Resumo

O diagnóstico clínico é um procedimento fundamental na prática da medicina. Ele é a base para a escolha de um tratamento eficaz. Todavia a habilidade de se identificar uma enfermidade que agride o paciente a partir dos sinais e sintomas que o mesmo apresenta não é uma habilidade demonstrada em igual escala por todos os médicos. Elaborar um diagnóstico correto não é uma tarefa trivial e exige, além do conhecimento médico e da experiência por parte do profissional de saúde, um elaborado raciocínio clínico.

Devido à enorme quantidade de elementos informativos que um médico necessita para prática da medicina, em especial para realizar diagnósticos, sistemas de informação computacionais se apresentam como uma poderosa ferramenta para o manuseio de todo esse volume de informações e para o processamento das incertezas médicas associadas.

Este documento propõe a utilização de uma ferramenta computacional distribuída para sistematizar todos os dados disponíveis à cerca de determinadas doenças e, a partir da informação proveniente desses dados, ser capaz de elaborar diagnósticos plausíveis diante de novos casos.

Neste trabalho foram desenvolvidas duas ferramentas para dar suporte ao profissional médico em sua prática clínica: (1) uma aplicação servidora, cujas atribuições são guardar os dados de forma sistematizada e deles extrair informações relevantes para os futuros diagnósticos, e (2) uma aplicação cliente, cujo principal objetivo é fazer consultas remotas às bases de dados da primeira aplicação citada. Como principais resultados obtidos podem-se ser destacados: as funcionalidades não triviais oferecidas pela ferramenta desenvolvida para profissionais de saúde, a possibilidade de realizar a mineração dos dados da aplicação servidora e a potencialidade (conveniência) trazida pela aplicação ao ser usada como ferramenta de apoio pedagógico por estudantes de medicina.

Abstract

The clinical diagnosis is a fundamental procedure in the practice of medicine. It is the basis to make the right choice for an effective treatment. The hability to produce a sound diagnosis is not a trivial task and requires, in addition to medical knowledge and experience by the health professional, an elaborate clinical reasoning. Due to the enormous amount of information that are needed for a doctor to practice medicine, computerized information systems are presented as a powerful tool in the management of the entire amount of information and processing of medical uncertainties associated.

This document proposes the use of a distributed computing tool to systematize all the available data about diseases and, from the information acquired from these data, be able to draw on plausible diagnoses of new cases.

In this study we developed two tools to support the medical professional in their clinical practice: (1) an application server, whose tasks are to save the data in a systematic way and extract relevant information for future diagnoses, and (2) a client application whose main goal is to make remote queries to the databases of the first mentioned application. As main results obtained we can highlighted: the non-trivial features offered by the tool developed for health professionals, the possibility of data mining application server and the capability (convenience) brought the application to be used as a tool to support teaching by medical students.

Sumário

Índice de Figuras

Índice de Tabelas

Tabela de Símbolos e Siglas

SAD – Sistema de Apoio a Decisão

NUTES – Núcleo de Telessaúde

HUOC – Hospital Universitário Oswaldo Cruz

UPE – Universidade de Pernambuco

CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico

SIG – Sistema de Informações Gerenciais

SPT – Sistema de Processamento de Transações

SGIH – Sistema Integrado de Gestão Hospitalar

SADC – Sistema de Apoio a Decisão Clínica

AD – Árvore de Decisão

ID3 – *Iteractive Dichotomiser 3*

CART – *Classification and Regression Tree*

TCP - *Transmission Control Protocol*

UDP - *User Datagram Protocol*

API - *Application Programming Interface*

GUI - *Graphical User Interface*

ES – Entrada e Saída

SE – Sistema Especialista

ROC - *Receiver Operating Characteristic*

UCI – *University of California, Irvine*

Capítulo 1

Introdução

1.1 Caracterização do Problema

A Semiologia Médica, ou Propedêutica, é o ramo da medicina relacionado ao estudo dos sinais e sintomas das doenças. Utilizando-se dos conceitos e técnicas adquiridos nessa área do conhecimento médico o profissional de saúde é capaz de criar hipóteses de diagnósticos e posteriormente diagnosticar corretamente um paciente. Todavia o domínio da Semiologia é muito complexo e de aquisição demorada e trabalhosa, implicando no domínio de vários componentes: conhecimento da fisiologia normal e dos múltiplos mecanismos de doença, mestria dos métodos e técnicas de colheita de dados, sejam eles a história clínica, a observação psicológica ou o exame físico, e a capacidade de interpretação dos dados recolhidos[1].

Os sinais e sintomas utilizados pelo médico para elaboração do diagnóstico são provenientes da anamnese e de exames complementares. Com estes dados em mãos, auxiliado de seu conhecimento técnico e de sua experiência clínica o médico é capaz de elaborar diagnósticos corretos.

No entanto, dado que a concepção de um diagnóstico requer tanto de uma experiência clínica anterior quanto de uma capacidade de interpretação dos sinais e sintomas observados no paciente, este processo admite imprecisão. Além dos fatores citados também é notória a existência de fatores externos ao caso clínico que podem vir a influenciar o diagnóstico final. Tais como fatores ligados ao estilo da prática do profissional, fatores relacionados com os recursos disponíveis no momento da consulta, incentivos financeiros, dentre outros [2].

1.2 Motivação

No cenário descrito se encontra a problemática do diagnóstico médico, uma atividade essencial na prática médica que além de admitir imprecisões em sua concepção, requer de experiência na prática da medicina interna para ser bem elaborado. Diversas ocorrências resultantes de casos de emergência médica onde medicamentos foram administrados incorretamente (levando em conta o perfil e o trajeto clínico do paciente) estão documentadas [3]. Infelizmente, na maioria dos casos, estas situações são constatadas tardiamente[4].

Em oposição ao panorama acima estão os Sistemas de Apoio a Decisão (SADs), com a capacidade de criar mecanismos capazes de armazenar informações de maneira ativa sob a forma de memória organizacional. A partir destas informações previamente registradas, os SADs podem auxiliar a resolver, auxiliados de técnicas de Inteligência Artificial, diversos tipos de problemas, tais como problemas de classificação, de otimização, dentre outros. Além destas características vale ressaltar que sistemas computacionais jamais estarão sujeitos a algumas características humanas indesejáveis, como fadiga ou falta de foco (*e.g.* por preocupação).

O projeto aqui descrito integra um projeto maior, cujo título é “Suporte Remoto a Diagnóstico Médico Utilizando Tecnologias Inteligentes” que vem sendo desenvolvido pelo Núcleo de Telessaúde (NUTES), sediado no Hospital Universitário Oswaldo Cruz (HUOC) da Universidade de Pernambuco (UPE). Onde o aluno autor deste trabalho teve a oportunidade de desenvolver atividades como pesquisador bolsista do CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico).

1.3 Objetivos e Metas

O diagnóstico correto é fundamental para a escolha do tratamento mais adequado a cada caso e, conseqüentemente, para a restauração da saúde do paciente. Assim sendo, o profissional de saúde deve exercer o seu ofício atrelado a uma margem mínima de erro, considerando as possíveis conseqüências subjacentes a um determinado erro clínico.

Diante disto existe uma séria problemática envolvida no processo: diversos fatores podem alterar a linha das hipóteses do médico no momento da elaboração do diagnóstico.

Para auxiliar o patamar de precisão requerido em diagnose médica e assim ser instrumental tecnológico para clínicos e profissionais de saúde, este trabalho se propõe a desenvolver um sistema de informação inteligente, que auxilie profissionais de saúde no momento da composição de diagnósticos plausíveis.

Além disso, o propósito da ferramenta é de aumentar a consistência em diagnósticos médicos, já que o suporte à decisão vai ser embasado em históricos conhecidos de doenças. Outra característica da ferramenta será auxiliar em diagnósticos diferencias, isto é, diante de enfermidades cujos sintomas são similares, o sistema deverá ser capaz de identificar qual doença aflige o paciente e justificar a escolha.

1.4 Organização do Documento

O documento está dividido em cinco capítulos, resumidos a seguir:

Capitulo 1: Introdução

Contém o texto introdutório sobre o trabalho, caracterizando o problema, abordando a motivação para resolvê-lo e apresentando os objetivos e metas do trabalho.

Capítulo 2: O Diagnóstico Médico

Neste capítulo está descrito como se dá o processo de tomada de decisão em diagnósticos médicos, abordando tópicos como a identificação de sinais e sintomas no paciente, o uso de atalhos cognitivos, e a criação de suposições diagnósticas.

Capítulo 3: Sistemas de Informação

No capítulo três é feita uma breve revisão da literatura no que diz respeito a Sistemas de Informação e suas classificações. Também são abordadas algumas tecnologias utilizadas em Sistemas Distribuídos.

Capítulo 4: O Sistema Desenvolvido

Nesse capítulo são apresentadas as ferramentas desenvolvidas. Nele estão descritas as suas funcionalidades e as tecnologias utilizadas durante o desenvolvimento.

Capítulo 5: Conclusão e Trabalhos Futuros

Nesse último capítulo são comentados os resultados obtidos como também as formuladas conclusões acerca destes resultados. Logo em seguida são feitas as considerações finais e listados os possíveis trabalhos futuros.

Capítulo 2

O Diagnóstico Médico

2.1 Heurísticas na Tomada de Decisão

Heurísticas são regras gerais de influência utilizadas pelo decisor para simplificar seus julgamentos em tarefas decisórias que envolvem incerteza. A tomada de decisão, seja sob risco ou sob incerteza, pode ser entendida a partir de modelos que visam normatizar a tomada de decisão. Os princípios clássicos envolvidos nessas situações são identificar as ações que maximizam a possibilidade de obter resultados desejáveis e minimizar a possibilidade de que ocorram resultados indesejáveis sob condições idealizadas [5]. No que diz respeito ao julgamento e tomada de decisão, as heurísticas assumem a função de simplificar o processamento cognitivo que julga alternativas possíveis associadas a incertezas.

Podem-se citar três tipos de heurísticas normalmente utilizadas em julgamentos sob incerteza e/ou sob risco: (a) ancoragem e ajustamento, geralmente utilizada em casos onde se necessita criar predições numéricas com valor inicial disponível; (b) disponibilidade de instâncias ou cenários, utilizada de acordo com a própria experiência obtida pelo decisor, geralmente associada à análise da plausibilidade de um desenvolvimento particular, e (c) representatividade, empregada quando o decisor detém dados estatísticos do determinado acontecimento a ser julgado, comumente utilizada para julgar a probabilidade de um evento ou objeto A pertencer à classe ou processo B.

2.1.1 Heurísticas no Diagnóstico Médico

Ao avaliar um paciente, os clínicos utilizam da heurística a fim de extrair informações relevantes dos dados coletados na avaliação clínica [6]. Neste cenário o uso de heurística é essencial, tendo em vista o complexo universo que compreende o domínio do estudo em questão. Com o apoio dos atalhos

cognitivos, traçados com o auxilio das heurísticas, o médico consegue diminuir a complexidade do problema a um nível acessível e assim elaborar o diagnóstico. Os três tipos de heurísticas descritos acima são utilizados pelos clínicos na criação de diagnósticos.

Ao avaliar um paciente, os clínicos freqüentemente ponderam a probabilidade de que as manifestações clínicas deste indivíduo sejam compatíveis com aquelas do grupo de pacientes com as principais hipóteses diagnosticas sob consideração [7]. Em outras palavras, o médico procura tornar o paciente um exemplo representativo, utilizando assim a heurística representativa. Apenas algumas características podem ser o suficiente para que um clínico experiente use a heurística representativa para chegar a uma hipótese diagnostica sensata. No entanto os médicos que usam da heurística representativa podem chegar a conclusões errôneas, caso não levem em consideração a prevalência intrínseca de dois diagnósticos concorrentes.

A heurística de disponibilidade aplicada à medicina envolve um histórico de casos vivenciados pelo médico e capacidade de recordá-los. A tomada de decisão tem como base a lembrança de casos e resultados anteriores. Problemas relacionados à memória interferem neste processo: casos mais recentes acabam se tornando mais preponderantes na avaliação clínica, assim como catástrofes raras (que geralmente são lembradas com uma clareza desproporcional a sua importância), dentre outros.

O terceiro método para se criar um atalho cognitivo, a ancoragem e ajustamento, também é comumente utilizado em diagnósticos. Neste caso o médico delimita uma grande área na qual assume que a patologia esteja inserida e, a partir disso, utiliza dos sinais e sintomas do paciente para se aproximar ao máximo da doença em questão para que, finalmente, possa criar seu diagnóstico. Esta heurística pode apresentar erros se na primeira decisão, relativa à grande área médica em questão, for mal definida: uma decisão errônea neste aspecto faria com que o médico tendesse a devagar durante a tentativa de ajustamento. Em doenças sistêmicas, por exemplo, nota-se a difícil especificação da grande área médica que causa todos os sintomas.

