamanho original deve-se posicionar o mouse sobre a caixa rotulada "Series" e pressionar o botão direito do mouse, aparecendo um menu conforme a *Figura 10.*

Estas operações são executadas conforme a seleção existente na *Figura 6.* Se não existir uma conexão ativa, ao se executar estas operações uma mensagem de erro será apresentada.

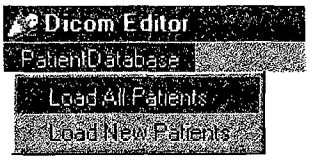


Figura 7: Menu de carga de pacientes

Figura 8: Menu de carga de estudos de um paciente

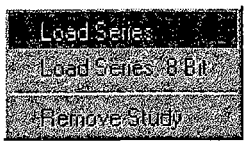


Figura 9: Menu de carga de series de um estudo

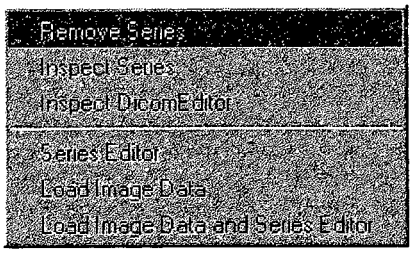


Figura 10: Menu de carga de imagens no tamanho original

Salvando um registro

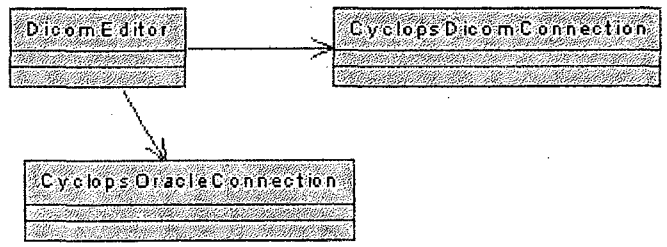
Para se salvar um registro no banco de dados, basta selecionar o paciente desejado, seu estudo e a série do estudo. Pressionando a opção de menu *Save* aparecerá as opções, bastando apenas selecionar a opção: *Save Changes*, conforme a *Figura 11.*

Caso não exista nenhuma conexão ativa, será apresentada uma mensagem de erro, o a gravação é efetuado conforme a seleção existente na *Figura 6.*

Figura 11: Menu de gravação do registro desejado

ANEXO 2

No que segue é apresentado a modelagem do sistema de conexão do DicomEditor ao Banco de Dados ORACLE em UML ("Unified Modeling Language"), bem como os métodos criados e alterados para a customização do software.



Métodos e variáveis criados na classe DicomEditor

Variáveis:

Nome da variável	Descrição	Tipo
sessionOracle	Armazena a sessão de conexão com o banco de dados.	CyclopsOracleConnection
listMenssages	Coleção de mensagens de erro.	Array
edEnvironment	Caixa de edição do ambiente de conexão.	InputFieldSpec
edPwd	Caixa de edição da senha de login do banco de dados.	InputFieldSpec
edUsername	Caixa de edição do usuário de login do banco de dados.	InputFieldSpec
typeDatabase	Checa se é banco de dados Oracle ou ODBC.	RadioButtonSpec
typeConnection	Checa se é acesso ao banco de	RadioButtonSpec

	dados ou ao software CTN.	
edEnvironmentView	Mostra o ambiente de conexão ativo na tela do DicomEditor.	InputFieldSpec
edNameView	Mostra o nome do usuário logado na tela do DicomEditor.	InputFieldSpec

Métodos criados:

Protocolo: menu-actions

closeOracleCyclops

Fecha a conexão do DicomEditor com o CyclopsOracleConnection

connectOracleCyclops

Abre a conexão do DicomEditor com o CyclopsOracleConnection

Protocolo: relational database

dbOracleConstrast: aImageData

Método de Replace dos dados

dbOracleCT: aImageData

Método de Replace dos dados

dbOracleGeneralImage: aImageData

Método de Replace dos dados

dbOracleImagePixel: aImageData

Método de Replace dos dados

dbOracleImagePlane: aImageData

Método de Replace dos dados

dbOracleLoadAllPatients

Carrega todos os pacientes.

dbOracleLoadSeriesIconsFor: aStudy

Carrega todos os Ícones de uma Série.

dbOracleLoadSeriesImagesFor: aStudy series: aSeries

Carrega todas as Imagens de uma Série.

dbOracleLoadStudiesFor: patient

Carrega todos os estudo de um paciente.

