Nomes: **RA:**

Giovanna Vendramini 173304

Vitoria Dias Moreira Pinho 188511

Trabalho 1 - Etapa 4
Chronos Parte 1

I. Introdução

A tireóide é a glândula localizada na parte inferior do pescoço, responsável pela produção de hormônios que controlam o metabolismo das células, garantindo assim, o funcionamento correto do corpo. Seus principais hormônios são o T3 e o T4, sendo que é controlada por hormônios liberados por outras glândulas. Quando os níveis de T3 e T4 estão baixos, o hipotálamo libera o TRH, provocando a também liberação do TSH pela glândula pituitária, que estimula a tireóide. Quando há um desequilíbrio nessa função, ocorrem os distúrbios da tireóide.

II. Resumo

Devido a alta incidência de pessoas com distúrbios da tireóide, o estudo sobre o diagnóstico e a prevenção do problema tornam-se cada vez mais necessários.

Dessa forma, o trabalho terá como intuito realizar o diagnóstico de pacientes com base em alguns valores de seus exames, como TSH, T3, T4 total e livre, além de predizer a possibilidade do indivíduo apresentar algum distúrbio na tireóide de acordo com suas informações gerais como idade, sexo, gravidez, se apresenta bócio, entre outros.

III. Requisitos

A. Funcionais

- Diagnosticar se o paciente apresenta hipotireoidismo e hipertireoidismo com base em exames clínicos;
- Predizer, de acordo com informações gerais do paciente como idade e sexo, se ele possui tendência a apresentar distúrbios da tireóide;
- Caso seja detectado desequilíbrio nos hormônios da tireóide, diagnosticar em qual categoria este pertence, dentre: hipotireoidismo, hipotireoidismo primário, hipotireoidismo compensado, hipotireoidismo secundário, hipertireoidismo, T3 tóxico, bócio, toxicidade secundária.

B. Não funcionais

- Deve apresentar um alto grau de confiabilidade, sendo capaz de acertar a maioria dos diagnósticos.

IV. Base de Dados

UCI Machine Leraning Repositor (http://archive.ics.uci.edu/ml/datasets/thyroid+disease) é um site que disponibiliza diversas Bases de Dados para uso em machine learning. As bases fornecidas para distúrbios de tireóide são do Instituto Garvan, na Austrália. Foram selecionadas três bases:

1. new-thyroid:
    a. Apresenta três casos: indivíduos normais, que possuem hipotireoidismo ou hipertireoidismo.
    b. 6 Atributos;
    c. 215 Instâncias;
    d. Atributos específicos: T3, T4, TSH; ...
2. allhyper:
    a. Apresenta se o indivíduo possui algum tipo de hipertireoidismo;
    b. 30 atributos;
    c. 2800 instâncias;
    d. Atributos gerais: idade, sexo, gravidez, se realizou cirurgia da tireóide; …
    e. Atributos específicos: TSH, T3, T4; ...
3. allhypo:
    a. Apresenta se o indivíduo possui algum tipo de hipotireoidismo;
    b. 30 atributos;
    c. 2800 instâncias;
    d. Atributos gerais: idade, sexo, gravidez, se realizou cirurgia da tireóide, ...
    e. Atributos específicos: TSH, T3, T4, ...

V. Ferramenta de Análise

A ferramenta de análise escolhida foi o Weka, de forma que o Weka permitirá a predição da doença a partir das bases de dados.

VI. Desenvolvimento do Modelo Conceitual

A) 1ª Versão

A primeira versão do modelo conceitual foi criada levando em consideração apenas a organização dos dados em tabelas. Dessa forma, havia três tipos de pacientes, referentes às três tabelas e aos exames que eles realizavam em cada uma delas. Por isso, não representava de forma correta um modelo conceitual.

