



UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ESCOLA DE INFORMÁTICA APLICADA

Implementação do Segmento Laboratorial do Protocolo HL7 Usando o MongoDB.

Geraldo Gabriel Crelier dos Santos

Orientador

Márcio de Oliveira Barros

RIO DE JANEIRO, RJ – BRASIL

DEZEMBRO DE 2014

# IMPLEMENTAÇÃO DO SEGMENTO LABORATORIAL DO PROTOCOLO HL7 USANDO O MONGODB

Projeto de Graduação apresentado à  
Escola de Informática Aplicada da  
Universidade Federal do Estado do Rio de  
Janeiro (UNIRIO) para obtenção do título de  
Bacharel em Sistemas de Informação.

GERALDO GABRIEL CRELIER DOS SANTOS

Prof. Dr. Márcio de Oliveira Barros

## **Implementação do Segmento Laboratorial do Protocolo HL7 Usando o MongoDB.**

Geraldo Gabriel Crelier dos Santos

Aprovado em \_\_\_\_/\_\_\_\_/\_\_\_\_

### **BANCA EXAMINADORA**

---

Prof. Dr. Márcio de Oliveira Barros (UNIRIO)

---

Prof. Dr. Astério Kiyoshi Tanaka (UNIRIO)

---

Prof. Dr. Luiz Carlos Montez Monte (UNIRIO)

Os autores deste Projeto autorizam a ESCOLA DE INFORMÁTICA APLICADA da UNIRIO a divulgá-lo, no todo ou em parte, resguardando os direitos autorais conforme legislação vigente.

Rio de Janeiro, \_\_\_\_ de \_\_\_\_ de \_\_\_\_

---

Geraldo Gabriel Crelier dos Santos

## **AGRADECIMENTOS**

- A Deus acima de tudo. Não importa como ele seja, sempre esteve presente.
- A minha família. Pela compreensão e ajuda nas horas mais difíceis.
- Ao corpo docente, diretoria da EIA (Escola de informática Aplicada) e do CCET da UNIRIO. Por todo suporte para que esta graduação seja uma realidade.
- Ao prof. Marcio Barros, por toda dedicação empregada para o sucesso desta publicação.
- A todos os meus amigos. Pelo apoio e compreensão nas minhas ausências, estas que serão devidamente compensadas.
- Ao Instituto Nacional de Cardiologia. Pelo apoio e pela percepção das diversas necessidades de sistemas para área médica.

## RESUMO

Ao longo do tempo, presencia-se a intensa evolução da Medicina promovida pelos avanços tecnológicos e das comunicações. Tais avanços promoveram o surgimento do Prontuário Eletrônico do Paciente (PEP). O desenvolvimento de avançados equipamentos médicos, em conjunto com novas técnicas de diagnóstico e intervenções ao paciente, fomentam o uso do Prontuário Eletrônico com o objetivo de apoiar a tomada de decisão, diminuindo custos e promovendo o bem-estar do paciente.

Peça fundamental para a tomada das decisões médicas via PEP, o Serviço de Análises Clínicas vem passando por grandes transformações promovidas pelo avanço tecnológico. Pesquisas são concluídas com mais rapidez, adicionando novos elementos para o perfeito desenrolar das atividades médicas. Tal avanço impulsionou a demanda por serviços de *Business Intelligence*, *Data Mining* e *Data Storage* cada vez mais eficientes e velozes. Neste contexto, busca-se por soluções que integrem cada vez mais os equipamentos e serviços dentro das instituições médicas. Protocolos universais de interoperabilidade, como o HL7, e de nomenclaturas, como o LOINC, contribuem para uma melhor sintonia entre os diferentes sistemas e dispositivos que atendam o Serviço de Diagnóstico Clínico.

Os fenômenos do *BigData* e *Cloud Computing* trouxeram novas demandas no que se refere à persistência de dados. Problemas que eram facilmente tratados pelo modelo relacional em sistemas monolíticos atualmente encontram obstáculos para atender às necessidades dos usuários em um contexto de dados cada vez mais volumosos e aplicações que promovem consultas com alto grau de abstração. A utilização de bancos de dados que garantem maior escalabilidade e flexibilidade constitui uma solução que traz vantagens para prospecção e cruzamento dos dados de forma ágil. Em conjunto com os protocolos de interoperabilidade, o uso de bancos não-relacionais pode atender às necessidades dos sistemas médicos com o objetivo de viabilizar, de forma cada vez mais eficiente, as atividades de diagnóstico e tomada de decisão médica.

Esta monografia analisa as principais características do protocolo HL7, focando em sua versão 3 na forma da implementação documental CDA (*Clinical Document Architecture*). Em conjunto com o MongoDB, pretende-se demonstrar as vantagens da integração entre bancos não-relacionais e HL7, com o objetivo de otimizar a interoperabilidade e persistência de dados que representam resultados de exames clínicos.

**Palavras chave:** HL7, MongoDB, CDA, padrões, sistemas de informação, interoperabilidade, prontuário eletrônico do paciente, noSQL.

## ABSTRACT

Over time, we observe and intense evolution of medical procedures promoted by advances in technology and communications. Such advances have promoted the emergence of the Electronic Health Record (EHR). The development of advanced medical equipment, along with new diagnostic techniques and interventions to the patient, promote the use of the Electronic Health Record to support decision making, reduce costs and promote the patient well-fare.

Keystone for taking medical decisions based on the EHR, the Clinical Diagnosis Service (CDS) has been undergoing major changes promoted by technological advancement. Research projects are completed more quickly, adding new elements to the execution of the medical procedures. Such an advance boosted the demand for Business Intelligence services, as well as faster and more efficient *Data Mining* and *Data Storage* services. In this context, there is a constant search for solutions that integrate more and more equipment and services within medical institutions. Universal interoperability protocols, such as HL7, and nomenclatures, such as LOINC, contribute to a better alignment of the different systems and devices that meet the Clinical Diagnosis Service.

The phenomena of *Big Data* and *Cloud Computing* brought new demands with regard to data persistency. Problems that were easily dealt with by the relational model in monolithic systems currently face obstacles to meet the needs of users in a context of increasingly large datamarts and applications that promote abstract queries. Databases designed for greater scalability and flexibility may bring forth benefits to such applications, allowing for ad-hoc queries and fast and unstructured cross-references. Along with interoperability protocols, non-relational databases can meet the needs of current medical systems, fastening the support for medical decision-making.

This document analyzes the main features of the HL7 protocol, focusing on its third version in the form of its documentary implementation: the CDA (*Clinical Document Architecture*). We use MongoDB to demonstrate the advantages of the integration between non-relational databases and HL7, aiming to optimize interoperability and data persistence representing clinical examinations.

**Keywords:** HL7, MongoDB, CDA, standards, information systems, interoperability, electronic patient record, NoSQL.

# Sumário

Lista de Figuras .....	9
Lista de Códigos.....	10
Lista de Tabelas .....	11
Glossario.....	12
1. Introdução.....	14
1.1 Contexto .....	14
1.2 Serviço de Patologia Clínica: Do Manual à Automação .....	14
1.3 Objetivo.....	16
1.4 Estrutura do Documento.....	17
2. HL7.....	18
2.1 As Primeiras Versões do Protocolo HL7 .....	18
2.2 A Versão 3 do Protocolo HL7.....	20
2.2.1 Reference Information Model – RIM .....	21
2.2.2. Abstrações/Refinamentos do RIM .....	27
2.2.3 Nomeando Artefatos.....	29
2.3 HL7 V3 CDA.....	30
2.4 Considerações Finais .....	31
3. NoSQL e MongoDB .....	33
3.1 Dificuldades no Padrão Relacional de Dados .....	33
3.2 Visão noSQL.....	33
3.3 Teoremas BASE e CAP .....	34
3.4 Tipos de Armazenamentos em Bancos noSQL.....	36
3.5 MongoDB.....	39
3.5.1 Arquitetura de Dados.....	39

3.5.2 Replicação .....	41
3.5.3 Sharding.....	42
3.5.4 GridFS .....	43
3.6 Considerações Finais .....	43
4. Exemplo de Implementação – Laboratório de Análises Clínicas.....	45
4.1 Proposta de Desenvolvimento .....	45
4.2 Casos de Uso .....	45
• Exportar CDA .....	46
• Armazenar Exame .....	46
• Buscar Resultados por Pedido de Exame .....	46
• Buscar Resultados por Paciente .....	46
• Buscar Resultados por Data: .....	46
• Buscar Resultados por Exame: .....	47
4.3 Diagramas de Classes .....	47
4.4 Ferramentas Utilizadas.....	48
4.4.1 Linguagens e IDE's .....	48
4.4.2 Bibliotecas e Ferramentas de Código .....	48
4.4.3 Programas acessórios.....	49
4.5 Funcionalidades do LabResults .....	49
4.6 Funcionalidades do LabQuery .....	50
4.7 Considerações Finais .....	54
5. Conclusão .....	55
5.1 Considerações gerais.....	55
5.2 Trabalhos Futuros.....	56
6. Referências Bibliográficas .....	57
ANEXO I .....	58
ANEXO II .....	59
ANEXO III .....	60
ANEXO IV .....	61



## Lista de Figuras

Figura 01: Esquema de comunicação HL7 v2. ....	19
Figura 02 – Componentes de uma mensagem HL7 v2.x. ....	19
Figura 03 – Classes principais e associativas do RIM. ....	22
Figura 04 – Classe Act do RIM. ....	23
Figura 05 – Classe Entity do RIM. ....	24
Figura 06 – Classe Role do RIM ....	25
Figura 07 – Diagrama para classes ActRelationship.....	26
Figura 08 – Diagrama para classes RoleLink.....	26
Figura 09 – Diagrama para classes Participation.....	27
Figura 10 – Refinamento do RIM. ....	27
Figura 11 – Teorema CAP .....	35
Figura 12 – Posição de alguns Bancos de Dados dentro do Teorema CAP. ....	36
Figura 13 – Posição dos bancos de dados em função de estrutura de registros. ....	36
Figura 14 – Mongo VUE (Cliente para MongoDB) – Banco de dados baseado em documentos 37	
Figura 15 – HareDB (Cliente para bancos HBase) – Banco de dados orientado por colunas. ....	37
Figura 16 – Neo4j: Banco de dados baseado em grafos .....	38
Figura 17 – Cliente Redis Desktop Manager: Banco de dados baseado em Chave/Valor .....	39
Figura 18 – MongoDB: Collection de documentos sobre exames. ....	40
Figura 19 - MongoDB: Documentos com composição variada. ....	40
Figura 20 - Lista com vários arquivos de diferentes bancos do MongoDB. ....	41
Figura 21 – Esquema de réplica de dados no MongoDB .....	42
Figura 22 – Esquema de sharding no MongoDB .....	43
Figura 23 Diagrama de Casos de Uso da Aplicação LabResults.....	45
Figura 24 – Diagrama de Casos de Uso da Aplicação LabQuery. ....	46
Figura 25 – Diagrama de Classes da Aplicação LabQuery. ....	47
Figura 26 – Cliente MongoDB MongoChef.....	49
Figura 27 – Tela principal da aplicação LabResults. ....	50
Figura 28 – Visão geral da aplicação LabQuery.....	50
Figura 29 – Pesquisa por Pedido. ....	51
Figura 30 – Pesquisa por Paciente. ....	51
Figura 31 – Pesquisa por Data Fixa.....	52
Figura 32 – Pesquisa por Período de Data. ....	52
Figura 33 – Pesquisa por Exame.....	53
Figura 34 – Pesquisa por Exame com busca por faixa de valores. ....	53

## **Lista de Códigos**

Código 01: Mensagem v2 do tipo ADT-A01 .....	18
Código 02 – Mensagem v2 em codificação xml .....	20
Código 03 – Exemplo de mensagem HL7 v3 .....	29

## **Lista de Tabelas**

Tabela 01 – Quadro de principais tipos de mensagens v2.x. ....	20
Tabela 02 – HMD em forma de tabela .....	28

## **Glossario**

### **A**

ANSI – American National Standards Institute.

### **B**

BI – Buisness Intelligence.

BSON – Binary JSON. Serialização binária do JSON.

### **C**

CBIS – Congresso Brasileiro de Informática em Saúde.

CCD – Continuity of Care Document.

CCA – Continuity of Care Record.

CDA – Clinical Document Architecture.

CFM – Conselho Federal de Medicina.

### **D**

DBA – Database Administrator.

DMIM – Domain Message Information Model.

### **F**

FHIR – Fast Healthcare Interoperability Resource.

### **G**

GUI – Graphic User Interface.

### **H**

HIS – Hospitalar Information System.

HL7 – Health Level 7. Protocolo de Interoperabilidade entre Sistemas Hospitalares.

HISPP – Healthcare Information Standards Planning Panel.

HMD – Hierarchical Message Description.

### **I**

IDE – Integrated Development Environment.

ISO – International Organization for Standardization.

### **J**

JSON – JavaScript Object Notation.

JVM – Java Virtual Machine.

### **L**

LIS – Laboratory Information System.

LOINC – Logical Observation Identifiers Names and Codes.

## **P**

PEP – Prontuário Eletrônico do Paciente.

POM – Project Object Manager.

## **R**

RAID – Redundant Array of Independent Disks.

RIM – Reference Information Model.

RMIM – Refined Message Information Model.

## **S**

SBIS – Sociedade Brasileira de Informática em Saúde.

SQL – Structured Query Language.

SNOMED-CT – Systematized Nomenclature of Medicine - Clinical Terms.

SOA – Service-Oriented Architecture.

## **V**

V2 – Health Level Seven Version 2.

V3 – Health Level Seven Version 3.

## **X**

XML – EXtended Markup Language.

# 1. Introdução

## 1.1 Contexto

Desde o início das atividades de assistência à saúde, que remontam há pelo menos 2500 anos, por Hipócrates, o registro das ocorrências e técnicas realizadas ao paciente compõe-se de elemento fundamental para o desenvolvimento da Medicina. O mesmo Hipócrates no século 5 a.c., ao incentivar aos seus médicos o registro escrito dos pacientes, defendia que o mesmo deveria refletir de forma cronológica todo o curso das ocorrências que acometeram o paciente, bem como suas possíveis causas, tratamentos executados e as consequências das intervenções realizadas. A este registro, denominamos **Prontuário do Paciente**.

Com o desenvolvimento de técnicas e estruturas de registro dos prontuários médicos, promovidas pelos profissionais de saúde, destacando as figuras de *Florence Nightingale (1855)*, *Willian Mayo (1907)* e *Lawrence Weed (1969)*, o acervo de todos estes documentos toma o comportamento de um verdadeiro banco de dados físico. Tal instrumento contribui para o refino das atividades de tomada de decisão do corpo médico e fomenta ainda o desenvolvimento da Medicina em uma escala crescente e ininterrupta, atravessando o nosso tempo.

Com o advento dos Sistemas de Informação e dos bancos de dados, a forma de alocação, tratamento e disposição dos dados relativos à base de conhecimentos da Medicina vem se adaptando a uma nova realidade, que exige não só precisão e clareza nas decisões tomadas, mas que as mesmas sejam executadas maximizando a rapidez, eficiência e economia. Isto posto, tem-se agora o chamado **Prontuário Eletrônico do Paciente (PEP)**. Define-se como PEP, o conjunto de registros eletrônicos, links para bases de conhecimento médico, dentre outras fontes de dados médicos providos por um sistema informatizado, com o objetivo de fornecer elementos para a tomada de decisão e assistência à pesquisa em qualquer área da saúde. Os registros e demais dados são catalogados de forma cronológica, são intercambiáveis com outros sistemas e disponibilizados de forma ampla e clara, porém atentando à segurança do conteúdo. Com o PEP, pretende-se alcançar a rapidez, facilidade de comunicação, economia e melhoria na guarda de documentação e nos fluxos de trabalho.

Para tanto, o PEP dispõe de normas, padrões e regulamentos para sua correta implementação. No Brasil, o Ministério da Saúde, o Conselho Federal de Medicina (CFM) e a Sociedade Brasileira de Informática em Saúde (SBIS) tomam a frente na regulação, promoção e processo de implementação do PEP. Dentre as normas e leis, podemos citar a Resolução CFM 1821 de 11/07/2007 e a portaria 2073 de 31/08/2011 do Ministério da Saúde. Um dos pontos que chama mais atenção é o aspecto da interoperabilidade entre os diversos sistemas e os artefatos gerados pelos mesmos. No escopo deste trabalho, focaremos no **Serviço de Patologia Clínica** e de que forma ele promove-se a geração, armazenamento e entrega dos dados de exames laboratoriais.

## 1.2 Serviço de Patologia Clínica: Do Manual à Automação

A Patologia Clínica é um componente fundamental dos processos de assistência à saúde, pois fornece os insumos que o profissional da Medicina necessita para a devida decisão sobre os procedimentos a serem tomados ao paciente. Atualmente,

o corpo médico vem se apoiando cada vez mais em exames e outras técnicas de apoio a fim de assegurar, não só no ponto de vista clínico, como jurídico e administrativo, dada a sua fé pública, o perfeito diagnóstico e possíveis atitudes clínicas a serem tomadas. De fato, a conduta adotada atualmente pela classe médica tem sido a prescrição dos procedimentos analíticos, como os exames de sangue, endoscopia, ultrassom, entre outros, a fim de levantar, em conjunto com o exame clínico e a entrevista ao paciente realizados no consultório, a máxima exatidão possível para que a correta prescrição seja adotada.

Dentro da área da Patologia Clínica, os exames de sangue vêm consolidando sua importância como elemento essencial para a averiguação do estado clínico de um paciente. Inicialmente, os exames eram executados de forma manual, utilizando equipamentos mecânicos e técnicas de cálculo para extrair os resultados. Em seguida, as informações obtidas eram escritas no laudo a ser anexado ao prontuário físico do paciente. A medida que novos equipamentos de análise clínica por automação foram criados, os fabricantes desenvolveram seus próprios sistemas que, na prática, já imprimiam os relatórios para serem anexados diretamente ao prontuário do hospital ou entregues ao paciente, no caso de laboratórios independentes.

Com o surgimento dos sistemas de informação voltados para administração dos diversos campos da Medicina, os resultados passaram a depender quase zero de cálculo humano, bem como o processamento e a entrega dos exames. Estes são realizados e armazenados em memória persistente, de forma automática. A inserção dos resultados no Sistema de Gerenciamento Hospitalar é realizada diretamente entre o equipamento de exame e o servidor, seja por meio de troca de mensagens, middleware ou manualmente, em caso de falha de comunicação.

O ponto conflitante reside no fato de que cada equipamento tinha o seu protocolo de comunicação, bem como um banco de dados interno, onde o modo como era exposto o exame poderia diferir da maneira como está no banco de dados do Sistema de Informação Hospitalar (*Hospital Information System - HIS*). Em suma, um programa específico teria que converter os campos do banco de dados do equipamento para o banco de dados do HIS. Em um cenário onde uma instituição médica usa um HIS conectado a um banco de dados que recebe as informações vindas de vários equipamentos de exames, cada qual com seu protocolo de comunicação e seu banco de dados interno, ao trocar um equipamento por outro, com uma nova metodologia, será necessário refazer a comunicação do equipamento para o banco de dados do cliente. Neste caso, utiliza-se um *middleware* que conterá os diversos protocolos para que se estabeleça a comunicação. Desta forma, a implementação de um novo equipamento, compreendendo a fase de testes com o HIS, pode levar até 3 meses para se concretizar.

A fim de minimizar a complexidade na implementação das interfaces de comunicação, tanto fabricantes como a comunidade de desenvolvedores e os profissionais da área médica buscaram uma forma de padronizar a comunicação entre os equipamentos e os sistemas hospitalares. Deste esforço surgiu a *Health Level Seven Organization*, que iniciou o desenvolvimento de um protocolo aberto a fim de padronizar as mensagens emitidas e recebidas pelos diversos equipamentos e sistemas hospitalares. O HL7 (*Health Level 7*) é o protocolo utilizado para promover a interoperabilidade entre os equipamentos, sistemas e repositórios existentes na

instituição médica. Seu nome deve-se ao fato que sua atuação encontra-se na sétima camada (aplicação) do modelo OSI.