dbOracleMR: aImageData

Método de Replace dos dados

dbOracleNMImage: aImageData

Método de Replace dos dados

dbOracleOverlay: aImageData

Método de Replace dos dados

dbOracleProcessorLoadSeriesIconsFor: aStudy

Executa o método "dbOracleLoadSeriesIconsFor: aStudy" com o controle de trabalho em background.

dbOracleProcessorLoadSeriesImagesFor: aStudy series: aSeries

Executa o método "dbOracleLoadSeriesImagesFor: aStudy" com o controle de trabalho em background.

dbOracleProcessorSaveChanges: aPatient study: aStudy series: aSeries

Executa o método "dbOracleSaveChanges: aPatient study: aStudy series: aSeries" com o controle de trabalho em background.

dbOracleSaveChanges: aPatient study: aStudy series: aSeries

Salva o registro que esta sendo manipulado (Paciente, estudo, série e imagens da série).

dbOracleSC: aImageData

Método de Replace dos dados

dbOracleSopCommon: aImageData

Método de Replace dos dados

dbOracleUS: aImageData

Método de Replace dos dados

dbOracleVoilut: aImageData

Método de Replace dos dados

dbOracleXRAY: aImageData

Método de Replace dos dados

Métodos alterados:

Método	Protocolo	Alteração
menuBar	menus	Adição dos itens referentes a conexão do banco de dados.
loadAllPatients	menu-actions	Adaptação do método para uso do CTN ou do banco de dados.
loadImageData	menu-actions	Executa o método de carga de uma imagem.
loadImageDataAndSeriesEditor	menu-actions	Executa o método de carga de uma imagem e abre o SeriesEditor.
openSeriesEditor	menu-actions	Controle para se abrir o SeriesEditor so quando as imagens tiverem sido carregadas.
loadSeriesFor:	actions	Adaptação do método para uso do CTN ou do banco de dados.
loadStudiesFor:	actions	Adaptação do método para uso do CTN ou do banco de dados.
SaveChanges	actions	Adaptação do método para uso do CTN ou do banco de dados.
initialize	initialize-release	Configuração do ambiente de trabalho.
changeTypeConnection	changing	Remodela o menu de Series conforme a conexão ativa (CTN ou banco de dados).

Métodos criados na classe CyclopsOracleConnection

Variáveis:

Nome da variável	Descrição	Tipo
usr	Variável com o nome do usuário de login do banco de dados.	String
pwd	Variável com a senha do usuário de login do banco de dados.	String
env	Variável com o ambiente de conexão do banco de dados.	String
App	Conexão ao banco de dados.	OracleConnection
Connection	Verifica se já existe uma conexão ativa.	Booleano
Session	Sessão de conexão do banco de dados.	OracleSession

Protocolo: Query

methodIconsBits: aStudy seriesNumber: aSeriesNumber

Este método executa um select que recupera todos os Ícones de uma série.

methodIconsBits: aStudy seriesNumber: aSeriesNumber imgID: aImgID

Este método executa um select que recupera uma imagem (Ícone) de uma série.

methodImagesBits: aStudy seriesNumber: aSeriesNumber

Este método executa um select que recupera todas as imagens de uma série.

methodImagesBits: aStudy seriesNumber: aSeriesNumber imgID: aImgID

Este método executa um select que recupera uma imagem de uma série.

methodImagesData: aStudy seriesNumber: aSeriesNumber

imageNumber: aImageNumber

Este método executa um select que recupera os dados de uma imagem.

methodPatients

Este método executa um select que recupera todos os pacientes.

methodPatients: aValue

Este método executa um select que recupera um paciente.

methodSeries: aStd

Este método executa um select que recupera todas as séries de um estudo.

methodSeries: aStudy seriesNumber: aSeriesNumber

Este método executa um select que recupera uma série de um estudo.

methodStudy: aValue

Este método executa um select que recupera todos os estudos de um paciente.

methodStudyID: aStudyID

Este método executa um select que recupera um estudo de um paciente.