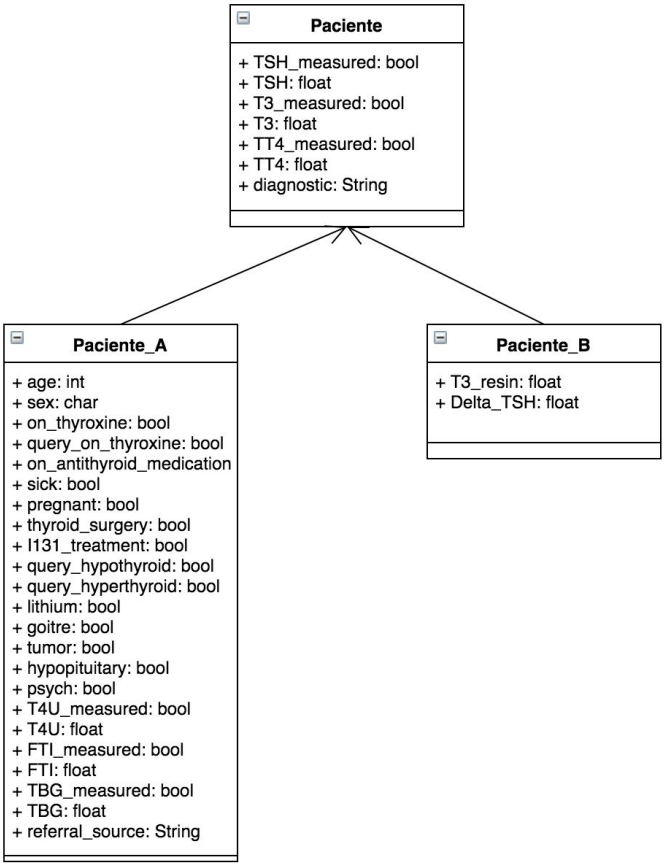

Fig.1: Primeira versão do modelo conceitual em UML.

B) 2ª Versão

Na segunda versão, o modelo já tomou uma nova forma e se aproximou melhor de um modelo conceitual. No entanto, ainda havia pontos a serem melhorados, como a classe doença que tinha dois herdeiros: hipertireoidismo e hipotireoidismo, e a classe exame que apresentava muitos atributos. As classes herdeiras hipertireoidismo e hipotireoidismo faziam com que essas instâncias de doença se tornassem classes, o que não estava correto. Já o exame se tornava algo muito complexo e de difícil atualização, com itens muito específicos.

