INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Sistema de informação e extração de conhecimento para análise e gestão de informação médica

Marco Paulo Machado Custódio

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática e de Computadores

Júri

Presidente:	Doutor João António Madeiras Pereira
Orientador:	Doutor Alexandre Paulo Lourenço Francisco
Coorientador:	Doutora Sara Alexandra Cordeiro Madeira
Vogais:	Doutora Ana Teresa Correia de Freitas
	Doutora Helena Sofia Andrade Nunes Pereira Pinto

Outubro de 2012

Resumo

A doença de Alzheimer causa declínios cognitivos nos doentes e tem sobre eles um efeito devastador, provocando um impacto extremamente negativo a nível socioeconómico nas sociedades modernas. Apesar da grande quantidade de informação disponível, a heterogeneidade dos formatos e modelos de dados e do seu significado dificulta a sua interpretação e integração. Os Sistemas de Informação que utilizam tecnologia Web Semântica permitem que a integração de dados e conhecimento distribuído por várias fontes heterogéneas e a sua utilização seja substancialmente mais fácil e eficiente, potenciando a descoberta e a partilha de novos conhecimentos.

Neste trabalho foi implementado um sistema de informação e extração de conhecimento para análise e gestão de informação médica apoiado por uma ontologia desenvolvida para o efeito denominada *Neuropsychological Test Ontology*. Esta descreve e relaciona conceitos relativos à aplicação de testes neuropsicológicos a pacientes potencialmente com doença de Alzheimer e serviu de base para a anotação semântica de dados clínicos reais, originalmente disponíveis como ficheiros Excel.

A aplicação Web efetua o carregamento dos dados anotados num repositório *triple store* permitindo a sua consulta e visualização, tendo em conta critérios de privacidade de dados. Para além disto interage com um *software* de prospeção de dados através de um Web Service. Este disponibiliza um serviço capaz de predizer diagnósticos e prever prognósticos num intervalo temporal pré-definido baseados em modelos de classificação, e um serviço de importação de dados. Através do serviço de importação de dados, é possível ajustar os modelos de acordo com os novos dados fornecidos pela aplicação Web.

Palavras-chave

Doença de Alzheimer, Web Semântica, Ontologias, Testes Neuropsicológicos, Integração de Dados Clínicos

Abstract

Alzheimer's disease causes cognitive decline in patients and has a devastating effect on them, causing an extremely negative socioeconomic impact in modern societies. Despite the huge amount of information available, the heterogeneity of formats and data models makes their integration and interpretation very difficult. Information Systems using Semantic Web technology allows the integration of data and knowledge spread across multiple heterogeneous sources and their use is substantially easier and more efficient, enhancing the discovery and sharing of new knowledge.

In this work, it was implemented a system of information and knowledge extraction for analysis and management of medical information supported by an ontology developed for this purpose and named Neuropsychological Test Ontology. This ontology describes concepts related to the application of neuropsychological tests to patients with potentially Alzheimer's disease and was the basis for the semantic annotation of real clinical data, originally available in excel files.

The developed web application performs the loading of semantically annotated data in a triple store repository allowing its search and visualization, taking into account private data criteria. In addition, interacts with data mining software through a Web Service that makes available a service able to predict diagnoses and prognoses in a predefined time interval based on classification models, and a data import service. Through the service import data, you can adjust the models according to new data provided by the web application, reflected in the behavior of these models.

Keywords

Alzheimer’s Disease, Semantic Web, Ontologies, Neuropsichological Tests, Clinical Data Integration

Índice

Lista de Figuras

Acrónimos

AD	Doença de Alzheimer (Alzheimer's disease)
ALS	Esclerose Lateral Amiotrófica (Amyotrophic Lateral Sclerosis)
API	Application Programming Interface
FOAF	Friend of a Friend
HTTP	Hypertext Transfer Protocol
IRI	Internationalized Resource Identifier
OBO	Open Biomedical Ontology
ODP	Ontology Design Patterns
OWL	Ontology Web Language
OWL-DL	Ontology Web Language Description Logic
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
REST	Representational State Transfer
RIF	Rule Interchange Format
SI	Sistemas de Informação
SKOS	Simple Knowledge Organization System
SPARQL	Simple Protocol and RDF Query Language
SQL	Structures Query Language
SWRL	Semantic Web Rule Language
TI	Tecnologias de Informação
UML	Unified Modeling Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WS	Web Semântica
WWW	World Wide Web
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSD	XML Schema Definition

1. Introdução

Esta dissertação de mestrado visa o desenho, desenvolvimento e implementação de um sistema de informação e extração de conhecimento para análise e gestão de informação médica, nomeadamente informação relativa à realização de testes médicos neuropsicológicos aplicados a pacientes, potencialmente com a doença de Alzheimer.

As doenças neurodegenerativas constituem desafios únicos para a medicina quer do ponto de vista científico quer devido ao seu efeito devastador nos doentes e família, e pelo impacto socioeconómico nas sociedades modernas. Declínios nas funções cognitivas e motoras, aliadas a outras evidências de degeneração neurológica, surgem naturalmente com o envelhecimento. Infelizmente todos teremos funções cerebrais alteradas, uns mais cedo ou a um ritmo mais acelerado que outros. Neste sentido, é prioritário na investigação atual distinguir os declínios motores e cognitivos do envelhecimento dos que ocorrem devido a processos patogénicos, e perceber os padrões individuais de diagnóstico e prognóstico (Noorbakhsh F., 2009).

Os avanços tecnológicos colocaram disponível um grande volume de dados levando a uma mudança fundamental nos paradigmas de investigação na biologia e na medicina. Em vez de efetuar testes restritivos focados no comportamento de um único gene como no passado, é agora possível analisar várias entidades biológicas em simultâneo. A análise destes resultados, aqui chamados ómicos (provenientes de genómica, transcriptómica, proteómica e metabolómica), constituem um enorme avanço na compreensão do funcionamento complexo dos processos biológicos e na modelação do organismo humano no seu todo. A biologia de sistemas tem tido sucesso no estudo destes dados com o objetivo de obter conhecimento fundamental acerca dos processos biológicos e das redes biológicas. No entanto, o seu foco à escala molecular ignora os efeitos da fisiologia e a variabilidade entre indivíduos, fruto da interação entre a biologia e o comportamento dos indivíduos ao longo da vida (Clermont, et al., 2009) (Gordon, 2007). A medicina de sistemas surgiu recentemente como aplicação da biologia de sistemas à prevenção, modelação e recuperação de condições patológicas na saúde humana (Clermont, et al., 2009), através da integração de informação médica nos modelos, nomeadamente condições ambientais, variáveis clínicas e dados de imagiologia.

A grande heterogeneidade dos formatos dos dados, dos modelos de dados e do seu significado dificulta a sua integração. Contudo, é sabido que utilização de abordagens integrativas de dados genómicos e clínicos permite a obtenção de um conhecimento mais

alargado sobre patologias cerebrais ao inferir relações entre dados genómicos, clínicos, e pessoais (Clermont, et al., 2009) (Gordon, 2007) (Noorbakhsh, et al., 2009) (Schadt, 2009). Uma perspetiva interessante sobre a manipulação de dados heterogéneos reside no facto de efetuando a sua definição formal e clarificação do seu significado, a utilização desses dados será substancialmente mais fácil e eficiente.

Com o crescente volume, complexidade e heterogeneidade dos dados, os cientistas necessitam de novas capacidades que assentem nas novas abordagens semânticas. O uso de Sistemas de Informação (SI) que utilizem a tecnologia Web Semântica é atualmente uma das soluções propostas, e mais promissoras, para integração de dados e conhecimento distribuído por várias fontes heterogéneas (Ruttenberg A., 2007). Estas abordagens facilitam a modelação do conhecimento científico, a verificação de hipóteses, o desenvolvimento de aplicações que analisam dados heterogéneos de diversos domínios e fontes, e a partilha de novos conhecimentos (Hey T., 2009).

Apesar dos esforços de integração, não foram ainda propostas abordagens integrativas que possam efetivamente ser usadas para estudar em profundidade as doenças neurodegenerativas a partir de dados genómicos e clínicos (Noorbakhsh F., 2009). Surge assim a necessidade da criação de um Sistema de Informação capaz de adquirir, organizar, analisar, correlacionar, interpretar, inferir e raciocinar sobre dados heterogéneos de doenças neurodegenerativas.

1.1. Projeto NEUROCLINOMICS

Esta dissertação de mestrado está integrada no projeto "NEUROCLINOMICS - Compreendendo as doenças neurodegenerativas através da integração de dados clínicos e genómicos". Este projeto propõe uma abordagem inovadora para a compreensão de doenças neurodegenerativas através da integração de dados heterogéneos. A utilização de um sistema sofisticado de descoberta de conhecimento integrando algoritmos avançados para prospeção de dados permitirá desvendar as potenciais ligações relevantes entre dados genómicos e clínicos. O objetivo é a identificação de marcadores de diagnóstico e prognóstico, de taxas de progressão da doença, e de perfis de doentes. Para além do desafio do estudo de doenças complexas, o desenvolvimento de algoritmos eficientes que possam efetivamente ser usados para integração de dados biomédicos é também um objetivo. Pretende-se que esta

investigação produza contribuições efetivas para o avanço do conhecimento em métodos de integração de dados heterogéneos, em particular na área da biomedicina.

No contexto deste projeto, serão consideradas duas doenças neurodegenerativas como casos de estudo: Esclerose Lateral Amiotrófica (ALS), causando problemas motores, e a doença de Alzheimer (AD), causando problemas cognitivos. Alguns dos problemas da maior relevância em ALS e AD são, respetivamente prever a falha respiratória (causa de morte mais frequente), e prever se um doente com problemas de cognitivos irá sofrer de AD no futuro.

Esta dissertação enquadra-se numa das tarefas do projeto. Esta designa-se por "T1 – Sistema de Informação e Descoberta de Conhecimento" e tem como objetivo o desenho e a implementação de um sistema de informação e de descoberta de conhecimento para estudar doenças neurodegenerativas, usando integração de dados heterogéneos e algoritmos de prospeção de dados para dados clínicos e ómicos.

1.2. Definição do problema

O protótipo funcional do sistema de informação e de extração de conhecimento tem os seguintes requisitos de alto nível:

- Ser um sistema de armazenamento de dados escalável. Deverá ser possível integrar informação relevante de várias fontes de dados remotas, nomeadamente dados clínicos e ómicos de ALS e AD. Os dados de ALS incluem dados demográficos, características da doença, evolução clínica, testes respiratórios e neurofisiologia. Os dados AD incluem história clínica, exames neurológicos, testes neurofisiológicos, avaliação laboratorial e imagens cerebrais. As fontes remotas são BRAINnet[1], ADNI[2] e dbSNP[3].
- Disponibilizar os dados através de uma interface Web. A Framework deverá ser extensível permitindo um fácil desenvolvimento de novas funcionalidades. Deverá ser possível inserir e usar dados de novos pacientes, e correr novamente os algoritmos de prospeção sobre novos dados. Os dados integrados de fontes externas deverão poder ser atualizados e sincronizados quando necessário. Deverá ser garantida a persistência dos

[1] http://www.brainnet.net/

[2] http://adni.loni.ucla.edu/

[3] http://www.ncbi.nlm.nih.gov/projects/SNP

resultados da análise dos dados. Deverá ser possível guardar e disponibilizar resultados de análise de dados para outros parceiros.

- Possibilidade de integração com *software* de prospeção de dados.
- Fornecer uma interface baseada em perfis de utilizadores, personalizada dependendo da doença em análise e dos interesses do utilizador. Garantir a integridade e privacidade dos dados. É necessária a implementação de políticas de acesso de utilizadores. Apesar de todos os utilizadores poderem usar os algoritmos para analisar dados próprios e provenientes de fontes públicas, deverá ser possível restringir informação e conhecimento apenas a utilizadores autorizados.

1.3. Metodologia

A metodologia seguida para a realização desta dissertação comportou as seguintes fases:

1. Enquadramento da dissertação no projeto NEUROCLINOMICS e identificação dos seus principais objetivos.
2. Revisão de literatura da Web Semântica, nomeadamente a compreensão da sua arquitetura, as tecnologias envolvidas e o seu contexto em aplicações nas áreas de biologia e medicina. Levantamento das ontologias já existentes e relevantes para o contexto da dissertação.
3. Análise e experimentação de ferramentas de suporte ao desenvolvimento de aplicações de Web Semântica, preferencialmente *software open source*, como por exemplo Virtuoso, Jena, Pellet, Protégé e Ontology Browser.
4. Compreensão e delimitação de conceitos e relações inerentes ao domínio das doenças neurodegenerativas e em especial de testes neuropsicológicos.
5. Criação de uma ontologia relativa a testes médicos neuropsicológicos.
6. Análise, desenho e implementação do protótipo funcional do sistema de informação e extração de conhecimento, reutilizando tecnologia *open source*.
7. Importação de dados clínicos relevantes das fontes internas ao projeto, nomeadamente testes neuropsicológicos aplicados pela equipa de médicos do Instituto de Medicina Molecular, parceiros no projeto Neuroclinomics.
8. Integração do sistema com *software* de prospeção de dados.
9. Implementação, no sistema de informação, de um controlo de acessos baseado em perfis, que possibilite a privacidade e integridade dos dados.

1.4. Introdução ao conteúdo dos capítulos

Este relatório encontra-se organizado de acordo com os capítulos que seguidamente se descreve. Neste capítulo 1 é explanado o assunto a tratar, orientando o leitor para a temática das doenças neurodegenerativas. É ainda definido o problema e a metodologia seguida para a sua resolução. No capítulo 2, o estado da arte, é efetuado a apresentação dos principais aspetos funcionais e tecnológicos no âmbito da tecnologia da Web Semântica bem como de alguns projetos relacionados. O capítulo 3 explica o desenvolvimento da ontologia de testes médicos neuropsicológicos, e clarifica os conceitos e relações entre as entidades ontológicas. No capítulo 4 é descrito a tecnologia e os aspetos gerais do sistema de informação e de extração de conhecimento para análise e gestão de informação médica. O capítulo 5 descreve a interface gráfica e a implementação das funcionalidades disponíveis no sistema de forma enquadrada nos diferentes perfis que os utilizadores podem assumir. No capítulo 6 são apresentadas as conclusões e as contribuições desta dissertação, bem como propostas para um trabalho futuro.

2. Estado da Arte

2.1. A Web

Fundamentalmente a partir do novo milénio a Internet entrou nas nossas vidas e alterou a forma como interagimos neste novo mundo. A forma de comunicar, de fazer negócios, de aprendizagem, de lazer alterou-se radicalmente com a rede das redes. O incremento da largura de banda associada à generalização do uso de computadores e dispositivos móveis com acesso à Internet em todo o Mundo fomentou este rápido crescimento da Web. Esta tem servido especialmente como um repositório de documentos, imagens, música, filmes, jogos e de pequenos textos não contextualizados usando linguagem natural.

Com o crescimento do número de utilizadores por todo o Mundo e com o sucesso das redes sociais e dos *sites* de partilha de documentos, a informação disponível tornou-se imensa dificultando bastante a sua procura. Por outro lado, a informação toma um papel decisivo na nossa sociedade global, tornando-se no foco principal das transações económicas, e não apenas num suporte para elas (Boisot, et al., 2004). A maioria da informação existente *online* está disponibilizada tendo em vista os utilizadores humanos. A forma de apresentação e a procura deste tipo de conteúdos, essencialmente sob a forma de hipertexto complementada com imagens e outros tipos de multimédia dominou os anos iniciais da Web.

Genericamente, a pesquisa e obtenção de informação na Web tem sido efetuada recorrendo principalmente a serviços especializados designados por motores de busca. Estes serviços efetuam a análise, indexação, catalogação e hierarquização de páginas relevantes para um dado conjunto de termos específicos fornecidos (palavras-chave). Como resultado da pesquisa é disponibilizado ao utilizador um conjunto de recursos (páginas, imagens, documentos, etc.) cujo conteúdo terá de ser analisado por este de modo a tentar encontrar a informação pretendida. Este é um processo moroso e pouco eficiente.

Desta forma, é de extrema importância que a informação existente na Web tenha significado, isto é que seja anotada semanticamente, e que tenha informação de contexto, de modo a que possa ser processada eficaz e eficientemente de forma integrada por diferentes sistemas. É precisamente estas duas lacunas que a Web Semântica visa colmatar, revolucionando o modo como descobrimos, acedemos, integramos e usamos a informação (Bizer, et al., 2011).

2.2. eCiência

Em meados dos anos 90, Jim Gray reconheceu que os grandes desafios para a tecnologia da nova era, caracterizada pelas imensas quantidades de dados disponíveis e pelo processamento intensivo de dados, viriam não do comércio mas da ciência. Para este autor, a ciência está a iniciar o seu 4º paradigma (Hey, et al., 2009). Historicamente a ciência sofreu os seguintes paradigmas:

- **Ciência Empírica** - na qual os cientistas recolhiam os dados das observações directas e analisavam esta informação, obtendo conclusões e construindo regras da natureza;
- **Ciência Teórica** - na qual os cientistas construíam modelos analíticos (fórmulas) coerentes com as observações e, através das quais podiam fazer predições;
- **Ciência Computacional** - na qual os cientistas utilizam intensivamente o computador como instrumento de trabalho, possibilitando desta forma simular modelos analíticos, validá-los e construir predições (ex.: extinção de uma dada espécie biológica ou extensão do buraco do ozono);
- **eCiência** - na qual a ciência é fortemente apoiada por sistemas computacionais que recebem eventos adquiridos por instrumentação, geram dados de simulações, armazenam dados em bases de dados, muitas delas distribuídas. Através da recolha, organização, sumarização, análise e visualização desta monumental massa de dados é possivel inferir novos dados, encontrar novas correlações e modelos analíticos capazes de efectuar novas predições.

O termo eCiência foi inicialmente utilizado por John Taylor em 2000 para designar o conjunto de ferramentas e tecnologias necessárias ao suporte desta nova ciência. Fê-lo ao reconhecer a importância vital e crescente que as Tecnologias de Informação (TI) desempenhavam na pesquisa científica na sua forma colaborativa, multidisciplinar e de processamento intensivo de dados. O conhecimento científico é todo ele interligado. Os investigadores compreenderam que os dados deveriam estar unidos numa única massa informacional formando uma base de dados global acessível a investigadores de diversos domínios do conhecimento. O sucesso da investigação reside também na possibilidade de partilha de informação entre os vários domínios, através da correlação de dados, factos, assunções e metodologias e no relacionamento desses conceitos. Contudo, esses dados tem um carácter heterogéneo dificultando a sua integração. A integração de dados, informação,

conhecimento e experiência é a chave do sucesso da investigação biomédica possibilitando no futuro uma melhoria da prática clínica.

Desta forma, as comunidades científicas necessitam de acordarem um vocabulário comum. Este processo de partilha de conhecimento científico gerado por diferentes investigadores, em diferentes tempos e lugares, e a atribuição do mesmo vocabulário e significado à informação, resultou no aparecimento de novas abordagens nos Sistemas de Informação disponíveis e nas tecnologias que os suportam. A influência crescente que a comunidade de inteligência artificial teve nas áreas de investigação conduziu a comunidade científica a procurar novas metodologias, para a resolução dos seus problemas. A generalidade das atuais capacidades requeridas pela eCiência necessitam de representação e mediação semântica. Esta necessidade floresce em parte com a crescente interdisciplinaridade da investigação moderna. A utilização de ontologias para anotação de dados permite não só adicionar definições formais de vocabulários, conceitos e termos explicando o seu inter-relacionamento, bem como relacioná-los com outros que residem em diferentes repositórios.

O trabalho desenvolvido pelas novas comunidades de Web Semântica resultou num número de padrões, linguagens e tecnologias que permitem auxiliar na modelação de dados, na representação da informação e na partilha de semântica dentro do contexto de um domínio particular de aplicação. O acordo para a existência de uma base comum e formal permite a introdução de capacidades de inferência, possibilitando a descoberta de novo conhecimento.

2.3. Web Semântica

Segundo Tim Berners-Lee (Berners-Lee, et al., 2001), a Web Semântica (WS) é uma extensão da Web atual na qual é atribuído à informação um significado bem definido, permitindo uma melhor cooperação entre sistemas computacionais e pessoas. O desenvolvimento de uma Web que permita o processamento da informação por humanos mas também por máquinas permitirá a resolução de problemas que até agora seriam complexos e muito demorados. Desta forma deverá existir uma preocupação crescente em encontrar, aceder e processar a informação disponibilizada na Web. Para esse fim, a Web Semântica contribui com a atribuição de metadados estruturados aos documentos e dados existentes de modo a atribuir-lhes um significado semântico bem definido.

Esta nova estratégia de representar a informação, apresentando-a num formato estruturado, usando padrões universalmente aceites e conhecidos, permite aos sistemas computacionais processar a informação a um nível semântico, e não apenas ao nível sintático. Com a atribuição de significado à informação, potencialmente todos os sistemas que o pretendam poderão processar essa informação, interligá-la com outra informação relevante, proveniente de fontes heterogéneas, raciocinar sobre a mesma, permitindo ainda a sua futura reutilização.

Neste contexto da WS, os sistemas computacionais terão portanto uma maior capacidade para compreender a informação da Web possibilitando uma melhor procura, filtragem e categorização, conduzindo a sistemas capazes de inferir novo conhecimento e informação, estendendo desta forma o poder representacional da mesma. O primeiro passo para conseguir este desígnio é, segundo Tim Berners-Lee, colocar na Web os dados de modo a que os sistemas possam naturalmente perceber, ou converter os dados nessa forma, possibilitando aos sistemas, de forma análoga aos humanos, ler, perceber e raciocinar sobre a informação na Web.

2.4. Arquitetura da Web Semântica

O *World Wide Web Consortium* (W3C) criado em 1994 é a principal organização internacional de padronização para a *World Wide Web* (WWW) e tem como lema conduzir a Web ao seu máximo potencial (Signore, 2006). Os seus principais objetivos são construir uma Web para todos, Web em tudo, Web como base de conhecimento, e que esta seja de confiança e tenha confidencialidade. Um dos seus propósitos é o desenvolvimento de protocolos comuns para promover a evolução da WWW e garantir a sua interoperabilidade.

É neste contexto que surgiu o conceito da arquitetura da Web Semântica, também designada por Camadas da Web Semântica ilustrada na figura 1. A tecnologia da WS consiste assim na utilização de padrões e tecnologias de uma forma hierárquica. Esta arquitetura por camadas define as linguagens e tecnologias usadas para um perfeito futuro funcionamento da Web. A abordagem por camadas assume que os níveis superiores utilizam as capacidades dos níveis inferiores e permite ainda que os níveis inferiores possam ser aplicados de modo estável mesmo que os níveis superiores ainda estejam a ser formulados.

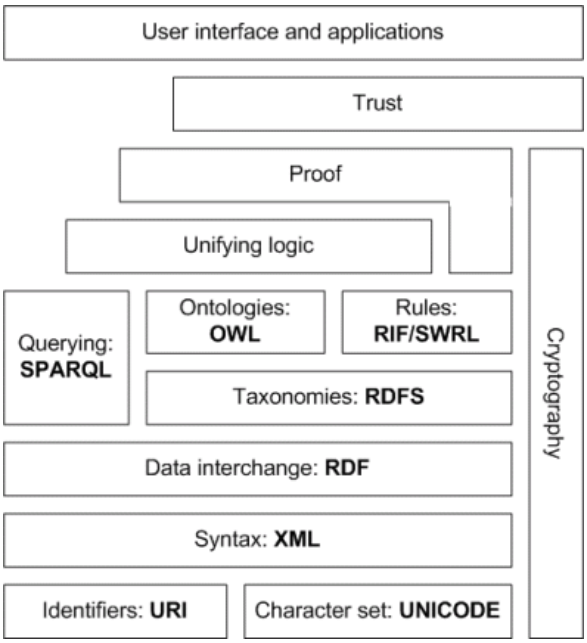

Fig. 1 – Arquitetura da Web Semântica[4], adaptado de W3C, segundo Berners-Lee [5]

A camada de base estabelece os conceitos fundamentais para toda a arquitetura através do conjunto dos possíveis caracteres e identificadores. O Unicode é um padrão que permite aos computadores representar e manipular, de forma consistente, texto de qualquer sistema de escrita. O URI (Uniform Resource Identifier) é um padrão para a identificação de recursos. A Web tem de ser vista como um espaço de informação universal, navegável através do mapeamento entre os URI e os recursos identificados. Cada recurso terá de ser identificado obrigatoriamente por um ou mais URI. O URI tem como função identificar e simultaneamente localizar o recurso.

Na camada seguinte surge a base sintática usada nas camadas superiores. O XML permite a criação de documentos compostos por informação estruturada. O XML tem uma importância primordial ao permitir definir estrutura própria aos documentos. Cada documento XML define o seu vocabulário próprio, mas tem a possibilidade de reutilizar vocabulários definidos noutros documentos referenciando outros espaços de nomes através de URI's. Assim, é possível evitar conflitos de vocabulário onde cada domínio pode identificar nomes que necessitam ser únicos no domínio local.

[4] http://www.obitko.com/tutorials/ontologies-semantic-web/semantic-web-architecture.html
[5] http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#(24)

A próxima camada estabelece como é feita a partilha de dados. O RDF é uma linguagem para representar formalmente metadados sobre os recursos na Web. O RDF fornece um modelo de dados baseado em grafos com o qual é possível estruturar e interligar informação. Um documento RDF é constituído por declarações RDF. Cada declaração é composta por um sujeito, um predicado e um objeto. O sujeito, que se quer descrever de modo semântico, é obrigatoriamente um recurso. A propriedade atribuída ao sujeito é designada por predicado e é também um recurso. O valor da propriedade que o sujeito tem é designado por objeto. O objeto pode ser um recurso mas poderá também ser um valor.

A camada *Taxonomies* permite a definição de um domínio do conhecimento ao definir classes, propriedades e hierarquias na informação. O conceito da hierarquia das classes, uma das principais características do RDFS, é semelhante ao das linguagens orientadas a objetos. No RDFS este conceito foi também estendido às propriedades. O RDF conjuntamente com o RDFS permite a definição de ontologias pois garante a reutilização do vocabulário entre documentos RDF e possibilita a geração de declarações específicas para um determinado domínio de conhecimento.

A camada *Ontologies* efetua a definição de ontologias através de linguagens como o OWL (Web Ontology Language). A ontologia permite de um modo formal definir a relação entre conceitos pela extensão do RDFS. Isto é conseguido pela adição de mais restrições como a cardinalidade, efetuando restrição de valores possíveis ou definindo características das propriedades como a transitividade. A OWL é baseada em lógica descritiva conferindo o poder de raciocínio à Web Semântica.

A camada *Rules* é constituída por um conjunto de regras lógicas que possibilitam a inferência de novo conhecimento e a tomada de decisão. Neste nível foram propostos os padrões RIF (Rule Interchange Format) ou SWRL (Semantic Web Rule Language). Estas duas linguagens de representação de regras garantem o suporte à descrição dessas mesmas regras, permitindo a descrição de relações que não conseguem ser descritas pela lógica de descrição usada no OWL.

A camada *Quering* é constituída pelo padrão SPARQL (Simple Protocol and RDF Query Language). É uma linguagem que permite a consulta a qualquer informação baseada em RDF guardada em repositórios designados por *Triple Store*. As aplicações podem desta forma obter a informação desejada de uma forma padronizada, de modo análogo ao uso do SQL nas bases de dados relacionais.

A camada *Unifying logic*, bem como as restantes camadas superiores desta arquitetura contêm tecnologias que ainda não têm padrões estáveis ou que apenas contêm fundamentos que idealmente deverão ser implementadas. A Lógica Unificada permitirá às aplicações utilizarem os modelos semânticos das camadas inferiores, com uma lógica formal uniforme e consistente, e retirarem conclusões. Esta camada pode ser entendida como uma lógica avançada, sendo que a camada Regras pode ser entendida como uma lógica mais simplificada.

