 **INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA**

**HYBRID ONTOLOGY MAPPING INTERFACE**

**Projeto Final de Curso**

Licenciatura em Engenharia Informática e Computadores

Ana Carolina Baptista

[41487@alunos.isel.ipl.pt](mailto:41487@alunos.isel.ipl.pt)

960314580

Eliane Almeida [41467@alunos.isel.ipl.pt](mailto:41467@alunos.isel.ipl.pt)

960271968

**Relatório de Progresso**

Orientadores:

Cátia Vaz, ISEL, [cvaz@cc.isel.ipl.pt](mailto:cvaz@cc.isel.ipl.pt)

José Simão, ISEL, [jsimao@cc.isel.ipl.pt](mailto:jsimao@cc.isel.ipl.pt)

Alexandre P. Francisco, IST, [aplf@ist.utl.pt](mailto:aplf@ist.utl.pt)

Abril de 2018

Índice

[Lista de Figuras 3](#_Toc512183398)

[Lista de Tabelas 4](#_Toc512183399)

[1 Introdução 5](#_Toc512183400)

[2 Descrição do problema 6](#_Toc512183401)

[3 Chaos Pop 7](#_Toc512183402)

[4 Arquitetura 10](#_Toc512183403)

[4.1 Descrição 10](#_Toc512183404)

[4.2 Hybrid Ontology Mapping Interface (H.O.M.I) 11](#_Toc512183405)

[4.2.1 Tecnologias 11](#_Toc512183406)

[4.2.2 Base de dados 12](#_Toc512183407)

[5 Progresso do projeto 13](#_Toc512183408)

[6 Referencias 13](#_Toc512183409)

[7 Bibliografia 13](#_Toc512183410)

# Lista de Figuras

[Figura 3.0.1 - Exemplo de ficheiro de dados 7](#_Toc512007291)

[Figura 3.0.2 - Exemplo de árvore gerada 8](#_Toc512007292)

[Figura 3.0.3 - Grafo de uma ontologia 8](#_Toc512007293)

[Figura 3.0.4 - Diagrama UML dos services do Chaos Pop 9](#_Toc512007294)

[Figura 4.1 – Arquitetura da aplicação 10](#_Toc512007295)

# Lista de Tabelas

Tabela 2.1 Tabela de comparação entre ferramentas

# Introdução

Preciso que se fale de ontologias e de OWL na introdução para não cair de para quedas no capitulo a seguir

# Descrição do problema

Com o crescimento da popularidade de OWL para a descrição de dados, naturalmente apareceram também alguns softwares para a edição e manipulação de OWL.

Neste capitulo iremos falar sobre alguns destes programas e sobre as diferenças entre eles. Por fim iremos explicar como o nosso projeto de insere nesta área bem no que o difere destes programas que já existem.

Iremos comparar 4 programas populares: Apollo, Protege e Swoop.

## 2.1 Apollo

Apollo é modelador de fácil utilização para o utilizador, de utilização gratuita e local. Nesta aplicação, um utilizador pode modelar ontologias com noções básicas de ontologias (classes, relações, instâncias, etc). Também é possível criar novas instancias a partir das classes presentes nessas ontologias. Alguns dos pontos fortes deste software é o seu corretor de tipos que mantem a consistência de tipos durante o processo, bem como o armazenamento das ontologias (em ficheiros).

Contudo, este programa é relativamente antigo e carece de uma interface gráfica intuitiva, ao contrario dos seus concorrentes. Apollo carece também de extração de informação em web e da possibilidade de dar uma experiencia multiutilizador aos seus utilizadores, para trabalhos em colaboração com mais do que um utilizador.

## 2.2 Protégé

Protege é um editor e modelador de ontologias, de utilização gratuita e tem uma vertente local bem como online (WebProtégé). Tem uma arquitetura baseada em plug-ins o que deu origem ao desenvolvimento de inúmeras ferramentas relacionadas com semântica web. Implementa um conjunto de estruturas modeladoras de conhecimento e ações que suportam a criação, modelação e manipulação de ontologias, complementadas com inúmeras formas de visualização desses dados (?). A customização proporcionada aos seus utilizadores é uma das características que torna esta aplicação numa das mais populares na área.