Protocolo: Connecting

closeOracleCyclops

Fecha a conexão

connectOracleCyclops

Cria a conexão

Protocolo: Actions

Estes métodos **inspect** são uma camada intermediaria para a realização de uma query no banco de dados, estes métodos chamam os métodos do protocolo query.

inspectAllPatients
inspectIconsBits: aStudy seriesNumber: aSeriesNumber
inspectIconsBits: aStudy seriesNumber: aSeriesNumber imgID: aImgID
inspectImageData: aStudy seriesNumber: aSeriesNumber
imageNumber: aImageNumber
inspectImagesBits: aStudy seriesNumber: aSeriesNumber
inspectImagesBits: aStudy seriesNumber: aSeriesNumber imgID: aImgID
inspectPatients: aValue
inspectSeries: aStudy
inspectSeries: aStudy seriesNumber: aSeriesNumber
inspectStudy: aValue
inspectStudyID: aValue

Os métodos de **update** executam os métodos de inspect buscando no banco de dados a informação desejada, se a informação não é encontrada ele adiciona, caso contrario ele altera.

updateIconsBits: aSeries
updateImagesBits: aSeries
updateImagesData: aSeries
updatePatients: aPatients
updateSeries: aSeries
updateStudy: aStud

O método de **field** serve apenas para realizar a atualização das variáveis de memória que irão ser gravadas no arquivo de banco de dados.

fieldsImagesData: aClass series: aSeries imageNumber: aImageNumber

ANEXO 3

Implementação de um Cliente Network DICOM em Smalltalk para Conexão ao Banco de Dados Oracle

Nelson Abu Samra Rahal Junior,CESUMAR Centro de Ensino Superior de Maringá, Brasil, nelson_abu@bol.com.br

Abstract

This work presents a conection with Oracle Database to the Client DICOM Network, showing the operation procedures of the conection classes together with a relational model created in the Oracle Database. In this way, the Client DICOM Network which uses the DICOM protocol has the freedom to opt for the CTN data server or Oracle Database.

Key words

DICOM, Cyclops, Smalltalk, Medical Information Technology and Oracle

Resumo

Este trabalho apresenta a conexão do Banco de Dados Oracle ao Cliente Network DICOM. Mostrando os procedimentos de operação das classes de conexão, juntamente com o modelo relacional criado no Banco de Dados Oracle. Desta maneira, o Cliente Network DICOM que utiliza o protocolo DICOM tem a liberdade de optar pelo uso do servidor de dados CTN ou uso do Banco de Dados Oracle.

Palavras Chaves

DICOM, Cyclops, Smalltalk, Informática Médica e Oracle.

1 Objetivo

Muitos dos dispositivos médicos utilizados em modernas clínicas, utilizam provedor de dados correspondente para o padrão DICOM [2]. Cyclops [1], um sistema desenvolvido para análise de imagens, baseadas em conhecimentos pela universidade de Kaiserslautern, é adequado a análise das imagens, mas necessitamos de uma interface DICOM para conectar ao DICOM databases de maneira fácil, para transferência de dados pelo sistema. Todos os softwares não comerciais avaliados para nossa demanda não foram satisfatórios em virtude disto foi implementado um Cliente Network DICOM, podemos conectar para a imagem do servidor CTN [3], baseado em SQL database. Para isto foi agregado valores necessários, como a implementação em orientação a objetos, facilitando a expansão e a flexibilidade para futuras novas versões do padrão DICOM. Nessa expansão, a agregação da conectividade com os recursos da implementação orientada a objetos ao banco de dados Oracle, permite ao usuário escolher qual o provedor de dados que melhor lhe agrada, Oracle ou servidor de dados CTN.

2 Métodos

2.1 Informações do Modelo

Cliente Network DICOM é a reprodução do padrão DICOM com uma estrutura orientada a objetos. O Cliente Network DICOM permite armazenar e recuperar dados através do servidor de dados CTN e o Banco de Dados Oracle. O servidor de dados CTN acessa os Banco de Dados SyBase, miniSql e Sql Server da Microsoft onde o modelo relacional das entidades já vem

definidas. O modelo relacional criado no Banco de Dados Oracle (Figura 01) teve que ser idealizado conforme o modelo de classes existentes no Cliente Network DICOM.

O Cliente Network DICOM, traz uma estrutura de classes detalhadas do mundo real, para trabalhar com as informações de Pacientes, Estudos de Casos, Séries de Imagens e Imagens (Figura 03). As classes de imagens estão ajustadas para receber informações do tipo CT (Tomografia Computadorizada), MR (Resonância Magnética), US (Imagens de Ultrasom), X-Ray (Imagens de Raio X) e outras, sendo o modelo, de fácil adaptação para qualquer outro tipo de dispositivo de imagens.