A camada *Proof* permite verificar as conclusões retiradas na camada anterior, ao demonstrar formalmente através de provas que o novo conhecimento inferido está de acordo com a lógica utilizada.

A camada *Cryptography* encontra-se de modo transversal à arquitetura pois permite garantir e verificar que as declarações da Web Semântica provem de fonte fidedigna, usando por exemplo uma assinatura digital nas declarações RDF.

A penúltima camada é designada por *Trust*. Esta permite adicionar conhecimento através da derivação de novas declarações. A confiança é conseguida fundamentalmente verificando que as premissas provêm de fonte segura e que é usada na derivação de uma lógica formal.

A camada de topo representa as aplicações através das quais os utilizadores podem aceder à Web Semântica, através das interfaces disponibilizadas.

2.5. Padrões, linguagens e tecnologias da Web Semântica

A partilha de informação, com o mesmo significado para computadores e humanos requer a existência de marcadores semânticos padrão que sejam interpretados num vocabulário comum. Desta forma, a W3C fomentou e propôs o aparecimento de novas tecnologias e linguagens como XML, XML Schema, RDF, RDF Schema, SPARQL, SKOS (Simple Knowledge Organization System) e OWL, que permitam garantir a interoperabilidade e cooperação entre sistemas computacionais com o objetivo de desenvolver um modelo tecnológico para a partilha de conhecimento assistido por máquinas.

2.5.1. URI

O *Uniform Resource Identifier* (URI) é uma sequência compacta de caracteres que identificam um recurso físico ou abstrato (Berners-Lee, et al., 2005). Os URI's permitem a identificação inequívoca de diferentes recursos como documentos, imagens, páginas Web, entre outros. O espaço de nomes de um documento XML bem como os recursos de declarações RDF são também identificados por URI. Tudo pode ter um URI e a extensibilidade dos URI permitem a introdução de identificadores para qualquer entidade imaginável.

2.5.2. XML e XML Schema

A *Extensible Markup Language* (XML) surgiu para colmatar as limitações do HTML na implementação das novas aplicações Web. É uma linguagem de marcação, através do uso de etiquetas (tags). Na informática estas linguagens são tipicamente usadas para fornecer informação adicional (metadados) a partes de documentos de modo a descrevê-las em maior detalhe. O aparecimento do XML permitiu a definição da estrutura e sintaxe dos documentos. A grande vantagem é a possibilidade de ter diferentes vistas da mesma informação e personalizar a apresentação desta informação. Sendo uma linguagem independente da plataforma e de outras linguagens permite a interoperabilidade entre sistemas computacionais.

O esquema XML (*XML Schema*) é escrito de acordo com a sintaxe XML, inclui tipos de dados, herança, regras de combinação de esquema, suporta espaços de nomes e permite a ligação da informação. O espaço de nome XML (*XML Namespaces*) é uma coleção de nomes, identificados por uma referência URI, que é usada em documentos XML como tipos de elementos e nomes de atributos. A cada conjunto de nomes é associado um prefixo de identificação e as etiquetas são unicamente identificadas pelo prefixo seguido do nome local.

2.5.3. Resource Description Framework (RDF)

Metadados é informação sobre a informação. O uso efetivo de metadados requer o estabelecimento de convenções apropriadas para a semântica, sintaxe e estrutura. O RDF é uma linguagem que serve de base para o processamento de metadados, permitindo a interoperabilidade entre sistemas computacionais, de modo a efetuarem trocas de informação

na Web. Originalmente foi concebido para atribuir metadados aos recursos Web, mas tornou-se na base para adicionar informação semântica aos recursos. O RDF não descreve a semântica mas fornece uma base comum para a expressar. O modelo RDF permite representar informação sobre recursos na forma de um grafo dirigido, consistindo em declarações sobre recursos, tipicamente usando os triplos "sujeito", "atributo" e "valor".

2.5.3.1. Conceito

O modelo de dados RDF, que pode ser descrito como um diagrama de relacionamentos entre entidades, consiste em 3 tipos de objetos:

- Recursos – Tudo o que é descrito por uma expressão RDF é designado por recurso. Pode ser um sítio Web, uma página Web, um elemento HTML ou XML dessa página, etc. Os recursos são sempre designados pelo seu URI.
- Propriedades – Uma propriedade é um aspeto específico, característica, atributo ou relação usada para descrever um recurso. Cada propriedade tem um significado específico, define os valores permitidos, os tipos de recursos que pode descrever e o relacionamento entre propriedades. Cada propriedade é identificada por um nome.
- Declarações – Um recurso específico conjuntamente com uma propriedade e o valor dessa propriedade para esse recurso designa-se por declaração RDF.

As três partes individuais da declaração são designadas respetivamente por sujeito, predicado e objeto. O objeto da declaração pode ser outro recurso ou pode ser um literal ou outro tipo de dados primitivo definido por XML. Ao conjunto de propriedades que referem um mesmo recurso designa-se descrição. O RDF representa informação através de declarações numa estrutura do tipo Sujeito-Predicado-Objeto:

- Sujeito: o recurso descrito na declaração;
- Predicado: a propriedade do recurso descrito;
- Objeto: o valor da propriedade do recurso descrito.

A declaração "O email do Pedro é pedroTomas@gmail.com" pode ser decomposta em:

- Sujeito: Pedro
- Predicado: email
- Objeto: pedroTomas@gmail.com

2.5.3.2. Representação

Como vimos o modelo RDF pode ser visto como uma estrutura em forma de grafo. Os documentos RDF podem ser descritos, utilizando alguns formatos como Notation 3 (N3), N-Triples, Turtle ou XML (RDF/XML). De seguida veremos um exemplo de representação de um conjunto de declarações nas notações mais usuais. Foi utilizada a ontologia FOAF[6] para descrever estas declarações. Como os URI's tornam-se muito extensos, estes são divididos em espaço de nomes e nome local. Desta forma é possível utilizar a notação mais curta no formato *prefixo : nomeLocal*. O URI http://xmlns.com/foaf/0.1/age é assim semelhante a foaf : age, sendo foaf o prefixo para http://xmlns.com/foaf/0.1/.

2.5.3.2.1. Modelo em Grafo

As declarações RDF podem ser facilmente representadas como grafos. Os sujeitos e os objetos são representados como vértices ou nós, e os predicados como arestas rotuladas. Conceptualmente podemos pensar que o sujeito está ligado a um objeto através de um predicado. No exemplo ilustrado na figura seguinte existem dois recursos, o Pedro e a Sandra, ambos do tipo Pessoa. O Pedro tem 23 anos e o seu email é pedroTomas@gmail.com. A Sandra tem 21 anos e o seu email é sandraLeal@gmail.com. O Pedro conhece a Sandra.

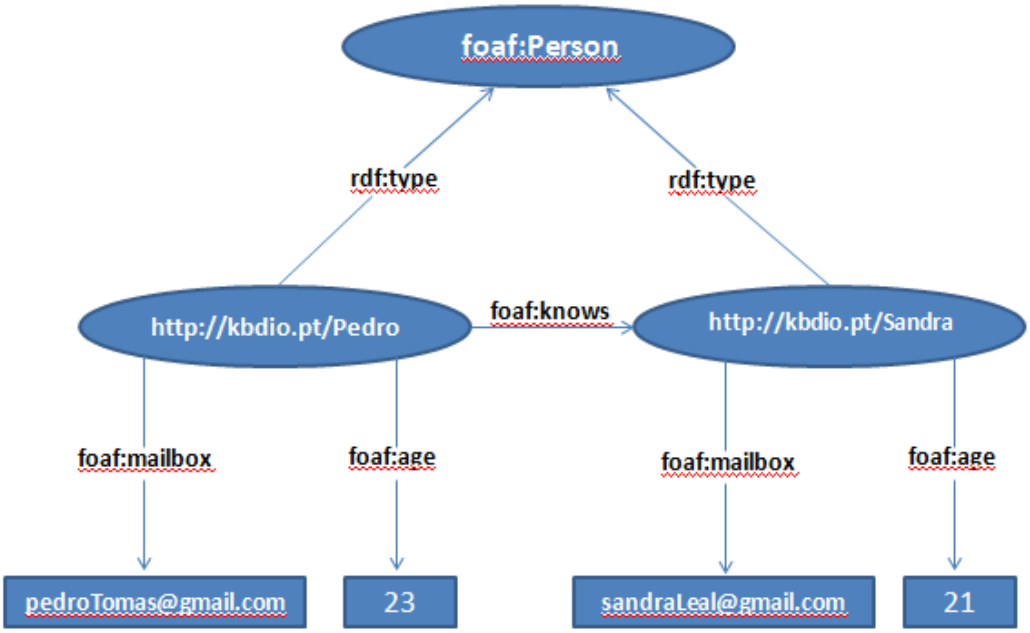

Fig. 2 – Modelo em Grafo de declarações RDF

[6] http://xmlns.com/foaf/spec/

2.5.3.2.2. Notação N3

As declarações RDF podem ser escritas na notação N3. Ao contrário do formato RDF/XML, especialmente utilizado entre sistemas computacionais, esta notação é bastante percetível para humanos. A notação toma a forma de "*sujeito predicado objeto .*". É possível abreviar esta notação no caso do um sujeito ter várias declarações, neste caso a notação será "*sujeito predicado objeto ; predicado objeto .*".

Como podemos observar no exemplo seguinte, nesta notação, os Literais são escritos entre parênteses e as referências a URI's são escritas entre "<" e ">", excetuando-se o caso da utilização prefixos para denominar os espaços de nomes destes URI.

```
@prefix rdf : <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf : <http://xmlns.com/foaf/0.1/> .
<http://kdbio.pt/Pedro> rdf:type foaf:Person ;
foaf:mailbox "pedroTomas@gmail.com" ;
foaf:age 23 ;
foaf:knows <http://kdbio.pt/Sandra> .
<http://kdbio.pt/Sandra> rdf:type foaf:Person ;
foaf:mailbox "sandraLeal@gmail.com" ;
foaf:age 21 .
```

2.5.4. RDFS

O RDF não tem mecanismos para descrever predicados nem suporta a descrição de relações entre predicados e outros recursos. O RDFS pode ser visto como a primeira tentativa para permitir expressar ontologias simples com a sintaxe RDF (Staab, et al., 2009).

O RDFS é uma linguagem de descrição de vocabulário que permite o uso de um conjunto de meta-propriedades como sejam classes (*rdfs:Class*), recursos (*rdfs:Resource*), propriedades (*rdf:Properties*), relações, etc. Estas primitivas são conceptualmente semelhantes às utilizadas no paradigma das linguagens de programação orientadas a objetos. O modelo de dados do RDFS permite construir o conceito de classe dentro de um domínio de informação. O conceito de classe permite agrupar objetos que tem características e comportamentos semelhantes. O conceito de herança permite a objetos herdar algumas propriedades e comportamentos da classe mãe.

Algumas dessa meta-propriedades usadas no RDFS são: elemento *rdf:type* permite especificar o tipo da instância; elemento *rdfs:Class* permite especificar que a instância é do

tipo classe; elemento *rdfs:subClassOf* permite modelar a hierarquia entre classes; elemento *rdfs:subPropertiesOf* permite modelar a hierarquia entre propriedades; elemento *rdfs:domain* permite restringir as instâncias de uma determinada propriedade a pertencerem a uma determinada classe; elemento *rdfs:range* permite restringir as instâncias de uma dada propriedade a terem valores de uma determinada classe.

Ao permitir a representação de algum tipo de conhecimento alicerçado nas suas principais primitivas de modelação (classes e propriedades, o relacionamento hierárquico entre classes e entre propriedades, as restrições de domínio e de tipo, e, as instâncias das classes), o RDFS permite a inferência simples. Contudo, apesar de poder ser considerada uma linguagem ontológica, tem um poder expressivo limitado, pois não suporta propriedades importantes como a negação, a disjunção, o inverso e a transitividade, não permitindo também efetuar restrições de cardinalidade e combinações booleanas de classes.

De seguida apresenta-se uma simples ontologia sobre animais de estimação.

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
@prefix myRdfs: <http://exemplo.pt/Pets#>
myRdfs:Animal rdf:type rdfs:Class .
myRdfs:Dog rdf:type rdfs:Class .
myRdfs:Cat rdf:type rdfs:Class .
myRdfs:Person rdf:type rdfs:Class .
myRdfs:Dog rdfs:subClassOf myRdfs:Animal .
myRdfs:Cat rdfs:subClassOf myRdfs:Animal .
myRdfs:isPetOf rdf:type rdf:Property .
myRdfs:isPetOf rdfs:domain myRdfs:Animal .
myRdfs:isPetOf rdfs:range myRfds:Person .
myRdfs:Boby rdf:type myRfds:Dog .
myRdfs:Pedro rdf:type myRfds:Person .
myRdfs:Boby myRdfs:isPetOf myRdfs:Pedro

2.5.5. OWL

A *Web Ontology Language* (OWL) é uma linguagem desenvolvida pelo W3C *Web Ontology Group*, com vista a possibilitar a publicação, extensão e partilha de ontologias na Web.

2.5.5.1. Ontologia

O termo ontologia provém de um conceito filosófico que se refere ao estudo do ser, efetuando uma descrição de entidades no mundo e do modo como elas se relacionam entre si.

A ontologia é um modelo de dados que representa um conjunto de definições de conceitos sobre uma temática em particular, i.e., uma especificação formal dos conceitos de um determinado domínio e das relações entre eles, de modo a permitir aos seus utilizadores partilharem a mesma terminologia e o mesmo significado, facilitando desta forma a comunicação entre eles (Guarino, 1998). Para uma ontologia poder ser interpretada de modo não ambíguo e ser usada por agentes de *software,* é necessário estabelecer a sintaxe e os formalismos semânticos. O uso de ontologias para a explicação do conhecimento implícito e explícito permite ultrapassar o problema da heterogeneidade semântica (Wache, et al., 2001).

A ontologia define um vocabulário comum necessário para a partilha de informação sobre uma temática específica. É uma descrição explícita formal de um domínio, consistindo em classes, que especificam os conceitos encontrados no domínio. As classes e as suas propriedades permitem descrever as várias características do modelo de dados e as suas restrições. Os fundamentos para modelar o domínio incluem classes, subclasses, propriedades, relações entre classes e propriedades, características das propriedades, restrições e instâncias.

Uma ontologia pode ser vista como um conjunto de entidades, também designadas por conceitos ou classes, que representam os conceitos do domínio e podem ser organizadas de forma hierárquica, permitindo uma especialização ou generalização de conceitos. As entidades têm propriedades, as quais correspondem a características e atributos que as descrevem. Estas propriedades podem ter restrições, que permitem aumentar a precisão da especificação. Cada entidade tem um conjunto de indivíduos ou instâncias. Assim, cada indivíduo tem as mesmas propriedades e restrições que a entidade a que pertence (Chaves, et al., 2011). A ontologia conjuntamente com o conjunto das suas instâncias de classe constitui a base de conhecimento. Desta forma, desenvolver uma ontologia inclui a definição de classes, a sua organização de forma hierárquica, a definição das suas propriedades, a descrição dos seus valores permitidos, e a criação de instâncias (Noy, 2001).

As ontologias tem tido uma importancia crescente em áreas como representação do conhecimento, integração de informação, recuperação de informação, comercio eletrónico e Web Semântica. A utilização eficaz de uma ontologia pressupõe não só a sua correta estruturação e definição da linguagem, como também, a possibilidade de utilização de ferramentas de raciocínio. O raciocínio pode ser usado para verificar a consistência da informação e para derivar relações implícitas, quer durante a fase de concepção da ontologia, quer durante a sua efetiva utilização (Staab, et al., 2009).

2.5.5.2. Linguagens de Ontologias

As linguagens ontológicas permitem a escrita, formal e explícita, de conceptualizações de modelos de domínios. Os principais requisitos destas linguagens são ter uma sintaxe e semântica bem definidas, ter um mecanismo eficiente de raciocínio e ter um poder expressivo suficiente (Staab, et al., 2009). Existem várias linguagens disponíveis para criar ontologias de modo a capturar conhecimento como sejam RDF, DARPA Agent Markup Language (DAML), Ontology Interchange Language (OIL), DAML+OIL e Simple HTML Ontology Extensions (SHOE). Atualmente, a OWL é a linguagem mais recente e contempla o mais completo conjunto de expressões para captura dos diferentes conceitos e relações que ocorrem nas ontologias (Wongthongtham, et al., 2007). As ontologias criadas com OWL permitem representar explicitamente a semântica exata de classes, das suas instâncias e propriedades, dentro de um mesmo domínio. A OWL é uma extensão do RDF/RDFS e tem um poder expressivo semântico maior do que o RDF e RDFS.

A semântica formal da OWL especifica como se pode derivar conhecimento que não seja explícito, i.e., factos que não se encontram na ontologia, mas que são possíveis de deduzir pela expressividade semântica e pelas regras de derivação inerentes à lógica descritiva subjacente. As deduções podem ser baseadas apenas numa única ontologia simples ou em combinações de ontologias.

Segundo a especificação OWL 1, a OWL é constituída por três sublinguagens variando em termos de poder expressivo e do processamento computacional necessário, que possibilitam aos utilizadores adotarem qualquer uma delas de acordo com as diferentes necessidades impostas pelo sistema que pretendem modelar:

- OWL Lite – é uma versão simplificada da linguagem. Permite a hierarquia de classes e restrições simples em propriedades como a cardinalidade binária (valores de 0 ou 1). Possui poder semântico suficiente para a especificação de ontologias simples, garantindo eficiência do ponto de vista computacional.
- OWL DL – é uma versão mais elaborada que a Lite, tendo esta como base. Tem uma correspondência com Lógicas de Descrição permitindo uma grande expressividade sem perda da completude computacional e da decidibilidade, i.e., garante a derivação de todos os factos verdadeiros e que a computação termina em tempo finito. Existe desta forma um bom compromisso entre a expressividade e a complexidade de raciocínio (Staab, et al., 2009). Dada a expressividade das Lógicas de Descrição, permite operações baseadas na teoria de

conjuntos, como a união, intersecção e complemento, e, permite efetuar restrições mais complexas ao nível das propriedades. Permite ainda a separação de tipos, i.e., uma classe não pode ser uma propriedade ou um indivíduo e vice-versa). Esta sublinguagem tem as propriedades lógicas e computacionais ideais para os sistemas de raciocínio.

- OWL Full – tem a expressividade da DL mas não impõe limitações sobre como os recursos se relacionam. Desta forma, tem a expressividade máxima e a liberdade sintática do RDF mas tem a limitação de não oferecer garantias quanto à eficiência computacional. As propriedades de completude e decidibilidade não são também garantidas.

Uma nova especificação da W3C, em 2009, reviu esta caracterização e genericamente acabou com a OWL Lite, transformando-a em 3 novas sublinguagens: OWL EL, QL e RL.

- OWL EL – permite algoritmos de tempo polinomial para todas as tarefas de raciocínio. É particularmente eficaz para aplicações em que são necessárias grandes ontologias, permitindo desta forma uma troca de poder expressivo pela performance.
- OWL QL – permite consultas com perguntas conjuntivas usando tecnologia padrão de base de dados relacionais. É particularmente eficaz para aplicações onde as ontologias são aligeiradas mas servem para organizar uma grande quantidade de indivíduos, ou onde é necessário aceder aos dados diretamente através de consultas relacionais.
- OWL RL – permite a implementação de algoritmos de tempo polinomial usando tecnologias de base de dados que estendem regras, operando diretamente nos triplos RDF. É particularmente eficaz para aplicações onde as ontologias são aligeiradas mas servem para organizar uma grande quantidade de indivíduos ou quando é necessário operar diretamente sobre os dados no formato RDF.

As representações de ontologias em linguagens baseadas em lógica tais como a Ontology Web Language (OWL) fornecem uma estrutura que suporta a disponibilização de informação baseada em inferência lógica (OWL Web Ontology Language Reference, 2004). As OWL DL oferecem um bom compromisso entre a expressividade e a computabilidade, e por isso surgiram várias aplicações designadas por *DL reasoners*, desenvolvidas maioritariamente por comunidades ligadas à inteligência artificial, que permitem verificar consistência, inferir, classificar e consultar dados ontológicos. Entre os mais utilizados na comunidade semântica destacam-se Pellet (Sirin, et al., 2007), FaCT++ (Tsarkov, et al., 2006) e RACERPro (Molle, et al., 2003).

2.5.5.3. Especificação OWL

2.5.5.3.1. Sintaxe

Apesar da existência de vários formatos sintáticos para especificar a linguagem OWL, a principal sintaxe do OWL assenta em RDF XML.

2.5.5.3.2. Cabeçalho

Os documentos OWL são também documentos RDF e por isso tem como raiz o elemento rdf:RDF que pode incluir a especificação de vários espaços de nomes. Segue-se o elemento owl:Ontology que pode conter comentários, versões de controlo e a inclusão de outras ontologias já existentes.

```
<rdf:RDF xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        xmlns:xsd ="http://www.w3.org/2001/XLMSchema#"
        xmlns:owl ="http://www.w3.org/2002/07/owl#">
<owl:Ontology rdf:about="">
        <rdfs:comment>My OWL ontology</rdfs:comment>
        <owl:priorVersion rdf:resource="http://www.ist.utl.pt/myOWL_old"/>
        <owl:imports rdf:resource="http://www.ist.utl.pt/tests"/>
        <rdfs:label>Test Ontology</rdfs:label>
</owl:Ontology>
```

2.5.5.3.3. Classes

O elemento owl:Class permite definir classes. De notar que este elemento é uma subclasse de rdfs:Class e que existem duas classes pré definidas, a classe owl:Thing e owl:Nothing. Todas as classes são subclasses de Thing e superclasses de Nothing. Por exemplo, podemos definir a classe cão, como sendo uma subclasse de animal. Podemos também dizer que cão e gato são classes disjuntas e que cão e canino são classes equivalentes.

```
<owl:Class rdf:ID="dog">
    <rdfs:subClassOf rdf:resource="#animal"/>
    <owl:disjointWith rdf:resource="#cat"/>
    <owl:equivalentClass rdf:resource="#canine"/>
</owl:Class>
```

2.5.5.3.4. Propriedades

Existem dois tipos de propriedades em OWL, as propriedades de dados e as propriedades de objetos. As primeiras são expressas como ***owl:DatatypeProperty*** e relacionam instâncias a valores tipificados em esquema XML. Normalmente, propriedades como idade, nome, altura, correio eletrónico, etc. são propriedades de dados. As segundas são expressas como ***owl:ObjectProperty*** e relacionam as instâncias das classes com outras instâncias (objetos). Por exemplo propriedades como "pertence a", "é animal de estimação de" são propriedades de objetos. Nestas propriedades é possível restringir o domínio e o tipo de valor da propriedade. É possível definir a propriedade inversa pelo elemento ***owl:inverseOf***. É ainda possível definir propriedades equivalentes pelo elemento ***owl:equivalentProperty***.

```
<owl:DatatypeProperty rdf:ID="age">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XLMSchema#nonNegativeInteger"/>
</owl:DatatypeProperty>

<owl:ObjectProperty rdf:ID="isPetOf ">
    <rdfs:domain rdf:resource="#animal"/>
    <rdfs:range rdf:resource="#person"/>
    <rdfs:subPropertyOf rdf:resource="#belongsTo"/>
    <owl:inverseOf rdf:resource="#hasPet"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasDomesticAnimal">
    <owl:equivalentProperty rdf:resource="#hasPet"/>
</owl:ObjectProperty>
```

2.5.5.3.5. Restrições de Propriedades

Genericamente, uma restrição de propriedade é definida pelo elemento ***owl:Restriction***, a qual contém o elemento ***owl:onProperty*** associado à restrição pretendida. Estas restrições

podem ser do tipo ***owl:allValuesFrom*** (todos os valores pertencem ao tipo especificado), ***owl:hasValue*** (tem um valor do tipo especificado) e ***owl:someValuesFrom*** (tem alguns valor do tipo especificado). Existem outros tipos de restrições como a cardinalidade em que é possível restringir valores máximos e valores mínimos utilizando os elementos ***owl:maxCardinality*** e ***owl:minCardinality***. No exemplo seguinte declara-se que os cães têm no mínimo 1 dono e que os cães não gostam de nenhum gato.

```
<owl:Class rdf:about="#dog">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#isPetOf"/>
            <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#dislikes"/>
            <owl:allValuesFrom rdf:resource="#cat"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

2.5.5.3.6. Propriedades Especiais

É possível definir diretamente algumas propriedades como a propriedade simétrica, transitiva, funcional ou funcional inversa, usando os elementos **owl:SymmetricProperty**, **owl:TransitiveProperty**, **owl:FunctionalProperty** e **owl:InverseFunctionalProperty**. A propriedade funcional define que a propriedade tem no máximo um valor para uma dada instância. A propriedade funcional inversa define que duas instâncias não podem ter o mesmo valor para aquela propriedade.

```
<owl:ObjectProperty rdf:ID="hasSameMaster">
    <rdf:type rdf:resource="&owl;TransitiveProperty" />
    <rdf:type rdf:resource="&owl;SymmetricProperty" />
    <rdfs:domain rdf:resource="#dog" />
    <rdfs:range rdf:resource="#dog" />
</owl:ObjectProperty>
```

2.5.5.3.7. Combinações Booleanas

A linguagem OWL permite a especificação de combinações booleanas de classes como a união, intersecção e o complemento de classes. O elemento complemento, ***owl:complementOf***, é análogo à disjunção e permite afirmar que as instâncias de uma classe não podem ser instâncias da outra classe complementar. O elemento união, ***owl:unionOf***, permite estabelecer que a nova classe é definida pela união das restantes classes, i.e., uma instância da nova classe deverá também pertencer a pelo menos uma das classes da união. O elemento intersecção, ***owl:intersectionOf***, permite estabelecer que a classe é definida pela intersecção das classes do argumento, i.e., que uma instância da nova classe deverá também pertencer a todas as classes da intersecção.

```
<owl:Class rdf:about="#dog">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:complementOf rdf:resource="#cat"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="cat">
    <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#siameseCat"/>
        <owl:Class rdf:about="#persianCat"/>
    </owl:unionOf>
</owl:Class>

<owl:Class rdf:ID="domesticAnimal">
    <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#animal"/>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#isPetOf"/>
            <owl:hasValue rdf:resource="#Person"/>
        </owl:Restriction>
    </owl:intersectionOf>
</owl:Class>
```

2.5.5.3.8. Enumerações

As enumerações são efetuadas através do elemento **owl:oneOf** e servem para definir classes listando todos os seus elementos.

```
<owl:Class rdf:ID="Time">
    <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="\#Hours"/>
        <owl:Thing rdf:about="\#Minutes"/>
        <owl:Thing rdf:about="\#Seconds"/>
    </owl:oneOf>
</owl:Class>
```

2.5.5.3.9. Instâncias

As instâncias das classes são declaradas como em RDF. De notar que apesar de duas instâncias poderem ter identificadores diferentes, é possível que sejam o mesmo indivíduo. Podemos criar a instância bobby da classe cão das duas seguintes formas equivalentes:

```
<rdf:Description rdf:ID="bobby">
        <rdf:type rdf:resource="#dog"/>
</rdf:Description>


<dog rdf:ID="bobby"/>
```

2.5.5.3.10. Tipos de dados

Os tipos de dados usados na linguagem OWL pertencem ao esquema XML e incluem os tipos de dados mais frequentemente utilizados como valores booleanos, valores inteiros, valores decimais, strings, datas, etc.. Contudo os tipos de dados mais complexos permitidos pelo esquema XML não podem ser utilizados pelo OWL.

2.5.6. SPARQL

O *Simple Protocol and RDF Query Language* (SPARQL) é simultaneamente uma linguagem de consulta a declarações RDF e também um protocolo que regula as mensagens entre os terminais SPARQL (endpoints) e os seus clientes. Estes terminais são aplicações

Web, que oferecem uma interface para pedidos do tipo HTTP GET ou POST, para acesso a um conjunto de dados RDF. Atualmente, existem na Web vários terminais de SPARQL disponíveis como o Bio2RDF ou o BioGateway. O OpenLink Virtuoso e o OpenRDF Sesame são dois exemplos de *software* que permitem a instalação de um terminal SPARQL.

A sintaxe da linguagem é semelhante à sintaxe SQL e disponibiliza um conjunto de comandos como SELECT, ASK, DESCRIBE e CONSTRUCT que permitem retornar dados mas não atualizá-los. Para isso é necessário utilizar uma extensão à linguagem denominada SPARUL, permitindo assim efetuar comandos como INSERT, MODIFY ou DELETE.

A consulta seguinte visa retornar todos os *emails* das pessoas que existem no conjunto de dados RDF, cuja idade é superior a 20 anos.