No entanto, deve-se atentar não só para a capacidade de integração de sistemas, mas de que forma pretendemos organizar os dados a fim de que representem as informações necessárias para a tomada de decisão e sejam facilmente consultados. Hoje, devido à grande abrangência e popularidade, os HIS apoiam-se em banco de dados relacionais (como o SQL Server, Oracle e MySQL) como repositórios de dados. No entanto, alguns questionamentos surgem sobre a visão relacional de dados, por exemplo:

- (i) até que ponto pode tornar-se custoso inserir novos parâmetros nas consultas?
- (ii) quando a forma de executar os exames se modifica, surgindo novos campos de dados, como expor estes dados em um modelo consistente com o anterior?
- (iii) quanto isto interfere no custo para implementar um HIS?

Em relação à segunda pergunta, se adicionarmos os campos na mesma tabela de um banco de dados relacional, criaremos valores nulos referentes aos resultados antigos, que ocuparão desnecessariamente a memória do servidor. Além deste fato, dependendo de como as tabelas foram estruturadas, os bancos relacionais podem levar a consultas gigantescas, caso decida-se usar alguma forma particular de cruzamento de dados, como pode ser comum em pesquisas científicas.

Levando em consideração tais questionamentos, torna-se desejável uma forma de compor um repositório de dados mais flexível que o modelo relacional. Voltamos então a nossa atenção para os bancos de dados não-relacionais. Estes bancos podem oferecer uma solução de acordo com a realidade de um repositório voltado mais para a consulta que a inserção. No caso da área de laboratórios, por exemplo, é muito comum a alteração da metodologia usada para um exame, o que geraria novos campos de dados a ser inseridos em um sistema. Não muito raro, criam-se artifícios para relacionar estes novos campos para os novos resultados de exames (Uma instituição médica pode processar 700000 resultados de exames por ano). A escalabilidade em um banco relacional torna-se custosa com a adição de tabelas, relacionamentos e/ou adição de campos extras com valores nulos nos registros anteriores as mudanças.

Quando abordamos a questão no contexto de instituições públicas de saúde brasileiras, temos o seguinte agravante: para fins de economia, muitos gestores não adquirem novos equipamentos de exames. Do contrário, faz-se licitações para compra de testes vinculados a comodato do aparelho (cujas características estão especificadas no Edital) para contratos com vigência de um ano (validade obrigatória de cada licitação deste tipo). Devido à natureza do pregão, que pressupõe aquisição pelo menor preço e não especificação a um determinado equipamento, não se pode garantir que a mesma empresa vencedora do pregão anterior, que atende atualmente a instituição, vença a nova licitação. Desta forma, a troca anual de equipamentos é uma possibilidade real, gerando alto custo para a equipe de TI implementar interfaces para o novo aparelho. Dependendo da complexidade da implementação, conforme já comentado, pode levar meses para que a mesma esteja plenamente funcional.

### **1.3 Objetivo**

Existem duas questões fundamentais a serem tratadas no campo dos Sistemas de Informação para a área de saúde e estas questões tem como objetivo a rapidez na geração e acesso aos dados médicos: a interoperabilidade e a persistência de dados.



Sobre a primeira, é necessário a adoção de um protocolo único de mensagens entre os equipamentos e sistemas hospitalares. Em um contexto em que o tempo é fundamental para assegurar o diagnóstico e o tratamento do paciente, além do fato de que novos equipamentos, sistemas e técnicas são introduzidos na comunidade médica constantemente, a implementação de uma “língua única entre os equipamentos” poderá diminuir o tempo de introdução de um aparelho ou software e reduzir os erros de comunicação comuns em interfaces entre sistemas. Em suma, poderemos trocar de equipamentos com maior rapidez, conforme o interesse da instituição médica.

Sobre a persistência de dados, as áreas voltadas à saúde caracterizam-se pelas constantes descobertas científicas e pelo aperfeiçoamento das técnicas utilizadas pelos profissionais da área. Geralmente as instituições médicas utilizam de SGBD's relacionais para armazenar suas informações. A facilidade de modelagem e rápida implementação destes bancos é posta em prova quando novas classes de informações necessitam ser inseridas ao repositório, em razão da adoção de uma nova técnica ou descoberta científica.

Quando a instituição também realiza pesquisas, implementar novas parametrizações para levantar e cruzar dados objetivando a formação de um conhecimento torna-se uma tarefa custosa.

Outra característica de um banco de dados médico é o número de consultas superior em relação à quantidade de inserções e, principalmente, de atualizações de dados. A necessidade de um banco mais voltado a consulta do que para a inserção dos dados torna-se visível neste contexto. Uma solução pode ser a adoção de bancos de dados não-relacionais, devido a sua maleabilidade no trato dos dados, facilidade de distribuição do banco de dados e busca paralela.

Diante destas questões, temos como objetivo mostrar de que forma uma solução envolvendo o protocolo de interoperabilidade entre sistemas de saúde mais utilizado no momento, o *HL7*, em conjunto com um banco de dados orientado a documentos, no caso o *MongoDB*, pode nos oferecer uma resposta ágil e flexível ao problema de consulta e armazenamento de resultados de exames clínicos, atendendo de forma otimizada às demandas da área de saúde quanto aos sistemas de informação. Focaremos na versão 3 do protocolo *HL7* e umas de suas implementações mais utilizadas, o *CDA (Clinical Document Architecture)*. Apresentaremos uma aplicação de armazenamento e consulta de resultados de exames para demonstrar as potencialidades que a abordagem orientada a documentos pode trazer a este contexto.

#### **1.4 Estrutura do Documento**

Esta monografia apresenta-se sob a seguinte estrutura de capítulos, além desta introdução ao problema:

- **Capítulo 2:** Apresenta uma descrição sobre o protocolo *HL7*, descrevendo as suas versões e focando na versão 3 e na implementação *CDA*;
- **Capítulo 3:** Apresenta os conceitos envolvidos nos bancos de dados não-relacionais, em especial o banco de dados selecionado para este trabalho, o *MongoDB*;
- **Capítulo 4:** Implementação de uma pequena aplicação para gerenciamento de exames, com alguns casos de usos que representam cenários comuns em um *HIS*;
- **Capítulo 5:** Apontamentos finais sobre o *HL7*, *MongoDB* e *NoSQL* em geral com propostas de futuros trabalhos.

## 2. HL7

O protocolo HL7 tem como objetivo a padronização da comunicação entre os sistemas médicos, permitindo tanto metodologias como diretrizes flexíveis, que poderão ser utilizadas por outras instituições de saúde nacionais e internacionais. Este conceito estende-se para os sistemas operacionais (Programas que executam os exames) dos próprios equipamentos analíticos (equipamentos que realizam exames), exibindo resultados universais a qualquer região do mundo.

Certificado pela ANSI/ISO, o protocolo HL7 é amplamente utilizado pelos EUA, Canadá, Austrália e pelos países europeus, legitimando a viabilidade de sua utilização. A *Health Level Seven Organization* promove anualmente congressos internacionais sobre interoperabilidade, nos quais são deliberados os novos caminhos a serem seguidos para o protocolo, tendo como objetivo a sua divulgação e utilização.

### 2.1 As Primeiras Versões do Protocolo HL7

A primeira versão do HL7 surgiu em 1987 e não passou de uma apresentação de algo a ser implementado, uma proposta de protocolo. Efetivamente não foi utilizada por alguma instituição médica. Seu intuito era apenas apresentar um protótipo de um padrão de intercomunicação.

A segunda versão do HL7 surgiu um ano depois, em 1988. Esta já se apresenta como um protocolo a ser seguido pela comunidade de profissionais ligados à área de Informática para Medicina. A partir do release 2.3, lançado em 1998, passa a ser adotada pelos fabricantes de equipamentos e sistemas hospitalares no mundo e até hoje, mesmo após os novos releases, continua como a versão mais utilizada pelos fabricantes.

As versões 2.x do protocolo compreendem um conjunto de mensagens estruturadas em campos delimitados pelo sinal de “|” (*pipe*). Cada tipo de mensagem é denominado por um código de 3 letras. Por exemplo: ADT – admissão/demografia de paciente, ORU – resultado de exames, entre outros. Cada mensagem das versões 2.X (Código 01) é dividida em segmentos e cada segmento divide-se em campos. Cada segmento possui seu próprio código. Dentro das mensagens encontramos os delimitadores, destacando-se o “|” já mencionado, o <cr> para indicar fim de segmento, bem como os delimitadores “^” e “&” para indicar, respectivamente, componente e subcomponente de um tipo de dados complexo (ex: um nome de paciente composto por prenome e último sobrenome). O padrão de intercomunicação entre mensagens das versões 2.X (Figura 01) corresponde ao envio de mensagem para o servidor, que responde com uma mensagem “ack” indicando o recebimento da mensagem enviada pelo equipamento.

```
MSH|^~\&|ADT1|CADPACIENTE|LABADT|CADPACIENTE|20052242048||ADT^A01|MSGADT  
1200522420485|P|2.3<cr>  
EVN|A01|20052242048<cr>  
PID|||PATID1234 5 M11||Silva^Marcelo^Loyola^II^Dr.|Maria Ferreira|19451227|M||A|Rua Minas  
Gerais 55||(48)9833-8956|(48)398-6665|Português|S|lutheran 989896632| 96556|  
98989^055^65||U|São Paulo|0|3|Brasileiro||Brasileiro||<cr>
```

Código 01: Mensagem v2 do tipo ADT-A01

Na figura 02, podemos ver o esquema dos componentes ontológicos da versão 2 e na Tabela 01, vemos alguns exemplos de códigos de mensagens.

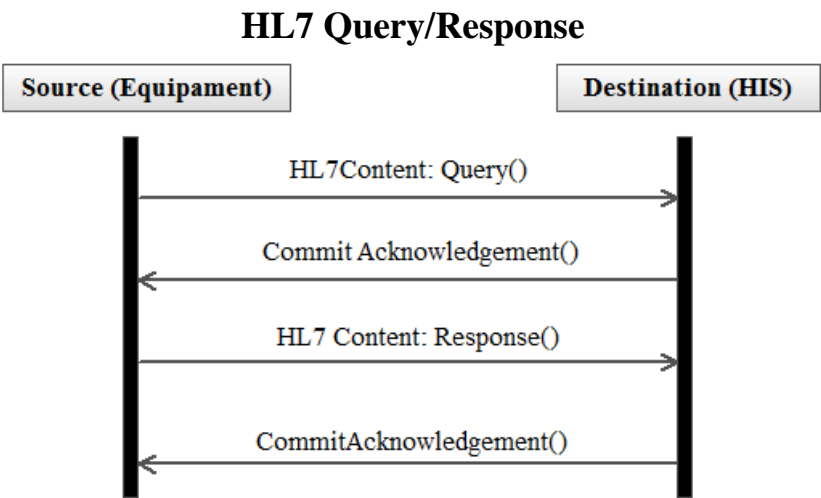


Figura 01: Esquema de comunicação HL7 v2.

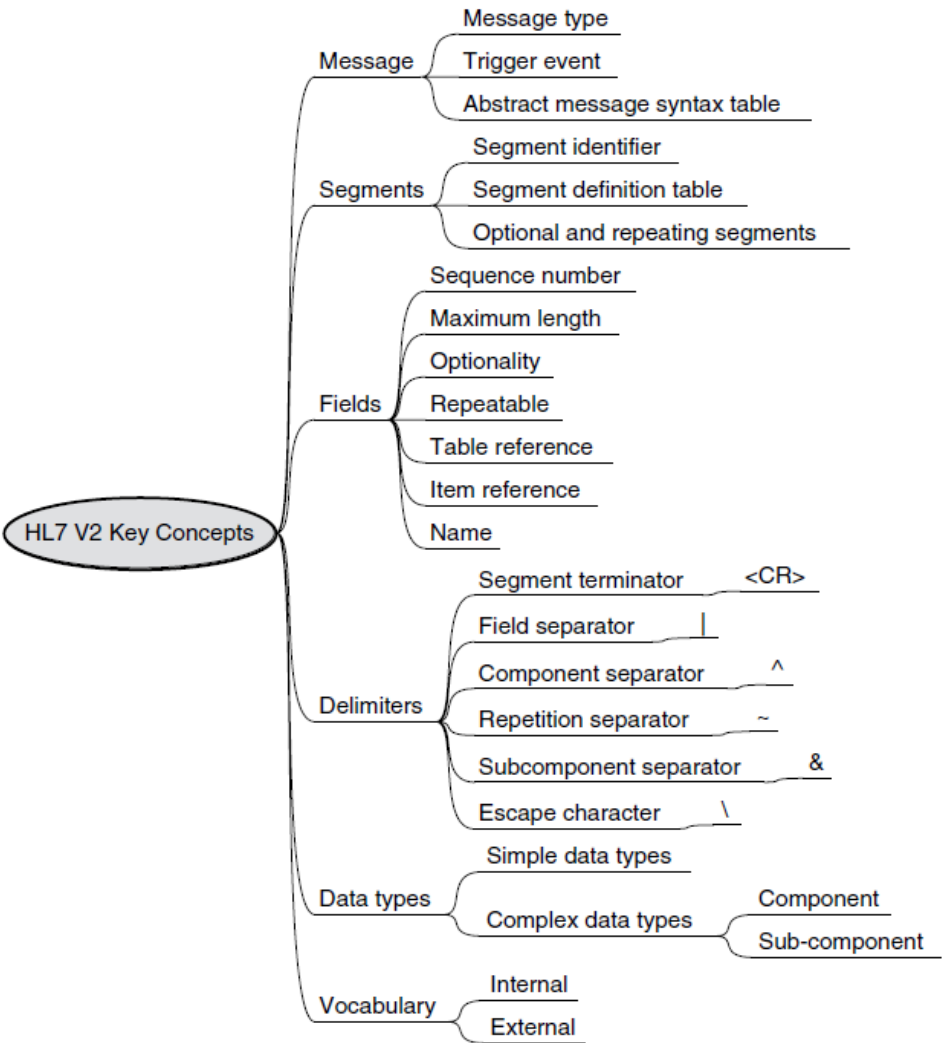


Figura 02 – Componentes de uma mensagem HL7 v2.x (BENSON,2010).

Nome	Descrição	Nome	Descrição
ACK	<i>General acknowledgement</i>	RAS	<i>Pharmacy Admin Message</i>
ADT	<i>ADT message</i>	RDE	<i>Pharmacy encoded order msg</i>
ARD	<i>Ancillary report</i>	RDR	<i>Pharmacy dispence info</i>
DFT	<i>Detail financial transaction</i>	RDS	<i>Pharmacy dispence message</i>
DSR	<i>Display response</i>	RGR	<i>Pharmacy dose message</i>
MCF	<i>Delayed acknowledgement</i>	RER	<i>Pharmacy prescript order info</i>
MSA	<i>Message acknowledgement</i>	ROR	<i>Pharmacy prescript order resp</i>
ORF	<i>Observation result</i>	RRA	<i>Pharmacy dispense acknowledge</i>
ORM	<i>Order message</i>	RRD	<i>Pharmacy admin acknowledge</i>
OBR	<i>Order acknowledgement msg</i>	RRE	<i>Pharmacy encoded order ack</i>
ORU	<i>Observ result/ unsolicited</i>	RRG	<i>Pharmacy give acknowledge</i>
OSQ	<i>Order status query</i>	GRY	<i>Query</i>
RAR	<i>Pharmacy Admin Info</i>	UDM	<i>Unsolicited Display message</i>

Tabela 01 – Quadro de principais tipos de mensagens v2.x.

A partir do *release 2.5* as mensagens passaram a poder ser codificadas em XML, oferecendo um recurso útil, por exemplo, para *web services*. Segue, no Código 02, um trecho de mensagem ADT codificada em XML.

```

<?xml version="1.0" encoding="UTF-8"?><ADT_A01>
<MSH>
<MSH.1>|</MSH.1>
<MSH.2>^~\&amp;</MSH.2>
<MSH.3><HD.1>MegaReg</HD.1></MSH.3>
<MSH.4><HD.1>XYZHospC</HD.1></MSH.4>
<MSH.5><HD.1>SuperOE</HD.1></MSH.5>
<MSH.6><HD.1>XYZImgCtr</HD.1></MSH.6>
<MSH.7><TS.1>20060529090131-0500</TS.1></MSH.7>
<MSH.9>
<MSG.1>ADT</MSG.1>
<MSG.2>A01</MSG.2>
<MSG.3>ADT_A01</MSG.3>
</MSH.9>
<MSH.10>01052901</MSH.10>
<MSH.11><PT.1>P</PT.1></MSH.11>
<MSH.12><VID.1>2.5</VID.1></MSH.12>
</MSH>
<EVN>
<EVN.2><TS.1>200605290901</TS.1></EVN.2>
<EVN.6><TS.1>200605290900</TS.1></EVN.6>
</EVN>
(...)
</ADT_A01>

```

Código 02 – Mensagem v2 em codificação xml (BENSON,2010).

## 2.2 A Versão 3 do Protocolo HL7

As versões 2.x trouxeram novos rumos para a interoperabilidade entre sistemas de dispositivos para uso hospitalar. No entanto, o seu desenvolvimento se deu por parte

de um grupo profissional essencialmente de clínicos com especialização em sistemas. Consequentemente, a arquitetura do padrão está sob um ponto de vista *end-user* e fortemente focado nos eventos inerentes ao mundo real dos usos de sistemas para saúde.

A sintaxe criada para as versões 2.x facilitou o desenvolvimento mais claro e ágil das aplicações de interfaceamento e promoveu maior liberdade no ponto de vista do cliente (ou seja, das instituições médicas em geral), que pôde trocar seus equipamentos hospitalares sem ter o alto custo de criar uma nova integração com o seu Sistema de Informação Hospitalar. Para as empresas e seus desenvolvedores, a adoção deste padrão único de interface oferece a possibilidade de um melhor gerenciamento de pessoal e recursos, uma vez que as empresas de equipamentos hospitalares podem otimizar o seu quadro de profissionais, ampliando-se o leque de desenvolvedores habilitados a trabalhar nos projetos de interface por não ficarem escravos de um padrão proprietário adotado por apenas uma empresa. Equipes agora são mais facilmente montadas. Os próprios desenvolvedores se utilizam desta vantagem de uma linguagem única, pois através da experiência podem evoluir suas capacidades, disseminando novas técnicas e contribuindo para seu aperfeiçoamento tanto profissional quanto da própria linguagem.

No entanto as mensagens do HL7 versão 2.X não estão sujeitas a restrições semânticas. Na realidade, na época do desenvolvimento das versões 2.X não foi sequer prevista a questão das restrições, deixando a cargo do DBA organizá-las no banco de dados. A falta de uma semântica bem definida traz um excesso de flexibilização que pode levar a ambiguidades quando falamos nas possíveis interpretações que um conteúdo possa ter. Por exemplo, sem a devida semântica é possível produzir um monitoramento uterino de Pacientes Masculinos! Assim, quando um equipamento tentar registrar este resultado em um banco de dados, se o mesmo não estiver habilitado a restrição de valores a serem inseridos, poderá aceitar a colocação do laudo deste monitoramento. As versões 2.X também não estão estruturadas para conceitos mais amplos de modelagem e orientação a objetos, assim como as tendências atuais que são utilizadas por aplicações Web.

Ainda assim, devido à grande solidificação de sua gramática e grande aceitação entre as empresas e os desenvolvedores, o HL7 versão 2.X continua sendo amplamente utilizado e ainda encontra-se em pleno desenvolvimento. Em paralelo à versão 2, surge em 2005 a versão 3 do HL7, criada pela norma ISO/HL7 21731. Esta versão consiste de uma nova metodologia de interface, baseada nos princípios da orientação a objetos, compreendendo não só os tipos de dados de uma classe, como os relacionamentos entre as mesmas. Os tipos de dados e os relacionamentos são descritos em um modelo geral de referência de dados, o RIM – *Reference Information Model*.

