UNIVERSIDADE FEDERAL DO SUL E SUDESTE DO PARÁ
INSTITUTO DE GEOCIÊNCIA E ENGENHARIAS
FACULDADE DE COMPUTAÇÃO
Bacharelado em Sistemas de Informação

SISTEMA PARA AUXÍLIO NA IDENTIFICAÇÃO DE SÍFILIS UTILIZANDO MINERAÇÃO DE DADOS, MODELAGEM 3D E PROCESSAMENTO DE VOZ

Janailda Bezerra da Silva
Márcia da Silva Souza
Paulo Roberto Cardoso Sousa

MARABÁ

2013

Janailda Bezerra da Silva
Márcia da Silva Souza
Paulo Roberto Cardoso Sousa

SISTEMA PARA AUXÍLIO NA IDENTIFICAÇÃO DE SÍFILIS UTILIZANDO MINERAÇÃO DE DADOS, MODELAGEM 3D E PROCESSAMENTO DE VOZ

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador:

Prof. Msc. Josué Leal Moura Dantas

MARABÁ

2013

Janailda Bezerra da Silva
Márcia da Silva Souza
Paulo Roberto Cardoso Sousa

SISTEMA PARA AUXÍLIO NA IDENTIFICAÇÃO DE SÍFILIS UTILIZANDO MINERAÇÃO DE DADOS, MODELAGEM 3D E PROCESSAMENTO DE VOZ

Trabalho de Conclusão de Curso, apresentado à Universidade Federal do Sul e Sudeste do Pará, como parte dos requisitos necessários para obtenção do Título de Bacharel em Sistemas de Informação.

Marabá 19 de Dezembro de 2013.

Conceito: Excelente.

_____

Prof. Msc. Josué Leal Moura Dantas
Faculdade de Computação/UFPA - Orientador

_____

Profª. Msc. Danielle Costa Carrara Couto
Faculdade de Computação/UFPA - Membro

_____

Prof. Esp. Gleison de Oliveira Medeiros
Faculdade de Computação/UFPA – Membro

MARABÁ

2013

DEDICATÓRIA

Dedico este trabalho a minha mãe Maria de Fátima Bezerra Silva, a meu irmão Jundeilton Bezerra Silva e a meu namorado Maciel Brito Silva.

Janailda Bezerra da Silva

Dedico este trabalho aos meus pais, Joaquim Jovita de Souza e Maria Joaquina da Silva Souza, a meus irmãos, e ao meu namorado Bruno Alves de Souza.

Márcia da Silva Souza

Dedico este trabalho aos meus pais, Gerson Nilton e Joanice Cardoso.

Paulo Roberto Cardoso Sousa

AGRADECIMENTOS

Agradeço primeiramente a Deus, por acreditar que nossa existência pressupõe outra infinitamente superior. A minha mãe Maria, a Meus irmãos Jundeilton, Gizerlidia, Jundeilda e Judinira, principalmente ao meu irmão Jundeilton pelo apoio e dedicação especial que foram fundamentais para que esse sonho começasse a ser possível, também a minha sobrinha Amanda Evangelista, e a toda minha família que com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa de minha vida. Ao meu namorado Maciel Brito por acrescentar razão e beleza aos meus dias. A meu professor orientador, Prof. Josué Leal, pelo auxílio, disponibilidade de tempo, sempre com uma simpatia contagiante e pelo fornecimento de material para pesquisa do tema, ao professor e coordenador do curso, pelo convívio, pelo apoio, pela compreensão e pela amizade. A todos os professores do curso, que foram tão importantes na minha vida acadêmica e no desenvolvimento deste TCC. Aos amigos e colegas, pelo incentivo e pelo apoio constantes. Em nome de Geiziane Silva, Edinei Lustosa, Aline Santana, Frederico Vince, Maria Eliane Sobrinho, Janilson Ribeiro, Jéssica Alves, Diego Alcântara, Leonildes Pontes, Aldemir Leal, Príscila Alves. À banca de avaliadores, que muito prontamente e com grande entusiasmo aceitaram fazer parte desse momento tão importante em minha vida. Aos alunos de minha classe nesta Faculdade que de tanto me perguntarem me fizeram pensar e, pensando, aprendi cada vez mais a buscar as respostas para satisfazer-lhes o interesse e me aperfeiçoar na matéria, com a humildade dos que aspiram à sabedoria repetindo a frase filosófica. “Só sei que nada sei”. E por último, e não menos importante obrigado a meus colegas e amigos de projeto, Márcia Souza e Paulo Roberto Cardoso.

Janailda Bezerra da Silva

AGRADECIMENTOS

Agradeço em primeiro lugar a quem nunca desistiu de mim, mesmo quando eu pensava que não iria conseguir Ele insistia em dizer que eu iria alcançar meus ideais, bastava eu confiar e ter fé Nele. Esta pessoa é o meu Deus a quem sou muito grata pela fortaleza que és em todas as situações postas em minha vida. Por segundo lugar, agradeço aos meus pais Joaquim Jovita de Souza e Maria Joaquina da Silva Souza, pelo incentivo, palavras motivadoras, pelo amor e confiança depositada em mim. Esta é uma forma de retribuir um pouco de tudo que eles fizeram e fazem por mim. Agradeço aos meus irmãos: Francisco Márcio, Marcos José, Mônica Souza e Magno Souza, pelo apoio e motivação. Agradeço ao meu namorado Bruno Alves por me dar inspiração e tornar meus dias mais felizes. Aos meus amigos e colegas da Igreja Sara Nossa Terra, a pastora Selma Barbosa e ao pastor Clerisnaldo juntamente com sua esposa, pastora Léia, por orarem ao meu favor e estarem ao meu lado. Agradeço aos meus colegas e amigos de turma de Sistemas 2009, em especial: Edinei Lustosa, Aldemir Leal, Aline Oliveira, Jéssica Alves, Janilson Ribeiro, Mª Eliane Sobrinho, Lidiane Fernandes, Príscila Alves, Leonildes Pontes e aos meus amigos de projeto Janailda Silva e Paulo Roberto. Agradeço também a todos os professores que contribuíram com minha graduação em especial ao Prof. Msc. Josué Leal Moura Dantas, por sua dedicação e orientação deste projeto. Por fim agradeço a todas as pessoas que acreditaram no meu potencial.

Márcia da Silva Souza

AGRADECIMENTOS

Agradeço primeiramente a Deus por estar sempre ao meu lado durante toda essa jornada e por sempre me mostrar os caminhos corretos, me abençoando e edificando. Em segundo lugar, agradeço aos meus pais Gerson Nilton e Joanice Cardoso pelo apoio durante essa batalha e por me mostrarem que, por mais que os obstáculos fossem difíceis, sempre poderia contar com apoio motivacional e financeiro deles. Agradeço, também, a minha avó Maria Auta Cardoso, a matriarca da minha família a qual dedico esta graduação, pois ela significa o início desta vitória e todas as vitórias que ainda virão. Agradeço as minhas tias pelo incentivo, pelos elogios, pelas belas palavras que sempre me motivaram, são elas: Cristiane Cardoso, Eliane Cardoso, Elizangêla Cardoso, Geusa Cardoso e Rosangela Cardoso. Agradeço aos meus irmãos, Lucas e Giovanna Cardoso, agradeço ao meu cachorro, "Bartolomeu", por sempre ficar no meu pé quando estava redigindo este trabalho. Agradeço, também, a todos os meus AMIGOS pelo companheirismo, por me ajudarem a atravessar este caminho, em especial agradeço a estes "irmãos": Janailda Silva, Márcia Souza, Ilva Santos, Janilson Ribeiro, Jéssica Alves, Aline Oliveira, Príscila Alves, Edinei Lustosa, Lidiane Fernandes, Leonildes Pontes, Leonardo May, Marianne Passos, Hienne Gomes, Camila Nobre e Verônica Danielle por me ajudar a concluir a formatação deste trabalho. Muito obrigado a cada um de vocês, vocês fazem parte da minha família, fazem parte da minha vida para sempre.

Paulo Roberto Cardoso Sousa

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

ABREVIATURAS E SIGLAS

3D	Três Dimensões
ADM	Administrador
API	Application Programming Interface (ou Interface de Programação de Aplicativos)
ASR	Automatic Speech Recognition (ou reconhecimento automático da fala)
CFG	Context free Grammar (ou Gramática Livre de Contexto)
CLR	Common Language Runtime
CREMESP	Conselho Regional de Medicina do Estado de São Paulo
CTA	Centro de Testagem e Aconselhamento
DLL	Dynamic-link library (ou Biblioteca de Ligação Dinâmica)
DST	Doença Sexualmente Transmissível
HMM	Hospital Municipal de Marabá
HTK	The Hidden Markov Model
IBM	International Business Machines
ISSO	International Organização for Standardization
J2SE	Java 2 Platform, Standard Edition
JNI	Java Native Interface
JVM	Java Virtual Machine
KDD	Knowledge Discovery in Databases (ou Descoberta de Conhecimento de Dados)
MBR	Raciocínio Baseado em Casos
OLE	Object Linkingand Embedding
RAF	Reconhecimento Automático de Fala
RAV	Reconhecimento Automático de voz
SAPI	Speech API Speech Application Programming Interface
SGBD	Sistema de Gerenciamento de banco de Dados
SUS	Sistema Único de Saúde
TTS	Text To Speech (ou fala em texto)
UML	Unified Modeling Language (ou Linguagem de Modelagem Unificada)
XML	Xtensible Markup Language

RESUMO

A utilização do reconhecimento de voz vem se difundindo há muito tempo no meio tecnológico, onde diversas pessoas e até mesmo empresas, utilizam esta técnica como forma de autenticação e segurança. O desafio de utilizar técnicas de mineração de dados sendo controlados por reconhecimento de voz trouxe a ideia de associar um agente 3D, tendo em vista uma melhor interação do usuário com o sistema. Visando contribuir e ampliar as técnicas e estudos de mineração de dados integrando às técnicas de reconhecimento de voz, o presente trabalho aborda sobre um sistema de auxilio ao diagnóstico de sífilis, utilizando um agente 3D como forma de interação entre o usuário e o sistema, onde o campo de pesquisa utilizado para o desenvolvimento do programa foi à cidade de Marabá PA. Durante a fase de elaboração, foram feitos levantamentos e pesquisas bibliográficas a respeito da sífilis, doença que utilizamos como objeto de estudo e análises, buscando maneiras de como incorporar e classificar os sintomas desta doença na base de dados do sistema. Para tanto, foi empregado, no estado da arte, os principais conceitos, aplicações, metodologias e ferramentas para o desenvolvimento deste projeto, bem como: técnicas de árvore de decisão, Coruja JLAPSAP, Java ALICE 3D e FreeTTS.

Palavras Chave: Mineração de Dados; Reconhecimento de Voz; Agente 3D; Sífilis.

ABSTRACT

The use of voice recognition has been a long time in spreading technological environment where diverse people and even companies use this technique as a form of authentication and security. The challenge of using data mining techniques being controlled by voice recognition brought the idea of ease of control of this method, which in this work will be associated with a 3D agent. To contribute and expand the technical studies and data mining techniques to integrate voice recognition, this paper discusses on a system of aid to the diagnosis of syphilis, using a 3D agent as a means of interaction between the user and the system, where there search field used for the development of the program was Maraba PA city. During preparation, surveys and literature searches were made regarding syphilis, a disease that used as an object of study and analysis, seeking ways on how to incorporate and classify the symptoms of this disease in the data base system. To do so, was employed in the art, key concepts, applications, methodologies and tools for the development of this project, as well as information systems, software engineering, programming, design analysis, among others.

Keywords: Data Mining; Voice Recognition; 3D Agent; Syphilis.

1. INTRODUÇÃO

Na área médica, o uso de ferramentas tecnológicas que auxiliam no processo de tomada de decisão para detectar doenças, pode se tornar um fator decisivo, que possibilita uma facilidade ao diagnóstico, reduzindo os riscos e custos aos pacientes e melhorando a eficácia dos resultados. As técnicas de Mineração de Dados (*Data Mining*) possibilitam a seleção de dados utilizados no processo de tomada de decisão.

A mineração de dados se caracteriza pela interação dinâmica entre milhares de dados, tornando possível o processo de tomada de decisão mais prático e descomplicado. Dentre as principais técnicas de mineração de dados, existem aquelas que são mais adequadas para descobrir um possível diagnóstico médico, para tanto, essas técnicas são: árvores de decisão e regras de classificação, as quais utilizam regras SE-ENTÃO para gerar e qualificar um futuro resultado (VILLE, 2006). Associar essas técnicas de mineração de dados para auxiliar no processo de tomada de decisão, juntamente com técnicas de reconhecimento de voz e modelagem 3D será o principal diferencial da proposta desse trabalho.

A mineração de dados é essencial para auxiliar no processo de solução de problemas de dados magnéticos produzidos e armazenados em grande escala, que são inviáveis de serem lidos e analisados por métodos tradicionais, como por exemplo, planilha de dados (FAYAAD et al., 1996). As informações contidas nos dados não são caracterizadas de forma explicita, sendo que dados operacionais não são relevantes quando estudados separadamente. Para isso, é preciso transformar esses dados em informação (CASTANHEIRA, 2008). Sendo assim, a Mineração de Dados é a parte mais importante do processo de descobrimento de conhecimento (*Knowledge Discovery in Databases – KDD),* dentre as técnicas e tarefas de *Data Mining* temos: Classificação, Regressão, Associação, Segmentação, Sumarização, Regras de associação, Árvores de Decisão, Raciocínio Baseado em Caso ou MRB, Algoritmos Genéticos e Redes Neurais Artificiais.

O uso de agentes animados 3D que utilizam síntese de voz para comunicação com usuário, tem se diversificado com o avanço continuo da tecnologia, e atualmente há no mercado tecnológico grandes ferramentas que interagem de maneira dinâmica e real, como exemplo, podemos citar o *TalkingTo Tom Cat*, um aplicativo desenvolvido pela

OutFit7 para o sistema operacional *Android* que utiliza síntese de voz como meio de interação entre o aplicativo e o usuário.

O presente trabalho será desenvolvido utilizando a técnica de mineração de dados especial (Árvores de Decisão e Regras de Classificação) associado a um agente animado 3D que conta ainda com processamento, reconhecimento de voz como meio de comunicação.

As técnicas supracitadas, mineração de dados e reconhecimento de voz, serão utilizadas para desenvolver um aplicativo na área da saúde. A rede de saúde pública do município de Marabá é vista como referência regional na demanda de pacientes do Sul e Sudeste do Estado do Pará. O município conta com apenas um Hospital Municipal para atendimentos a diversos casos de médio e grave porte, além disso, conta com um hospital do tipo maternidade, para atender somente grávidas e um Hospital Regional para atendimento de casos de alta complexidade (riscos de mortes). Toda essa estrutura para atender aproximadamente 700 mil habitantes. A rede de saúde pública de Marabá vem sofrendo dificuldades para atender toda essa demanda. Os fatores principais são: a grande demanda de pacientes da região e a carência de profissionais qualificados na área. As políticas e os programas de prevenção contra doenças vêm sendo desempenhadas com frequência no município, contudo não têm sido suficientemente eficazes para evitá-las.

Através de pesquisas e estudos realizados na Secretaria de Saúde Marabá, foram feitos estudos e levantamentos das principais doenças da região, portando o presente estudo propõe auxiliar na identificação de uma das principais doenças da região, reduzindo o tempo de espera do atendimento ao paciente. Para isso, propomos elaborar um sistema aplicando uma técnica de *Data Mining* recolhendo informações de forma a identificar os sintomas dos pacientes e gerando o resultado automaticamente para auxiliar no diagnóstico da doença. Outro aspecto é a construção de uma animação 3D, que interaja com o paciente via comando de voz, facilitando a comunicação entre o paciente e o sistema. A vantagem de se utilizar animações com processamento de voz é uma melhor interação entre o paciente (muitas vezes leigo em informática) e o sistema.

Para o desenvolvimento da interface foi utilizado o JAVA ALICE 3D para construção de um agente animado que possuirá em sua base dados, informações sobre as características da sífilis. O mesmo estabelecerá uma comunicação com o paciente,

onde através de uma série de perguntas gerenciadas por uma técnica de mineração de dados, atingiremos uma “tomada de decisão”, onde assim será pressuposto a possível contaminação do paciente pela bactéria da sífilis, agilizando, então, o diagnóstico médico.

Objetiva-se com este trabalho implementar uma interface utilizando uma agente 3D e reconhecimento de voz para interagir com o usuário, afim de gerar uma base de dados através de técnicas de mineração de dados, com os sintomas dos pacientes, para auxiliar no diagnóstico de médico para pacientes possivelmente infectados pela sífilis do município de Marabá e região. A fim de se obter um resultado satisfatório desta implementação, pôde-se:

- Levantar os dados das doenças mais frequentes de Marabá e região, fornecidos pela Secretaria Municipal de Saúde de Marabá, através de documentos digitais do controle de epidemias e escolher uma para implementação inicial.
- Modelar e implementar o banco de dados utilizando o SGBD MySQL.
- Construir um objeto 3D, utilizando a API do Java para interação com o usuário e adição de reconhecimento de voz, aplicando-se uma técnica de mineração de dados para identificar a possível doença do usuário.
- Modelar o sistema utilizando alguns diagramas especificados na UML (Linguagem de Modelagem Unificada).

A proposta baseia-se em auxiliar na detecção da Sífilis, agilizando o diagnóstico. Para facilitar o entendimento, o presente trabalho está dividido da seguinte maneira: No capítulo 2 aborda sobre mineração de dados, o capítulo 3 explica como funcionam as técnicas de reconhecimento de voz na plataforma Linux, o capítulo 4 explana sobre os fundamentos da computação gráfica, o capítulo 5 trata dos trabalhos relacionados, o capítulo 6 mostra as ferramentas utilizadas para desenvolvimento do projeto, o capítulo 7 o estudo de caso do trabalho e por fim os capítulos 8 e 9 abordam as considerações finais e as referências bibliográficas, respectivamente.

2. MINERAÇÃO DE DADOS

Este capítulo descreverá assuntos relacionados à Mineração de Dados e árvore de decisão, descrevendo suas funcionalidades como também maneiras e usabilidades da mesma. Para isso foi feito uma introdução e em seguida uma abordagem sobre áreas de aplicação, técnicas e tarefas. Finalizando demonstrando a construção de uma árvore de decisão.