2.2 Formulação da Hipótese Diagnostica

Estudos cognitivos[8] acerca dos processos de raciocínio em clínicos experientes afirmam que estes agrupam as informações em sua memória de forma segmentada (na forma de pacotes ou “pedaços”). Estes segmentos são manuseados e organizados de forma a criar hipóteses diagnósticas. Como a memória de curto prazo geralmente guarda apenas 7 a 10 itens por vez, a quantidade de informação que pode ser integrada ativamente para a formulação das hipóteses também é limitada. Neste ponto entram em cena os atalhos cognitivos anteriormente citados.

Estas hipóteses irão direcionar não só a anamnese, selecionando as perguntas que mais se adéqüem às opções em questão, como também o exame físico e a necessidade de solicitar exames complementares e, caso necessário, indicar os que mais se prestam a cada caso.

Embora as heurísticas de representatividade e de disponibilidade possam desempenhar funções importantes na formulação das primeiras hipóteses diagnósticas, o grau de agudeza da doença do paciente também pode ser muito importante. Os clínicos aprendem a importância de considerar diagnósticos pouco prováveis em suas hipóteses diagnósticas baseados no fato de que diagnósticos relativamente raros ou catastróficos não são facilmente identificados, a menos que sejam considerados explicitamente. Nas situações não-emergenciais, a prevalecência dos diagnósticos alternativos em potencial deve desempenhar um papel muito mais importante na formulação das hipóteses diagnósticas. Nunca é demais enfatizar o valor de se realizar um levantamento clínico rápido e sistemático dos sintomas e sistemas orgânicos, para evitar que indícios importantes e pouco evidentes passem despercebidos.

Como a formulação e a avaliação das hipóteses diagnósticas pertinentes é uma habilidade que nem todos os clínicos possuem no mesmo grau, podem ocorrer erros nesse processo e tais erros podem gerar conseqüências trágicas, caso o paciente tenha uma doença aguda e grave[7].

Também é válido salientar que em quadros clínicos reais os sintomas podem diferir dos descritos nos manuais, então se torna fundamental a adaptação do processo diagnóstico aos desafios do mundo real.

O especialista, mesmo estando habituado a tratar certos casos corriqueiros, aborda cada caso considerando seriamente os indícios de que o diagnóstico inicial possa estar errado. Muitas vezes, os pacientes fornecem informações que aparentemente não se encaixam em qualquer uma das hipóteses diagnósticas principais em consideração. A diferenciação entre os indícios verdadeiros e as pistas falsas somente é possível com prática clínica e com experiência incorporada.

2.3 Principais Influências na Tomada de Decisões Clínicas

Anos de pesquisa acerca das variações dos padrões da prática clínica elucidaram algumas forças que determinam as decisões clínicas. Como dito anteriormente, o uso de atalhos cognitivos proporciona uma explicação parcial, todavia existem vários outros fatores que desempenham um papel fundamental na formulação das hipóteses diagnósticas. Por definição, estes fatores podem ser agrupados em três categorias que se sobrepõem: (1) fatores relacionados a características pessoais e estilo de prática do médico; (2) fatores relacionados com o contexto no qual atua; e (3) fatores de incentivo financeiro.

2.3.1 Fatores Relacionados ao estilo de prática

Um dos principais papéis de um clínico dentro da assistência médica é assegurar que toda a assistência necessária seja prestada com alto nível de qualidade. Os fatores que influenciam esta função são o conhecimento, o treinamento e a experiência do médico. Para praticar medicina baseada em evidências os médicos devem estar familiarizados com estas evidências. Como é de se esperar, especialistas normalmente conhecem melhor as evidências relacionadas às enfermidades de sua área do que os clínicos

gerais, logo, os especialistas têm uma probabilidade maior de realizar um diagnóstico correto e um tratamento com sucesso.

2.3.2 Fatores Relacionados ao Contexto da Prática

Os fatores reunidos neste grupo estão relacionados aos recursos disponíveis para a prática clínica e ao cenário no qual atua. “Demanda induzida pelo médico” é uma expressão que descreve a surpreendente capacidade de alguns médicos de se adaptar ao ambiente no qual atua e usar os recursos médicos que lhe são disponíveis. Muitas vezes a falta de recursos tecnológicos ou de especialistas para pareceres e procedimentos também estão enquadrados neste grupo.

2.3.3 Incentivos Financeiros

Questões financeiras podem exercer influências estimulantes ou inibitórias sobre a prática médica. Em geral o médico pode ser remunerado por serviços prestados ou por salário [6]. No sistema de pagamento por serviço prestado, quanto maior a produção, ou seja, quanto maior o número de pacientes atendidos, maior o retorno financeiro do médico. Isso estimula a redução do tempo da consulta, visando aumentar a produtividade. Quando a forma de pagamento é por salário o médico dedica um tempo maior ao paciente. Custos de exames também podem exercer influência sobre a forma de raciocinar do médico.

A tomada de decisões clínicas pode ser entendida como uma complexa relação entre estratagemas cognitivos usados para simplificar grandes quantidades de informação e as particularidades de cada profissional, particularidades estas relativas à sua formação, treinamento e experiência.

A ferramenta proposta neste trabalho utiliza os principais tipos de heurísticas citados no início do capítulo. Um dos fundamentos da aplicação, como será visto mais adiante, é organizar de forma sistemática as informações. Esta nova organização das informações tende a auxiliar o clínico quando este vier a fazer uso da heurística da representatividade. As informações serão apresentadas de forma clara e objetiva.

Outro tipo de heurística, a ancoragem e ajustamento, pode ser percebida durante o funcionamento da ferramenta. De acordo com que lhe são passados os dados o próprio algoritmo tende a delimitar um escopo no qual a doença em questão está inserida. Conforme os dados lhe são passados o sistema diminui o escopo até conseguir extrair uma sugestão de diagnóstico plausível.

A heurística de disponibilidade também é afetada ao se fazer uso da aplicação proposta. Como dito antes, ao utilizar este tipo de heurística o médico utiliza sua memória para se basear em casos previamente vivenciados no momento da elaboração do diagnóstico. Esta heurística é potencializada pela ferramenta que, ao receber as informações, incorpora as experiências vividas por vários profissionais de saúde e as cataloga de maneira compreensível.

Capítulo 3

Sistemas de Informação

Um Sistema de Informação pode ser definido tecnicamente como um conjunto de componentes inter-relacionados que coleta (ou recupera), processa, armazena e distribui informações destinadas a apoiar decisões, a coordenação e o controle de uma organização[9]

A figura 2 representa os componentes básicos de um Sistema de Informação. Neste sistema está contida informações sobre o ambiente que o cerca.

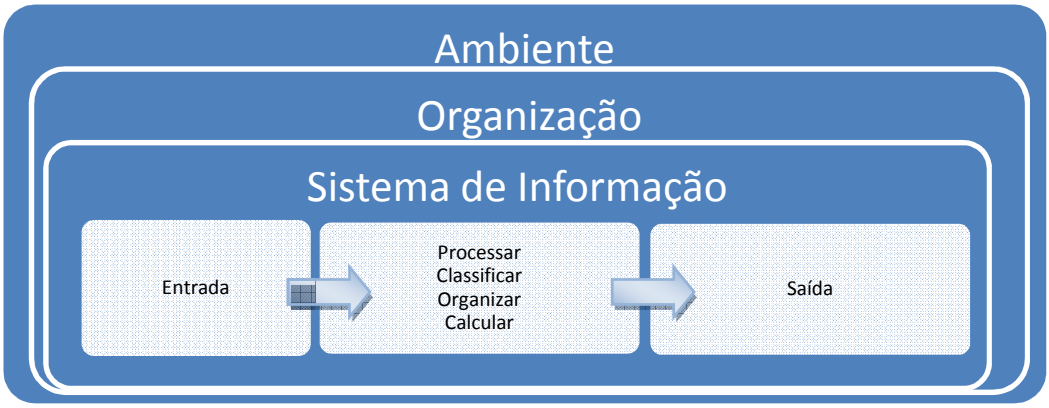

Figura 1. Funções de um Sistema de Informação

As informações (o resultado do processamento e organização de dados de tal forma que represente um significado, ou seja, o significado extraído a partir de um dado) são produzidas a partir de três atividades básicas descritas acima - entrada, processamento e saída. Além destas três atividades é comum a existência de outro fator: O *feedback*, ou retroalimentação, que consiste no envio das informações produzidas na saída para a entrada, de forma que estas informações são utilizadas para análise e refinamento do sistema.

Como qualquer organização está dividida por diferentes níveis, interesses e especialidades, existem sistemas com variados objetivos. Apesar disso, pode-se dividir os Sistemas de Informação em quatro principais níveis:

sistemas do nível operacional, do nível de conhecimento, do nível gerencial e do nível estratégico.

Sistemas do Nível Operacional

Sistemas do nível operacional dão apoio aos gerentes operacionais, sua principal função é acompanhar atividades e transações básicas da organização, como recursos humanos, fluxo de produção, etc.

Sistemas do Nível de Conhecimento

Sistemas do nível de conhecimento atendem aos funcionários responsáveis pela gestão dos dados da empresa. O propósito geral dos sistemas deste nível é auxiliar o controle do fluxo de documentos gerados.

Sistemas do Nível Gerencial

Sistemas do nível gerencial dão suporte a atividades e procedimentos administrativos dos gerentes, tais como monitoração, controle e tomada de decisões. Uma característica deste tipo de sistema é a produção periódica de relatórios sobre as mais recentes operações efetuadas pelos níveis inferiores da organização. Dentre os sistemas de informações deste nível pode-se citar os de suporte a decisão imediata, no qual os gerentes podem apoiar suas decisões numa base de conhecimento organizacional estruturada.

Sistemas do Nível Estratégico

Sistemas do nível estratégico são utilizados pela mais alta cúpula das corporações para enfrentar questões estratégicas e tendências de longo prazo. Além de criar informações inerentes a estruturação da empresa e dos processos contidos nela, também é responsável por analisar o ambiente externo e, a partir daí, oferecer bases para tomadas de decisões de longo prazo. Um sistema de previsão de tendências de venda no período de cinco anos atende ao nível estratégico[9].

A Figura 2 apresenta os seis principais tipos específicos de sistemas de informação correspondentes a cada nível organizacional explanado acima.

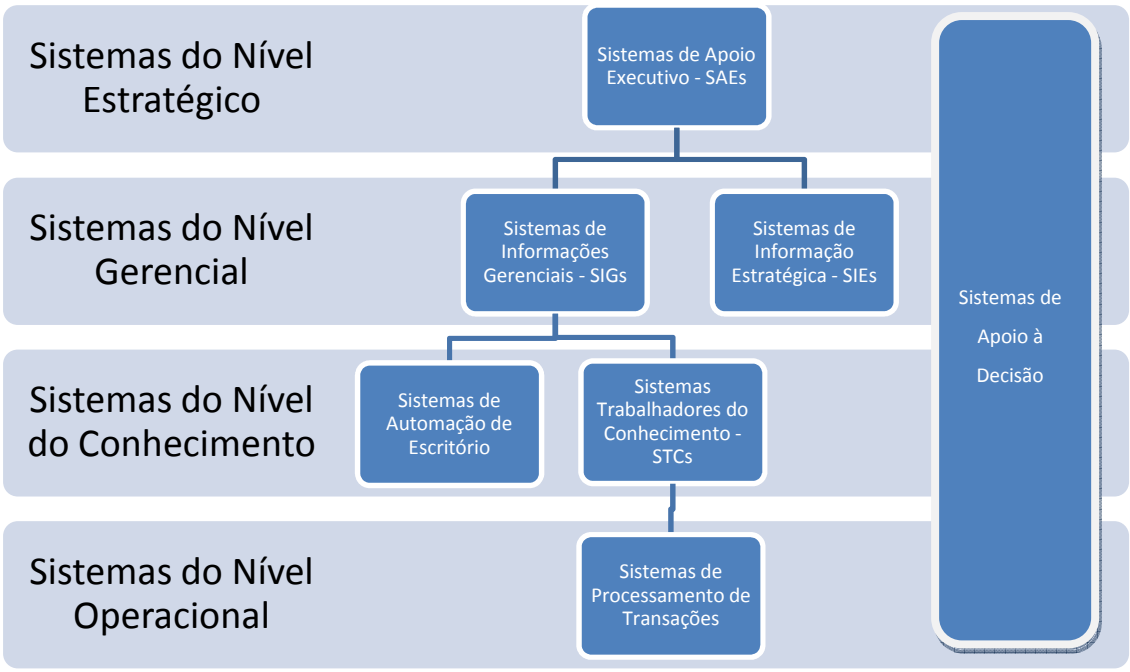

Figura 2. Os seis principais tipos de Sistemas de Informação

3.1 Sistemas de Apoio a Decisão

Sistemas de apoio a decisão (SADs) surgiram como uma especialização dos sistemas de informações gerenciais (SIGs), combinando as informações que por definição já estariam contidas nestes e novas técnicas de computação (incluindo conceitos de retro-alimentação). SADs podem ser inseridos em qualquer nível da organização. Os SADs ajudam os gerentes a tomar decisões não-usuais, que se alteram com rapidez e que não são facilmente especificadas com antecedência[9]. Quando utilizados pelos gestores da organização geralmente trabalham com as informações previamente obtidas dos sistemas de processamento de transações (SPTs) e dos SIGs. Pela sua própria concepção os SADs apresentam um maior poder analítico frente a outros tipos de Sistemas de Informação. Geralmente são dotados de inteligência artificial e são construídos com base numa variedade de modelos para analise de dados e informações.