### **2.2.1 Reference Information Model – RIM**

O RIM é a espinha dorsal do HL7 V3. Ele consiste de um modelo relacional único, composto por um conjunto de classes genéricas. Tal modelo de referência foi proposto em 1992 pela ANSI/HISPP (*Healthcare Information Standards Planning Panel*), porém seu desenvolvimento de fato iniciou-se em 1997 através dos modelos preliminares construídos pelos membros da organização.

Os modelos são focados nas relações entre Eventos e Entidades. Cada Entidade participa de diversos eventos das mais variadas formas, as quais podem gerar consequências nas relações em que estão conectadas. O intuito é garantir a melhor

integração entre sintaxe e semântica dos dados, algo que não poderia ser implementado diretamente no HL7 versão 2.X.

O primeiro release do RIM, desenvolvido pela ANSI/HL7, surgiu em 2003. Em 2006, a ISO aprova o RIM e lança sobre a norma ISO-21731. Atualmente o RIM encontra-se no release 5 (R5 versão 2.41.3 – Ver Anexo I), publicado em 15/04/2013. O RIM procura organizar os dados dentro das melhores práticas de *design* e desenvolvimento de sistemas, tendo como objetivo manter o equilíbrio entre a estrutura sintática e a semântica dos dados. O modelo torna-se um ponto de partida para os desenvolvedores utilizarem, nos projetos de interfaces, os conceitos contemporâneos de desenvolvimento de sistemas para computação distribuída, como o SOA, por exemplo, bem como os outros paradigmas de computação existentes.

O modelo RIM divide-se em 3 estruturas de classes principais: Ações (*Acts*), Entidades (*Entities*) e Papéis (*Roles*). Em conjunto com estas três classes, temos três classes associativas: *ActRelationship*, *Participation* e *RoleLink* (Figura 03).

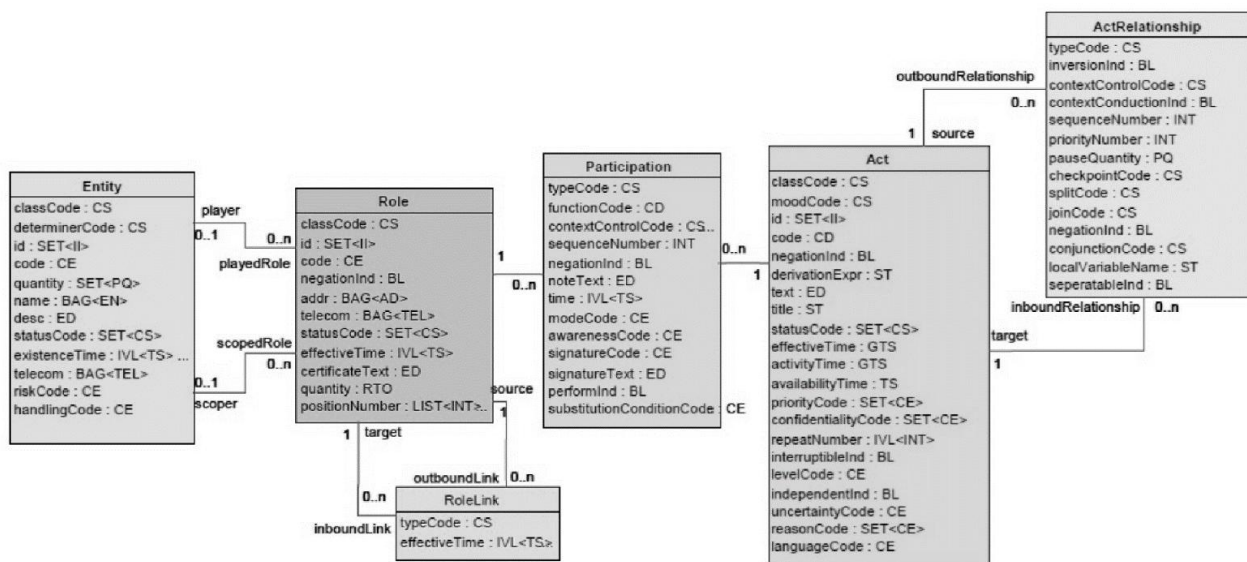


Figura 03 – Classes principais e associativas do RIM (extraído de: <http://www.hl7.org>).

### a) Acts

Atos são classes que armazenam eventos planejados através do RIM. A representação de um *Act* envolve a sua identificação (ex: um exame de sangue), os atores envolvidos (Médicos, pacientes, etc.), locais, equipamentos utilizados e demais objetos que envolvem o universo de um determinado *Act*. Pode-se afirmar que o ponto central de uma mensagem HL7 é o *Act*, pois através deste conseguiremos uma melhor visualização de rastreabilidade, que mostrará um determinado domínio do Modelo de referência RIM, isto é, podemos constatar com mais clareza a posição das diversas classes e sua representação dentro de uma conjectura abordada pelo RIM. Veja o Domínio das classes *Act's* na Figura 04.

Pode-se dizer que o *Act* é a classe mestra do RIM, pois todas as outras são instanciadas em função dos eventos do *Act*. A Classe *Act* é bem abrangente e dentre seus principais atributos estão o *moodCode*, que determina qual situação temporal um *Act*. Entende-se como situação temporal, o tipo de resultado emitido por um

equipamento. Se é temporário ou permanente. Exemplo: Um resultado parcial de cultura de urina é temporário pois sofre variações até sua conclusão dentro de 4 dias em média e um resultado de glicose é permanente pois o que é emitido pelo equipamento é o resultado em definitivo. e o *statusCode* (indica, por exemplo, um exame em andamento, paciente em alta, etc).

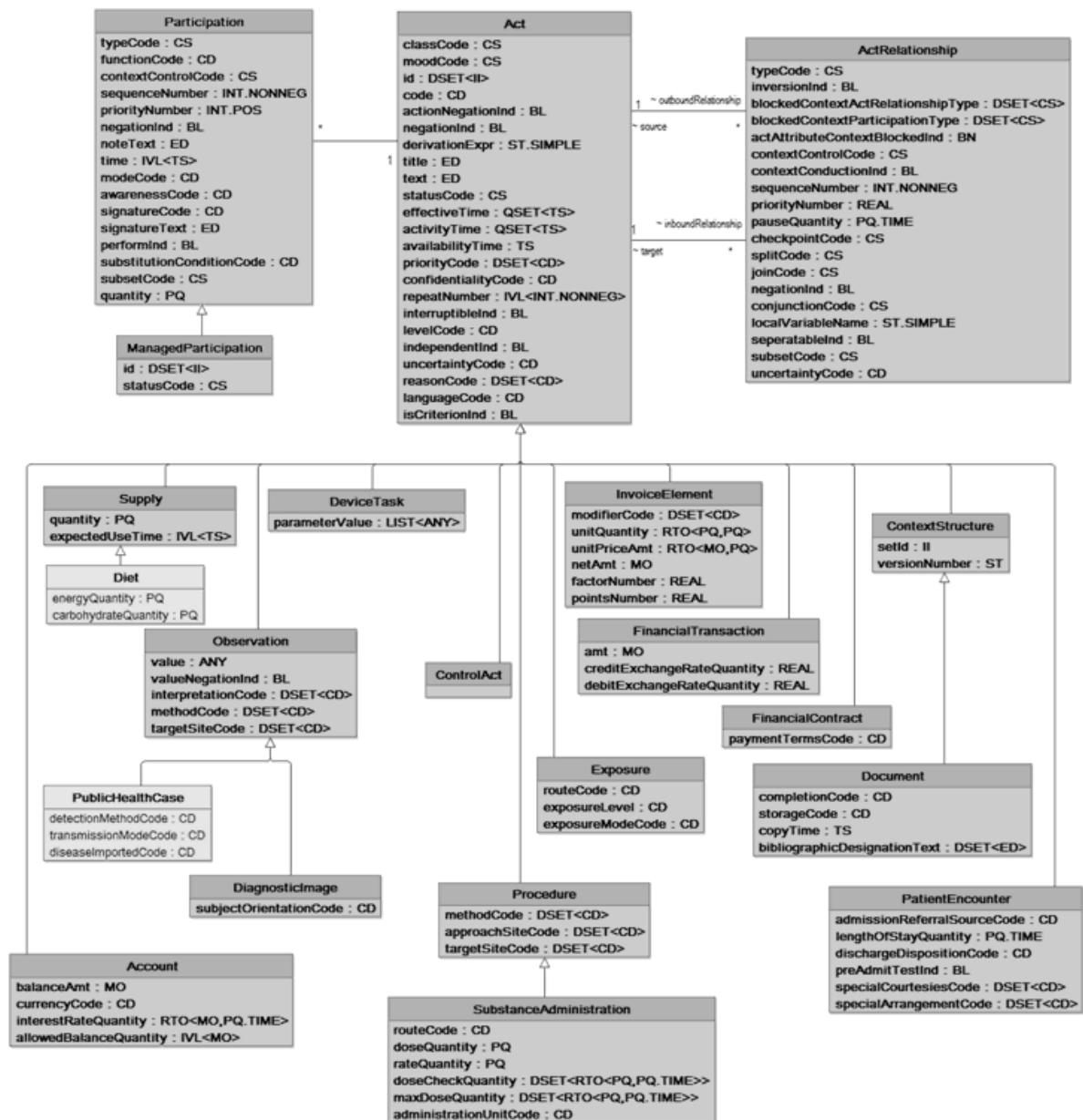


Figura 04 – Classe Act do RIM (extraído de: <http://www.hl7.org>).

## b) Entities

Entidades são classes que identificam os atores e demais objetos que operam os atos envolvidos no domínio do RIM que estamos modelando. Na Figura 05, vemos o diagrama das classes *Entity*.

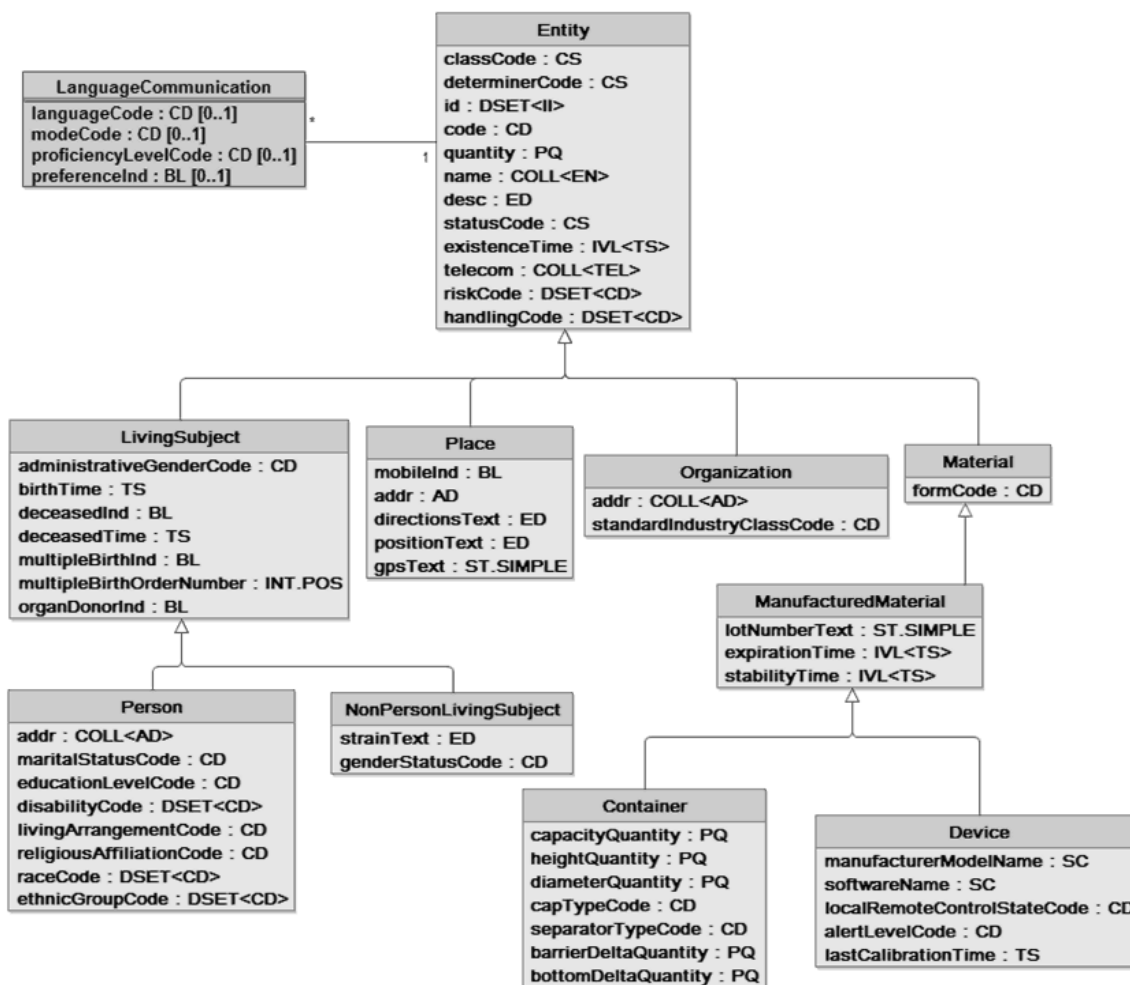


Figura 05 – Classe Entity do RIM (extraído de: <http://www.hl7.org>).

Cada entidade possui um *classCode* que indica o tipo de objeto representado, assim como um segundo atributo estrutural, o *determinerCode*, este que poderá individualizar determinado *classCode* de uma entidade. Por exemplo, em uma coleta para biópsia, temos um *classCode* para Tecido Biológico e um *determinerCode* para tecido pulmonar. As classes Entidades são representadas em quatro especializações principais a saber:

- i) **LivingSubject**: Representa o domínio das Entidades viventes (pessoas, animais, microrganismos, plantas). Convém informar que o HL7 é utilizado inclusive na área veterinária (em conjunto com o código de nomenclaturas SNOMED-CT);
- ii) **Material**: Corresponde a algum tipo de equipamento, insumo ou outros tipos de objetos concretos não-orgânicos. Podemos especificar um analisador de glicose em um *classCode* e um número serial deste analisador de glicose no *determinerCode*;
- iii) **Place**: Refere-se a lugares em geral: Cidade, bairros, endereços, etc. Catalogamos informações como endereços, por exemplo;
- iv) **Organization**: salas (por exemplo, sala de Raio-X), departamentos (ex: enfermaria de traumatismo-ortopedia), Instituições (ex. hospitais), Organizações



governamentais e ONGs, etc. Da mesma forma que as demais classes, temos código para a Organização e poderemos expandir para algum departamento da mesma.

Muito importante informar que alguns atributos de uma Entidade são encontrados também nas classes Papéis (*Roles*), como os atributos *id*, *code*, *name*, entre outros. Mas o que determinará se dado atributo será usado em uma *Entity* ou *Role* é o tipo de persistência, que indica se é uma escrita imutável ou não. Uma determinada pessoa possui um *id*, que pode ser o CPF, mas esta pode ser um profissional (por exemplo, um enfermeiro) que terá o seu *id* em *Role*. Repare que este enfermeiro pode se tornar o chefe da enfermaria, sendo seu *id* modificado para o código de chefia. Em resumo, um *id* em *Entity* sempre será o CPF, mas o *id* em *Role* pode ser modificado conforme as circunstâncias.

### c) Roles

Apresentado como a terceira classe principal do RIM, *Roles* são definidos como uma especialização de uma *Entity*. Na Figura 06 vemos o diagrama das classes *Roles*.

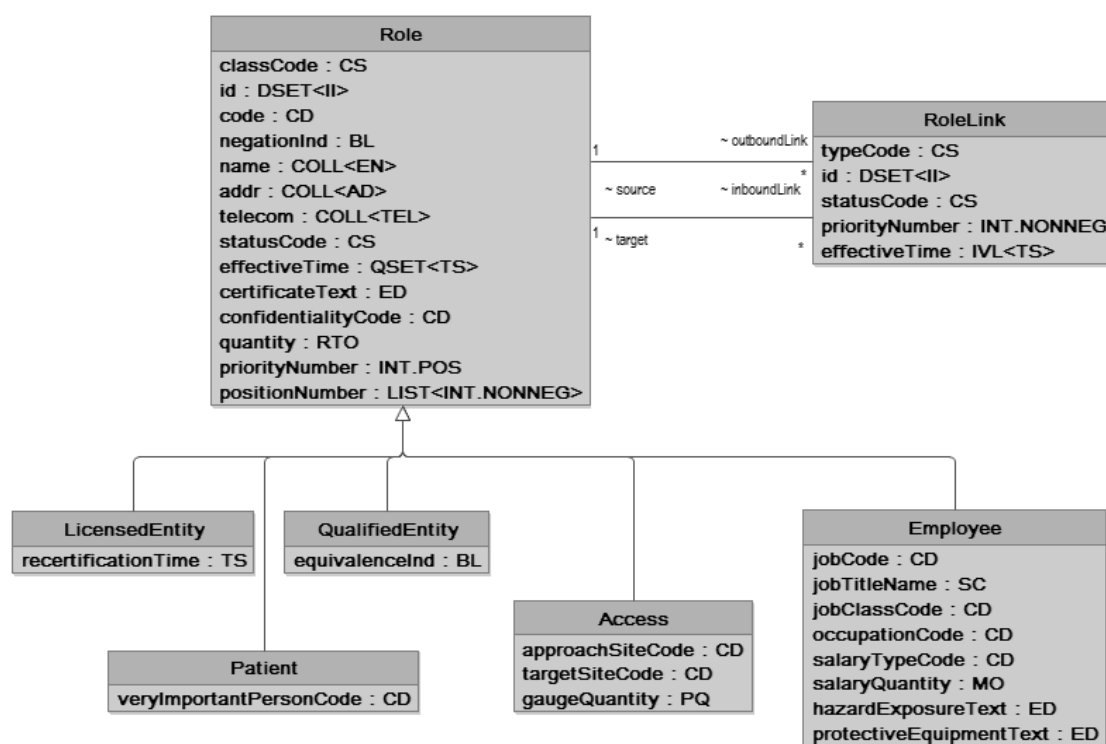


Figura 06 – Classe Role do RIM (hl7.org).

Desta forma, pessoas, por exemplo, podem ser profissionais, as quais terão este objeto *Role* vinculado. Estas mesmas pessoas poderão se tornar chefes. Desta forma, outro *Role* será vinculado a esta Entidade. Convém ressaltar que toda instância de uma *Entidade* está vinculada a uma instância *Papel* e este *Papel* existe porque sua criação foi justificada para ser usada por alguma Entidade.

Em conjunto com as classes principais, temos as seguintes classes associativas a seguir:

#### d) ActRelationship

Os *ActRelationships* representam os relacionamentos entre dois *Acts*, os quais podem ser em qualquer quantidade. Dentre os tipos de relacionamentos, especificados pelo atributo *typeCode* da Classe *ActRelationship*, podemos citar Composições de um ato em outro, requisições de teste, gerar documentação, entre outros. Na figura 07 vemos o diagrama das classes *ActRelationship*.