2.1 Mineração de Dados

Mineração de Dados é uma área de pesquisa multidisciplinar, incluindo tecnologia de bancos de dados, inteligência artificial, aprendizado de máquina, redes neurais, estatística, reconhecimento de padrões, sistemas baseados em conhecimento, recuperação da informação, computação de alto desempenho e visualização de dados, também pode ser considerada uma técnica para determinar padrões de comportamentos em grandes bases de dados, auxiliando na tomada de decisão, (FAYYAD, 1996).

> Mineração de Dados é um ramo da computação que teve início nos anos 80, quando os profissionais das empresas e organizações começaram a se preocupar com os grandes volumes de dados informáticos estocados e inutilizados dentro da empresa. Nesta época, *Data Mining* consistia essencialmente em extrair informação de gigantescas bases de dados da maneira mais automatizada possível. Atualmente, *Data Mining* consiste, sobretudo, na análise dos dados após a extração, buscando-se, por exemplo, levantar as necessidades reais e hipotéticas de cada cliente para realizar campanhas de marketing.

A Mineração de Dados é considerada por muitos autores como uma parte do processo KDD (Descoberta de Conhecimento de Dados), sendo considerada também, a etapa mais importante desse processo. Segundo Goebel e Gruenwald (1999), o termo KDD é usado para denotar todo o processo de transformar dados de baixo nível em alto nível de conhecimento. De acordo com Fayyad; Piatetsky-Shapiro e Smyth (1996, p.40-41), uma definição simples de KDD é: “Descoberta de Conhecimento em Bancos de Dados é o processo não trivial de identificação original, válido, potencialmente útil, e em última análise compreensível nos padrões de dados”.

Ao analisar os principais conceitos de mineração de dados, podemos definir que o principal objetivo da mesma é verificar a interação entre os dados e fornecer

informações para critérios de comparação de dados futuros baseado em dados do passado. “Os resultados obtidos com a mineração de dados podem ser usados no gerenciamento de informação, processamento de pedidos de informação, tomada de decisão, controle de processo e muitas outras aplicações” (DIAS, 2002, p.2).

2.1.1 Áreas de Aplicação de Técnicas de Mineração

Conforme Dias (2002, apud VIVEROS et al., 1996; MANNILA, 1997; CRATOCHVIL, 1999) as áreas a seguir são as principais para aplicação das técnicas de mineração de dados:

- Marketing: Técnicas de mineração de dados são aplicadas para descobrir preferências do consumidor e padrões de compra, com o objetivo de realizar marketing direto de produtos e ofertas promocionais, de acordo com o perfil do consumidor.
- Detecção de fraudes: Muitas fraudes óbvias (tais como, a compensação de cheque por pessoas falecidas) podem ser encontradas sem mineração de dados, mas padrões mais sutis de fraude podem ser difíceis de ser detectados, por exemplo, o desenvolvimento de modelos que predizem quem será um bom cliente ou aquele que poderá se tornar inadimplente em seus pagamentos.
- Medicina: Caracterizar comportamento de paciente para prever visitas, identificar terapias médicas de sucesso para diferentes doenças, buscar por padrões de novas doenças.
- Instituições governamentais: Descoberta de padrões para melhorar as coletas de taxas ou descobrir fraudes.
- Ciência: Técnicas de mineração de dados podem ajudar cientistas em suas pesquisas, por exemplo, encontrar padrões em estruturas moleculares, dados genéticos, mudanças globais de clima, oferecendo conclusões valiosas rapidamente.
- Controle de processos e controle de qualidade: Auxiliar no planejamento estratégico de linhas de produção e buscar por padrões de condições físicas na embalagem e armazenamento de produtos.

2.1.2 Técnicas e Tarefas de Mineração de Dados

Em seu trabalho, Amo (2003) afirma que é importante distinguir o que é uma tarefa e o que é uma técnica de mineração de dados: "A tarefa consiste na especificação do que estamos querendo buscar nos dados, que tipo de regularidades ou categoria de padrões tem interesse em encontrar, ou que tipo de padrões poderiam nos surpreender (por exemplo, um gasto exagerado de um cliente de cartão de crédito, fora dos padrões usuais de seus gastos)". A técnica de mineração consiste na especificação de métodos que nos garantam como descobrir os padrões que nos interessam. Dentre as principais técnicas utilizadas em mineração de dados, temos técnicas estatísticas, técnicas de aprendizado de máquina e técnicas baseadas em crescimento-poda-validação.

2.1.2.1 Principais Técnicas de Mineração de Dados

Existem diversas formas de resolver diversos problemas, porém, no contexto de Mineração de Dados, não há uma técnica que resolva todos os problemas (BALLARD et al., 1998). É necessário conhecer bem o ambiente o qual o problema está inserido, para escolher a melhor técnica de mineração de dados. As principais técnicas de mineração de dados são: Regras de Associação, Árvores de Decisão, Raciocínio Baseado em Casos MBR, Algoritmos Genéticos e Redes Neurais Artificiais. A seguir, temos uma breve especificação de cada técnica (DIAS, 2001):

- Regras de associação: estabelece uma correlação estatística entre atributos de dados e conjuntos de dados;
- Árvore de decisão: hierarquização dos dados, baseando-se em estágios de decisão (nós) e na separação de classes e subconjuntos;
- Raciocínio baseado em casos ou MBR: baseado no método do vizinho mais próximo combina e compara atributos para estabelecer hierarquia de semelhança;
- Algoritmos Genéticos: métodos gerais de busca e otimização, inspirados na Teoria da Evolução, onde a cada nova geração, soluções melhores têm mais chance de ter "descendentes";

- Redes neurais artificiais: modelos inspirados na fisiologia do cérebro, na qual o conhecimento é fruto do mapa das conexões neuronais e dos pesos dessas conexões.

De acordo com Dias (2002, apud HARRISON, 2002,), a escolha das técnicas de mineração de dados dependerá da tarefa específica a ser executada e dos dados disponíveis para análise. Dias (2002, apud BERRY; LINOFF, 2002), diz que a técnica de seleção de dados deve ser dividida em dois passos:

1. Traduzir o problema de negócio a ser resolvido em séries de tarefas de mineração de dados;

2. Compreender a natureza dos dados disponíveis em termos de conteúdo e tipos de campos de dados e estrutura das relações entre os registros.

O presente trabalho é voltado para seleção de dados de acordo com os sintomas dos pacientes que irão interagir com o sistema, para tanto, será estabelecido uma ordem de inclusão e exclusão de dados conforme sugerida pelo paciente. Nesse sentido, o objetivo do trabalho se enquadra em Arvores de Decisão e na tarefa de Classificação, uma vez que irá se identificar a doença pré-estabelecida na base de dados, neste caso, as principais doenças do município de Marabá – PA.

2.1.2.2 Principais Tarefas de Mineração de Dados

A mineração de dados é uma técnica que vem sendo difundida gradativamente, com isso, suas tarefas também vêm evoluindo paralelamente e se diversificando cada vez mais. Essas tarefas podem gerar diferentes tipos de conhecimentos, sendo necessária a definição da técnica logo no início do processo de mineração de dados para se saber qual tipo de conhecimento deve ser extraído (CASTANHEIRA, 2008). As tarefas de Mineração de Dados devem ser utilizadas de acordo com a necessidade da problemática, para isso, Dias (2002) classifica basicamente os parâmetros de escolha de técnicas e tarefas das ferramentas de mineração de dados:

- Classificação: constrói um modelo de algum tipo que possa ser aplicado a dados não classificados a fim de categorizá-los em classes, o objetivo é

descobrir um relacionamento entre um atributo meta (cujo valor será previsto) e um conjunto de atributos de previsão.

- Regressão: usada para definir um valor para alguma variável contínua desconhecida.
- Associação: usada para determinar quais itens tendem a ser adquiridos juntos em uma mesma transação.
- Segmentação: Processo de partição de uma população heterogênea em vários subgrupos ou grupos mais homogêneos.
- Sumarização: Envolve métodos para encontrar uma descrição compacta para um subconjunto de dados.

2.2 Árvore de Decisão

De forma geral, Arvores de Decisão são técnicas de classificação de dados, sendo utilizadas vantajosamente no processo de tomadas de decisão, levando em consideração os atributos mais importantes e de fácil compreensão para a maioria das pessoas (CREPALDI et al., 2011). Pozzer (2006, p.1) afirma que:

> Árvores de decisão são ferramentas que podem ser utilizadas para dar ao agente a capacidade de aprender, bem como para tomar decisões. A ideia de aprendizado é que os perceptores (elementos do agente que percebem o mundo) não sejam usados apenas para agir, mas também para aumentar a capacidade do agente de agir no futuro. O aprendizado ocorre na medida em que o agente observa suas interações com o mundo e seu processo interno de tomada de decisões.

Segundo Soma et. al., 2004 (apud GAMA, 2004, p. 4), as Árvores de Decisão utilizam a estratégia *dividir-e-conquistar* ("*divide-and-conquer*"), onde as árvores são construídas utilizando-se de apenas alguns atributos. As Árvores de Decisão são uma das técnicas de aprendizado de máquina ("*machinelearning*"), onde um problema complexo é decomposto em subproblemas mais simples.

As Árvores de Decisão podem consistir basicamente em (GARCIA, 2000):

- Nodos (nós) que representam os atributos;
- Arcos (ramos), provenientes dos nodos e que recebem os valores possíveis para estes atributos (cada ramo descendente corresponde a um possível valor deste atributo).

- Nodos folha (folhas da árvore), que representam as diferentes classes de um conjunto de treinamento, ou seja, cada folha está associada a uma classe.
- Cada percurso na árvore (da raiz à folha) corresponde a uma regra de classificação.

2.2.1 Construção da Árvore de Decisão

O processo de construção de uma Árvore de Decisão inicia-se a partir de um conjunto de treinamento, que contém exemplos com classes previamente conhecidas (dados históricos), Soma et al. (2004, p.5), diz que:

> Para construir uma árvore de decisão com uma alta taxa de predição é necessário fazer a escolha correta das características que serão utilizadas como teste na combinação dos casos. Estes testes devem gerar uma árvore com o menor número possível de subconjuntos, fazendo com que cada folha da árvore contenha um número significativo de casos. O ideal é escolher os testes de modo que a árvore final seja a menor possível.

As Árvores de Decisão são semelhantes às regras de decisão SE-ENTÃO. A árvore de decisão chega a sua decisão pela execução de uma sequência de testes. Cada nó interno da árvore corresponde a um teste do valor de uma das propriedades, e os ramos deste nó são identificados com os possíveis valores do teste. Cada nó folha da árvore especifica o valor de retorno se a folha for atingida (POZZER, 2006).

A seguir temos na figura 1, um exemplo de uma possível árvore de decisão para o problema de diagnosticar pacientes:

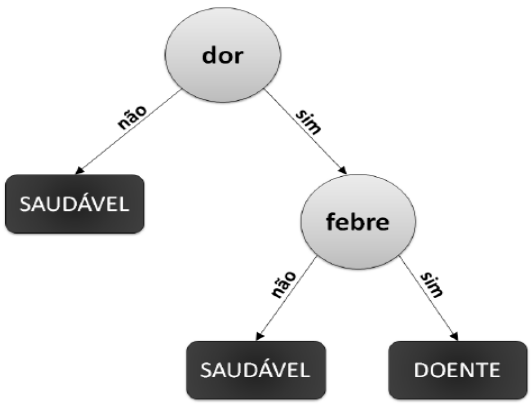


Figura 1: Exemplo de árvore de decisão.
FONTE: Basgalupp, 2010.

Através deste método é possível diagnosticar um paciente como saudável ou doente: para isso, basta partir do nodo raiz da árvore e ir percorrendo-a, através das respostas aos testes dos nodos internos, até chegar a um nodo folha, o qual indica a classe correspondente do novo paciente. Além da obtenção da classe, a grande vantagem é que a trajetória percorrida até o nodo folha representa uma regra, facilitando a interpretabilidade do modelo pelo usuário, no caso um médico.

2.3 Resumo do Capítulo

Esse capítulo abordou assuntos relacionados à Mineração de Dados e árvore de decisão, bem como a maneira correta de criar e manipulá-las.

O próximo capítulo irá tratar da síntese e reconhecimento de voz relacionada ao tema proposto. Será apresentada uma série de abordagens, técnicas e propostas para o desenvolvimento do trabalho.

3. RECONHECIMENTO DE VOZ

Este capítulo busca descrever algumas tecnologias de reconhecimento de Voz sua importância para interação, humana-máquina, bem como o relato de algumas APIs. Para isto foi feita uma abordagem histórica da evolução e valorização da mesma ao longo do tempo e os resultados alcançados atualmente.

3.1 Reconhecimento de Voz

Segundo Louzada apud Olson, (1956), o reconhecimento de fala vem sendo ativamente estudado desde 1950 com o estudo inicial de ideias básicas da fonética acústica. Por volta de 1950 até 1959, os estudos foram liderados por pesquisadores

americanos e ingleses, pesquisadores como Olson e Belar, do *RCA Laboratories*, mas foi a partir dos anos 80 que o processo de pesquisa no reconhecimento da fala aumentou de forma crucial. Ao contrário do que se via anteriormente, o processo de pesquisa passou a ser realizado a partir de frases fluentes. As pesquisas nos anos 80 foram caracterizadas pelas aproximações estatísticas, especialmente pelos Modelos Ocultos de Markov, Louzada (apud FERGUSON, 1980).

Nos últimos anos, o desempenho dos computadores pessoais tem evoluído com o advento de processadores cada vez mais velozes, fato que viabiliza o uso das tecnologias de voz, tecnologias que possuem grande potencial para a criação de aplicativos que possibilitem uma eficaz interação humano-computador, sendo a fala um mecanismo natural para tal interação. A tecnologia de processamento de voz encontra-se bastante avançada e, em escala mundial, existe vasta disponibilidade de software, tanto comercial quanto acadêmico. A maioria dos mesmos assume a disponibilidade de um reconhecedor e/ou sintetizador, conhecidos genericamente na literatura como *engines*[1], os quais podem ser programados via *Application Programming Interface* (API).

Com a possibilidade de comunicação através de linguagem natural falada percebemos um importante passo para a melhoria da interação dos usuários com aplicativos de computador. Esse desafio é considerado por muitos como um dos mais importantes da computação moderna.

> O reconhecimento da fala por computador consiste em mapear um sinal acústico, capturado por um transdutor (microfone ou telefone) em um conjunto de palavras. A finalidade básica de um sistema de reconhecimento de fala, comumente conhecido como sistemas Reconhecimento Automático de Fala (RAF), é produzir como saída a sequência de palavras ou sentenças correspondentes ao sinal de entrada (transcrição). Os sistemas de reconhecimento de fala podem ser caracterizados e avaliados por vários parâmetros como: vocabulário (vocabulary), dependência de locutor (enrollment), modo de fala (speakingmode), perplexidade (perplexity), condições adversas (SNR) (YNOGUT, 1999, p.1).

O objetivo é após a transdução da fala como sinal analógico para sinal digital, poder reconhecer o comando falado de tal forma que tenha o maior grau de verossimilhança possível, para que sejam então, realizados com maior exatidão (maior que setenta por cento) os comandos armazenados na gramática do sistema, demonstrando assim a aplicabilidade do reconhecimento de fala.

---

1 Programa de computador e/ou conjunto de bibliotecas, para simplificar e abstrair o desenvolvimento de jogos eletrônicos ou outras aplicações com gráficos em tempo real, para videogames e/ou computadores rodando sistemas operacionais.

3.1.1 Automatic Speech Recognition (ASR) e Text To Speech (TTS)

O ASR ou RAV (Reconhecimento Automático de Voz) consiste em mapear um sinal acústico, capturado por um transdutor, usualmente um microfone ou telefone, em um conjunto de palavras, ou seja, dada uma entrada em forma de um sinal, onda acústica, produzir uma saída em forma de sequência de fonemas, palavras ou sentenças correspondentes ao sinal de entrada (SILVA, 2008).

Text-to-speech (TTS) é constituído por módulos que convertem textos em linguagem natural em voz sintetizada. O principal desafio é fazer com que a voz sintetizada soe menos robotizada e mais humana (DANTAS, 2011). Dentre os fatores que dificultam atingir esse objetivo, encontra-se a modelagem da prosódia, uma das principais responsáveis para que a voz sintetizada carregue o ritmo e emoção que um ser humano empregaria.

3.2 APIs de Voz

Uma API especifica uma interface, que no nosso caso, suporta aplicações em TTS e ASR, essa última tanto em aplicações com gramática para comando e controle, como para ditado. Dessa forma, as APIs não contêm apenas as funcionalidades de TTS e ASR, mas também métodos e eventos que permitem ao programador abstrair requisitos de baixo nível do *engine*. Os *engines* possuem sua própria API, mas existem pelo menos duas especificações desenvolvidas para uso geral: o *Speech API ou Speech Application Programming Interface* (SAPI), (MICROSOFT, 2013) e a *Java Speech API* (JSAPI, 2013).

Um exemplo é o "Coruja" que tem sua descrição encontrada em Fala Brasil (2013) qual consiste em uma API que utiliza o Julius, um *engine* ASR completo integrado à plataforma.NET, possibilitando o desenvolvimento de aplicativos com ASR em PB para o Windows. O objetivo dos autores do Coruja é prover ferramentas que permitam o desenvolvimento de aplicativos aurais, sem que isso dependa de plataforma. Por isso o software Coruja foi expandido como a adição do API em Java para o software Coruja (JLaPSAPI, 2013).

A JLaPSAPI permite a utilização da *Java Speech Application Programming Interface* (JSAPI) (JSAPI, 2013), possibilitando usar-se a importante característica de portabilidade que Java possui.

3.2.1 Engine Julius

Um *engine* voltado para o reconhecimento da fala é um programa de computador e/ou conjunto de bibliotecas, para simplificar e abstrair o desenvolvimento de aplicações voltadas para essa área. Existem três tipos de *engines* segundo Sampaio Neto (2006); dependentes de locutor, independentes de locutor e adaptativas.