O conjunto de classes criados para a conexão com o Banco de Dados Oracle (Figura 2) teve de ser adaptado ao modelo, de maneira que utiliza-se das outras classes já existentes, sendo implementado apenas os mecanismos de solicitação de dados do Banco de Dados e armazenamento, onde os dados solicitados através de comandos SQL, são convertidos para a estrutura do Cliente Network DICOM e encaminhados para as suas demais classes. Uma vez necessário salvar estes dados, esses dados são convertidos do modelo de classes existente para linguagem SQL, realizando as operações de manutenção do Banco de Dados.

Figura 1:

Modelo relacional das tabelas de banco de dados implementadas no Oracle.

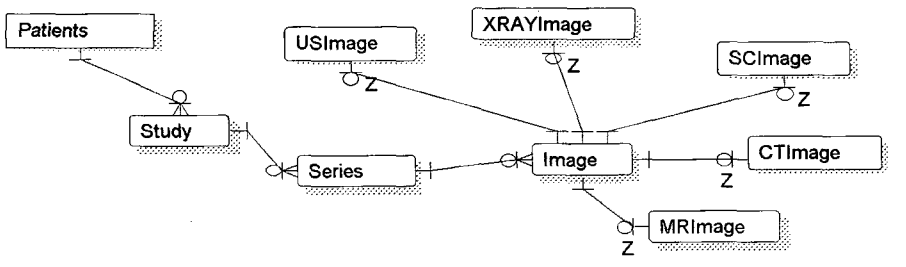


Figura 2:

Modelo relacional de classes, no padrão UML: este conjunto providencia o acesso ao banco de dados Oracle, juntamente com a conexão da Figura 03 para a manutenção da compatibilidade.