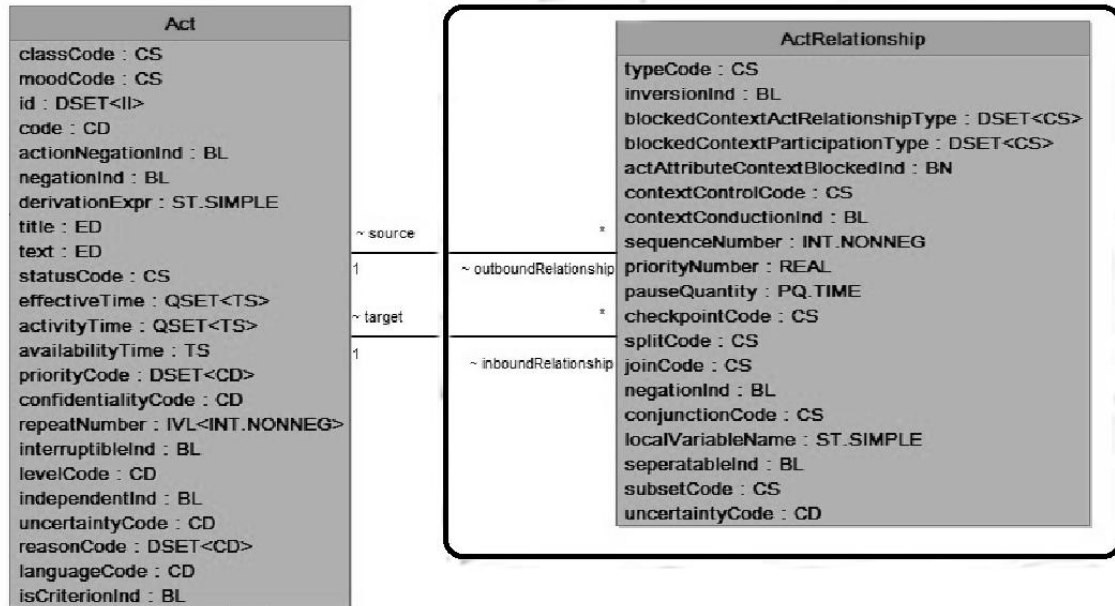


Figura 07 – Diagrama para classes ActRelationship (hl7.org).

#### e) RoleLink

As Classes *RoleLink* (Figura 08) estabelecem os relacionamentos entre dois papéis. Por exemplo Um profissional com o *Role* “Gerente” tem estabelecida a relação entre os profissionais do *Role* “Recepcionista” Com o *RoleLink* “Chefe\_Imediato”. O objetivo é estabelecer uma ligação direta entre diferentes Entidades através dos Papéis.

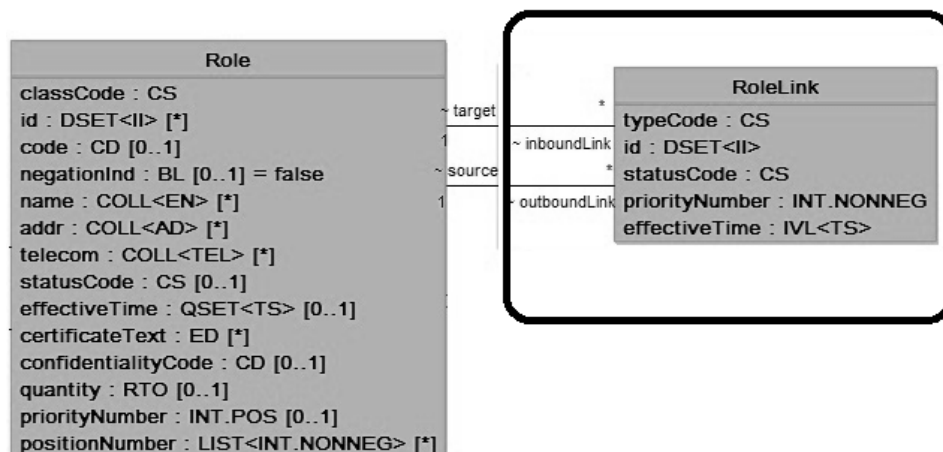


Figura 08 – Diagrama para classes RoleLink (hl7.org).

## f) Participation

As Classes *Participation* (Figura 09) definem o envolvimento de um *Role* dentro de um *Act*. Elas mostram de que forma uma entidade sobre um determinado Papel participa em um ato. Exemplificando, poderemos especificar de que forma uma *Entity* “Médico” dentro do *Role* “Clínico” atuam em um *Act* “Prescrição”. A associação descrita pelo *typeCode* pode variar desde o executante do *Act*, local do *Act*, participante do *Act* (em caráter passivo), entre outros.

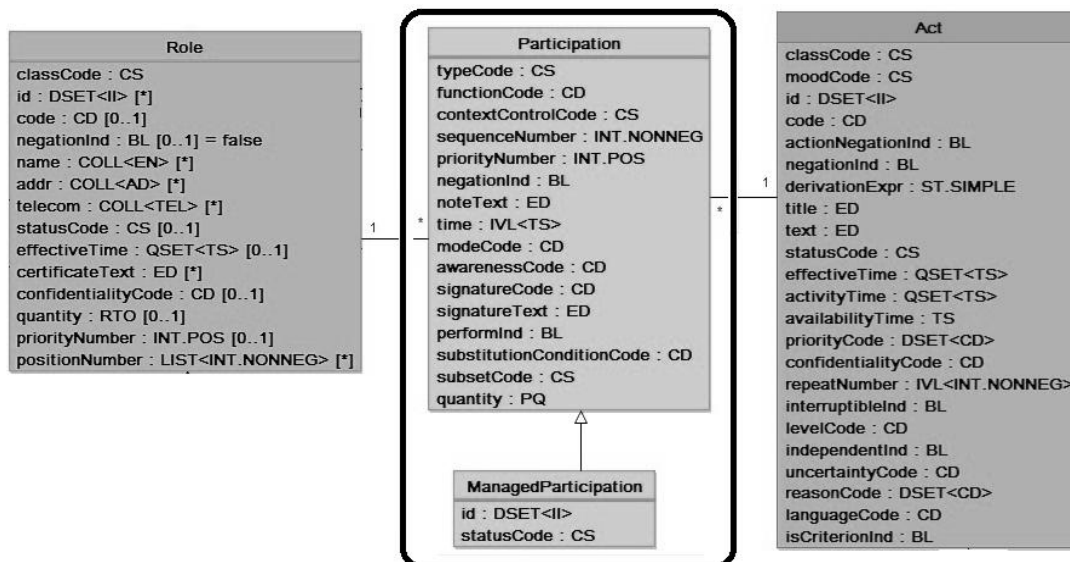


Figura 09 – Diagrama para classes Participation (hl7.org).

## 2.2.2. Abstrações/Refinamentos do RIM

Existem 3 refinamentos do RIM até chegar à mensagem HL7 V3 dentro de um conjunto de restrições definidas. A saber (Figura 10 - Do mais abstrato para o mais específico): DMIM (*Domain Message Information Model*), RMIM (*Refined Message Information Model*), HMD (*Hierarchical Message Description*) e as MT's (*Messages Type*).

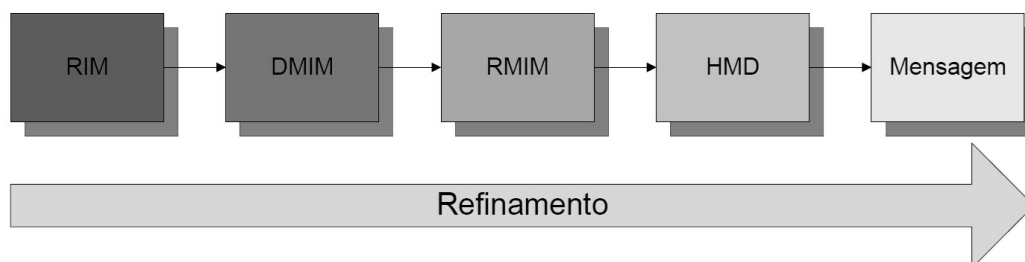


Figura 10 – Refinamento do RIM.

### a) DMIM (*Domain Message Information Model*)

O DMIM é o modelo extraído dentro de um primeiro nível de refinamento do RIM. Ele engloba o universo de objetos instanciados pertencentes a um determinado domínio. Entende-se como domínio o conjunto das classes e relacionamentos que realizam processos vinculados a um determinado objetivo, estipulado em um

minimundo. Por exemplo, suponhamos um conjunto composto pelos seguintes processos: coleta de amostras, processamento de exames, cadastro de exames, emissão de resultados. A este conjunto de processos e objetos participantes podemos atribuir a criação do Domínio “Laboratório” (Exemplo de DMIM em Anexo II).

### b) RMIM (*Refined Message Information Model*)

Corresponde a um modelo extraído dentro de um segundo nível de refinamento. Define-se como o conjunto dos objetos instanciados que participam de um determinado processo dentro de um domínio. Usando como exemplo o domínio Laboratório, pode-se extrair por exemplo o RIMM do domínio delimitado pelo processo “Processamento de Exames” (Exemplo de RMIM em Anexo III).

### c) HMD (*Hierarchical Message Description*)

É a forma tabulada de um RIMM apresentada na forma de um *Schema* xml, xmd ou em tabela. A sua existência deve-se à preferência que alguns desenvolvedores têm sobre a representação de um processo. Importante ressaltar que tanto o HMD quanto o RIMM representam a mesma informação. Uma representação de HMD em forma tabular pode ser vista na Tabela 02.

	Element Name	Card	Mand	Conf	Rim Source	of Message Element Type	Src	Domain	CS	Abst	Nt
	CDA (POCD_HD000040) Hierarchical Description										
	ClinicalDocument	0..1			Document	ClinicalDocument	N				
1	typeId	1..1	M	R	InfrastructureRoot	II	D				Default: @root="2.16.84.0.1.113883.1.3"
2	classCode	1..1	M	R	Act	CS	D	DOCCLIN	CNE		@extension="POCD_HD000040"
3	moodCode	1..1	M	R	Act	CS	D	EVN	CNE		Default: DOCCLIN
4	id	1..1		R	Act	II	D				Default: EVN
5	code	1..1		R	Act	CE	D	DocumentType	CWE		
6	title	0..1			Act	ST	D				
7	effectiveTime	1..1		R	Act	TS	D				
8	Confidentiality Code	1..1		R	Act	CE	D	x_BasicConfidentialityKind	CWE		
9	languageCode	0..1			Act	CS	D	HumanLanguage	CNE		Default: @codeSystem="2.16.840.1.113883.6.121"
10	setId	0..1			ContextStructure	II	D				
11	versionNumber	0..1			ContextStructure	INT	D				
12	copyTime	0..1			Document	TS	D				DesignNote: Deprecated
13	recordTarget	1..*			Act	SET<RecordTarget>	N				
14	typeCode	1..1	M	R	Participation	CS	D	RCT	CNE		Default: RCT
15	Context ControlCode	1..1	M	R	Participation	CS	D	OP	CNE		Default: OP
16	patientRole	1..1			Participation	PatientRole	N				
17	classCode	1..1	M	R	Role	CS	D	PAT	CNE		Default: PAT
18	id	1..*			Role	SET<II>	D				
19	addr	0..*			Role	SET<AD>	D				
20	telecom	0..*			Role	SET<TEL>	D				
21	patient	0..1			Role	Patient	N				
22	classCode	1..1	M	R	Entity	CS	D	PSN	CNE		Default: PSN
23	id	0..1			Entity	II	D				DesignNote: Deprecated
24	name	0..*			Entity	SET<PN>	D				

Tabela 02 – HMD em forma de tabela (extraído de: <http://wiki.siframework.org/CDA+-Merged+Publication+Considerations>).

#### d) MT (*Message Type*)

Os MT's são os diversos tipos de mensagens que representam a informação encontrada nos RIM's e HMD's. Seguindo o exemplo dado anteriormente, corresponde às mensagens dentro do RIM/HMD's "Processamento de exames". São descritos em xml.

#### 2.2.3 Nomeando Artefatos

Os Artefatos (Mensagens, Papéis, Modelos do RIM, etc) são nomeados segundo um padrão cujo formato é *SSDD\_AAnnnnnRRVV*, onde os primeiros quatro caracteres (*SSDD*) identificam os domínios, os dois caracteres seguintes (*AA*) correspondem ao tipo de artefato e os seis caracteres seguintes (*nnnnnn*) correspondem a um número de identificação atribuído pelo desenvolvedor para o artefato. Os demais caracteres não são obrigatórios. Os dois caracteres seguintes (*RR*) são referentes a uma Organização Afiliada ao HL7. O default é o código *UV* (Universal) e este é o mais amplamente utilizado. Algumas empresas costumam colocar seu próprio código, quando registrado no *HL7 International*. Finalmente, os dois últimos caracteres (*VV*) correspondem ao código de revisão do artefato. O padrão é *00* e este campo não costuma ser alterado. Um exemplo de mensagem para um exame sanguíneo de glicose está descrito abaixo no Código 03:

```
<POLB_IN224200 ITSVersion="XML_1.0" xmlns="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <id root="2.16.840.1.113883.19.1122.7" extension="CNTRL-3456"/>
  <creationTime value="200202150930-0400"/>
  <!-- The version of the datatypes/RIM/vocabulary used is that of May 2006 -->
  <versionCode code="2006-05"/>
  <!-- interaction id= Observation Event Complete, w/o Receiver
Responsibilities -->
  <interactionId root="2.16.840.1.113883.1.6" extension="POLB_IN224200"/>
  <processingCode code="P"/>
  <processingModeCode nullFlavor="OTH"/>
  <acceptAckCode code="ER"/>
  <receiver typeCode="RCV">
    <device classCode="DEV" determinerCode="INSTANCE">
      <id extension="GHH LAB" root="2.16.840.1.113883.19.1122.1"/>
      <asLocatedEntity classCode="LOCE">
        <location classCode="PLC" determinerCode="INSTANCE">
          <id root="2.16.840.1.113883.19.1122.2" extension="ELAB-3"/>
        </location>
      </asLocatedEntity>
    </device>
  </receiver>
  <sender typeCode="SND">
    <device classCode="DEV" determinerCode="INSTANCE">
      <id root="2.16.840.1.113883.19.1122.1" extension="GHH OE"/>
      <asLocatedEntity classCode="LOCE">
        <location classCode="PLC" determinerCode="INSTANCE">
          <id root="2.16.840.1.113883.19.1122.2" extension="BLDG24"/>
        </location>
      </asLocatedEntity>
    </device>
  </sender>
  <!-- Trigger Event Control Act & Domain Content -->
</POLB_IN224200>
```

Código 03 – Exemplo de mensagem HL7 v3 (retirado de [http://www.ringholm.de/docs/04300\\_en.htm](http://www.ringholm.de/docs/04300_en.htm) ).

Podemos ver em “*extension*”, localizado na tag “<*interactionId*>”, o valor *POLB\_IN224200*. Os quatro primeiros caracteres referem-se ao Domínio “Laboratório”. Os dois caracteres após o underscore correspondem a uma mensagem do tipo interação (mensagem com informações que serão levadas para um *web service listener*, por exemplo). Os seis caracteres restantes referem-se ao código do exame (por padrão os códigos referentes a exames e demais procedimentos laboratoriais são atribuídos segundo ao padrão identificador LOINC - *Logical Observation Identifiers Names and Codes*). O LOINC consiste em um banco de dados de códigos identificadores sobre procedimentos de patologia clínica, sejam exames, amostras, reagentes utilizados, metodologias, entre outros. Os códigos LOINC universalizam o significado de cada procedimento que será gravado em um banco de dados médico.

Outro código muito utilizado e ainda mais abrangente é o SNOMED-CT (*Systematized Nomenclature of Medicine - Clinical Terms*). Embora esteja em crescente utilização na Europa e EUA, sua utilização no Brasil ainda encontra-se no início. No entanto, cabe ressaltar que apesar da portaria 2073 do Ministério da Saúde, não existe ainda na prática um padrão único para interoperabilidade. O que ocorre comumente é a mescla do HL7 versão 2.X com padrões para faturamento (TISS - *Troca de Informações na Saúde Suplementar*, TUSS – *Terminologia Unificada em Saúde Suplementar*), classificação de doenças (CID - *Classificação Internacional de Doenças*) e, no caso da saúde pública, a tabela de faturamento ambulatorial (SIA-SUS - *Sistema de Informações Ambulatoriais/Sistema Único de Saúde*) e faturamento hospitalar (SIH- *Sistema de Informações Hospitalares/Sistema Único de Saúde*) do SUS. Em última análise, os códigos de faturamento são identificadores para os códigos dos procedimentos médicos em geral.

### 2.3 HL7 V3 CDA

O CDA (*Clinical Document Architecture*) é um padrão de implementação do HL7 v3 que estabelece a troca de informações HL7 entre sistemas na forma de arquivos XML de compreensível leitura, chamados de *Documento Clínico*. Entende-se como Documento Clínico (*Clinical Document - CD*) um resultado de exames, uma ficha de cadastro, laudo técnico ou documentos consolidados provenientes das informações extraídas de sistemas diversos.

O CDA pode conter figuras e arquivos multimídia, como áudio e vídeo. Como o padrão de mensagens utilizadas é o HL7 V3, o CDA segue todas as características fortemente tipadas de semântica e sintaxe desta versão do protocolo. Desta forma, obtemos documentos que são facilmente lidos por homens e processados por máquinas. A complexidade de um CD varia de acordo com a quantidade de informações presentes (até mesmo provenientes de outro CD). O Anexo IV ilustra o RMIM de um CDA.

O CDA é composto por 3 níveis descritos a seguir:

- **Nível 1:** Corresponde ao *Header* e o tipo de Documento Clínico (relatório de admissão, resultado de exames, entre outros). O *Header* provê identificação e tipo do documento, visando sua rastreabilidade (número do documento, versão do CDA), indica qual é o tema do documento, bem como informações gerais do paciente, como nome ou número de prontuário. O conteúdo do *Header* fica entre as tags <*ClinicalDocument*> e <*StructuredBody*>

- **Nível 2:** Corresponde ao *Body* do documento, ou seja, nele ficam as informações pertinentes ao tipo de documento clínico que estamos trabalhando. O nível é composto por um conjunto de informações que pode variar de 1 a n conjuntos. A cada conjunto de informações dá-se o nome de Seção (tag <section>). Cada seção pode conter um bloco de informações textuais (tag <text>), n entradas CDA (tag <entry>) e n referências externas (tag <reference>). Por exemplo, em um documento em que estejam descritos o cadastro completo do paciente, o resultado do último exame e medicamentos prescritos, teremos 3 seções: cadastro, exame e da prescrição. Quanto às referências externas, são os arquivos e outros objetos fora do escopo do HL7 que serão anexados ao documento, como, por exemplo, uma reprodução da foto do paciente.
- **Nível 3** – Localizado dentro do *Body*, refere-se ao conteúdo da tag <entry>. Desta forma, corresponde aos códigos HL7 V3 que utilizamos para extrair as informações para o CDA. Nas entradas constam inclusive os relacionamentos com outros CDA's que porventura possam ocorrer (CDA de um exame relacionado ao CDA do Prontuário do Paciente). No Anexo III, temos o RIMM do CDA.

Devido à natureza da tipificação sintática e semântica do CDA, são comuns a existência de templates criados pela Comunidade Médica para utilização nos processos realizados. Desta forma, poderemos aumentar ainda mais a interoperabilidade inclusive entre diferentes instituições quando as mesmas usam o mesmo template, bem como auxiliar na produção economizando tempo na criação de um formato de documento.

## 2.4 Considerações Finais

Atualmente, a questão da interoperabilidade utilizando o HL7 encontra-se por caminhos controversos. Apesar da grande abrangência pretendida pela Versão 3, esta ainda encontra alguns obstáculos, dentre os quais:

- a) Obstáculos na implementação de procedimentos mais complexos e novos, uma vez que a medida que novos procedimentos e técnicas na Medicina são lançados, uma nova representação semântico-sintática precisa ser desenvolvida e incorporada ao padrão V3;
- b) As mensagens V3 geradas podem atingir um grande tamanho, necessitando de espaços maiores no servidor para armazenar um conteúdo que é menor nas mensagens V2;
- c) Ainda sobre as mensagens V3, devido à solidez, simplicidade e grande número de profissionais habilitados da versão 2.x, em contraste com um número ainda pequeno de desenvolvedores capazes de implementar adequadamente a versão 3, o mercado em geral tem tomado a postura de ainda adotar a versão 2.x deixando a cargo dos *middlewares* a função semântica do protocolo entre os sistemas e mensagens. A ideia é deixar que realmente as pequenas ferramentas (*listeners*, *wrappers* e *parsers*) façam o trabalho de intercâmbio das informações, ainda que mesmo dentro de uma arquitetura SOA.

No entanto, assim como vem acontecendo com o HTML, passamos por um período de transição de paradigmas de protocolo. No caso do HL7, esta transição é mais delicada, pois as versões 2.x e 3 não são compatíveis. O protocolo V3, apesar de novo, vem com uma proposta interessante de garantir a integridade de uma informação.

