INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Discovering Badges

LEANDRO DANIEL PINTO GOMES
Outubro de 2016

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Discovering Badges

Leandro Daniel Pinto Gomes

1100624

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientadora: Isabel de Fátima Silva Azevedo

Porto, Outubro 2016

Resumo

A evolução tecnológica da última década generalizou o uso da internet para a substituição ou complemento de múltiplos processos de aprendizagem, formais ou informais, e evidenciou novas perspetivas para o desenvolvimento de soluções com altos níveis de disponibilidade, escalabilidade e flexibilidade. Por outro lado, a certificação de conhecimentos e competências não mais está confinada a certas instituições ou universidades, quebrou as barreiras físicas e difundiu-se pelo mundo da internet ao ponto de se falar em certificação digital de competências. No contexto dos *Open Badges* esta generalização levou à necessidade da existência de mecanismos que permitam aos seus utilizadores o acesso a informação personalizada e relevante.

Nesse sentido, foi realizado um estudo aprofundado e avaliadas as diferentes perspetivas e abordagens aos problemas existentes do ecossistema dos *Open Badges*, mais concretamente na descoberta de oportunidades de experiências de aprendizagem. Após investigação, foi introduzido o *standard* xAPI e a sua potencial associação com os *Open Badges*. Essa associação revelou-se a base para o desenho, conceção e implementação de uma solução que se revelou adequada ao problema proposto. É proposta uma arquitetura por microserviços com componentes independentes e reutilizáveis, salientando-se ainda o suporte total a integração com sistemas externos com vista ao grande objetivo do projeto, a interoperabilidade.

Os testes efetuados à aplicação e a avaliação da solução permitem afirmar o resultado deste projeto como um contributo científico e técnico relevante para a área deste trabalho de mestrado.

Abstract

The technological developments of the past decade widespread the use of the Internet for the replacement or addition of multiple learning processes, formal or informal, and highlighted new perspectives for the development of solutions with high levels of availability, scalability and flexibility. On the other hand, the certification of knowledge and skills is no longer confined to certain institutions or universities, breaking the physical barriers and spreading throughout the world of the Internet to the point of talking about digital competences certification.

In the Open Badges context, this generalization led to the need of mechanisms that allow their users access to personalized and relevant information. Having this has our focus, it was conducted a serious study and evaluated the different perspectives and approaches to the problems of the Open Badges ecosystem, specifically the discovery. After the investigation, the standard xApi and its potential association with the Open Badges was introduced. This combination proved to be the basis for the design and implementation of a suitable response to the problem. The proposed architecture was a microservices architecture with independent and reusable componentes, with fully external systems integration capability for the ultimate goal of the project, interoperability.

The tests performed on the implementation and evaluation of the solution allow us to state the outcome of this project as a scientific contribution and relevant technical to the area of this master's work.

Keywords: *Open Badges*, *Digital Badges*, OBI, *xAPI*, *LRS, Microservices*, *Software Engineering*

Agradecimentos

Este trabalho foi o culminar de um ciclo de 6 anos de estudo, no fundo o ultrapassar de mais uma etapa na minha vida académica e pessoal. As pessoas que me rodeiam no dia-a-dia e que, de alguma forma, me ajudaram durante este longo período merecem sem dúvida o meu reconhecimento.

Um agradecimento especial à minha família, por todo o apoio e companhia nas inúmeras tardes e noites de trabalho que, sem dúvida, permitiram a entrega desta dissertação dentro do prazo estabelecido.

Agradeço também à minha orientadora, Isabel de Fátima Silva Azevedo, pelo apoio durante a realização da dissertação e por contribuir com opiniões e ideias que me permitiram melhorar o trabalho realizado.

À instituição de ensino, pela oportunidade concedida e por garantir todos os meios que possibilitaram a realização desta dissertação.