3.2 Sistemas de Informação em Medicina

O grande crescimento das organizações de saúde trouxe a elas uma grande complexidade no que diz respeito à gestão do conhecimento, isto trouxe a necessidade do uso de sistemas de informação dentro destas organizações. A freqüência do uso de Sistemas Integrados de Gestão Hospitalar (SIGH) em grandes hospitais vem crescendo a cada ano. Vem sendo alavancada pela necessidade da organização sistematizada da informação dentro destes centros médicos. A Figura 3 demonstra as principais atribuições de um SIGH.

A gestão do conhecimento em organizações de saúde é fundamentada nos recursos humanos da organização, recursos como especialidades individuais, capacidade de resolução de problemas e, principalmente, aprendizagem acumulada. O objetivo principal desta gestão deve trazer à equipe que compõe a organização a mais perfeita partilha do conhecimento possível.

Esta partilha de conhecimento clínico permite agilizar processos e incrementa a qualidade da comunicação entre unidades de saúde [4]. Sistemas de Informação podem ser organizacionalmente dispostos em relação a sua finalidade no campo da saúde. Dentro dos diversos tipos de sistemas de informação voltados para a área médica existe um tipo que está ligado intimamente ao conteúdo deste trabalho: os Sistemas de Apoio a Decisão Clínica.

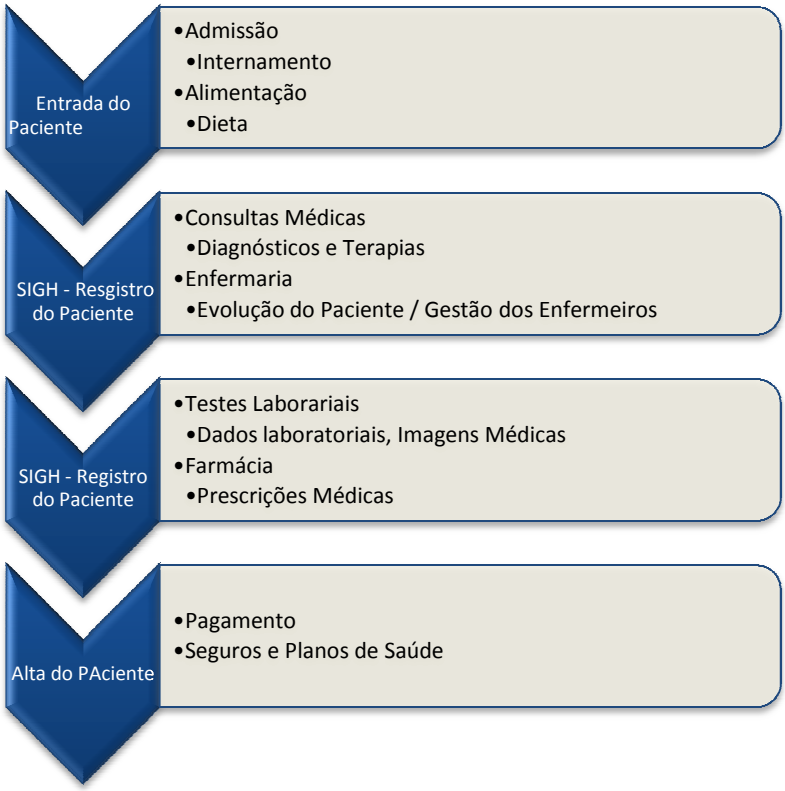

Figura 3. SIGH - Rotinas

Sistemas de Apoio à Decisão Clínica

O Sistema de Apoio à Decisão Clínica (SADC) surge como o resultado da aplicação de SADs na área de saúde, mais especificamente no apoio na elaboração de diagnósticos médicos. A concepção de um diagnóstico médico é o resultado da análise das informações a cerca do paciente, informações clínicas e informações patológicas. A elaboração de um diagnóstico, assim como a obtenção de respostas a questões respeitantes à etiologia de determinadas doenças, pode ser assistida por SADCs [4].

Sistemas informatizados de apoio à decisão baseados em algoritmos matemáticos e IA tentam simular o raciocínio médico em suas tomadas de decisão[10], utilizando a capacidade organizacional provida pelos SIs em medicina. Técnicas como Redes Neurais, modelos Bayesianos e Árvores de decisão, utilizadas nesses tipos de sistemas, são instrumentos computacionais e matemáticos que podem auxiliar na escolha de ações médicas.

Existem exemplares de SADCs em uso rotineiro em diversas instituições ao redor do mundo. Por exemplo, pode-se citar o DXplain [11], desenvolvido pelo Hospital de Massachusetts - EUA, um SDAC que a partir de um determinado conjunto de dados clínicos a cerca de um paciente consegue definir um conjunto de possíveis diagnósticos deste caso. O QMR[12](*Quick Medical Reference*) é outro exemplo representativo de um SAD clínica. O QMR tem associado uma extensa base de dados onde pode-se encontrar informação médica sobre doenças, sintomas, sinais, e informação laboratorial. O QMR tem capacidade de sugerir diagnósticos e terapias, descrever as origens, sintomas e a natureza de determinadas patologias, e apresentar resultados atualizados de testes laboratoriais. Muitos outros repositórios na internet fornecem referências a SADCs baseados no uso de Inteligência Artificial atualmente em uso[12]. Além disso, um volume considerável de dados clínicos podem ser reunidos de modo a criar bases de dados globais, como são os casos da Medline[14] e dos recursos do NCBI[15].

3.3 Árvores de Decisão

A indução por Árvores de Decisão (AD) é uma das formas mais simples, e ainda assim, mais bem-sucedidas de algoritmos de aprendizagem[13]. Esta técnica se baseia na abordagem “dividir para conquistar” [14], isto é, ela divide o conjunto de amostras de treino sucessivamente gerando assim diversos subconjuntos, estas divisões findam no momento em que todos os elementos de cada subconjunto gerado pertençam a uma mesma classe.

A partir destes resultados das sucessivas divisões, que, na verdade, são os dados amostrais organizados de maneira compacta, pode-se montar uma estrutura de árvore pronta para classificar novos exemplos. A Figura 4 apresenta a representação de um classificador

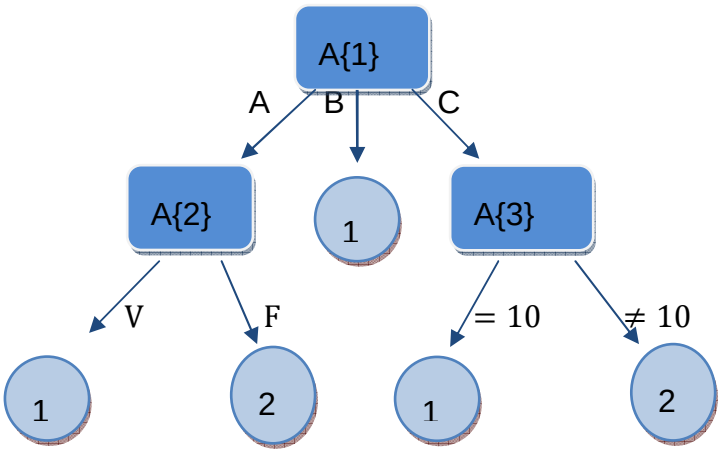

Figura 4. Exemplo de um classificador utilizando uma Árvore de Decisão.

No exemplo acima cada nó retangular representa um atributo. Estes atributos estão dispostos na árvore (em forma de nós) de acordo com seu potencial de classificação, isto é, atributos que conseguem dividir melhor (mais a frente serão abordadas maneiras para se mensurar a qualidade desta divisão) o conjunto de treino se encontrarão em níveis mais altos. Cada aresta é um possível valor relacionado ao atributo ao qual pertence. E os círculos no final de cada ramo da árvore (seriam as folhas da mesma) representam uma classe. Desta forma um teste de classificação seria feito percorrendo a árvore e, ao final do caminhamento, teria se a resposta se a determinada amostra pertenceria à classe "*1*" ou a classe "*2*".

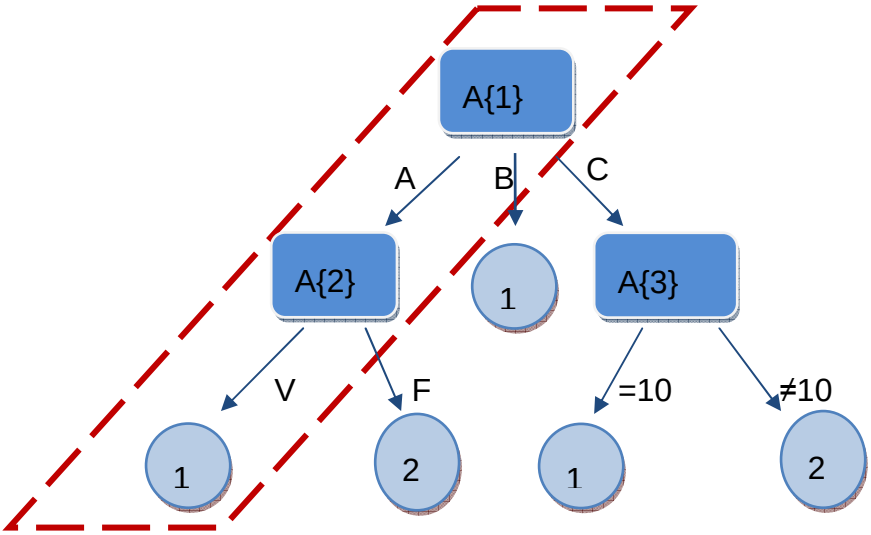

Figura 5. Exemplo de teste de classificação.

Na Figura 5 pode-se ver um exemplo de teste de classificação. O teste sempre deve ser iniciado no nó raiz, neste caso no nó cujo atributo é A{1}. A partir deste nó deve-se caminhar ao longo da árvore de acordo com os valores associados a cada atributo da determinada amostra. No exemplo dado, a amostra tem o valor "*A*" para o atributo A{1} e valor "*V*" (que poderia ser interpretado como verdadeiro, por exemplo) para o atributo A{2}, desta forma a amostra é classificada como pertencente à classe "*1*".

Através de rotas dentro da árvore pode-se também extrair regras a cerca da classificação dos atributos. A seguinte regra pode ser obtida através do caminhamento da amostra da Figura 5: *"Se A{1} = A e A{2} = V , então 1*"

3.3.1 Técnicas de Construção de Árvores de Decisão

Como foi detalhado na seção anterior, árvores de decisão são excelentes estruturas para trabalhar com problemas onde o objetivo principal é a classificação dos atributos que lhe foram passados, todavia também é notória a aplicação de ADs em problemas de regressão. No entanto, para diferentes tipos de aplicação se fazem necessárias diferentes técnicas para construção das ADs. Considerando os tipos de aplicações (tipos de problemas existentes) e as diversas técnicas para construção de árvores de decisão pode-se estabelecer a seguinte relação:

Problemas de Classificação: problemas onde o resultado da predição é a classe a qual os dados pertencem são chamados problemas de classificação. Uma das técnicas mais indicadas para resolução deste tipo de problema é o ID3 (*Iterative Dichotomiser 3*) proposto por Quinlan[14];

Problemas de Regressão: existem também problemas cujo resultado da predição não é uma classe mas número real (problemas de regressão). Para este tipo de problema existe outra técnica também proposta por Quinlan: o C4.5 [15];

Problemas mistos: também pode ser encontrados problemas com características tanto de classificação quanto de regressão, para este tipo

de problema foi proposto o CART (*Classification and Regression Trees*) [16].