O uso da V3 vem encontrando seu lugar por causa do CDA, que vem sendo adotado com considerável velocidade. Como O CDA já vem embarcado com uma estrutura mais compreensível e menos burocrática que as *V3 messages*, tem conquistado a simpatia da comunidade de desenvolvedores. Já estão disponíveis alguns templates, como o CCD (*Continuity of Care Document*) e o CCR (*Continuity of Care Record*) para o CDA e para mensagens respectivamente, estes muito utilizados nos EUA/Canadá.

Sobre as mensagens V3, cabe salientar a presença cada vez maior de *frameworks* que buscam simplificar a estrutura do código. Dentre os principais, destaca-se o FHIR (*Fast Healthcare Interoperability Resources*). Usando uma versão simplificada do RIM e um excelente suporte para o JSON, o FHIR pretende oferecer um caminho mais rápido para o desenvolvedor implementar uma variante do V3 mais enxuta e com suporte para o CDA, sem perder a essência do HL7. O FHIR tem encontrado um expressivo número de adeptos dispostos a utilizá-lo, tanto é que a própria Organização *HL7 International* já o incorporou no conjunto de seus componentes.



## 3. NoSQL e MongoDB

### 3.1 Dificuldades no Padrão Relacional de Dados

Durante as últimas quatro décadas, as propriedades ACID e os conceitos de “modelagem bidimensional” dos bancos SQL (*Ralph Kimball, 1997*) têm sido a indicação para resolver as necessidades de persistência dos sistemas de informação, contribuindo com a criação de *Data Warehouses* responsáveis por guardar grandes quantidades de dados de forma homogênea.

O uso constante do SQL em suas diversas distribuições, motivado pela facilidade de implementação, gerou um grande mercado que se guia pela premissa da concorrência entre os diferentes SGBD's, ainda que orientados por esta mesma linguagem. Tal situação reflete no próprio corpo de profissionais, que acabam se conduzindo a trabalhar e se especializar “no que já está consolidado”.

Entretanto, com o advento do conceito de *BigData*, a dinâmica das relações entre os dados toma formas cada vez mais intensas e variadas. Representar toda esta nova dinâmica na visão bidimensional de entidades e relacionamentos materializados pelas tabelas, pode tornar-se uma alternativa custosa no ponto de vista de implementação. Para emular esta nova abordagem dos dados no modelo relacional, são criadas nos SGBD's técnicas de armazenamento e provisão de disponibilidade, adaptando uma visão multidimensional de dados em um ambiente bidimensional.

No entanto, à medida que o universo de dados de uma organização cresce, aumenta também a dificuldade de implementá-las e realizar as prováveis manutenções tanto no banco como no serviço de BI e *Data Mining*, bem como a introdução de mecanismos gerenciadores de fluxo de dados que acabam por vezes reduzidos o tempo de acesso e processamento no lado Servidor.

### 3.2 Visão noSQL

O termo “noSQL” (*Not Only SQL*) tem sua origem ligada ao nome de um banco de dados aberto relacional criado por *Carlo Strozzi (1998)* que não utilizava o padrão SQL. Ao longo dos anos, as alternativas de banco de dados que fogem do paradigma relacional culminam no fim da primeira década dos anos 2000 com o surgimento do movimento noSQL.

Este movimento não tem como objetivo eliminar a utilização do padrão SQL por uma alegada consideração de obsolescência. Pelo contrário, o movimento pretende reformular o papel do paradigma relacional como solução única dentro do universo da persistência de dados. A proposta é apresentar diversas soluções de repositórios para resolver as necessidades advindas com o alcance cada vez mais amplo e descentralizado que informação vem passando, consequência do fenômeno da Internet.

A rede mundial de computadores por si só, levou a percepção de novas formas de se pensar os dados, bem como as informações que se podem extrair dos mesmos. Por este motivo, muitos preferem utilizar o termo noREL para os bancos que fogem do

modelo adotado pelo SQL. Os dados em um banco noSQL possuem as seguintes características:

**a) Escalabilidade horizontal:** Capacidade do banco distribuir-se em diversas máquinas que operam simultaneamente. O processamento dos dados ocorre de forma paralela e valoriza a aquisição de novos hardwares, ao contrário do investimento em upgrade em uma única máquina à medida que o repositório cresce. Ao utilizar diversos servidores ao invés de um único, o tempo de busca pode se reduzir significativamente dependendo do volume de dados contidos no repositório;

**b) Ausência de esquema ou esquema flexível:** A estrutura dos dados não se define por um esquema relacional, previamente concebido no momento da modelagem do banco. Isto facilita a implementação da escalabilidade horizontal dos dados. No entanto, a não existência de esquema enfraquece a integridade dos dados existentes;

**c) Suporte a Replicação:** A replicação de dados em um banco noSQL é mais natural devido às suas características de escala horizontal e a não existência das burocracias impostas pelos esquemas;

**d) API's de Repositório Simplificadas:** Em vez de grandes SGBD's, os bancos noSQL possuem API's simples, focadas nas operações mais elementares de manipulação dos dados, principalmente nas de consulta, permitindo uma grande rapidez na recuperação dos valores solicitados. A responsabilidade por estabelecer as relações entre os dados é transferida para as aplicações que acessam o repositório. Consequentemente desonera o trabalho do servidor e agiliza o fluxo na rede;

**e) Consistência Volátil:** Não se pode assumir que os dados estarão sempre consistentes a todo tempo devido à ausência da homogeneidade estrutural promovida pelos esquemas e relações. Notadamente, alguns bancos não relacionais promovem maior consistência que outros, devido à presença de um pequeno esquema intrínseco no próprio registro, como ocorre nos bancos orientados a documento (Ver Seção 3.3).

### 3.3 Teoremas BASE e CAP

Quando tratamos de banco de dados distribuídos, as características de atomicidade, consistência, isolamento e durabilidade presentes nas propriedades ACID, são substituídas por um novo conceito de propriedades de dados chamado de BASE (*Basically Available, Soft-estate, Eventually consistency*).

Esta nova estrutura promove uma nova visão de como pensar os dados, onde o foco passa a ser montar em implementações de serviços as regras para processamento dos dados, deixando ao banco o foco na disponibilidade dos dados. Desta forma sacrifica-se a consistência de dados em prol do desempenho e velocidade, características muito procuradas principalmente em sistemas via web.

A adoção das propriedades BASE retira assim uma enorme carga de responsabilidades no banco, eliminando esquemas, relações e demais estruturas agregadoras. Desta forma, damos um enorme grau de liberdade aos dados contidos no repositório.

Partindo-se da premissa levantada pelo esquema BASE, surge a criação do Teorema CAP (*Consistency, Availability e Partition Tolerance*), introduzida por *Eric A Brewer* em 2000. Em seus trabalhos, *Brewer* sugere que as características pretendidas em sistemas distribuídos sejam as seguintes (Figura 11):

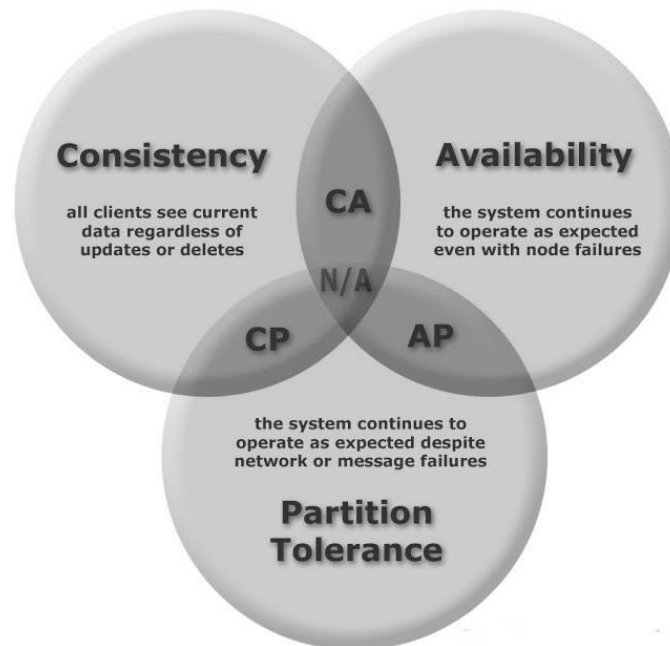


Figura 11 – Teorema CAP (extraído de: nosqltips.blogspot.com).

**a) Consistência:** Garantir a integridade das relações e/ou dos dados após sua execução, ou seja, uma vez que o registro tenha sido inserido, o mesmo se encontra imediatamente disponível. Os SGBD's que seguem as propriedades ACID possuem obrigatoriamente esta característica;

**b) Disponibilidade:** O sistema foi concebido e implementado de forma a garantir que esteja ativo durante um período pré-estabelecido ou mesmo indeterminado, sendo tolerante a falhas seja de hardware ou software e sua disponibilidade ocorre até mesmo quando no momento de atualização seja de sistema ou de infra-estrutura física;

**c) Tolerância de Particionamento:** Propriedade de um sistema continuar funcionando mesmo quando enfrentar problemas na rede, podendo se dividir em outras partições.

Contudo, segundo o teorema CAP, não se pode garantir a coexistência plena e simultânea das três características. A existência de uma estrutura com duas características de repositório, sacrificará a eficiência da terceira. Segundo estes conceitos, chega-se à definição de três tipos de sistemas (Figura 12):

**i) Sistemas CA (*Consistency – Availability*):** Os sistemas CA sempre estarão em funcionamento e prontos para ser acessados. No entanto, a falta de particionamento pode gerar interrupções no funcionamento do sistema, até que a manutenção no cluster falho se resolva. Os sistemas relacionais e o Neo4j seguem este conceito;

**ii) Sistemas CP (*Consistency – Partition Tolerance*):** Os sistemas CP são facilmente escaláveis e estarão sempre com seus dados íntegros. Entretanto quando um

dado é envolvido em uma determinada operação no banco, o mesmo dado poderá ficar indisponível durante a duração da atividade que o envolve. Ex: MongoDB, BigTable;

**iii)Sistemas AP (Availability – Partition Tolerance):** No caso dos Sistemas AP, os dados sempre estarão disponíveis para acesso/edição e o BD possui total escalabilidade à medida que o volume aumenta. Neste caso, como estes sistemas replicam muito os dados ou mesmo quando trabalham em regime de *read-consistency* (realiza o sincronismo dos dados replicados em um certo momento após a edição), isto pode levar a intervalos de inconsistências. Ex: Apache Cassandra, Riak.

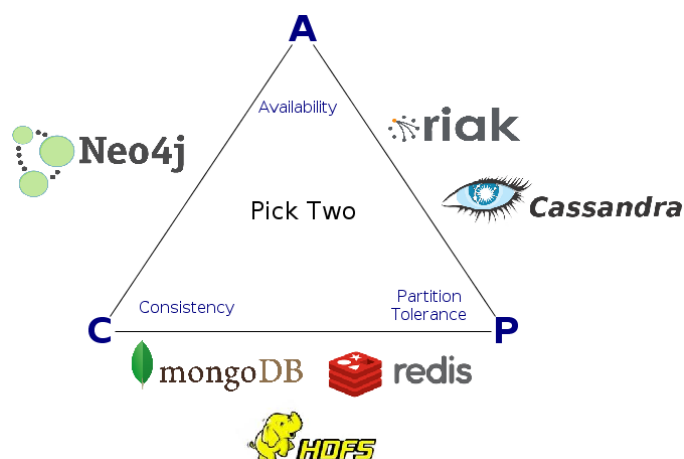


Figura 12 – Posição de alguns Bancos de Dados dentro do Teorema CAP.  
(extraído de: [blogs.mulesoft.org](http://blogs.mulesoft.org)).

### 3.4 Tipos de Armazenamentos em Bancos noSQL

Devido à natureza flexível dos bancos não relacionais, existe um enorme leque de abordagens de armazenamento dos dados. Contudo, os principais sistemas de bancos noSQL podem ser agrupados com a seguinte classificação quanto ao armazenamento dos dados (Figura 13).

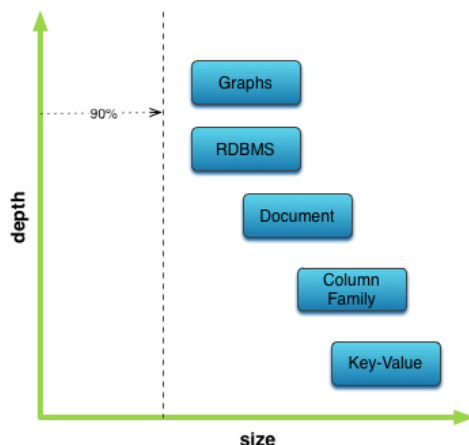


Figura 13 – Posição dos bancos de dados em função de estrutura de registros.  
(extraído de: <http://neo4j.com>).

**a) Orientado em Documentos:** O repositório consiste em registros em forma de documentos, onde reside uma coleção de dados. Informações detalhadas podem ser extraídas do conteúdo dos documentos e relações podem ser montadas com os outros registros através de campos controlados pelo sistema que acessa e mantém o banco de

dados. A Figura 14 mostra o registro no MongoDB de um documento no formato JSON sobre o cadastro de uma pessoa. O CouchDB é um outro exemplo de banco orientado por documento;

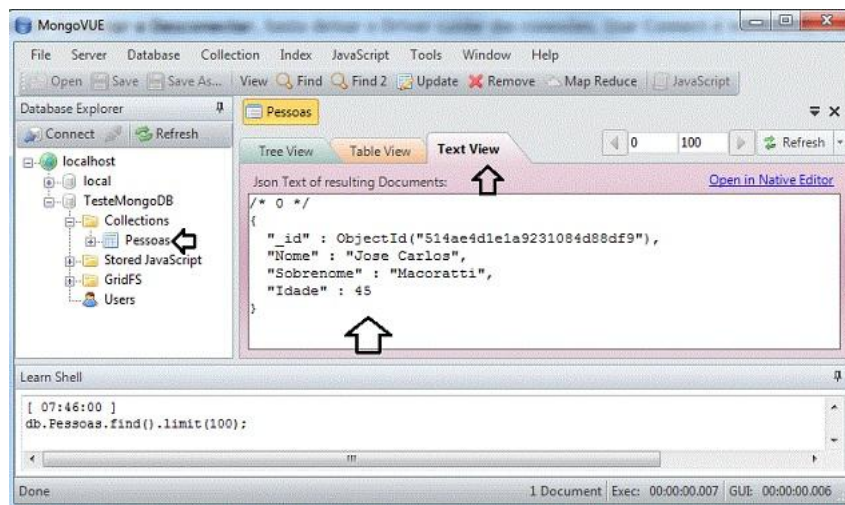


Figura 14 – Mongo VUE (Cliente para MongoDB) – Banco de dados baseado em documentos (extraído de: <http://www.macoratti.net/>).

**b) Orientado por colunas:** Os registros são indexados através da tripla Linha/Coluna/TimeStamp. As linhas e colunas são identificadas por chaves e cada versão de um dado é indexada pelo seu *TimeStamp*. As colunas são independentes entre si e pertencem a uma determinada “Família”, de forma que cada serviço forme tabelas de acordo com as suas necessidades, conferindo assim uma grande versatilidade aos sistemas que as utilizam. Na Figura 15 temos uma tabela formada por colunas das famílias “Company” e “Person”, que estão agrupadas de modo a estabelecer uma relação de Empregados de uma companhia. Outros exemplos de SGBD’s orientados por colunas são: BigTable, Hadoop e o Apache Cassandra;

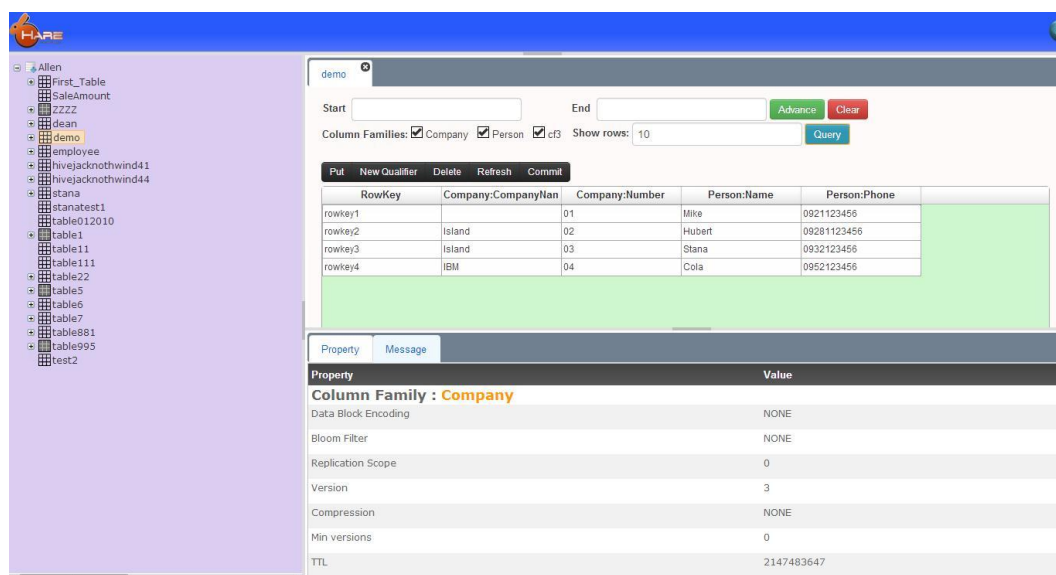


Figura 15 – HareDB (Cliente para bancos HBase) – Banco de dados orientado por colunas. (extraído de: <http://sourceforge.net/projects/haredbhbbaseclie/>).

**c) Orientado a Grafos:** Os bancos orientados a grafos, como o Neo4j (Figura 16), e o HiperGraphDB, seguem o princípio das operações sobre as estruturas em grafos. Nestes grafos, os nós são as entidades, as arestas são os relacionamentos e as propriedades representam os atributos. Além destas características, nestes tipos de bancos de dados temos funcionalidades comuns à Teoria dos Grafos, como busca no menor caminho, nós vizinhos, subgrafos, etc. Na Figura 16 temos várias entidades em um banco de dados sobre cinema, representando atores, críticos, filmes, entre outros conceitos. Cada entidade possui uma série de arestas que estabelecem relações com outras entidades. Devido à facilidade para estabelecer múltiplas relações entre as entidades através das arestas, a replicação de dados torna-se desnecessária bem como a própria redundância dos registros.

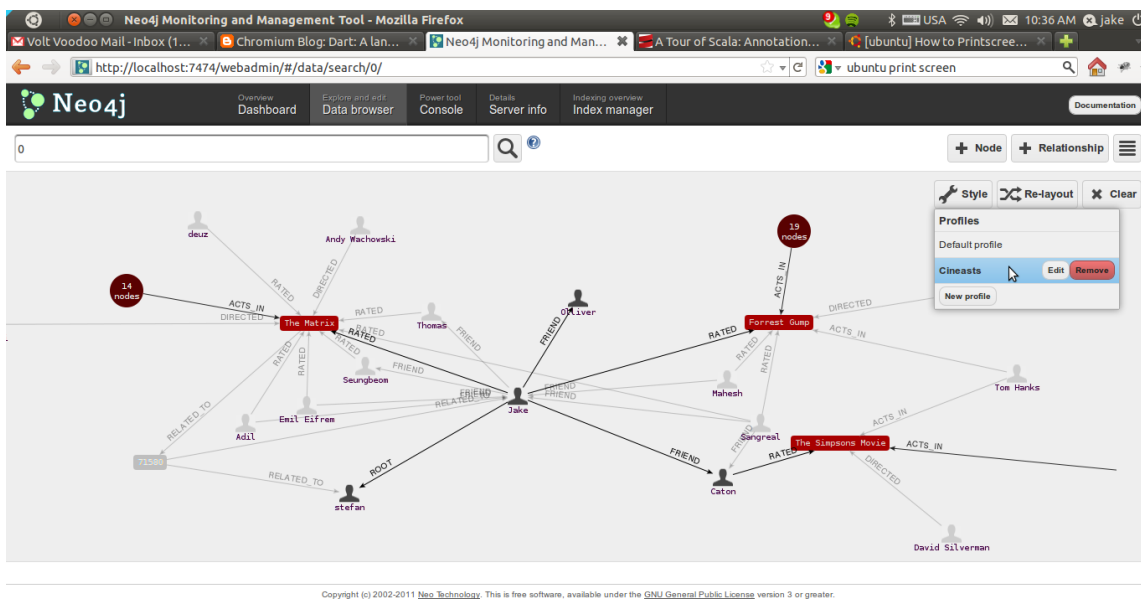


Figura 16 – Neo4j: Banco de dados baseado em grafos (extraído de: <http://data.story.lu/>).