Por fim, a todas as pessoas que de uma maneira ou de outra contribuíram para que esta dissertação fosse concluída. O meu sincero agradecimento a todos.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

AHP	*Analytic Hierachy Process*
API	*Application Programming Interface*
CSS	*Cascading Style Sheet*
HASTAC	*Humanities, Arts, Science and Technology Alliance and Collaboratory*
HTTP	*Hypertext Transfer Protocol*
IIS	*Internet Information Services*
JSON	*Javascript Object Notation*
JSON-LD	*Javascript Object Notation – Linked* Data
LMS	*Learning Management System*
LRS	*Learning Record Store*
MOOC	*Massive Online Open Course*
OBI	*Open Badges Infrastructure*
PNG	*Portable Network Graphics*
REST	*Representational State Transfer*
SWOT	*Strengths, Weaknesses, Opportunities and Threats*
UML	*Unified Modelling Language*
URI	*Uniform Resource Identifier*
URL	*Uniform Resource Locator*

1 Introdução

No próximo capítulo o leitor terá oportunidade de tomar conhecimento do enquadramento do projeto, dos objetivos propostos, da abordagem utilizada bem como da estrutura geral deste documento.Enquadramento

1.1 Enquadramento

O ambiente que nos rodeia está repleto de informação e a utilização massiva da Internet e das tecnologias de informação possibilita o acesso, em tempo real, aos mais diversos conteúdos e áreas do conhecimento que até então podiam estar inacessíveis. Foi perante esta realidade no contexto educativo que se desenvolveu o conceito dos *digital badges* e, mais concretamente, dos *Open Badges*. Os *Open Badges* baseiam-se num conceito existente em alguns *websites* como *stackoverflow.com*, que é o reconhecimento de competências. Uma das características mais relevantes é o facto de certificarem a aprendizagem formal ou informal e descreverem de que forma foi alcançada. Organizações como a NASA, Disney-Pixar, Intel aderiram aos *Open Badges* de tal forma que são emissores e *displayers* dos mesmos.

A natureza da certificação e validação dos *Open Badges*, impôs uma especificação rigorosa dos metadados (Alliance 2016b) que devem fazer parte da estrutura de dados de um *badge*, de forma a este ser compatível com a especificação técnica da Infraestrutura dos *Open Badges*. Esta infraestrutura suporta a emissão, coleção e *display* dos *badges* e foi lançada em março de 2013 em conjunto com a sua especificação. Após o lançamento e apesar de algumas resistências à adoção e preocupações com a credibilidade dos *badges*, têm vindo a ser cada vez mais aceites e a ter maior adesão por parte de todos os *stakeholders*. Um dos desafios que ainda está por responder, diz respeito à dificuldade em encontrar *badges* que sejam relevantes para o utilizador construir o seu percurso de conhecimento baseado nos interesses pessoais do indivíduo e das competências a adquirir. Um primeiro passo foi feito nesse sentido, tendo sido desenvolvido um protótipo denominado *Mozilla OB Discovery* (Mozilla 2016a). Com base nessa primeira abordagem de resposta ao problema, foi desenvolvido um estudo aprofundado sobre o ecossistema dos *Open Badges* e das perspetivas para o futuro, que culminou no

desenvolvimento de um protótipo modular que possibilitasse ao utilizador uma forma de encontrar aquilo que procura no que concerne as experiências de aprendizagem, formais ou informais.

A proposta de valor é o elemento base na modelação do negócio e revela-se essencial quando se pretende saber como e quando o negócio vai gerar receita, identificar parceiros, a natureza das principais operações e como e de que forma se vão capturar e reter os clientes. O relacionamento com os clientes é um ponto essencial para a estratégia de marketing, por isso revela-se particularmente importante o estabelecimento de uma relação de proximidade com os clientes, com vista a auxiliar o desenvolvimento ou o melhoramento dos processos e especificações no ecossistema da emissão e angariação de *Open Badges*. Do ponto de vista da análise de negócio do projeto, o cenário de negociação será o integrado, com o objetivo de plena cooperação na persecução de objetivos comuns.