Os *engines* desenvolvidos com dependência de locutor necessitam de um longo e cansativo treinamento adicional. Além do sistema se adaptar às características acústicas do usuário, o locutor também acaba ajustando seu estilo de voz para conversar com o computador. Esses softwares não demandam muita capacidade da máquina, proporcionam bom desempenho e são mais empregados em aplicações de uso individual, ou para pessoas com deficiência na fala. Já nos *engines* com independência de locutor, os treinamentos extras são dispensáveis, pois os mesmos são implementados para serem utilizados por qualquer pessoa, não interessando a sua origem, sendo ideal para ambientes públicos. Por último, existem os *engines* adaptativos, que tampouco necessitam de treinamento, se aprimorando, contudo, com o uso. Com isso, podem ser falhos quando utilizados por um grupo muito variado. Tanto os *engines* independentes, como os adaptativos, precisam dispor de uma elevada capacidade computacional e apresentam resposta razoável (SAMPAIO NETO, 2006).

O Julius é um reconhecedor *open-sourceengine* para ASR de alto desempenho para grandes vocabulários e independente de locutor que permite uma grande gama de opções além de suportar modelos (acústicos e de linguagem). Seu pacote de distribuição traz uma API que pode ser acessada via código em linguagem C/C++. Dentre as dificuldades em se usar essa API, está o fato da mesma não seguir uma especificação, portanto, o código da aplicação que controla o Julius não pode ser reaproveitado para manipular outro *engine* (FALA BRASIL, 2013).

3.2.2 Sapi e Jsapi

Atualmente, existe um grande número de empresas no mercado que apresentam soluções para que um desenvolvedor possa incorporar a tecnologia de voz em seus aplicativos. Empresas como a IBM e Microsoft possuem soluções que adotam padrões como Voice XML (Extensible Markup Language) e Salt. A maioria dessas empresas adota uma API para o desenvolvimento de aplicativos, como SAPI (*Speech API*) da Microsoft e a JSAPI (*Java™ Speech API*) da Sun.

A SAPI provê aos programadores acesso aos serviços fornecidos por um *engine* de ASR e/ou TTS. A Microsoft disponibiliza gratuitamente na Web o kit de desenvolvimento Speech SDK, com *Dynamic-link library*, Biblioteca de Ligação Dinâmica (DLL's) que suportam objetos *Object Linkingand Embedding* (OLE), característica essencial para que possa ser utilizado por linguagens de programação conhecidas como de alto nível. (MICROSOFT, 2013).

A *Java™ speech API* é uma especificação que permite aos desenvolvedores incorporarem tecnologia de voz aos seus aplicativos escritos em Java. JSAPI especifica uma interface que independe da plataforma de desenvolvimento e suporta sistemas para ditado, reconhecedores e sintetizadores de voz. Ambas permitem o emprego das tecnologias de TTS e ASR. A JSAPI deixa a desejar num ponto de extrema importância em TTS, ao não permitir o redirecionamento das amostras de áudio para arquivos e/ou dispositivos além da saída de áudio padrão. Porém, ambas suportam o uso de gramáticas *context free grammar* ou Gramática Livre de Contexto (CFG) e de ditado (probabilísticas) (JSAPI, 2013).

A SAPI utiliza-se da XML para incluir “marcas” prosódicas (formatar) o texto a ser sintetizado, ou seja, configurar os parâmetros de síntese de voz como velocidade, afinação, timbre e língua. Já com JSAPI, também é possível trabalhar com esses parâmetros tanto através de invocação de procedimentos e/ou métodos, como pela JSML (*Java Synthesis Markup Language*).

3.2.3 A JLaPSAPI

A JLaPSAPI é uma API proposta, desenvolvida a partir da LaPSAPI, API de reconhecimento de fala, que permite o uso do decodificador Julius (conseqüentemente do *engine* Coruja) em aplicativos Java. Apesar de a LaPSAPI ser suficiente e robusta para o desenvolvimento de aplicativos, esta não segue nenhuma especificação reconhecida, ou seja, um programa escrito em cima dessa API, não poderia trocar seu engine ASR (JLaPSAPI, 2013).

JLaPSAPI opera sobre a API do Coruja (LaPSAPI) para controlar o *engine* Julius, evitando a reimplementarão de funcionalidades básicas já implementadas na atual versão do Coruja (Coruja 0.2). A comunicação entre o código em Java e o código C++ é provida pela *Java Native Interface* (JNI). O acesso ao Coruja é feito através de código especificado pela JSAPI. O programador tem, agora, a possibilidade de alternar entre a Coruja e qualquer outro *engine* que siga a especificação JSAPI, como o Sphinx-4, sem a necessidade de alteração no código da sua aplicação Java (LaPSAPI, 2013).

Coruja é um software para reconhecimento de voz em Português Brasileiro. Composto por uma API desenvolvida em C++ com suporte a CLR (*Common Language Runtime*) para controle em tempo real do decodificador Julius e um modelo acústico (LaPSAM) construído com a ferramenta *The Hidden Markov Model* (HTK), consiste de uma série de módulos de biblioteca e ferramentas (FALA BRASIL, 2013).

3.3 Resumo do Capítulo

Esse capítulo abordou assuntos relacionados ao reconhecimento de voz através de APIs,

O próximo capítulo irá explanar sobre computação gráfica, será apresentada uma série de abordagens, técnicas e propostas que estão disponíveis na literatura para em seguida traçar um comparativo com o projeto em questão.

4. COMPUTAÇÃO GRÁFICA

Este capítulo tratará sobre os conceitos essenciais de computação gráfica, focando para Java 3D e Java Alice 3D, descrevendo variâncias, dispositivos e possibilidades para uma implementação 3D.

4.1 Computação Gráfica

Azevedo e Conci (2003), afirmam que a computação gráfica é matemática e arte, podendo ser encarada como uma ferramenta não convencional que permite o artista transcender das técnicas tradicionais de desenho ou modelagem, proporcionando-o um novo impulso, abrindo novos horizontes e fornecendo meios para se fazer um novo tipo de arte, onde a matemática é a linguagem do homem com a natureza, isto é, similar a natureza em computadores.

Segundo a ISO (*International Organização for Standardization*), a computação gráfica é um conjunto de ferramentas e técnicas para converter dados para ou de um dispositivo gráfico através do computador (AZEVEDO E CONCI, 2003).

As primeiras técnicas de interação, surgiu em 1962, com a tese de Ivan Sutherland (Sketchpad *–A Man-Machine Graphical Communication System*) que usava o teclado e a caneta ótica para desenhar, apontar e escolher alternativas, incluindo noções de estruturação de dados (AZEVEDO E CONCI, 2003).

4.1.1 Interação com Computação Gráfica 3D

“A definição de Homem-Computador tem sido evoluída ao longo do tempo. A princípio tratava basicamente de uma seqüência de estímulos e respostas, como na interação de corpos físicos” (BARBOSA E SILVA, 2010, p. 20). Barbosa (apud CARD; MORAN; NEWELL, 1983) continua dizendo que, com o surgimento das pesquisas de

base cognitiva, decorreu-se a destacar a interação como a comunicação com máquinas ao invés de a operação de máquinas.

Segundo Multigner (1994), o conceito de interação vem da física, incorporado pela sociologia, psicologia social e por fim no campo da informática transmutando em interatividade. Na Física o conceito está relacionado ao comportamento de partículas que tem seu movimento alterado a partir do movimento de outras partículas. Na Sociologia e psicologia social, a interação está direcionada à ação humana ou social. O termo interatividade surge através da arte, em meados de 1960. Porém, somente no início dos anos 70 torna-se conhecido no campo da informática (SOUSA, 2008). Alguns autores não fazem diferença entre a definição de interação e interatividade, uns citam interação como relação humana e interatividade como a relação homem-máquina (SILVA, 1998).

Em geral, uma interação entre homem-computador, pode-se dizer que é tudo o que acontece quando uma pessoa se une a um sistema para realizar tarefas visando algum objetivo (BARBOSA; SILVA, 2010; apud HIX; HARTSON, 1993).

Conforme Sousa (2008) com os sistemas 3D, a interação está diretamente ligada aos dispositivos que tornam esta interação possível, denominados dispositivos de entrada e saída (E/S). Azevedo e Conci (2003), fala que os dispositivos de E/S, são elementos críticos da computação gráfica, por quais há a interação com o sistema na busca de uma extensão dos limites do corpo e uma melhor comunicação com a máquina.

4.2 Dispositivos de Entrada e Saída

Os dispositivos de entrada, são componentes que permitem a interação e movimentação com os sistemas, surgindo a cada dia novos dispositivos com recursos que agilizam na tarefa de interação (AZEVEDO; CONCI, 2003). Na figura 2, mostra alguns dispositivos mais usados na computação gráfica, que se podem citar: Teclado, Mouse, *Joysticks*, *Tablet* e Mesa Digitalizadora.

Figura 2: Dispositivo de entrada.
FONTE: Americanas, 2013.

Os dispositivos de saída são classificados em duas categorias de acordo com a forma pela qual as imagens são geradas, vetoriais (seguimento de reta) e matriciais (arranjo de elementos em duas direções) (AZEVEDO; CONCI, 2003). Realizando a decodificação dos dados em informações que possam ser interpretadas pelo usuário. Vejamos na figura 3, alguns destes dispositivos como: Caixas De Som, Monitores, Impressoras, *Plotters*[2].

Figura 3: Dispositivo de saída.
FONTE: Kalunga, 2013.

4.2.1 Dispositivo de Entrada e Saída 3D

De acordo com Azevedo e Conci (2003), com a popularização dos sistemas 3D e barateamento dos componentes eletrônicos, teve-se o crescimento de variados tipos de dispositivos de E/S específicos para determinados ambientes tridimensionais.

[2] Impressora de grande porte designada a realizar impressões de grandes formatos ou de grandes dimensões com alta qualidade.

Ultimamente, podemos presenciar em lojas de informática, simuladores de esqui, skate, tênis e outros, sem muita dificuldade. Dentre alguns dispositivos de E/S 3D, na figura 4, pode-se citar: Digitalizador Tridimensional, Scanners Tridimensionais, Luvas, Capacetes, *3D Controllers*, dentre outros.

Figura 4: Dispositivo de entrada e saída 3D.
FONTE: Gizmag, 2013; Sabbatini, 1993.

Além das tecnologias de dispositivos de E/S utilizadas para a interação entre homem-computador, existem outras tecnologias como as interfaces gráficas que permitem modelagem de objetos tridimensionais (personagens 3D), grafos de cenas e outros, possibilitando traçar o real no virtual. Um exemplo destas tecnologias é a API Java Alice 3D, implementada tanto na sua linguagem natural como na linguagem Java.

4.3 Linguagem Java e Java Alice 3D

4.3.1 Linguagem Java

Java é uma linguagem de programação orientada a objetos, que foi desenvolvida pela Sun Microsystems (SOUSA JUNIOR, 2011), independente de plataforma. Segundo Manssour (2003), Atualmente, é uma das linguagens mais utilizadas para o desenvolvimento de sistemas, e pode ser obtida gratuitamente. No entanto, ela é tanto compilada como interpretada, ou seja, o compilador transforma o programa em *bytecodes*, que consiste em um tipo de código de máquina específico da linguagem Java; o interpretador, disponível na JVM (*Java Virtual Machine*) que pode ser instalada

em qualquer plataforma, transforma os *bytecodes* em linguagem de máquina para execução, sem que seja necessário compilar o programa novamente.

Para execução de programas gráfico em Java é preciso utilizar o J2SE (*Java 2 Platform, Standard Edition*), incluindo o compilador a JVM e a API (*Application Programming Interface*) que engloba os pacote contendo as classes responsáveis pelas funcionalidades de entrada e saída, interface gráfica, coleções, em meio a outras (MANSSOUR, 2003). Além do mais, existe a *Java Standard Extension* API, que inclui outros pacotes, tais como acesso a banco de dados e o *Java Media Framework*, que suporta tecnologias gráficas e multimídia.

4.3.2 Java Alice 3D

O software ALICE foi criado em 1992 com a finalidade de possibilitar a fácil criação de ambientes, tanto interativos como não interativos, com várias possibilidades de uso, adequados a situações diversas, podendo ser utilizado tanto em contextos educacionais, como em outros ambientes. Alice 3D é um ambiente de programação tridimensional grátis e aberto, desenvolvido em Java e multi plataforma de fácil utilização no qual podem ser criadas animações e interações entre personagens e objetos lembrando muito jogos de vídeo game. Foi desenvolvido com o objetivo de minimizar as falhas denotadas no inicio da aprendizagem da programação (ALICE, 2010).

Sua interface gráfica possibilita desenvolver programas tridimensionais, com animação e interativos, com facilidades de criação de códigos dos programas, uma vez que não é necessário digitar qualquer linha de comando, bastando apenas clicar e arrastar objetos, métodos e funções existentes no ALICE, suportando linguagens orientadas a objetos, tais como C++ e Java (ALICE, 2010). Dispõe ainda de um conjunto de objetos e métodos fixos, que utilizados em conjunto geram os movimentos, ou eventos, e animações da aplicação. Além disso, permite que os mesmos métodos fixos sejam combinados de várias formas para gerar novos métodos e outros eventos (TCHEMRA; GRINKRAUT, p. 17-23).

4.4 Resumo do Capítulo

Esse capítulo abordou assuntos relacionados à computação gráfica, com foco para a Java Alice 3D, dispositivos de entrada e saída que tem facilitado o dia a dia das pessoas nessa interação humana-máquina, com objetivo de enfatizar a relevância da ferramenta a ser utilizada para desenvolvimento do trabalho.

O próximo capítulo irá tratar sobre a metodologia adotada no processo de desenvolvimento do trabalho bem como uma descrição técnica das ferramentas que foram utilizadas no processo de analise, projeto e estruturação e implementação do Sistema.

5. TRABALHOS RELACIONADOS

Este capítulo descreve três trabalhos relacionados encontrados durante a pesquisa de desenvolvimento deste trabalho. A seção seguinte menciona um pouco de cada trabalho, apresentando suas conclusões básicas assim como algumas críticas.

5.1 Proposta de Sistemas Relacionados

A proposta apresentada por Borges e Carvalho (2009) tem como objetivo desenvolver uma interface que, utilizando a tecnologia de Reconhecimento de Voz, possa melhorar o pronto-atendimento nas emergências das unidades hospitalares. Sendo desenvolvida em Java e utilizando a Java Speech API e a IBM ViaVoice como mecanismos para tratar, processar e manipular os dados requeridos na interface, auxiliando no preenchimento dos campos da interface e produção de um relatório médico utilizado nos atendimentos emergenciais em unidades hospitalares.

Nessa monografia o sistema foi desenvolvido para funcionar em computadores *desktop,* caso o sistema seja usado em duas ou mais máquinas, o indicado seria que a gramática fosse acessada através da internet, pois desta forma as alterações feitas na

mesma serão automaticamente disponibilizadas, sem a necessidade de alterá-la em cada máquina, eliminando a necessidade da paralisação do atendimento. Na figura 5 apresenta a interface principal deste sistema.

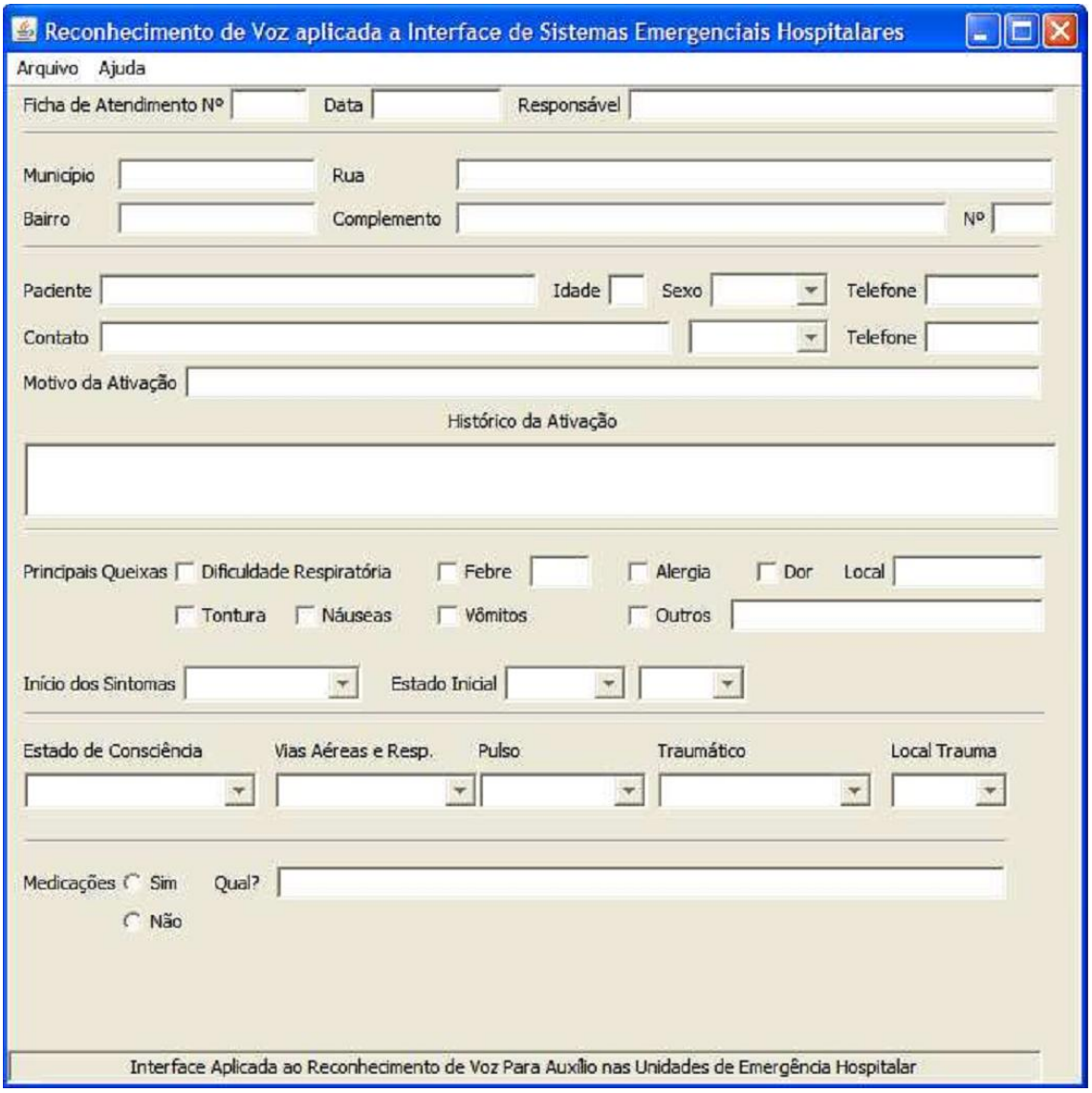


Figura 5: Interface do sistema emergencial hospitalares.
FONTE: Borges e Carvalho, 2009.