**d) Orientado em Chave/Valor:** Os dados são armazenados em uma tabela *hash* onde os clientes acessam o conteúdo de um registro através da sua chave. Na Figura 17, apresenta-se um exemplo no Redis com a chave e o valor registrados no formato JSON. A granularidade fina dos dados favorece a escalabilidade e o acesso aos registros. Como os dados são acessados diretamente no banco, sem intermédio de estruturas agrupadoras, são muito utilizados como forma de cache de consultas em SQL (Abdallah, 2012).

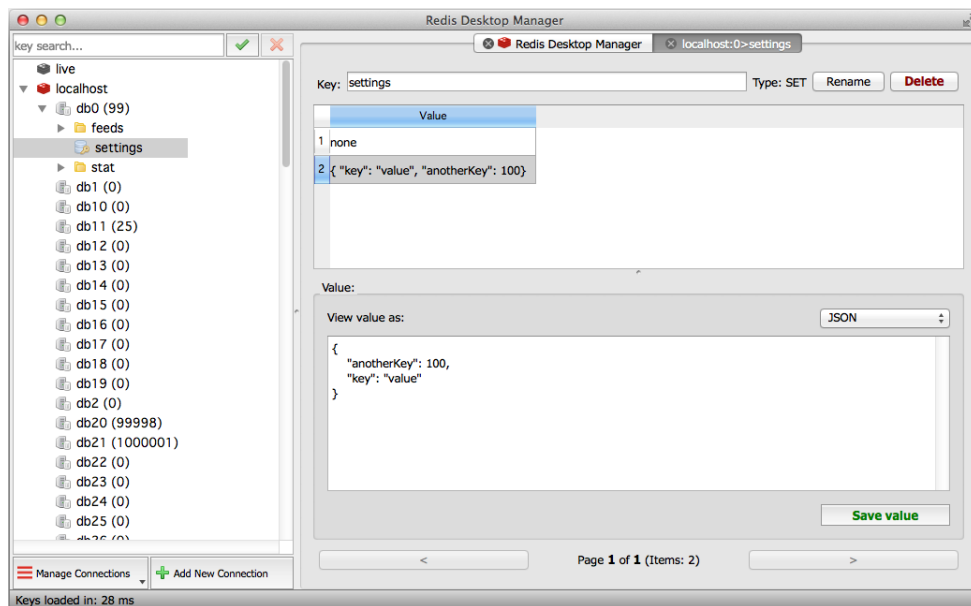


Figura 17 – Cliente Redis Desktop Manager: Banco de dados baseado em Chave/Valor – (extraído de: <http://www.macoratti.net/>).

### 3.5 MongoDB

Desenvolvido pela 10gen, teve sua primeira versão em 2009. Trata-se de um banco de dados não relacional, orientado a documentos, de código aberto e escrito em C++. O MongoDB encontra-se no padrão de sistemas de banco de dados *Consistency-Partition Tolerance*, valorizando a consistência de dados, capacidade horizontal de escala e se adaptando muito bem como solução para banco de dados distribuídos.

#### 3.5.1 Arquitetura de Dados

Os documentos são endereçados por meio de uma chave única e estão formatados em JSON. Entretanto, o armazenamento no banco ocorre em BSON (*Binary JSON*), que é a forma binária do formato JSON, com a vantagem de armazenar não só *Strings*, mas também outros tipos de dados (ex. *boolean*, *integer*, *double* e *date*). Todas as inserções de dados são realizadas de forma aninhada, ou seja, os documentos podem conter dados, *arrays* de dados e outros

A portabilidade, clareza e simplicidade do JSON torna o registro documental mais fácil de manipular e ler, de forma que apresenta-se como uma alternativa interessante para os sistemas clientes, sejam eles *web* ou não, pois oferece um código mais enxuto que o oferecido por arquivos XML, por exemplo. Podemos adicionar mais informações a cada documento sem precisar alterar as configurações do banco de dados, pois o mesmo não possui esquema.

Todos os documentos armazenados no MongoDB estão agrupados em *collections* (Figura 18). Cada *collection* corresponde ao conjunto de documentos que possuem a mesma finalidade informativa. Por sua vez, devido à liberdade gerada pela ausência de esquema, os documentos possuem total independência entre si. Cada registro no MongoDB é uma entidade livre, cujos dados mantém ligações semânticas apenas no contexto do próprio documento.

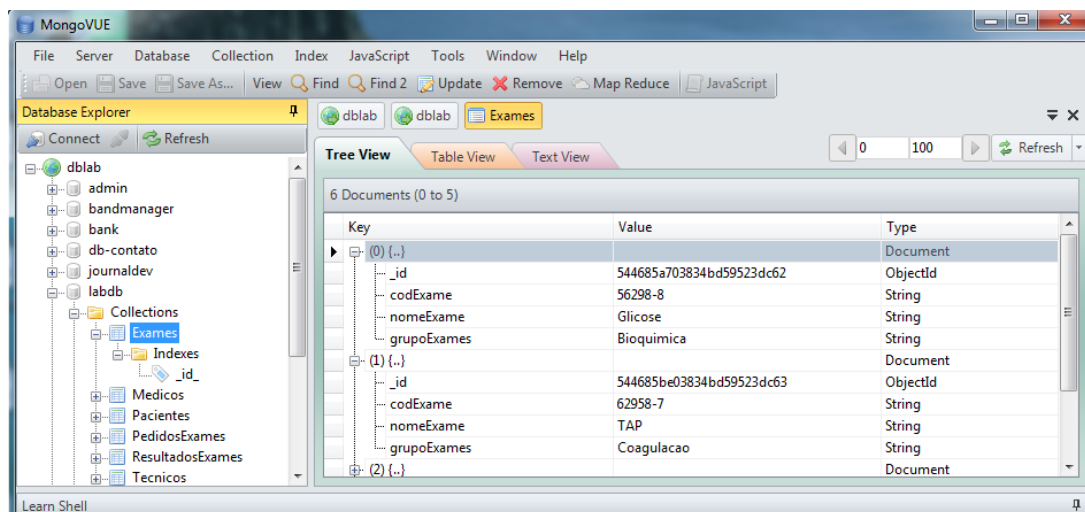


Figura 18 – MongoDB: Collection de documentos sobre exames.

Referências a outros documentos ou *collections* podem ser feitas no registro, porém recomenda-se a replicação dos dados afim de evitar a rigidez no banco, uma vez que é mais ágil procurar o valor solicitado no proprio documento. A replicação também reduz as chances de perda de dados em um banco distribuído por vários servidores, quando uma das máquinas apresentar falha. Outra forma de referência entre documentos se dá por meio de implementações por software, ou seja, funções que possam estabelecer relações entre valores de dois ou mais documentos. No entanto, tal referência ocorre apenas no escopo da aplicação e não é refletida na estrutura do banco de dados.

Exemplificando a flexibilidade de uma *collection*, uma dada coleção chamada “Pessoas” poderia ter documentos sobre pessoas distintas, uns com dados sobre data de nascimento e outros sem data de nascimento, mas com registro sobre o CPF sem que ocorram valores nulos. Na Figura 19, um exemplo análogo com uma coleção “Exames”.

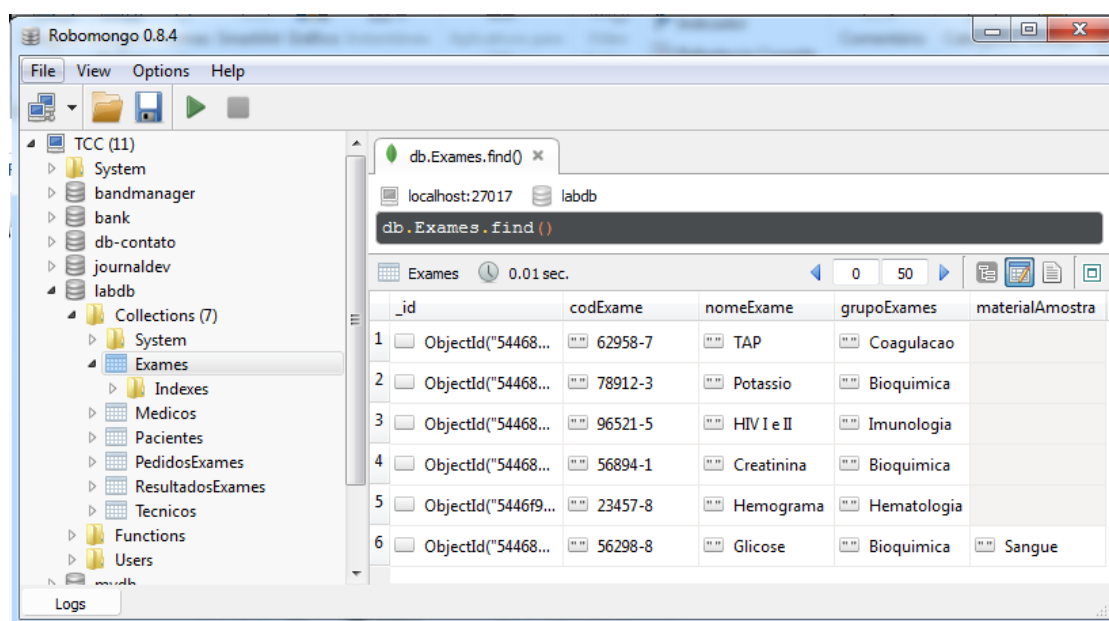
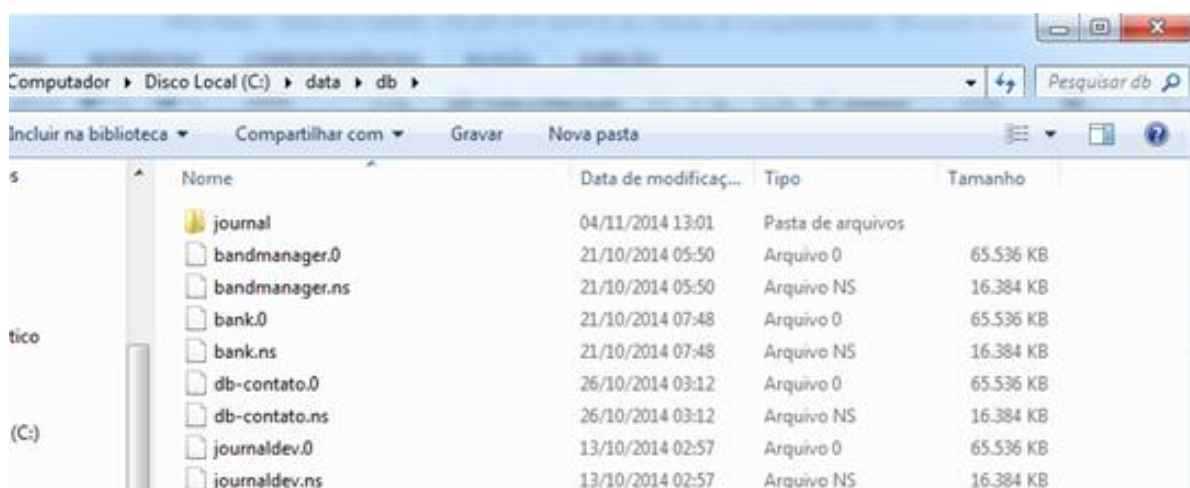


Figura 19 - MongoDB: Documentos com composição variada.



No MongoDB, assim como em todos os bancos orientados a documentos, os registros ocupam o espaço em disco referente apenas aos valores que foram incluídos. Cabe salientar que se necessário alterar/acrescentar campos replicados em documentos de uma coleção, cada documento terá que ser alterado de forma independente.

Cada banco de dados no MongoDB é composto pelos seguintes conjuntos de arquivos (Figura 20): (i) um arquivo com a estrutura “*nome-do-banco.ns*” onde encontram-se todos os *namespaces* e estruturas de indexação das *collections* do banco; e (ii) conjunto de arquivos com a estrutura “*nome-do-banco.i*”, sendo i um número inteiro. Ao criar um banco de dados, é criado o arquivo “.ns” e um segundo arquivo com final “.0”. O MongoDB reserva 64Mb para criação do banco de dados. Quando o número de registros e *collections* chega até a metade do espaço, é criado um segundo arquivo com o final “.1” com 128MB. A medida que o banco cresce, novos arquivos sequenciais são criados com o dobro do tamanho do arquivo anterior. A partir de 2GB, os demais arquivos possuirão este mesmo tamanho.



Nome	Data de modificaç...	Tipo	Tamanho
journal	04/11/2014 13:01	Pasta de arquivos	
bandmanager.0	21/10/2014 05:50	Arquivo 0	65.536 KB
bandmanager.ns	21/10/2014 05:50	Arquivo NS	16.384 KB
bank.0	21/10/2014 07:48	Arquivo 0	65.536 KB
bank.ns	21/10/2014 07:48	Arquivo NS	16.384 KB
db-contato.0	26/10/2014 03:12	Arquivo 0	65.536 KB
db-contato.ns	26/10/2014 03:12	Arquivo NS	16.384 KB
journaldev.0	13/10/2014 02:57	Arquivo 0	65.536 KB
journaldev.ns	13/10/2014 02:57	Arquivo NS	16.384 KB

Figura 20 - Lista com vários arquivos de diferentes bancos do MongoDB.

### 3.5.2 Replicação

A replicação (Figura 21) provê via sincronização dos dados entre vários servidores e o aumento da disponibilidade dos dados, através da sua cópia de um servidor primário para demais servidores secundários. O objetivo é a redução da possibilidade de perda de dados se porventura um servidor sofrer danos irreparáveis. A replicação é muito utilizada não só para backup como também para agilizar a busca por algum dado. É boa prática administrativa a réplica dos dados mais utilizados em um banco.

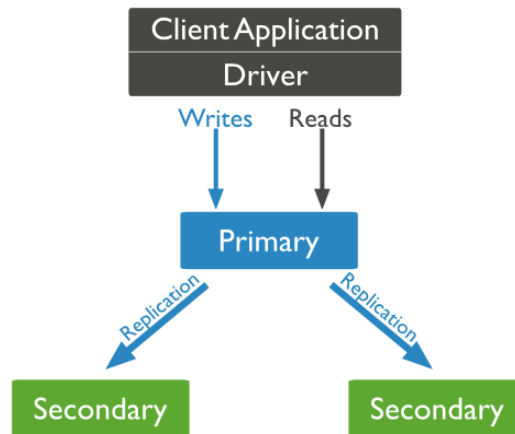


Figura 21 – Esquema de réplica de dados no MongoDB (<http://docs.mongodb.org/manual>).

### 3.5.3 Sharding

Denomina-se Sharding, o processo de distribuição de dados em várias máquinas, com o objetivo de aumentar a escalabilidade horizontal à medida que o banco de dados toma maior volume. Uma representação gráfica pode ser vista na Figura 22. Os componentes principais no processo de *sharding* são os seguintes:

- i) **Shards:** São os fragmentos do banco de dados que estão espalhados pelos clusters. Por estarem distribuídos em máquinas distintas, todo processamento de dados ocorre também de forma distribuída isto é, em paralelo, aumentando consideravelmente a rapidez nas operações de dados;
- ii) **Config Servers:** São responsáveis por armazenar os metadados do conjunto de clusters para os shards. Através dos locais armazenados pelos *Config Servers* chega-se ao cluster onde está armazenado o *sharding* de dados solicitado;
- iii) **Query Routers:** São instâncias do MongoDB carregadas nos clusters responsáveis por realizar a interface com os aplicativos-clientes. Eles enviam as solicitações de dados para os shards e retornam com os dados solicitados. Nos *Config Servers*, várias instâncias de *Query Routers* são carregadas afim de executar as solicitações de dados requeridas pelos Sistemas-Clientes.

Segundo o esquema apresentado na Figura 22, o Driver Mongo da aplicação, quando da realização de uma consulta, informa aos *Query Routers* os parâmetros pretendidos. Em seguida, os *Query Routers* procuram paralelamente nos *shards* pelos documentos que correspondem aos parâmetros. Esta procura em paralelo pode conferir um aumento significativo na velocidade de busca. Cabe ainda ressaltar que, conforme mencionado na seção anterior, é boa prática replicar os shards de modo a evitar perda de dados.

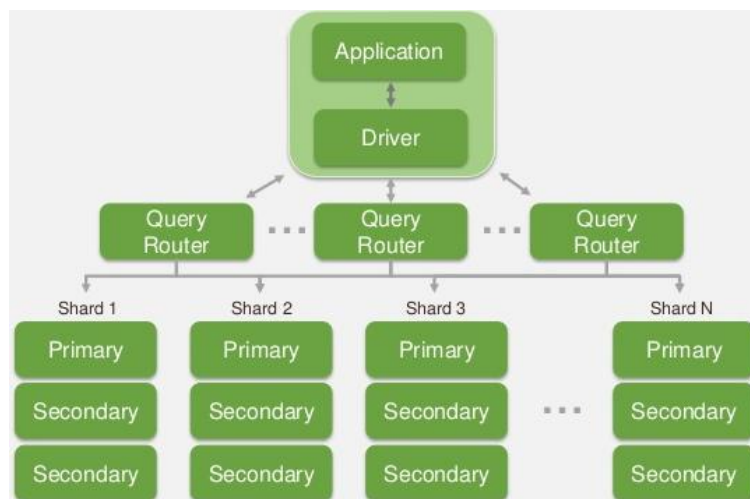


Figura 22 – Esquema de sharding no MongoDB (<http://docs.mongodb.org/manual>).

### 3.5.4 GridFS

Todo registro BSON no MongoDB possui uma limitação de tamanho, podendo atingir no máximo 16MB. Caso um registro ultrapasse este tamanho, torna-se necessário dividir em partes distintas e armazenar como documentos diferentes. O sistema de GridFS utiliza duas *collections*: uma para armazenar os metadados dos fragmentos de registros e outra *collection* para armazenar os arquivos relacionados nos metadados. O GridFS é uma alternativa indicada quando interessa apenas acessar um pequeno grupo de dados de um documento. Os dados mais procurados ficam em um dos fragmentos do Grid, podendo até utilizar replicação a fim de aumentar a disponibilidade.

### 3.6 Considerações Finais

É importante salientar que não existe um formato de banco de dados que resolva a totalidade das situações existentes com a máxima eficiência. Dependendo da aplicação, um determinado SGBD pode ser mais indicado que outro, da mesma forma que vemos no uso de linguagens e frameworks quando se desenvolve uma aplicação. Não raro, no universo de um sistema pode-se encontrar a coexistência de múltiplos SGBD's, cada qual responsável por resolver alguma determinada necessidade do sistema. Isto confere uma capacidade evolutiva ao sistema, caracterizada pelo que se chama de “persistência poliglota” (*Sadalage & Fowler, 2012*).

O MongoDB é uma excelente alternativa para repositórios cujo objetivo é encontrar conjuntos de dados relacionados bem definidos. Por exemplo, coleções de comentários sobre uma postagem em um blog, coleções de relatórios sobre qualquer tema, entre outros. Dentre os casos de uso do MongoDB mais conhecidos podemos citar: **Foursquare**, para a geolocalização dos check-ins cadastrados; **Bit.ly**, como repositório dos links cadastrados; **Forbes**, em seu sistema de CMS (Content Management System); e **CartolaFC**, na aplicação do “mural” (*chat* entre os participantes).