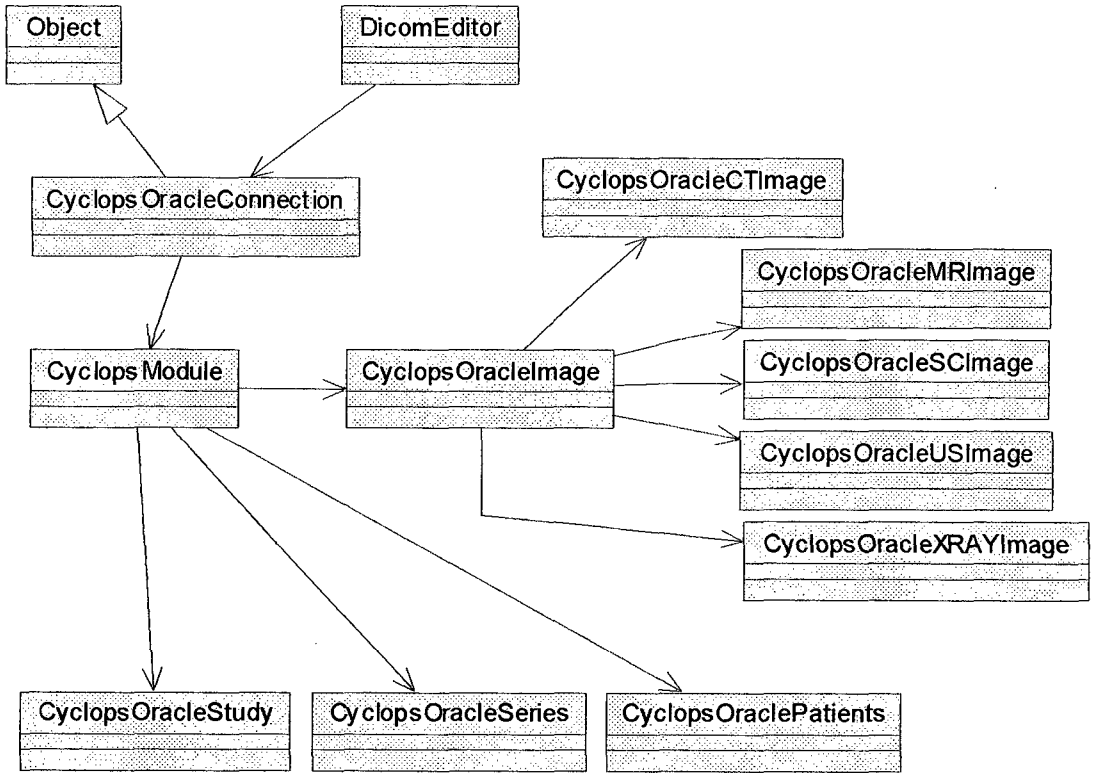
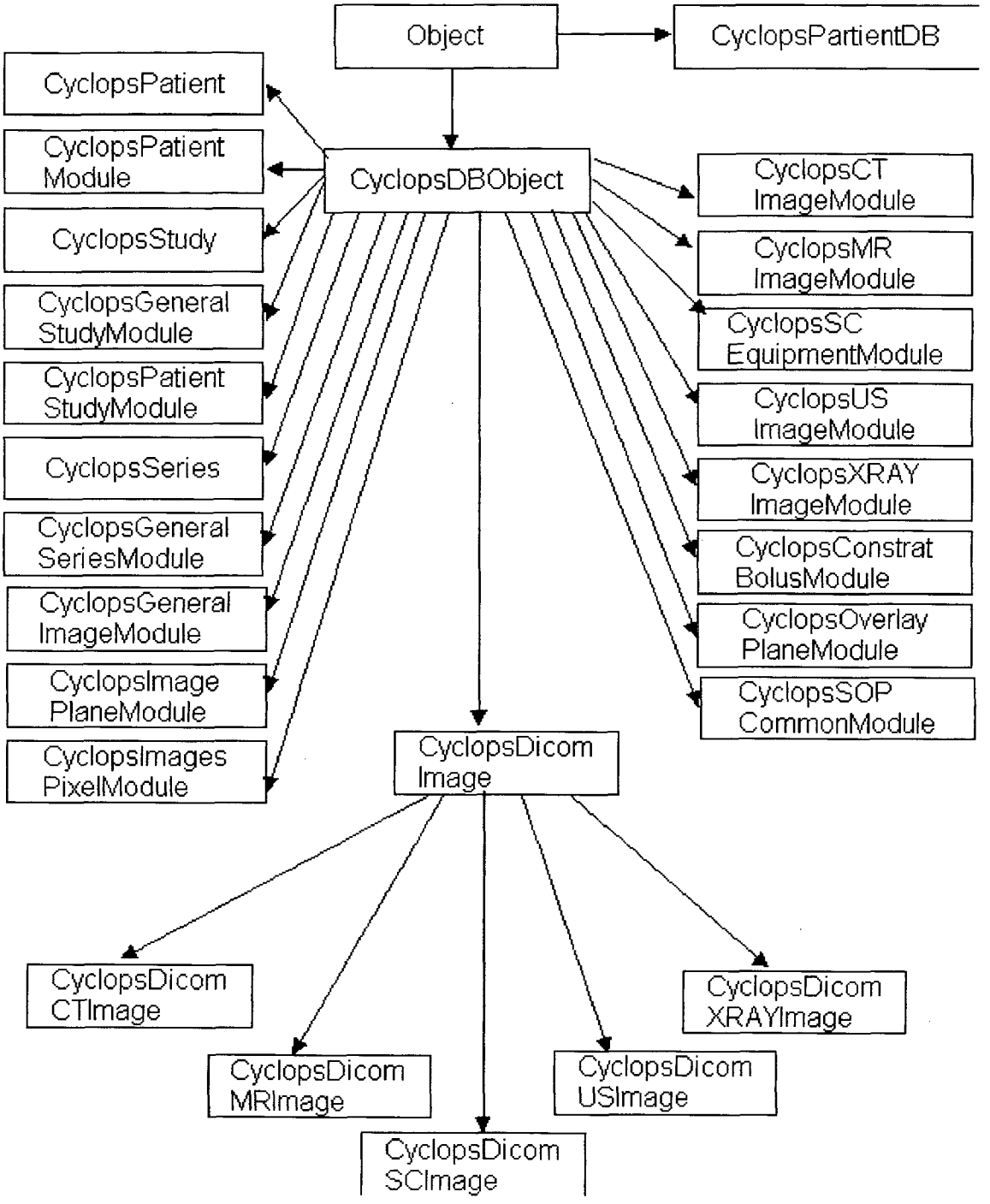


Figura 3:

Todas as classes existentes neste modelo, estão hierarquicamente abaixo da classe Object. Estas classes dão origem ao Cliente Network DICOM.