Como a problemática que envolve os SDACs, aqui proposta, é caracterizada por ser categoricamente um problema de classificação na próxima sub-seção será explanado detalhadamente como funciona o algoritmo ID3.

3.3.2 O Algoritmo ID3

O algoritmo ID3 constrói a AD recursivamente de maneira *top-down*, ou seja, começa na raiz e se estende até as folhas. A sequência (simplificada) de etapas realizadas por ele pode ser conferida na Figura 6.

O critério utilizado para a escolha do atributo que melhor divide a amostra (conforme descrito no item 1 da Figura 6) no ID3 é o Ganho de Informação. O ganho de informação surge de cálculos baseados numa medida chamada entropia[15] [17].

1. Escolher um atributo que melhor divida os valores do atributo de saída;
2. Criar uma ramificação separada para cada valor do atributo escolhido;
3. Dividir as instâncias em subgrupos de acordo com o valor de cada instância para o atributo em questão;
4. Para cada subgrupo, finalize o processo de seleção de atributo se:
    a. Todos os membros do subgrupo apresentem o mesmo valor para o atributo de saída, crie uma folha com o valor do atributo de saída;
    b. O subgrupo apresente quantidades desprezíveis para cada possibilidade, crie um no folha com o valor da maioria;
5. Para cada subgrupo criado que ainda não tenha sido rotulado como folha, repita os processos acima.

Figura 6. Algoritmo ID3 Simplificado, extraído de Zoby, 2006 [6]

O papel da entropia seria representar o grau de aleatoriedade de um determinado conjunto de instâncias, de forma que o ganho de informação de um atributo seria então a redução da entropia geral resultante da escolha do determinado atributo. A entropia pode ser definida pela Fórmula 1 a seguir:

$$Entropia(S) = \sum_{i=1}^{n} -p_i log_2 p_i \quad (1)$$

Onde: S é o conjunto de exemplos

n é o número de classes

p_i é a probabilidade de S pertencer a classe i.

A probabilidade p_j pode ser facilmente descrita através da Fórmula 2:

$$p_i = \frac{|S_i|}{|S|} \quad (2)$$

Onde: $|S_i|$ é o número de exemplos classificados como i

$|S|$ é o número total de exemplos do conjunto S

Uma classificação é considerada perfeita quando todos os membros do conjunto S pertencem à mesma classe, nesta a ocasião a entropia terá valor igual a zero. Caso um determinado conjunto tenha o mesmo número de representantes de cada classe a entropia assumirá o valor de 1 (um), neste caso é dito que os membros foram classificados ao acaso. Se no determinado conjunto S houver um número diferente de representantes de cada classe a entropia deverá assumir um número entre 0 e 1, de acordo com a Figura 7.

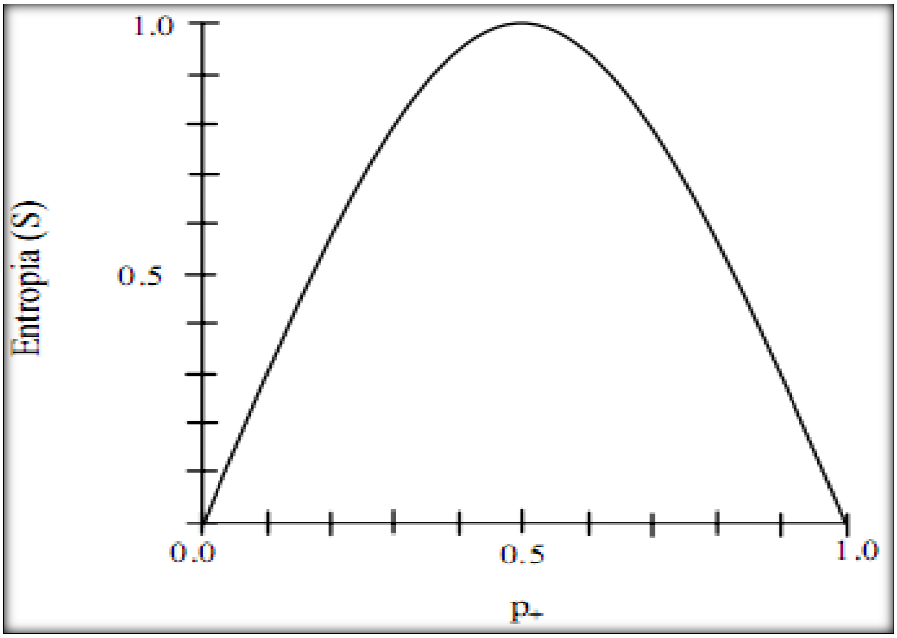

Figura 7. Gráfico da Entropia

A medida do ganho de informação então será simplesmente a redução esperada da entropia causada pela divisão de exemplos de acordo com o atributo escolhido[6]. Desta forma o ganho de informação relativo a um determinado atributo A é definido, na Fórmula 3, por:

$$Ganho\ (S, A) = Entropia(S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} Entropia(S_i) \quad (3)$$

Onde: $Ganho\ (S, A)$ é o ganho do atributo A sobre o conjunto S

$|S|$ é o subconjunto S no qual o atributo A tem valor i

De acordo com o ganho de informação, o atributo com maior ganho, portanto o mais informativo dentre os calculados e que melhor prediz o atributo meta será escolhido para subdividir o conjunto de exemplos. Como pode ser notado na etapa 5 da Figura 6, o algoritmo ID3 é recursivo, formando novos nós a cada chamada até que, por fim, crie as folhas. O ganho será usado em cada chamada ao algoritmo para eleger qual o atributo dentre os restantes deverá gerar o próximo nó. Na aplicação desenvolvida as hipóteses diagnósticas estarão nas folhas das árvores.

3.4 Sistemas Distribuídos

Um sistema distribuído é uma "coleção de computadores independentes que se apresenta ao usuário como um sistema único e consistente"[18], em outra definição, o mesmo pode ser entendido como uma "coleção de computadores autônomos interligados através de uma rede de computadores e equipados com software que permita o compartilhamento dos recursos do sistema: hardware, software e dados"[19].

A partir destes conceitos torna-se clara a necessidade da existência de um software específico para realizar a comunicação e integração de um sistema espalhado por mais de um ponto de acesso. Este tipo de software é chamado de Middleware. Um middleware apresenta serviços comuns de infra-estrutura de software, necessários à grande maioria das soluções de sistema atualmente desenvolvidas no contexto moderno das aplicações comerciais e científicas[20].

Existe uma série de fatores que devem ser observados durante a execução de um middleware para um bom desempenho do mesmo. Estes fatores, também chamados de requisitos não funcionais, são responsáveis por prover determinadas características necessárias aos sistemas distribuídos.

Uma lista dos requisitos não funcionais que os sistemas de middleware devem prover foi proposta por Emmerich[21]. No decorrer desta seção serão explanadas todas estas funcionalidades que um middleware deve oferecer.

3.4.1 Requisitos Não-Funcionais para Middlewares

É notória a presença de uma série de características comuns a qualquer componente se destina a fornecer comunicação entre mais de uma aplicação. Uma lista de requisitos não-funcionais comuns a maioria dos módulos de comunicação encontrados nos mais diversos sistemas é apresentada por Emmerich[21]. Estes requisitos serão discutidos a seguir.

Comunicação de Rede

É a mais básica premissa para classificar um sistema como middleware. Trata-se do suporte à troca de informações e/ou dados de controle através de uma rede de computadores na qual o hospedeiro do middleware está inserido. Geralmente usada a camada transparente de transporte, TCP e UDP. Cabe aos middlewares a complexa tarefa de conversão das estruturas de dados e objetos (quando em ambientes orientados a objeto) em bytes e rajadas de bytes.

Coordenação

Um processo de comunicação pode funcionar de duas maneiras distintas no que diz respeito a coordenação: ele pode trabalhar de forma síncrona ou de forma assíncrona. Na coordenação síncrona o componente que espera o processamento de sua requisição por um segundo componente é mantido bloqueado até obter resposta (ou seja, até que o segundo componente termine o processamento sobre sua requisição e lhe envie a resposta). Na segunda maneira (coordenação assíncrona) não há o bloqueio dos componentes que requisitam o processamento, desta forma, o componente que é requisitado ficará responsável pela coordenação e sinalização de todo o processamento.

Confiabilidade

Diz respeito à garantia de que a informação chegará ao seu destino com corretude e pontualidade. Muitas vezes confiabilidade é sinônimo de redundância, por este fato é comum classificar confiabilidade e desempenho como requisitos conflitantes.

Outro principio da confiabilidade lida com conceito de transação. Uma transação define a atomicidade de um fluxo de execução, isto é, caso alguma das instruções por ventura não venha a ser executada, todo o fluxo de instruções deverá ser desfeito.

Escalabilidade

Em resumo, trata-se da capacidade de uma solução se adequar a um possível aumento de carga sem acarretar grandes impactos no que diz respeito à distribuição física e lógica da solução. O principal método para abordar a escalabilidade é usar a transparência de aplicações, que permite que um serviço seja requisitado sem que o requisitor saiba necessariamente a localização física ou lógica do executor.

Heterogeneidade

É o requisito que torna possível a um sistema trabalhar aplicações heterogêneas, ou seja, que usam tecnologias diferentes em pelo menos um destes três aspectos: plataformas de hardware, sistemas operacionais ou linguagens de programação.

Todos esses aspectos devem ser considerados ao se desenvolver uma solução distribuída, a fim que a mesma possa ser usada em diferentes contextos. Garantir a heterogeneidade de um sistema significa garantir a interoperabilidade entre dispositivos e/ou aplicativos com as mais diversas características.

3.4.2 Middleware Orientado a Objetos

A idéia básica dos *Middlewares* orientados a objetos consiste em oferecer às aplicações desenvolvidas em linguagens de programação orientadas a objetos

métodos capazes de se comunicar com outras aplicações. A Figura 8 mostra uma visão geral sobre as partes componentes de um *middleware* orientado a objetos. Neste contexto pode-se notar a existência de dois elementos essenciais para que a comunicação seja estabelecida: o *stub* e o *skeleton*.

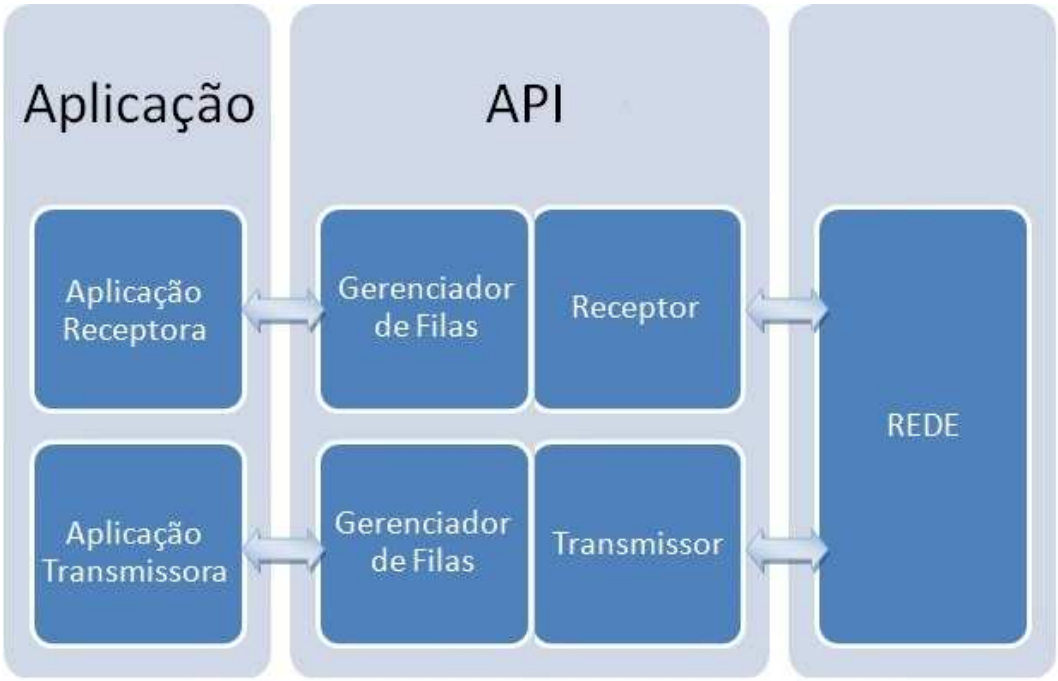

Figura 8. *Middleware* Orientado a Objetos

O *stub* é o componente responsável pela conversão de chamadas e de objetos em dados a serem enviados pela rede, enquanto que o *skeleton* aparece como o elemento que converte dados da rede em chamadas e objetos. Estes componentes dão à aplicação a possibilidade de realizar chamadas remotas de métodos com parâmetros e retornos definidos como objetos de forma transparente e simples.