1.2 Objetivos

Com este trabalho pretende-se efetuar um estudo de caracterização da área de domínio dos *Open Badges* e as limitações relacionadas com a descoberta de *badges*. O trabalho a desenvolver nesta tese visa também propor e desenvolver uma solução flexível que permita a pesquisa, partilha e descoberta de *badges* para além do ecossistema tradicional dos mesmos, bem como a exploração de diferentes mecanismos na área de recomendações de *badges*. Outro objetivo está relacionado com a disponibilização de funcionalidades relacionadas com a construção de percursos de aprendizagem. O estudo efetuado irá culminar com a implementação prática de um sistema de informação que responda à problemática levantada, com base nos pressupostos de ser um sistema compatível com a infraestrutura existente e que garanta a integridade dos metadados dos *badges* e a interoperabilidade entre sistemas.

1.3 Abordagem metodológica

O trabalho desenvolvido foi constituído por três fases principais: estudo do problema, conceitos e tecnologia, desenvolvimento de uma solução e, por fim, avaliação da solução apresentada. Na fase de estudo do problema foram levantados todos os conceitos associados aos *Open Badges*¸ bem como do contexto em que se inserem, o valor, a tecnologia envolvida, o ecossistema e as novas possibilidades que se avizinham.

Após o estudo inicial foi desenhada e implementada uma solução que respondesse ao problema da descoberta de *Open Badges,* recorrendo a padrões de Engenharia no contexto atual das aplicações *web*.

Por último, foi efetuada uma análise à solução proposta no sentido de avaliar o seu valor no contexto do problema em questão.

1.4 Estrutura

Neste primeiro capítulo, *Introdução*, é feita uma breve apresentação do tema da tese, do seu enquadramento temático e dos objetivos principais.

No segundo capítulo, *Estado da Arte*, são introduzidos alguns conceitos sobre os *Open Badges*, é discutida a sua importância e valor, bem como tecnologias e soluções existentes.

No terceiro capítulo, *Valor, Ameaças e Oportunidades*, é efetuada uma análise SWOT[1] ao ecossistema dos *Open Badges* bem como análise do valor subjacente à proposta de solução do problema deste projeto.

No quarto capítulo, *Arquitetura da Solução*, são descritos todos os conceitos, padrões, orientações e decisões tomadas para a arquitetura, bem como os componentes mais relevantes.

No quinto capítulo, Requisitos, são apresentados os requisitos funcionais e não funcionais da solução desenvolvida bem como das entidades envolvidas.

No sexto capítulo, *Desenho da Solução*, são relatados os conceitos do domínio do problema, é descrito o comportamento esperado da solução no que se refere aos requisitos funcionais e não funcionais, bem como das orientações e estratégias de implementação dos diferentes componentes. Neste capítulo são também descritas as abordagens de referência para avaliação da solução proposta.

No sétimo capítulo, *Implementação,* é descrita a implementação dos casos de uso, abordando as tecnologias utilizadas, a metodologia de desenvolvimento e ilustrando a apresentação final das interfaces para com o utilizador.

No oitavo capítulo, *Avaliação da solução*, revela-se de que forma é que se pretende avaliar a solução desenvolvida e quais as métricas e metodologias envolvidas.

O nono capítulo, *Conclusões*, descreve o balanço final deste estudo e traçam-se os passos para um trabalho futuro.

[1] SWOT - Strengths, Weaknesses, Opportunities e Threats

1.5 Convenções de escrita

O autor optou por não traduzir alguns termos que considera essenciais manter na língua inglesa, para melhor compreensão dos mesmos no contexto dos *Open Badges*. Exemplos desses termos são *earner*, *issuer* ou o próprio termo *badge*. Na utilização de termos em inglês é usada a notação em itálico com fonte 11 Calibri.

Exemplos de extratos de código são apresentados noutra fonte, nomeadamente na fonte 9,5 Consolas e com 1cm de indentação à direita da margem esquerda. Relativamente a tabelas, estas são apresentadas num formato padrão com os cabeçalhos a negrito.

2 Estado da arte

Esta secção tem por objetivo o levantamento do estado da arte da área de estudo em causa, termos e conceitos relacionados, bem como descrever o grau de desenvolvimento da mesma, e identificar desafios para o futuro deste ecossistema.