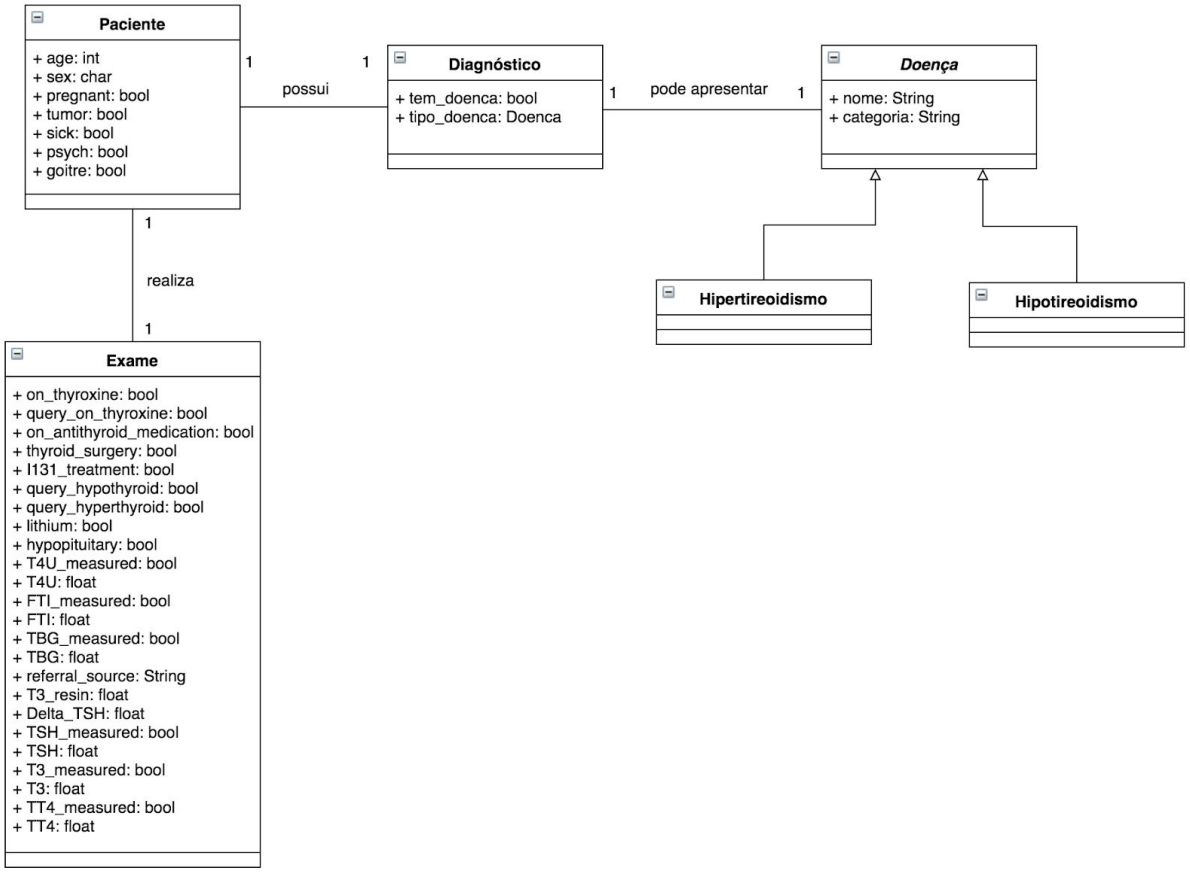

Fig.2: Segunda versão do modelo conceitual em UML.

C) 3ª Versão (Versão Final)

A versão final corrige os problemas da versão dois, eliminando as classes herdeiras de doença e criando uma nova classe item, que instancia cada um dos itens do exame separadamente. Nesse novo modelo, exame se torna uma agregação de itens de exame.

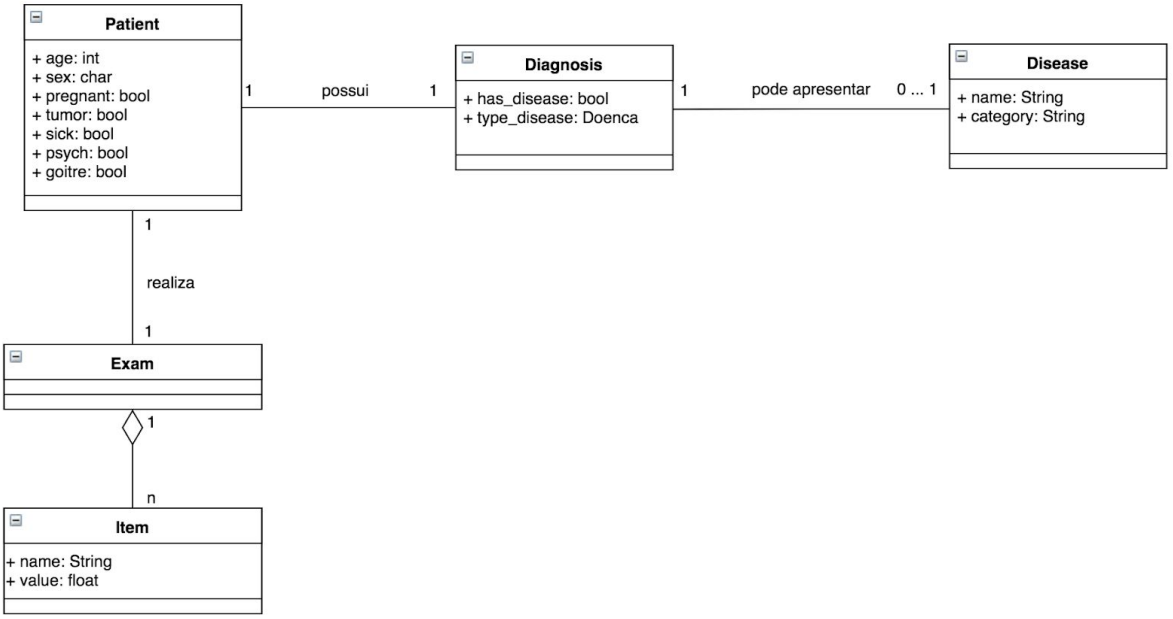

Fig.3: Diagrama final do modelo conceitual em UML.

D) 4ª Versão (Versão para Banco de Dados de Grafos)

A quarta versão foi utilizada para a criação do banco de dados na forma de grafo. Nele os atributos do paciente se tornam uma nova classe, assim como as sub categorias de doença também passam a pertencer à classe disease, havendo uma agregação entre o paciente e os atributos que possui, bem como as sub categorias de doença estão associadas às super categorias principais (hipertireoidismo e hipotireoidismo).

O paciente está diretamente associado à doença que possui e aos exames que realizou, sendo que se houver um valor associado ao item do exame, esse terá seu valor indicado na aresta.