Todos os requisitos não funcionais anteriormente descritos podem ser atendidos por *middlewares* orientados a objetos, conforme descrito em [22].

A necessidade de distribuição do sistema proposto surigiu pela própria natureza da aplicação. Posto que se trata de um SI (mais especificamente um SADC) e que tem como uma de suas funções a disseminação da informação. Diante desta necessidade optou-se por utilizar um *middleware* orientado a

objetos que tanto atende as especificações dos requisitos não-funcionais citadas por Emmerich, como as necessidades do sistema.

Capítulo 4

SADC Utilizando Árvores de Decisão

O objetivo deste trabalho é o desenvolvimento de um sistema de apoio à decisão para a prática diagnóstica da clínica médica. Além de criar hipóteses plausíveis para diagnósticos com diferentes sintomas, o sistema é capaz de elaborar diagnósticos diferenciais, ou seja, diante de casos com quadros sindrômicos semelhantes o sistema deverá sugerir os diagnósticos mais cabíveis a cada um. Uma das principais características do modelo desenvolvido é sua capacidade de adaptação às diferentes áreas médicas. O único pré-requisito para que o sistema atue numa nova área médica seria a aquisição de dados desta determinada área.

O sistema pode ser dividido em módulos, aonde o principal componente se encontra no módulo que cria o diagnóstico médico. Este componente é uma técnica de Inteligência Artificial discutida no tópico 3.3, Árvores de Decisão. Com esta técnica o sistema é capaz de organizar sistematicamente diversas enfermidades de acordo com seus quadros clínicos, além de poder fazer predições de diagnósticos médicos ao ser apresentado a um novo caso.

Um dos requisitos não funcionais do sistema diz respeito à distribuição da informação. Como dito anteriormente, na seção 3.2 Sistemas de Informação em Medicina, a partilha do conhecimento clínico é de fundamental importância na gestão do conhecimento na área de saúde. Desta forma é de vital importância que o sistema tenha a possibilidade de ser acessado por mais de um médico simultaneamente. Este requisito foi atendido dividindo o sistema em duas ferramentas, uma como aplicação servidora e outra como aplicação cliente, de maneira que haja somente uma instância da árvore em execução (o que evita inconsistência dos dados) e que esta instância esteja disponível para qualquer um que dispor do sistema. Para atender a este requisito foi utilizado

um middleware orientado a objeto (visto na seção 3.4.2 Middleware Orientado a Objetos), mais adiante será descrita de forma mais detalhada o funcionamento da comunicação entre as ferramentas.

Todo o sistema foi desenvolvido com a linguagem de programação Java [23] e foi projetado para execução em computadores pessoais, estações de trabalho, servidores ou laptops. Tal linguagem foi escolhida por atender às necessidades do projeto, dentre as quais podemos destacar: benefícios de uma linguagem orientada a objetos (*e.g.* modularidade, extensibilidade, etc.) e sua característica multiplataforma, não vinculando arquiteturas ou sistemas operacionais aos componentes do sistema (servidor e clientes). Deste modo torna-se viável a utilização de cada componente do sistema a partir das mais diversas arquiteturas (desde celulares a super-computadores) e sistemas operacionais.

4.1 Funcionalidades do Sistema

Os requisitos funcionais do sistema foram obtidos por meio de entrevistas com médicos de diversas áreas e com estudantes de medicina vinculados ao Hospital Universitário Oswaldo Cruz, à Faculdade de Ciências Médicas da Universidade de Pernambuco e à Faculdade de Medicina da Universidade Federal de Pernambuco. Com isso o sistema propõe-se a atender as principais necessidades dos mais diferentes perfis de profissionais, tanto o profissional experiente, como também médicos recém formados ou até mesmo auxiliar no processo de ensino.

O sistema está dividido em duas plataformas: a plataforma servidora, que armazenará os dados, a árvore e os demais recursos oferecidos aos usuários e a plataforma cliente, cuja única funcionalidade será fazer as consultas remotas ao sistema decisor.

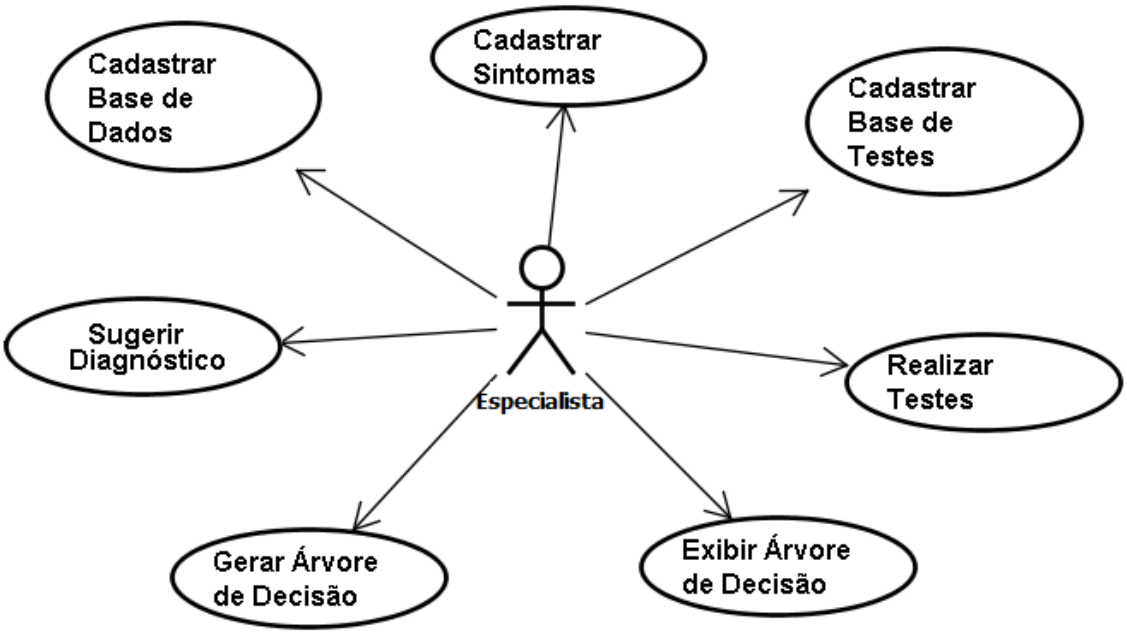

Figura 9. Casos de Uso da Aplicação Servidora

Todas as funcionalidades presentes no aplicativo servidor podem ser visualizadas no diagrama de casos de uso ilustrado na Figura 9. O Agente Especialista representa o médico, o único ator desta plataforma. Ele representa o usuário responsável pela manutenção da informação, por exemplo, um médico ou um estudante de medicina. A este agente decisor são permitidas as seguintes ações:

Cadastrar Base de Dados: Cadastrar a base de dados que será utilizada no treinamento da Árvore de decisão;

Cadastrar Base de Testes: Cadastrar a base que servirá de teste para avaliação da corretude dos diagnósticos do sistema;

Cadastrar Sintomas: Cadastra um novo sintoma manualmente. Ao se cadastrar uma base de dados todos os sintomas contidos nela serão cadastrados automaticamente.

Gerar Árvore de Decisão: Após o cadastro de dados, o sistema utiliza o algoritmo ID3 (conforme discutido no tópico 3.3) para gerar a árvore de decisão.

Realizar Testes: Após a construção da árvore de decisões o sistema pode testar uma base de dados previamente cadastrada a fim de obter informações como taxa de erro.

Sugerir Diagnóstico: O sistema deverá ser capaz de elaborar um diagnóstico coerente com sintomas lhe apresentados de acordo com a base de dados cadastrada.

Exibir Árvore de Decisão: Exibe a topologia da árvore, assim pode-se analisar como a árvore determinou o diagnóstico.

A Figura 10 ilustra o caso de uso do aplicativo cliente. Como é possível de se constatar, este aplicativo também contém apenas um ator. A este ator só é permitida uma única ação: Sugerir diagnóstico. Na aplicação cliente o ator Decisor representa tanto o médico que procura um auxilio no diagnóstico quanto um estudante que por ventura utilize a aplicação como ferramenta pedagógica. Esta ação consiste apenas numa solicitação ao servidor, tornando o aplicativo cliente ágil e leve, sendo portanto capaz de ser utilizado por qualquer dispositivo que interprete Java e que tenha acesso a internet.

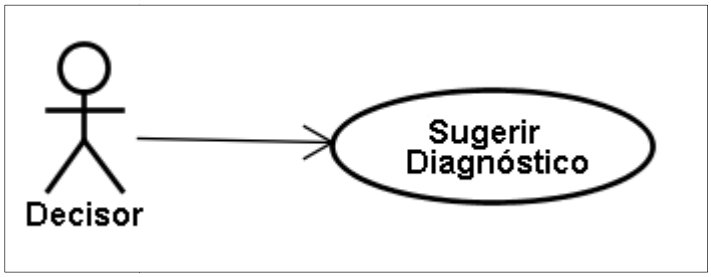

Figura 10. Casos de Uso da Aplicação Cliente

4.2 Arquitetura do Sistema

O sistema está dividido em duas ferramentas, cada uma destas ferramentas está dividida em módulos (ou pacotes). Com o emprego destes módulos qualquer alteração em determinada parte do código não influi no restante do sistema, desde que as interfaces de comunicação entre os módulos permaneçam inalteradas. O servidor e o cliente têm diferentes propósitos, portanto contêm diferentes módulos.

Investigando os mecanismos de cada aplicação pode-se perceber que uma característica dos módulos pode ser entendida como agrupar classes de acordo com sua funcionalidade e com o que tal classe representa para o sistema. Será descrita mais detalhadamente a arquitetura de cada aplicação.

4.2.1 A Aplicação Servidora

Conforme pode ser observado na Figura 11, a aplicação servidora está dividida em seis pacotes: `es`, `dados`, `arvore`, `comunicação`, `fachada` e `gui`. O pacote fachada contém apenas uma classe: a classe `Fachada`. A função desta classe é servir de interface entre a `gui`, as classes que implementam a lógica do sistema e o módulo de comunicação com o cliente. Para manter o domínio da lógica e dos dados que envolvem o sistema, a classe `Fachada` contém objetos que representam cada um dos pacotes manipulados durante o processamento.