2.1 Contexto

Nos dias de hoje, a *web* permite o acesso, em tempo real, aos mais diversos temas e conteúdos, revolucionando a aprendizagem de tal forma, que esta ocorre agora, em qualquer lugar e a qualquer momento. Num estudo recente (Mozilla 2011), grande parte dessa mudança é devida ao facto de o nosso mundo ser muito diferente do que era quando o atual sistema educativo foi desenvolvido e padronizado. Com a *web* e os seus princípios fundamentais de abertura, universalidade e transparência, foram transformando os caminhos que o conhecimento é feito, partilhado e valorizado criando oportunidades de aprendizagem mais relevantes. Estas características da *web* permitem o aumento do acesso a informação, ao mesmo tempo que nos presenteia com novas formas de aprender e novas competências a alcançar.

Aprender não ocorre somente nas escolas, mas estende-se através de múltiplos contextos, experiências e interações não sendo um conceito isolado ou individual, mas inclusivo, social, informal, participativo, criativo e durante toda a vida (Mozilla 2011). O mesmo autor refere que os cursos não estão apenas confinados às salas de aula ou universidades, mas a iniciativas de aprendizagem livres, tais como o *MIT OpenCourseWare*[2], *Peer-2-Peer University*[3] (P2PU) e *OERCommons*[4]. Estes projetos são inteiramente gratuitos para o utilizador e são uma alternativa à educação tradicional. Apesar de a aprendizagem, formal ou informal, poder ocorrer à margem do contexto educativo tradicional, estabelecimento de ensino ou de

[2] http://ocw.mit.edu/index.htm
[3] https://www.p2pu.org
[4] https://www.oercommons.org

formação especializada, o reconhecimento de competências é ainda muito complicado de assegurar. É necessário arranjar formas de garantir o reconhecimento e a certificação das várias competências que um indivíduo pode adquirir na sua vida, profissional ou pessoal (Mozilla 2011). A aprendizagem noutros contextos que não o educativo deve ser um processo bidirecional, de aprendizagem mas também de partilha constante. Não pode ser vista simplesmente como consumo de informação, mas em vez disso os alunos devem ser participantes ativos e produtores, num processo de aprendizagem contínuo, orientado para os interesses pessoais de cada um (Mozilla 2011).

2.2 Badges

Os *Badges*, como o nome o sugere, são acessórios que pretendem demonstrar algum feito ou conquista. No contexto da *internet*, eles têm o mesmo significado que os *badges* físicos, no entanto, são apenas uma representação portável e atualizada, muito comum a outras áreas de negócio.

2.2.1 Badges digitais

Conforme é descrito no relatório sobre o potencial e valor na utilização de *badges* digitais na educação de adultos (Finkelstein et al. 2013), um *badge* digital é uma nova forma de capturar e comunicar o que um determinado indivíduo sabe e consegue demonstrar. Consiste, como a própria designação o sugere, numa representação visual de certificação de competências e consegue oferecer ao mesmo tempo evidências de suporte às mesmas.

Os *badges* digitais não são uma ideia ou um conceito recente, são usados por grupos de escuteiros para reconhecimento de competências e feitos dos seus membros há diversos anos (Kriauciunas, Ragauskas 2013). Os mesmo autores referem que os próprios *web developers* usam emblemas digitais para encorajar os utilizadores de internet a adotarem comportamentos adequados na utilização das aplicações *web*.

Segundo é citado no relatório da Educause[5] (D. T. Hickey et al. 2015), o ano de 2011 foi o ano em que os emblemas digitais começaram a revolucionar a forma como a aprendizagem e a correspondente certificação são reconhecidas. O mesmo relatório refere que os emblemas digitais podem conter reivindicações específicas e provas detalhadas que os suportam. Essas alegações e provas podem incluir *links* para informações adicionais, tais como o trabalho do aluno, assinaturas digitais, perfis institucionais, informações sobre o curso e colegas.