```
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?mail
WHERE {
?person rdf:type foaf:Person .
?person foaf:mailbox ?mail .
?person foaf:age ?age .
FILTER (?age >= 20)
}
```

2.6. Linked Data

Uma década depois de ter surgido, a Web Semântica não permite ainda integrar de forma inequívoca grande parte da informação da Web, de modo a que os sistemas computacionais possam compreender o seu conteúdo possibilitando uma contextualização entre estes e os utilizadores humanos. Contudo, o desenvolvimento e maturação das tecnologias que suportam a WS permitiram que estas fossem aplicadas com sucesso em projetos de diversas áreas do conhecimento. Nos tempos atuais, a Web Semântica é fundamentalmente constituída por ilhas de conhecimento (Berners-Lee T., 2001), isto é, nichos de conhecimento específico de uma temática em particular. Desta forma, são já várias as comunidades científicas que criaram repositórios de dados distribuídos, sobre temáticas particulares.

Recentemente, com a maturidade da WS, tem-se assistido a um movimento de convergência entre as diversas ontologias que possibilita a criação de um verdadeira plataforma de conhecimento através da interoperabilidade entre repositórios e ontologias. A possibilidade da interligação entre estes repositórios conduziu ao paradigma *Linked Data*

(Bizer, et al., 2011), um conjunto de princípios e tecnologias que visam a partilha e reutilização de informação de modo massivo, num espaço de dados global, a que as aplicações podem aceder, permitindo também a descoberta de novos dados.

A *Linked Data* refere-se aos dados disponibilizados na Web de tal forma que são facilmente processados por máquinas, sendo o seu significado definido explicitamente, e que estes dados são ligados bidireccionalmente a outros conjuntos de dados externos (Bizer, et al., 2009). Idealmente, as aplicações tenderão a operar sobre este vasto conjunto de dados distintos através de mecanismos de acesso padronizados. Desta forma, a *Linked Data* é o meio para se alcançar o objetivo da WS, da construção de uma Web global de dados, em que estes possam ser automaticamente processados e integrados por sistemas computacionais.

Foram várias as organizações que adotaram a *Linked Data* como um meio de disponibilizar a sua informação na Web. Este espaço global designado por Web de dados (*Web of Data*) forma um colossal grafo global constituído por biliões de declarações RDF de inúmeras fontes, cobrindo tópicos como localizações geográficas, pessoas, companhias, livros, genes, proteínas, fármacos, testes clínicos, entre outros (Bizer, et al., 2011).

O projeto Linking Open Data da comunidade W3C SWEO foi pioneiro na aplicação dos princípios da *Linked Data*. O seu objetivo é alargar a Web com dados partilhados, disponibilizando vários conjuntos de dados de diversas fontes e temáticas sob a forma de triplos RDF e efetuar ligações, também através de RDF, entre esses conjuntos de forma a permitir a sua interligação. Em Setembro de 2011 existiam no projeto 295 conjuntos de dados e mais de 31 biliões de triplos RDF interligados por cerca de 504 milhões de ligações.

As ligações RDF permitem navegar entre dados de diversas fontes. Os motores de busca podem desta forma seguir as ligações RDF e como resposta às consultas devolver não ligações para páginas HTML, mas informação estruturada, que poderá ser usada noutras aplicações. Os utilizadores e as aplicações podem começar a navegação num determinado conjunto de dados, e, progressivamente atravessar a Web seguindo as ligações RDF e não ligações HTML. Apesar de ser vasta, confusa e inconsistente esta base de dados global terá no futuro um valor imensurável.

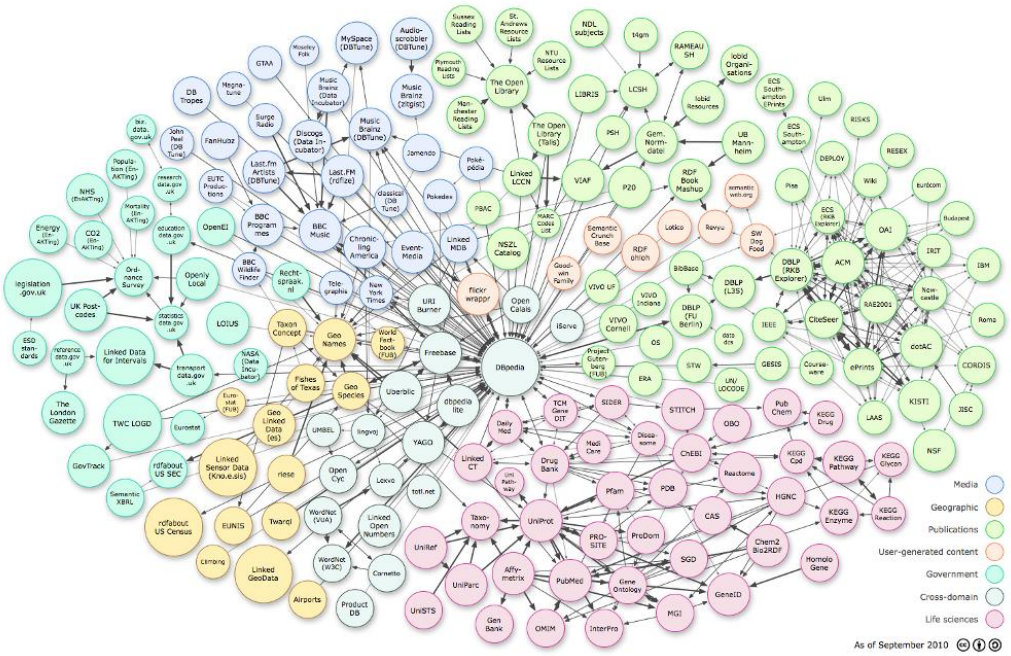

Fig. 3 – Fontes de dados interligados pelo projeto Linked Open Data[7]

2.7. Integração de Dados

A partilha da informação necessita de permitir o acesso aos dados mas também que os dados acedidos possam ser processados e interpretados pelo sistema remoto (Wache, et al., 2001). A integração de dados é a tarefa de combinar os dados que residem em diferentes fontes e fornecer ao utilizador uma visão unificada dos dados (Cali, et al., 2001). Para responder de forma eficiente à crescente complexidade da investigação, a integração deverá ser feita a vários níveis conceptuais. Deverá, por exemplo, integrar-se a informação do corpo humano e dos seus processos, os dados clínicos que um paciente tem guardado em várias clínicas, e integrar-se ainda, o conhecimento digital capturado em metadados, ontologias e modelos (Hunter, et al., 2010).

O significado da informação tem de ser compreendida pelos sistemas de modo a garantir a interoperabilidade semântica entre sistemas de informação heterogéneos. Os conflitos semânticos ocorrem quando os sistemas não fazem a mesma interpretação da informação (Wache, et al., 2001). Existem três causas principais para os conflitos semânticos:

- Conflitos de confusão – ocorrem quando sistemas computacionais atribuem significados diferentes à mesma informação;

[7] http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

- Conflitos de escala – ocorrem quando se usam diferentes sistemas de referência para medição;
- Conflitos de nome - ocorrem quando se utilizam esquemas de nomes diferentes.

Segundo Ouksel (Ouksel, et al., 1999) a interoperabilidade nos sistemas computacionais pode classificar-se de acordo com a dimensão heterogénea e das possíveis soluções em quatro níveis:

- Heterogeneidade dos sistemas – diferença no *hardware* e/ou sistemas operativos;
- Heterogeneidade sintática – diferença no formato de representação dos dados;
- Heterogeneidade estrutural – diferença nos modelos ou estrutura dos dados;
- Heterogeneidade semântica – diferença na interpretação do significado dos dados.

Existem duas abordagens genéricas para a integração de diferentes fontes de dados, designadamente a integração por *warehouse* e a integração por federação. É ainda possível efetuar uma combinação de ambas as abordagens (Cheun Kei-Hoi, 2007).

- Integração por *warehouse* – consiste na importação dos dados de fontes externas para um repositório local designado por *warehouse*. As consultas são executadas sobre os dados contidos neste repositório permitindo uma maior disponibilidade e eficiência de resposta. Esta abordagem elimina problemas como constrangimentos de rede, baixos tempos de resposta e indisponibilidade temporária dos dados. Tem ainda como vantagem permitir uma filtragem, validação, modificação e anotação dos dados obtidos. Contudo é necessário construir e manter a *warehouse*, existindo a possibilidade dos dados estarem desatualizados. Desta forma é necessária uma gestão cuidadosa de modo a periodicamente atualizar os seus dados.
- Integração por federação – consiste na obtenção de dados por consulta de fontes externas. Pressupõe um sistema de *Middleware* designado por *Mediador* que, após a consulta efetuada por um utilizador ou aplicação em tempo real, efetua a mediação entre o esquema federado único e os esquemas locais existentes nas diversas fontes de dados. As vantagens desta abordagem são a atualização dos dados e evitar e necessidade de um novo sistema de armazenamento. Por outro lado carece de problemas relacionados com constrangimentos de rede, com baixos tempos de resposta e com a indisponibilidade temporária dos dados.

2.7.1. Integração de dados através das Ontologias

As ontologias representam conceitos e as suas relações dentro de um domínio específico, permitindo uma representação conceptual computacionalmente processável. No contexto das ciências da computação e da informação, uma ontologia define um conjunto de primitivas representativas de um determinado domínio de conhecimento. Estas primitivas são fundamentalmente classes, atributos e relações, incluindo os seus significados e restrições, de forma a garantir a sua aplicação de forma lógica e consistente. No contexto dos Sistemas de Base de Dados uma ontologia pode ser vista ao nível da abstração do modelo de dados, i.e., a um nível semântico, enquanto que os modelos hierárquicos e relacionais encontram-se num nível lógico e físico. As ontologias permitem especificar vocabulário comum usado na partilha de dados entre diferentes sistemas, fornecer serviços para responder a questões, e, oferecer serviços para facilitar a interoperabilidade entre sistemas e bases de dados heterogéneas. Relativamente ao sistema de bases de dados, permite exportar, traduzir, inquirir e unificar dados entre diferentes sistemas e serviços. Devido à sua independência ao nível do modelo de base de dados, as ontologias são usadas para integrar bases de dados heterogéneas, permitindo a sua interoperabilidade entre sistemas díspares, e especificando interfaces para serviços independentes baseados em conhecimento (Gruber, 2007).

Inicialmente, o desenvolvimento de ontologias nas áreas científicas foi efetuado por peritos em áreas do conhecimento que possuíam poucas noções de desenvolvimento formal de ontologias conduzindo à criação de várias ontologias, muitas delas em sobreposição de conceitos, não permitindo a sua utilização interdisciplinar e integração. Atualmente tais desenvolvimentos são efetuados por equipas multidisciplinares, com noções em diversas áreas como biologia, ciências da computação ou filosofia, de modo a unificar o conhecimento específico de um determinado domínio (Antezana, et al., 2009c).

De acordo com a sua granularidade, as ontologias podem ser classificadas em 4 grupos como ilustra a figura 4 (Kienast, et al., 2011):

- Ontologias Topo de Nível – descrevem conceitos gerais independentes de domínios, que podem ser reutilizados nas ontologias de nível mais baixo. BFO (Basic Formal Ontology) e DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) são dois exemplos deste grupo de ontologias.

- Ontologias Topo de Domínio – descrevem conceitos chave de um domínio específico, servindo de interface entre as ontologias de topo de nível e de domínio. As ontologias UMLS (Unified Medical Language System) e a BioTop enquadram-se neste grupo.
- Ontologias de Domínio – descrevem apenas conceitos específicos de um dado domínio. OBO (Open Biological and Biomedical Ontologies) e GO (Gene Ontology) pertencem a ontologias deste grupo.
- Ontologias Locais – descrevem a semântica de recursos de informação simples.

Fig. 4 – Classificação das ontologias de acordo com a granularidade (Kienast R., 2011)

As abordagens para integração de dados baseadas em ontologias utilizam uma arquitetura de três camadas onde a camada semântica funciona como o mediador entre a camada de apresentação e a camada de dados. Desta forma, o acesso às diversas fontes de dados é efetuado de um modo transparente usando uma linguagem de consulta unificada, como por exemplo SPARQL. Segundo Wache (Wache, et al., 2001) existem três possíveis abordagens para esta integração, devidamente ilustradas na figura 5:

- Por ontologia única global – Nesta abordagem é apenas utilizada uma única ontologia global para integrar todos os dados de diferentes fontes. Esta ontologia pode ser constituída com base em ontologias especializadas, mas todos os recursos são relacionados com a ontologia global única (ver Fig. 5 a).
- Por múltiplas ontologias – Nesta abordagem a semântica de cada recurso é descrita por uma ontologia local, sendo necessária a criação de um mapeamento entre as ontologias locais (ver Fig. 5 b).

- Híbrida – Esta abordagem utiliza as duas anteriores para efetuar a integração de dados. Assim, os recursos são descritos por ontologias locais que utilizam um vocabulário global partilhado (normalmente também uma ontologia). Desta forma, o mapeamento é feito entre a ontologia partilhada e as ontologias locais, e não entre as ontologias locais (ver Fig. 5 c).

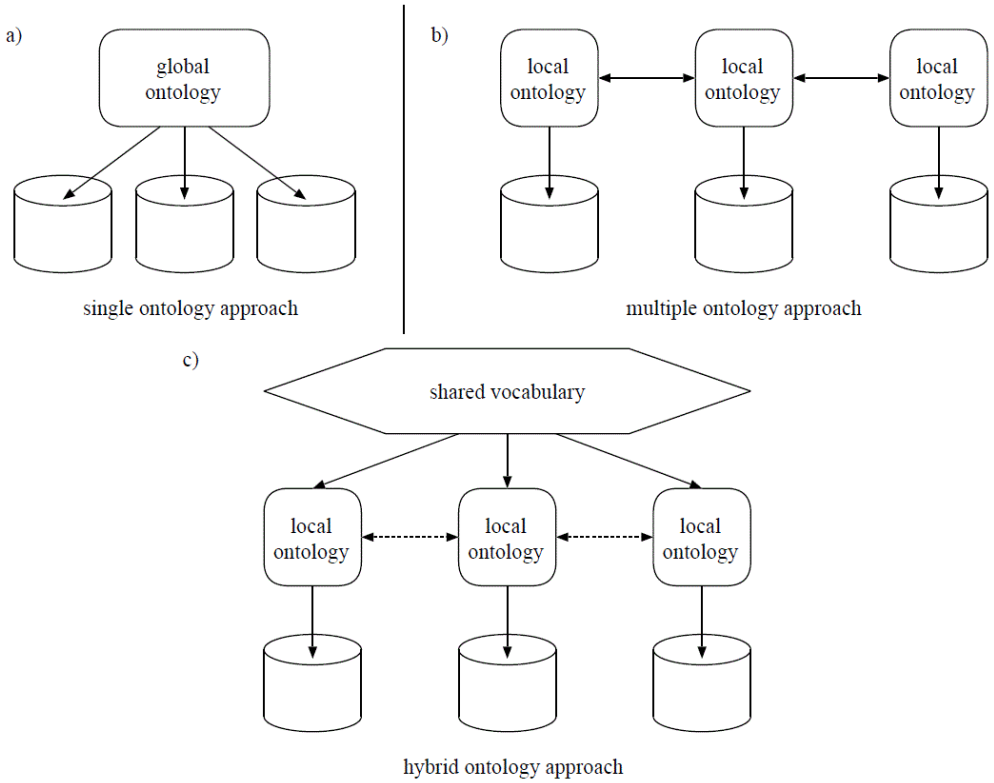

Fig. 5 – Abordagens de integração baseadas em ontologias, segundo Wache (Wache H., 2001).

2.7.2. Aspetos críticos da integração de dados

Seguidamente listam-se alguns dos aspetos mais críticos da integração de dados usando abordagens da Web Semântica, alguns deles ainda não completamente resolvidos pela tecnologia WS.

- Designação uniforme – A camada de base da arquitetura da tecnologia de Web Semântica define o uso de URI para designar recursos através da identificação global única.
- Extração de informação semântica de conhecimento já existente – O processo de extração da informação semântica associada aos dados que existem em diversas fontes deverá ser feita de modo automático ou semiautomático. Estas tarefas auxiliam na anotação de dados

de acordo com ontologias já existentes mas também poderão servir para a criação de novas ontologias associadas a um domínio específico.

- Desenvolvimento, manutenção e qualidade nas ontologias – As ontologias não deverão ser entidades conceptuais imutáveis, deverão por isso adaptar-se aos novos conhecimentos, podendo ser adicionados, alterados, substituídos ou removidos os conceitos e as entidades que define.

- Mapeamento, fusão, alinhamento e integração de ontologias – Não é possível com uma única ontologia cobrir os diversos domínios que existem. O mapeamento entre as diferentes ontologias, locais ou de domínio, é assim fundamental para colmatar a sua heterogeneidade permitindo desta forma a integração dos seus dados. A fusão, alinhamento e integração permite a reutilização de ontologias já existente para a formação de novas ontologias.

- Consultar dados RDF – Os repositórios de dados RDF deverão disponibilizar os chamados *SPARQL endpoints*, i.e., fornecer uma interface que permita aceder aos seus dados RDF através da linguagem SPARQL. Por exemplo, a Framework Jena Ontology API, com a sua extensão ARQ fornece uma abordagem de integração por federação, enquanto a Framework Sesame tem uma abordagem de integração por warehouse.

- Visualização – Deverá ser disponibilizada uma interface gráfica para navegação e visualização nos grafos RDF, de modo a permitir ao utilizador fácil e eficazmente encontrar informação relevante.

- Disponibilidade – Para se conseguir a disponibilidade dos dados é necessário ter presente que apesar de muitas ontologias serem de livre acesso, outras existem que carecem de licenciamento. Os dados anotados pelas ontologias poderão também estar indisponíveis por dificuldades técnicas, por restrições de ordem legal ou devido à privacidade dos dados.

- Formatos de ontologias diferenciados – Apesar da Web Semântica definir ontologias no formato OWL, existem vários repositórios que utilizam outros formatos como por exemplo OBO. Assim, é necessário efetuar um mapeamento desses formatos para um formato comum, por exemplo OWL.

- Uso de diferentes línguas – O uso de diferentes linguagens, quer nas ontologias quer nos dados por elas anotadas, podem ser foco de problemas. Apesar da generalidade das ontologias desenvolvidas pelas comunidades científicas serem definidas em língua inglesa, os dados por elas anotados poderão não ser, causando dificuldades de usabilidade para os utilizadores.

2.8. Bio-ontologias

As ontologias do domínio da biologia e da biomedicina designam-se por Bio-ontologias. A generalidade destas ontologias encontram-se disponíveis sobre a coordenação do projeto Open Biomedical Ontologies (OBO) (Smith, et al., 2007). Este projeto tem como objetivos coordenar as ontologias de modo a estruturar o vocabulário partilhado nos domínios biológico e biomédico, garantindo a inexistência de sobreposição de conceitos, e estabelecer um conjunto de princípios com vista a estruturar o desenvolvimento de novas ontologias. Atualmente, o projeto tem sob sua alçada mais de uma centena de ontologias, que são largamente aceites como referência na comunidade científica.

As Bio-ontologias podem ser implementadas através de diferentes linguagens de representação de conhecimento, que variam na sua sintaxe, semântica, expressividade e capacidade de raciocínio. As mais utilizadas na bioinformática são a OBO e a OWL, já referida na Secção 2.5.5. Normalmente, a capacidade destas linguagens não é totalmente utilizada nas bio-ontologias devido à falta de expressividade e à falta de rigor semântico, resultando numa limitação da representação do domínio (Aranguren, et al., 2008). Para colmatar estas lacunas, tem sido proposto uma metodologia de padrões de desenho para ontologias, cujo conceito é semelhante ao usado nas linguagens orientadas a objetos. As *Ontology Design Patterns* (ODP) são soluções já testadas para problemas de modelação que repetidamente aparecem aquando do desenho de ontologias (Aranguren, et al., 2008).

2.8.1. Disease Ontology

A Disease Ontology (DO) é uma ontologia pertencente à fundação OBO, que foi desenvolvida com a finalidade de proporcionar à comunidade biomédica descrições consistentes, reutilizáveis e sustentáveis de termos de doenças humanas, características de fenótipos e vocabulário de conceitos médicos relacionados. A DO integra semanticamente vocabulários médicos e de doenças através de um extenso mapeamento cruzado dos seus termos com outras ontologias como MeSH, ICD, NCI’s thesaurus, SNOMED and OMIM.

O navegador disponibilizado em http://disease-ontology.org permite navegar pelos conceitos da ontologia, estando estes organizados num grafo acíclico dirigido, de modo que ao navegar a partir da raiz obtemos termos cada vez mais específicos.

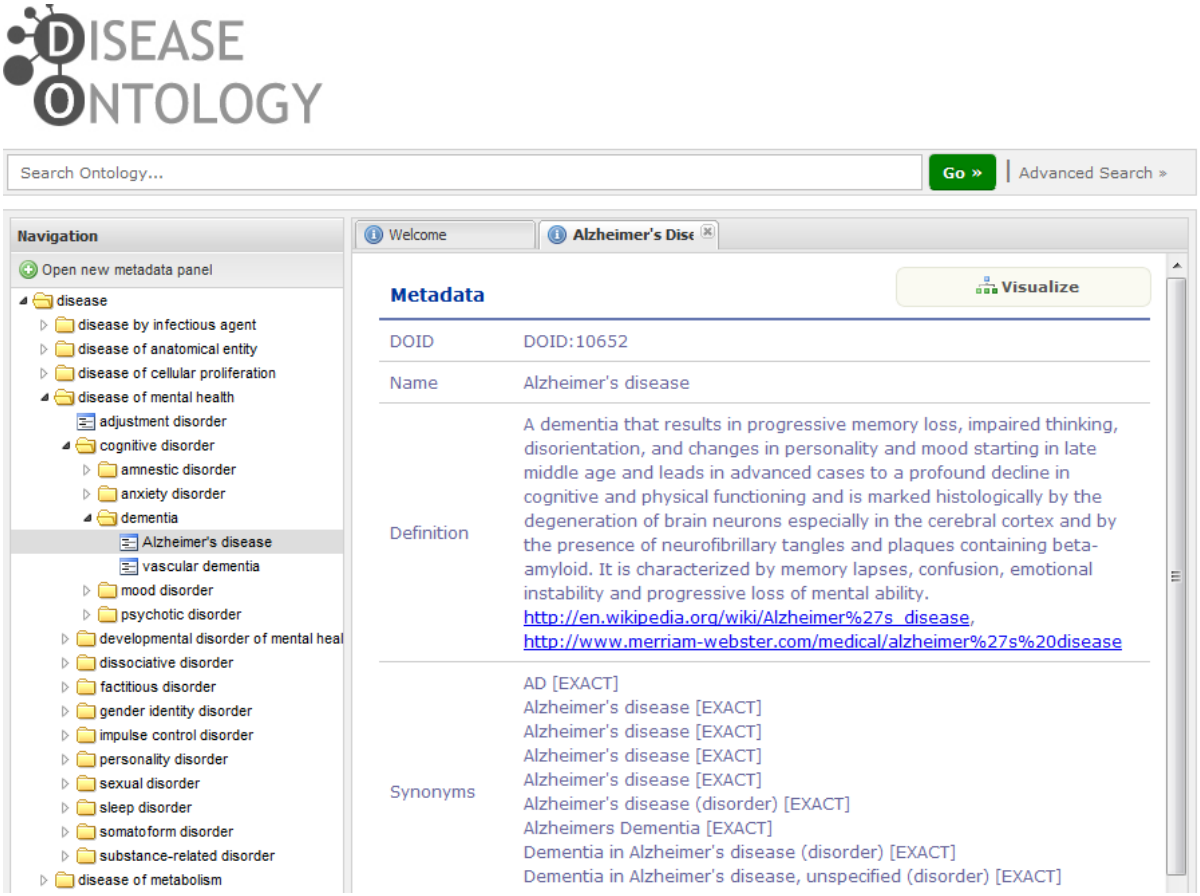

Fig. 6 – Navegador da Disease Ontology

2.9. Biologia de Sistemas Semânticos

A Biologia de Sistemas assenta num modelo matemático ou computacional que permite a simulação do comportamento de um sistema biológico complexo. As simulações servem para validação do modelo. Simultaneamente, as simulações permitem efetuar a predição do comportamento do sistema sob novas condições e conduzir ao aparecimento de novas hipóteses. A recente abordagem da Biologia de Sistemas utiliza uma descrição semântica do conhecimento sobre sistemas biológicos de modo a facilitar a análise de dados integrados. Esta abordagem permitiu a criação de modelos de alta qualidade de sistemas biológicos

(Antezana, et al., 2009a). O termo Biologia de Sistemas Semânticos foi inicialmente atribuído por Erick Antezana a esta fusão da Biologia de Sistemas com a Web Semântica.

A Biologia de Sistemas Semânticos tende a seguir uma nova abordagem de investigação, assente em parte nas premissas de Hiroaki Kitano sobre a pesquisa conduzida por hipóteses (Kitano, 2002). Inicialmente o conhecimento é integrado numa base de dados. Os dados são verificados para determinar a sua consistência. A consulta e o raciocínio automático conduzem a hipóteses que permitem modelar novas experiências. A partir da experimentação são gerados novos dados que podem validar ou negar as hipóteses. A informação gerada é depois integrada no repositório, formando um processo cíclico (Antezana, et al., 2009c). O novo ciclo iterativo de investigação em Biologia de Sistemas Semânticos é ilustrado na Fig. 5.

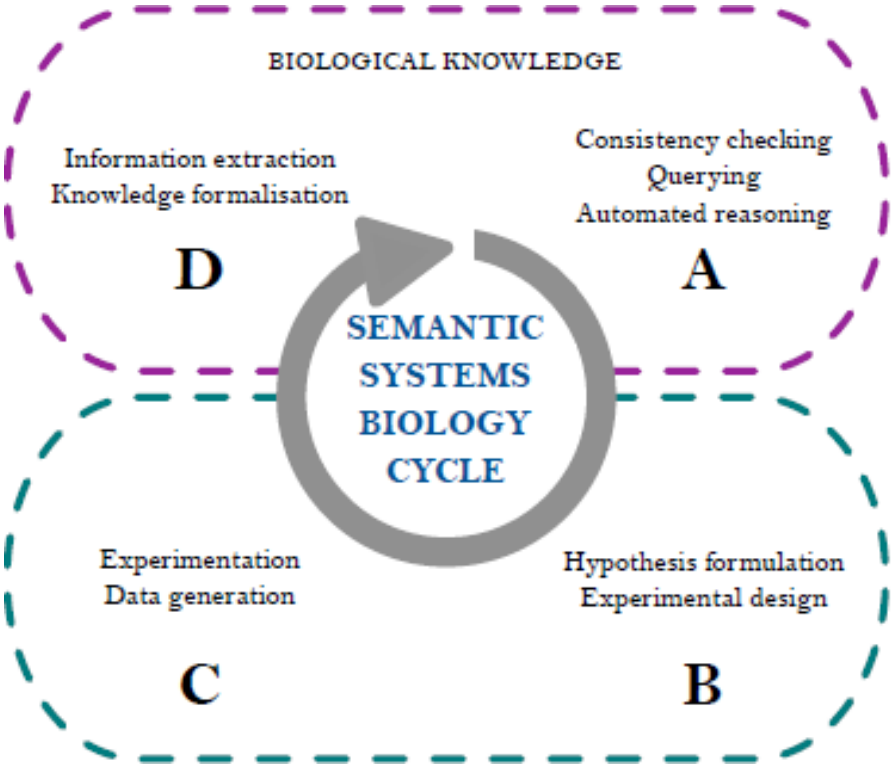

Fig. 7 – Ciclo de investigação na Biologia de Sistemas Semânticos (E., et al., 2009).

2.10. Aplicações Semânticas na área da Biomedicina

De seguida listam-se alguns projetos que utilizam a tecnologia Web Semântica na área da biomedicina.

- Bio2RDF (Belleau, et al., 2008) – É um projeto que utiliza tecnologia Web Semântica para fornecer dados interligados como suporte para a descoberta de conhecimento biológico.

Utiliza técnicas de integração de dados, quer semânticas quer sintáticas. Neste projeto, os documentos de bases de dados públicas mais relevantes em bioinformática, como Kegg, PDB, MGI e NCBI são disponibilizados no formato RDF, possibilitando a sua reutilização e integração.

- BioGateway (Antezana, et al., 2009a) – É um projeto que segue uma abordagem assente na Biologia de Sistemas Semânticos. Estes sistemas usam a descrição semântica de conhecimento sobre sistemas biológicos para facilitar a análise integrada de dados, utilizando uma abordagem "bottom-up" e efetuando uma geração de hipóteses conduzidas pelos dados. Este projeto permite o acesso a uma base de conhecimento centralizada, que guarda informação no formato RDF de diversas fontes públicas, tais como, as ontologias da fundação OBO, do projeto GOA (Gene Ontology Annotations), NCBI taxonomy e SwissProt. O sistema permite consultas usando a linguagem SPARQL, e dispõe de uma interface gráfica que permite a navegação através dessa informação e obtenção de resultados sob a forma de uma rede de recursos.

- BioPortal (Noy, et al., 2009) – É uma das aplicações mais conhecidas no domínio das ontologias. Permite o acesso às ontologias biomédicas mais comuns e disponibiliza ferramentas para trabalhar com elas. Permite navegar a biblioteca de ontologias, pesquisar um termo em várias ontologias, navegar mapeamentos entre termos de diferentes ontologias, anotar texto com termos de ontologias, pesquisar recursos biomédicos para um dado termo, etc.

- Cell Cycle Ontology (Antezana, et al., 2009b) - O projeto integra e gere conhecimento sobre compostos celulares envolvidos na execução do ciclo da célula bem como sobre a sua regulação. Este conhecimento é extraído de um conjunto de fontes já existentes tais como GO, UniProt, InAct, GOA e NCBI. Após a sua integração, a ontologia é disponibilizada através de formalismos de representação de conhecimento, tais como OBO, OWL e RDF. A existência de um serviço de consultas através da linguagem SPARQL permite que utilizadores, quer humanos quer sistemas computacionais, efetuem consultas a uma base de dados no formato RDF.

- OntoCAT (Adamusiak, et al., 2011) – Esta ferramenta não é mais do que uma biblioteca de *software* que permite a pesquisa e integração de ontologias de fontes heterogéneas em larga escala. Esta solução configurável e robusta disponibiliza uma interface de programação para consultas a recursos de ontologias heterogéneas, incluindo ficheiros locais com formato OBO ou OWL, e acesso a *Web Services* como Ontology Lookup Service

(OLS) e o BioPortal. Esta biblioteca fornece uma API simples para manipular ontologias. Pode ser facilmente usada em programas desenvolvidos em Java, Bioconductor/R (pacote do OntoCAT) e em clientes de RESTful *Web Services*.

- RICORDO (Bono, et al., 2011) – Este projeto é baseado em métodos de representação de conhecimento formais, incluindo o uso de ontologias e de ferramentas associadas. A abordagem seguida neste projeto foi a de suportar a interoperabilidade semântica entre recursos biomédicos através de uma ontologia baseada em anotações. Os DMR (Data and Model Resources) são, basicamente, os dados eletrónicos gerados pela prática ou pela investigação na área biomédica, como por exemplo dados de imagiologia, modelos matemáticos ou texto simples. Estes metadados, sob a forma de anotações, são declarações que mapeiam os recursos em entidades ontológicas. Esta abordagem retira ambiguidade às anotações em recursos, podendo estas ser processadas por sistemas computacionais capazes de inferir novos dados.

2.11.Sumário

Devido aos progressos tecnológicos, quer na área da biomedicina quer na área das ciências da computação, os cientistas tem atualmente acesso a uma quantidade de dados inimaginável e de inquestionável valor. Contudo, estes dados de vários domínios do conhecimento são representados sob formas e formatos distintos, residindo em múltiplos repositórios de informação dispersos pelo mundo. Para colmatar esta heterogeneidade e possibilitar a integração entre estes dados distintos, as comunidades científicas tendem a utilizar hoje uma abordagem semântica denominada Web Semântica.

A tecnologia Web Semântica permite adicionar significado aos dados, i.e., adicionar anotações sob a forma de metadados, por exemplo através de triplos RDF, possibilitando aos sistemas computacionais processar esta nova informação. Estes metadados são em geral grafos dirigidos constituídos por um conjunto de relações entre dados, que fornecem desta forma a base conceptual para a definição de conceitos e a atribuição de significado a qualquer recurso na Web. A possibilidade de navegação nestes grafos dirigidos viabiliza a integração de dados e a sua consulta através de uma linguagem padronizada criada para o efeito, a linguagem SPARQL.

As tecnologias WS disponibilizam linguagem como RDFS e OWL para a definição de ontologias. O sucesso dos sistemas computacionais que utilizam esta tecnologia está

diretamente ligado à escolha ou construção de ontologias que contribuam fortemente para um sistema de gestão de dados eficaz e eficiente que integre dados heterogéneos. A ontologia poderá ser construída combinando ontologias de domínio já existentes, parte delas, ou ainda, estendendo essas ontologias. Estas ontologias aplicacionais têm um papel importante na definição do domínio de conhecimento que irá ser explorado, e desta forma, facilitam a integração de diferentes tipos de informação, como sejam dados genómicos e clínicos. Tal sistema combina métodos de extração de dados, de conversão de vários formatos de dados e uma variedade de fontes de informação (Aranguren, et al., 2008).

Nos últimos anos tem existido um incremento na quantidade e qualidade de ontologias, incluindo as do domínio da biomedicina, como atesta a fundação OBO. O esforço de interligação de ontologias de domínio e de topo de domínio, de forma a possibilitar a criação de um espaço virtual global de acesso aos dados, é conhecido como *Linked Data*.

Atualmente o uso das tecnologias da WS na comunidade científica tem vindo a ganhar notoriedade, criando condições favoráveis para uma maior integração de dados e, consequentemente, para a inferência de novo conhecimento. A Biologia de Sistemas Semânticos surge naturalmente como uma nova abordagem para a ciclo de investigação na área da biologia.

Um número significativo de projetos na área da biomedicina utilizam o ciclo de investigação da Biologia de Sistemas Semânticos e tem como objetivo a pesquisa e a extração de conhecimento de várias fontes. Relativamente à localização dos dados, seguem abordagens federativas e de *warehouse*. A primeira abordagem faz uso de uma camada de *software* que virtualiza os acessos às diferentes fontes. Algumas dessas fontes originais não estão sob formatos RDF ou OWL sendo por isso necessário efetuar uma transformação. Os novos dados são guardados depois em repositórios apropriados (e.g., Virtuoso), os quais permitem a formulação de consultas que abrangem diversas fontes. A segunda abordagem necessita de uma camada adicional denominada *middleware* na qual os dados são extraídos das fontes originais e guardados num repositório centralizado garantindo um mais rápido acesso.

3. Ontologia de Testes Médicos Neuropsicológicos

