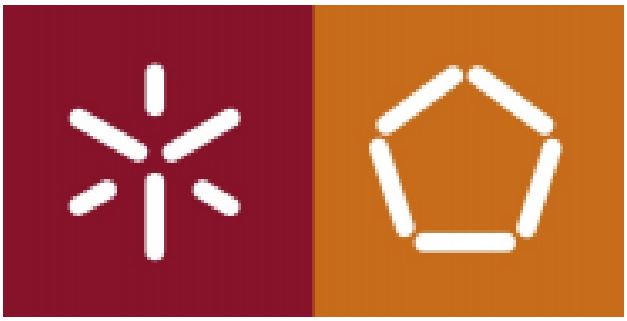


Universidade do Minho

Relatório do Trabalho Prático
Interoperabilidade Semântica

ORCid Search

Hugo Carvalho (A74219)

Luís Lima (A74260)

15 de Junho de 2018

Resumo

Este documento descreve o trabalho prático desenvolvido no âmbito da unidade curricular de **Interoperabilidade Semântica**, do perfil de especialização de **Engenharia de Conhecimento**, tendo como objetivo a implementação de um sistema onde serão aplicados métodos de Interoperabilidade Semântica, em particular arquiteturas baseadas em serviços Web.

Conteúdo

Lista de Figuras

1 Introdução

Vivemos numa sociedade que procura informação por todo o lado e a qualquer altura. Cada dia que passa, os meios tecnológicos facilitam o acesso a informação e a sua propagação. Isto cresce de importância quando precisamos de um sistema de controlo, onde a informação deve transitar com determinadas características básicas mas essenciais como a disponibilidade, confidencialidade e autenticidade, que possa interoperar entre diversas organizações públicas de uma forma segura e fiável.

Muitos sistemas numa grande empresa não foram desenvolvidos para serem capazes de comunicar com outros sistemas. Estes foram desenvolvidos com interfaces, protocolos e formas de comunicação diferentes dificultando muito a sua integração. Para permitir que aplicações empresariais possam comunicar, são utilizados padrões e formatos de arquivos entre os sistemas. Isso pode ser bom para algum tipo de integração pequena, mas pode ser inviável quando o número de aplicações a serem integradas se tornam bastante grandes.

Para que um sistema comunique com o outro, o sistema teria que aprender novas tecnologias e novas Interfaces de Programação de Aplicações (APIs) de cada sistema para que pudesse realizar a integração. Para integrar dois ou três sistemas, não haveria muito problema, porém a integração de vários sistemas seria bastante complicada.