Os *badges* podem representar diferentes níveis de trabalho e envolvência, desde os mais elementares aos mais específicos ou globais. Surgiram como uma aposta na certificação de

[5] Educause – Organização sem fins lucrativos dos Estados Unidos da América dedicada ao desenvolvimento de tecnologias da informação no ensino superior

competências, nomeadamente nos programas de educação básica na educação de adultos, uma vez que não tendo credenciais formais como diplomas ou certificados, conseguem desta forma obter competências funcionais relevantes para os seus trabalhos (Kriauciunas, Ragauskas 2013).

Apesar de o conceito de *badges* digitais ser recente, temos o exemplo da Wikipedia, que oferece *badges* aos colaboradores pelo seu trabalho árduo e diligência (Doherty & Sharma 2015).Um outro exemplo é o Foursquare[6], que utiliza com sucesso *badges* para estabelecer objetivos, motivar comportamentos, representar conquistas e comunicar o sucesso em muitos contextos (Frith 2013).

2.2.2 Open Badges

2.2.2.1 Definição

Um *badge* digital é uma representação *online* de uma competência adquirida. Os *Open Badges* são na prática, uma evolução dos *badges* digitais uma vez que se propõe a certificar competências e aptidões dos alunos (*learners* ou *earners*) através de organizações creditadas e possuem informação suficiente para acesso futuro e validação. O objetivo dos *Open Badges* é ajudar as pessoas de todas as idades a ganharem e a exporem as competências adquiridas, desbloqueando novas oportunidades profissionais e educacionais (Mozilla 2015).

Como é possível verificar no *website* da Mozilla MacArthur Foundation (Mozilla MacArthur Foundation 2016), os *Open Badges* "*allows you to verify your skills, interests and achievements through credible organizations. And because the system is based on an open standard, you can combine multiple badges from different issuers to tell the complete story of your achievements — both online and off. Display your badges wherever you want them on the web, and share them for employment, education or lifelong learning".*

Apesar da ideia subjacente aos *Open Badges* não ser recente, a formulação dos *digital badges* para reconhecimento de méritos, competências e conquistas nos mais variados contextos de aprendizagem é realmente inovador e potenciador de um ecossistema ímpar de partilha e aprendizagem. O projeto *Open Badges* da *Mozilla* foi lançado na sua primeira versão estável no ano de 2013, sendo dessa forma uma tecnologia e um paradigma muito recente e que está apenas a dar os primeiros passos.

Segundo é citado no artigo de opinião de Ian Quillen (Ian Quillen 2013) "*I don't see badges replacing degrees as something that is going to happen tomorrow. But I see it as more incremental*", o que vai de encontro às ideias de base deste ecossistema, ser gratuito e aberto a todos no sentido da convergência na estruturação e emissão de *badges*. Desta forma, também é possível reconhecer de uma forma inequívoca a autenticidade quer para as entidades emissoras bem como para os utilizadores que os recebem.

[6] https://pt.foursquare.com/

No que se refere às suas características base, o autor do relatório (Finkelstein et al. 2013) identifica algumas das características dos *badges* digitais que são únicas, desde a ligação persistente às fontes emissoras que podem desta validar a competência, assim como o de representarem um *standard* para representações de competências e serem portáveis e disponíveis para apresentação em conjunto com *badges* de fontes completamente diferentes. Segundo é reforçado no artigo sobre *badges* e ética (D. T. Hickey et al. 2015), uma das características de destaque é o facto de os *Open Badges* serem interoperáveis de tal forma que podem circular sem dificuldade em qualquer tipo de rede digital.

No que se refere à sua constituição, os Open Badges possuem informação necessária para a sua validação, autenticidade, fonte emissora e valor. Destacam-se assim os dados do *earner*, do *issuer*, o critério de obtenção do *badge*, evidência (representação autêntica ou conexão para atestar trabalho realizado), data de emissão, data de expiração e um certificado de validação (conexão para validação do *open badge*).

No que se refere às suas características, os *Open Badges* são, nomeadamente:

- Livres e públicos: Não são proprietários. É software livre e público e segue um *standard* técnico que qualquer organização pode usar para criar, emitir e verificar *badges* digitais.

- Transferíveis: É possível amealhar *badges* de várias fontes, *online* e *offline*, e passíveis de partilha em redes sociais, *websites* de emprego e muitos outros.