No âmbito desta dissertação foi desenvolvida uma ontologia que serve de base para a anotação de dados médicos respeitantes a testes neuropsicológicos. Esta ontologia, designada por Neuropsychological Test Ontology (NTO), mapeia e descreve conceitos no âmbito de testes neuropsicológicos aplicados a pacientes potencialmente com a doença de Alzheimer. Esta estrutura de representação do conhecimento serviu de base para o restante desenvolvimento do sistema. Os dados clínicos reais disponíveis originalmente como ficheiros Excel foram anotados semanticamente de acordo com a NTO. Estes dados são relativos a 1642 avaliações médicas efetuadas ao longo dos últimos 22 anos por médicos do Instituto de Medicina Molecular (IMM) a 92 dos seus pacientes. Cada avaliação contém, para além de dados relativos ao paciente em questão, os resultados da aplicação de vários testes neuropsicológicos. No final da avaliação o médico atribui ao paciente um diagnóstico relativamente à doença de Alzheimer, i.e., diagnostica o paciente como sendo "normal" ou como tendo a doença de Alzheimer numa das suas fases.

3.1. Testes neuropsicológicos

A aplicação de testes neuropsicológicos têm como objetivo avaliar a saúde mental dos pacientes, identificando e quantificando sintomas cognitivos, funcionais e comportamentais com vista a efetuar um diagnóstico, e, avaliar e monitorizar a evolução da doença de Alzheimer. Existe uma grande diversidade destes testes disponíveis, sendo alguns aplicados isoladamente ou em conjunto com outros formando as denominadas baterias de testes, das quais se destaca a Bateria de Lisboa para avaliação da Demência, por estar validada para a população Portuguesa.

3.2. Desenvolvimento da ontologia

Apesar dos esforços e da já longa experiência no desenvolvimento de ontologias, na comunidade ainda não existe concordância relativamente à metodologia para a construção de ontologias. No desenvolvimento da NTO segui os princípios e critérios descritos em Ontology Development 101 para o desenho de ontologias (Noy, et al., 2001). Noy e McGuinness propõem para o desenho de ontologias uma metodologia iterativa contemplando 7 etapas:

determinação do domínio e do âmbito da ontologia; possibilidade de reutilizar ontologias já existentes; enumeração de todos os termos e relações do domínio de conhecimento; definição de classes e sua hierarquização; especificação das propriedades e das relações entre as classes; definição do tipo de valor das classes e das suas propriedades, incluindo aspetos como a cardinalidade; criação das instâncias individuais das classes do domínio.

A definição de conceitos relevante para o domínio foi conseguida analisando um vasto conjunto de publicações e de entrevistas com membros do projeto Neuroclinomics, especialmente com médicos do Instituto de Medicina Molecular. De salientar que a hierarquização dos testes neuropsicológicos foi elaborada segundo a publicação "A Compendium of Neuropsychological Tests: Administration, Norms and Commentary, Third Edition" (Strauss, et al., 2006).

De seguida passo a descrever os conceitos e suas relações com outros conceitos tidos em conta para a criação da ontologia. Um médico tem um conjunto de pacientes que são submetidos periodicamente a um conjunto de avaliações médicas. Quer os médicos, quer os pacientes têm um conjunto de informações pessoais como sejam o seu nome, data de nascimento, sexo, correio eletrónico, fotografia, escolaridade, etc. que poderão ser disponibilizados. As avaliações médicas, efetuadas numa data específica, correspondem à execução de um conjunto de testes médicos neuropsicológicos. À data da avaliação médica, o paciente tem um conjunto relevante de dados, como seja a duração da doença de Alzheimer, o grupo da bateria de testes de Lisboa onde se enquadram e a sua idade, que importam especificar. No final desta avaliação, o paciente irá ter um diagnóstico, relativo à doença de Alzheimer, o qual poderá especificar uma fase desta doença. Cada teste neuropsicológico poderá ter um resultado, normalizado ou não, uma duração e uma versão. Os testes poderão também conter vários componentes, isto é, várias partes do teste, cada uma com a sua duração e o seu próprio resultado, podendo este ser normalizado. Os testes e os seus componentes poderão estar associados a algumas características como sejam a similaridade, uso de vocabulário, desenho de objetos, etc. Existem alguns testes neuropsicológicos mais complexos, designados por baterias de testes, que agrupam um conjunto de outros testes mais simples. A Bateria de Lisboa para Avaliação da Demência (BLAD) é a bateria de testes principal utilizada nos dados reais disponíveis, sendo complementada por outros testes tais como o Trail Making Test, o Toulouse-Pierón Test ou o Californian Verbal Learning Test.

Esta aquisição de conhecimento permitiu definir questões às quais a ontologia deveria ser capaz de responder como por exemplo:

- Quais os testes que se podem administrar a um paciente de idade superior a 70 anos, se apenas dispomos de 10 minutos?

- Quais os testes que se relacionam com as palavras como por exemplo "Similarity"?

- Em que datas foram observados resultados inferiores a 3.1 no componente "Total" do teste "Letter Cancelation Task" para um determinado paciente?

- Quais os pacientes diagnosticados com "Pré-MCI" numa determinada avaliação que têm no componente "Total" do teste "Verbal Paired Associated" valores superiores a 13,5?

A NTO reutiliza ontologias já existentes, especificamente a Human Disease Ontology (DOID) [8] e a Friend of a Friend (FOAF) [9] que permitem reutilizar, respetivamente, os conceitos e propriedades de Doença de Alzheimer e de pessoa. A DOID está integrada na fundação Open Biomedical Ontologies e permite a integração de dados biomédicos associados a doenças humanas. Nesta ontologia é feita uma hierarquização das doenças humanas bem como da sua correta terminologia e conceitos relacionados. Através da utilização desta ontologia é possível reaproveitar os conceitos de doença e identificar univocamente a doença de Alzheimer, pelo IRI http://purl.obolibrary.org/obo/DOID_10652. A FOAF é uma ontologia que define um vocabulário RDF para descrever as propriedades básicas das pessoas (nome, idade, título, email, homepage, etc.) e os seus interesses, atividades e relações. Através da utilização da FOAF é possível descrever os pacientes, os médicos e as relações entre estes.

A NTO consiste em classes (que representam os conceitos gerais de domínio como Pessoa, Avaliação médica, etc.), subclasses (que correspondem a especializações da superclasse como Paciente, Médico), instâncias (que representam dados específicos como sejam as fases da Doença de Alzheimer "Mild Cognitive Decline", "Moderate Cognitive Decline", etc.), propriedades (que representam as relações binárias entre os conceitos e instâncias como por exemplo a propriedade "has Medical Test" que relaciona Avaliação Médica com Teste Médico) e restrições (que limitam as relações como por exemplo o Teste de Toulouse-Piéron tem no máximo 1 componente de teste relativo ao índice de dispersão).

[8] http://purl.obolibrary.org/obo/doid.owl

[9] http://xmlns.com/foaf/0.1/

A classe Score foi criada para permitir a representação de dados estruturados complexos que possam simultaneamente ter várias dimensões, por exemplo um valor quantitativo, qualitativo ou unidade de medida. Desta forma é possível representar conceitos como a duração da doença como hasValue=2 e hasUnit="years", ou representar o diagnóstico de uma avaliação como hasValue=1.0 e hasQuality="MCI" (uma das fases da doença de Alzheimer). Existem ainda testes em cujo resultado tem relevância não só o valor obtido na sua execução, mas também o tempo de execução do mesmo. É possível representar estes resultados com duas instâncias da classe Score, a primeira com hasValue=19.4, hasUnit="in 20" e hasQuality="Excelent" e a segunda com hasValue=56 e hasUnit= "seconds".

Na tabela seguinte descreve-se textualmente algumas das entidades (classes) mais relevantes no domínio, a sua hierarquização e as suas relações com outras entidades. As subclasses dos testes psicológicos não são descritas de forma exaustiva neste documento, devido ao seu grande número e à sua similaridade, podendo fazer-se a devida analogia com "Neuropsychological Test", "Memory Test" e "Clock Drawing Test".

Classe	Descrição	Extensões	Relações
Doctor	Representa a entidade médico responsável pelo paciente e pela execução de uma avaliação médica.	Especialização da classe Person da ontologia FOAF	Relação com Evaluation, pois é a entidade que supervisa o episodio médico ou avaliação médica.
Pacient	Representa a entidade paciente que executa os testes neuropsicológicos.	Especialização da classe Person da ontologia FOAF	Relação com Evaluation pois é a entidade que sobre a qual recai a avaliação médica.
Patient Data	Representa dados específicos do paciente numa determinada avaliação médica		Relação com Evaluation, pois é a informação relativa à avaliação. Relação com Score, relativamente ao diagnóstico do paciente na avaliação médica.
Score	Representa a entidade resultado para dados complexos, sendo possível atribuir um valor numérico, mas também a unidade, a qualidade e a escala desse valor.	Generalização da classe Diagnostic.	Relação com PatientData, com MedicalTest, TestComponent e com Person.
Evaluation	Representa a entidade avaliação médica ou episódio médico, na qual o paciente é observado pelo médico e onde é sujeito a um conjunto de testes médicos. Essa avaliação pode resultar num diagnóstico.		Relação com Patient, pois a avaliação é de um paciente. Relação com Doctor, pois a avaliação é efetuado por um determinado médico. Relação com PatientData, pois nessa avaliação o paciente tem um conjunto de dados específicos. Relação com MedicalTest, pois a avaliação pressupõe a elaboração de um conjunto de testes médicos. Relação com Diagnosis, pois a avaliação pode resultar num diagnóstico.

Diagnosis	Representa a entidade diagnóstico de um paciente relativamente a uma doença, podendo enquadrar-se numa fase da doença.	Especialização de Diagnosis	Relação com Evaluation, pois o diagnóstico resulta de uma determinada avaliação médica. Relação com Disease, pois o diagnóstico respeita a determinada doença. Relação com Disease Stage, pois o diagnóstico pode enquadrar-se numa determinada fase da doença.
Disease	Representa a entidade doença	Generalização de Alzheimer's Disease.	Relação com Diagnosis, pois o diagnóstico é relativo a uma doença. Relação com Disease Stage, pois algumas doenças tem evoluções designadas por fases de doença.
Alzheimer's Disease	Representa a entidade doença de Alzheimer, equivalente à entidade DOID 10652 da ontologia DOID.	Especialização de Disease.	
Disease Stage	Representa a entidade fase de determinada doença	Generalização de Alzheimer's Stage	Relação com Diagnosis, pois o diagnóstico pode enquadra-se numa fase de determinada doença.
Alzheimer's Stage	Representa a entidade fase da doença de Alzheimer.	Especialização de Disease Stage	
Medical Test	Representa a entidade teste médico, na qual numa determinada avaliação médica um paciente é submetido a exames médicos. Estes exames podem ser constituídos por vários componentes.	Generalização dos testes do foro psicológico. Um teste médico, designado por bateria de testes pode ter vários outros testes médicos.	Relação com Evaluation, pois os testes são efetuados no âmbito de uma determinada avaliação médica. Relação com TestComponent pois os testes podem conter vários componentes que importa especificar.
Test Component	Representa a entidade componente do teste, na qual os testes se podem subdividir em partes diferenciadas.	Generalização de vários componentes de testes como CVLT_Component.	Relação com MedicalTest, os componentes pertencem a um teste. Relação com Score, pois o resultado, a média e a duração do componente são caracterizados pela entidade Score.
Test Characteristics	Representa a entidade característica do teste ou do componente do teste, na qual é possível determinar uma característica ou atributo qualitativo		Relação com MedicalTest e com TestComponent, pois os testes ou os seus componentes podem evidenciar algumas características especificas.
Psychological Test	Representa a entidade teste médico do foro psicológico.	Especialização de Medical Test. Generalização de NeuropsychologicalTest, AttitudeTest, etc.	
Neuro psychological Test	Representa a entidade teste médico do foro neuropsicológico.	Especialização de PsychologicalTest. Generalização de DementiaSpecificTest, IntelligenceTest, MemoryTest, etc.	
Memory Test	Representa o teste médico neuropsicológico do tipo memória.	Especialização do Neuropsychological Test. Generalização de ClockDrawingTest, CubeTest, etc.	
Clock Drawing_Test	Representa o teste médico neuropsicológico do tipo memória de desenho do relógio.	Especialização de MemoryTest	

.

As tabelas seguintes descrevem as propriedades existentes, as entidades que possuem essas propriedades (Domínio) e seus possíveis tipos de valores (Tipo de Valor).

Propried. (Objetos)	Domínio	Tipo de Valor	Descrição
belongsTo	PatientData	Patient	O paciente ao qual pertencem os dados específicos, como a duração da doença, o grupo a que pertencem.
evaluatedBy	Evaluation	Doctor	O médico responsável pela avaliação médica.
fromDisease	DiseaseStage	Disease	A doença a que a fase se reporta.
fromMedicalTest	TestComponent	MedicalTest	O teste médico a que o componente pertence. Propriedade inversa de hasTestComponent.
fromPatient	Evaluation	Patient	O paciente a que a avaliação médica se refere. Propriedade inversa de hasMedicalEpisode.
hasBLADGroup	PatientData	Score	O resultado estruturado do grupo da bateria de testes BLAD a que um paciente pertence numa avaliação.
hasCharacteristics	TestComponent	TestCharacteristics	Característica de um teste ou componente do teste.
hasComponentDuration	TestComponent	Score	O resultado estruturado da duração do componente do teste.
hasComponentResult	TestComponent	Score	O resultado estruturado da avaliação da componente.
hasComponentZScore	TestComponent	Score	O resultado estruturado da normalização ZScore do componente.
hasDiagnosis	PatientData	Diagnosis	O diagnóstico produzido após determinada avaliação médica.
hasDiseaseDuration	PatientData	Score	O resultado estruturado da duração da doença à data da avaliação médica.
hasDiseaseStage	Diagnosis	DiseaseStage	A fase da doença do paciente nesse diagnóstico.
hasMedicalEpisode	Patient	Evaluation	A avaliação médica pertencente ao paciente. Propriedade inversa de fromPatient.
hasMedicalTest	Evaluation	MedicalTest	O teste médico pertencente a determinada avaliação.
hasPatientData	Evaluation	PatientData	Os dados específicos do paciente para determinada avaliação médica.
hasScholarity	Patient	Score	O resultado estruturado do nível de escolaridade de um paciente.
hasScore	TestComponent e MedicalTest	Score	O resultado estruturado de um teste ou de um componente do teste.
hasStage	Disease	DiseaseStage	A fase da doença.
hasSubTest	MedicalTest	MedicalTest	Os subtestes de uma bateria de testes médicos.
hasTestComponent	MedicalTest	TestComponent	O componente do teste. Propriedade inversa de fromMedicalTest.
hasTestDuration	MedicalTest	Score	O resultado estruturado da duração do teste médico.
hasTestResult	MedicalTest	Score	O resultado estruturado da avaliação do teste médico.
hasTestZScore	MedicalTest	Score	O resultado estruturado após a normalização do teste, de acordo com alguns dados do paciente como seja a idade, a escolaridade, etc.
relatedTo	Diagnosis	Disease	A doença a que o diagnóstico se refere.
diagnosedBy	Disease	MedicalTest	O teste médico que possibilita diagnóstico da doença.
allowsDiagnosisOf	MedicalTest	Disease	A doença que poderá ser diagnosticada pelo teste.

Propried. (Dados)	Domínio	Tipo	Descrição
diagnosticSource	Diagnosis	String	A fonte do diagnóstico, i.e., elaborada de forma manual ou automática.
hasBirthDate	Person	dateTime	Data de nascimento de uma pessoa.
hasDate	Evaluation	dateTime	A data da avaliação médica.
hasIdPatient	Patient	positiveInteger	O identificador numérico único associado ao paciente.
hasPatientAge	PatientData	nonNegativeInteger	A idade do paciente aquando da avaliação médica.
hasQuality	Score	Literal	O atributo qualidade de um determinado resultado, como por exemplo Excelente ou Satisfaz
hasTestVersion	MedicalTest	Literal	A versão do teste médico.
hasTestVersionDate	MedicalTest	Literal	A data da versão do teste.
hasUnit	Score	Literal	O atributo unidade ou escala de um determinado resultado, como por exemplo metros ou segundos
hasValue	Score	Decimal ou integer	O atributo valor de um resultado, (exemplo 15,67).

A Figura 8 representa o modelo conceptual da NTO, onde podemos esquematicamente observar as entidades, as suas propriedades e relações mais relevantes.

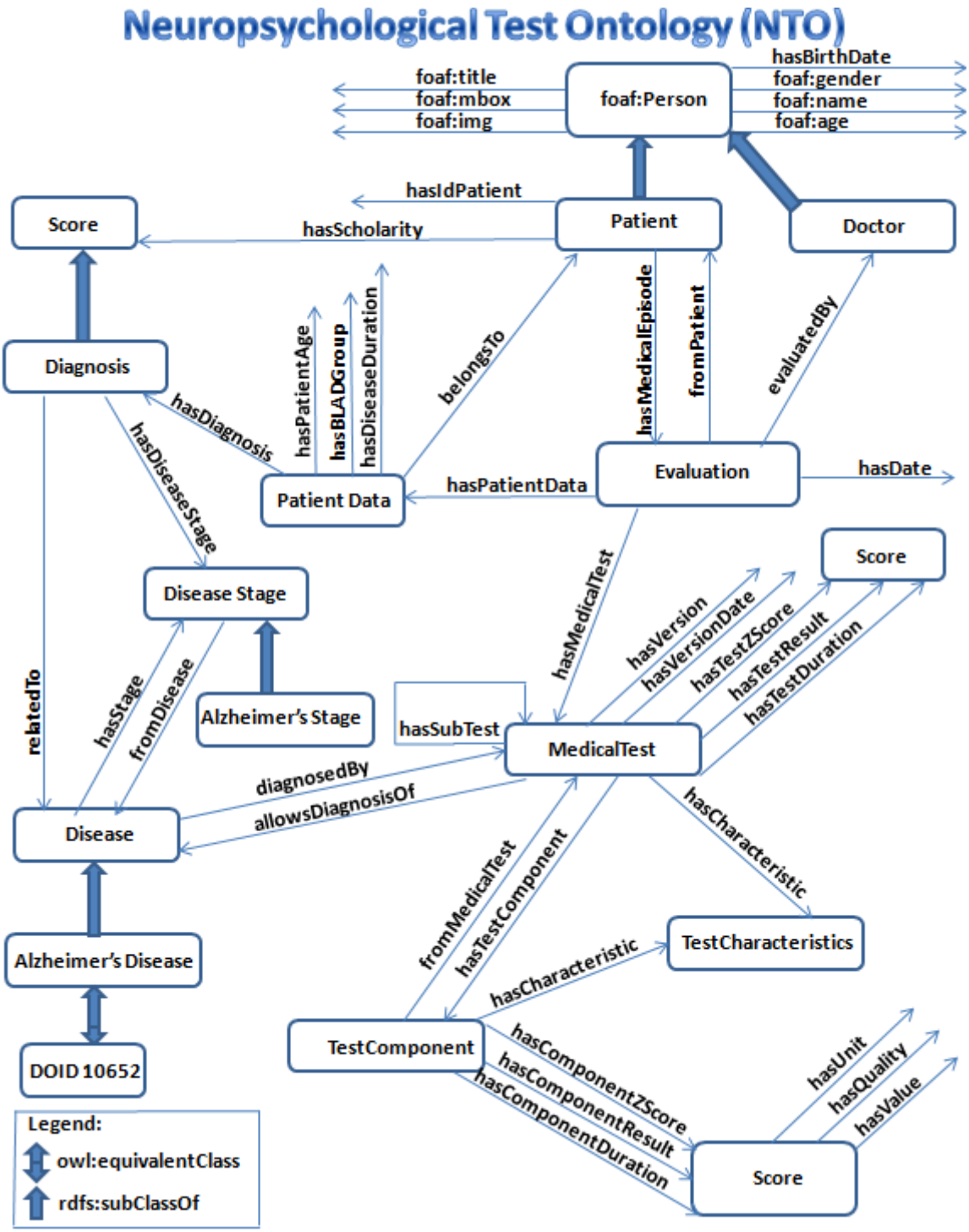

Fig. 8 – Esquema conceptual de entidades e relações da Neuropsychological Test Ontology (NTO)

Estes conceitos e relacionamentos foram transcritos para a linguagem OWL-DL utilizando a ferramenta Protégé. As figuras seguintes mostram respetivamente entidades genéricas, *object properties*, *data properties* e instâncias da NTO no Protégé (Fig.9), as entidades dos testes neuropsicológicos (Fig. 10), e dos componentes dos testes (Fig.11).

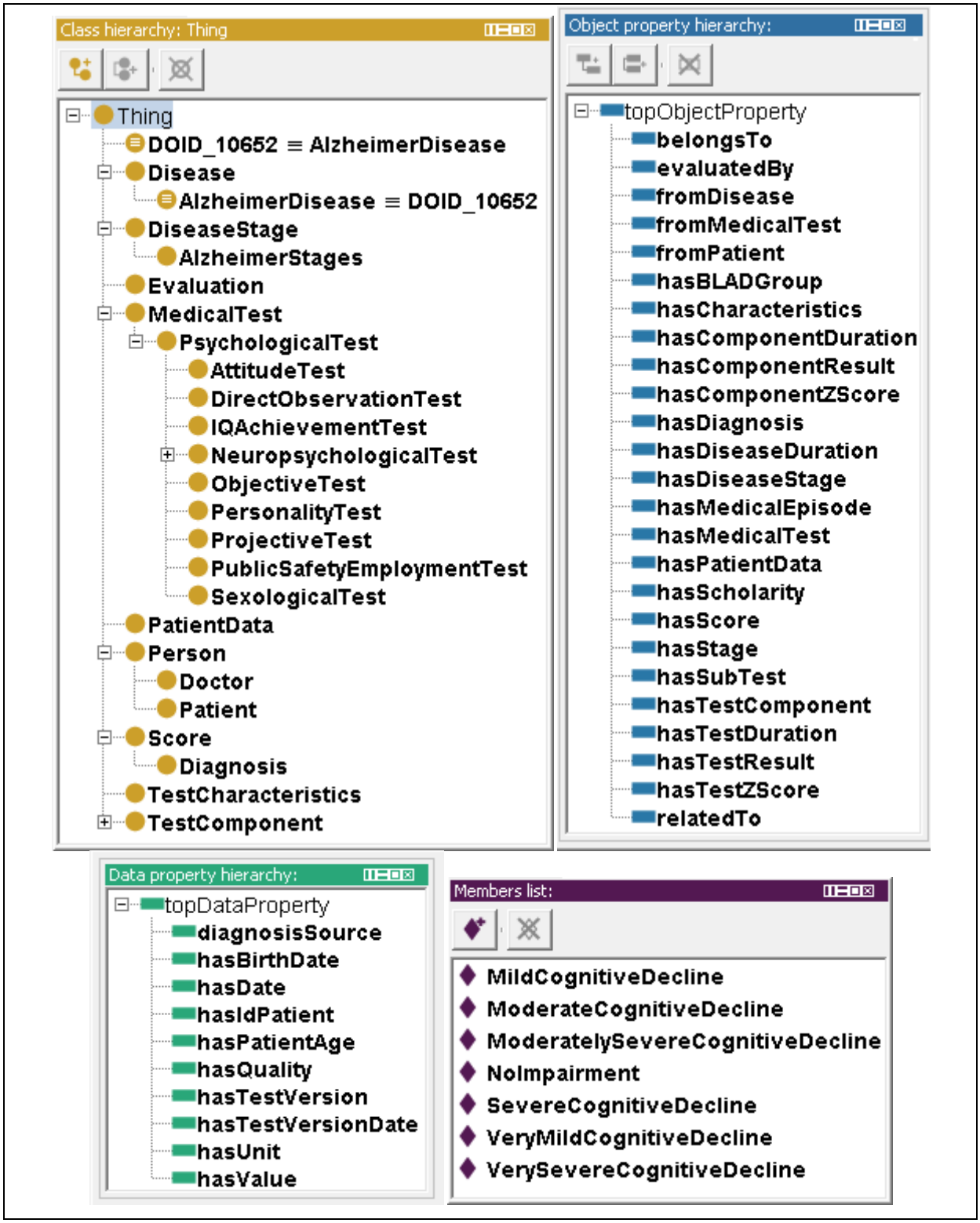

Fig. 9 – Hierarquia de classes, Object Properties, Data Properties e Instâncias de Alzheimer's Stage da NTO

+ Neuropsychological Test
  + Achievement Test
    - The Gray Oral Reading Tests—iV Edition
    - Wechsler Individual Achievement Test III
    - Wide Range Achievement Test—3
    - Woodcock-Johnson III
  + Attention Test
    - Brief Test Of Attention
    - Children's Paced Auditory Serial Addition T.
    - Color Trails Test
    - Comprehensive Trail Making Test
    - Continuous Performance Test
    - Integrated Visual Auditory Cont. Perform T.
    - Paced Auditory Serial Addition Test
    - Ruff 2 & 7 Selective Attention Test
    - Symbol Digit Modalities Test
    - Test of Everyday Attention
    - Test of Everyday Attention for Children
    - Test of Variables of Attention
    - Toulouse Pierón Test
    - Trail Making Test
    - World Recall Test
  + Excutive Functions Test
    - Behavioral Assess. Dysexecutive Syndrome
    - Cambridge Neuropsych. Test Auto Battery
    - Category Test
    - Cognitive Estimation Test
    - Compreension Orders Test
    - Delis-Kaplan Executive Function System
    - Design Fluency Test
    - Five-Point Test
    - Hayling and Brixton Tests
    - Ruff Figural Fluency Test
    - Self-Ordered Pointing Test
    - Stroop Test
    + Verbal Fluency Test
      - Phonologic Fluency Test
      - Semantic Fluency Test
    - Wisconsin Card Sorting Test
  + General Cognitive Functioning
    - Bayley Scales of Infant Development-II Edition
    - Blessed Test
    + Caculation Test
      - Mental Calculation Test
    - Cognitive Assessment System
    - Dementia Rating Scale-2
    - DigitSpan
    - Kaplan Baycrest Neurocognitive Assessment
    - Kaufman Brief Intelligence Test
    - Lisbon Battery for Demency Avaliation
    - Mini-Mental State Examination
    - National Adult Reading Test
    - NEPSY
    - Neuropsychological Assessment Battery
    - Raven's Progressive Matrices
    - Repeatable Bat. for the Ass. of Neurop. Status
    - Stanford-Binet Intelligence Scales - V Edition
    - Subjectives Memory Complaints Test
    - The Speed Capacity of Lang Processing Test
    - The Test of Nonverbal Intelligence
    + Visual-perceptual speed
      - Coding Test
      - Symbol Search Test
    - Wechsler Abbreviated Scale of Intelligence
    - Wechsler Adult Intelligence Scale—III
    - Wechsler Intelligence Scale for Children—iV
    - Wechsler Preschool Primary Scale Intel—III
    - Woodcock-Johnson III Cognitive Abilities
  + LanguageTest
    - Boston Diagnostic Aphasia Examination
    - Boston Naming Test—2
    - Dichotic Listening—Words
    - Expressive One-Word Picture Vocabulary Test
    - Expressive Vocabulary Test
    - Multilingual Aphasia Examination
    - Peabody Picture Vocabulary Test-III
    - Token Test
+ LanguageTest
  - Token Test
  + Verbal Comprehension Test
    - Comprehension Test
    - Information Test
    - Similarities Test
    - Vocabulary Test
+ Memory Test
  - Autobiographical Memory Interview
  - Benton Visual Retention Test
  - Brief Visuospatial Memory Test—Revised
  - Brown-Peterson Task
  - Buschke Selective Reminding Test
  - California Verbal Learning Test-II
  - California Verbal Learning Test Children
  - Children's Memory Scale
  - Doors and People Test
  - Hopkins Verbal Learning TestRevised
  - Logical Memory Test
  - Recognition Memory Test
  - Rey Auditory Verbal Learning Test
  - Rey-Osterrieth Complex Figure Test
  - Rivermead Behavioural Memory Test–II
  - Ruff-Light Trail Learning Test
  - Sentence Repetition Test
  - Verbal Abstration Test
  - Verbal Paired Associated
  - Visual Memory Test
  - Wechsler Memory Scale–Third Edition
  - Wide Range Assessment Memory Learning II
  + Working Memory Test
    - Arithmetic Test
    - Letter Number Sequencing Test
+ Motor Function Test
  - Finger Tapping
  - Grip Strength
  - Grooved Pegboard
  - Motor Coordenation Test
  + Motor Initiative Test
    - Grafo MotorInitiative Test
  - Purdue Pegboard Test
  - Writing Test
+ Personality And Mood Test
  - Beck Depression Inventory—II Edition
  - Behavior Rating Inventory Executive Function
  - Geriatric Depression Scale
  - Instrumental Activities of Daily Living
  - Minnesota Multiphasic Personality Inventory
  - Personality Assessment Inventory
  - Scales of Independent Behavior Revised
  - Trauma Symptom Inventory
+ Sensory Function Test
  - Finger Localization Test
  - Orientation Test
  - Right-Left Orientation Test
  - Rivermead Assessm. Somatosensor Perform.
  - Smell Identification Test
  - Tactual Performance Test
+ Visual Perception Test
  - Balloons Test
  + Cancellation Test
    - Bells Test
    - Digit Cancellation Test
    - Letter Cancellation Task
  - Clock Drawing Test
  - Draw Cube Test
  + Facial Recognition Test
    - Public Faces Test
  - Hooper Visual Organization Test
  - Judgment of Line Orientation
  - Object Identification Test
  + Perceptual Reasoning Test
    - Block Design Test
    - Figure Weights Test
    - Matriz Reasoning Test
    - Picture Completion Test
    - Visual Puzzles Test
  - Snodgrass and Vanderwart
  - Visual Object and Space Perception

Fig. 10 – Hierarquia de Classes dos Testes Neuropsicológicos da NTO

+ Test Component
  + AsTest Component
    - AS_Cut
    - AS_Time
    - AS_Total
  + Blessed Test Component
    - Blessed Daily Activities Component
    - Blessed Habits Component
    - Blessed Personality Component
    - Blessed Total Component
  + CVLT Component
    - CVLT Evaluated
    - CVLT List A Evocation 1Component
    - CVLT List A Evocation 2 Component
    - CVLT List A Evocation 3 Component
    - CVLT List A Evocation 4 Component
    - CVLT List A Evocation 5 Component
    - CVLT List A Intrusion Component
    - CVLT List A Long Interval Component
    - CVLT List A Long Interval CS Component
    - CVLT List A Long Interval Intrusions Component
    - CVLT List A Long Int.l Intrus. Sem. Help Comp.
    - CVLT List A Long Interval Persever. Component
    - CVLT List A Long Int. Persev. Sem. Help Comp
    - CVLT List A Long Interval Semantic Help Comp.
    - CVLT List A Perseverations Component
    - CVLT List A Recognition Long Interval Comp.
    - CVLT List A Small Interval Component
    - CVLT List A Small Interval CS Component
    - CVLT List A Small Interval Intrusions Component
    - CVLT List A Small Int. Intrus. Sem. Help Comp.
    - CVLT List A Small Int. Perseverations Comp.
    - CVLT List A Small Int. Pers.. Sem. Help Comp
    - CVLT List A Small Interval Semantic Help Comp.
    - CVLT List A Total 1-5 Component
    - CVLT List B CS Component
    - CVLT List B Intrusions Component
    - CVLT List B Perseverations Component
    - CVLT List B Recognition No Relation Comp.
    - CVLT List B Recognition Not Shared Component
    - CVLT List B Recognition Prototype Component
    - CVLT List B Recognition Shared Component
    - CVLT List B Total Component
  + Digit Span Component
    - Digit Span backward
    - Digit Span forward
    - Digit Span Total
  - Facial Recognition Component
  - Geriatric Depression Scalen Component
  - Line Orientation Judgment Component
  + Logical Memory Component
    - Logical Memory A
    - Logical Memory A (with Interference)
    - Logical Memory A (with Interference) Cued
    - Logical Memory A Cued
    - Logical Memory B
    - Logical Memory B (with Interference)
    - Logical Memory B (with Interference) Cued
    - Logical Memory B Cued
    - Logical Memory Total
    - Logical Memory Total (with Interference)
  - MSE Component
  + MVI_Component
    - Word Recall with Interference Cued
    - Word Recall with Interference Free
    - Word Recall with Interference Recognition
    - Word Recall with Interference total
  + Orientation Component
    - Orientation Personal
    - Orientation Spatial
    - Orientation Temporal
  + Orientation Component
    - Orientation total
    - Orientation_MSQ
  + Other Test Component
    - Calculation Component
    - Draw of a Clock Component
    - Draw of a Cube Component
    - Grafo Motor Initiative Component
    - Mental Calculation Component
    - Motor Coordenation Component
    - Motor Initiative Component
    - Naming Component
    - Objects Identification Component
    - Orders Compreension
    - Orientation Right Left
    - Phonologic Fluency
    - Reading Component
    - Repetition Component
    - Verbal Abstration Component
    - Writing Component
  + Public Faces Component
    - Public Faces Find Word
    - Public Faces Missing Word
  - Raven Progressive Matrices Component
  + Snodgrass Vanderwart Component
    - Snodgrass and Vanderwart Clinical Impression
    - Snodgrass and Vanderwart Missing Naming
    - Snodgrass and Vanderwart Stop Word
  + Stroop Component
    - Stroop Colors
    - Stroop Inference Word
    - Stroop Reading
  - Subjective Memory Complaints Component
  + TMT Component
    - Trail Making Test Evaluated
    - Trail Making Test Part A Errors
    - Trail Making Test Part A Time
    - Trail Making Test Part B Completed
    - Trail Making Test Part B Errors
    - Trail Making Test Part B Time
  + Token Component
    - Token Colors Component
    - Token Complete Component
    - Token Orders Component
  + Toulouse Pierón Component
    - Toulouse-Pierón Dispersion Index
    - Toulouse-Pierón Execution
    - Toulouse-Pierón Work Efficiency