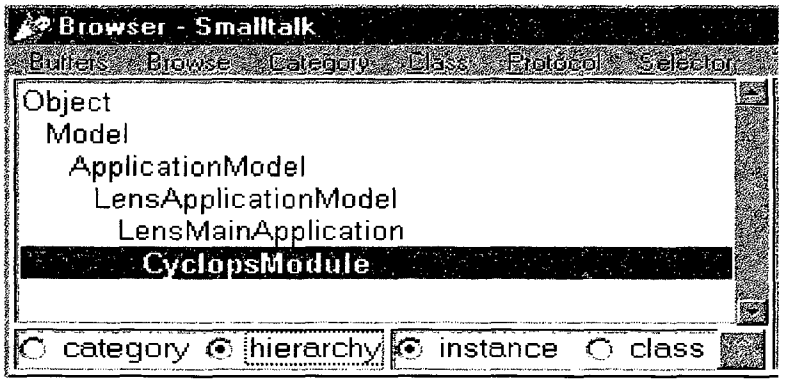
2.2 Serviços de Classes

O Cliente Network DICOM, define e seta operações genéricas com os dados (armazenamento, leitura, busca, movimentação, etc.). Estas operações são encaminhadas ao modelo de armazenamento de dados configurado para uso (servidor de dados CTN ou Oracle). Para trabalhar com estas operações com o Banco de Dados Oracle foi criado um conjunto de classes denominadas "Cyclops Oracle". Onde existe uma classe Cyclops Oracle Connection que tem o objetivo de fazer o interfaciamento entre a classe de acesso ao Banco de Dados (Cyclops Module) e as operações solicitadas pelo aplicativo (Cyclops Dicom Editor).

Cyclops Module (Figura 4) realiza a conexão física com o Banco de Dados Oracle, Associação com as classes de tabelas do Banco de Dados (Cyclops Oracle Patients, Cyclops Oracle Séries, Cyclops Oracle Study, etc.) e os métodos de armazenamento e recuperação de dados em SQL.

Desta maneira através do aplicativo Cyclops Dicom Editor é disparado uma operação de armazenamento ou recuperação de dados, que é processada pela classe Cyclops Oracle Connection através da associação com a classe Cyclops Module que possui os métodos de conexão ao Banco de Dados e os métodos com as operação de manutenção de dados. A classe Cyclops Module faz a associação com as classes de entidades de Banco de Dados e retorna o resultado da operação solicitada. O resultado é encaminhado para o Cyclops Oracle Connection que transmite às classes do Cyclops Dicom Editor, ajustadas a receber os dados, independentemente do mecanismo de armazenamento e recuperação de dados configurado pelo usuário (Oracle ou servidor de dados CTN).

Figura 4:



2.3 Dicom Editor

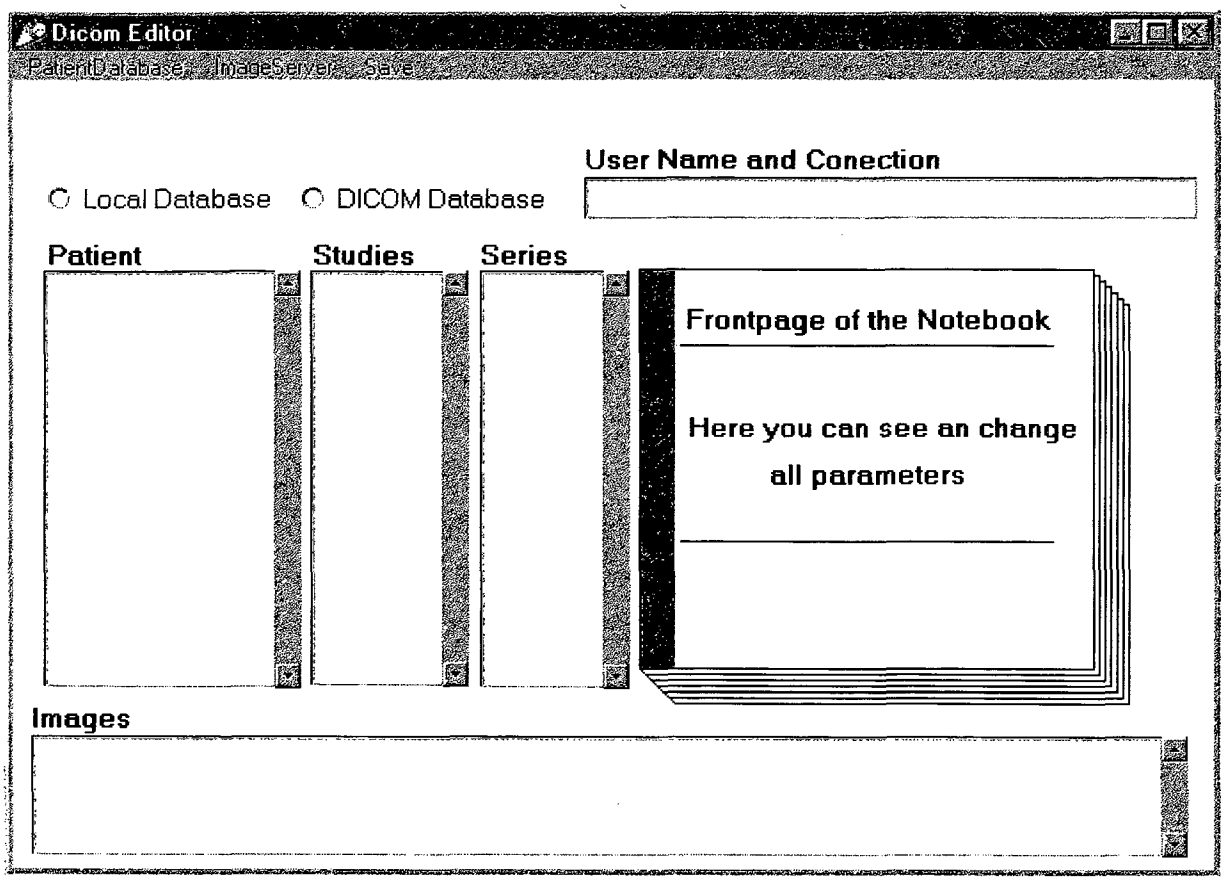
O Dicom Editor (Figura 5) dá acesso a todos os pacientes armazenados no Banco de Dados. Após selecionar um paciente o Editor enumera todos os estudos. Selecionando um estudo especifico, é mostrado todos as séries de imagens existentes. O Notebook a esquerda da tela, mostra todos os os atributos dos pacientes, estudos e séries, sendo possível alterar e armazenar estas modificações no Banco de Dados.