- Passíveis de conjugação: Independentemente da fonte emissora, podem ser conjugados com outros de forma a demonstrarem um percurso, histórico ou não, de competências e aptidões.

- Baseados em evidências: Contém informação complementar em metadados que conectam o *issuer*, os critérios e as provas de verificação e validação.

Tendo em conta os dados existentes nos *Open Badges*, a natureza digital dos mesmos torna-os gratuitos, possíveis de serem descobertos através de um simples método de pesquisa, apresentados nos mais diferentes ambientes em oposição com os certificados ou diplomas tradicionais (Finkelstein et al. 2013).

Como é mencionado no relatório sobre *badges* e ética (D. Hickey et al. 2015), não é apenas a realização de competências que podem ser recompensadas através da emissão de *Open Badges*, eles podem ser utilizadas para premiar e reforçar comportamentos positivos. Os mesmos autores reforçam que como entidade emissora, num contexto escolar tradicional, seria possível atribuir *Open Badges* aos alunos que, por exemplo, fizessem os trabalhos de casa regularmente ou mesmo a professores ou outros colaboradores que tivessem uma postura exemplar no desenrolar da sua profissão.

2.2.2.2 OBI

"*Learning today happens everywhere. But it's often difficult to get recognition for skills and achievements gained outside of school. Mozilla's Open Badges project is working to solve that problem, making it easy to issue, earn and display badges across the web. The result: recognizing 21st century skills, unlocking career and educational opportunities, and helping learners everywhere level up in their life and work."* ((Mozilla 2011))

De forma a concretizar o potencial do ecossistema de emblemas digitais, é fundamental que as interações que envolvam emblemas digitais sejam *open source* e não proprietárias (Pearson Learning Solutions 2013). No final de 2011, a Mozilla, a HASTAC e a fundação MacArthur uniram-se a uma comunidade alargada de colaboradores para desenvolver um standard técnico, *open source*, que permitisse a qualquer organização a criação, emissão, gestão e validação de emblemas digitais. Em março de 2013 foi lançada a primeira versão da especificação dos *Open Bagdes* e *Open Badges Infrastructure (OBI)*.

A infraestrutura dos *Open Badges* é baseada numa especificação e infraestrutura técnica, denominada OBI (Jovanovic & Devedzic 2014). A OBI define o *issuer* do *badge* como um *provider* de aprendizagem que atribui um *Open Badge* pela conclusão de uma determinada tarefa (tarefas) e/ou alcance de um determinado objetivo (metas). O emissor cria os critérios que o *earner* necessita de satisfazer de modo a obter o *badge*, podendo combinar *Open Badges* de diferentes emissores, exibi-los na *web*, e partilhá-los em redes de emprego ou educação. A OBI disponibiliza um conjunto de APIs *open source* que permitem a integração ou modificação para uso próprio, dos serviços de *Open Badges* com aplicações existentes, *websites* e redes sociais.

A infraestrutura dos *Open Badges* suporta uma nova abordagem para a avaliação do conhecimento e reconhecimento, dando a potenciais empregadores, grupos profissionais e comunitários, escolas, instrutores e alunos um quadro mais completo de *earners* de *badges* com determinados conhecimentos, competências e aptidões.

Segundo o artigo de investigação sobre novos meios de motivação e reconhecimento da aprendizagem (Jovanovic & Devedzic 2014), verifica-se os *Open Badges* permitem que os alunos possam verificar suas competências, interesses e realizações através de organizações credíveis. A mesma fonte refere que as informações sobre a organização emissora do *badge*, os critérios de emissão, a data em que foi emitido, e as provas da realização estão ligados ao arquivo de imagem do emblema, embutidos em metadados para acesso e revisões futuras. Essa informação visa valorizar a conquista dos *Open Badges* e como a informação é codificada num formato padrão (compatível com a OBI), facilita significativamente a transferência de valor nos diferentes contextos.