Em Morais et. al. (2012) o autor propõe a experiência da construção de um sistema móvel de suporte à decisão para apoio ao diagnóstico médico utilizando inteligência computacional. Aborda aplicação simultânea de diferentes metodologias ágeis, aplicabilidade de dispositivos móveis em saúde e utilização de inteligência artificial em sistemas de apoio a decisão médica. O sistema é composto pelos módulos aplicação móvel e servidor. Na aplicação móvel utilizou-se a tecnologia Java e a plataforma *Android*. A principal funcionalidade do sistema é prover o diagnóstico de

asma utilizando mineração de dados através de árvore de decisão por classificação, sendo estas funcionalidades executadas com ou sem conexão com o servidor. A figura 6 apresenta umas das interfaces deste sistema.

Figura 6: Interface do sistema InteliMed.
FONTE: Morais et. al., 2012.

Na proposta descrita em Pires et. al. (2004) propõe uma arquitetura para suporte ao desenvolvimento de sistemas de prontuário eletrônico integrados a sistemas de auxílio ao diagnóstico com o propósito de compartilhar casos e conhecimentos clínicos entre a comunidade médica na Internet. Também é apresentada uma aplicação baseada em um estudo de caso sobre sinais, sintomas e diagnósticos relacionados à dor abdominal aguda, tendo o paradigma simbólico, em especial a Árvore de Decisão, como técnica de aprendizado de máquina utilizada. Na figura 7 apresenta uma das interfaces deste sistema.

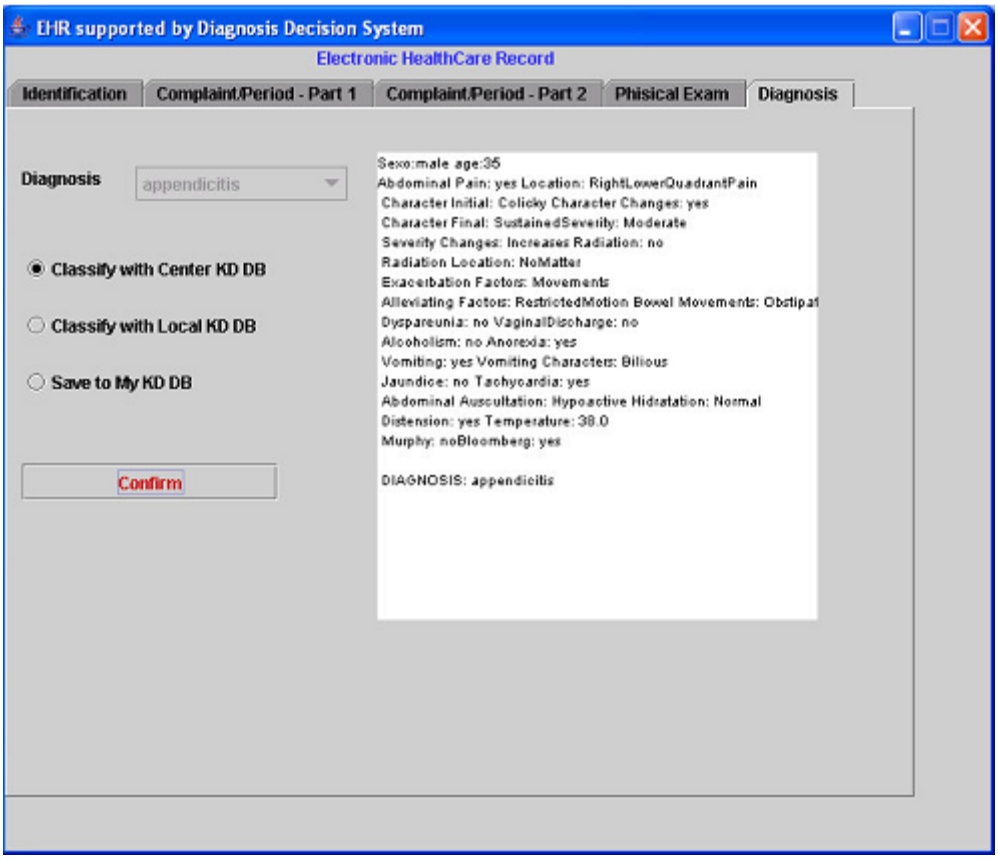


Figura7: Interface do EHR supported by Diagnosis Decision System.
FONTE: Pires et. AL., 2004.

Os trabalhos aqui apresentados foram divididos conforme os tópicos apropriados aos temas chaves de seu contexto e são resumidos no Quadro 1.

Quadro 1: Relação dos Trabalhos Relacionados.

	Projeto	Proposta
Sistema na Área da Saúde	(Borges e Carvalho, 2009)	Desenvolvimento de uma interface utilizando a tecnologia de reconhecimento de voz para o pronto-atendimento nas emergências das unidades hospitalares.
	(Morais et. al. 2012)	Desenvolvimento de um sistema de apoio à decisão descentralizado por meio do uso de dispositivos móveis implementado para diagnosticar Asma.
	(Pire et. al., 2004)	Desenvolvimento de um sistema de auxílio ao diagnóstico com o propósito de compartilhar casos e conhecimentos clínicos entre a comunidade médica na internet.

O principal diferencial do nosso projeto em relação aos demais projetos acima mencionados é a forma de interação do usuário com o sistema, uma vez que o paciente ou o médico utilitário do aplicativo irá interagir de forma direta com o mesmo, utilizando um microfone como dispositivo externo de entrada de dados, ou seja, o usuário irá “conversar com o sistema”, objetivando analisar se o paciente é um possível portador de sífilis. O ambiente de execução será do sistema será o Linux, por se tratar de uma plataforma de ambiente livre e por ser a plataforma compatível com a API de reconhecimento de voz. O presente sistema não será disponibilizado para funcionamento via internet, e sim um aplicativo que funcione em computadores individuais, não sendo necessária conexão web para o funcionamento do mesmo.

5.2 Resumo do Capítulo

Este capítulo abordou acerca de alguns trabalhos semelhantes ao desenvolvimento do projeto aqui desenvolvido, bem como suas principais características, funcionalidades e diferenças para com o sistema aqui proposto. O próximo capítulo irá abordar sobre os materiais e métodos utilizados para a implementação do projeto.

6. MATERIAIS E MÉTODOS

Neste capítulo é explanado sobre o procedimento metodológico que foi utilizado para o desenvolvimento deste trabalho como também as ferramentas empregadas durante cada fase do processo de desenvolvimento do sistema proposto.

6.1 Metodologia

Primeiramente para o desenvolvimento deste trabalho, foram realizadas algumas investigações sobre alguns temas de relevância fundamental para o desenvolvimento do

mesmo, os quais foram: engenharia de software, análise e projeto de sistema, mineração de dados, desenvolvimento de sistemas em Java 3D, síntese de reconhecimento de voz e por fim sobre modelo de processo.

Em segundo momento, o processo de desenvolvimento partiu para três fazes: Análise e Especificação de Requisitos, Projeto e Estruturação e por ultimo Implementação do Sistema, finalizando com a documentação. Conforme Sousa (apud CORREA, 1999), tais etapas foram interligadas através de um modelo de processo cíclico evolutivo, permitindo um desenvolvimento mais dinâmico e interativo.

Análise e especificação de requisitos: primeiramente, buscamos fazer o levantamento de requisitos e bibliografias que componha as informações necessárias na Secretária Municipal de Saúde de Marabá e CTA Marabá, para termos uma base de informações. Procurando compreender a necessidade do software, como a utilização do mesmo irá beneficiar na resolução do problema e definir quais tecnologias seriam adequadas para este desenvolvimento.

Etapa do projeto onde foi feita a identificação e análise dos propósitos e finalidades do Sistema, buscando como base de informações bibliografias e dados da Secretária Municipal de Saúde de Marabá e CTA Marabá. A partir de então foi possível entender os problemas e compreender a necessidades do mesmo, partindo assim, para a definição das tecnologias e ferramentas que seriam utilizadas no desenvolvimento do sistema. Para identificação de alguns dos requisitos foi realizada uma entrevista com o infectologista do instituto CTA de Marabá, com objetivo principal de adquirirmos informação sobre a doença infecto contagiosa (Sífilis) e o comportamento dos pacientes cometidos pela mesma bem como a postura do profissional no processo de consulta ao paciente.

A partir dessa atividade inicial foi possível realizar o levantamento e análise dos requisitos, que é segundo Guedes (2006) uma das primeiras fases de engenharia de um software, sendo assim um processo fundamental para compreensão do problema e determinação das funções a serem desempenhadas pelo Sistema. Propondo amenizar as deficiências deste processo apresentando as funcionalidades desejadas e de maior relevância para o Sistema.

Projeto e Estruturação: Nesta etapa foram preparados os diagramas UML, a entidade relacional entre os diversos objetos que constituem a estrutura de funcionamento do

software, dentre outros atributos do projeto do Sistema. A modelagem do Sistema foi feita utilizando a UML (*Unified Modeling Language* - Linguagem de Modelagem Unificada) através do desenvolvimento do diagrama de caso de uso, diagrama de classes e diagrama de objetos que tem por objetivo definir as características do sistema, tais como seus requisitos, seu comportamento e sua estrutura lógica, e assim fornecer múltiplas visões do sistema a ser modelo (GUEDES, 2006).

Implementação: A última etapa coube a implementação do sistema, ou seja, a produção do sistema com as suas funcionalidades já estabelecidas, reconhecimento de voz e o ambiente Java 3D codificado e importado do software Alice 3D na IDE NetBeans versão 6.8, no qual foram agrupadas todas as implementações.

6.2 Ferramentas

Para o processo de desenvolvimento do projeto do sistema proposto foi necessário utilizar um modelo da engenharia de software, o modelo escolhido foi, Processo Unificado por ser um modelo que abstraem muitas das melhores práticas empregadas no desenvolvimento de software atualmente, cujas principais são desenvolver software iterativamente, gerenciar requisitos, modelar graficamente um software, utilizar arquiteturas baseadas em componente, verificar a qualidade do mesmo e controlar suas alterações, (KRUCHTEN, 2003).

Durante a etapa de Análise e Especificação de Requisitos e no início da etapa de Projeto e Estruturação foi realizada toda a modelagem do sistema, utilizando a linguagem UML, que de acordo com Fowler (2005), é uma linguagem visual utilizada para modelar sistemas computacionais por meio do paradigma de orientação a objetos. Fazendo-se uso da ferramenta Visual Paradigm For UML Community Edition versão 10.2. Este aplicativo é gratuito, rico em recursos, permite criar os mais diversos diagramas UML e a importação e exportação para diversos formatos. (VISUAL PARADIGM, 2013).

Outra ferramenta utilizada para modelar um fluxograma de processo do sistema foi o *Biz Agi Process Modeler*, um instrumento para criação de mapas mentais, fluxogramas e diagramas em geral, que permite aos usuários constituírem graficamente vários processos e as relações existentes em cada etapa (SIGJUS, 2013).

Na etapa de desenvolvimento e construção do Sistema foi o software Alice 3D versão 3.1, um ambiente de programação tridimensional grátis e aberto, desenvolvido em Java e multi plataforma de fácil utilização no qual podem ser criadas animações e interações entre personagens e objetos lembrando muito jogos de vídeo game. Foi desenvolvido com o objetivo de minimizar as falhas denotadas no inicio da aprendizagem da programação (ALICE, 2010).

Outra ferramenta de desenvolvimento utilizada foi o NetBeans IDE versão 6.8, um ambiente integrado de desenvolvimento que permite a criação de programas tanto desktop como web, construídos por meio de linhas de código Java e bibliotecas (plug-ins) que diversas ferramentas podem ser combinadas criando um ambiente de desenvolvimento integrado, entretanto exige máquina robusta (NETBEANS, 2013).

Para gerenciamento do banco de dados utilizou-se a ferramenta MySQL, uma ferramenta de gerenciamento de banco de dados gratuito, eficiente e otimizado para múltiplas aplicações, que possui um conjunto de recursos muito práticos, é multi plataforma, é compatível com várias linguagens de programação (SANTOS; SILVA, 2013).

O phpMyAdmin é uma ferramenta que gerencia o MySQL desenvolvida em PHP,com ela foi possível realizar farias tarefas em relação ao banco de dados como: criar, remover, alterar, inserir, mudar configurações de usuários, executar códigos SQL etc. (PHPMYADMIN, 2013).

Para o reconhecimento de voz utilizou-se a API JLaPSAPI uma ferramenta de reconhecimento de voz, a qual, segue a especificação JSAPI(*Java Speech API*). De acordo com o Fala Brasil (2013), nela é utilizado reconhecimento de fala em português brasileiro utilizando a linguagem Java, suportando reconhecimento de texto livre.

Para a gravação da voz de utilização no sistema empregou-se a ferramenta padrão de gravação de voz do sistema operacional Ubuntu versão 10.04, a qual se precisou de uma ferramenta de conversão de áudio MP3, o *Sound Converter* versão 1.4.4, um aplicativo conversor de som simples para o ambiente GNOME,ele lê arquivos de som em qualquer formato suportado pelo GStreamer e gerá-los em *Ogg Vorbis*, FLAC, ou formato WAV, ou MP3 (SOUNDCONVERTER, 2013).

Outra ferramenta utilizada foi o *iReport* e *JasperReports* para a criação do relatório, o iReport é uma ferramenta para facilitar os testes e principalmente facilitar a

criação do layout dos relatórios. Mas quem gera de fato os relatórios em pdf, xls, html e outros formatos é o *JasperReports*. O *JasperReports* é um poderoso framework open-source escrito em Java para geração de relatórios. Ele permite gerar dinamicamente relatórios em diversos formatos; entre eles: PDF, HTML, XLS, CSV e XML. (JASPERSOFT CORPORAÇÃO, 2013).

6.3 Resumo do Capítulo

Este capítulo descreveu a metodologia e os materiais e utilizados para o desenvolvimento desta proposta.

O próximo capítulo irá tratar do estudo de caso do presente trabalho.

7. ESTUDO DE CASO

Em Marabá, de acordo com dados do Centro de Testagem e Aconselhamento, a sífilis é a DST mais comum, dentre as outras, nos pacientes da região e atingiu mais de 500 pessoas entre 2008 a 2012, ficando atrás somente das hepatites virais (SMS, 2013). Utilizamos a sífilis como doença objeto para desenvolver o sistema, uma vez que esta doença possui sintomas específicos sendo caracterizada por latências e sua evolução possui estágios e sintomas diferentes, e quando no seu estágio mais elevado pode prejudicar praticamente todos os órgãos do corpo.

Estudos revelam que há uma enorme deficiência de profissionais na área médica da rede pública na região Norte e Nordeste do país, segundo dados do CREMESP (2011), a população médica brasileira é mal distribuída pelo país. Os usuários do Sistema Único de Saúde (SUS) contam com quatro vês menos médicos que os do setor privado. Do total de médicos ativos no país, a região Sudeste tem 2,61 para cada 1.000 habitantes. Já o Norte do país tem menos de um médico (0,98) para cada 1.000 habitantes.

Diante desse quadro percebe-se uma necessidade cada vez maior de agilidade no processo de consulta ao paciente e diagnostico de doenças. O uso de ferramentas tecnológicas que auxiliam no processo de tomada de decisão para detectar doenças, se

torna um dispositivo importante e decisivo que possibilita uma agilidade na consulta ao paciente, facilitado o diagnóstico, reduzindo a espera e custos aos pacientes e melhorando a eficácia dos resultados.

Para tanto, uma boa solução para contribuir de forma satisfatória nesse processo de consulta ao paciente seria desenvolver um Sistema para auxílio na identificação de doenças como a sífilis (doença escolhida para compor o questionário da pré-consulta) que proporcione dinamização e agilidade no processo de detecção de doenças. Dessa forma, o Sistema será desenvolvido no ambiente NetBeans IDE 6.8 e deverá conter uma interface exportada do Alice 3D na qual haverá objetos como: uma sala, médico, mesa e cadeira demonstrando um ambiente de um consultório médico, a API para reconhecimento de voz (Coruja JLAPSAPI) e um banco de dados utilizando a técnica de arvores de decisão para compor o projeto e ser possível gerar o relatório.

O paciente irá interagir com o Sistema via comando de voz para responder ao questionário referente aos sintomas da doença. O Sistema deverá reconhecer as respostas e gerar um relatório com o resultado da pré-consulta.

7.1 Etapa de Análise e Especificação de Requisitos

A etapa de análise e especificação de requisitos foi um processo um tanto simples, uma vez que, a proposta do trabalho é de um sistema de pré-consulta médica. Tal Sistema possuirá um banco de dados onde será cadastrado o nome do paciente, CPF, idade e o pré-diagnóstico, possibilitando emitir um relatório do mesmo com a pesquisa do seu CPF. Na gramática estão estruturadas as respostas pré-definidas, as quais serão reconhecidas pela API de reconhecimento de voz ao comando de voz do paciente ao responder as perguntas referentes ao questionário, tais respostas serão comparadas as existentes na gramática, mineradas pela árvore de decisão, gravando assim, o pré-diagnóstico. No entanto foi preciso, ter um diálogo com um infectologista, (sem emissão de documento), apenas para um melhor entendimento na elaboração das perguntas do questionário. Com os dados obtidos construímos uma árvore de decisão com as perguntas do questionário (figura 8), as quais o paciente será submetido para se chegar ao objetivo esperado. No APÊNDICE A, contém a Gramática utilizada para o Reconhecimento de Voz dada à entrada de dados pela voz do paciente.