  + Verbal Fluency Component
    - Verbal Fluency
    - Verbal Fluency Perseverations
  + VPAL_Component
    - Verbal Paired-Associate Learning Difficult
    - Verbal Paired-Associate Learning Easy
    - Verbal Paired-Associate Learning Interference Dif.
    - Verbal Paired-Associate Learning Interf. Easy
    - Verbal Paired-Associate Learning Total
  + WAIS Component
    - Wechsler Adult Intelligence Scale - Cubes
    - Wechsler Adult Intelligence Scale - Picture Compl.
    - Wechsler Adult Intelligence Scale - Similarities
    - Wechsler Adult Intelligence Scale - Symbol Search
    - Wechsler Adult Intelligence Scale - Vocabulary
  + Wechsler Visual Memory Component
    - Visual Memory Image A Component
    - Visual Memory Image B Component
    - Visual Memory Image C1 Component
    - Visual Memory Image C2 Component
    - Visual Memory Image Total Component
    - Wechsler Visual Memory Copy Component

Fig. 11 – Hierarquia dos Componentes dos Testes da NTO

3.3. Sumário

Neste capítulo descreveu-se a criação da ontologia *Neuropsychological Test Ontology* que permite mapear e descrever conceitos relativos à aplicação de testes neuropsicológicos a pacientes potencialmente com a doença de Alzheimer. Esta estrutura de representação do conhecimento é a base para o restante desenvolvimento do sistema, e, permitiu anotar semanticamente dados reais, originalmente disponíveis como ficheiros Excel. A ontologia criada representa o esforço para formular um esquema conceptual rigoroso e exaustivo dentro deste domínio de conhecimento, contendo todos os elementos relevantes e as suas relações. Desta forma, existindo uma concordância semântica relativamente aos conceitos e suas relações é possível a partilha de dados e conhecimento de modo coerente e consistente entre várias aplicações deste domínio.

4. Visão Geral do Sistema

No âmbito desta dissertação foi desenvolvido um protótipo aplicacional com vista à implementação do sistema de informação e de extração de conhecimento proposto, que comporta uma aplicação Web e um *Web Service* que permite, respetivamente, a utilizadores e sistemas computacionais, a interação com dados médicos anotados de acordo com uma ontologia específica desenvolvida para o efeito (Ver capítulo 3).

4.1. Tecnologias e ferramentas

Este subcapítulo descreve um conjunto de tecnologias e ferramentas utilizadas no desenvolvimento do sistema de informação.

4.1.1. Java EE

O protótipo foi implementado utilizando a linguagem de programação Java. Para desenvolver aplicações em Java o sistema operativo necessita de ter instalado o Java Development Kit (JDK) para a compilação do código fonte. O JDK já contém o Java Runtime Environment (JRE) necessário para correr aplicações Java. O Java Enterprise Edition (Java EE) é uma plataforma para desenvolvimento de aplicações empresariais na linguagem de programação Java, disponibilizando aos programadores um conjunto de API's (Application Programming Interfaces) que reduzem o tempo e a complexidade no desenvolvimento e aumentam o desempenho dos sistemas. O Java EE fornece bibliotecas e serviços que suportam a escalabilidade, acessibilidade, integridade e outros requisitos para as aplicações empresariais. A última versão, o Java EE 6, que já contêm nela embutida o JDK.

4.1.2. Eclipse

O Eclipse Hélios foi o ambiente de desenvolvimento integrado (IDE) escolhido. O Eclipse é uma plataforma *open source* que permite o desenvolvimento de aplicações de forma integrada. Ao possibilitar a sua extensão através de vários Plug-in desenvolvidos por programadores independentes, esta plataforma tornou-se bastante flexível e poderosa. Para o desenvolvimento deste protótipo foi instalado no eclipse o Plug-in do ZK Framework, ZK Studio[10], permitindo simplificar o desenvolvimento de aplicações ZK Web.

[10] http://books.zkoss.org/wiki/ZK%20Studio%20Essentials/Introduction

4.1.3. ZK Framework

ZK Framework é uma plataforma *open source* para desenvolvimento de aplicações Web escritas em linguagem de programação Java, utilizando a tecnologia Ajax. Permite uma fácil criação de interfaces gráficas para aplicações Web sem a necessidade de escrita de código em Javascript. O ZK utiliza uma nova linguagem, o ZK User Interface Markup Language (ZUML) para o desenho das interfaces gráficas. A ZUML baseia-se em componentes XUL e XHTML, e herda todas as funcionalidades disponíveis ao XML, separando a interface do utilizador da lógica da aplicação. Os motores de eventos do lado do cliente e do lado do servidor desempenham um papel fundamental nesta arquitetura, permitindo que os eventos despoletados pelos utilizadores da aplicação sejam encapsulados e processados no lado no servidor, sendo este processo transparente para o programador. A sincronização necessária para este efeito entre os clientes e o servidor é garantida por código Ajax, utilizando uma abordagem *server+client fusion*, que aproveita as vantagens das abordagens centradas no servidor (*server-centric)* e das abordagens centradas no cliente *(client-centric)*. Desta forma, os utilizadores obtêm uma interatividade semelhante a uma aplicação de Desktop.

Antes de desenvolver aplicações Web em Java usando a Framework ZK Ajax é necessário instalar um servidor aplicacional. O servidor escolhido foi o Apache Tomcat.

4.1.4. Apache Tomcat

O servidor Apache Tomcat[11] é um dos mais populares servidores aplicacionais, sendo também um *Web Container* ou *Servlet Container*. Foi desenvolvido pelo projeto *Jakarta* da *Apache Software Foundation*. Um servidor Web processa pedidos HTTP efetuados pelos Navegadores Web e devolve páginas Web, normalmente no formato HTML. Quando se pretende adicionar conteúdo dinâmico às aplicações Web é necessário implementar em Java uma classe especial, o Java Servlet.

O Tomcat implementa as especificações para a tecnologia Java Servlets e Java Server Pages (JSP). Os Java Servlets são componentes Web baseados em Java responsáveis pelo processamento dos pedidos HTTP e por gerar as respostas correspondentes, normalmente no formato HTML, de acordo com os requisitos da aplicação. O recipiente de Servlets do Tomcat

[11] http://tomcat.apache.org/

designa-se por Catalina e implementa também as especificações para as JSP. O motor de JSP denomina-se Jasper e é o responsável pela compilação dos ficheiros JSP em código Java sob a forma de Servlets, que poderão posteriormente ser processados pelo Catalina. O Tomcat pode ser utilizado como motor de Servlet e JSP com o seu servidor Web interno Apache, denominando-se por Apache Tomcat, ou utilizado conjuntamente com outros servidores Web. Neste protótipo, foi utilizado o Apache Tomcat, na sua versão 7, que requer uma versão Java 1.6 ou superior e implementa a especificação Servlet 3.0 e JSP 2.2.

4.1.5. Apache Jena

Jena [12] é uma Framework Java que disponibiliza um conjunto de ferramentas e de bibliotecas Java que facilitam o desenvolvimento de aplicações de Web Semântica, que utilizam ontologias em RDF, RDFS e OWL. O motor de inferência do Jena consegue fazer dedução através da aplicação de um modelo a uma determinada ontologia. Desta forma, é possível derivar novas declarações que o modelo não expressava explicitamente. O Jena possui ainda uma extensão denominada ARQ, que lhe permite consultar modelos ontológicos. Para se utilizar a Framework Jena conjuntamente com a sua extensão ARQ é apenas necessário incluir estas bibliotecas no projeto do Eclipse.

A Framework Jena inclui:

- uma API para ler, processar e escrever dados RDF em formato XML, N-triples e Turtle;
- uma API para manipular ontologias em OWL e RDFS;
- um motor de inferência baseado em regras de modo a permitir raciocinar;
- modelos de armazenamento eficiente de grandes quantidades de triplos RDF, quer em modo de memória quer de modo persistente;
- um motor de pesquisa compatível com a última especificação SPARQL;
- servidores que permitem que os dados RDF sejam publicados noutras aplicações usando vários protocolos, incluindo o SPARQL.

[12] http://jena.apache.org/

4.1.6. Openlink Virtuoso Universal Server

O *Openlink Virtuoso Universal Server* é um servidor universal multiplataforma que implementa funcionalidades de servidor Web, servidor de ficheiros e de gestor de bases de dados. Permite atuar como um motor de bases de dados virtual gerindo um vasto tipo de bases de dados como DB2, SQL Server, Oracle, Sybase, etc. Ao fornecer uma única interface para aceder às bases de dados, este acesso é efetuado transparentemente para o utilizador, podendo aceder facilmente à informação distribuída pelos vários conjuntos de servidores heterogéneos. O *Openlink Virtuoso Universal Server* inclui suporte a variados protocolos *standard* de acesso como XML, XPATH, XSLT, HTTP, HTTPS, SQL, SPARQL, WSDL, UDDI, SOAP, WebDAV, SMTP, JDBC e ODBC, etc. Corre um múltiplos sistemas operativos como Windows, Linux, MacOS X, Solaris, etc.

No contexto desta dissertação, o *Openlink Virtuoso Universal Server* foi utilizado especialmente como sistema de gestão de dados em formato RDF, pois disponibiliza armazenamento nativo em *triple store*. A solução de armazenamento em *triple store* é bastante convencional utilizando uma única tabela de quarto colunas. A coluna G para o grafo, a coluna P para o predicado, a coluna S para o sujeito e a coluna O para o objeto (Erling, et al., 2007). Estes grafos são representados como um modelo abstrato que pode ser populado com dados de ficheiros, de bases de dados, de URI's ou da combinação destes. Estes modelos podem ser questionados através do protocolo SPARQL e atualizados pelo protocolo SPARUL. O Virtuoso disponibiliza um SPARQL *endpoint* e uma interface para que a Framework Jena possa aceder e manipular os dados dos grafos através do *Virtuoso Jena RDF Data Provider*.

4.1.7. Protégé

Neste projeto foi utilizado o editor de ontologias Protégé[13], na sua versão OWL 4.2.0. É um *software* grátis, de código aberto e independente da plataforma, desenvolvido em cooperação pela Universidade de Stanford e pela Universidade de Manchester. Este *software* disponibiliza uma interface intuitiva e robusta para o desenvolvimento de ontologias, através da manipulação dos seus diversos painéis, possibilitando o desenho hierárquico de classes, a descrição de propriedades, a construção de restrições e regras, a definição de conceitos e comentários, etc. O Protégé suporta o desenvolvimento de ontologias em várias linguagens,

[13] http://protege.stanford.edu/

incluindo a OWL e permite a interoperabilidade com outros sistemas de representação de conhecimento. Uma das grandes vantagens é a sua poderosa API de Java, que permite uma fácil integração de Plug-in e serve de interface para outros programas. Através da instalação de Plug-in disponíveis, é possível alargar as suas funcionalidades básicas a outras áreas de acordo com as necessidades específicas do utilizador.

4.2. Infraestrutura

A aplicação Web foi desenvolvida sobre a *ZK Framework*, uma ferramenta de código fonte aberto que permite desenvolver aplicações cliente-servidor, com tecnologia *Ajax*, em linguagem de programação Java. A tecnologia *Ajax* efetua chamadas assíncronas de procedimento remoto ao servidor, e por isso, permite o envio e receção de dados sem interferir no comportamento e apresentação de páginas Web. Garante desta forma um maior desempenho e uma melhor usabilidade.

O sistema contém um repositório de dados do tipo *triple store* implementado pelo servidor *Virtuoso* e um servidor aplicacional Apache *Tomcat* no qual está alojado a aplicação Web desenvolvida. Os dados de testes médicos neuropsicológicos cedidos por médicos do IMM encontram-se disponibilizados em ficheiros no formato Excel. Estes dados são posteriormente anotados semanticamente de acordo com a ontologia de testes médicos previamente desenvolvida e carregados no repositório. O servidor *Virtuoso* dispõe de um motor de consultas SPARQL e um módulo de integração com a *API Jena*. Esta é uma ferramenta de código fonte aberto para desenvolvimento de aplicações em Java para ambientes de Web Semântica. O *Jena* permite trabalhar programaticamente com RDF e com ontologias em linguagem *RDF Schema* ou *OWL*, e dispõe de um conjunto de *reasoners* internos, podendo ainda incorporar *reasoners* externos como o *Pellet*.

Foi ainda implementado um *Web Service*, construído com base na tecnologia Java EE e SOAP, que disponibiliza um serviço que assenta num *software* de prospeção de dados, nomeadamente um sistema de suporte à decisão, baseado num modelo desenvolvido com auxílio do WEKA, capaz de predizer diagnósticos e prever prognósticos num intervalo temporal pré-definido. A aplicação Web consome este Serviço Web, escolhendo uma das operações disponíveis. Cada operação de diagnóstico e prognóstico a 2, 3 e 4 anos recebe como parâmetros o classificador a utilizar e um conjunto de dados de uma avaliação médica, e, devolve um intervalo de confiança.

A infraestrutura é ilustrada na Fig. 12.

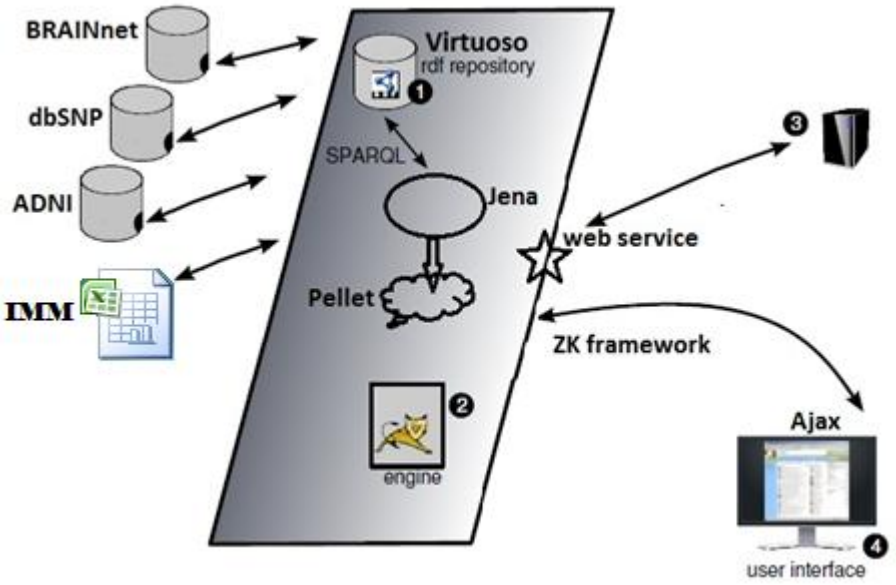

Fig. 12 – Infraestrutura aplicacional, adaptada de Kammergruber (Kammergruber, et al., 2010).

4.3. Arquitetura

O sistema implementa a arquitetura Cliente-Servidor de várias camadas de *software* (N-Tier Software Architecture), especificamente com 5 camadas constituídas pela camada de dados originais, de extração de dados, de dados semânticos, de serviços semânticos e camada aplicacional.

A camada dados originais compreende o conjunto de fontes de dados disponíveis, internas e externas, no seu formato original tais como ficheiros Excel do Instituto de Medicina Molecular com os dados clínicos dos pacientes.

A camada de extração é responsável pela qualidade da extração semântica dos dados provenientes da camada de fonte de dados, i.e. efetua a anotação semântica de acordo com a ontologia NTO através de processos que incluem a biblioteca Jena.

A camada de dados semânticos fornece o armazenamento persistente em *triple store* para os dados semânticos que contêm os metadados extraídos das fontes de dados pela camada de extração de dados. O sistema Virtuoso também armazena a ontologia nativa NTO.

A camada de serviços semânticos fornece serviços à camada superior (apresentação) acedendo e efetuando o necessário processamento sobre os repositórios de dados semânticos. Esta camada fornece a interface necessária para a lógica aplicacional aceder aos dados lógicos

da aplicação através do Jena, e, permite ainda a consulta de dados através da consola SPARQL, que é vista como um dos importantes serviços permitindo que o conhecimento esteja facilmente acessível.

A camada de apresentação fornece a interface para a interação com o utilizador permitindo-lhe a utilização da aplicação e a consequente visualização da informação. Esta interface é disponibilizada aos *browsers* através de HTML, Javascript e Ajax. A navegação é baseada em menus que disponibilizam opções de acordo com as permissões do utilizador. Permite ainda ao utilizador introduzir novos dados, editar ou exportar dados.

A arquitetura de camadas do sistema é devidamente ilustrada na figura seguinte.

Fig. 13 – Arquitetura de camadas do Sistema

4.4.Sumário

Neste capítulo descreveu-se a arquitetura e as tecnologias e ferramentas utilizadas na implementação do sistema de informação e de extração de conhecimento através de um protótipo aplicacional, que comporta uma aplicação Web e um *Web Service.*

5. Implementação do Sistema

Este capítulo visa descrever a interface gráfica e a implementação das várias funcionalidades disponíveis no sistema de informação, enquadradas nos vários papéis que os utilizadores podem assumir.

5.1.Interface Gráfica

Neste subcapítulo é apresentada e descrita a interface gráfica do sistema de informação proposto. A tela principal do sistema é constituída por 4 áreas distintas:

- Barra Superior – A barra superior contém as áreas de navegação gerais disponíveis para um dado papel do utilizador. É nesta barra que se encontra também o botão de Login permitindo efetuar a autenticação na aplicação através do protocolo OpenID. Após uma autenticação válida é possível efetuar o *Logout* ou escolher um dos possíveis papéis que o utilizador pode dispor, através de uma *combobox*.
- Barra Lateral Esquerda – Esta barra contém as áreas de navegação específicas de acordo com a área geral previamente escolhida.
- Parte Central – Esta é a área principal onde se pode visualizar a informação detalhada, de acordo com a opção escolhida na navegação específica.
- Barra Inferior – Esta barra contém algumas informações gerais como o IP do utilizador, a versão da aplicação, e o papel e tema de apresentação escolhido.

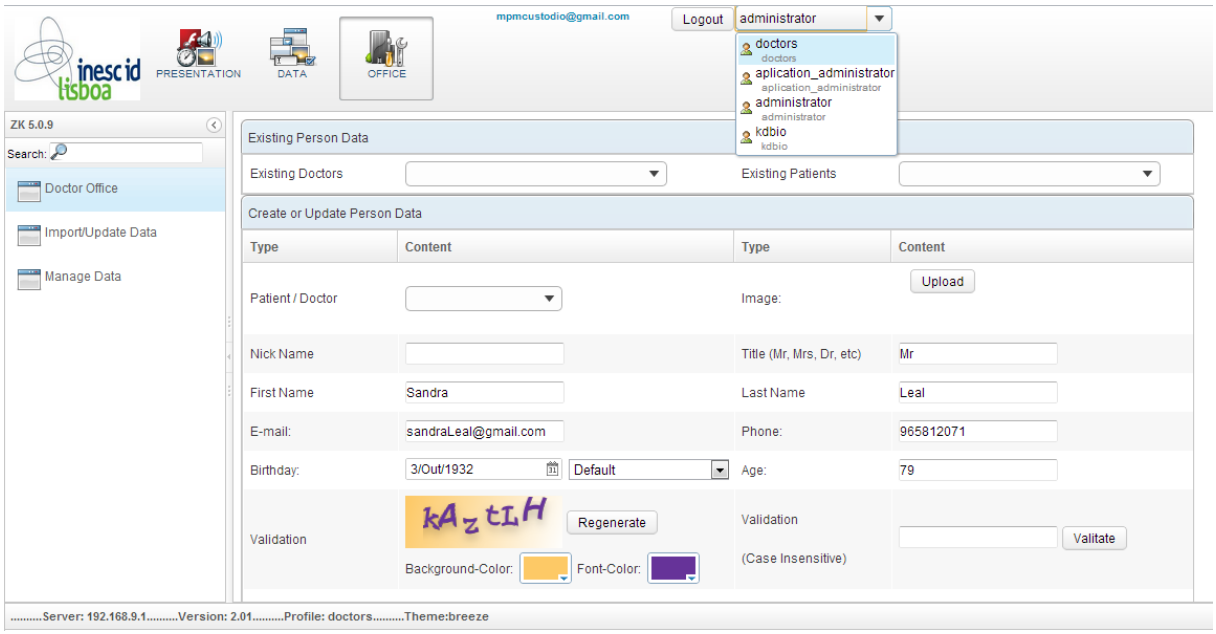

Fig. 14 – Interface gráfica do Sistema de Informação

5.2. Controlo de Acessos

Em sistemas computacionais com múltiplos utilizadores, pode existir a necessidade de limitar o acesso de utilizadores a alguns recursos protegidos. Neste sistema de informação, é permitido o acesso de utilizadores anónimos a um conjunto de funcionalidades que se assumem sem criticidade. Contudo, é necessário um eficaz controlo de acessos a outras funcionalidades. O botão de *Login*, existente na barra superior, permite despoletar este processo de controlo de acesso.

No âmbito da segurança da informação, o controlo de acessos é composto por processos de autenticação, autorização e auditoria, com vista a permitir ou negar o acesso de um sujeito a um determinado objeto ou recurso do sistema. A autenticação permite identificar quem acede ao sistema, a autorização determina o que o sujeito pode efetuar sobre determinado recurso, e, a auditoria permite verificar os recursos que o sujeito acedeu ou tentou aceder.

5.2.1. Identificação e Autenticação

Na identificação o utilizador apresenta ao sistema a sua identidade, sendo esta verificada através de uma credencial durante a autenticação. De acordo com Matt Bishop (Bishop, 2005) existem 4 categorias de credenciais de autenticação; de conhecimento, como uma senha; de posse, como um cartão de acesso; de localização, como por exemplo num determinado computador; e de biometria, como por exemplo a iris. Os sistemas computacionais utilizam uma ou mais destas categorias de credenciais.

Apesar dos progressos ao nível da identificação biométrica, na generalidade das aplicações Web a identificação é efetuada recorrendo normalmente à categoria de conhecimento, onde o utilizador tem um nome de utilizador e utiliza uma senha como credencial. O maior problema desta categoria reside no facto da possibilidade das credenciais ficarem comprometidas. Como os sistemas de autenticação necessitam de guardar a informação de autenticação para poderem efetuar a validação das identificações, é vital que estes guardem essa informação de forma segura. Quando as credenciais são transmitidas pela rede é crucial protegê-las das escutas de rede.

5.2.1.1. Single Sign-On

Os utilizadores acedem a cada vez mais aplicações, tendo por isso necessidade de utilizar várias credenciais. A dificuldade na gestão e memorização destas credenciais levou ao aparecimento do *single sign-on*. O *single sign-on* foi definido por Jan Clercq como a possibilidade de um utilizador se autenticar uma única vez perante uma autoridade de autenticação e depois poder aceder a outros recursos protegidos sem necessidade de uma nova autenticação (Clercq, 2002). O sistema de *single sign-on* apenas tem um conjunto de credenciais por utilizador, tornando a tarefa da gestão de senhas bastante mais simplificada. O custo de desenvolvimento de novas aplicações que utilizem este sistema de autenticação é reduzido pois não tem necessidade de ter os seus próprios mecanismos de autenticação.

Existem dois problemas nesta solução. O primeiro reside no facto de ao existir uma falha de segurança no sistema de *single sign-on*, esta compromete todas as aplicações que a utilizam. O segundo problema surge no nível de segurança garantido, pois as aplicações ao partilharem a mesma infraestrutura de autenticação partilham também o mesmo nível de segurança.

5.2.1.2. Protocolo OpenID

O protocolo OpenID é uma solução de *single sign-on* para a Internet e surgiu inicialmente em 2005 para permitir autenticação no *site* do Livejournal.com (Recordon, et al., 2006). Existem 2 entidades no protocolo, o *OpenID Provider* (Provider) e o *Relying Party* (RP). O primeiro é o responsável pela autenticação dos utilizadores. O segundo é um serviço que utiliza o protocolo OpenID para comunicar com um *Provider*. Esta separação permite aos utilizadores escolherem o *Provider* que desejem para efetuarem a autenticação.

Para se utilizar este protocolo, o utilizador necessita previamente de criar uma conta num dos *Provider's* existentes, como por exemplo Google, Yahoo, Verisign, AOL, etc. Quando o utilizador acede a uma aplicação que necessita de autenticação, o *Relying Party* pede-lhe para escolher um *Provider* através do seu identificador. Este identificador, que é único para um utilizador, deverá ser um URL ou um Extensive Resource Identifier (XRI) e permite ao *RP* descobrir o *Provider* associado. Quando o *RP* o localiza, redireciona o utilizador para o *Provider* através de um pedido de autenticação em HTTP. O *Provider* pode livremente escolher o método de autenticação, pois não é especificado pelo protocolo. Após a autenticação, o *Provider* redireciona o utilizador de volta para o *RP*. O sucesso da

autenticação é descrito nesta mensagem de resposta. Quando o *RP* recebe esta resposta necessita de verificar a sua assinatura. Para este efeito, pode utilizar uma chave previamente acordada com o *Provider* ou então reenviar-lhe a resposta, esperando que este a verifique e desta forma autentique o utilizador. As figuras seguintes mostram as várias fases do protocolo no protótipo. Na primeira figura é possível ver os *OpenId Provider's* que o utilizador pode escolher após ter carregado no botão *Login*. Na segunda figura podemos ver o ecrã que é apresentado ao utilizador pelo *OpenId Provider* Google, após o utilizador ter sido redirecionado para este. De notar que neste caso o utilizador já estava devidamente autorizado, graças ao protocolo de *Single Sign-on,* tendo por exemplo previamente efetuado a sua autenticação no *Gmail*. A terceira figura demonstra a autenticação com sucesso, sendo o utilizador novamente redirecionado para a aplicação original. De referir que o utilizador pode ter mais do que um papel. Nesse caso, a aplicação escolhe o papel para o utilizador de forma aleatória, permitindo que, caso este o deseje, possa mais tarde alterar através da *combobox* que surge ao lado do botão de *Logout*.

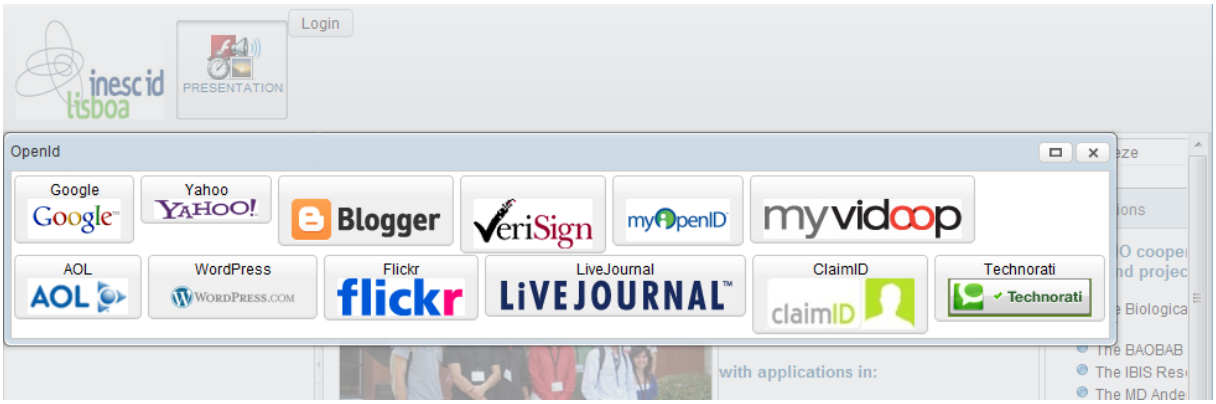

Fig. 15 – OpenID Provider's disponíveis para o utilizador

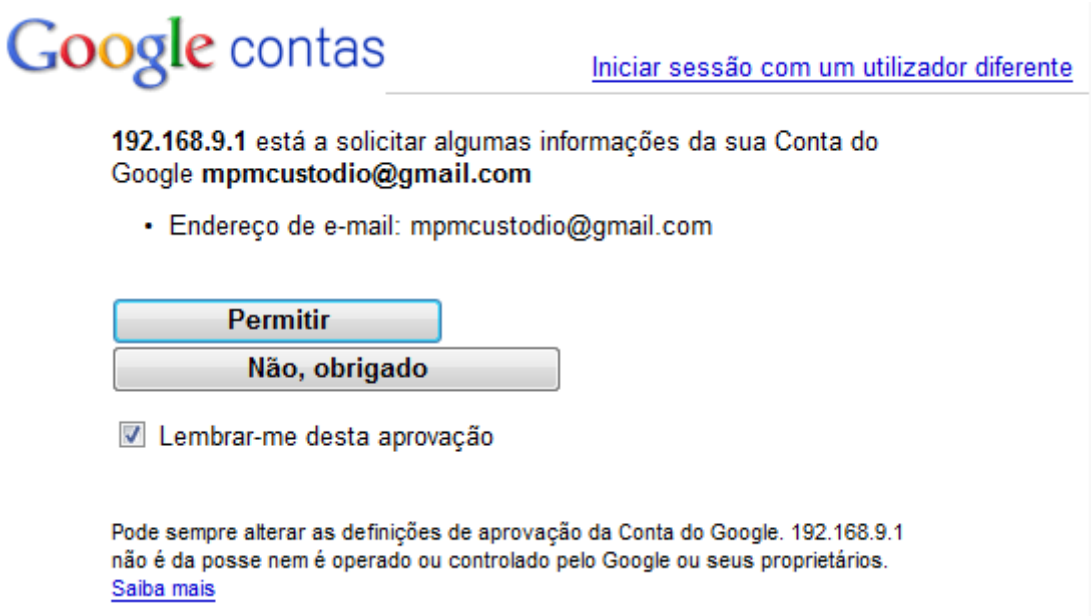

Fig. 16 – Redirecionamento do utilizador para o OpenID Provider escolhido

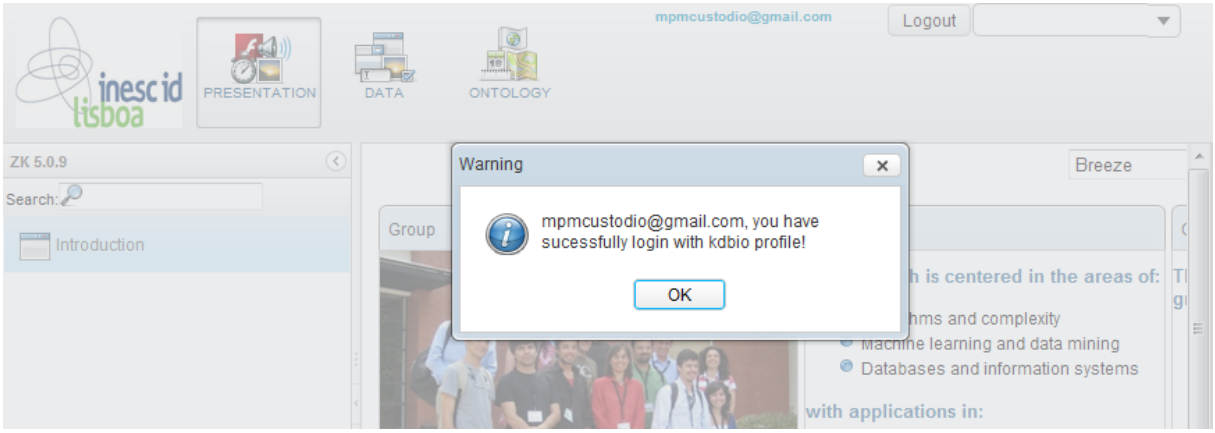

Fig. 17 – Autenticação efetuada com sucesso e respetivo redireccionamento o Relying Party

5.2.2. Autorização

Após a correta validação da identidade do utilizador, é necessário saber as permissões que este dispõe dentro do sistema. É o mecanismo de autorização que define as permissões que o utilizador tem sobre os recursos e funcionalidades do sistema. Alguns recursos poderão estar acessíveis a todos os utilizadores, enquanto que outros recursos deverão apenas estar disponíveis para um grupo restrito de utilizadores. Estes requisitos são normalmente expressos por políticas de controlo de acessos que especificam quem pode aceder a que recursos. Os mecanismos de controlo de acessos são utilizados para implementar essas políticas, de modo a assegurar que os pedidos dos utilizadores para aceder a determinados recursos, apenas são satisfeitos, se forem autorizados pelas políticas (Ferraiolo, et al., 1992).

O Controlo de Acessos Baseado em Papéis (*Role-based access control* - RBAC) surgiu como alternativa aos controlos de acessos discricionário (*Discretionary Access Control* - DAC) e obrigatório (*Mandatory Access Control* - MAC), devido ao seu potencial para reduzir a complexidade e custos da administração do controle de acessos. No DAC, o controle de acesso é determinado pelo proprietário do recurso, que decide quem tem permissão de acesso e quais os privilégios. No MAC é o administrador do sistema, e não o proprietário do recurso, que define as políticas de controlo de acessos aos recursos. A ideia básica do RBAC é a introdução do conceito de papel, que atua como ligação entre os utilizadores e as permissões. Esta abordagem de associar utilizadores e permissões a papéis permite simplificar a gestão das permissões. Um utilizador pode utilizar uma permissão desde que ative um papel e que esse papel tenha a respetiva permissão (Sandhu, et al., 1996).

Este protótipo utiliza o controlo de acessos baseado em papéis permitindo ao administrador da aplicação personalizá-la de modo a adaptar-se às necessidades de grupos de utilizadores distintos. Neste modelo de acesso os utilizadores da aplicação, identificados pelo seu *email*, podem associar-se a cinco grupos de utilizadores distintos, também denominados por papéis ou perfis de acesso: Administrador da Aplicação, Médico, Secretária, KDBio e Anónimo. Cada um destes papéis tem associado um conjunto de permissões (leitura, inserção, atualização e remoção) sobre serviços disponibilizados pela aplicação. Este modelo de acesso permite uma especificação detalhada sobre o que cada grupo de utilizadores pode fazer sobre um dado serviço. Assim, é possível especificar por exemplo que apenas os utilizadores com o perfil de Médico podem inserir e remover registos clínicos, mas que os utilizadores com o papel de Secretária apenas podem atualizar esses mesmos registos. As opções de navegação na aplicação, através do menu geral e do menu específico, são também elas sustentadas como serviços, possibilitando desta forma uma navegação personalizada por papéis.

Com vista à possibilidade de reutilização deste modelo noutras aplicações, foi criado uma nova dimensão, a aplicação, que permite associar os papéis e os serviços a uma determinada aplicação específica. Este modelo de acesso é mapeado através de declarações RDF formando um grafo RDF denominado PermissionGraph, que reside no Virtuoso.

Seguidamente são listadas como exemplo algumas consultas SPARQL efetuadas sobre este grafo. A primeira consulta permite listar todos os serviços pertencentes à aplicação app_1. A segunda consulta permite extrair todas as permissões existentes para a aplicação app_1. Estas permissões são carregadas inicialmente do grafo RDF e guardadas num objeto persistente em memória, que permitirá a sua rápida manipulação.