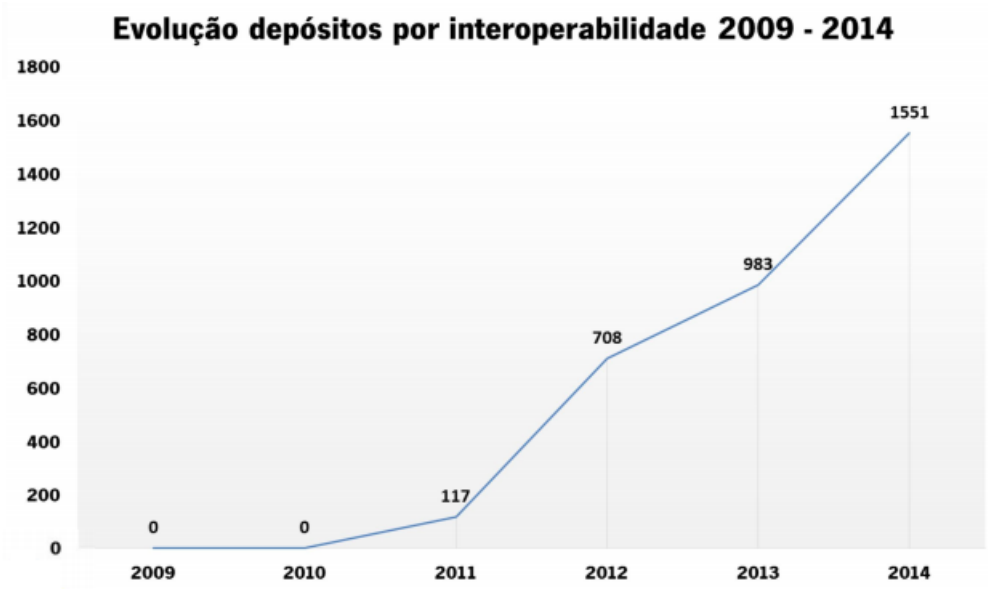


Figura 1: Evolução do número de repositórios de interoperabilidade [1]

É aqui que surge a interoperabilidade, capaz de tratar um grande volume de informações em formatos diversos e oriundos de fontes de informação diferentes. Entende-se por interoperabilidade a interação de dois ou mais sistemas (equipamentos, sistemas de informação ou bases de dados) para trocar informações de acordo com um conjunto de regras definidas e a nível semântico, que se traduz na capacidade de diversos sistemas compartilharem informações conceptualmente compatíveis entre si.

2 Contextualização

O sistema de comunicação científica, nos últimos anos, tem sofrido imensas transformações bastante profundas, relativas a uma emergente comunidade do conhecimento fundamentada nas redes de informação. Estas alterações passam mais propriamente pelos novas formas e meios de publicação dos resultados científicos e pelo aumento da diversidade de fontes e meios para aceder e disseminar a informação.

Cada vez mais, é de elevada complexidade preservar e guardar uma lista atualizada das publicações que os investigadores desenvolvem, quer pela extensa variedade de serviços de informação disponíveis onde os trabalhos podem ser publicados e distribuídos, quer pela quantidade de trabalhos desenvolvidos em co-autoria. Nos processos de administração desta informação existiu sempre a problemática da identificação clara do autor, sendo na grande parte dos casos apenas possível ao nível da instituição.

O *ORCID* (Open Researcher and Contributor Identifier), pretende resolver este problema através da atribuição de um identificador universal e único a cada investigador.

Um *ORCiD iD* é um identificador único, grátis e persistente que acompanha cada pessoa ao longo da sua carreira. Os Registos *ORCID* conservam informações importantes como o nome ou variações de nome, e-mail, instrução, afiliação e atividades, tais como publicações, doações e outros trabalhos académicos, por meio de uma interface que permite administrar a privacidade dos seus dados. O *ORCiD iD* simplifica a atualização de dados e informações à medida que proporciona a integração entre o indivíduo (pesquisador/académico) e o fluxo das suas atividades profissionais, desde a submissão de manuscritos até a publicação de artigos, inscrição em bases de dados, atualização de informações e publicações, coleta e validação de informações e currículos por entidades como órgãos de financiamento, editores, universidades e grupos de pesquisa.

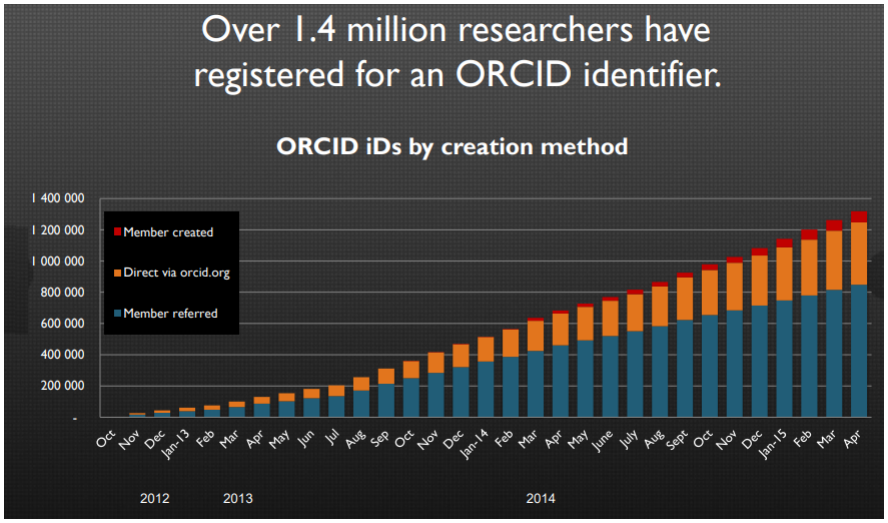


Figura 2: Quantidade de ORCiD criados (2012 – 2015) [2]