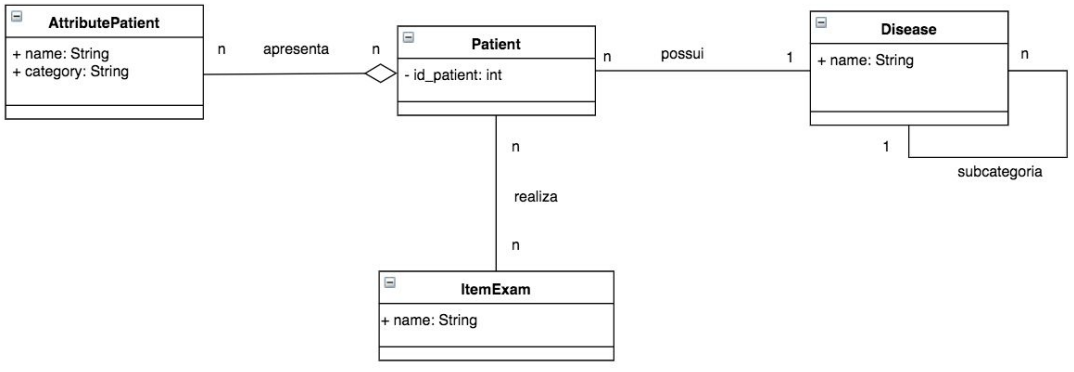

Fig.4: Diagrama do modelo conceitual em UML para o banco de dados de grafos.

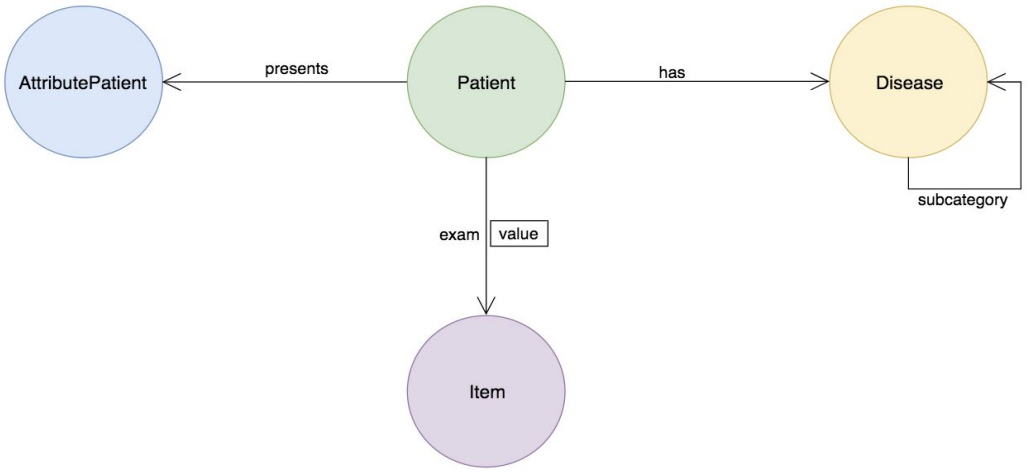

Fig.5: Representação do banco de dados de grafos criado no Neo4J.

E) 5ª Versão (Versão para Banco de Dados Web)

A quinta versão foi utilizada para a criação do banco de dados web. A diferença neste é que, assim como no modelo de grafos, o paciente possui menos propriedades associadas, havendo novamente uma independência com seus atributos. Dessa forma, o paciente apenas aponta para os atributos e itens que possui e está associado a uma única subcategoria de doenças. Semelhante ao modelo de grafos, o paciente aponta para os atributos/itens desejados, entretanto, para que os valores sejam armazenados é preciso guardar uma cópia da referência do atributo para si, o que gera maior repetição de dados, quando comparado com o modelo anterior.