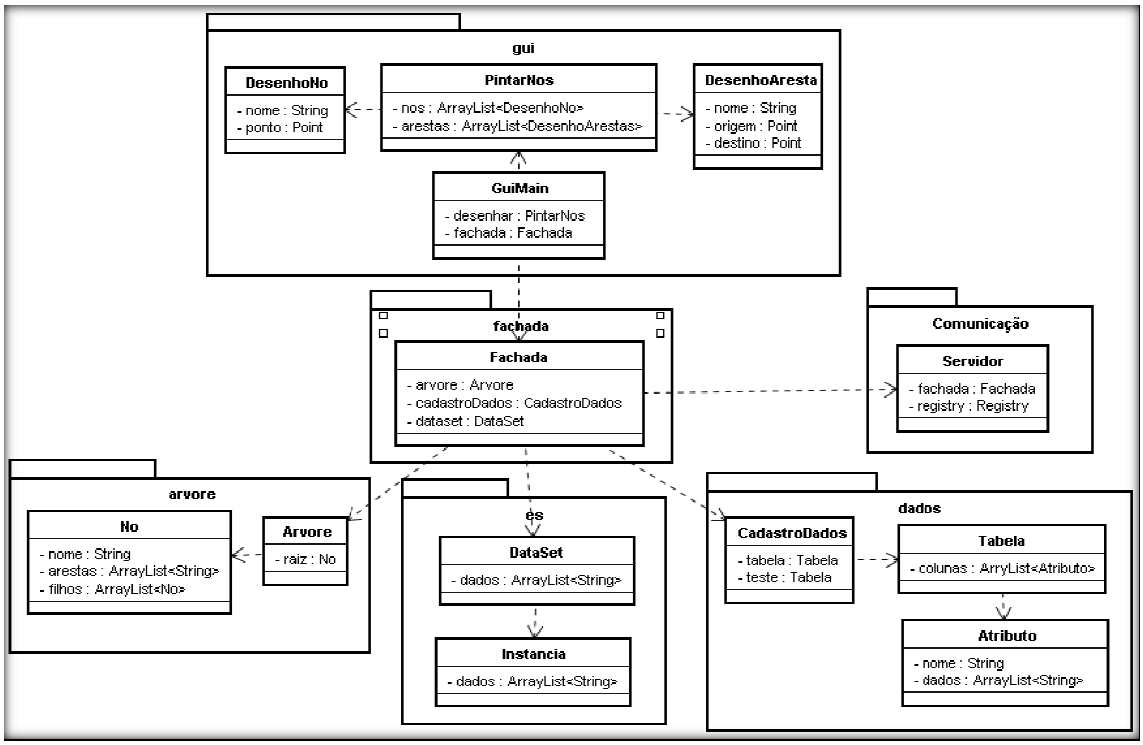

Figura 11. Diagrama de Classes da Aplicação Servidora

O pacote `es` é responsável pela manipulação de arquivos dentro do sistema. Utilizando objetos `DataSet` ele é capaz de ler e escrever em arquivos e, com auxilio de um objeto `Instancia`, é capaz de analisar um dado arquivo

e dele extrair as informações, como tabelas incluídas, atributos, número de classes, etc. Para o correto funcionamento do sistema, foi definida uma organização especifica para os arquivos que guardam as bases de dados a serem utilizadas como entrada do sistema. Este formato de arquivo tanto é utilizado para as bases de treinamento quanto para as bases de teste. O formato definido é o mais simples possível: na primeira linha têm-se os nomes dos atributos em questão (divididos por ponto-e-vírgula) e nas linhas seguintes os valores correspondentes a cada amostra (também divididos por ponto-e-vírgula). A especificação de cada atributo (como a declaração de possíveis valores e o total número de classes) não é feita de maneira explícita, a própria aplicação extrai esses dados automaticamente do arquivo. Um exemplo do formato dos arquivos pode ser visto na Figura 12 mostrada a seguir:

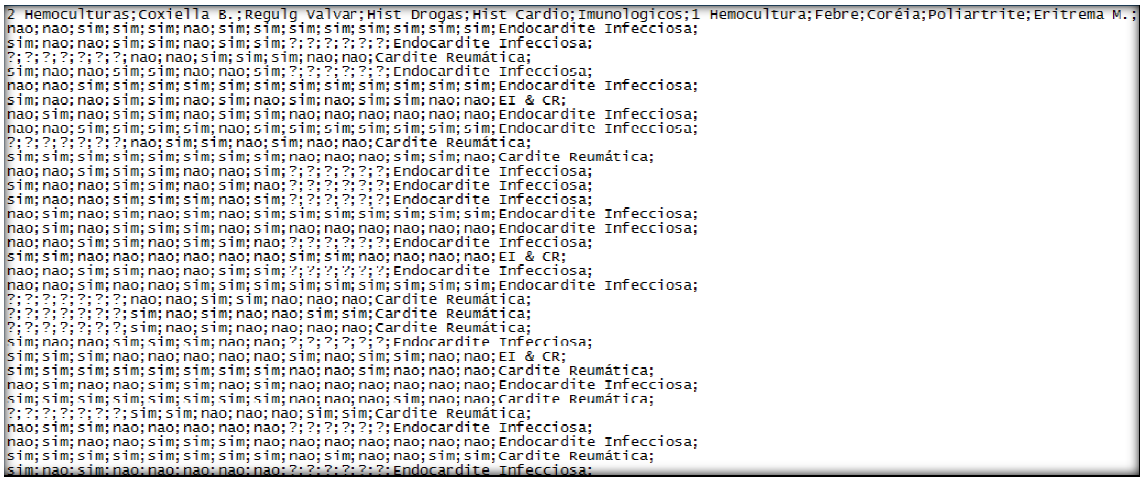

Figura 12. Exemplo de Arquivo Contendo base de Dados

Assim que um arquivo é cadastrado na `Fachada` ele é analisado e, a partir das informações colhidas são criados tantos objetos `Atributo` quanto necessários e estes serão incluídos numa instancia da classe `Tabela`. A partir daí pode-se dar inicio ao treino de uma árvore de decisão. O diagrama de seqüência ilustrado pela Figura 13 representa a rotina para criação da árvore. Após esta rotina ser concluída o usuário tem a possibilidade de fazer consultas a base, sugerir diagnósticos a partir de sintomas e visualizar a topologia lógica da árvore.

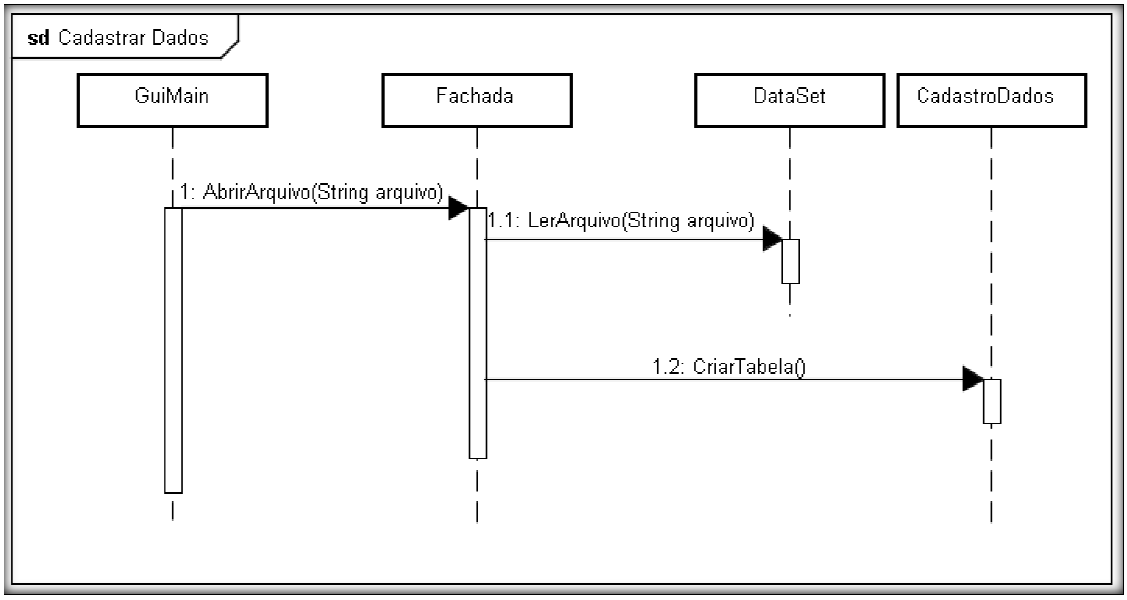

Figura 13. Diagrama de Sequência Cadastrar Dados

O pacote de dados armazena e manipula toda a informação bruta do sistema. Assim que um arquivo é interpretado pela classe `DataSet` é criada uma instância da classe `CadastroDados` dentro do objeto `Fachada`. Na `Tabela`, que é um atributo da classe `CadastroDados`, estarão armazenadas todas as informações que estavam contidas no arquivo. Além de manter as informações armazenadas sistematicamente a tabela manipula os dados nela inseridos a fim de obter informações requisitadas para construção da árvore, como cálculo de entropia, geração de sub-tabelas, etc.

O módulo de comunicação contém a classe `Servidor`, esta classe é responsável por atender às solicitações dos clientes remotos. Para isso ele utiliza o conjunto de protocolos Java RMI [24] para manter a comunicação com seus clientes. Em suma o servidor recebe um conjunto de sintomas e, depois de consultar a fachada, retorna o diagnóstico elaborado.

No pacote `arvore` estão contidas apenas duas classes: `Arvore` e `No`. A classe `No` implementa cada nó da árvore. Cada nó tem um nome, uma lista de filhos e uma lista de arestas (nas quais os filhos estão vinculados). A classe `Arvore` contem apenas um atributo, o atributo raiz. Esta classe serve basicamente para construir a árvore de decisão e manter guardado o nó raiz da árvore.

E, finalmente, tem-se o módulo `gui`. Nele está implementada a interface gráfica. A interface prima por manter um ambiente simples e de fácil acesso as funcionalidades do sistema, conforme pode ser vislumbrado na Figura 14.

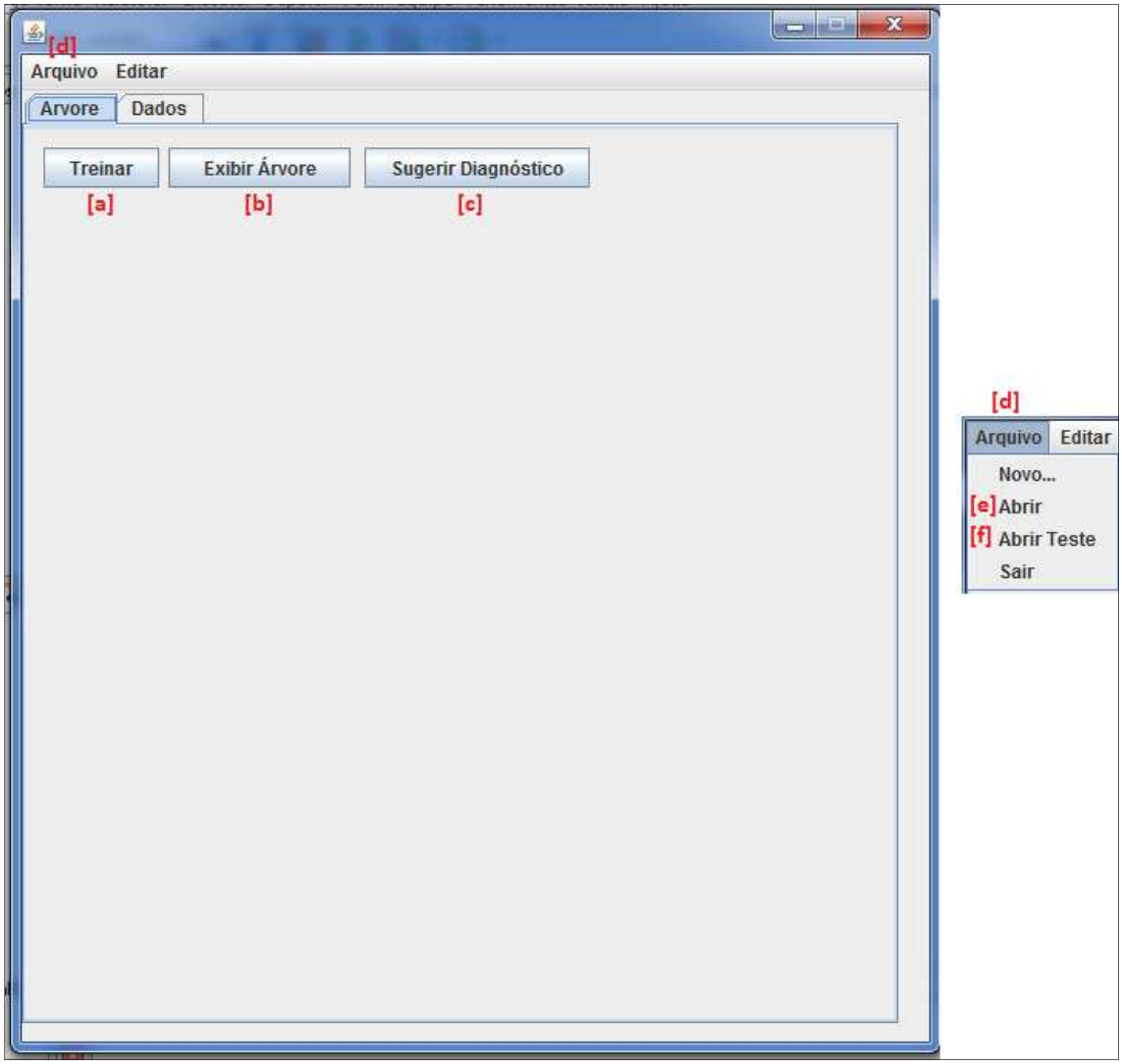

Figura 14. Tela Principal da Aplicação Servidora

Todas as funcionalidades citadas na seção anterior podem ser acessadas a partir dos menus ou botões presentes na tela principal da Aplicação. Ao clicar no botão (d) a o menu “arquivo” será aberto e serão dispostas as opções para cadastrar uma nova base de dados em “abrir” (e) e cadastrar uma nova base de teste em “abrir testes” (f). Após inserir os dados necessários, o usuário terá a opção de treinar o SADC no botão “treinar” (a), exibir a árvore criada pelo sistema em (b) ou (c) para sugerir diagnóstico para uma doença a partir dos sintomas apresentados.