Esta ferramenta foi adaptada ao novo módulo Cyclops Oracle, tendo adcionado em sua interface as informações do usuário que esta logado e o tipo de conexão que esta configurada (Oracle ou servidor de dados CTN). Seus menus foram adequados ao novo modulo, customizados a um agrupamento de operações, desta maneira o Dicom Editor ficou mais amigavél em sua customização.

Os métodos de armazenamento e recuperação de dados, foram todos configurados ao novo módulo, ajustados a realizar a operação solicitada, conforme o mecanismo de conexão ativo no Dicom Editor.

Figura 5:

Tela do aplicativo DICOM Editor com os recursos de acesso ao banco de dados Oracle.



3 Conclusões e Futuros Trabalhos

O Cliente Network DICOM tinha a necessidade de manipular seus dados através do servidor de dados CTN, que é um aplicativo externo da ferramenta (Cliente Network DICOM), sendo uma caixa preta, onde não existe o acesso para alterar os procedimentos de manipulação dos dados conforme as nossas necessidades. O servidor de dados CTN acessa a alguns Bancos de Dados, mas estes Bancos de Dados não tem a portabilidade para todas as plataformas de trabalho que o projeto exige, desta maneira foi selecionado o Banco de

Dados Oracle, que é um Banco de Dados comercial e portável para quase todos as plataformas necessárias. Trazendo uma facilidade na implementação do novo módulo (Cyclops Oracle) por ter acesso nativo na ferramenta de desenvolvimento Smalltalk da ObjectShare versão 3.0.

Com este novo módulo o projeto Cyclops cria-se uma nova imagens de seus produtos, onde estas imagens trazem uma visão mais comercial dos produtos criados, juntamente com a segurança que o Banco de Dados Oracle oferece.

No futuro haverá a necessidade da conexão a outros Bancos de Dados que possuam as mesmas características de portabilidade, segurança e facilidade de publicação de dados em Web, idênticas ao Banco de Dados Oracle, para permitir aos clientes do projeto Cyclops optar pelo seu Banco de Dados que mais lhe convier.

Referências

[1] A. von Wangenheim, Cyclops - Ein Modell zur wissensbasierten Bildinterpretation arn Beispiel von Kernspintomographien. Dissertation, University of Kaiserslautern (1996).

[2] ACR/NEMA Standard, Digital Imaging and Communications in Medicine (DICOM).

[3] http://dicomctn.wustl.edu/DICOM/ctn.html.