```
SELECT * FROM <http://kdbio.pt/PermissionsGraph>
WHERE
{
?s <http://kdbio.pt/service> ?o  .
?s <http://kdbio.pt/serviceApplication> <http://kdbio.pt/application/app_1>
}
```

```
SELECT * FROM <http://kdbio.pt/PermissionsGraph>
WHERE
{
<http://kdbio.pt/application/app_1> <http://kdbio.pt/application> ?nameApp  .
?idProfile <http://kdbio.pt/profileApplication> <http://kdbio.pt/application/app_1> .
?idProfile <http://kdbio.pt/profile> ?nameProfile   .
?idPermission <http://kdbio.pt/permissionProfile> ?idProfile      .
?idPermission <http://kdbio.pt/permissionService> ?idService      .
?idPermission <http://kdbio.pt/permissionFilter> ?idFilter        .
?idService <http://kdbio.pt/service> ?nameService   .
?idService <http://kdbio.pt/serviceApplication> <http://kdbio.pt/application/app_1>.
?idFilter <http://kdbio.pt/filter> ?nameFilter      .
?idUser <http://kdbio.pt/usersProfile> ?idProfile   .
?idUser <http://kdbio.pt/users> ?nameUser
}
ORDER BY ASC(?nameProfile) ASC(?nameService)
```

De seguida é apresentado parte do código em Java que permite a criação deste modelo de acesso. O ficheiro *application.properties* permite configurar vários parâmetros da aplicação, sendo posteriormente acedidos por Library.getProperty(property_name). As propriedades PERMISSION_URL e PERMISSION_GRAPH foram configurados respetivamente com as Strings "http://kdbio.pt/" e "PermissionsGraph". O construtor da classe VirtuosoConfig recebe como parâmetro o parâmetro "http://kdbio.pt/PermissionsGraph" correspondendo ao nome do Grafo RDF. Esta classe permite o acesso a este Grafo no Virtuoso mediante o fornecimento do nome de utilizador, da respetiva senha e do URL da instância Virtuoso. Todos estes dados são também parametrizáveis no ficheiro de configuração.