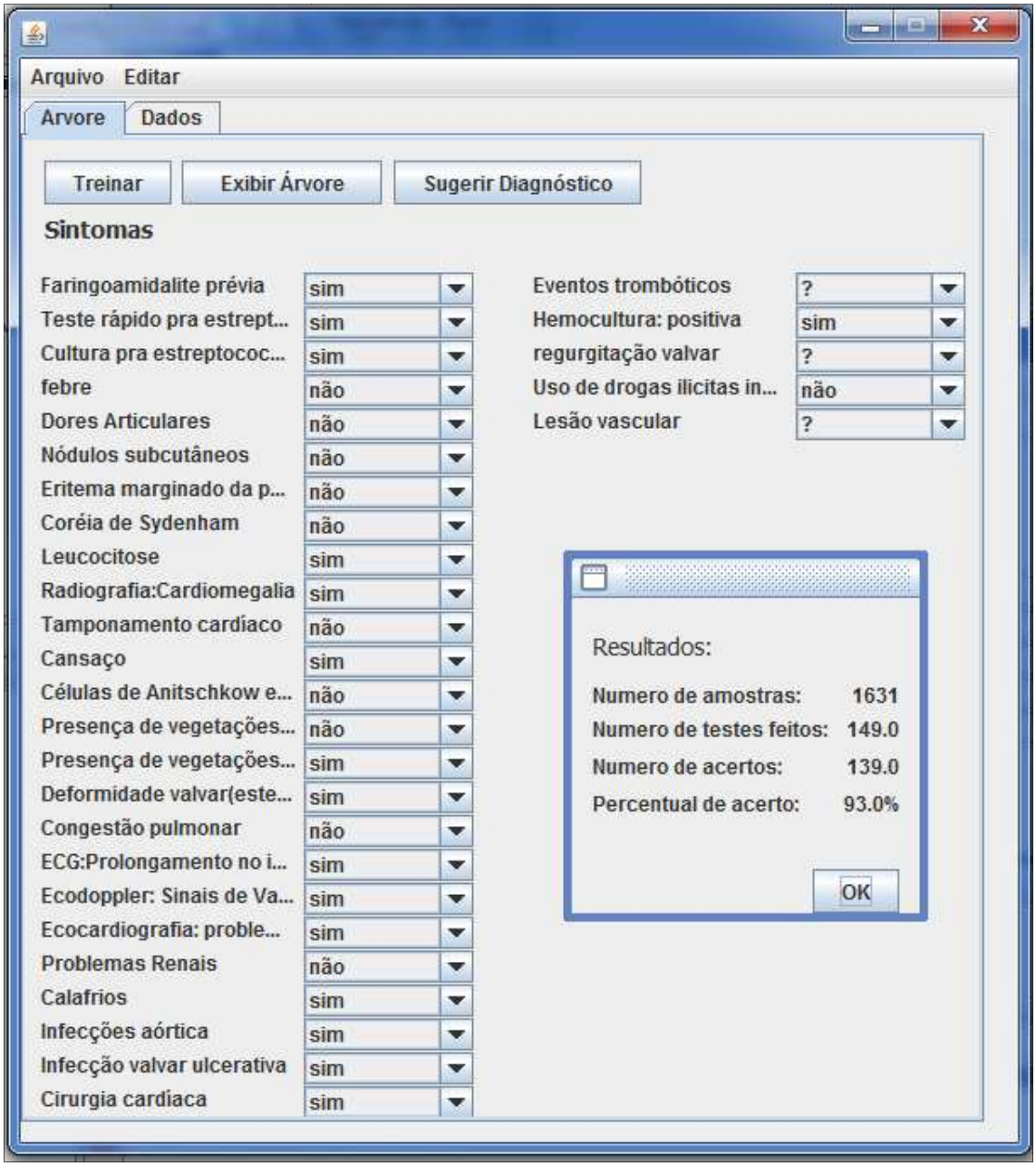

Figura 15. Tela da Aplicação Servidora após Treinamento

A Figura 15 ilustra a interface logo após o treinamento. A partir da base de dados e da base de testes previamente cadastradas a aplicação calcula o percentual de acerto e exibe para o usuário. Depois de ter sua árvore treinada a aplicação lista os sintomas cadastrados e habilita o usuário para que o mesmo possa descrever quais os sintomas são apresentados pelo paciente e, eventualmente, como eles são apresentados. Assim que os sinais e sintomas

forem inseridos o usuário tem a possibilidade de predizer qual a enfermidade o paciente apresenta.

4.2.2 A Aplicação Cliente

Um dos principais cuidados tomados durante a fase de concepção do sistema foi conceber uma aplicação cliente simples, leve e portável. Para que estas características fossem alcançadas optou-se por manter toda a complexidade do sistema no servidor. Além de manter a aplicação cliente leve, computacionalmente falando, a decisão de manter a complexidade no centro do sistema (no servidor) mantém a consistência do modelo proposto. Isto é, sempre os padrões de diagnósticos serão os mesmos, já que estarão baseados na árvore de decisão contida na aplicação servidora. O diagrama de classes da aplicação cliente é elucidado na Figura 16.

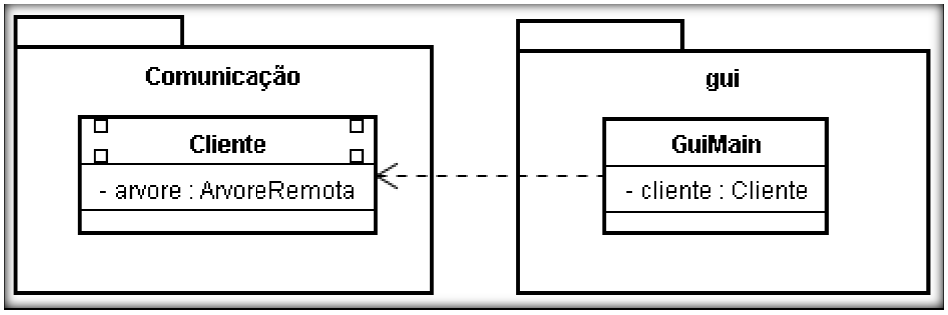

Figura 16. Diagrama de Classes da Aplicação Cliente

Como se pode notar, a aplicação contém apenas dois módulos: `comunicação` e `gui`. Dentro do módulo `gui` encontra-se `GuiMain`, a interface gráfica do aplicativo. Conseqüentemente, não há treinamento (i.e. adaptação na classe Cliente); ela somente é capaz de obter sugestões de diagnósticos. Esta interface repassa as ações efetuadas pelo usuário decisor para a uma instância da classe `Cliente`, localizada no módulo de comunicação. A partir daí o objeto `Cliente` fará requisições ao servidor. Assim que suas solicitações forem atendidas e respondidas o cliente repassa as informações adquiridas à `GuiMain`.

4.3 Testes e Validação do Sistema

Para aferir a corretude e confiabilidade da árvore de decisão inclusa no sistema desenvolvido foram elaborados diversos testes de comportamento. Para a execução dos testes foram utilizadas bases de dados simbólicas extraídas do repositório da UCI [25], ver detalhes no Anexo A.

Para testar os resultados obtidos pelo sistema, a partir da árvore de decisão treinada, foi utilizada a ferramenta WEKA [29] como base para comparações. A opção por escolher tal ferramenta para comparações deu-se devido ao fato desta ser largamente aceita e usada pela comunidade científica, bem como por ser de utilização livre. Foi constatado que a topologia da árvore de decisão criada por ambas as ferramentas era similar, bem como as taxas de erro atribuída às suas predições. Desta forma foi possível comprovar a corretude do algoritmo implementado, isto é, que o mesmo condiz com a proposta de Quinlan em [14].

Após ser verificado que o funcionamento da AD estava de acordo com o esperado surge a necessidade de validação do sistema. Por validar entenda-se mensurar a capacidade de classificação do sistema. Nesta etapa foram usados dois tipos de base de dados, (1) uma base de dados extraída do repositório da UCI, utilizada como *benchmark* em diversos trabalhos relacionados a computação inteligente, e (2) uma base de dados gerada artificialmente a partir de dados extraídos de Robbins, 2005 [26].

O propósito de cada uma das bases seria validar o sistema em determinados aspectos. A primeira base deveria verificar a confiabilidade do sistema e a segunda verificaria a capacidade do mesmo de realizar diagnósticos diferenciais.

Para esta validação foram usados dois métodos de avaliação: o percentual de erro e a curva ROC (*Receiver Operating Characteristic*) gerada [27]. O percentual de erro trata-se simplesmente da taxa obtida a partir do quociente entre os erros de predição do sistema e o número total de testes realizados.

Já a análise da curva ROC tem sido utilizada em medicina, radiologia, psicologia e outras áreas por muitas décadas e, mais recentemente, foi introduzida a áreas como aprendizado de máquina e mineração de dados. Ela se baseia em duas medidas: a sensibilidade e a especificidade. A sensibilidade representa a proporção de verdadeiros positivos, e a especificidade representa a proporção de verdadeiros negativos. Na Tabela 1 pode-se ver mais claramente a que se referem estas medidas.

Tabela 1. Tabela de Contingência (Matriz de Confusão)

		Valor Verdadeiro (confirmado por análise)	
		positivos	negativos
Valor Previsto (predito pelo teste)	positivos	**VP** Verdadeiro Positivo	**FP** Falso Positivo
	negativos	**FN** Falso Negativo	**VN** Verdadeiro Negativo

Os resultados obtidos através dos testes de validação foram bastante satisfatórios. Na base de dados `car.data`, extraída do repositório da UCI e aqui utilizada como *benchmark*, o sistema conseguiu classificar os dados com uma taxa de acertos bastante elevada. O percentual de erro foi de 6%. Na base criada artificialmente, a `cardites.data`, o sistema também obteve um bom desempenho, conseguindo taxas de acerto superiores a 87%. Vale salientar que o intuito do teste utilizando a base de dados artificial foi conseguir elaborar um diagnóstico diferencial correto, o que, por si só, já é se mostra como um imenso desafio.

Na Figura 17 está ilustrada a curva ROC de cada uma das duas validações. A análise da curva geralmente é feita a partir do cálculo da área inserida entre a curva em si e o eixo horizontal. Um classificador perfeito corresponderia a uma linha horizontal no topo do gráfico, porém esta dificilmente será alcançada.

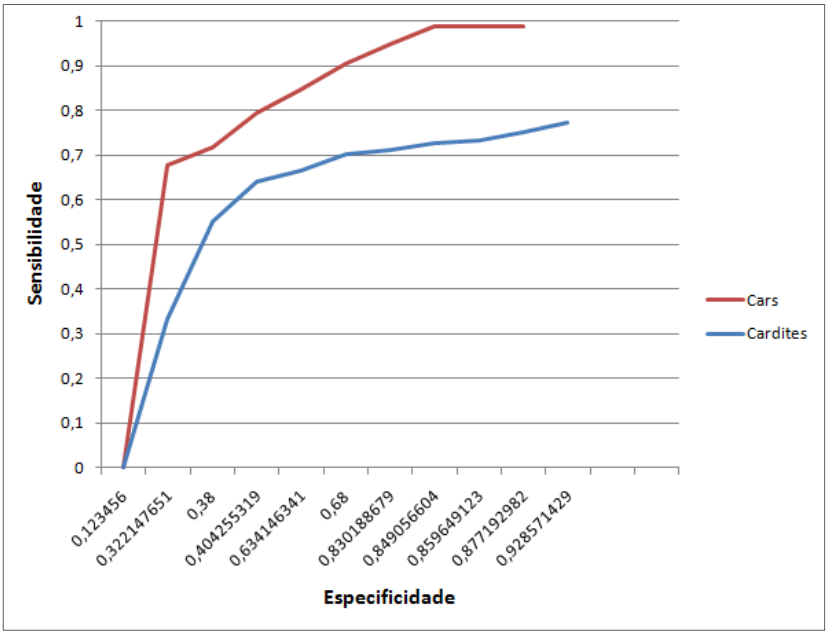

Figura 17. Curva ROC

Na prática, curvas consideradas boas estarão entre a linha diagonal e a linha perfeita, onde quanto maior a distância da linha diagonal, melhor o sistema. A linha diagonal indica uma classificação aleatória. Como pode ser percebido, ambos os testes apresentaram um bom desempenho.