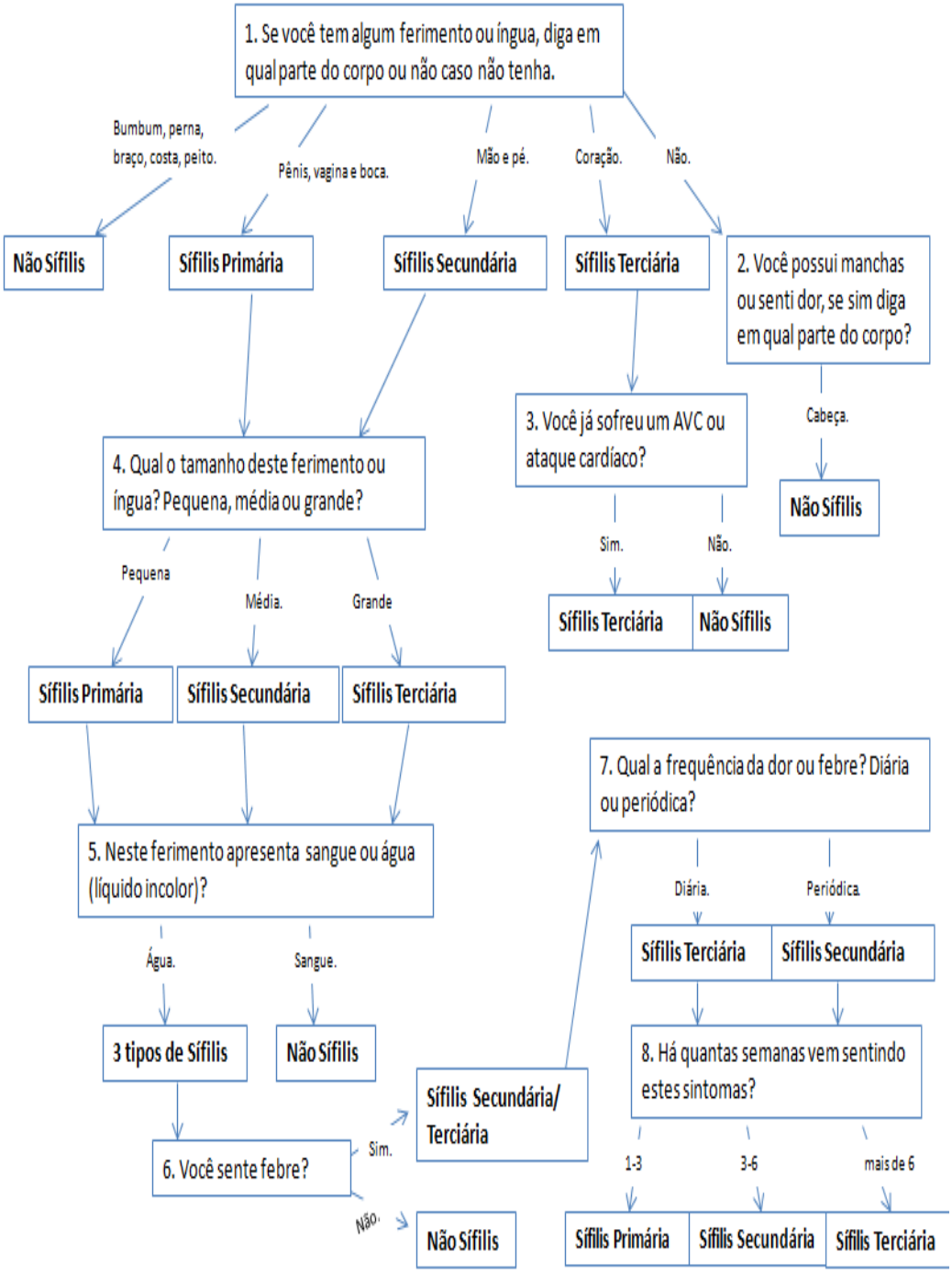


Figura 8: Árvore de decisão com o questionário.

Ainda na etapa de Análise e Especificação de Requisitos, foi produzido um Sumário Executivo, vide APÊNDICE B para entendimento da funcionalidade do sistema. Este documento foi produzido após uma análise e entendimento do objetivo do trabalho.

Assim, foi definido que haveria dois níveis de usuário, usuário administrador (responsável por ativar e gerenciar o sistema) usuário paciente (que ira se submeter à pré-consulta respondendo ao questionário via comando de voz). Após esta definição de usuários, foi desenvolvido um Documento de Requisitos, vide APÊNDICE C, que versa sobre a especificação dos requisitos funcionais. Os quadros 2 e 3 a seguir, lista os requisitos funcionais e não funcionais do sistema.

Quadro 2: Requisitos Funcionais do Sistema.

Código	**Requisitos Funcionais**
RF01	Cadastrar usuário paciente.
RF02	Solicitar pré-consulta ou pré-diagnóstico.
RF03	Emitir Relatório de pré-diagnostico.

Quadro 3: Requisitos não Funcionais do Sistema.

Código	**Requisitos não funcionais do Projeto**
RS01	Não será responsabilidade do sistema o diagnostico final da pré-consulta.
RS02	Não será competência do sistema o controle de quem usa o mesmo.
RS03	O sistema deverá rodar em plataforma Desktop.
RS04	O sistema deverá rodar no sistema operacional Ubuntu 10.04.
RS05	O Sistema deverá rodar com o mínimo de 56kps de banda.

Em seguida elaborou-se o diagrama de caso de uso de alto nível, ilustrado na Figura 9 visando proporcionar uma visão externa geral das funcionalidades que o sistema deverá oferecer aos seus usuários.

Sendo definido que o ADM será responsável por cadastrar paciente, solicitar a pré-diagnóstico e relatório de pré-diagnostico. O paciente após fornecer seus dados, ira responder ao questionário via comando de voz. Com os casos de uso de alto nível

definidos, cada caso de uso foi expandido no formato descritivo, a fim de demonstrar detalhadamente quais são os passos necessários de iteração com o sistema para que o usuário possa atingir seus objetivos. Tal documentação poderá ser encontrada no APÊNDICE D.

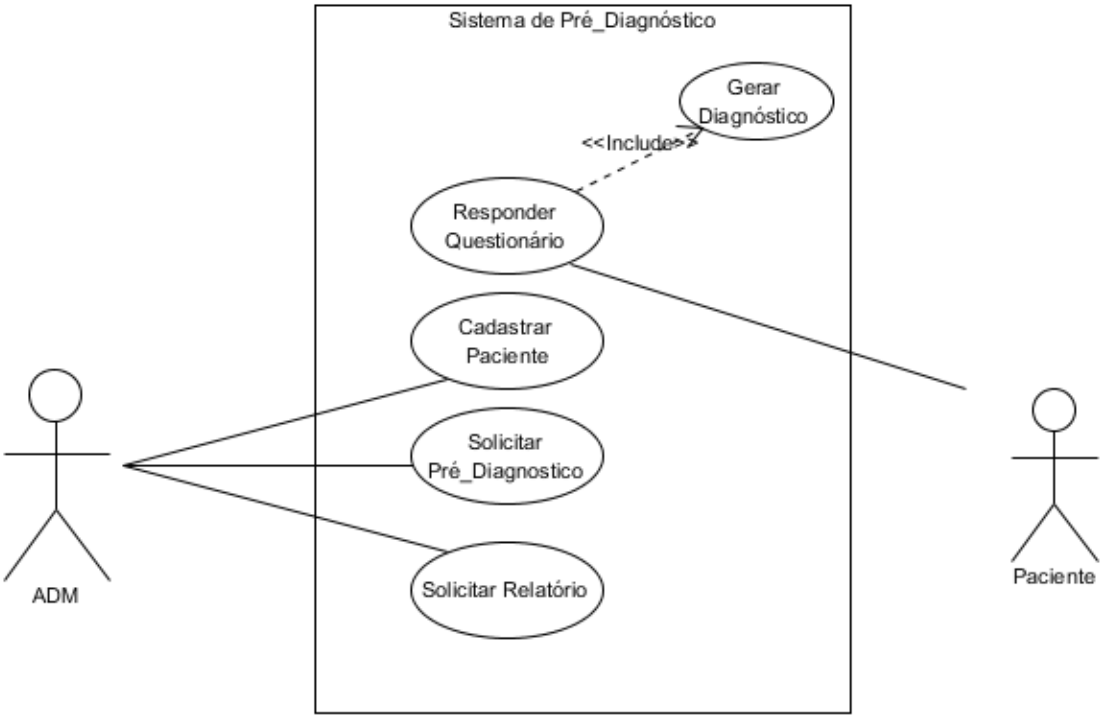


Figura 9: Diagrama de caso de uso de alto nível.

As atividades desenvolvidas nessa etapa do projeto foram necessárias para o detalhamento da modelagem do projeto, assim definido partimos para a fase de Projeto e Estruturação.

7.2 Etapa de Projeto e Estruturação

Inicialmente na etapa de Projeto e Estruturação, foi modelado o diagrama de classes do projeto, com o objetivo de permitir a visualização das classes que irão compor o sistema, que tem o intuito de projetar a maneira como a informação deve ser gerenciada no sistema proposto, apresentando seus atributos e operação, os quais representam as atividades a serem desenvolvidas no sistema.

A figura 10 representa o diagrama de classes do sistema, se observa os relacionamentos entre as entidades, como denota a classe Pre-Consulta com a classe Paciente e Relatório, que a classe Pre-Conulta, pré-diagnostica um ou mais Pacientes e que o mesmo é pré-diagnosticado por uma Pre-Consulta que possui um Relatório a qual pertence a uma Pre-Consulta.

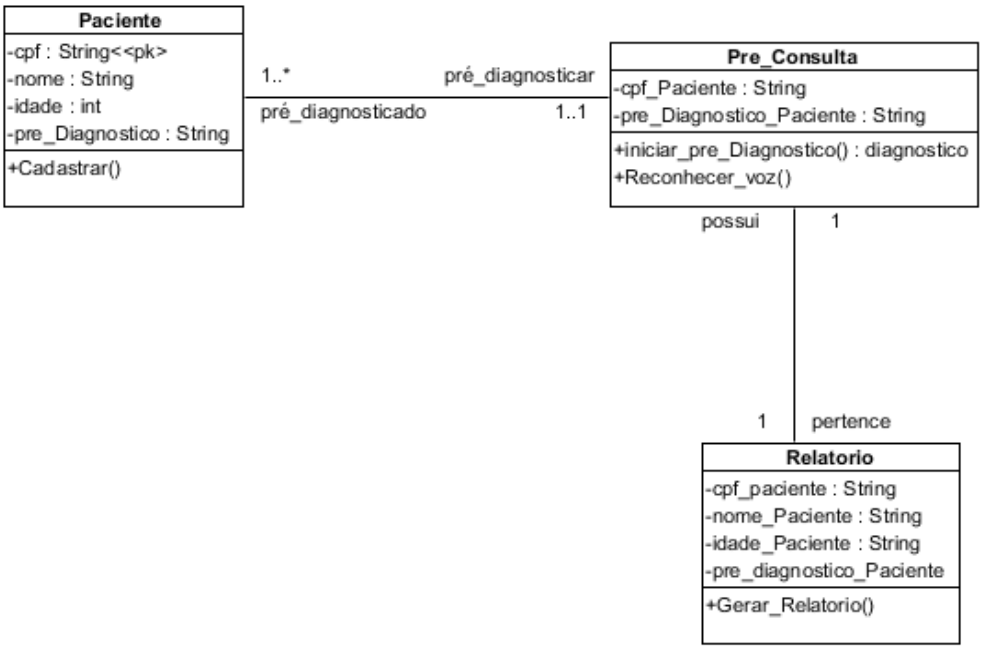


Figura 10: Diagrama de Classes do Sistema.

O desenvolvimento do Sistema proposto foi realizado através da modelagem do mesmo. A partir desta modelagem obteve a estrutura do sistema, bem como a definição do usuário do sistema e sua forma de acesso. Para o detalhamento da modelagem do projeto foram desenvolvidos dois diagramas de atividades sendo um em nível de ADM e outro em nível de Paciente, demonstrados na figura 11 e 12 as atividades dos mesmos. Nesta mesma etapa foi desenvolvido um fluxograma de processo para melhor entendimento, vide APÊNDICE E, sendo possível assim partir para a implementação do Sistema.

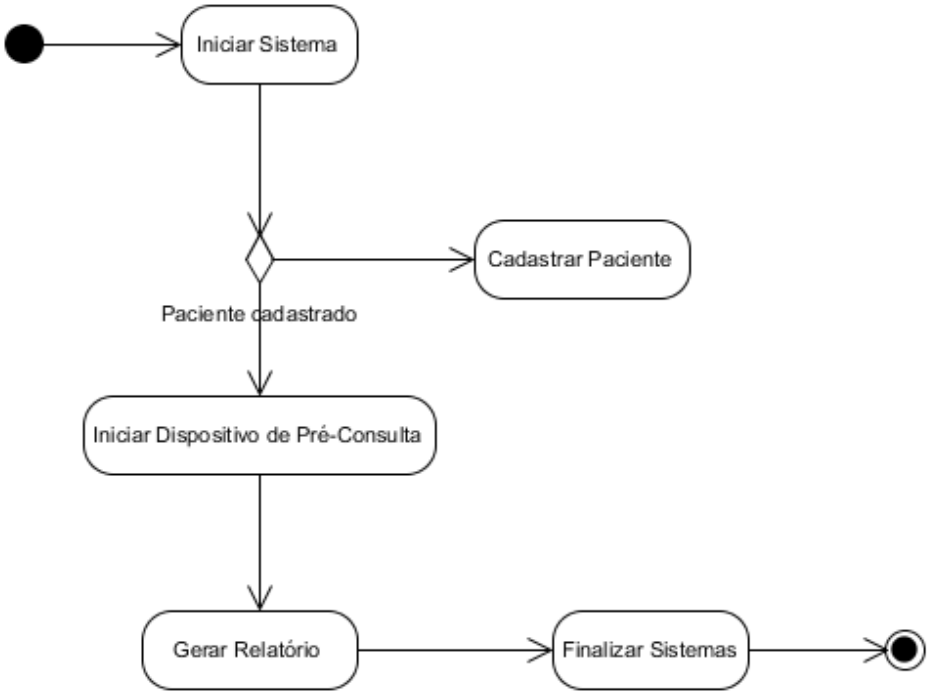


Figura 11: Diagrama de atividades em nível de administrador.

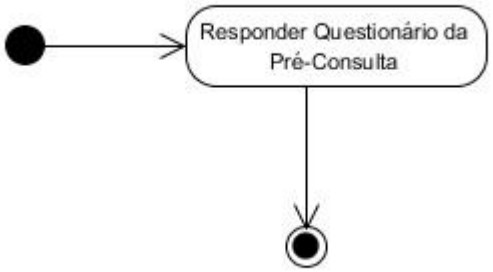


Figura 12: Diagrama de atividades em nível de paciente.

7.3 Etapa de Implementação

Na etapa de implementação, foi construída inicialmente a modelagem conceitual do banco de dados do sistema. Através desta modelagem foi criada a tabela do banco de dados, com suas configurações, sendo manipulada no phpMyAdmin MySQL. A figura 13 demonstra o modelo conceitual do banco de dados do protótipo do sistema. Em

seguida na figura 14 apresenta a interface de implementação do objeto 3D desenvolvido no Alice 3D, logo após mostra as interfaces desenvolvidas do Sistema. Essas interfaces especificam as principais telas do Sistema disponíveis para os usuários. Na vide APÊNDICE F encontra-se os códigos fonte das principais classes do projeto.

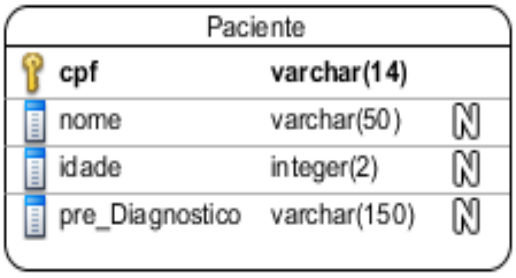


Figura 13: Modelagem do Banco de Dados.

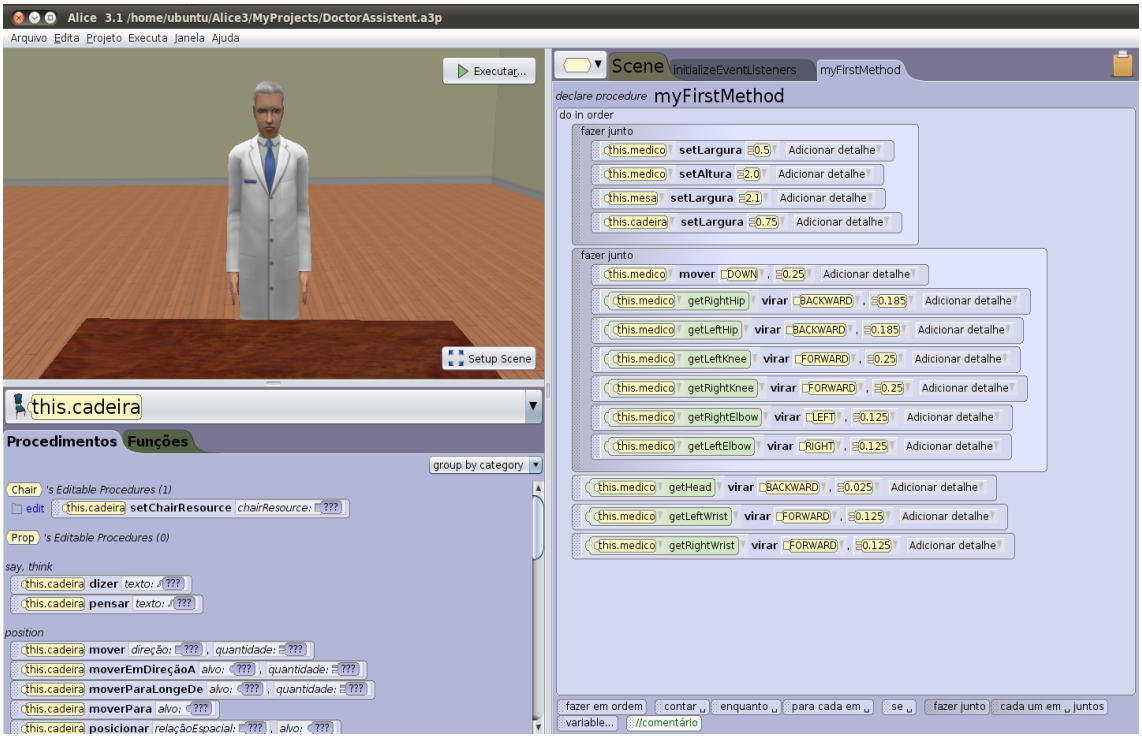


Figura 14: Tela Principal do Alice 3D.

Para ter acesso ao Sistema e assim ao questionário, o usuário administrador deverá abrir o Sistema, como mostra a figura 15, realizar o cadastro do usuário paciente, que fornecerá seu Nome, CPF e Idade (figura 16). Após realizar o cadastro o usuário

administrador entrar com o CPF do usuário paciente, o qual poderá dar inicio ao pré-diagnostico, como ilustra a figura 17.