```
//CONFIGURAÇÕES E ACESSO AO GRAFO VIRTUAL
String httpProj = Library.getProperty("PERMISSION_URL");
String httpPermissions = Library.getProperty("PERMISSION_GRAPH");
VirtuosoConfig virtConf = new VirtuosoConfig(httpProj + httpPermissions);
VirtGraph graph = virtConf.getGraph();

//CRIAÇÃO DO NÓ GENERICO PARA A APLICAÇÂO
Node applicationGeneric = Node.createURI(httpProj + "application");

//CRIAÇÃO DA INSTANCIA APLICAÇÃO app_1 DENOMINADA POR neuroclinomics
Node application1 = Node.createURI(httpProj + "application/app_1");
Node applicationName1 = Node.createURI("neuroclinomics");
graph.add(new Triple(application1, applicationGeneric, applicationName1));

//CRIAÇÃO DO NÓ GENERICO PARA OS PAPEIS
Node profileApplicationGeneric = Node.createURI(httpProj + "profileApplication");
Node profileGeneric = Node.createURI(httpProj + "profile");

//CRIAÇÃO DA INSTANCIA PAPEL pf_1 DENOMINADO administrator
Node profile1 = Node.createURI(httpProj + "profile/pf_1");
Node profileName1 = Node.createURI("administrator");
graph.add(new Triple(profile1, profileGeneric, profileName1));
graph.add(new Triple(profile1, profileApplicationGeneric, application1));

//CRIAÇÃO DO NÓ GENERICO PARA OS SERVICOS
Node serviceGeneric = Node.createURI(httpProj + "service");
Node serviceApplicationGeneric = Node.createURI(httpProj + "serviceApplication");

//CRIAÇÃO DA INSTANCIA SERVIÇO src_1 DENOMINADO OFFICE
Node service1 = Node.createURI(httpProj + "service/srv_1");
Node serviceName1 = Node.createURI("OFFICE");
graph.add(new Triple(service1, serviceGeneric, serviceName1));
graph.add(new Triple(service1, serviceApplicationGeneric, application1));

//CRIAÇÃO DO NÓ GENERICO PARA OS FILTROS
Node filterGeneric = Node.createURI(httpProj + "filter");

//CRIAÇÃO DA INSTANCIA FILTRO flt_1 DENOMINADO select
Node filter1 = Node.createURI(httpProj + "filter/flt_1");
Node filterName1 = Node.createURI("select");
graph.add(new Triple(filter1, filterGeneric, filterName1));

//CRIAÇÃO DA INSTANCIA FILTRO flt_2 DENOMINADO insert
Node filter2 = Node.createURI(httpProj + "filter/flt_2");
Node filterName2 = Node.createURI("insert");
graph.add(new Triple(filter2, filterGeneric, filterName2));

//CRIAÇÃO DO NÓ GENERICO PARA AS PERMISSÕES
Node permission1 = Node.createURI(httpProj + "permission/prms_1");
graph.add(new Triple(permission1, permissionProfileGeneric, profile1));
graph.add(new Triple(permission1, permissionServiceGeneric, service1));
graph.add(new Triple(permission1, permissionFilterGeneric, filter1));
graph.add(new Triple(permission1, permissionFilterGeneric, filter2));

//CRIAÇÃO DO NÓ GENERICO PARA OS UTLIZADORES
Node uUsersGeneric = Node.createURI(httpProj + "users");

Node uUser2 = Node.createURI(httpProj + "users/usr_2");
Node uMail2 = Node.createLiteral("mpmcustodio@gmail.com");
graph.add(new Triple(uUser2, uUsersGeneric, uMail2));

//CRIAÇÃO DO NÓ GENERICO PARA A ASSOCIAÇÃO UTILIZADOR PAPEL
Node uUsersProfileGeneric = Node.createURI(httpProj + "usersProfile");

//CRIAÇÃO DA INSTANCIA ASSOCIAÇÃO UTILIZADOR uUserX a PAPEL profileY
graph.add(new Triple(uUser1, uUsersProfileGeneric, profile2));
graph.add(new Triple(uUser2, uUsersProfileGeneric, profile1));
graph.add(new Triple(uUser2, uUsersProfileGeneric, profile2));
```

O código Java seguinte demonstra como pode ser verificada a permissão que um dado utilizador possui para inserir valores num determinado serviço, aqui designado por *serviceName*. De referir que quer o utilizador quer as permissões foram previamente guardadas no sistema e é possível aceder a elas através da variável de sessão.