O identificador ORCiD garante uma visibilidade mundial, da mesma forma que cria um padrão e desambigua o nome verdadeiro do pesquisador, efetuando a diferenciação de diferentes pesquisadores. Além disso, o ORCID iD facilita a ligação e atualização das publicações e atividades por meio da integração e sincronização de trocas de dados entre organizações que lhe são confiáveis. O procedimento de vinculação é rápido e transparente. Desta forma, não será essencial atualizar a sua carreira ou atividades, e publicações em bases de dados e currículos distintos.

3 Especificação do Problema

Com base na descrição anterior sobre ORCid's, este trabalho surge com o objetivo de apresentar ao utilizador, através de um *browser*, toda a informação organizada de forma simplista e intuitiva. Neste contexto, com a utilização da API ORCid, o grupo definiu um conjunto de dados que são relevantes para a posterior apresentação ao utilizador.

Ao longo desta secção do relatório será descrito todo o problema, bem como todas as respetivas decisões para a sua implementação.

3.1 Descrição dos Dados

Tendo em conta a análise do problema feita anteriormente, definiu-se que seria importante apresentar ao utilizador os atributos: *Title, Type, Local, Year, EID, WOS e ISSN*.

Em seguida apresenta-se uma tabela com a descrição de cada um destes atributos, bem como do seu tipo de dados associado.

Atributo	**Descrição**	**Tipo de Dados**
Title	Título da publicação	String
Type	Tipo de publicação	String
Local	Local de Publicação do Artigo	String
Year	Ano da ultima edição da publicação	Int
EID	Identificador do artigo no Scopus	String
WOS	Identificador do artigo no Web of Science	String
ISSN	Identificador para cada título de publicação	String

Tabela 1: Descrição dos Atributos

3.2 Estratégia de Implementação

Com o intuito de facilitar o acesso aos dados pretendidos por parte do *back-end*, decidiu-se consultar todos os dados diretamente através de consultas à API ORCid. Esta decisão apresenta a vantagem de ser possível obter os dados sempre atualizados quando se faz uma pesquisa, o que poderia não acontecer caso estivéssemos a consultar dados a uma base de dados local. Por outro lado, apresentar a vantagem de não depender de recursos locais para a sua correta execução.

Deste modo, procedeu-se à implementação desde procedimento através do interpretador *Node.js*. O passo mais importante consistiu na implementação correta do modo de obtenção dos dados, uma vez que estes são essenciais para o funcionamento eficaz da aplicação. Neste sentido, apresenta-se, em seguida, um esquema que permite compreender todo o raciocínio implementado no código desenvolvido.

Como podemos verificar no esquema, sempre que é invocado um pedido de dados ao módulo ORCID Searcher, este conecta-se à API através do método *getDataFromAPI()*. Este método basicamente consiste em estabelecer um *request* à API, passando o número de ORCid como parâmetro. Em seguida, e tal como descrito no enunciado do trabalho, a API devolve o resultado num formato *xml* com um conjunto variado de dados relativos ao ORCid escolhido. De volta ao módulo ORCid Searcher, este faz o *parsing* dos dados, guardando numa estrutura *publications* todos os dados pretendidos. Por fim, estes dados são enviados para a aplicação, sendo mostrados ao utilizar através do *browser*.