## 2.3 Swoop

Swoop é um browser e também um editor *open-source*, de utilização gratuita e local. Esta ferramenta contém validadores de OWL e oferece várias formas de visualização gráfica de OWL. É composto também por um ambiente de edição, comparação e fusão entre múltiplas ontologias. As suas capacidades de *hyperlinks* proporciona uma interface de navegação fácil aos seus utilizadores. Um utilizador pode também reutilizar dados ontológicos externos ao colocar os links para a entidade externa ou importando a ontologia completa, pois não é possível importar ontologias parciais, mas é possível realizar pesquisas por múltiplas ontologias.

Durante a nossa pesquisa sobre outras ferramentas e bibliotecas que já existem nesta área, encontramos também, para além de outros editores com características diferentes dos apresentados acima, alguns mapeadores de XML para OWl (Ontmalizer[[1]](#footnote-1) e JXML2OWL[[2]](#footnote-2))

A nossa ferramenta visa juntar os pontos fortes destas aplicações numa só aplicação

Na nossa ferramenta (HOMI) iremos ter uma vertente local bem como uma vertente online, uma visualização simples e fácil para os nossos utilizadores bem como a opção de apenas criar uma nova instancia a partir de uma dada ontologia ou então criar ima nova instancia a partir de uma ontologia e também um exemplo concreto descrito em XML através do mapeamento de conceitos presentes na ontologia.

Tabela Comparação entre várias ferramentas e a nossa aplicação

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Característica  -  Programa | Criação de novas instâncias | Fácil Utilização | Armazenamento | Interface Intuitiva | Versão Online | Mapeia XML para OWL |
| Apollo | Sim | Sim | Sim, em ficheiros | Não | Não | Não |
| Protégé | Sim | Sim | Sim | Sim | Sim | Não |
| Swoop | Sim | ? probably | Sim, em modelos HTML | Sim | Não | Não |
| Ontomalizer & JXML2OOWL | Sim | Kinda | Não | Não contem interface | Não | Sim |
| HOMI | Sim | Sim | Sim | Sim | Sim | Sim |

# 3 Chaos Pop

Chaos Pop é uma API que foi desenvolvida para permitir mapear ontologias dado um caso concreto da mesma.

Para que um mapeamento seja realizado são indispensáveis dois ficheiros: um OWL referente à uma ontologia (*OntologyFile*) e outro que representa um caso concreto da ontologia em questão (*DataFile*). Neste momento, os ficheiros que são suportáveis como *DataFile* são de extensões JSON e XML.

Aquando da submissão de ficheiros *DataFile* e *OntologyFile*, a informação presente nestes são guardadas numa base de dados remota. Quando pretende-se realizar o mapeamento de dados, é necessário buscar os ficheiros que tenciona mapear, sejam eles um ou mais *DataFile’*s e *OntologyFile’*s.

No que diz respeito ao *DataFile* os dados são guardados em *nodes,* formando assim uma árvore. Por exemplo, se for submetido o ficheiro XML da Figura 3.0.1, a árvore abstraída a partir dos *nodes* deste será a apresentada na Figura 3.0.2.

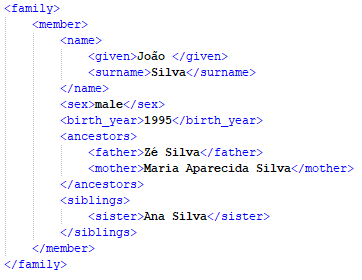


Figura 3.. - Exemplo de ficheiro de dados

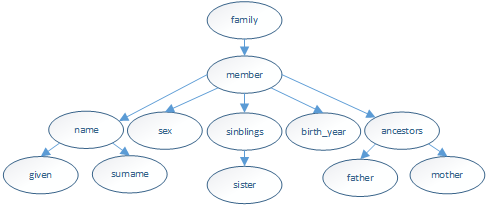


Figura 3.. - Exemplo de árvore gerada

Por outro lado, quando é submetido um *OntologyFile,* a informação que contém neste é mantida em base de dados da seguinte forma:

* uma *ontology* pode ter *classes, data properties* e *object properties*, sendo assim é possível obter estes dados referentes a uma ontologia com o identificador da mesma;
* uma *class* pode ter *data properties* e *object properties,* logo é possível aceder a esta informação com base no identificador da ontologia na qual a classe pertence e no identificador da classe em questão.