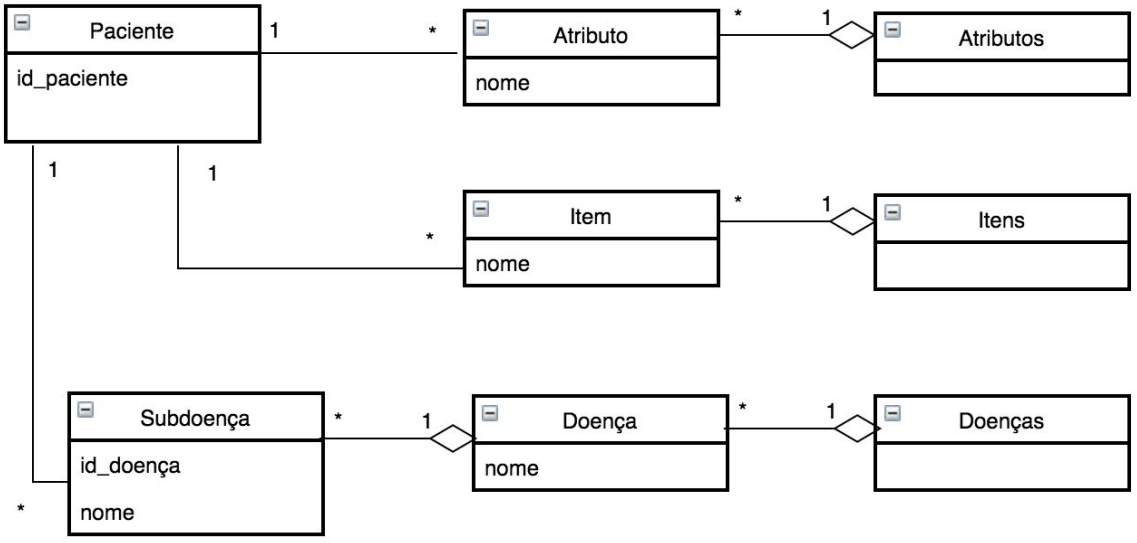

Fig.6: Diagrama do modelo conceitual em UML para o banco de dados web

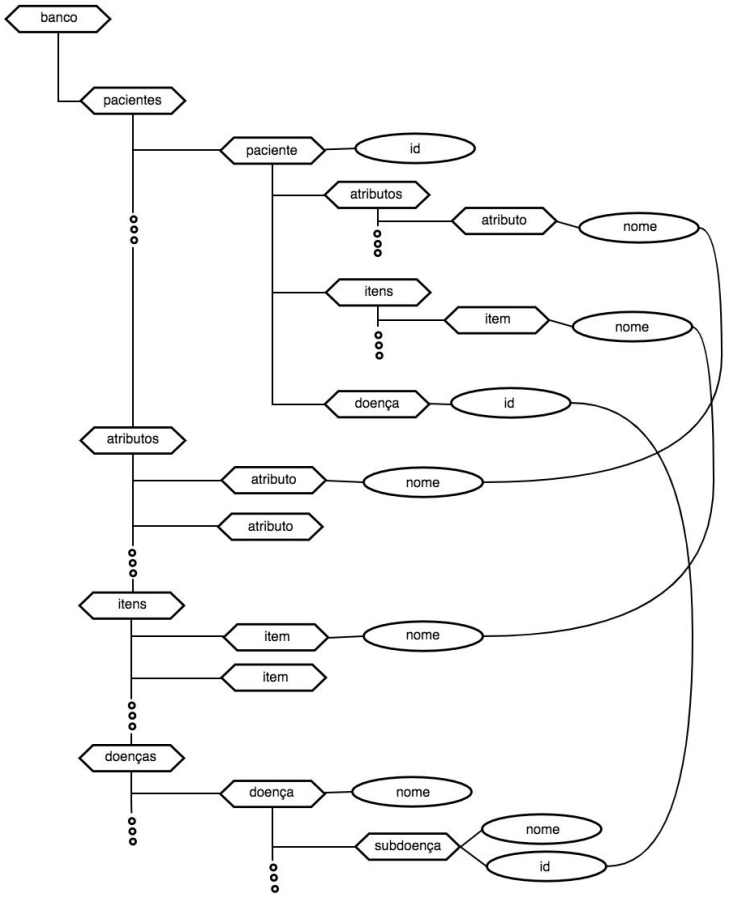

Fig.7: Representação do banco de dados web criado para XML

VII. Modelo Lógico

A. Modelo Lógico Utilizado em SQL

O modelo relacional pode ser organizado nas relações abaixo.

DISEASE (id_disease, name, category)
ITEM (id_item, id_patient, name, result, value) CHE: id_patient para patient(id_patient)
PATIENT (id_patient, id_disease, age, sex, pregnant, tumor, sick, psych, goitre) CHE: id_disease para disease(id_disease)

É possível dizer que o modelo relacional se mostrou bem condizente com o modelo conceitual. As modificações adotadas foram a adição de chaves primárias e chaves estrangeiras para o relacionamento entre as tabelas e exclusão das tabelas Diagnosis e Exam. A justificativa para a exclusão da primeira se dá porque esta dificultaria o processo de consulta direta da doença de determinado paciente, apenas para apresentar a informação de um diagnóstico negativo. Ou seja, a

motivação para sua construção era indicar os casos em que o paciente não foi diagnosticado com nenhum distúrbio da tireóide, problema que foi resolvido ao atribuir ao paciente diretamente o id_disease (da tabela Disease), e permitindo que tal campo seja nulo (caso em que has_disease seria negativo). Já a exclusão da tabela Exam apresenta uma justificativa análoga, do ponto de vista de modelo conceitual a atribuição de itens de exame ao paciente dificulta o entendimento e organização do problema. Entretanto, do ponto de vista lógico atribuir o item de exame diretamente ao identificador do paciente torna direto o acesso à informação, facilitando a geração de consultas e sem perda de informação (já que não havia nenhuma informação referente apenas ao exame).