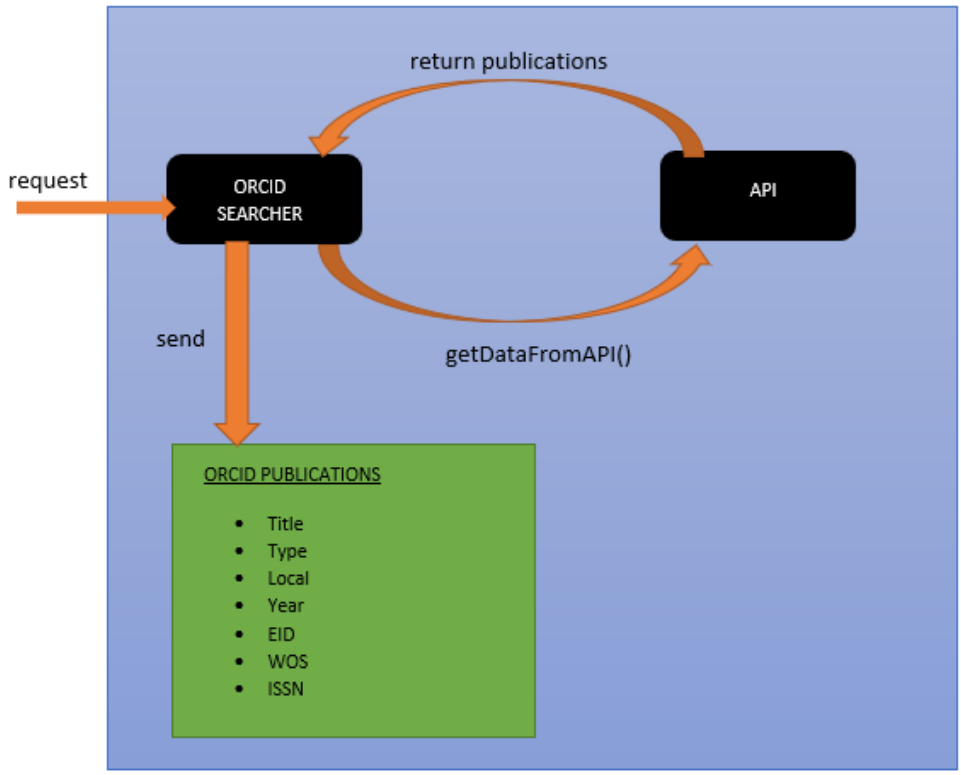


Figura 3: Arquitetura do Sistema

3.3 *Packages* Utilizados

Para a correta implementação do sistema, o grupo optou por recorrer à utilização de alguns *packages* associados ao *Node.js*. Descreve-se, em seguida, cada um destes, com a sua respetiva funcionalidade:

- **request** - *Package* para realizar os pedidos http à API ORCid
- **xml2js** - Conversão do formato *xml* para objetos JS
- **express** - Definição de rotas e criação da aplicação web
- **pug-bootstrap** - Utilização da biblioteca Bootstrap recorrendo ao motor de templates Pug
- **perf_hooks** - Análise de tempos de execução dos pedidos à API

4 Implementação

4.1 Métodos Desenvolvidos

Para a compreensão do código da aplicação desenvolvida, é necessário perceber como funcionam as rotas e o modo de recolha de dados.

Em relação às rotas, tal como podemos constatar no código em baixo, existem os métodos *get* para fazer o render da página, e *post* para obter dados no *back-end* para um determinado ORCID. É de realçar que neste caso existem apenas duas páginas, uma de pesquisa ("index") e outra de resultados("search"). Em seguida apresenta-se o pseudo-código associado a todo este procedimento, com alguns comentários que facilitam a sua compreensão.

```
init(app) {
    app.get("/", (req,res) => this.index.call(this, req, res));
    app.post("/search", (req,res) => this.getData.call(this, req, res));
    app.get("/search/:orcid",  (req,res) => this.search.call(this, req, res));
}

index(req, res) {
    // render da página "index"
}

getData(req, res) {
    let orcid = req.body.orcid;
    res.redirect("/search/" + orcid);
}

search(req, res) {
    let orcid = req.params.orcid;
    var t0 = performance.now();

    //data.init para ir recolher os dados associados a um ORCID
    data.init(orcid, function(dados) {
        var t1 = performance.now();
        var time = ((t1-t0)/1000).toFixed(2);
        console.log(time+"s");
        // render da página "search"
    });

}
```