Figura 15: Carregando o Sistema de Pré-Diagnóstico.

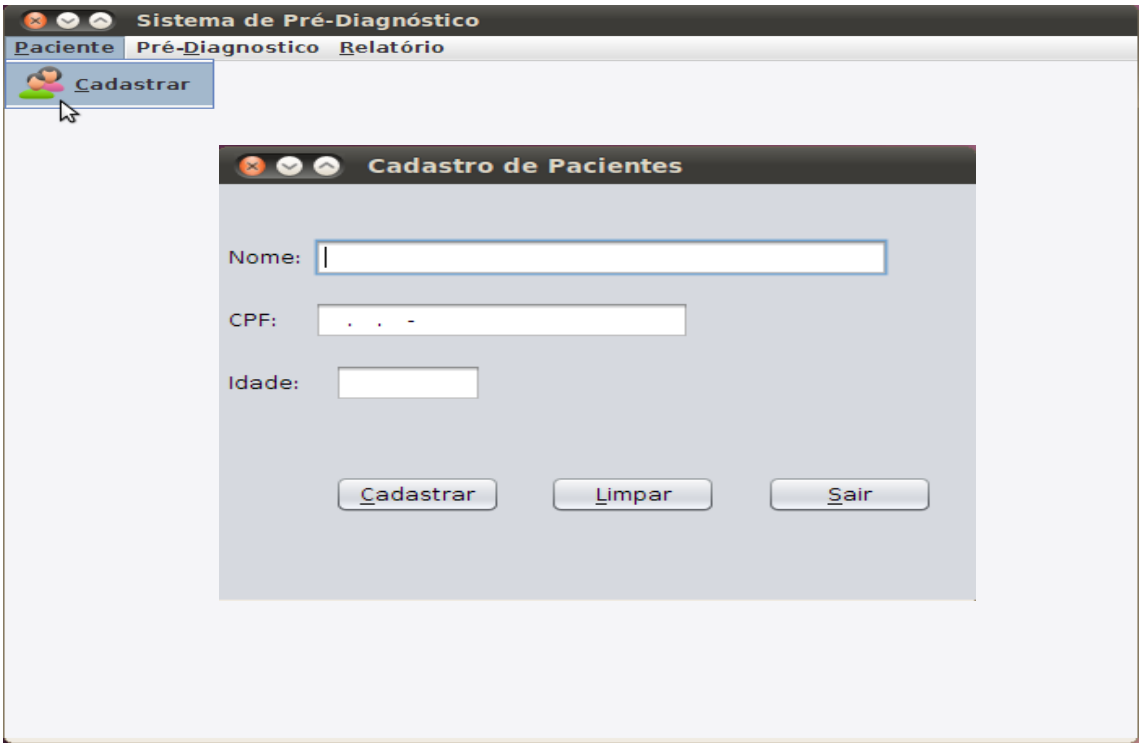


Figura 16: Tela Inicial do Cadastro do Paciente.

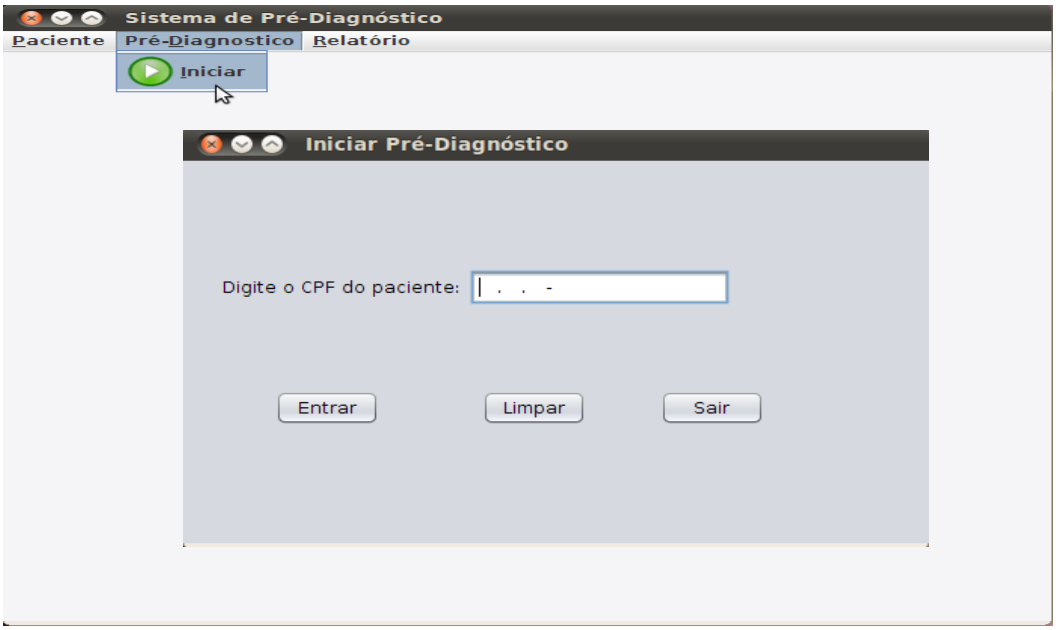


Figura 17: Tela para Iniciar o Pré-Diagnóstico

Esta é a tela principal de interação do usuário com o objeto 3D, onde o objeto ficará “ouvindo” as entradas de dados via voz do paciente, iniciando assim o pré-diagnóstico, conforme pode ser visto na Figura 18.

Figura 18: Objeto 3D de Interação Via Voz.

A figura 19 ilustra a tela de emissão de relatório, que pode ser gerado pelo usuário administrador ao fim da pré-consulta clicando em relatório e em seguida no menu “GERAR”da tela principal. Clicando em Gerar será inseri o CPF do usuário e clica em “OK”, gerando assim o relatório que poderá ser impresso e destinado ao médico especialista, como mostra a Figura 20.

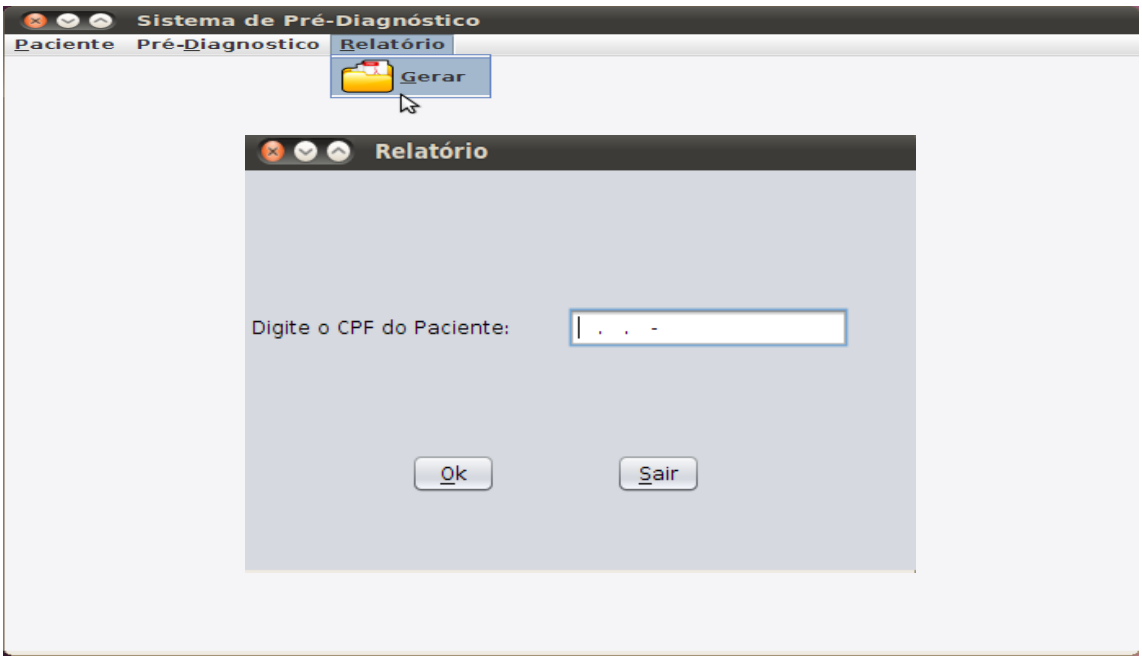


Figura 19: Tela Gerar Relatório.

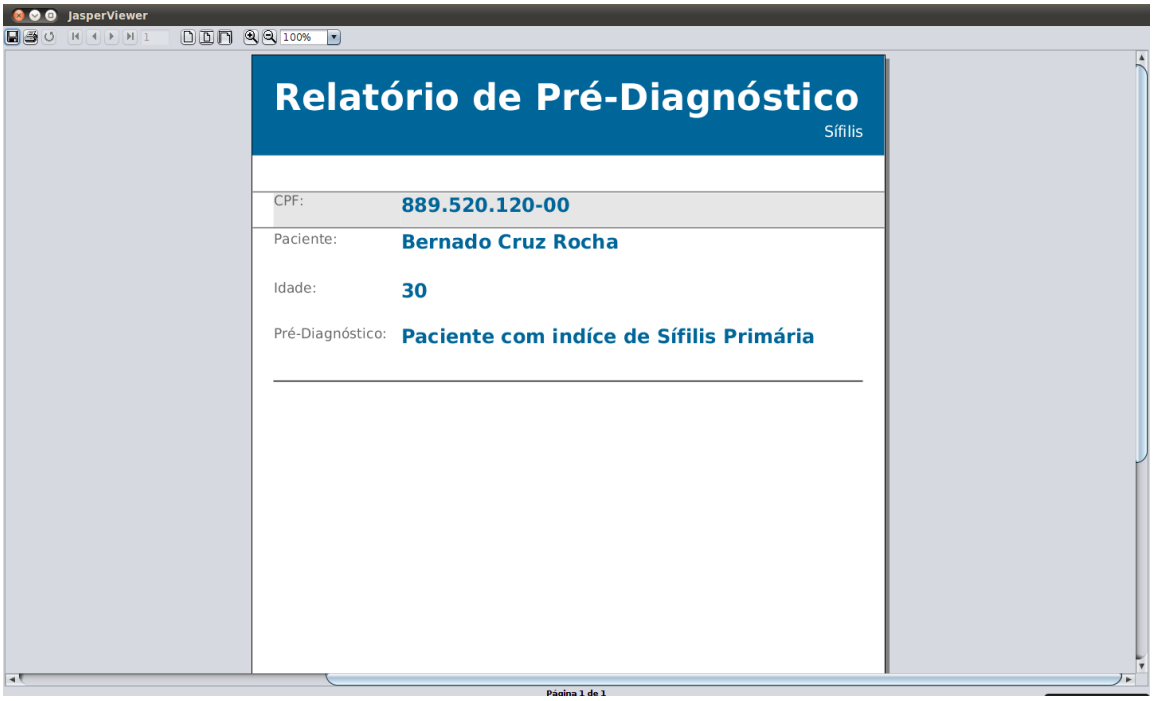


Figura 20: Tela de Relatório.

7.4 Resumo do Capítulo

Este capítulo descreveu o estudo de caso sobre a presente proposta, apresentando a modelagem e implementação, bem como a detalhamento dessas atividades desenvolvidas.

O próximo capítulo irá tratar da conclusão do presente trabalho, descrevendo resumidamente o trabalho desenvolvido e resultados obtidos.

8. CONCIDERAÇÕES FINAIS

A utilização do método de mineração de dados mostra a ideia de dinamismo de uma base de dados que se comunica com uma máquina local que gera árvores de decisões e classifica um usuário como possível infectado ou não. Durante a fase de desenvolvimento, podemos verificar o bom desempenho do uso dessa metodologia de uso de uma ferramenta controlada por voz, uma vez que utilizá-la, simplifica o processo de interação do usuário com o sistema. Isso traz a ideia da importância da aplicação das técnicas de mineração de dados para o uso de desenvolvimento de sistemas interativos.

As técnicas de mineração de dados, em especial árvores de decisão e regras de classificação, foram de suma importância para elaboração e conclusão da base de dados do agente 3D, que utiliza técnicas de reconhecimento de voz como ponte de comunicação.

Nesse trajeto, houve muitas dificuldades encontradas para o desenvolvimento do sistema como: adolescer uma árvore de decisão que se adequasse aos objetivos propostos, principalmente no que se refere às ferramentas para construção do sistema, pois foi difícil encontrar ferramentas especificas para construção das respectivas técnicas. A utilização da tecnologia JLAPSAPI para reconhecimento de voz na plataforma Linux que não tínhamos familiaridade com as mesmas. Porém, apresentaram bons resultados, tal como a programação em linguagem JAVA para o desenvolvimento da árvore de decisão e do agente 3D, tendo em vista que a quantidade de ferramentas específica para essas respectivas técnicas é escassa e as poucas encontradas nas pesquisas bibliográficas e levantamento de requisitos possuem literatura em outro idioma.

Portanto, conclui-se que é perfeitamente viável a utilização de uma técnica de mineração de dados em sintonia com o JAVA ALICE 3D e o JLAPSAPI na criação de um sistema de interação 3D para auxílio na detecção de possíveis infectados por sífilis.

8.1 Trabalhos Futuros

Para trabalhos futuros pretendem-se algumas funções como:

- Alterar dados cadastrais do paciente
- Remover e atualizar os seus dados

- Adicionar mais campos cadastrais como endereço e telefone
- Inserir outros tipos de doenças e
- Realização dos testes de usabilidade e aceitação junto à sociedade.

9. REFERÊNCIAS BIBLIOGRÁFICAS

ALICE, 2010. An Educational Software that teaches students Computer Programming in a 3D Environment. Carnegie Mellon University, Pittsburgh, EUA. Disponível em: http://www.alice.org/. Acesso em: 10 de Jul. 2013.

AMERICANAS. Disponível em: <http://www.americanas.com.br>. Acesso em: 01 Abr. 2013.

AMO, Sandra de. Técnicas de Mineração de Dados, Universidade Federal de Uberlândia, 2003. Disponível em <http://www.lsi.ufu.br/documentos/publicacoes/ano/2004/JAI-cap5.pdf>. Acesso em: 01 Mar. 2013.

AZEVEDO, Eduardo; CONCI, Aura. Computação Gráfica: geração de imagens. Rio de Janeiro: Campus/Elsevier, 8º reimpressão, 2003.

BALLARD, Chuck. HERREMAN, Dirk. Data Modeling Techniques for Data Warehousing.IBM, InternationalTechnicalSupportOrganization, 1998.

BARBOSA, Simone D. Junqueira; SILVA, Bruno Santana da. Interação Humano – Computador. Rio de Janeiro: Elsevier, 2010.

BARROS, Edson de Almeida Rego. ALICE BRASIL, 2010. In: ALICE BRASIL.São Paulo.Anais. 2010

BICHO, Alessandro L., et al. Programação Gráfica 3D com OpenGL, Open Inventor e Java 3D. Revista Eletrônica de Iniciação Científica (REIC). Rio Grande do Sul – RS, vol. II, n. I. Março 2002. .

BORGES, Sidney Reis; CARVALHO, Vithor Tibiriçá. Reconhecimento de Voz Aplicada a Interface de Sistemas Emergenciais Hospitalares. Salvador, 2009. 86 f. Monografia (Graduação em Informática) Curso de Bacharelado em Informática, Universidade Católica de Salvador, Salvador, 2009.

CARD, S.; MORAN, T. P.; NEWELL, A. The Psychology of Human-Computer Interaction.Hill-sdale, NJ: Lawrence Erlbaum Associates, 1983.

CASTANHEIRA, Luciana Gomes. Aplicação de Técnicas de Mineração de Dados em Problemas de Classificação de Padrões. Belo Horizonte, 2008. 95 f. Dissertação (Mestrado em Engenharia Elétrica) Departamento de Engenharia Elétrica, Universidade Federal de Minas gerais, Belo Horizonte, 2008.

CORREA, Alexandre L. Uma Arquitetura de apoio para Análise de Modelos Orientados a Objetos. Rio de Janeiro, 1999. Dissertação de Mestrado, Universidade Federal do Rio de Janeiro / COPPE, Rio de Janeiro, 1999.

CRATOCHVIL, Anda. Data mining techniques in supporting decision making.Master Thesis - UniversiteitLeiden, Leiden, 1999.

CRATOCHVIL, Anda. Data mining techniques in supporting decision-making. Master Thesis - Universiteit Leiden, Leiden, 1999.

CREMESP, Conselho Regional de Medicina do Estado de São Paulo; Demografia Médica. http://www.cremesp.org.bracesso em 03 de dezembro de 2013.

CREPALDI, Paola Guarisso et al. Um Estudo Sobre a Árvore de Decisão e Sua Importância na Habilidade de Aprendizado. Londrina: INESUL, 2011. Disponível em: <http://www.inesul.edu.br/revista/arquivos/arq-idvol_15_1320100263.pdf>. Acesso em: 26 Fev. 2013.

DANTAS, Josué; et al. Um sistema para Melhorar a Usabilidade de um Gerenciador de Correio Eletrônico Baseado em Reconhecimento de Fala. In: Proceedings of the 8th Brazilian SYMPOSIUM IN INFORMATION AND HUMAN LANGUAGE TECHNOLOGY, 2011, Cuiabá-MT. Sociedade Brasileira de Computação. Cuiabá, 2011. p. 107–114. Disponível em: <http://www.lbd.dcc.ufmg.br:8080/colecoes/stil/2011/0012.pdf >. Acesso em: 17 Mar. 2013.

DIAS, Maria Madalena. Parâmetros na Escolha de Técnicas e Ferramentas de Mineração de Dados, Departamento de Informática, Universidade Estadual de Maringá, 2002. Disponível em:<http://eduem.uem.br/ojs/index.php/ActaSciTechnol/article/download/2549/1569>. Acesso em: 28 Fev. 2013.

FALA BRASIL - LAPS. Laboratório de Processamento de Sinais. UFPA. Disponível em: <http://www.laps.ufpa.br/falabrasil/index.php>. Acesso em 20 Fev. 2013.

FAYYAD, Usama, Piatetsky-Shapiro, Gregory, and Smyth, Padhraic. From Data Mining to Knowledge Discovery in Databases, American Association for Artificial Intelligence, 1996. Disponível em: < http://www.kdnuggets.com/gpspubs/aimag-kdd-overview-1996-Fayyad.pdf >. Acesso em: 28 Fev. 2013.