B. Modelo Lógico Utilizado em Cypher

Foram necessárias adaptações no modelo relacional de forma que houvesse as tabelas referentes aos relacionamentos. Isso é, além das tabelas que seguem as classes do modelo conceitual, identificando doença, itens do exame, paciente e atributos, deve haver também as tabelas de arestas, que relacionam as classes, como por exemplo conectar um paciente com uma doença, atributo ou item de exame. Sendo que na aresta EXAM que relaciona um paciente ao item de exame, se um item de exame possuir valor, esse será apresentado na aresta.

O modelo relacional foi organizado nas relações abaixo.

DISEASE (id_disease, name)
ITEM (id_item, name)
PATIENT (id_patient)
ATTRIBUTE_PATIENT(id_attribute, name)
EXAM(id_patient, id_item, value) CHE: id_patient para PATIENT(id_patient) CHE: id_item para ITEM(id_item)
HAS(id_patient, id_disease) CHE: id_patient para PATIENT(id_patient) CHE: id_disease para DISEASE(id_disease)
PRESENTS(id_patient, id_attributte) CHE: id_patient para PATIENT(id_patient) CHE: id_attributte para ATTRIBUTE_PATIENT(id_attributte)

C. Modelo Lógico Utilizado em XML

Não foram necessárias muitas adaptações do modelo conceitual para o lógico, já que o modelo conceitual foi pensado para seguir a estrutura do xml.

PATIENT(<u>id_patient</u>, attributes, items, id_disease) CHE: items para ITEMS(items) CHE: attributes para ATTRIBUTES(attributes) CHE: id_disease para SUBDISEASE(id_disease)
ITEM (name)
ATTRIBUTE (name)
ITEMS(items, item) CHE: item para ITEM(nome)
ATTRIBUTES(attributes, attribute) CHE: attribute para ATTRIBUTE(nome)
DISEASES(diseases, name_disease) CHE: name_disease para DISEASE(name_disease)
DISEASE(name_disease, id_disease) CHE: id_disease para SUBDISEASE(id_disease)
SUBDISEASE(id_disease, name)

VIII. Vantagens do Modelo de Grafo

As vantagens que o modelo de grafos pode oferecer, quando comparado ao utilizado em SQL, são a possibilidade de evitar a repetição de atributos, itens e doenças, uma vez que eles serão criados uma única vez e sua relação com o paciente será feita através de arestas, o que também torna mais visual o relacionamento e permite que se faça análises através de caminhos.

Outra vantagem é não ter a necessidade de utilizar diversas tabelas, realizando o join, para toda query que for fazer, pois pode ser feita diretamente através de um caminho em Cypher, sendo simples relacionar vértices diferentes.

Por fim, tornou-se possível criar os vértices de sub categorias de doença e um caminho para as super categorias. Dessa forma, um paciente está conectado à subcategoria da doença que possui e essa se relaciona com a doença mãe.

IX. Vantagens do Modelo Web

As vantagens que o modelo XML pode oferecer é referente à categorização dos dados, ou seja, separa-se os dados de pacientes, itens, atributos e doenças, sendo criado apenas links entre eles. Dessa forma, torna-se mais simples manter a consistência dos dados, uma vez que o dado é apontado por outras categorias e uma alteração nele reflete em todo o banco imediatamente.

Além disso, os dados são apresentados de maneira estruturada, de forma que ao mesmo tempo que é facilmente apresentada ao usuário, pode ser rapidamente interpretada pelo sistema.

Por conseguinte, realizar análises através desse modelo se torna eficaz pela organização dos dados e sua categorização, principalmente.

Já o modelo RDF possui vantagens por apresentar seus dados de forma linkada, havendo um relacionamento entre os conceitos. Esse modelo torna fluída a análise dos dados e navegar por tópicos relacionados, por exemplo.