Ao verificar o método *search*, verifica-se a invocação do *data.init* que realiza a recolha de dados, invocando outro método responsável por esta tarefa (ficheiro "index.js"do módulo pasta **data**):

```
let CollectData = require("./collectdata.js");

module.exports.init = function(orcid, cb) {
    return (new CollectData().getDataFromAPI(orcid, cb));
}
```

O método *getDataFromAPI* recebe como parâmetro o ORCid pretendido e um callback associado, para permitir a sua execução sequencial quando invocada antes do render. Como podemos

observar no código em baixo, o método realizar um pedido *request* para obter o xml associado ao ORCid. Obtendo este resultado, emite um sinal ("update") para prosseguir para o *parsing* do ficheiro. Ao longo de todo o *parser*, os dados são inseridos num *array* de publicações, que será posteriormente retornado através do callback.

```
getDataFromAPI(orcid, cb) {
        var publications = [];
        request.get(url, function (error, response, body) {
                if (!error && response.statusCode == 200) {
                        xmltxt.data = body;
                        xmltxt.emit('update');
                }
        });

        xmltxt.on('update', function () {
                parseString(xmltxt.data, function (err, result) {
                        var group = result["activities:works"]["activities:group"];
                        for (var i1 = 0; i1 < group.length; i1++) {
                                var workSummary = group[i1]["work:work-summary"]
                                for (var i2 = 0; i2 < workSummary.length; i2++) {
                                        // object publication
                                        var publication = {
                                            title: "",
                                            type: "",
                                            local:"",
                                            year: "",
                                            eid: "",
                                            wos: "",
                                        issn: "" };

                                        // recolha de dados


                                        publications.push(publication);
                                }

                        }
                });
                cb(publications);
        });

}
```

4.2 Página Web

Através da utilização do *bootstrap* e do motor de templates *pug*, elaboraram-se duas páginas que serão geradas consoante os pedidos do utilizador. Em seguida apresentam-se duas imagens referentes ao *layout* de cada uma delas.