FOWLER, Martin. UML essencial: um breve guia para a linguagem-padrão de modelagem de objeto / Martin Flowler; trad. João Tortello. 3. ed. Porto Alegre: Bookman, 2005.

GAMA, J. Árvores de Decisão, 2000. Disponível em: <http:/www.liacc.up.pt/~jgama/Mestrado/ECD1/Arvores.html>. Acesso em: 01 Mar. 2013.

GARCIA, Simone Carboni. O Uso de Árvores de Decisão na Descoberta de Conhecimento na Área da Saúde. Porto Alegre, 2003. 88f. Dissertação (Mestrado em Ciência da Computação) Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2003.

GIZMAG. Disponível em: < http://www.gizmag.com>. Acesso em: 01 Abr. 2013.

GOEBEl, Michael; Gruenwald, Le. Asurvey of data mining ano know ledge discovery software tools. Departamento de Ciência da Computação da Universidade de Auckland, Nova Zelândia, 1999. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.24 >. Acesso em: 01 Mar. 2013.

GUEDES, Gilleanes T. A. UML: uma abordagem prática / Gilleanes T. A. Guedes. 2. ed. São Paulo: Novatec Editora, 2006.

HARRISON, T.H. Intranet data warehouse. São Paulo: Editora Berkeley Brasil, 1998.

HIX, D.; HARTSON, H. Developing User Interfaces: Ensuring Usability Through Product and Process. New York, NY: John Wiley e Sons, 1993.

JASPERSFT – CORPORAÇÃO. A Ferramenta de Desenvolvimento Relatório de JasperReports e JasperReports Servidor. Disponível em: <http://community.jaspersoft.com/project/ireport-designer>. Acesso em 14 Dez. 2013.

Jlapsapi. Disponível em: < http://www.laps.ufpa.br/falabrasil/jlapsapi/>. Acesso em: 20 Fev. 2013.

JSAPI. Disponível em: <http://www.java.sun.com/products/java-media/speech/>. Acesso em: 18 Mar. 2013.

JULIUS. The Julius book rev.4.1.2.Akinobu LEE, part 4.1.2 - ENG. Disponível em: <http://jaist.dl.sourceforge.jp/julius/37581/Juliusbook-part-4.1.2-en.pdf>. Acesso em: 17 Mar. de 2013.

KALUNGA. Disponível em: <http://www.kalunga.com.br>. Acesso em: 01 Abr. 2013.

KRUCHTEN, Philipe. Introdução ao RUP – Rational Unified Process. 2. Ed. Rio de Janeiro: Editora Ciência Moderna Ltda. 2003.

LOUZADA, JailtonAlkimin. H. F. Olson and H.Belar, Phonetic Typewriter, J. Acoust. Soc. Am., 28(6): 1072-1081, 1956.

LOUZADA, JailtonAlkimin. J. FERGUSON, Ed., Hidden Markov Models for Speech, IDA, Princeton, NJ, 1980.

MANNILA, Heikki. Methodsandproblems in data mining.Departamento de Ciência da Computação da Universidade de Helsinki, 6 Delphi, 1997. Proceedings… Delphi: Lecture Notes in Computer Science, 1997. p. 41-55.

MANSSOUR, Isabel Harb; Introdução a Java 3D.Disponível em: <http://www.inf.pucrs.br/manssour/Publicacoes/TutorialSib2003.pdf>. Acesso em: 06 Mar. 2013.

MICROSOFT, Speech API 5.3. Text Grammar Format. Disponível em: <http://msdn.microsoft.com/pt-pt/library/ms723632%28v=VS.85%29.aspx>. Acesso em: 18 Mar. de 2013.

MORAIS, Dyego, et al.; Sistema Móvel de Apoio a Decisão Médica Aplicado ao Diagnóstico de Asma – InteliMED. In: SIMPÓSIO BRASILEIRO DE SISTEMAS DE INFORMAÇÃO, 21, 2009, São Paulo. Anais. São Paulo: EACH-USP, 2012. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/sbsi/2012/0051.pdf>. Acesso em 20 Nov. 2013.

MULTIGNER, Gilles. Sociedad interactiva o sociedad programada? In: FUNDESCO (org.). Apuntes de La sociedadinteractiva. Cuenca, Espanha UIMP, 1994. p. 421-428. Disponível em: <http://dialnet.unirioja.es/servlet/articulo?codigo=1227351>. Acesso em: 17 Mar. 2013.

NETBEANS - DOWNLOAD. Kit para Desenvolvimento Java e outras Plataformas. 2013. Disponível em:<http://netbeans.org/downloads/6.8>. Acesso em: 15 Dez. 2013.

PALMER, I. Essential Java 3D Fast - Developing 3D GraphicsApplications in Java.1º ed. SPRINGER VERLAG NY, 2001, 279p.

PAVARINI, Larissa. Estudo e Implementação do Método Massa-Mola para Deformação em Ambientes Virtuais de Treinamento Médico Usando a API Java 3D. Marília, 2006. 146 f. Dissertação (Mestrado em Ciência da Computação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília-SP, 2006.

PHPMYADMIN. Trazendo o MySQL para a web. Disponível em: <http://www.phpmyadmin.net>. Acesso em 15 Nov. 2013.

PIRES, D. F., et al.; Uma Arquitetura para Suporte ao Compartilhamento do Conhecimento Clínico em Sistemas PEP Integrados a Sistemas de Auxílio ao Diagnóstico. In: Anais do IX Congresso Brasileiro de Informática em Saúde. Ribeirão Preto-SP, 2004. Disponível em <http://www.sbis.org.br/cbis9/arquivos/593.PDF> acesso em 20 Nov. 2013.

POZZER, Cesar Tadeu. Aprendizado por Árvores de Decisão, Universidade Federal de Santa Maria, Rio Grande do Sul, 2006. Disponível em :< http://www.ngd.ufsc.br/files/2012/04/mn_DecisionTrees.pdf> Acesso em: 28 Fev. 2013.

SABBATINI, R. M. E. Realidade Virtual em Medicina. Revista Informática Médica, v. 2, n. 2, 1993. Disponível em: <http://www.informaticamedica.org.br>. Acesso em: 01 Abr. 2013.

SAMPAIO NETO, Nelson Cruz. Desenvolvimento de Aplicativos Usando Reconhecimento e Síntese de Voz. Universidade Federal do Pará – UFPA, 2006.

SANTOS, Jonis Nogueira dos; SILVA José Adelar Souza da.SGBD MySQL. Taquara. 2013. Disponível em: https://fit.faccat.br/~jonis/Artigo_mySQL.pdfAcesso em 15 Nov. 2013.

SMS. Secretaria Municipal de Saúde de Marabá. Departamento de DST e Epidemias. Marabá-PA, 2013.

SELMAN, Daniel. Java 3D Programming: A guide to key concepts and effective techniques. Nova York: Manning Publications, 2002.

SIGJUS – Sistemas Integrados de gestão da Justiça Federal. Disponível em: <http://portal.cjf.jus.br/sigjus/documentacao/ferramenta-bizagi>. Acesso em 14 Dez. 2013.

SILBERSCHATS, Abraham; KORTH, Henry F.; SUDARSHAN, S. Sistema de banco de dados; tradução de Daniel Vieira. 3.ed. Rio de janeiro: Elsevier, 2006.

SILVA, Marcos. O que é Interatividade. Boletim Técnico do Senac. Volume 24 – Número 2 –Maio/Agosto 1998. Disponível em: <http://http://www.senac.br/BTS/242/boltec242d.htm>. Acesso em: 18 Mar. 2013.

SILVA, P., Sistemas de Reconhecimento de Voz para o Português brasileiro utilizando os Corpora Spoltech e OGI-22, Trabalho de conclusão de curso, Universidade Federal do Pará, Instituto de Tecnologia, 2008.

SOMA, Nei Yoshihiroet al. Data Mining Como Suporte À Tomada De Decisões - Uma Aplicação No Diagnóstico Médico. ITA Divisão da Ciência da Computação, São José dos Campos, São Paulo, 2004

SOUND CONVERTER – Convertor de Som simples para ambiente GNOME. Disponível em: <http:// apps.ubuntu.com/cat/applications/quantal/soundconverter>. Acesso em: 14 Dez. 2013.

SOURCEFORGE. Freetts 1.2. 2001. Disponível em: <http://freetts.sourceforge.net/>. Acesso em: 20 Fev. 2013.

SOUSA, Washington C. Braga de. PLARVICE 3D – Plataforma em Realidade Virtual não-Imersiva para Criação de Estruturas Tridimensionais. Belém, 2008. 128 f. Trabalho de Conclusão de Curso (Graduação em Engenheiro Eletricista) de Engenharia Elétrica, Universidade Federal do Pará, Belém, 2008.

SOUZA JUNIOR, Márcio de. Desenvolvimento De Aplicações Gráficas Com O Java 3D. Divinópolis, 2011. 43 f. Trabalho de Conclusão de Curdo (Graduação em Bacharel em Sistemas de Informação) de Sistemas de Informação, Faculdade Pitágoras de Divinópolis, 2011.

SUN, Microsystem. The Java 3D ™ API Specification, 2000. Disponível em: <http:// http://docs.oracle.com/cd/E19869-01/index.html>. Acesso em: 20 Mar. 2013.

TCHEMRA, A. H; GRINKRAUT, M.L. Ensino da Matemática com o software ALICE, 2010. In: ALICE BRASIL. São Paulo. Anais. 2010, p. 17-23.

VILLE, Barry de. Decision Trees For Business Intelligence and Data Mining. 1. ed. North Carolina: SAS Publishing, 2006.

VISUAL PARADIGM FOR UML. Visual Paradigm for UML Community Edition 1.0 revisão. Disponível em:<http://visual-paradigm-for-uml-community edition.visualparadigm.blueprograms.com>Acesso em: 15 Nov. 2013.

VIVEROS, Maria S. et al. Applying data mining techniques to a health insurance information system.IBM ResearchDivision22. 1996, Bombay. Proceedings…Bombay: IIT Bombay, 1996. p. 286-295.

YNOGUTI,Carlos Alberto. Reconhecimento de fala Continua Usando Modelos Ocultos de Markov. Unicamp, 1999.

APÊNDICE A – Gramática Utilizada para o Reconhecimento de Voz.

Gramática para o Reconhecimento de Voz

```
grammarDiag;



public<comando> = sim | não | iniciar | sair
| certo | sangue | água | pequena
| média | grande | diária | periódica | uma
| duas | três | quatro | cinco | seis
| sete | oito| nove | dez | onze | doze;



public<localCorpo> = perna | braço | costa
| peito | coração | mão | pé | pênis | vagina
| boca | verilha | veria | viria | bumda | bumbum
| cabeça;
```

APÊNDICE B – Sumário Executivo

Para um bom funcionamento do sistema fez-se necessário estabelecer níveis de usuários e acessos que foram classificados como:

1 – Nível de usuário para administrador (ADM)

Será disponibilizada a tela principal do sistema para o usuário ADM, o qual fará acesso logo após iniciar o sistema, não sendo necessário LOGIN e SENHA, na tela principal o usuário ADM poderá cadastrar paciente, iniciar o pré-diagnostico e gerar o relatório. O ADM cadastra o paciente com Nome, CPF e Idade, poderá realizar o pré-diagnostico a qualquer tempo e quantas vezes forem necessárias, e ao termino poderá solicitar ao Sistema o relatório logo após a pré-consulta.

2 – Nível de usuário paciente pré-diagnostico

Neste nível o usuário paciente após fornecer seus dados pessoais para o cadastro, ira responder ao questionário via comando de voz, logo após o ADM iniciar o pré-diagnostico. Ressaltando que o contato do usuário com o sistema será via comando de voz, a interação se iniciara logo após o ADM inserir o CPF do paciente e iniciar o pré-diagnostico.

APÊNDICE C – Documento de Requisitos

Descrição dos Requisitos Funcionais do Sistema

[RF01] - Gerenciar usuário administrador	
Descrição:	O gerenciamento do ADM será composto pelo cadastramento do paciente, iniciar o pré-diagnostico e emitir o relatório. O sistema possuíra um frame pra cadastrar o paciente com nome, idade e CPF, onde será feito o cadastro de cada paciente antes da pré-consulta.
Fontes:	Administrador do sistema.
Usuário:	O usuário ADM, o cadastro do paciente também poderá ser realizado pelo próprio, uma vez que, não é necessário cadastrar LOGIN e SENHA.
Informações de Entrada:	Para realizar o cadastro, iniciar o pré-diagnostico e emitir relatório serão necessárias as seguintes informações: Dados pessoais do paciente (Nome, Idade, CPF), clicar no menu o botão Iniciar e inserir o CPF do paciente para iniciar e ao fim clicar no menu o botão Gerar para imprimir.
Informações de Saída:	Após a conclusão de cada operação o sistema retornará para o estado inicial.
Restrições Lógicas:	- O sistema não possui restrição lógica. - Não sendo responsabilidade do mesmo o controle de quem acessa o as funcionalidades do Sistema.
Restrições Tecnológicas:	- Para ter acesso a este requisito é necessário que apenas o sistema esteja funcionando

[RF02] – Gerenciar Paciente.	
Descrição:	O paciente terá acesso ao Sistema através da voz respondendo ao questionário, o Sistema reconhece as resposta do usuário.
Fontes:	Paciente
Usuário:	O usuário que deseja se consultar e se submeter antes ao questionário de pré-diagnostico
Informações de Entrada:	Para se submeter ao questionário, basta esperar a inicialização da pré-consulta após o seu cadastro.
Informações de Saída:	Após concluir o questionário, será solicitado um relatório com os dados do paciente e o pré-diagnostico do estagio da doença. Finalizando assim a pré-consulta.
Restrições Lógicas:	- Não há restrição para essa atividade.
Restrições Tecnológicas:	- Para ter acesso a este requisito é necessário que o programa esteja rodando corretamente o paciente realize seu cadastro.

[RF03] – Salvar dados do paciente	
Descrição:	O paciente deverá fornecer seus dados pessoais e esperar a iniciar o pré-diagnostico e começar a interação respondendo ao questionário via comando de voz.
Fontes:	O usuário Administrador
Usuário:	O paciente que deseja se consultar, e antes se submeter a uma pré-consulta.
Informações de Entrada:	Para se submeter à pré-consulta o paciente deverá fornecer seus dados pessoais (Nome, Idade e CPF) para que seja identificado no relatório de pré-diagnostico.
Informações de Saída:	Após a conclusão do questionário o Sistema estará apto a emitir o relatório com os dados do paciente e pré-diagnostico.
Restrições Lógicas:	- O paciente não poderá fazer alterações nas respostas dadas pra determinada pergunta.
Restrições Tecnológicas:	- Para ter acesso a este requisito é necessário que o programa esteja rodando corretamente e que o paciente forneça seus dados.

[RF04] - Emitir Relatório de pré-diagnostico	
Descrição:	Este relatório visa exibir informações relacionadas às respostas do paciente ao questionário.
Fontes:	O próprio Sistema emitirá logo após a solicitação do ADM.
Usuário:	Os pacientes.
Informações de Entrada:	Para obter o relatório é preciso que seja respondido a todas as perguntas do questionário.
Informações de Saída:	Ao final da operação, é emitido o relatório com o pré-diagnostico do índice da doença.
Restrições Lógicas:	- o paciente deve responder ao questionário.
Restrições Tecnológicas:	**-** Para ter acesso a este requisito é necessário que, além de estar com o programa rodando corretamente, responder as perguntas do questionário.

APÊNDICE D – Casos de Uso Abordado de Forma Descritiva

UC 01 – Caso de Uso Responder Questionário.

Ator Principal: paciente (que deseja se submeter à pré-consulta).

Descrição: Este caso de uso descreve as etapas necessárias para que o usuário possa responder ao questionário de pré-diagnostico.

Pré-condições:

1. O usuário deve inicialmente fornecer seus dados pessoais para assim começar o questionário.

Pós-condições:

O usuário deve responder a todo o questionário, para assim, o Sistema traçar o pré-diagnostico do mesmo.

Fluxo Evento Principal:

1. Informar dados pessoais e esperar que o ADM solicite o pré-diagnostico.
2. O usuário informa as resposta a todas as perguntas do questionário via comando voz, para obter o relatório de pré-diagnostico ao finalizar o mesmo.
3. O Sistema ao registra a resposta a ultima pergunta. E está pronto para emitir o relatório do paciente.

UC 02 – Caso de Uso Cadastrar Paciente.

Ator Principal: ADM.

Descrição: Este caso de uso descreve as etapas necessárias para que o usuário possa cadastrar um paciente.

Pré-condições:

O usuário deve estar com o programa inicializado.

Pós-condições:

Deve esta na tela de cadastro e obter os dados do paciente.

Fluxo Evento Principal:

1. O usuário ADM insere os dados pessoais do paciente (Nome, Idade, CPF) nos campos necessários para completar seu cadastro.
2. O Sistema registra e finaliza podendo assim dar início ao questionário.

UC 03 – Caso de Uso Solicitar Pré-Diagnostico.

Ator Principal: ADM.

Descrição: Este caso de uso descreve as etapas necessárias para o usuário iniciar a o pré-diagnostico respondendo ao questionário.

Pré-condições:

O paciente deve estar cadastrado no Sistema.

Pós-condições:

Após o cadastro do paciente o usuário ADM solicitará o pré-diagnostico iniciando o questionário.

Fluxo Evento Principal:

1. O usuário após cadastrar o paciente, ira solicitar o pré-diagnostico. Iniciando assim a interação do usuário com o Sistema via comando de voz.

UC 04 – Caso de Uso Solicitar Relatório.

Ator Principal: ADM

Descrição: Este caso de uso descreve as etapas necessárias para que o usuário obtenha um relatório com Pré-diagnostico da pré-consulta do paciente.

Pré-condições:

O ADM deve ter realizado o cadastro do paciente previamente, e o paciente deve já ter sido submetido ao questionário.

O ADM ao solicitar o relatório deve fornecer o CPF do paciente.