Capítulo 5

Conclusão

Devido à enorme quantidade de elementos informativos que um médico necessita para prática da medicina, sistemas de informação computacionais se apresentam como uma poderosa ferramenta no manuseio de todo esse volume de informações e também para o processamento das incertezas médicas associadas. Dentre as atividades realizadas pelos médicos, o diagnóstico, caracteriza-se como uma das mais importantes etapas. A partir dele será fundamentado todo o tratamento visando à restauração da saúde de um paciente.

A proposta deste trabalho foi o desenvolvimento de um sistema computacional que auxiliasse o profissional da saúde na elaboração de diagnósticos diferenciais. Para atingir esta meta foi realizada uma revisão bibliográfica tanto na área médica, abordando assuntos: semiologia, medicina baseada em evidências e heurísticas para formulação de diagnósticos, quanto na área computacional, abordando temas como sistemas de informação, sistemas distribuídos e inteligência artificial.

Desta forma foram desenvolvidas duas ferramentas para dar suporte ao profissional clínico: uma aplicação servidora, cujas atribuições são guardar os dados de forma sistematizada e deles extrair informações, e uma outra aplicação cliente, cujo principal objetivo é fazer consultas remotas à primeira aplicação citada.

Como principais resultados obtidos podem-se ser destacados a utilidade da ferramenta desenvolvida junto aos profissionais de saúde, a possibilidade de realizar a mineração dos dados da aplicação servidora e a conveniência trazida pela aplicação ao ser usada como ferramenta de apoio pedagógico por estudantes de medicina.

.

Ao final deste trabalho, algumas melhorias e trabalhos futuros podem ser apontados em relação à prova de conceito desenvolvida. São eles:

Integração a um SIGH

Integração do SDAC desenvolvido a um amplo SIGH. Assim a sistematização da informação, já incluída nos SIGHs, poderia se dar de forma ainda mais racional. Essa proposta poderia oferecer mais efetivamente subsídios para mineração de dados em ambientes específicos (*i.e.* determinada cidade ou região, além de grupos específicos de indivíduos, por exemplo). Assim os eventuais treinamentos das árvores de decisão poderiam ser mais rápidos e o conseqüente processamento, mais eficiente.

Com algumas funcionalidades do SIGH o novo sistema poderia se tornar uma interessante ferramenta a ser usada a favor da saúde pública, especialmente na geração de diagnósticos contextuais, que facilmente seriam utilizados dada a organização sistemática dos dados.

Investigar outros modelos de Construção de AD

O ID3, algoritmo núcleo do módulo inteligente do sistema é comprovadamente um ótimo classificador, mas apresenta algumas limitações. A principal delas é a capacidade de trabalhar apenas com valores simbólicos, isto é, nominais. Em muitos casos as entradas da árvore (os sintomas) têm de passar por tratamento para serem discretizados, perdendo assim uma significativa quantidade de informação neste processo.

Também seria interessante a investigação de outros aspectos da técnica como algoritmos de poda, novos critérios de seleção de atributo, novos critérios de parada de treinamento e de determinação das classes associadas às folhas.

Integração a um Sistema Especialista

A integração a um Sistema Especialista (SE) traria novas funcionalidades à ferramenta proposta. Com um segundo módulo inteligente representado pelo SE o novo sistema híbrido poderia, além de auxiliar a elaboração do

diagnóstico do paciente, apresentar deduções para tratamento de cada determinado caso.

Integração a outros Sistemas Inteligentes

Como foi apresentado, a ferramenta proposta elabora diagnósticos plausíveis a cerca da enfermidade que aflige a paciente baseada em informações passadas pelo clínico. Uma interessante funcionalidade a ser acrescentada seria a utilização de outras técnicas de computação inteligente para analisar resultados brutos de exames.

O uso de redes neurais, por exemplo, em tomografias poderia gerar atributos a serem inseridos nas árvores de decisão. Assim como as redes neurais, técnicas como redes bayesianas, lógica difusa e algoritmos genéticos poderiam ser alternativas interessantes para tais mecanismos de indução em eventuais futuras expansões do sistema.

Bibliografia

[1] J. Ducla Soares, Semiologia Médica – Princípios, Métodos e Interpretação, 2007.

[2] D.L. KASPER, A.S. FAUCI, D.L. LONGO, E. BRAUNWALD, S.L. HAUSER, and J.L. JAMESON, Tomadas de decisões em medicina clínica. Harrison: Medicina Interna., McGraw-Hill, 2002.

[3] L. Braun, F. Wiesman, H. Herik, A. Hasman, and E. Korstein, "Towards patient-related information needs," Journal of Medical Informatics, 2006.

[4] J.B. Vasconcelos, R. Henriques, andÁ. Rocha, Modelo para o desenvolvimento de Sistemas de Apoio à Decisão Clínica para a prática da Medicina Baseada na Evidência, Porto, 2003

[5] L.M. Tonetto, L.L. Kalil, W.V. Melo, D. Di, G. Schneider, and L.M. Stein, "O papel das heurísticas no julgamento e na tomada de decisão sob incerteza," vol. 23, 2006, pp. 181-190.

[6] E.A. Zoby, Sistema de Apoio à Decisão para o Diagnóstico Médico de Doenças Sexualmente Transmissíveis (SADM - DST), Recife: 2006.

[7] Harrison, Medicina Interna, Rio de Janeiro: McGraw-Hill, 2002.

[8] D.M. Eddy, "Anatomy of a Decision," JAMA, 1990, pp. 441-443.

[9] K.C. Laudon and J.P. Laudon, Sistemas de Informação Gerenciais, São Paulo: Pearson Prentice Hall, 2004.

[10] E. De Andrade Zoby, Sistema de Apoio a Decisão para o Diagnóstico Médico de Doenças Sexualmente Transmissíveis, Trabalho de Conclusão de Curso (Engenharia da Computação) Escola Politécnica de Pernambuco, UPE, Recife: 2006.

[11] Http://lcs.mgh.harvard.edu/projects/dxplain.html, "MGH Laboratory of Computer Science - projects - dxplain." , último acesso em 25 de novembro de 2009

[12] Http://www.openclinical.org/aisp_qmr.html, "OpenClinical: QMR", último acesso em 25 de novembro de 2009

[13] Http://www.openclinical.org/aisinpracticeDSS.html, "OpenClinical: AI Systems in Clinical Practice: Decision support systems." ,último acesso em 25 de novembro de 2009

[14] http://www.medline.com, último acesso em 25 de novembro de 2009

[15] http://www.ncbi.nlm.nih.gov, " National Center of Biotechnology Information", último acesso em 25 de novembro de 2009

[16] P. RUSSEL, S.; NORVIG, Artificial Inteligence: A Modern Approach. Upper Saddle River., 1995.

[17] J. Quinlan, "Induction of Decision Trees," Machine Learning, vol. 1, 1986, pp. 81 - 106.

[18] J. Quinlan, C4.5 : Programs for Machine Learning, San Mateo: Morgan Kaufmann, 1993.

[19] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, Classification and Regression Trees, Belmont: Wadsworth, 1984.

[20] J. Pearl, Entropy, Information and Rational Decisions, Los Angeles: 1978.

[21] V.S. Tanenbaum A.S., "Distributed Systems: Principles and Paradigms", Prentice-Hall, 2002

[22] C.G. Dollimore J., Kindberg T., Distributed Systems: Concepts and Design, Wesley, Addison, 2005.

[23] P. A.Bernstein, "Middleware: A Model for Distributed System Services," Communications of the ACM, vol. 39, 1996, pp. 87-98.

[24] W. Emmerich, "Software Engineering and Middleware: A Roadmap," Second International Workshop on Software Engineering and Middleware, Limerick: 2000, pp. 119-129.

[25] E.G. Calábria, "Hermes - Um Middleware Orientado a Mensagem para Ambientes Corporativos," Dissertação (Mestrado em Ciências da Computação) – Centro de Informática, UFPE, Recife, 2004.

[26] Http://java.sun.com, "Developer Resources for Java Technology," SUN CORPORATION, último acesso em 25 de novembro de 2009

[27] H.M. Deitel and P.J. Deitel, Java How to Program, Prentice Hall, 2000.

[28] Http://archive.ics.uci.edu/ml/datasets.html, "UCI Machine Learning Repository: Data Sets," 2009", último acesso em 25 de novembro de 2009

[29] I.H. Frank, Data Mining: Practical Machine Learning Tools and Techniques, San Francisco: Morgan Kaufmann, 2005.

[30] V. KUMAR, A.K. ABBAS, and N. FAUSTO, Robbins e Cotran: Patologia: Bases Patológicas das Doenças, Elsevier, 2005.

[31] Http://www.anaesthetist.com/mnm/stats/roc/Findex.htm, "ANAESTHESTIST.COM, “Receiver Operating Curves: An Introduction”.", último acesso em 25 de novembro de 2009

Anexo A

A UCI-*Machine Learning* Repository mantém 187 base de dados destinados ao treinamento de algoritmos inteligentes. Dentre estas bases podem ser encontradas bases numéricas, categóricas, para uso em classificações, para uso em regressões, etc. Os dados deste repositório foram escolhidos como parâmetros de teste para o algoritmo implementado por dois motivos principais: (1) a similaridade do formato dos seus arquivos com o formato usado pela aplicação, (2) a notória utilização de alguns destas bases como *benchmarks* na comunidade acadêmica (em destaque nas comunidades de computação inteligente).

Para demonstrar como a aplicação interpreta os dados contidos num arquivo e, ao mesmo tempo, ilustrar como se apresentam as bases de dados disponíveis no repositório da UCI, será dado como exemplo parte da base `car.data`. Todavia, antes de expor a base em si, é interessante estudar sua estrutura. Junto com a base de dados também se encontra disponível no endereço eletrônico da UCI um arquivo contendo as informações a cerca do conteúdo daquela base, no caso o `car.names`, que pode ser visto a seguir:

```
| names file (C4.5 format) for car evaluation domain

| class values

unacc, acc, good, vgood

| attributes

buying:   vhigh, high, med, low.

maint:    vhigh, high, med, low.

doors:    2, 3, 4, 5more.

persons:  2, 4, more.

lug_boot: small, med, big.

safety:   low, med, high.
```

Como pode ser visto, esta base de dados se destina a fazer a avaliação de carros. Estes carros podem ser classificados como `unacc`, que seriam carros com padrões inaceitáveis, `acc`,que seriam carros com as mínimas condições para os tornar aceitáveis, `good`, carros considerados bons e `vgood`, que seriam excelentes. Como atributos estes carros teriam características como preço de venda, custo de manutenção, número de portas, capacidade de passageiros, tamanho da mala e segurança estimada do carro. Com as características da base já em mente torna-se mais fácil a compreensão da mesma. A seguir um trecho da base `car.data`:

```
Buying,Maint,Doors,Persons,Lug_boot,Safety, Evaluation
vhigh,vhigh,2,2,small,low,unacc
vhigh,vhigh,2,2,small,med,unacc
vhigh,vhigh,2,2,small,high,unacc
vhigh,vhigh,2,2,med,low,unacc
vhigh,vhigh,2,2,med,med,unacc
vhigh,vhigh,2,2,med,high,unacc
vhigh,vhigh,2,2,big,low,unacc
vhigh,vhigh,2,2,big,med,unacc
vhigh,vhigh,2,2,big,high,unacc
```

Como pode ser constatado o formato de armazenamento de dados no arquivo em muito se assemelha ao formato utilizado pela aplicação. A principal diferença entre os dois modelos é o uso de vírgula nas bases do repositório da UCI no lugar do ponto-e-vírgula que, como foi visto, é utilizado na aplicação desenvolvida. A disposição dos dados na tabela representa uma tabela onde as colunas são separadas por vírgulas e as linhas separadas por quebras-de-linha. O diagrama a seguir torna mais fácil esta visualização:

Buying,	Maint,	Doors,	Persons,	Lug_boot,	Safety,	Evaluation
vhigh,	vhigh,	2,	2,	small,	low,	unacc
vhigh,	vhigh,	2,	2,	small,	med,	unacc
vhigh,	vhigh,	2,	2,	small,	high,	unacc

Para análise dos dados contidos nas bases de dados a ferramenta se comporta da seguinte maneira: primeiramente lê a primeira linha e a cada *parser* (no caso da aplicação ponto-e-vírgula) interpreta que existe um novo atributo para classificar uma das características do objeto tratado pela tabela. Ao passar pelo último atributo a ferramenta compreende que este será o atributo que funcionará como classificador do objeto tratado pela base. Ao ler as demais linhas à aplicação, além de guardá-las sistematicamente, guarda informações a como os valores que podem ser assumidos, número de vezes que cada valor aparece na tabela, etc.