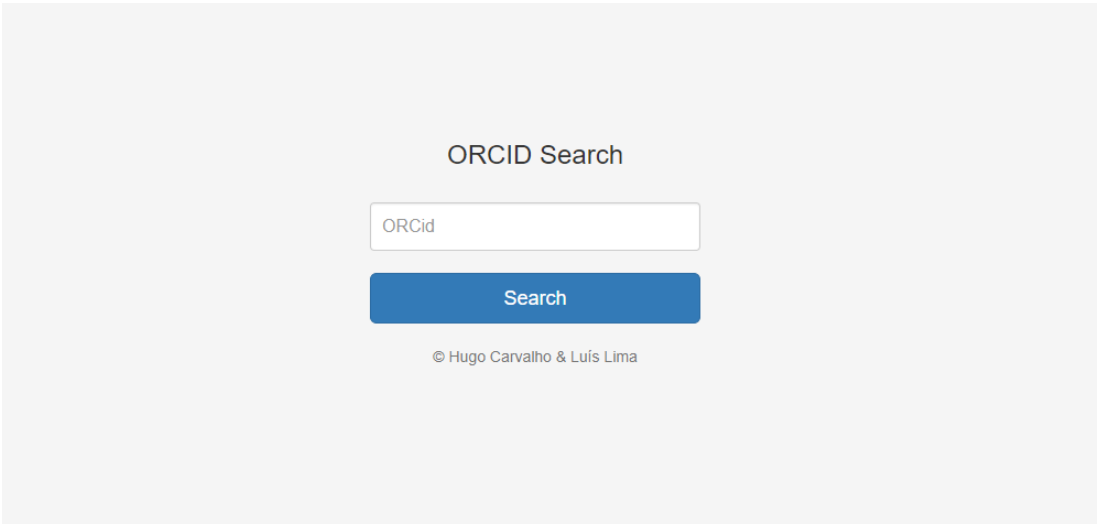
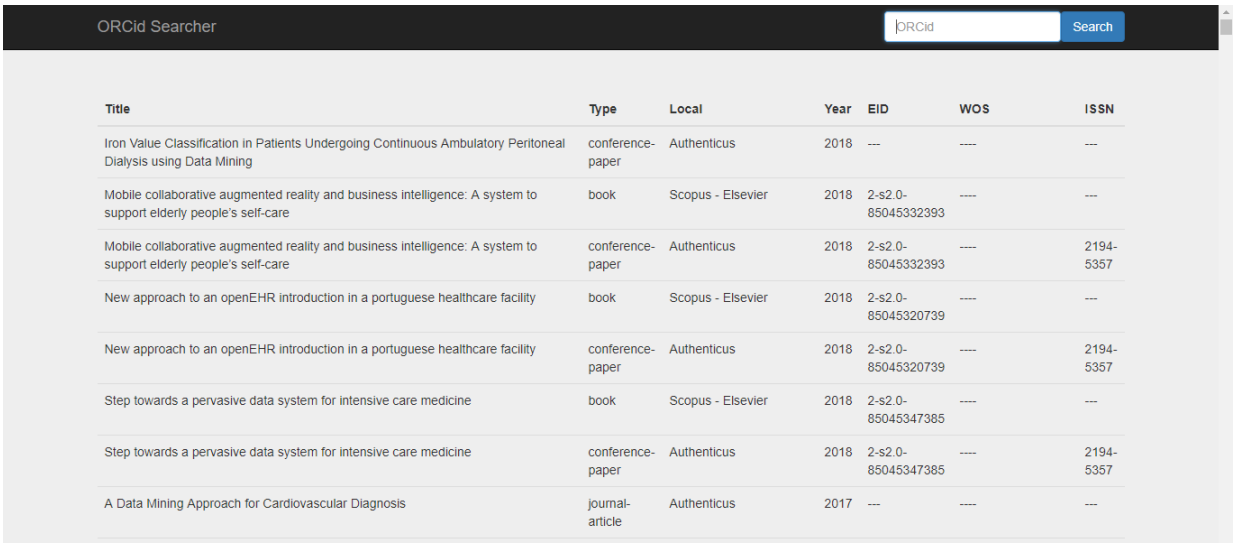


Figura 4: Aplicação Web - Página inicial



Title	Type	Local	Year	EID	WOS	ISSN
Iron Value Classification in Patients Undergoing Continuous Ambulatory Peritoneal Dialysis using Data Mining	conference-paper	Authenticus	2018	---	----	---
Mobile collaborative augmented reality and business intelligence: A system to support elderly people's self-care	book	Scopus - Elsevier	2018	2-s2.0-85045332393	----	---
Mobile collaborative augmented reality and business intelligence: A system to support elderly people's self-care	conference-paper	Authenticus	2018	2-s2.0-85045332393	----	2194-5357
New approach to an openEHR introduction in a portuguese healthcare facility	book	Scopus - Elsevier	2018	2-s2.0-85045320739	----	---
New approach to an openEHR introduction in a portuguese healthcare facility	conference-paper	Authenticus	2018	2-s2.0-85045320739	----	2194-5357
Step towards a pervasive data system for intensive care medicine	book	Scopus - Elsevier	2018	2-s2.0-85045347385	----	---
Step towards a pervasive data system for intensive care medicine	conference-paper	Authenticus	2018	2-s2.0-85045347385	----	2194-5357
A Data Mining Approach for Cardiovascular Diagnosis	journal-article	Authenticus	2017	---	----	---

Figura 5: Aplicação Web - Página de Resultados

5 Análise de Desempenho

Com o intuito de analisar o desempenho do sistema implementado, foi utilizado o *package* perf_hooks que permite efetuar registos de tempo. Neste sentido, realizaram-se 10 testes utilizando o ORCid fornecido no enunciado: "0000-0003-4121-6169". Em seguida apresenta-se um gráfico de variação temporal destes registos.