Pós-condições:

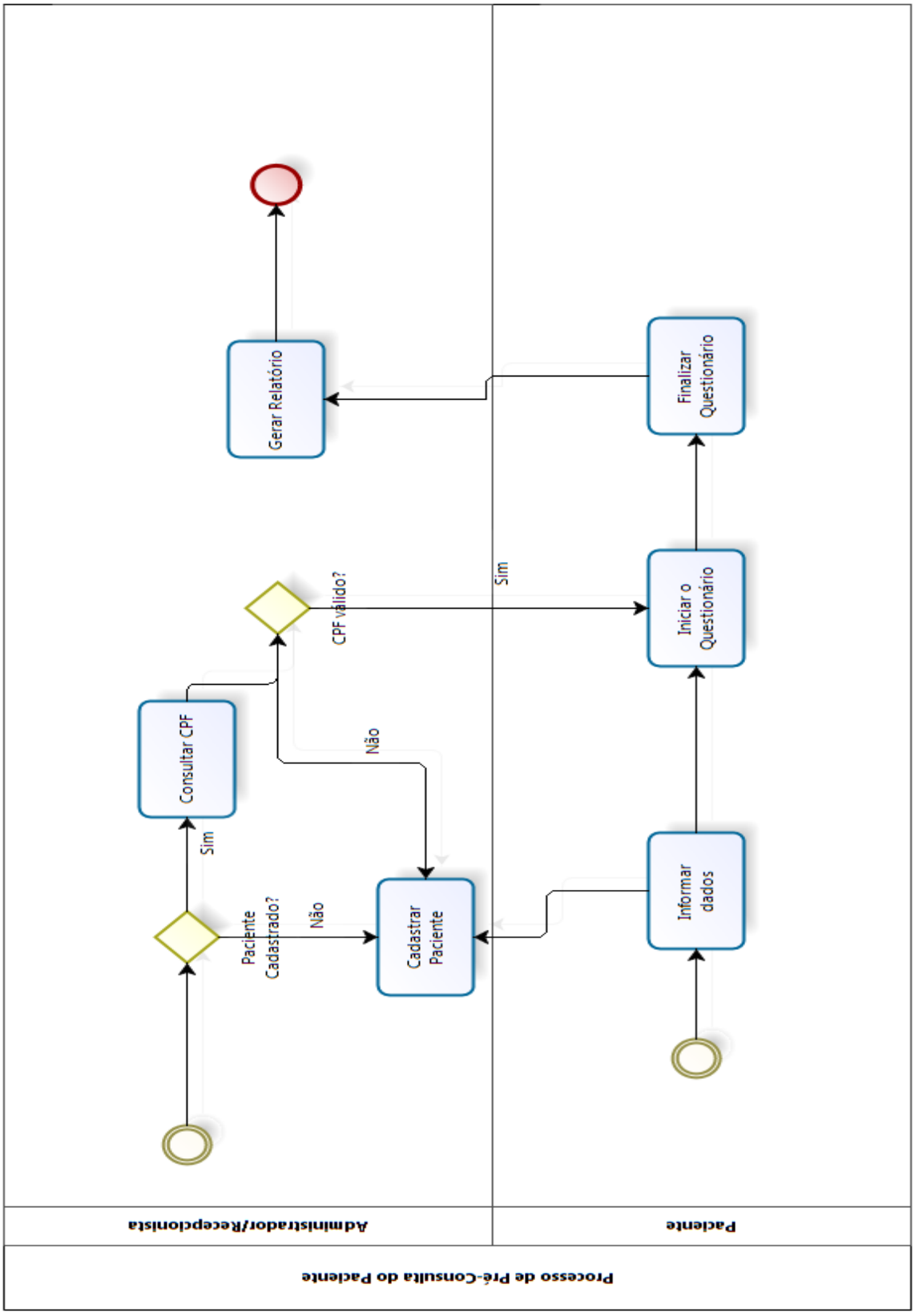
O ADM deve solicitar o Pré-diagnostico clicando no botão Solicitar Pré-diagnostico

Fluxo Evento Principal:

1. O ADM pós realizar o cadastro do paciente, iniciar o pré-diagnostico e esperar que o mesmo responda, deve solicitar para o sistema que o mesmo gere o relatório de pré-diagnostico clicando no botão relacionado.

2. O Sistema informa um relatório de pré-diagnostico com os dados pessoais do paciente fornecidos no cadastro e um resultado para o estágio da doença (Primaria, Secundária, Terciária ou sem Índice).

APÊNDICE E – Fluxograma de processo.



APÊNDICE F – Códigos Fonte do Projeto

Classe Recognize.java

```
importjava.io.FileReader;
importjava.io.IOException;

importjavax.speech.AudioException;
importjavax.speech.Central;
importjavax.speech.EngineException;
importjavax.speech.EngineStateError;
importjavax.speech.recognition.DictationGrammar;
importjavax.speech.recognition.GrammarException;
importjavax.speech.recognition.Recognizer;
importjavax.speech.recognition.RecognizerModeDesc;
importjavax.speech.recognition.ResultAdapter;
importjavax.speech.recognition.RuleGrammar;

public class Recognize extends ResultAdapter {

	public static Recognizer rec;
	public static RuleGrammar gram, gram2;
	public static DictationGrammardic;

	public static void main() {

		try {

		RecognizerModeDescrmd = (RecognizerModeDesc) Central
				.availableRecognizers(null).firstElement();

	rec = Central.createRecognizer(rmd);// cria o reconhecedor
				System.out.println("rec created");

				rec.allocate();//carrega os arquivos do coruja
FileReader reader = new
FileReader(System.getProperty("user.home")+"/NetBeansProjects/CTADiagnostic/cta.grammar");
				// faz a leitura do arquivo de gramática de comando e controle
				gram = rec.loadJSGF(reader);
				//cria a variável para que os comandos possam ser reconhecidos
				dic = rec.getDictationGrammar("dicSr");
				//cria a variável para ser possível o ditado
				dic.setEnabled(false);
				//nessa aplicação não iremos usar o ditado, por isso ele é pausado
				gram.addResultListener(new RecoListener());
				//cria o listener para a variável da gramática
				rec.resume();
				//inicia o reconhecedor
				System.out.println("rec resume");

		} catch (IOException e) {
				e.printStackTrace();
		} catch (IllegalArgumentExceptione ) {
```

```
                    e.printStackTrace();
                } catch (SecurityException e) {
                    e.printStackTrace();
                } catch (EngineStateError e) {
                    e.printStackTrace();
                } catch (AudioException e) {
                    e.printStackTrace();
                } catch (EngineException e) {
                    e.printStackTrace();
                } catch (GrammarException e) {
                    e.printStackTrace();
                }
        }

}
```

Classe RecoListener.java

```
importjavax.speech.recognition.Result;
importjavax.speech.recognition.ResultAdapter;
importjavax.speech.recognition.ResultEvent;
importjavax.speech.recognition.ResultToken;

//Essa classe é o Listener do reconhecedor que fica sempre esperando algum comando ser dito,
processa aquilo que foi
//dito e manda o resultado para a classe Translator.java que retorna o comando falado.
public class RecoListener extends ResultAdapter{
   @Override
        public void resultAccepted(ResultEvent e) {
                String Recognized = null;
                Scene myScene= new Scene();

        try {
                        Result r = (Result) (e.getSource());
                        ResultTokentokens[] = r.getBestTokens();
                        for (inti = 0; i<tokens.length; i++){
                                Recognized = tokens[i].getSpokenText();
                        }
System.out.println(Recognized);
                        myScene = Translator.translate(Recognized.trim());

                }catch (Exception e1) {
                        e1.printStackTrace();

                }
        }

}
```

ClasseTranslator.java

```
import meuplayer.MP3;
importjava.sql.Connection;
importjava.sql.DriverManager;
importjava.sql.PreparedStatement;
importjava.sql.SQLException;
importjavax.swing.JOptionPane;
importclasses.Iniciar;

public class Translator {

public static Scene translate(String s) throws SQLException {
	Integer op = 22;
intsifisPrim = 0;
intsifisSecu = 0;
intsifisTer = 0;
intsemIndice = 0;
	MP3 musica = new MP3();
//	 String teste=Consultar.cpfPac.getText().toString();

if ((s.equals("sair"))) {
op = 0;
	}
if ((s.equals("iniciar"))) {
op = 1;
	}
if ((s.equals("certo"))) {
op = 2;
	}
if ((s.equals("braço") || s.equals("braços") || s.equals("perna") || s.equals("pernas")
		|| s.equals("costas"))) {
op = 3;
	}
if ((s.equals("coração"))) {
op = 4;
	}
if ((s.equals("mão") || s.equals("mãos") || s.equals("pé") || s.equals("pés"))) {
op = 5;
	}
if ((s.equals("pênis") || s.equals("verilha") || s.equals("veria") || s.equals("viria")
		|| s.equals("vagina") || s.equals("boca"))) {
op = 6;
	}
if ((s.equals("bumda") || s.equals("bumbum"))) {
op = 7;
	}
if ((s.equals("não"))) {
op = 8;
	}
if ((s.equals("sangue"))) {
op = 9;
	}
if ((s.equals("água"))) {
op = 10;
```

```
        }
if ((s.equals("pequena"))) {
op = 11;
        }
if ((s.equals("média"))) {
op = 12;
        }
if ((s.equals("grande"))) {
op = 13;
        }
if ((s.equals("diária"))) {
op = 14;
        }
if ((s.equals("periódica"))) {
op = 15;
        }
if ((s.equals("uma") || s.equals("duas") || s.equals("três"))) {
op = 16;
        }
if ((s.equals("sim"))) {
op = 17;
        }
if ((s.equals("cabeça"))) {
op = 18;
        }
if ((s.equals("quatro") || s.equals("cinco") || s.equals("seis"))) {
op = 19;
        }
if ((s.equals("sete") || s.equals("oito") || s.equals("nove") || s.equals("dez")
                || s.equals("onze") || s.equals("doze"))) {
op = 20;
        }


switch (op) {
case (0):
                String path = "/home/ubuntu/Música/tcc/obrigado.mp3";
musica.play(path);

try {
Recognize.rec.deallocate();
System.out.println("Saindo do Programa");
System.exit(0);
newCTADiagnostic().setVisible(true);

                } catch (Exception exp) {
System.out.println("Erro: Saindo do Programa!"
                        + exp.toString());
                }

break;

case (1):
                String path2 = "/home/ubuntu/Música/tcc/inicio.mp3";
musica.play(path2);
break;
```

```
case (2):
        String path3 = "/home/ubuntu/Música/tcc/ferimentoIngua.mp3";
musica.play(path3);
break;

case (3):
semIndice++;
        String path4 = "/home/ubuntu/Música/tcc/tamanho.mp3";
musica.play(path4);
break;

case (4):
        String path5 = "/home/ubuntu/Música/tcc/avc.mp3";
musica.play(path5);
break;

case (5):
sifisSecu++;
        String path10 = "/home/ubuntu/Música/tcc/tamanho.mp3";
musica.play(path10);
break;

case (6):
sifisPrim++;
        String path11 = "/home/ubuntu/Música/tcc/tamanho.mp3";
musica.play(path11);
break;

case (7):
semIndice++;
        String path12 = "/home/ubuntu/Música/tcc/tamanho.mp3";
musica.play(path12);
break;

case (8):
semIndice++;
        String path13 = "/home/ubuntu/Música/tcc/dorMancha.mp3";
musica.play(path13);
break;

case (9):
semIndice++;
        String path14 = "/home/ubuntu/Música/tcc/sair.mp3";
musica.play(path14);
break;

case (10):
sifisPrim++;
sifisSecu++;
sifisTer++;
        String path15 = "/home/ubuntu/Música/tcc/tempoSintomas.mp3";
musica.play(path15);
break;

case (11):
sifisPrim++;
        String path16 = "/home/ubuntu/Música/tcc/sangueAgua.mp3";
```

```
musica.play(path16);
break;

case (12):
sifisSecu++;
        String path17 = "/home/ubuntu/Música/tcc/sangueAgua.mp3";
musica.play(path17);
break;

case (13):
sifisTer++;
        String path18 = "/home/ubuntu/Música/tcc/sangueAgua.mp3";
musica.play(path18);
break;

case (14):
sifisTer++;
        String path19 = "/home/ubuntu/Música/tcc/sair.mp3";
musica.play(path19);
break;

case (15):
sifisSecu++;
        String path20 = "/home/ubuntu/Música/tcc/sair.mp3";
musica.play(path20);
break;

case (16):
sifisPrim++;
        String path22 = "/home/ubuntu/Música/tcc/febre.mp3";
musica.play(path22);
break;

case (17):
sifisSecu++;
sifisTer++;
        String path25 = "/home/ubuntu/Música/tcc/frequenciaDorFebre.mp3";
musica.play(path25);
break;

case (18):
sifisSecu++;
        String path26 = "/home/ubuntu/Música/tcc/febre.mp3";
musica.play(path26);
break;

case (19):
sifisSecu++;
        String path30 = "/home/ubuntu/Música/tcc/febre.mp3";
musica.play(path30);
break;

case (20):
sifisTer++;
        String path31 = "/home/ubuntu/Música/tcc/febre.mp3";
musica.play(path31);
break;
```

```
        }


while (true) {

if ((semIndice>sifisPrim) && (semIndice>sifisSecu) && (semIndice>sifisTer)) {
StringsemSif = Integer.toString(semIndice);
semSif = "Paciente sem indíce de Sífilis";

try {
Class.forName("com.mysql.jdbc.Driver").newInstance();
            } catch (Exception ex) {
ex.printStackTrace();
            }

            Connection conexao =
DriverManager.getConnection("jdbc:mysql://localhost:3306/diagnostic", "root", "root");
PreparedStatementpst;
try {


              String sql = "UPDATE Paciente set pre_Diagnostico=? "+"WHERE cpf=?";
pst = conexao.prepareStatement(sql);
pst.setString(1, semSif);
pst.setString(2, Iniciar.cpfPac.getText().toString());
pst.executeUpdate();
pst.clearParameters();

System.out.println(Iniciar.cpfPac.getText().toString());
            } catch (Exception ex) {
ex.printStackTrace();
            }

           // JOptionPane.showMessageDialog(null, semSif);
         }
if ((sifisPrim>semIndice) && (sifisPrim>sifisSecu) && (sifisPrim>sifisTer)) {
            String sifPri = Integer.toString(sifisPrim);
sifPri = "Paciente com indíce de Sífilis Primária";

try {
Class.forName("com.mysql.jdbc.Driver").newInstance();
            } catch (Exception ex) {
ex.printStackTrace();
            }

            Connection conexao =
DriverManager.getConnection("jdbc:mysql://localhost:3306/diagnostic", "root", "root");
PreparedStatementpst;
try {
              String sql = "UPDATE Paciente set pre_Diagnostico=? "+"WHERE cpf=?";
pst = conexao.prepareStatement(sql);
pst.setString(1, sifPri);
pst.setString(2, Iniciar.cpfPac.getText().toString());
pst.executeUpdate();
pst.clearParameters();

System.out.println(Iniciar.cpfPac.getText().toString());
```

```
            } catch (Exception ex) {
ex.printStackTrace();
            };
           //  JOptionPane.showMessageDialog(null, sifPri);

         } else if ((sifisSecu>semIndice) && (sifisSecu>sifisPrim) && (sifisSecu>sifisTer)) {
            String sifSec = Integer.toString(sifisSecu);
sifSec = "Paciente com indíce de Sífilis Secundária";

try {
Class.forName("com.mysql.jdbc.Driver").newInstance();
            } catch (Exception ex) {
ex.printStackTrace();
            }

            Connection conexao =
DriverManager.getConnection("jdbc:mysql://localhost:3306/diagnostic", "root", "root");
PreparedStatementpst;
try {
                String sql = "UPDATE Paciente set pre_Diagnostico=? "+"WHERE cpf=?";
pst = conexao.prepareStatement(sql);
pst.setString(1, sifSec);
pst.setString(2, Iniciar.cpfPac.getText().toString());
pst.executeUpdate();
pst.clearParameters();

System.out.println(Iniciar.cpfPac.getText().toString());
            } catch (Exception ex) {
ex.printStackTrace();
            }

            //JOptionPane.showMessageDialog(null, sifSec);
         } else if ((sifisTer>semIndice) && (sifisTer>sifisPrim) && (sifisTer>sifisSecu)) {
            String sifTer = Integer.toString(sifisTer);
sifTer = "Paciente com indíce de Sífilis Primária";

try {
Class.forName("com.mysql.jdbc.Driver").newInstance();
            } catch (Exception ex) {
ex.printStackTrace();
            }

            Connection conexao =
DriverManager.getConnection("jdbc:mysql://localhost:3306/diagnostic", "root", "root");
PreparedStatementpst;
try {
                String sql = "UPDATE Paciente set pre_Diagnostico=? "+"WHERE cpf=?";
pst = conexao.prepareStatement(sql);
pst.setString(1, sifTer);
pst.setString(2, Iniciar.cpfPac.getText().toString());
pst.executeUpdate();
pst.clearParameters();

System.out.println(Iniciar.cpfPac.getText().toString());
            } catch (Exception ex) {
ex.printStackTrace();
            }
```

```
            //JOptionPane.showMessageDialog(null, sifTer);

        }

return null;
      }


   }
}
```

Classe Main.java

```
importorg.lgna.story.SProgram;

public class Main extends SProgram implements Runnable{

static void dispose() {
Main.dispose();
  }

private Scene myScene = new Scene();

public Main(){
super();
   }


public Scene getMyScene() {
returnthis.myScene;
   }


public static void main(final String[] args)
              {

                    final Main story = new Main();
story.initializeInFrame(args);
story.setActiveScene(story.getMyScene());
          Recognize.main();


        }

public void run() {
Main.main(null);
   }

}
```

Classe MP3.java

```
importjava.io.BufferedInputStream;
importjava.io.File;
importjava.io.FileInputStream;
importjavazoom.jl.player.Player;


publicclass MP3 {
/**
   * Objeto para nosso arquivo MP3 a ser tocado
   */
   /**
   * Objeto Player da biblioteca jLayer. Ele tocará o arquivo
   * MP3
   */
private Player player;
  /**
   * Construtor que recebe o objeto File referenciando o arquivo
   * MP3 a ser tocado e atribui ao atributo MP3 da classe.
   *
   * @param mp3
   */

   /* Método que toca o MP3
*/
public void play(String caminho) {
try {
            File mp3File = new File(caminho);
FileInputStreamfis     = new FileInputStream(mp3File);
BufferedInputStreambis = new BufferedInputStream(fis);
this.player = new Player(bis);
System.out.println("Tocando!");
this.player.play();
System.out.println("Terminado!");
          }
catch (Exception e) {
System.out.println("Problema ao tocar " + caminho);
e.printStackTrace();
}
}

}
```