Segundo o relatório relacionado com o potencial e valor do uso de *badges* digitais na idade adulta (Finkelstein et al. 2013), o componente de imagem de um *badge* digital é um *proxy* para um conjunto de informações subjacentes que transmitem a história da conquista. Segundo os mesmos autores, os componentes de base de um *badge* incluem grande parte dos mesmos elementos que se esperam encontrar numa moeda oficial e de que precisamos para determinar a sua validade, autenticidade, origem e valor, tais como:

- Recipiente: a quem se refere a conquista
- *Issuer*: O indivíduo ou organização que assume a responsabilidade pela emissão do *badge*.
- Critério e descrição: O que o *earner* necessitou de demonstrar para adquirir o badge.
- Evidência: Uma representação autêntica ou conexão com o trabalho realizado
- Data de emissão: Data precisa de quando o *badge* foi emitido
- Data de expiração: data a partir da qual as credenciais concedidas não são mais válidas
- Certificado ou afirmação: uma conexão com um formulário oficial de verificação e validação

2.2.2.3 Funcionamento

O projeto *Open Badge* fornece uma definição técnica de como os *Open Badges* devem ser criados, guardados e partilhados, promovendo uma aproximação para a interoperabilidade dos próprios badges (D. Hickey et al. 2015). A Figura 1 representa o processo de emissão, recolha e partilha de badges por um individuo.

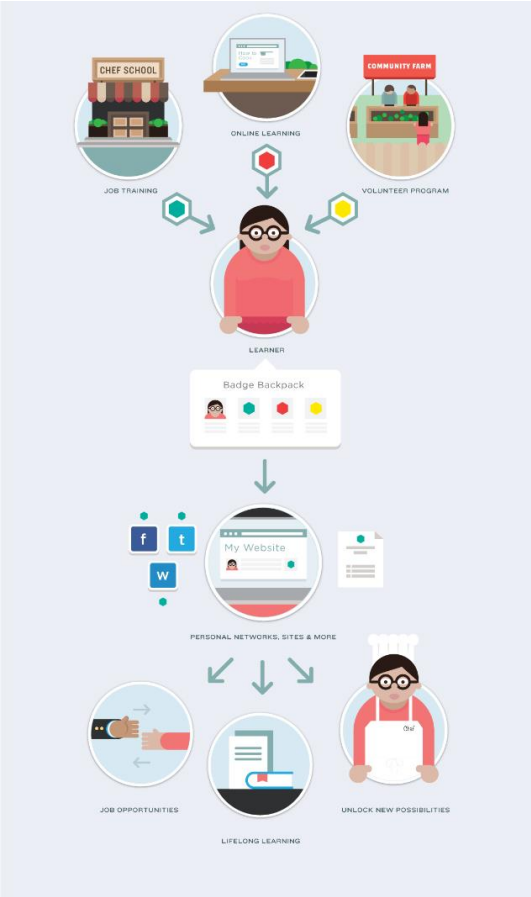

Figura 1 – Esquema representativo do Processo de emissão, recolha e partilha de *badges* (Mozilla MacArthur Foundation 2016)

Com base na análise da Figura 1, é possível destacar o componente *Backpack* que é um repositório de *Open Badges* amealhados pelo *earner*. Esta é uma plataforma web (fase Beta) disponibilizada pela Mozilla para dar suporte aos alunos (*learners* ou *earners*) na recolha de *badges* das mais variadas fontes emissoras (D. Hickey et al. 2015). O *Backpack* disponibiliza aos seus utilizadores um repositório centralizado para armazenamento dos seus *badges*.

Uma das características importantes a destaca prende-se com o facto de a infraestrutura OBI permitir que os utilizadores tenham outros tipos de repositório que não o *Mozilla Backpack*, tornando possível a coexistência de vários repositórios ao mesmo tempo.

O processo de criação, validação e publicação de um *badge* pode ser representado na seguinte sequência.