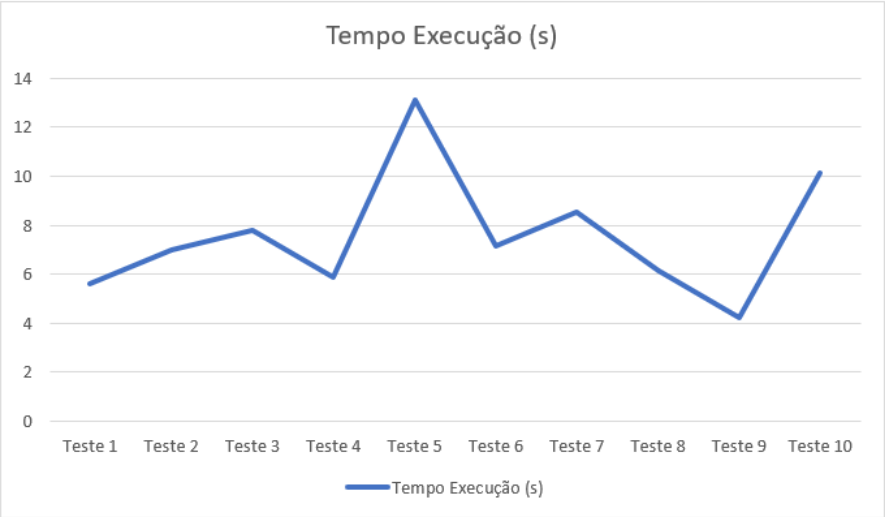


Figura 6: Análise de Performance

Como podemos observar, o pico máximo de execução situa-se em cerca de 13s enquanto o menor regista cerca de 4s. Apesar destes valores serem um pouco elevados para uma simples consulta à API, estes seriam esperados uma vez que engloba todo o processo de *request*, *parsing* e envio de dados para a aplicação. Deste modo, os valores apresentados são bastantes satisfatórios tendo em conta o objetivo proposto.

Caso o objetivo fosse reduzir o tempo de execução, uma solução que o grupo pensou em implementar passava por efetuar um pequeno *cron job* que efetuasse o acesso à API e povoasse uma base de dados local. Assim, quando existisse um pedido no *browser*, o ORCID Searcher realizava a consulta de dados à base de dados e não diretamente à API.

6 Conclusões e Trabalho Futuro

Terminada a realização do projeto, é então possível concluir que todo o processo desde o momento inicial foi crucial para o obtenção de resultados concretos e de extremo interesse para a implementação do sistema ORCiD.

Inicialmente foi importante perceber toda a parte teórica que este sistema tem envolvido para que as fases posteriores fossem realizadas de uma forma mais simples e com o objetivo pretendido. De seguida, foi necessário perceber o tipo e forma dos dados com que estávamos a lidar para que seja criada uma estratégia/arquitetura para a resolução do problema. Usamos então o *Node.js* e um conjunto variado de *packages* que nos auxiliaram na criação do sistema.

Posto isto, passou-se a implementação do sistema, mais propriamente o funcionamento das rotas e a recolha e envio dos dados entre os vários sistemas. Foram renderizadas duas páginas, uma de pesquisa do ORCid pretendido e ainda outra para apresentar os resultados conforme o ORCid escolhido anteriormente. Estas duas páginas foram criadas usando uma framework web, o *boostrap*, e o motor de templates *pug*.

Em suma, o resultado foi positivo. A implementação deste sistema foi conseguida e todo o trabalho foi recompensado com o resultado, o que leva a que o grupo considere que realizou um trabalho bastante bom, uma vez que compreendeu e implementou todos os conceitos e metodologias lecionadas.

Referências

[1] http://repositorium.sdum.uminho.pt/bitstream/1822/37437/4/ConfOA2015_posterA0_RepositoriUM.pdf

[2] https://jref2015.sciencesconf.org/data/pages/20150703_Couperin_Brown.pdf