As principais **vantagens** do MongoDB são: (i) **Alta performance**, pois as consultas retornam tudo o que é necessário saber sobre o documento; (ii) **Esquema aberto**, que favorece os processos de cache, migrações e oferece flexibilidade das estruturas de registros; (iii) **OpenSource**, que resulta em menores gastos com soluções

de persistência de dados; e (iv) **Capacidade de distribuição**, possibilitando alta disponibilidade e performance em bancos de dados distribuídos.

Como **desvantagens** do MongoDB podemos citar: (i) **Uso considerável de espaço em disco**, pois a natureza do registro por documentos utiliza grande espaço de disco. Convém analisar o custo de instalação de discos de maior capacidade, uso de RAID ou a distribuição do banco em diversas máquinas; (ii) **Grande consumo de memória RAM**, pois as estações que operam os sistemas clientes do MongoDB necessitam de um considerável espaço de memória RAM para armazenar os índices e arquivos usados no processamento de dados do banco; e (iii) **Ser muito dependente da implementação para queries mais elaboradas**, uma vez que queries mais específicas, que envolvem referências entre vários documentos, precisam ser devidamente construídas no software clientes do banco.

No Capítulo 4, utilizamos o MongoDB como repositório para manter conjuntos de resultados de exames.

## 4. Exemplo de Implementação – Laboratório de Análises Clínicas

### 4.1 Proposta de Desenvolvimento

Apresentaremos a implementação de um sistema para demonstrar a capacidade de integração entre o MongoDB e o protocolo HL7. O sistema consiste de um conjunto de programas que alimentam uma base de dados MongoDB com resultados de exames colhidos através de um programa que simula um equipamento médico (gerador de resultados de exames). Os programas são brevemente descritos a seguir, sendo detalhados adiante neste capítulo:

- **Maquinas Virtuais:** Pequenos softwares que realizam tarefas simuladas quando um equipamento de análises clínicas envia os resultados no formato HL7 CDA para uma pasta de resultados emitidos.
- **LabResults:** Programa responsável pela integração de dados entre os equipamentos de exames e o MongoDB. Monitora a pasta de resultados emitidos em busca de novos resultados para serem enviados ao banco de dados;
- **LabQuery:** Ferramenta destinada à procura de resultados de exames registrados no MongoDB, segundo diferentes tipos de critérios de busca.

### 4.2 Casos de Uso

Os casos de uso identificados para cada ferramenta são brevemente descritos a seguir nas aplicações LabResults (Figura 23) e LabQuery (Figura 24):

#### a) LabResults

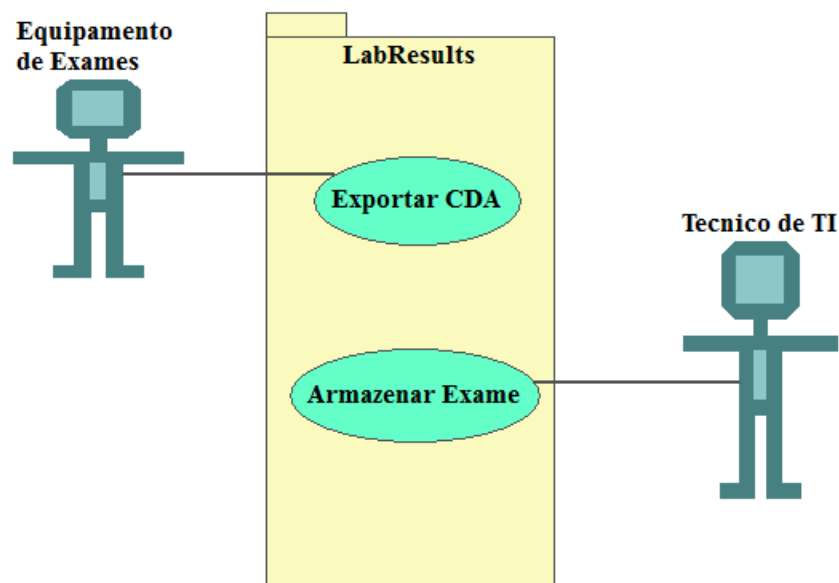


Figura 23 Diagrama de Casos de Uso da Aplicação LabResults.

- **Exportar CDA:** Este caso de uso consiste em mover o CDA contido em uma pasta que simula o repositório de um equipamento de exames. Cada máquina virtual move os arquivos HL7 CDA para a pasta de resultados “*inbox*” do servidor.
- **Armazenar Exame:** Este caso de uso consiste em processar os arquivos de documentos enviados pelos equipamentos de exames, transformando o HL7 CDA para o formato JSON, que será enviado ao banco de dados do Mongo. O Técnico de TI aciona o botão “Iniciar”, o que dispara este processo. Os CDA’s de exames contidos na pasta de resultados emitidos são processados e enviados ao MongoDB como registros de documentos. Após o envio, o CDA é movido para uma pasta de resultados enviados.

#### b) LabQuery

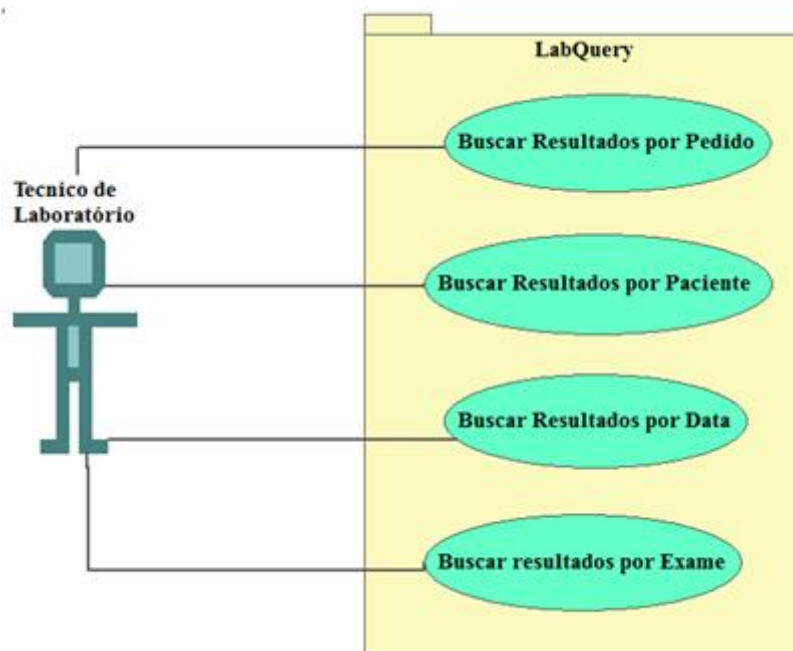


Figura 24 – Diagrama de Casos de Uso da Aplicação LabQuery.

- **Buscar Resultados por Pedido de Exame:** Este caso de uso consiste na procura de resultados de exames parametrizados pelo número de pedido de exame. Ao encontrar registros que correspondam ao pedido dado, é exibida uma tabela com os campos comuns a todos os Exames (numero do pedido de exames, data do exame, nome do Paciente, nome do exame e médico solicitante). Ao clicar na linha sobre o exame desejado, aparece na tela de saída o resultado proveniente do registro.
- **Buscar Resultados por Paciente:** Este caso de uso consiste na procura de resultados de exames parametrizados pelo número de prontuário ou nome do paciente. Os exames são exibidos na mesma forma que no caso de uso anterior.
- **Buscar Resultados por Data:** Este caso de uso tem o objetivo de procurar os exames realizados em uma determinada data ou período de dias. O sistema também busca resultados a partir ou até uma determinada data.

- **Buscar Resultados por Exame:** Este caso de uso tem como objetivo encontrar resultados de exames através do nome do Exame. Pode-se adicionar parâmetros por valores pretendidos, seja um determinado valor ou uma faixa de valores.

### 4.3 Diagramas de Classes

Apresentamos abaixo um diagrama com as classes utilizadas na implementação do LabQuery (Figura 25). A estrutura dos demais softwares é muito simples e, portanto, não será detalhada neste documento.

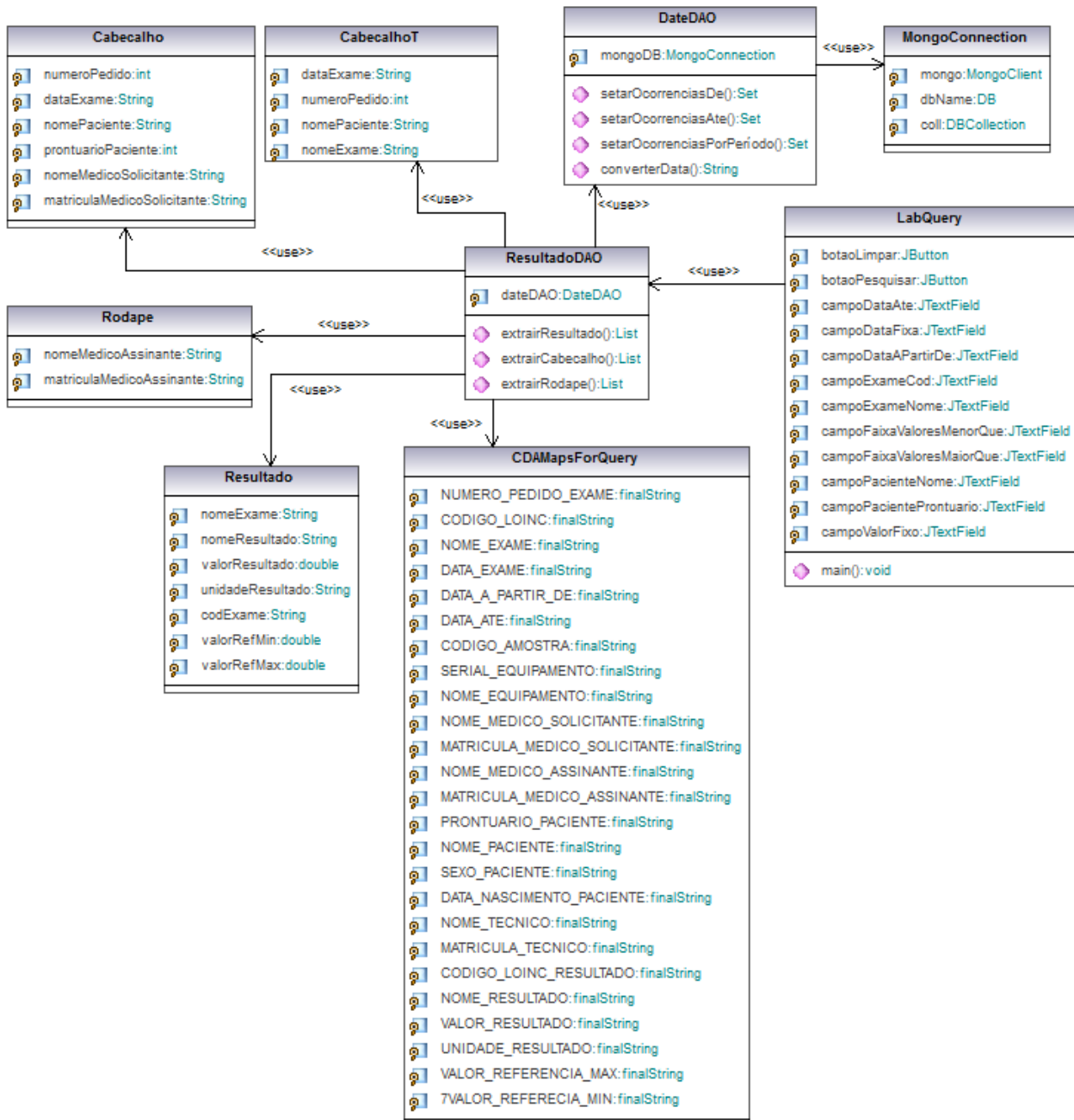


Figura 25 – Diagrama de Classes da Aplicação LabQuery.

- a) **LabQuery** – Classe principal, contendo o formulário e a janela principal da aplicação;
- b) **Cabecalho** – Classe responsável por formar os cabeçalhos dos resultados de exames que são exibidos na tela;
- c) **Rodape** – Classe destinada a montar o rodapé dos resultados de exames que são exibidos na tela.
- d) **Resultado** – Classe responsável por montar a estrutura de corpo de texto referente ao resultado que sai na tela de resultados;
- e) **ResultadoDAO** – Classe responsável por acessar os dados no banco *labdb* para montagem do corpo do resultado;
- f) **DateDAO** – Classe destinada a acessar os dados no banco *labdb* para operações relativas a manipulação e conversão de datas;
- g) **MongoConnection** – Classe destinada a criar uma conexão com o banco de dados MongoDB;
- h) **CabecalhoT** – Classe responsável por montar os cabeçalhos dos resultados de exames que são exibidos na tabela de resultados;
- i) **CDAMapsForQueries** – Classe contendo constantes que correspondem ao caminho dentro do documento que refere a um determinado dado (por exemplo, o nome de um exame). Destina-se a promover um bom encapsulamento dos dados e a aumentar a clareza quando na consulta ao banco de dados.

#### 4.4 Ferramentas Utilizadas

Foram utilizadas nesta implementação as seguintes ferramentas:

##### 4.4.1 Linguagens e IDE's

a) **Java 1.8** – É uma plataforma computacional e linguagem de programação compilada/interpretada. O código Java é compilado para uma JVM (*Java virtual Machine*) instalada para um determinado Sistema Operacional. Quando da execução do programa, a JVM interpretará os códigos, gerando um programa para uso no Sistema Operacional instalado;

b) **NetBeans** – IDE (*Integrated Development Environment – Ambiente de Desenvolvimento Integrado*) multilinguagem. Suporta uma grande quantidade de pluguins e linguagem de programação.

##### 4.4.2 Bibliotecas e Ferramentas de Código

a) **Swing** – Biblioteca para criação de GUI's (*Graphic User Interface*) para aplicativos Java. É um dos principais gerenciadores de forms para Java Desktop.



**b) Maven** – É uma ferramenta de automação de compilação utilizada para construir e gerenciar projetos não só em Java como em outras linguagens (ex: Ruby, C#). Constitui-se de plug-ins de automação (*builds*) e de um arquivo XML, denominado POM (*Project Object Manager*), que é responsável por gerenciar as dependências e plug-ins utilizados no projeto. Os plug-ins e suas dependências são baixados de repositórios Maven, destacando-se o *mvn Repository* (<http://mvnrepository.com/>);

**c) Java MongoDB Driver** – Biblioteca Maven para criação de aplicativos que utilizam o MongoDB. Possui recursos para criação e edição de registros, coleções e instâncias do MongoDB.

#### 4.4.3 Programas acessórios

**a) MongoChef** – Desenvolvido pela 3T Software Labs, é um cliente *freeware* (para uso não comercial) para MongoDB. Permite, a criação e edição de banco de dados, coleções e registros do MongoDB, bem como visualização dos dados na estrutura JSON, na forma de tabelas ou árvores de registros (Figura 26).

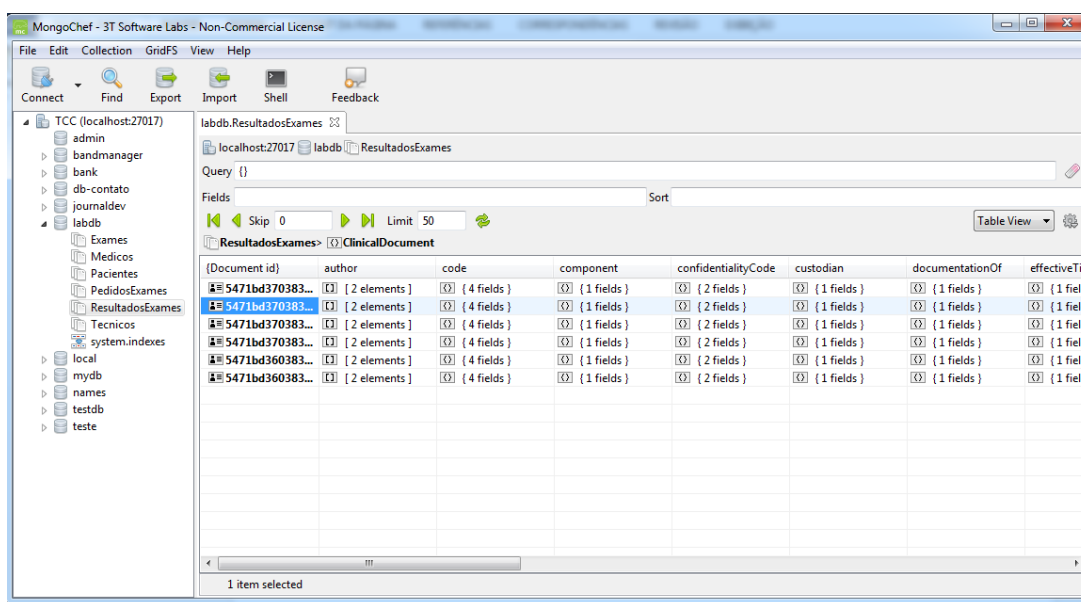


Figura 26 – Cliente MongoDB MongoChef.

#### 4.5 Funcionalidades do LabResults

O propósito desta aplicação (Figura 27) é receber, converter e enviar os resultados de exames para o banco de dados. O programa vasculha a existência de documentos HL7 CDA dentro do diretório especificado no campo “Entrada de CDA”, analisa cada arquivo, converte em uma string JSON e envia para registro no banco e na coleção especificados nos campos “Nome do DB” e “Nome da Collection”. Após o processamento, o documento é movido para a pasta especificada no campo “Saída de CDA”.

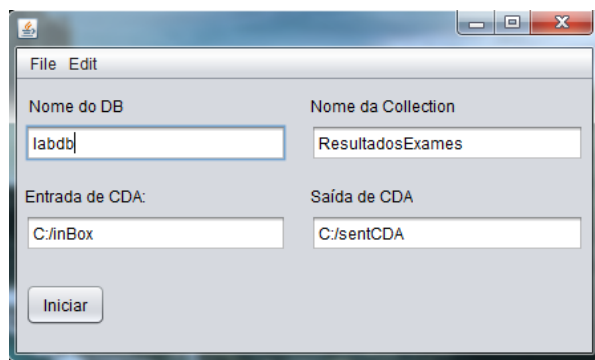


Figura 27 – Tela principal da aplicação LabResults.

## 4.6 Funcionalidades do LabQuery

O propósito da aplicação é exibir os resultados de exames baseados nas consultas seguindo os critérios propostos em cada opção (Figura 28). De uma maneira geral, as consultas são realizadas na *Área de Queries*. Caso exista registros correspondentes à consulta, os cabeçalhos dos resultados são exibidos em uma tabela correspondendo a um exame por linha. Ao clicar na linha desejada, aparecerá o resultado na tela de *Exibição de Resultado*. Segue uma breve descrição das funcionalidades.

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999265	15/04/2014	Valeria	GLICOSE	Douglas
999119	22/03/2014	Lucas	GLICOSE	Roberto
999123	23/03/2014	Marcus	GLICOSE	Bruna

**Área de Queries**

**Tela de Exibição de Resultado**

Pedido: 999265  
 Data Exame: 15/04/2014  
 Nome Paciente: Valeria  
 Prontuario Paciente: 34720  
 Medico Solicitante: Douglas, CRM: 1

Cod. Exame: 14635-4 Exame: GLICOSE  
 GLICOSE: 103.0mg/dL

Valor de Referência:  
 Min: 70.0mg/dL  
 Max: 100.0mg/dL

Medico Assinante: Douglas  
 CRM: 1

Figura 28 – Visão geral da aplicação LabQuery.

**a) Seleção por Pedido:** A consulta realiza-se segundo o número do pedido de exame que o Médico Solicitante prescreve ao paciente. Os exames referentes ao pedido são exibidos na tabela interativa e o resultado de cada linha correspondente é exposto na Tela de Exibição (Figura 29).