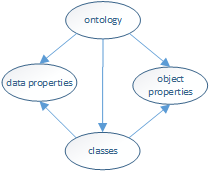


Figura 3.. - Grafo de uma ontologia

Quando for realizar o mapeamento de uma ontologia, está ao critério do utilizador decidir quais dados desta ele pretende obter, assim como a ordem com que deseja mapear estes.

Para submeter ficheiros, obter dados referentes aos ficheiros submetidos e mapear os conceitos existentes é necessário ter conhecimento dos *endpoints* existentes no Chaos Pop que satisfazem estas necessidades. Estes *endpoints* são pertencentes ao *services* eestão divididos e implementados em determinadas classes de acordo com a sua funcionalidade. O diagrama UML apresentado abaixo mostra as funções relativas a cada *endpoint.*

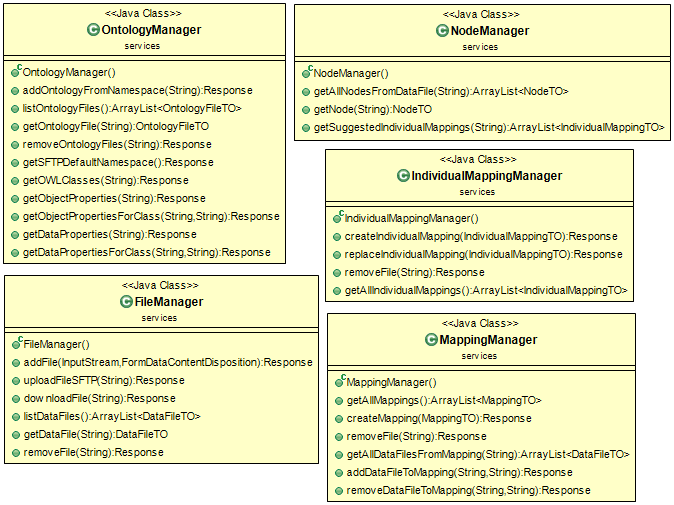


Figura 3.. - Diagrama UML dos services do Chaos Pop

# Arquitetura

Neste capítulo descreveremos detalhadamente cada módulo da aplicação, assim como a forma como estes interagem entre si e as tecnologias utilizadas na implementação de cada um.

## **4.1 Descrição**

A Figura 4.1 mostra a arquitetura da aplicação, na qual está representada a interação entre os distintos módulos existentes. Esta interação inicia-se quando o utilizador (*User*) insere um ficheiro com a definição de uma ontologia (*Ontology File*) e, opcionalmente, um segundo ficheiro (*Data File*). Estes ficheiros irão ser submetidos a API Chaos Pop. De seguida irá ser gerada uma interface gráfica onde o usuário poderá anotar valores aos vários conceitos presentes no *Ontology File* ou anotar os conceitos do *Data File* com os termos do *Ontology File*. No final deste processo, é gerado um novo ficheiro OWL que contém os dados descritos de acordo com *Ontology File*. Iremos também dar a opção ao usuário de guardar os ficheiros de input e output numa base de dados remota.

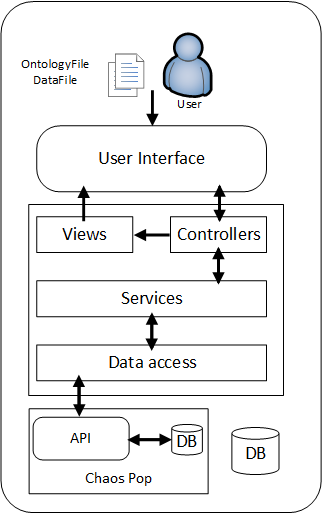


Figura . – Arquitetura da aplicação

## **4.2 Hybrid Ontology Mapping Interface (H.O.M.I)**

A aplicação foi implementada por níveis de acesso, na qual cada nível apenas comunica com o superior ou inferior, nunca saltando um nível durante a comunicação. Esta implementação proporciona uma maior facilidade na manipulação das funcionalidades.