[4] P.Chen, The Entity-Relationship Approach to Logical Data Base Design.

[5] http://www.inf.ufsc.br/~cyclops

GLOSSÁRIO

SQL	Linguagem estruturada de pesquisa
ORACLE	Banco de dados relacional
RDBMS	Sistema de gerenciamento de banco de dados relacional
TPS	Transações por segundo
UML	Linguagem unificada de modelagem
PK	Chave primaria
FK	Chave estrangeira
CTN	Central Test Node
ODBC	Open Database Connectivity (Abertura de conexão a banco de dados)
DER	Diagrama de entidade-relacionamento

REFERÊNCIAS BIBLIOGRÁFICAS

AULT, Michael R. Oracle 8 black book. Arizona: Coriolis Group Books, 1997.

ABBEY, Michael. Oracle: guia do usuário. São Paulo: Makron Books, 1997.

BARBIERI, Carlos. Modelagem de Dados. Rio de Janeiro: Infobook, 1994.

BECK, Kent. Smalltalk best practice patterns. London: Prentice-Hall, 1997.

BODROWSKI, Steven M. Dominando o Oracle. São Paulo: Makron Books, 1995.

BOOCH, Grady. UML, Guia do usuário. Rio de Janeiro: Campus, 2000.

BURLESON, Donald K. Oracle - aplicações em bancos de dados. Rio de Janeiro: Ciência Moderna, 1996.

CHEN, Peter. Modelagem de dados. São Paulo: McGraw-Hill, 1990.

COUGO, Paulo Sérgio. Modelagem conceitual e projeto de banco de dados. Rio de Janeiro: Campus, 1997.

DATE, C. J. Guia para o padrão SQL. Rio de Janeiro: Campus, 1989.

DICOM RT Working Group. DICOM in radiotherapy. Obtido via internet: http://www.sgsmp.ch/bull983b.html

ERIKSSON Hans-Erik, Penker Magnus. UML toolkit. New York: John Wiley & Sons, 1961.

FREEZE, Wayne S. SQL - Guia de referências do programador. Rio de Janeiro: Ciência Moderna, 1998.

FURLAN, José Davi. Modelagem de objetos através da UML. São Paulo: Makron Books, 1998.

GIACHETTI, Andrea. A study on the use of CTN as Dicom Image server. Obtido via internet: http://www.crs4.it/~giach/GROUP/ctn.html

GIACHETTI, Andrea. Medical images transfer on heterogeneous networks. Obtido via internet: http://www.crs4.it/~giach/GROUP/trans.html

HURSCH, Jack L. Usando Oracle versão 6.0. Rio de Janeiro: Campus, 1991.

LARMAN, Craig. Utilizando UML e padrões: Uma introdução à análise e ao projeto orientado a objetos. Porto Alegre: Bookman, 2000.

LEWIS, Simon. The art and science of Smalltalk. New York: Hewlett-Packard, 1995.

MORAIS, Rinaldo de Oliveira. Oracle 7 server: conceitos básicos. São Paulo: Érica, 1995.

OBJECTSHARE. Database connect application developer's guide. California: ObjectShare, 1999.

PARCPLACE-DIGITALK. VisualWorks database tools, tutorial and cookbook. Sunnyvale: 1995.

PINSON, Lewis J. An introduction to object-oriented programming and Smalltalk. New York: Addison-Wesley, 1988.

PLETZKE, Jonathan. Advanced Smalltalk. New York: John Wiley & Sons, 1997.

PRESSMAN, Roger S. Engenharia de software. São Paulo: Makron Books, 1995. 3° ed.

RAHAL JUNIOR, Nelson A. S. Implementação de um Cliente Network DICOM Em Smalltalk para conexão ao banco de dados Oracle. São Paulo: LAPTEC 2000 I Congresso de lógica aplicada à tecnologia, 2000.

RAMALHO, José Antonio Alves. Oracle - Personal Oracle 7.3 & Power Objects. São Paulo: Makron Books, 1997.

REZENDE, Denis Alcides. Engenharia de software e sistemas de informação. Rio de Janeiro: Brasport, 1999.

TAYLOR, David A. Object-oriented technology: a manager's guide. New York: Addison-Wesley, 1995.

WANGENHEIM, von Aldo. Object-oriented implementation of a DICOM network client in Smalltalk. Florianopolis: UFSC.

ÍNDICES

ÍNDICE DE AUTORES

ÍNDICE SISTEMÁTICO