**Selecionar Resultados:**

- ☒ Seleção por Pedido: 999123
- ☐ Seleção por Paciente:
  - Nome:
  - Prontuário:
- ☐ Seleção por Data:
  - Data Fixa:
  - Período:
    - A partir de:
    - Até:
- ☐ Seleção por Exame:
  - Cod. (LOINC):
  - Nome:
  - Valor Fixo:
  - Valor:

**Table of Results:**

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999123	23/03/2014	Marcus Vinicius Crelier	POTASSIUM	Dra. Bruna Araújo
999123	23/03/2014	Marcus Vinicius Crelier	GLICOSE	Dra. Bruna Araújo

**Detailed View:**

```

#####
Pedido: 999123
Data Exame: 23/03/2014
Nome Paciente: Marcus Vinicius Crelier
Prontuario Paciente: 12345
Medico Solicitante: Dra. Bruna Araújo, CRM: 0351651

-----
Cod. Exame: 12814-0   Exame: POTASSIUM
POTASSIUM: 3.4mmol/L

-----
Valor de Referência:
Min: 3.5mmol/L
Max: 5.0mmol/L

-----
Medico Assinante: Dr. Douglas Magno
CRM: 3855195
#####
  
```

Figura 29 – Pesquisa por Pedido.

**b) Seleção por Paciente:** A consulta realiza-se segundo o número do prontuário ou nome do paciente (Figura 30). Os exames do paciente são exibidos na tabela interativa e o resultado de cada linha correspondente é exposto na Tela de Exibição.

**Selecionar Resultados:**

- ☐ Seleção por Pedido:
- ☒ Seleção por Paciente:
  - Nome: Valeria Monteiro
  - Prontuário:
- ☐ Seleção por Data:
  - Data Fixa:
  - Período:
    - A partir de:
    - Até:
- ☐ Seleção por Exame:
  - Cod. (LOINC):
  - Nome:
  - Valor Fixo:
  - Valor:

**Table of Results:**

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999265	15/04/2014	Valeria Monteiro	GLICOSE	Dr. Filipe Taucel
999265	15/04/2014	Valeria Monteiro	POTASSIUM	Dr. Filipe Taucel

**Detailed View:**

```

#####
Pedido: 999265
Data Exame: 15/04/2014
Nome Paciente: Valeria Monteiro
Prontuario Paciente: 34720
Medico Solicitante: Dr. Filipe Taucel, CRM: 849651

-----
Cod. Exame: 12814-0   Exame: POTASSIUM
POTASSIUM: 5.3mmol/L

-----
Valor de Referência:
Min: 3.5mmol/L
Max: 5.0mmol/L

-----
Medico Assinante: Dr. Douglas Machado
CRM: 3855195
#####
  
```

Limpar Campos    Pesquisar

Figura 30 – Pesquisa por Paciente.

**c) Seleção por Data:** Os resultados são exibidos pelo critério de data fixa (Figura 31) ou período de datas (Figura 32) no formato DD/MM/AAAA. O período

pode ser “a partir”, “até” ou “apartir/até”. Os resultados são expostos da mesma forma que nos casos anteriores.

**Selecionar Resultados:**

- ☐ Seleção por Pedido:
- ☐ Seleção por Paciente:
- ☒ Seleção por Data:
  - Data Fixa: 15/04/2014
  - Período: A partir de: Até:
- ☐ Seleção por Exame:

Cod. (LOINC): Nome: Valor Fixo: Valor:

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999265	15/04/2014	Valeria Monteiro	GLUCOSE	Dr. Filipe Taucei
999265	15/04/2014	Valeria Monteiro	POTASSIUM	Dr. Filipe Taucei

#####  
 Pedido: 999265  
 Data Exame: 15/04/2014  
 Nome Paciente: Valeria Monteiro  
 Prontuario Paciente: 34720  
 Medico Solicitante: Dr. Filipe Taucei, CRM: 849651  
 -----  
 Cod. Exame: 12814-0 Exame: POTASSIUM  
 POTASSIUM: 5.3mmol/L  
 -----  
 Valor de Referência:  
 Min: 3.5mmol/L  
 Max: 5.0mmol/L  
 -----  
 Medico Assinante: Dr. Douglas Machado  
 CRM: 3855195  
 #####

Figura 31 – Pesquisa por Data Fixa.

**Selecionar Resultados:**

- ☐ Seleção por Pedido:
- ☐ Seleção por Paciente:
- ☒ Seleção por Data:
  - Data Fixa:
  - Período: A partir de: 22/03/2014 Até: 23/03/2014
- ☐ Seleção por Exame:

Cod. (LOINC): Nome: Valor Fixo: Valor:

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999119	22/03/2014	Lucas Moura	POTASSIUM	Dr. Roberto Costa
999119	22/03/2014	Lucas Moura	GLUCOSE	Dr. Roberto Costa
999123	23/03/2014	Marcus Vinicius Crelier	POTASSIUM	Dra. Bruna Araújo
999123	23/03/2014	Marcus Vinicius Crelier	GLUCOSE	Dra. Bruna Araújo

#####  
 Pedido: 999123  
 Data Exame: 23/03/2014  
 Nome Paciente: Marcus  
 Prontuario Paciente: 12345  
 Medico Solicitante: Dra. Bruna Araújo, CRM: 0351651  
 -----  
 Cod. Exame: 14635-4 Exame: GLUCOSE  
 GLUCOSE: 91.0mg/dL  
 -----  
 Valor de Referência:  
 Min: 70.0mg/dL  
 Max: 100.0mg/dL  
 -----  
 Medico Assinante: Dr. Douglas Magno  
 CRM: 3855195  
 #####

Figura 32 – Pesquisa por Período de Data.

**d) Seleção por Exame:** A pesquisa é realizada seguindo critérios de nome ou código LOINC dos exames (Vide seção 2.2.3). A grande vantagem do código LOINC é ser universal, tanto para o exame quanto para o tipo de amostra que foi utilizada. Ao entrar com o nome ou código do exame (Figura 33), surge a tabela interativa com os resultados encontrados.

The screenshot shows the LabQuery application interface. On the left, under 'Selecionar Resultados:', the 'Seleção por Exame:' option is selected. The search criteria are: Cod. (LOINC): [empty], Nome: GLICOSE, and Valor Fixo: [empty]. The main panel displays a table of results:

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999265	15/04/2014	Valeria	GLICOSE	Douglas
999119	22/03/2014	Lucas	GLICOSE	Roberto
999123	23/03/2014	Marcus	GLICOSE	Bruna

Below the table, a detailed view for 'Pedido: 999123' is shown, including patient information (Name: Marcus Vinicius Crelier, ID: 12345, Doctor: Dra. Bruna Araujo, CRM: 0351651), exam details (Cod. Exame: 14635-4, Exame: GLICOSE, Value: 91.0mg/dL), and reference values (Min: 70.0mg/dL, Max: 100.0mg/dL). The attending physician is Dr. Douglas Magno (CRM: 3855195).

Figura 33 – Pesquisa por Exame.

Os exames podem ser encontrados baseando-se nas faixas de valores (Figura 34). Isto é útil para encontrarmos valores fora da escala de referência. Convém ressaltar a utilização do ponto como caractere de ponto flutuante. Caso exista um resultado com valor acima ou igual a mil, o valor é exibido em formato contínuo, sem caracteres delimitadores à exceção apenas da parte decimal.

The screenshot shows the LabQuery application interface. On the left, under 'Selecionar Resultados:', the 'Seleção por Exame:' option is selected. The search criteria are: Cod. (LOINC): [empty], Nome: POTASSIUM, Valor Fixo: [empty], and a value range (Faixa de Valores) with 'Maior que: 2.6' and 'Menor que: 5.7'. The main panel displays a table of results:

PEDIDO	DATA	PACIENTE	EXAME	MEDICO SOL.
999119	22/03/2014	Lucas	POTASSIUM	Roberto
999123	23/03/2014	Marcus	POTASSIUM	Bruna
999265	15/04/2014	Valeria	POTASSIUM	Douglas

Below the table, a detailed view for 'Pedido: 999265' is shown, including patient information (Name: Valeria, ID: 34720, Doctor: Douglas, CRM: 1), exam details (Cod. Exame: 12814-0, Exame: POTASSIUM, Value: 5.3mmol/L), and reference values (Min: 3.5mmol/L, Max: 5.0mmol/L). The attending physician is Douglas (CRM: 1).

Figura 34 – Pesquisa por Exame com busca por faixa de valores.

#### **4.7 Considerações Finais**

Devido a quantidade de dados que um CDA pode armazenar, esta implementação não traduz todo o universo de consultas e cruzamentos de dados possíveis. No entanto, pode-se afirmar que, com a ausência de esquema no MongoDB aliado a estrutura sintática e semântica dos dados contidos nos documentos HL7 CDA, o desenvolvedor terá liberdade para criar implementações mais facilmente e de mais alto nível de abstração. Poder-se-ia por exemplo consultar se os pacientes de uma determinada cidade (dado incluso no CDA) possuem características de glicose alta. Este exemplo traduz os benefícios da desnormalização dos dados e da ausência de esquema nos registros contidos no MongoDB.

Finalmente, como é da natureza dos registros de resultados de exames, este banco de dados não tem como objetivo principal as alterações nos registros, pois afinal é um repositório de documentos emitidos por equipamentos. Desta forma, não faz sentido sacrificar a flexibilidade de consulta em prol da inserção/atualização de registros, característica comum no SQL e bancos relacionais.

## 5. Conclusão

### 5.1 Considerações gerais

A grande demanda por serviços cada vez mais ágeis e precisos de diagnósticos e análises clínicas motiva a criação de padrões de interoperabilidade mais abrangentes e que possam possibilitar a crescente expansão da base de conhecimentos do campo das Análises Clínicas. Repositórios de dados facilmente escaláveis e flexíveis permitem que se possa acompanhar os constantes avanços da área de diagnóstico laboratorial.

Ao lado da questão dos repositórios, padrões e nomenclaturas garantem o caráter universal dos dados, de modo que todas as nações possam interpretar da mesma forma uma determinada informação. Neste contexto, o uso do HL7 torna-se imprescindível. Quase todos os equipamentos utilizados atualmente possuem suporte ao protocolo HL7, seja na Versão 2.x ou a Versão 3. A versão 2 continua ainda muito utilizada devido a consolidada base de empresas e profissionais de suporte existentes. Em se tratando da Versão 3 do HL7, convém acrescentar que diferentes tipos de implementação deste padrão vêm buscando seu espaço no mercado. Uma das mais promissoras é o FHIR, cujo padrão amigável de mensagens menores e mais simples, orientadas a recursos bem como extremamente voltada a distribuição de repositório de dados, tem sido alvo do interesse da comunidade de desenvolvedores. Ela tem como objetivo buscar uma solução mais enxuta para a exibição de mensagens na versão 3 do HL7, visando desta forma abandonar definitivamente o formato V2.x.

No Brasil destacam-se os esforços para uma normatização das informações em saúde pública, materializada pela Portaria Nº 2.073 de 31/08/2011 e as resoluções formadas nos conselhos de Saúde e em encontros como o CBIS (*Congresso Brasileiro de Informática em Saúde*) promovido pela SBIS (*Sociedade Brasileira de Informática em Saúde*). No entanto, as preocupações residem no sentido da confiabilidade e sigilo das informações dos pacientes para que somente usuários habilitados possam acessá-las.

O fato de se usar uma padronização documental como o HL7 CDA e códigos de nomenclatura de exames como o LOINC, permitem que os resultados possam ser entendidos em alcance global, pois a maior parte dos equipamentos de exames modernos já têm estes protocolos habilitados. Desta forma, diferentes instituições médicas em pontos bem distintos do mundo, ao acessar este modelo de dados, poderão compreender sem falhas, o histórico do paciente, contribuindo para uma rápida e precisa tomada de decisão sobre possíveis procedimentos a serem tomados. Por outro lado, o uso do HL7 resolve satisfatoriamente a questão da interoperabilidade entre sistemas, tornando não só a implementação de novos equipamentos quanto o interfaceamento de resultados menos trabalhosos de se executar.

Nesta pequena implementação procura-se demonstrar um pouco das potencialidades de se usar um banco de dados não-relacional com um dos principais padrões de interoperabilidade médica. Neste caso, utiliza-se um banco orientado a documentos (MongoDB) como repositório de resultados de exames no formato do HL7 CDA, padronizado para ser compreendido universalmente.

Com relação ao MongoDB, é possível que existam alternativas mais flexíveis, ainda que com implementação custosa, mas o equilíbrio de uma abordagem que garanta consistência de dados, um custo de implementação baixo, o reduzido tempo de consulta

e a flexibilidade de organização dos dados traduzem como uma solução aparentemente adequada para as rápidas mudanças que ocorrem neste segmento da área de saúde. Tem-se um cenário de alta probabilidade de modificações na metodologia de análise clínica e de geração de novas fontes de análise, necessitando assim de um banco de dados com tabelas flexíveis, que possam rapidamente comportar novos campos sem perda de desempenho em um contexto em que o número de consultas supera em muito a quantidade de inserções e atualizações de registros.

## **5.2 Trabalhos Futuros**

Propõe-se os seguintes trabalhos futuros neste tema:

- a) Implementação de novas funcionalidades, aproveitando os dados contidos no documento HL7 CDA não considerados na aplicação LabQuery;
- b) Construção de módulos para análises clínicas utilizando bancos orientados por colunas como o Hadoop em conjunto com o HL7 V3 na implementação FHIR;
- c) Integração de novos tipos de documentos clínicos mais enxutos como o GreenCDA e geração de grafos, com bancos orientados a documentos. Sugere-se o uso do nodeJS como linguagem para construção de sistemas que utilizam o MongoDB;
- d) Sistemas laboratoriais voltados a pesquisa clínica com o uso de FHIR e banco de dados orientados a grafos como o HiperGraph.



## 6. Referências Bibliográficas

ABDILLAH Fajri, Python Redis [How to]: Cache Python MySQL Result using Redis– Disponível em: <<http://clasense4.wordpress.com/2012/07/29/python-redis-how-to-cache-python-mysql-result-using-redis/>> Acessado em 02/11/2014.

BENSON Tim, *Principles of Health Interoperability HL7 and SNOMED* ED Springer, 3ed. – 2010.

BREWER Eric A, *Towards Robust Distributed Systems* – Disponível em <<https://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf>> Acessado em 30/10/2014.

CHODOROW Kristina, *50 Tips & Tricks for MongoDB Developers*, ED O'Reilly Media, 1ed. – 2011.

CHODOROW Kristina, DIROLF Michael, *MongoDB: The Definitive Guide*, ED O'Reilly Media, 1ed. – 2010.

FHIR – Disponível em: <<http://www.hl7.org/implement/standards/fhir/>>. Acessado em 20/07/2014.

HL7 Protocol – Disponível em: <<http://www.interfaceware.com/manual/ch-2.html>>. Acessado em 20/03/2014.

HL7 Standards – Disponível em: < <http://www.hl7standards.com>>. Acessado em 09/04/2014

HL7 V3 Guide – Disponível em: < <http://www.hl7cn.com/help/v3guide/>>. Acessado em 01/08/2014

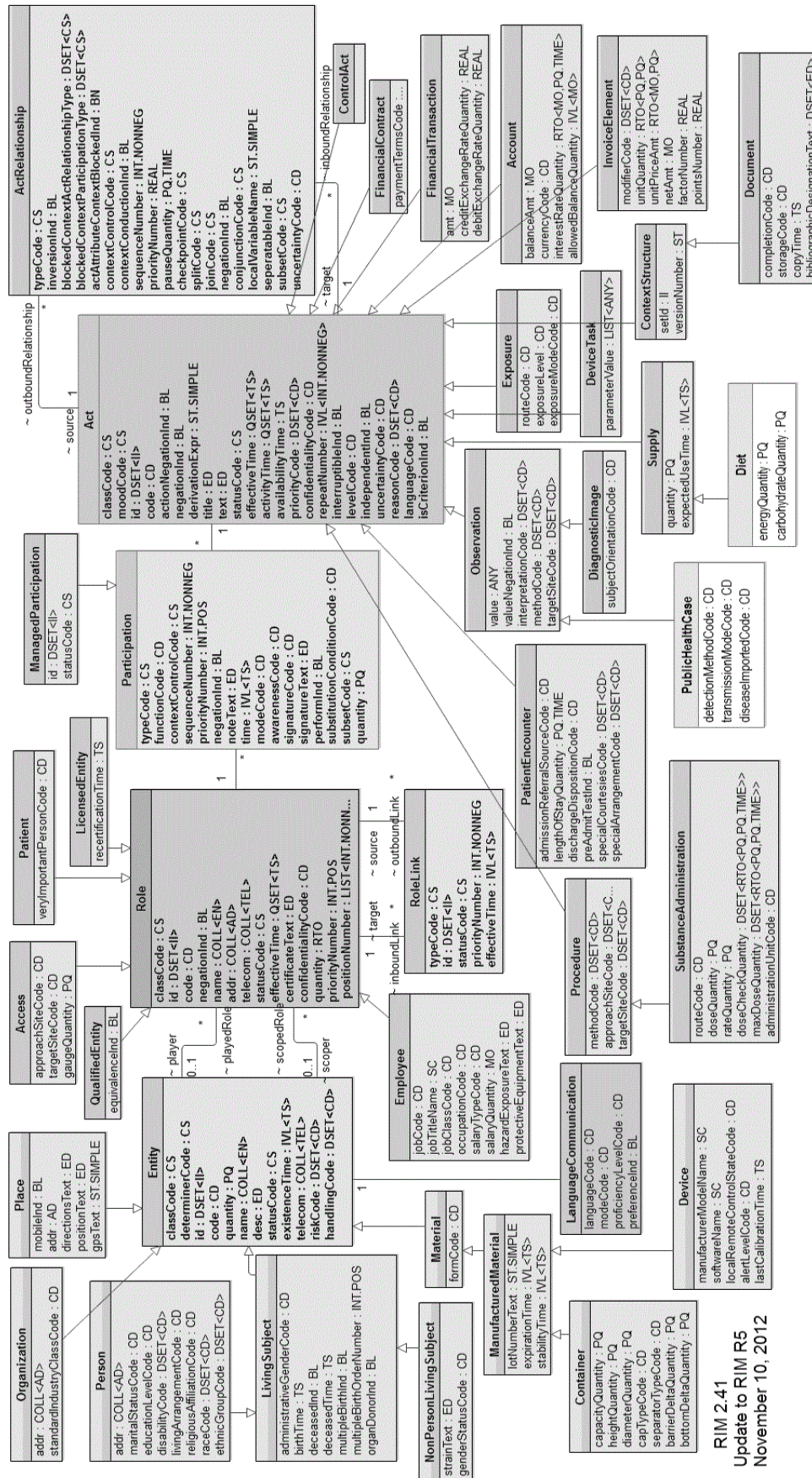
KIMBALL Ralph, *A Dimensional Modeling Manifesto* – Disponível em: <<http://www.kimballgroup.com/1997/08/a-dimensional-modeling-manifesto/>>. Acessado em 30/10/2014.

PLUGGE Eelco, MEMBREY Peter, HAWKINS Tim, *The Definitive Guide to MongoDB – The NoSQL Database for Cloud and Desktop Computing*, ED Apress, 1ed. – 2010.

SADALAGE Pramod J, FOWLER Martin, *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*, ED Addison-Wesley Professional, 1ed. – 2012.

Site Oficial da Organização Health Level Seven – Disponível em: <[www.hl7.org](http://www.hl7.org)>. Acessado em 13/02/2014.

STROZZI Carlo, *NoSQL Relational Database Management System* – Disponível em: <[http://www.strozzi.it/cgi-bin/CSA/tw7/I/en\\_US/NoSQL/Home%20Page](http://www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/NoSQL/Home%20Page)>. Acessado em 30/10/2014.



RIM 2.41  
 Update to RIM R5  
 November 10, 2012



[illegible]

ANEXO III: RMIM para “Specimen” (Amostra a ser analisada por algum Laboratório)  
(Extraído de [http://wiki.hl7.org/index.php?title=RMIM\\_Diagram\\_Representation](http://wiki.hl7.org/index.php?title=RMIM_Diagram_Representation)).