Os níveis existentes são:

* ***controllers:*** contém todos os *endpoints* possíveis de serem acedidos através da *user interface* e comunica com *views*e *services*;
* ***services:*** nível intermédio entre *controllers* e *data access*, contém toda a lógica necessária para a execução das operações disponíveis;
* ***data access:***  engloba todo o acesso aos dados. Seja este acesso na API Chaos Pop ou na nossa própria base de dados;
* ***views:*** inclui todas as representações visuais utilizadas em *user interface.*

Em cada um dos níveis existe diferentes módulos, nestes estão implementadas funções usufruindo de algumas tecnologias das quais temos conhecimento.

### **4.2.1 Tecnologias**

**Node.js**

**O que é?** Um *runtime* de JavaScript, que pelo facto de processar o código JavaScript desvinculando-o do browser, possibilita o desenvolvimento de aplicações estáveis e rápidas.

**Por que escolhemos?** Escolhemos esta tecnologia porque já tivemos experiências em outras unidades curriculares e concluímos que fornece uma maneira fácil de construir uma aplicação.

**Onde utilizamos?** Todo o projeto é realizado utilizando esta tecnologia.

**D3.js**

**O que é?** Uma ferramenta para JavaScript que associa os dados ao *Document Object Model(DOM)* e permite manipular estes gerando gráficos usando diretamente padrões web como o HTML e o CSS.

**Por que escolhemos?** Optamos por tal pelo facto de suportar comportamentos dinâmicos de interação e animação e grandes conjuntos de dados. Outra característica que nos chamou a atenção foi com facilidade obter gráficos bonitos visualmente.

**Onde utilizamos?** Utilizamos na apresentação dos dados referentes aos ficheiros de entrada, de modo a obter uma representação intuitiva e de fácil entendimento destes por parte do utilizador.

**Electron**

**O que é?** Um framework utilizado para criar aplicações multiplatformas desktop com tecnologias web (HTML, JavaScript e CSS).

**Por que escolhemos?** Como a nossa aplicação será desenvolvida na tecnologia JavaScript e esta é a suportada pelo Electron, decidimos utilizá-lo para assim não ter a necessidade de aprender a usar uma nova tecnologia no desenvolvimento do sistema desktop.

**Onde utilizamos?** Na criação da aplicação desktop iremos aplicar esta tecnologia.

**Express**

**O que é?** Um framework back-end em Node.js que cria rotas, middlewares, entre outras para facilitar a criação de API’s. Este cria e obtém dados a partir do servidor, independentemente da linguagem que irá utilizá-los.

**Por que escolhemos?** Por já termos experiências com esta tecnologia em outras alturas e a sua utilização ser fácil.

**Onde utilizamos?** Toda as rotas disponíveis na camada *controllers* estão implementadas com base neste.

### **4.2.2 Base de dados**

O armazenamento dos dados relativos aos ficheiros *OntologyFile* e *DataFile* está sendo realizado na base de dados do ChaosPop. Posteriormente, tencionamos alterar isto de modo a que tenhamos a nossa própria base de dados.

# Progresso do projeto

# Referencias

<https://www.w3.org/wiki/Ontology_editors> [21/04/18]

<http://www.ef.uns.ac.rs/mis/archive-pdf/2013%20-%20No2/MIS2013-2-4.pdf> [21/07/18]

<https://www.w3.org/wiki/SemanticWebTools> [22/04/18]

# Bibliografia

1. https://github.com/srdc/ontmalizer [↑](#footnote-ref-1)
2. http://jxml2owl.projects.semwebcentral.org/index.html [↑](#footnote-ref-2)