```
TicketPermissaoOperacao tpo_Servico =
        ServicoPermissoes.servico_obtemPermissao(session, serviceName);
if(tpo_Servico!=null)
{
        if(tpo_Servico.isPermissaoInsert())
        {
                //Put your code here
        }
}
```

Fig. 18 – Esquema de permissões para o papel de doctor no serviço de office

Os quadros que se seguem listam a relação entre os papéis dos utilizadores e as opções de navegação geral e especificas disponibilizadas de acordo com as permissões definidas. Assim, no primeiro quadro visualizamos os papéis e respetivas opções de navegação geral associadas. No segundo quadro podemos observar as funcionalidades específicas disponíveis para cada opção de navegação geral.

PAPÉIS / NAV. GERAL	PRESENTATION	DATA	OFFICE	ONTOLOGY	PERMISSIONS
ANONYMOUS	**X**	**X**			
DOCTORS	**X**	**X**	**X**		
KDBIO	**X**	**X**		**X**	
ADMINISTRATOR	**X**				**X**

PRESENTATION	DATA	OFFICE	ONTOLOGY	PERMISSIONS
INTRODUTION	**MANAGE DATA** **PROGNOSTIC**	**DOCTOR OFFICE** **INPUT/UPDATE DATA**	**ONTOLOGY BROWSER** **MEDICAL TESTS** **SPARQL CONSOLE**	**SERVICES** **PROFILE** **FILTERS** **USERS** **USERS VS PROFILE** **PERMISSIONS**

5.3. Funcionalidades

Seguidamente serão descritas as várias funcionalidades específicas disponibilizadas aos vários utilizadores de acordo com o seu papel.

5.3.1. Introdução

Esta funcionalidade permite visualizar a página inicial da aplicação, onde é descrita informação geral sobre o grupo KDBIO.

5.3.2. Anotação de Dados

Esta funcionalidade permite anotar semanticamente dados relativos à realização de testes médicos neuropsicológicos, efetuados a pacientes numa determinada avaliação médica, e inseri-los num grafo RDF do sistema Virtuoso, denominado DATA_GRAPH.

Os médicos do IMM, Dr. Alexandre Mendonça e Dr.ª Manuela Guerreiro, parceiros no projeto *Neuroclinomics*, efetuaram desde 1989 um conjunto de avaliações médicas a pacientes potencialmente com a Doença de Alzheimer. Estes pacientes foram submetidos a uma bateria de testes denominada por *Bateria de Lisboa para Avaliação da Demência* (BLAD) e um conjunto de outros testes neuropsicológicos individuais como sejam *Toulouse-Pierón, Trail Making Test, Stroop Test, Wechsler Adult Intelligence Scale e California Verbal Learning Test*. A BLAD é composta por testes como *Letter Cancellation Task, Digit Span, Clock Draw, Interpretation of Proverbs, Raven Progressive Matrices, Naming Public Faces, Verbal Paired-Associate Learning, Logical Memory, Word Recall with Interference, Snodgrass and Vanderwart,* entre outros. De salientar, que existem alguns destes testes, que devido à sua complexidade, são decompostos em múltiplos resultados específicos aos quais designamos por componentes dos testes. Por exemplo, o teste *Verbal Paired-Associate Learning* é decomposto nos seguintes componentes: *Easy, Difficult, Total, Easy with Interference e Difficult with Interference*.

Os dados disponibilizados pela equipa de médicos do IMM encontram-se em folhas Excel, com um esquema similar ao da figura abaixo disponibilizada. Estes ficheiros contêm na primeira linha os nomes das colunas e nas restantes linhas informação relativa a uma determinada avaliação médica efetuada a um dado paciente. Cada avaliação médica contém assim, a informação geral do paciente e os resultados dos testes realizados nessa avaliação médica, sendo repartidos pelas colunas existentes no ficheiro. As primeiras colunas contêm o

identificador numérico do paciente, a data da avaliação médica e os dados pessoais do paciente nessa avaliação como sejam a idade, o diagnóstico efetuado pelo psicologista, o código do diagnóstico, a duração da doença, o grau de escolaridade, o sexo, a data de nascimento do paciente e o grupo de controlo a que pertence para a *BLAD*. As restantes colunas contêm o resultado de vários testes neuropsicológicos, ou, caso existam, dos seus componentes. Por exemplo, o teste *Raven Progressive Matrices* contém apenas uma coluna referente ao seu resultado, enquanto o teste *Digit Span*, contém as colunas referentes aos valores dos componentes *forward, backward* e *total*.

Na figura seguinte podemos verificar que o paciente identificado com o nº 2 tem 5 linhas no ficheiro, correspondendo a 5 avaliações médicas distintas. O paciente nasceu a 26-01-1938 e tem o código 1 no género, correspondendo ao sexo masculino. Na avaliação datada de 04-01-2007, correspondente à linha 7, o paciente tinha 68 anos de idade, foi-lhe atribuído o diagnóstico de MCI cujo código é 1. À data de avaliação o paciente tinha 16 anos de escolaridade, pertencia ao grupo 8 da BLAD e tinha-lhe sido diagnosticado a doença de Alzheimer há 5 anos. Nessa avaliação médica o paciente foi submetido a um conjunto de testes neuropsicológicos, dos quais apenas se evidenciam 2 na figura. No teste de *Letter Cancellation Task* (AS), o paciente obteve os valores 16, 34 segundos e 4,7 para suas componentes *Cut*, *Time* e *Total*, representados nas colunas K a M. No *Digit Span Test*, o paciente obteve os valores 5, 4 e 11 para as suas componentes de *Forward*, *Backward* e *Total*.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Patient_Id	Age	DiagNPS	Diagnosis_code	Disease_duration	Date	School	Group	Gender	Birth	As_cut	As_time	As_tot	DS_forw	DS_back	DS_tot
4	2	64	MCI	1,0	1,00	28-11-2002	16	7	1	26-01-1938				5	4	10
5	2	65	MCI	1,0	1,50	05-06-2003	16	8	1	26-01-1938				5	5	12
6	2	66	MCI	1,0	3,00	22-12-2004	16	8	1	26-01-1938	16	37	4,3	5	4	11
7	2	68	MCI	1,0	5,00	04-01-2007	16	8	1	26-01-1938	16	34	4,7	5	4	11
8	2	71	MCI amnésico	1,0	7,50	15-04-2009	16	8	1	26-01-1938	14	40	3,5	5	3	9
9	3	68	MCI md	1,0	3,00	16-04-2001	4	5	1	13-09-1932	15	55	2,7	5	2	7
10	4	83	MCI amnésico	1,0	4,00	09-01-2008	16	8	1	09-10-1924	15	34	4,4	5	4	9
11	5	69	MCI	1,0	5,00	06-12-2006	4	5	1	17-09-1937	16	39	4,1	5	3	8

Fig. 19 – Esquema do ficheiro de dados médicos

A anotação semântica da informação relativa à avaliação médica de 04-01-2007 foi efetuada resumidamente da seguinte forma:

1) Paciente
   a) Criação do recurso paciente, caso ainda não exista, e associá-lo à classe *Patient*.

b) Associação do paciente ao seu identificador através da propriedade *hasIdPatient*.

c) Atribuição da data de nascimento ao paciente através da propriedade http://xmlns.com/foaf/0.1/birthday.

d) Atribuição do sexo ao paciente através da propriedade http://xmlns.com/foaf/0.1/gender.

e) Atribuição da escolaridade ao paciente. É criado um recurso da classe Score. Este irá ter na propriedade *hasValue* o valor inteiro 16 e na propriedade *hasUnit* o valor "years". Este *Score* é associado ao paciente através da propriedade *hasScholarity*.

Sujeito	Predicado	Objeto
myData:Patient/p_2	rdf:type	myOnt:Patient
myData:Patient/p_2	myOnt:hasIdPatient	2
myData:Patient/p_2	foaf:birthday	1938-01-26
myData:Patient/p_2	foaf:gender	Male
myData:Patient/p_2	hasScholarity	Score/scorePatient_2
myData:Score/scorePatient_2	rdf:type	myOnt:Score
myData:Score/scorePatient_2	myOnt:hasValue	16"^^<http://www.w3.org/2001/XMLSchema#int>
myData:Score/scorePatient_2	myOnt:hasUnit	Years

2) Avaliação médica

a) Criação do recurso avaliação médica e associá-lo à classe *Evaluation*.

b) Associar o recurso ao paciente através da propriedade *fromPatient*.

c) Atribuição da data da avaliação médica ao recurso através da propriedade *hasDate*.

Sujeito	Predicado	Objeto
myData:Evaluation/mEp_6	rdf:type	myOnt:Evaluation
myData:Evaluation/mEp_6	myOnt:fromPatient	myData:Patient/p_2
myData:Evaluation/mEp_6	myOnt:hasDate	2007-01-04

3) Dados do Paciente na avaliação médica

a) Criação do recurso dados do paciente e associá-lo à classe *PatientData*.

b) Associar a avaliação ao recurso através da propriedade *hasPatientData*.

c) Associar o recurso ao paciente através da propriedade *belongsTo*.

d) Atribuição da idade do paciente ao recurso através da propriedade *hasPatientAge*.

e) Atribuição do grupo da BLAD ao recurso através da propriedade *hasBLADGroup*.

f) Atribuição do diagnóstico ao recurso. É criado um recurso da classe *Diagnosis*, uma subclasse de *Score*. Este irá ter na propriedade *hasValue* o valor inteiro 1, na propriedade *hasQuality* o valor "MCI" e na propriedade *hasSource* o valor "Manual". Este *Score* é associado ao recurso através da propriedade *hasDiagnosis*.

Sujeito	Predicado	Objeto
myData:PatientData/pData_3	rdf:type	myOnt:PatientData
myData:PatientData/pData_3	myOnt:belongsTo	myData:Patient/p_2
myData:Evaluation/mEp_6	myOnt:hasPatientData	myData:PatientData/pData_3
myData:PatientData/pData_3	myOnt:hasPatientAge	"68"^^<http://www.w3.org/2001/XMLSchema#int>
myData:PatientData/pData_3	myOnt:hasBLADGroup	"8"^^<http://www.w3.org/2001/XMLSchema#int>
myData:PatientData/pData_3	myOnt:hasDiagnosis	myData: Score/score_46
myScore:score_46	rdf:type	myOnt:Diagnosis
myScore:score_46	myOnt:hasSource	Manual
myScore:score_46	myOnt:hasQuality	"MCI"^^<http://www.w3.org/2001/XMLSchema#string>
myScore:score_46	myOnt:hasValue	1

4) Testes Médicos

a) Criação do recurso Teste Médico e associá-lo a uma subclasse da classe *MedicalTest*, neste caso específico à classe *LetterCancelationTask*.

b) Associar a avaliação médica ao recurso através da propriedade *hasMedicalTest*.

Sujeito	Predicado	Objeto
myData:MedicalTest/mTest_92	rdf:type	myOnt:LetterCancellationTask
myData:Evaluation/mEp_6	myOnt:hasMedicalTest	myData:MedicalTest/mTest_92

5) Componentes de Teste

a) Criação do recurso Componente de Teste Médico e associá-lo a uma subclasse da classe *TestComponent*, neste caso *AS_Time*.

b) Associar o teste médico ao recurso através da propriedade *hasTestComponent*.

c) Atribuição do valor do componente ao recurso. Para o efeito é criado um recurso da classe *Score*, que tem na propriedade *hasValue* o valor 34 e na propriedade *hasUnit* o valor "seconds". Este *Score* é associado ao recurso através da propriedade *hasComponentResult*.

Sujeito	Predicado	Objeto
myData:TestComponent/tComp_165	rdf:type	myOnt:AS Time
myData:MedicalTest/mTest 92	myOnt:hasTestComponent	myData:TestComponent/tComp 164
myData:TestComponent/tComp_165	myOnt: hasComponentResult	myData: Score/score 228
myData: Score/score 228	rdf:type	myOnt:Score
myData: Score/score_228	myOnt:hasValue	34
myData: Score/score_228	myOnt:hasUnit	seconds

Embora a estrutura do ficheiro Excel se tenha mantido imutável ao longo deste trabalho, foi decidido, por questões de flexibilização do sistema, permitir a troca de colunas, a alteração dos nomes das colunas, e a adição de mais colunas para alojar novos testes. Para permitir esta flexibilidade, foi criado um ficheiro de configuração do mapeamento da ontologia, denominado OntologyMap.xls. Este ficheiro permite efetuar as anotações semânticas para cada coluna do ficheiro de dados de acordo com a ontologia NTO. A primeira linha contém os cabeçalhos das colunas do ficheiro de dados. As restantes linhas contêm os vários identificadores de classes e propriedades da ontologia, como sejam o nome do componente do teste, o nome do teste, o nome da propriedade Score, o valor da propriedade Score, o nome do cabeçalho no ficheiro de dados para descodificar o valor *ZScore*, e, o tipo de dados caso seja diferente de decimal. A título de exemplificação, a figura seguinte contém 4 casos típicos para efetuar o mapeamento dos dados de acordo com a ontologia.

No caso mais simples, respeitante à coluna B, o teste neuropsicológico *Mental Calculation Test* não têm componentes e toda a informação respeitante ao teste encontra-se numa única coluna. Assim, se o ficheiro de dados contiver um valor que pertença à coluna cujo cabeçalho é *M_Calc*, então esse valor será anotado semanticamente como pertencente ao teste cujo identificador na ontologia é *MentalCalculationTest*.

O segundo caso, respeitante à coluna C, o teste *Object Identification Test* não tem componentes, mas contém um valor *ZScore*, algures numa outra coluna do ficheiro. Essa coluna deverá ter o cabeçalho com o nome *Identification_Z*, para desta forma ser possível associar o valor do *ZScore* ao valor do teste respetivo. Assim, se o ficheiro de dados contiver um valor que pertença à coluna cujo cabeçalho é *Ident*, então esse valor será anotado semanticamente como pertencente ao teste cujo identificador na ontologia é *ObjectIdentificationTest*.

O terceiro caso, respeitante às colunas D, E e F, o teste *Letter Cancellation Task* têm 3 componentes, respetivamente *AS_Cut*, *AS_Time* e *AS_Total*. Para o caso da coluna D, se o

ficheiro de dados contiver um valor que pertença à coluna cujo cabeçalho é *As_cut*, então esse valor será anotado semanticamente como pertencente à propriedade *hasValue* do *Score*. Este *Score* pertencerá ao teste cujo identificador na ontologia é *LetterCancellationTask*. O componente *AS_Time*, representado na coluna E, mede a duração do teste em segundos. Neste caso é necessário para além de se guardar o valor numérico do tempo, guardar também as unidades de medida, neste caso segundo. Assim, o Score deste componente terá na propriedade *hasValue* o valor numérico e na propriedade hasUnit a descrição seconds. O componente *AS_Total*, representado na coluna F, contém também um valor *ZScore*. Este valor será o correspondente ao valor da coluna que contém o cabeçalho com o nome *CancellationTask_Z*.

O último caso, respeitante às colunas G e H, o teste *Wechsler Adult Intelligence Scale* (WAIS-III) contém 5 componentes, sendo apenas aqui representadas como exemplo 2 delas, a *Cubes* e a *Similarities*. Para a componente Cubes da coluna G o procedimento é em tudo identifico ao efetuado para a coluna F. A componente Similarities, representada na coluna H, tem o valor String na linha 7, indicando que os seus possíveis valores têm um tipo de dados diferente de decimal, neste caso String. Com esta solução, é possível anotar semanticamente valores com vários tipos de dados, desde que seja definido o seu tipo.

	A	B	C	D	E	F	G	H
1	*File Colunm Name / Ontology*	M_Calc	Ident	As_cut	As_time	As_tot	WAIS_cubos	WAIS_semelhanças
2	*Component Test Name*			AS_Cut	AS_Time	AS_Total	WAIS_Cubes_ Component	WAIS_Similarities_ Component
3	*Test Name*	MentalCalculation Test	ObjectIdentification Test	LetterCancellation Task	LetterCancellation Task	LetterCancellation Task	WAIS-III	WAIS-III
4	*Score Property Name*				hasUnit			
5	*Score Property Value (seconds, etc)*				seconds			
6	*Zscore Header Name*		Identification_Z			CancellationTask_Z	CubesWAIS_Z	
7	*Data Type of Value (if different from decimal)*							String

Fig. 20 – Ficheiro de configuração OntologyMap.xls

5.3.3. Gestão de Pacientes

Esta funcionalidade permite efetuar a gestão da informação geral relativa aos pacientes e aos médicos. Neste ecrã, dados como o nome, data de nascimento, fotografia, alcunha, título, correio eletrónico e telemóvel podem ser inseridos ou atualizados no sistema. Os pacientes são também associados ao seu respetivo médico. Antes de inserir ou atualizar esta informação

é necessário validar um *captcha*. Os *captcha* são testes de desafio-resposta, onde numa imagem aparecem letras destorcidas que o utilizador deverá descodificar. São usados especialmente em aplicações Web, para garantir que a resposta é gerada por uma pessoa e não por uma máquina que tenta aceder indevidamente. Este requisito é assegurado pela reconhecida dificuldade de resolução destes testes por sistemas computacionais.

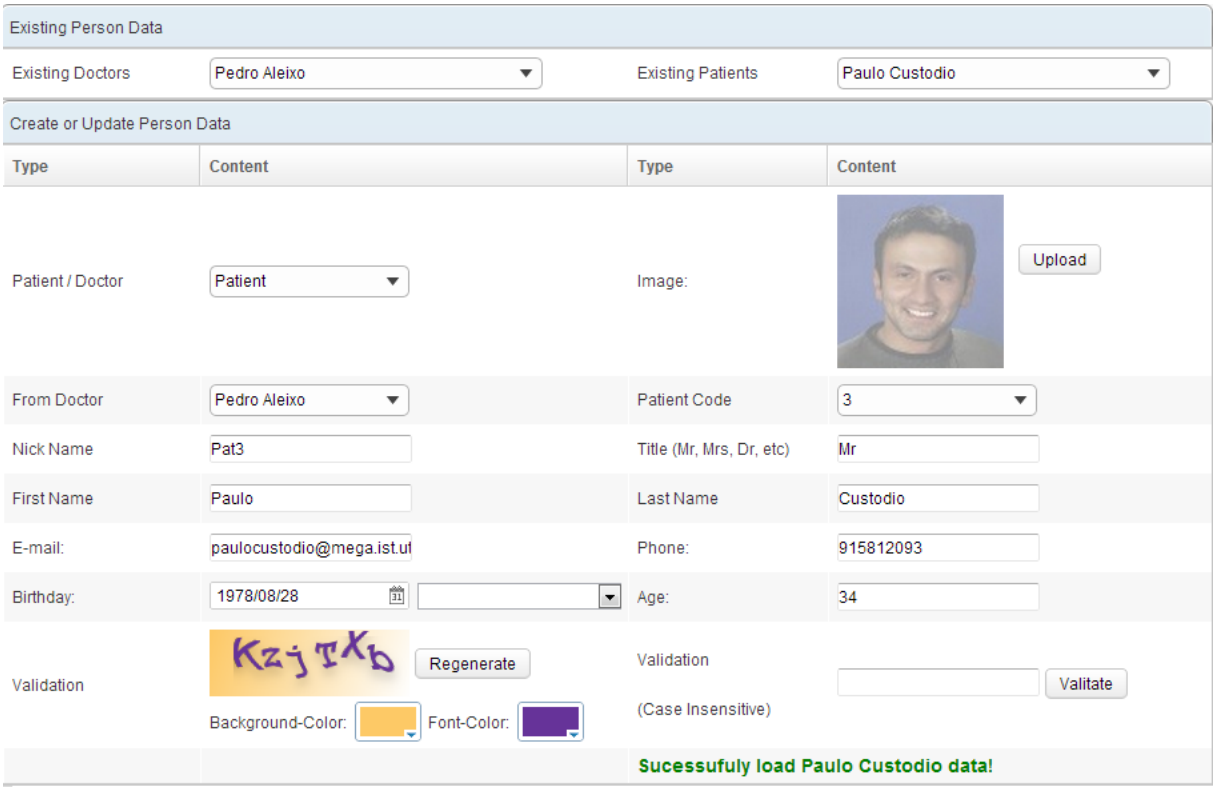

Fig. 21 – Funcionalidade Doctor Office

A confidencialidade dos dados pessoais e dos testes médicos associados a pacientes reais é um assunto de grande criticidade. Esta problemática foi solucionada efetuando uma separação ao nível de grafos entre os dados pessoais e os dados dos testes médicos. Os dados dos testes médicos são guardados no grafo denominado DATA_GRAPH, identificando a que pertencem apenas com um número correspondente ao identificador do paciente. Utilizando apenas este grafo não é possível efetuar-se a descodificação do paciente real. Já os dados pessoais dos pacientes são guardados no grafo privado FOAF_GRAPH, não estando expostos na consola SPARQL. Este grafo contém também o identificador do paciente permitindo desta forma a associação entre estes dados e os dados do grafo DATA_GRAPH.

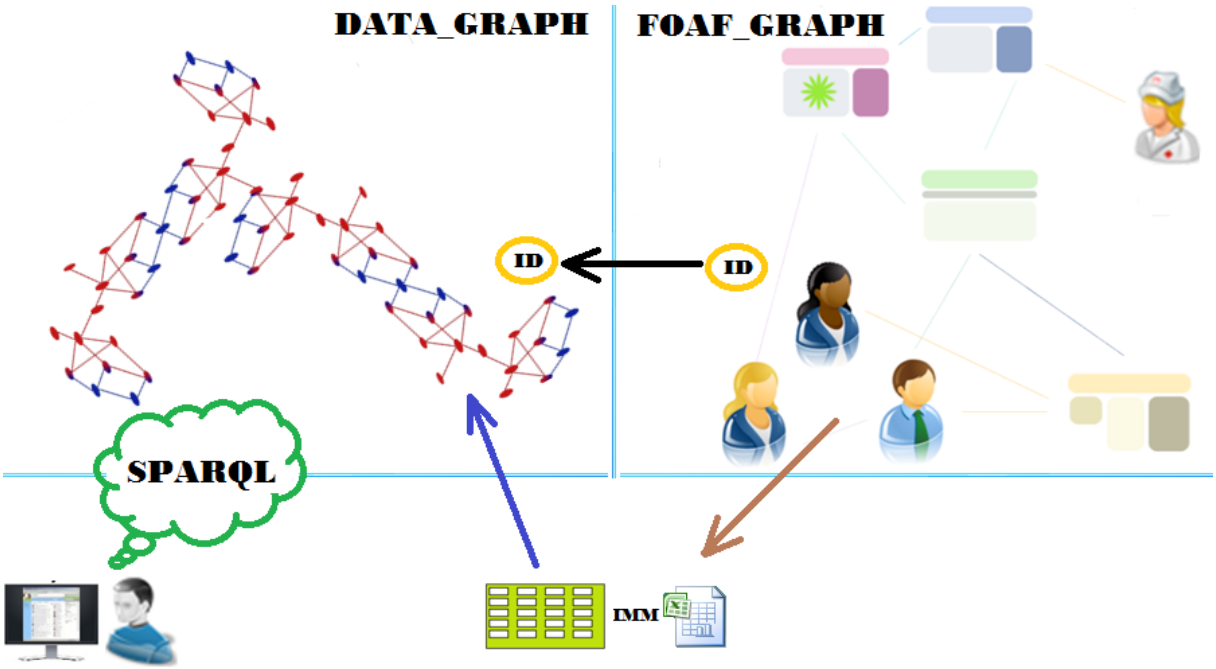

Fig. 22 – Separação de grafos de dados clínicos e de dados pessoais no Virtuoso

5.3.4. Visualização de Dados

Esta funcionalidade permite visualizar os dados dos testes médicos anotados dos pacientes. Como é possível ver na figura seguinte, o utilizador pode escolher um conjunto de opções para filtrar os resultados. Estes filtros podem ser usados isoladamente ou combinados. Desta forma é possível obter resultados filtrados pelo identificador do paciente, pela data exata da avaliação médica, pelo teste médico e pelo sexo dos pacientes. Existem ainda os filtros de início e fim de data da avaliação médica e os filtros de início e de fim de data de nascimento.

Os resultados obtidos são apresentados ao utilizador sob a forma de uma *grid view*, em vários níveis de informação, sendo permitido ao utilizador escolher apenas os dados pretendidos do nível seguinte.

- Nível de informação 1 – É constituído por dados gerais do paciente como o seu identificador, o sexo, a data de nascimento e o nível de escolaridade.
- Nível de informação 2 – É constituído por dados relativos à avaliação médica, do paciente escolhido, tais como, a data da avaliação, a idade do paciente, o grupo a que o paciente pertence relativamente à *Bateria de Lisboa para Avaliação da Demência*, a duração da doença e o diagnóstico efetuado pelo médico.

- Nível de informação 3 – Constituído pelos dados relativos aos testes médicos da avaliação escolhida tais como o seu comentário, valor efetivo e um valor normalizado denominado *ZScore*. Este valor é normalmente normalizado de acordo com fatores como a idade, o sexo, a escolaridade face à população em análise. De referir que os testes simples podem ter um valor e um *ZScore* associado, mas os testes compostos apenas tem um valor e *ZScore* associado aos seus componentes.
- Nível de informação 4 – Constituído pelos dados relativos aos componentes do teste médico escolhido tais como o seu comentário, valor efetivo e um valor normalizado.

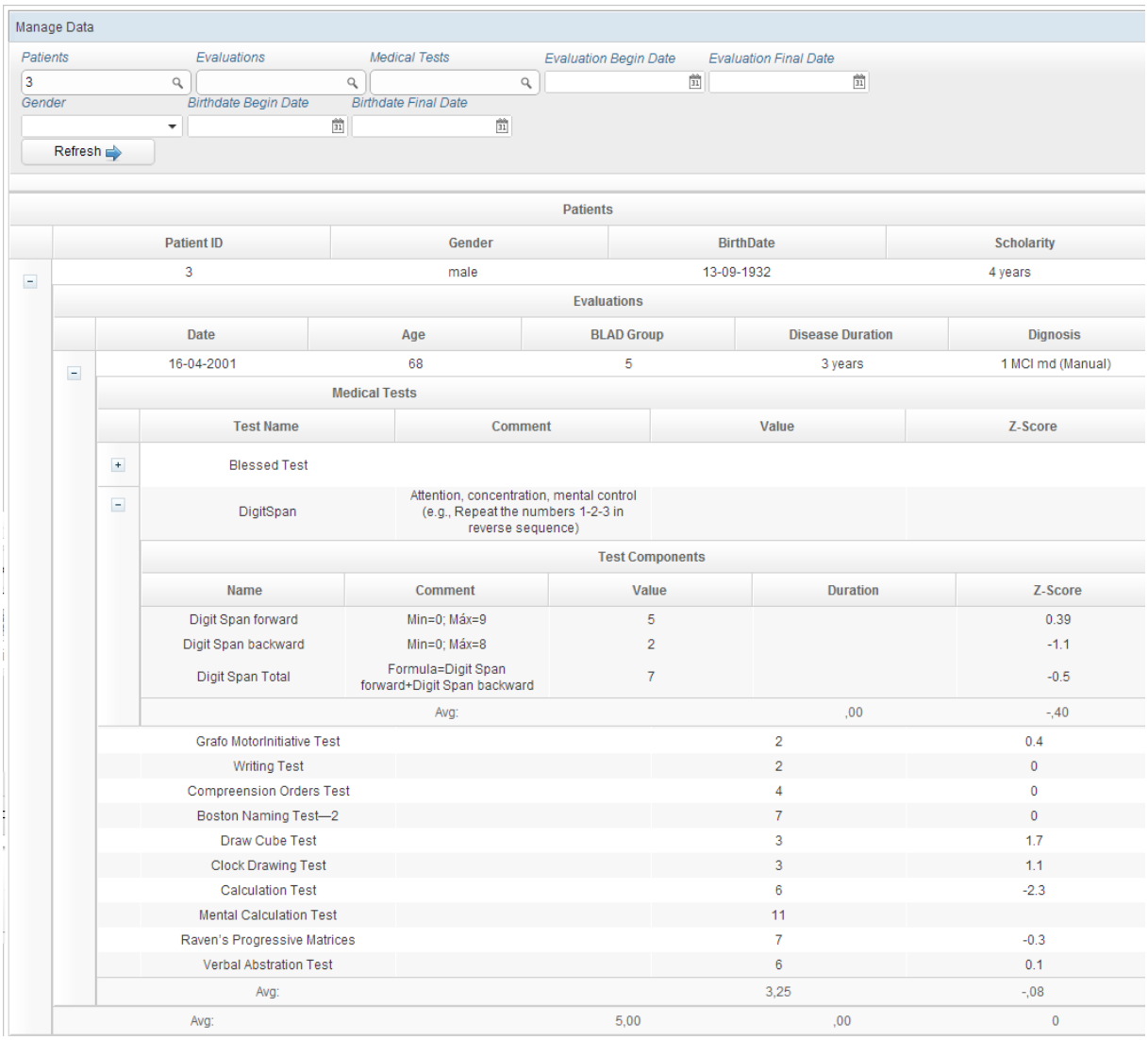

Fig. 23 – Funcionalidade de Manage Data

Esta funcionalidade, quando efetuada por um utilizador com o perfil *doctor* permite visualizar, para os seus pacientes, conjuntamente com o identificador do paciente o respetivo nome. Desta forma, o médico pode escolher facilmente o paciente cujos dados pretende

visualizar, não se colocando o problema da confidencialidade destes dados em relação aos seus próprios pacientes.

5.3.5. Diagnóstico e Prognóstico

Esta funcionalidade permite a interação desta aplicação com um sistema de suporte à decisão capaz de predizer diagnósticos e prever prognósticos num intervalo temporal pré-definido. A Mild Cognitive Impairment (MCI) é considerada uma fase preliminar da doença de Alzheimer. Os doentes diagnosticados com MCI tem uma maior probabilidade de evoluírem para as fases posteriores da doença. O sistema de suporte à decisão é um *software* de prospeção de dados, baseado num modelo desenvolvido com auxílio do WEKA, que processa um vasto conjunto de dados reais relativos a testes neuropsicológicos de pacientes já diagnosticados com MCI ou AD, e aplica-lhes modelos de classificação criados por técnicas de prospeção de dados. A importação de novos dados cedidos pela aplicação Web, através de um *Web Service*, permite ao *software* o reajuste dos modelos refletindo-se no seu comportamento.

O sistema permite para uma dada avaliação médica, atribuir automaticamente o diagnóstico do paciente relativamente a fase da doença de Alzheimer em que este se encontra. Nos casos em que o doente se encontra na fase de *Mild Cognitive Impairment* (MCI) é ainda possível efetuar o prognóstico a 2, 3 e 4 anos da evolução da doença no paciente. Esta funcionalidade necessita de consumir um *Web Service* que disponibiliza as operações de diagnóstico e prognóstico a 2, 3 e 4 anos. Cada uma destas operações recebe como parâmetros o classificador a utilizar e um conjunto de dados de uma avaliação médica. Os classificadores disponíveis são Redes Neuronais (NN), Naive Bayes (NB), de K-vizinhos mais próximos (KNN), Support Vector Machines (SVM) de Kernel Polinomial (Poly), SVM Gaussiano de Funções de Base Radial (RBF) e Árvore de Decisão J48.

Cada operação do *Web Service* devolve um intervalo de confiança entre 0 e 1. No caso do problema do diagnóstico, se o modelo retornar 0 significa que o paciente tem 0% de probabilidade de se encontrar na fase MCI, e, se retornar 1 significa que o paciente tem 100% de probabilidade de se encontrar nessa fase. No problema do prognóstico a 2, 3 ou 4 anos, se o modelo retornar 1 significa que tem 100% de probabilidade de não evoluir enquanto se retornar 0 significa que tem 0% de probabilidade de não evoluir, i.e. evoluirá para a fase de demência. A figura seguinte permite observar que na avaliação médica de 22-11-2004 relativa ao paciente 9 é atribuído um diagnóstico de 0,7999 e de 1.0, quando se utiliza respetivamente

o classificador Knn e Poly, obtendo a probabilidade do doente ter MCI de 79, 99% e de 100%. Relativamente à probabilidade a 2 anos de a doença evoluir é de 0,9479 para o classificador RBF e de 0,753 para o classificador J48, significando isto que a probabilidade de não evolução da doença de MCI para demência é de 94,79% e de 75,3%, respetivamente.

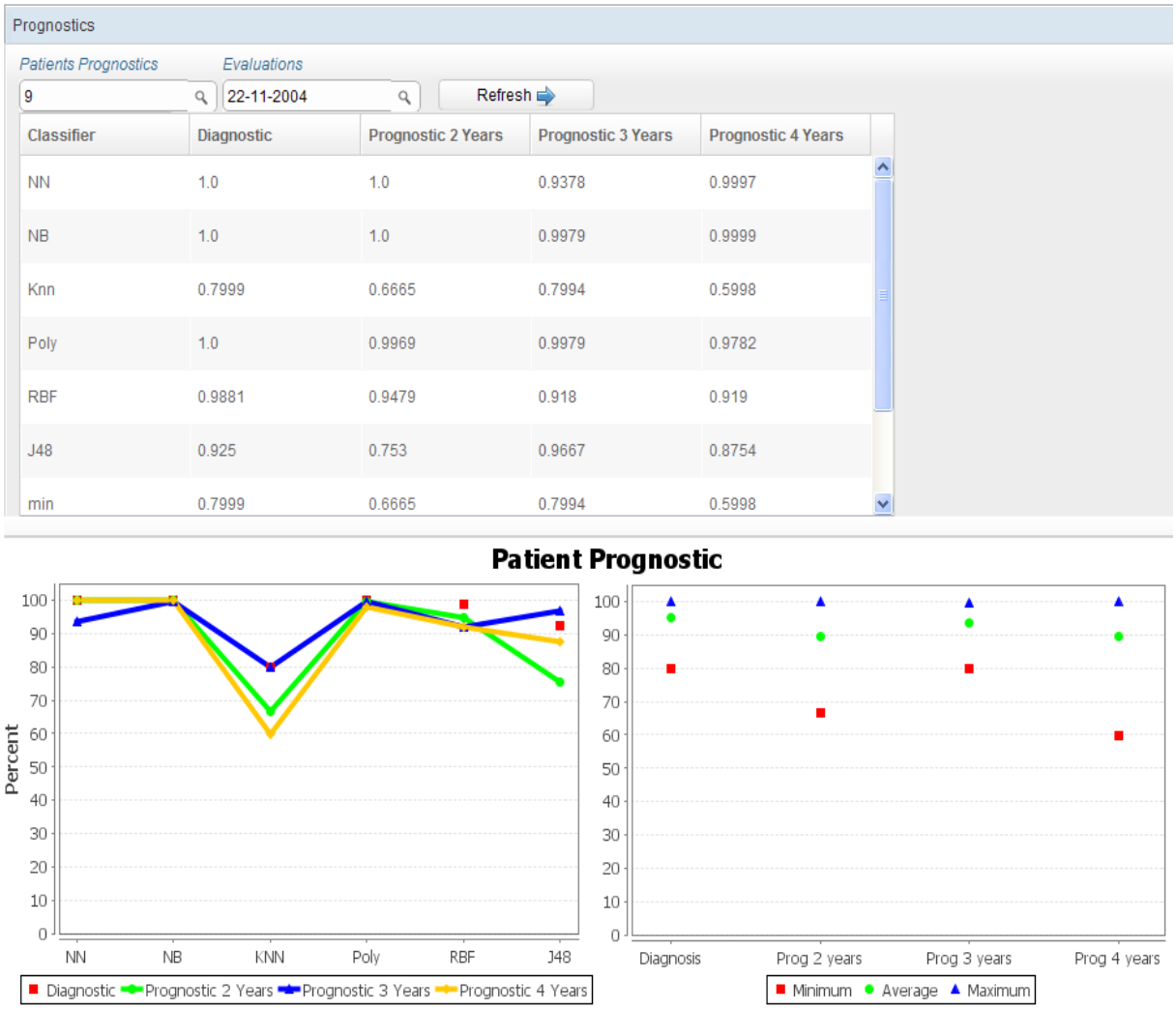

Fig. 24 – Funcionalidade de Prognostic

5.3.6. Navegação na Ontologia

Esta funcionalidade permite uma navegação pela ontologia Neuropsychological Test Ontology (NTO), visualizado a sua hierarquia de classes, instâncias, propriedades, anotações e tipos de dados. O código foi devidamente adaptado do projeto Ontology-browser, disponível em http://code.google.com/p/ontology-browser/.

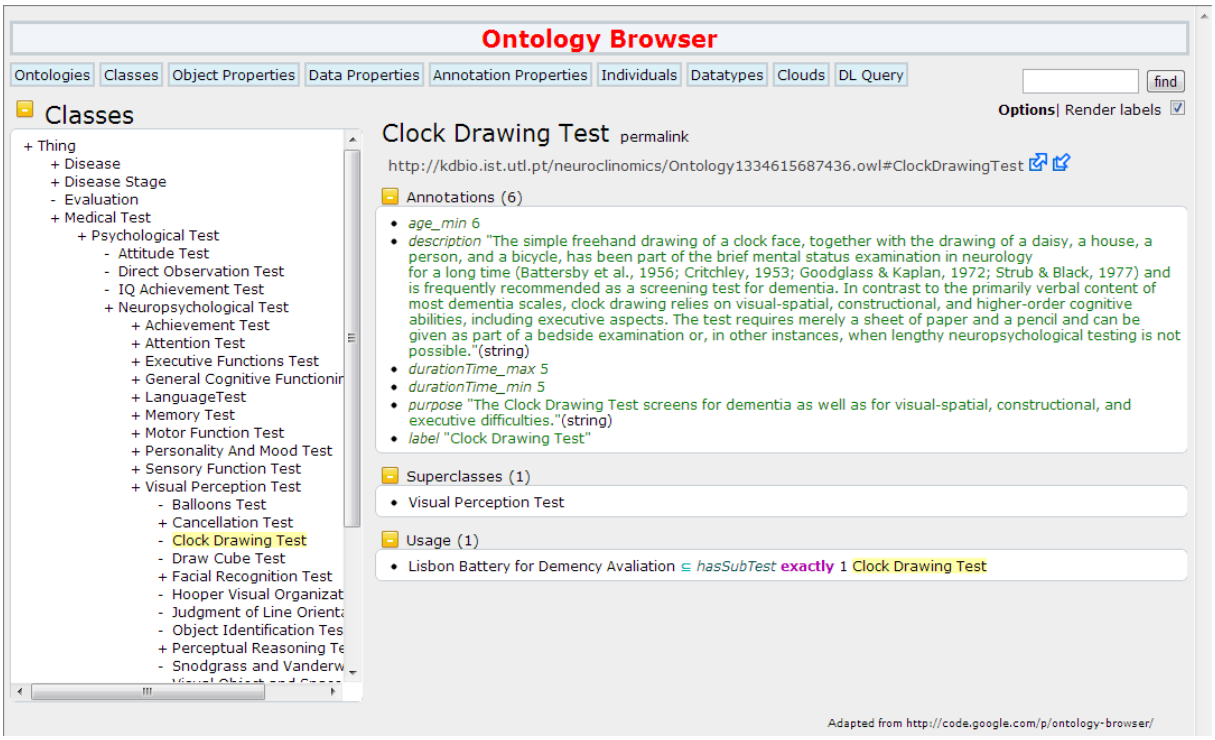

Fig. 25 – Funcionalidade de Ontology Browser

5.3.7. Consultas Sparql

Esta funcionalidade permite ao utilizador efetuar consultas no grafo DATA_GRAPH que reside no servidor Virtuoso. Este grafo contém a ontologia NTO e os dados RDF de testes médicos anotados. O utilizador tem desta forma uma ferramenta útil e flexível de modo a poder interagir com os dados semânticos e sobre eles obter as respostas pretendidas. Esta consola divide-se em 2 componentes. O componente da consulta encontra-se do lado esquerdo e permite ao utilizador escrever as questões e submetê-las, tendo a possibilidade de escolher o formato da resposta como RDF/JSON. Os resultados são apresentados no componente do lado direito.

O JavaScript Object Notation (JSON) é um formato de intercâmbio de dados que permite definir estruturas de dados de uma forma ligeira e legível para humanos. É sintaticamente mais simples, legível e pequena do que o formato XML. O RDF/JSON representa um conjunto de triplos RDF como um conjunto de estruturas de dados enquadradas. Os triplos sujeito, predicado e objeto são estruturados como { “sujeito” : { “predicado” : [objeto] } }.

A figura seguinte demonstra uma consulta para obter os 10 primeiros testes pertencentes ao tipo Teste Psicológico e os respetivos resultados.

Fig. 26 – Funcionalidade de SPARQL Console

5.3.8. Testes Neuropsicológicos

Esta funcionalidade permite de uma forma intuitiva visualizar os testes psicológicos existentes na ontologia NTO, bem como as suas características. A árvore situada do lado esquerdo permite verificar a hierarquia dos testes e escolher o teste pretendido. Após a escolha do teste é possível observar algumas das suas caraterísticas como finalidade, descrição, comentários, idade mínima e máxima dos pacientes recomendada para a realização eficaz dos testes, e, a duração mínima e máxima do teste em minutos. É ainda possível observar do lado direito um ficheiro em formato pdf com enunciado do teste escolhido, caso exista disponível.

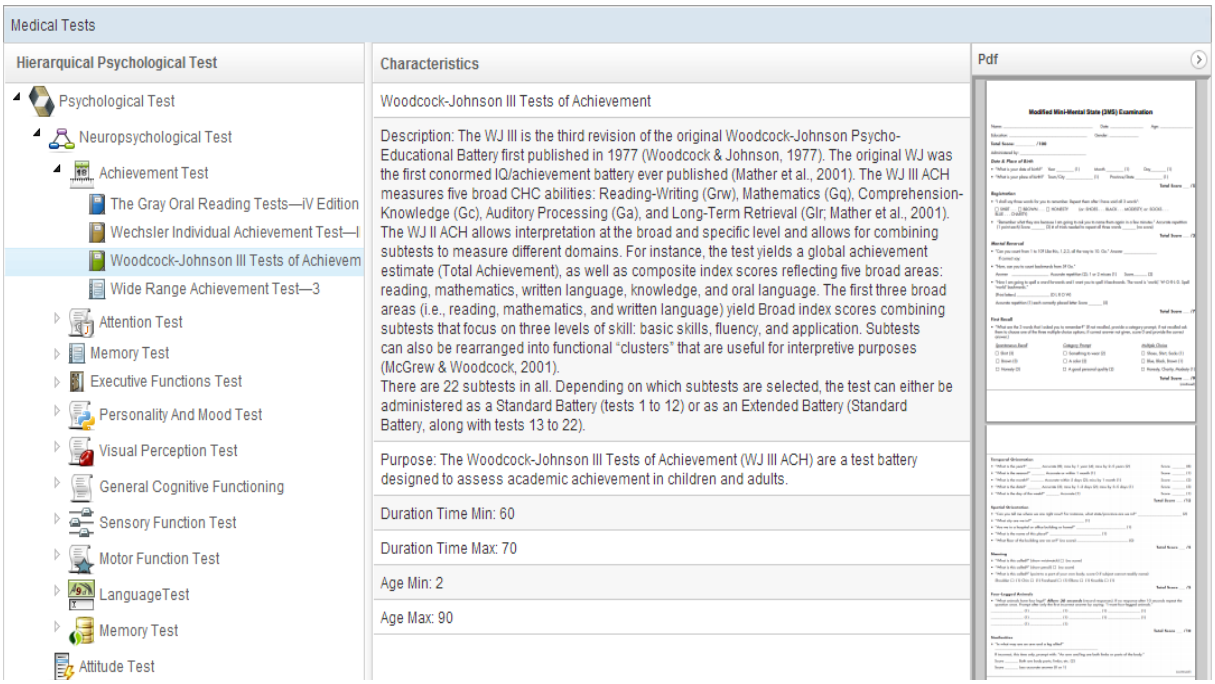

Fig. 27 – Funcionalidade de Medical Tests

5.4. Sumário

Neste capítulo descreveu-se a interface gráfica disponibilizada ao utilizador, as funcionalidades do sistema e as abordagens adotadas, de onde se destaca: a temática do controle de acessos através de permissões por papéis e utilização de OpenId; o carregamento no repositório *triple store* de dados anotados semanticamente de acordo com uma ontologia NTO*;* a problemática da privacidade dos dados e a sua visualização; e a interação com um Web Service para permitir atribuir automaticamente aos pacientes diagnósticos e efetuar prognósticos sobre a evolução da doença de Alzheimer, de acordo com os dados disponíveis.

6. Conclusões

Nesta dissertação foi implementado um protótipo aplicacional de um sistema de informação e extração de conhecimento para análise e gestão de informação médica. Inicialmente foi desenvolvida e apresentada uma ontologia denominada *Neuropsychological Test Ontology* que permite descrever e relacionar conceitos sobre a temática da aplicação de testes neuropsicológicos a pacientes potencialmente com a doença de Alzheimer. Esta estrutura de representação do conhecimento serviu de base para a anotação semântica de dados reais, originalmente disponíveis como ficheiros Excel, relativos à aplicação de testes médicos neuropsicológicos pela equipa de médicos do IMM.

O sistema contém um repositório de dados *triple store* implementado pelo servidor *Virtuoso* e um servidor aplicacional Apache *Tomcat* no qual está alojado a aplicação Web desenvolvida. O servidor *Virtuoso* dispõe de um motor de consultas SPARQL e um módulo de integração com a *API Jena*. A aplicação Web foi desenvolvida sobre a *ZK Framework*, utilizando tecnologia *Ajax* e linguagem de programação Java, e integrando a API Jena. Esta API permitiu programaticamente efetuar o carregamento dos dados anotados no repositório.

A aplicação Web utiliza um controle de acessos através de permissões por papéis e a tecnologia OpenId para a autenticação dos utilizadores. Do conjunto de funcionalidades disponíveis destacam-se: a visualização da ontologia NTO, a introdução de novos pacientes e médicos no sistema; o carregamento de novos dados anotados no repositório *triple store;* a visualização de informação relativa às avaliações médicas; a consulta de dados através de uma interface SPARQL; a interação com um *Web Service* que permite atribuir automaticamente aos pacientes diagnósticos e efetuar prognósticos sobre a evolução da doença de Alzheimer.

A aplicação interage com um *software* de prospeção de dados através de um *Web Service* implementado com tecnologia SOAP, que disponibiliza um serviço capaz de predizer diagnósticos e prever prognósticos num intervalo temporal pré-definido e ainda um serviço de importação de dados. Este *software* permite para uma dada avaliação médica, atribuir automaticamente o diagnóstico do paciente relativamente a fase da doença de Alzheimer em que este se encontra, efetuando ainda o prognóstico a 2, 3 e 4 anos da evolução da doença no paciente nos casos em que o doente ainda se encontra na fase MCI. Isto é conseguido através da aplicação de modelos de classificação criados por técnicas de prospeção de dados. A importação de dados cedidos pela aplicação Web, através do *Web Service* permite que estes modelos possam ser reajustados refletindo-se no seu comportamento.

A problemática da privacidade dos dados médicos foi salvaguardada separando a informação pessoal dos pacientes da sua informação médica, residindo esta em grafos distintos, e, permitindo apenas a consulta dos dados médicos por via de consola SPARQL.

6.1. Contribuições e trabalho futuro

A ontologia criada representa o esforço para formular um esquema conceptual rigoroso e exaustivo dentro deste domínio de conhecimento dos testes médicos neuropsicológicos. Ao existir uma concordância semântica relativamente aos conceitos e suas relações é possível a partilha dados e conhecimento de modo coerente e consistente entre várias aplicações deste domínio. Abre-se assim caminho para que dados de testes neuropsicológicos realizados por pacientes em todo mundo possam ser anotados, de modo a que outros médicos e investigadores possam pesquisá-los e extrair dai novo conhecimento.

Como trabalho futuro proponho a inserção no repositório de dados clínicos provenientes de fontes externas ao projeto como sejam a BRAINnet, ADNI e dbSNP. Estes dados deverão ser anotados semanticamente de acordo com a NTO de modo a disponibilizar um vasto conjunto de dados integrados potenciando a extração e a partilha de novos conhecimentos. Por outro lado, a ontologia desenvolvida focou-se essencialmente em testes neuropsicológicos realizados a pacientes potencialmente com doença de Alzheimer. A extensão desta ontologia a outras doenças neurodegenerativas como o caso da doença Esclerose Lateral Amiotrófica deverá ser também um trabalho a realizar no futuro.

Referências

Adamusiak, T., Burdett, T. e Kurbatova, N. 2011. *OntoCAT - simple ontology search and integration in Java, R and REST/Javascript.* s.l. : BMC Bioinformatics, 2011.

Antezana, E., et al. 2009a. *BioGateway: A Semantic Systems Biology Tool forthe Life Sciences.* s.l. : BMC Bioinformatics, 2009.

Antezana, E., et al. 2009b. *The Cell Cycle Ontology: an application ontology for the representation and integrated analysis of the cell cycle process.* s.l. : Genome Biology, 10:R58, 2009/2.

Antezana, Erick, Kuiper, M. e Mironov, V. 2009c. *Biological knowledge management:the emerging role of the semantic web technologies.* s.l. : Briefings in Bioinformatics, 10(4):392–407, 2009.

Aranguren, M. E., et al. 2008. *Ontology Design Patterns for bio-ontologies: a case study on the Cell Cycle Ontology.* s.l. : BMC bioinformatics, 9(Suppl 5):S1., 2008.

Belleau, F., et al. 2008. *Bio2RDF: Towards a mashup to build bioinformatics knowledge systems.* s.l. : Journal of Biomedical Informatics, 2008.

Berners-Lee, Tim, Fielding, R. e Masinter, L. 2005. *Uniform resource identifier(uri): Generic syntax.* s.l. : RFC3986, 2005.

Berners-Lee, Tim, Hendler, J. and Lassila, O. 2001. *The Semantic.* s.l. : Scientific American, 284, 34-43., 2001.

Bishop, Matt. 2005. *Introduction to Computer Security.* s.l. : Addison-Wesley, 2005. ISBN: 0-321-24744-2.

Bizer, C., Heath, T. e Berners-Lee, T. 2009. *Linked Data - The Story So Far.* s.l. : International Journal on Semantic Web and Information Systems, 2009.

Bizer, Heath, Tom e Christian. 2011. *Linked Data: Evolving the Web into a Global Data Space (1st edition). Synthesis Lectures on the Semantic Web: Theory and Technology, 1:1, 1-136.* s.l. : Morgan & Claypool., 2011.

Boisot, M. e Canals, A. 2004. *Data, information and knowledge: have we got it right?* s.l. : Journal of Evolutionary Economics., 2004. Vols. 14(1):43–67.

Bono, B., et al. 2011. *The RICORDO approach to semantic interoperability for biomedical data and models: strategy, standards and solutions.* s.l. : BMC Bioinformatics, 2011.

Cali, Andrea, et al. 2001. *Accessing data integration systems through conceptual schemas.* s.l. : Lectures Notes in Computer Science, 2001.

Chaves, Ana Paula e Steinmacher, Igor. 2011. *OntoDiSENv1: an Ontology to Support Global Software Development.* Campo Mourão, Brasil : Federal Technological University of Paraná, 2011.

Cheun, Kei-Hoi, et al. 2007. *Semantic Web Approach to database integration in the life sciences.* s.l. : in C. J. O. baker & Kei-Hoi Cheun Semantic Web, Springer US, pp. 11-30, 2007.

Clercq, Jan De. 2002. *Single Sign-on Architectures.* s.l. : in Infrastructure Security International, 2002.

Clermont, Gilles., et al. 2009. *Bridging the gap between systems biology and medicine.* 2009. 1(9):88.1-88.6.

Erling, Orri e Mikhailov, Ivan. 2007. *RDF support in the Virtuoso DBMS.* s.l. : In Conf. on Social Semantic Web., 2007.

Ferraiolo, David F. e Kuhn, D. Richard. 1992. *Role-based access controls.* s.l. : In Proceedings of the 15th National Computer Security Conference, 1992.

Gordon, E. 2007. *Integrating genomics and neuromarkers for the era of brain-related personalized medicine.* s.l. : Personalized Medicine., 2007. 4(2):201-215.

Gruber, T.R. 2007. *Ontology. Disponível em: <http://tomgruber.org/writing/ontology-definition-2007.htm>.* s.l. : Acedido em 17SET2012, 2007.

Guarino, Nicola. 1998. *Formal ontology and information systems.* Italia : International Conference on Formal Ontology, 1998.

Hey, T., Tansley, S. e Tolle, K. 2009. *The Fourth Paradigm: Data-Intensive Scientific Discovery.* 2009.

Hunter, P., et al. 2010. *A vision and strategy for the virtual physiological human in 2010 and beyond.* s.l. : The Royal Society, 2010.

Kammergruber, W. C. e Ehms, K. 2010. *A Corporate Tagging Framework as Integration Service for Knowledge Workers.* s.l. : Proceedings of I-KNOW, 2010. pag. 4.

Kienast, R. e Baumgartner, C. 2011. *Semantic Data Integration on Biomedical Data Using Semantic Web Technologies.* s.l. : Bioinformatics - Trends and Methodologies, Mahmood A. Mahdavi (Ed.), ISBN: 978-953-307-282-1, InTech, 2011.

Kitano, H. 2002. *Systems Biology: a brief overview.* s.l. : Science, 2002. 295:1662-4..

Molle, R. e Haarslev, V. 2003. *Racer: An OWL Reasoning Agent for the Semantic Web.* Halifax, Canada : In Proceedings of the International Workshop on Applications, Products and Services of Web-based Support Systems, in conjunction with the 2003 IEEE/WIC International Conference on Web Intelligence, 2003.

Noorbakhsh, F., Overall, C. M. e Power, C. 2009. *Deciphering complex mechanisms in neurodegenerative diseases: the advent of systems biology. Trends in Neurosciences.* 2009. 32(2):88-100.

Noy, N. e McGuinness, D.L. 2001. *Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05.* Stanford. : Stanford Medical Informatics, 2001. DOI:10.1.1.136.5085.

Noy, N. F., et al. 2009. *BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Res.BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Res.* 2009.

Ouksel, Aris M. e Sheth, Amit. 1999. *Semantic Interoperability in Global Information Systems.* s.l. : SIGMOID Record 28(1):5-12, 1999.

OWL Web Ontology Language Reference. **W3C. 2004.** s.l. : W3C recommendation, 2004. disponível em http://www.w3.org/TR/owl-ref/.

Recordon, David e Reed, Drummond. 2006. *OpenID 2.0: A Platform for User-Centric Identity Management.* Alexandria, Verginia, USA, : in Conference on Computer and Communications Security Proceedings of the second ACM workshop on Digital identity management, 2006. ISBN: 1-59593-547-9..

Ruttenberg, A., et al. 2007. *Advancing translational research with the Semantic Web.* s.l. : BMC Bioinformatics, 2007.

Sandhu, R. S., et al. 1996. *Role-based access control models.* s.l. : IEEE Computer, 29(2)., 1996.

Schadt, E. 2009. *Molecular networks as sensors and drivers of common human diseases.* s.l. : Nature 461, 218-22, 2009. Nature, 461:218-223, February 2009.

Signore, Oreste. 2006. *Representing Knowledge in the Semantic Web.* 2006. W3C Italy.

Sirin, E., et al. 2007. *Pellet: A practical OWL-DL reasoner.* s.l. : Jornal of Web Semantics, 5(2):51–53, 2007.

Smith, B., et al. 2007. *The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration.* s.l. : Nature Biotechnology 25, 1251 - 1255., 2007.

Staab, S. e Studer, R. 2009. *Handbook on Ontologies.* s.l. : International Handbooks on Information Systems, second ed., 2009.

Strauss, Esther, Sherman, Elisabeth M. S. e Spreen, Otfried. 2006. *A Compendium of Neuropsychological Tests: Administration, Norms and Commentary.* Oxford University Press : Third Edition, 2006.

Tsarkov, D. e Horrocks, I. 2006. *FaCT++ Description Logic Reasoner: System Description.* s.l. : Lecture Notes in Computer Science. 4130. pp. 292–297, 2006.

Wache, H., et al. 2001. *Ontology-Based Integration of Information — A Survey of Existing Approaches.* Seattle, WA : in Proc. of the IJCAI-01 Workshop: Ontologies and Information Sharing, 2001. pp. pp. 108-117.

Wongthongtham, Pornpi, Chang, Elizabeth e Dillon, Tharam. 2007. *Ontology modelling notations for software engineering knowledge representation.* s.l. : In: IEEE International Conference on Digital Ecosystems and Technologies, 2007.