O emissor cria um *badge* e torna-o disponível para a comunidade de *earners* no seu *website*. Quando algum *earner* for *premiado com o badge*, este envia-o para o seu *backpack*. O *badge* torna-se portátil através da Issuer API que irá questionar o *earner* para adicionar o emblema no seu *backpack*. Se por acaso o emissor quiser armazenar o *badge* fora da infraestrutura OBI, é possível enviá-lo para o *Baking Service* da Mozilla onde os metadados são incorporados nos ficheiros de formato PNG. Os *displayers* solicitam *badges* ao *backpack* do utilizador. Os *badges* que façam parte de um grupo público no *backpack* podem ser descobertos através do e-mail por meio da Issuer API, bem como pode ser o próprio utilizador a partilhar esses *badges* através de *websites* ou redes sociais.

2.2.2.4 Intervenientes

No que toca a intervenientes do sistema, o artigo (D. Hickey et al. 2015) descreve o modelo e este é intencionalmente simples, empregando três principais partes interessadas:

- *Issuers*, que criam os *Open Badges* para a realização de tarefas e atividades, bem como emitem *badges* para reconhecerem competências.
- *Learners* ou *earners*, que são indivíduos que se propõe a completar as tarefas, demonstram competências e consequentemente amealham os *Open Badges.*
- Revisores, que são as entidades ou indivíduos que atuam como *endorsers* dos *badges* e asseguram a integridade dos mesmos.
- *Displayers*, que disponibilizam uma plataforma aos *earners* para exporem os seus *Open Badges* nos mais variados contextos.

2.3 Contexto Educativo

Os *badges* oferecem transparência à avaliação das realizações individuais e estão disponíveis para investigação. Desta forma, os *badges* digitais com os seus metadados fazem parte do

currículo profissional das pessoas e documentam aprendizagens, competências adquiridas e experiências num formato fácil de partilhar (Gamrat & Zimmerman 2015).

Os *badges* digitais têm sido usados em redes sociais e jogos *online* em cinco áreas-chave, nomeadamente: definição metas e promoção de *feedback* sobre a realização do objetivo, aprovisionamento de instruções sobre o que são possíveis atividades, construção da reputação de um utilizador com base em interesses, servir como um símbolo de *status* e conquistas, e mostrar afiliação com uma comunidade.

A maioria das escolas e universidades conseguem dizer o que os seus alunos fizeram, enquanto que os prestadores menos formais, muitas vezes, não o conseguem. Esta falta de especificidade e credibilidade significa que, a maioria dos programas de educação vão experimentar grandes transformações quando tentarem especificar a aprendizagem a ser reconhecida e como vão fornecer evidências de apoio à validação da aprendizagem (D. T. Hickey et al. 2015).

No artigo sobre aprendizagem colaborativa (Loi et al. 2015) é referido que no ambiente educacional, os educadores usam uma variedade de abordagens para fornecerem aos alunos uma educação teórica e crítica, bem como avaliam a competências adquiridas pelos alunos num curso. Segundo os mesmos autores, os *badges* digitais - bem como os emblemas no escutismo - são uma forma de expressar competências e podem ser utilizados dentro de um curso ou programa curricular como veículo de avaliação eficaz do trabalho do aluno.

No entanto, algumas pesquisas têm estudado a motivação na utilização de *badges*, o que mostra que as implementações bem-sucedidas de *badges* podem tem a ver com a forma como os alunos são motivados individualmente (Abramovich et al. 2013). A motivação, neste contexto, é explicitamente ligada a um conceito chamado *gamification*. *Gamification* é o uso de elementos de *design* de jogo num outro contexto (Deterding et al. 2011). A utilização de técnicas de *gamification* visam influenciar os desejos das pessoas pela competição, conquista, reconhecimento e expressão pessoal (Association for Project Management 2014). Na educação, *gamification* não é apenas uma forma de tornar a aprendizagem divertida, mas também de aumentar a motivação dos alunos para determinados tópicos ou questões. Quando os alunos participam num jogo, eles são motivados a terem sucesso porque o mesmo é divertido, desafiador e fornece estruturas de recompensa. No relatório sobre a descentralização de badges em contexto educativo (Santos et al., 2012), é referido que a aprendizagem baseada em jogos já existe há muito tempo, mas enfrenta ainda um desafio: os seres humanos sempre tiveram a capacidade de envolver e aprender através de jogos, mas a unidade natural de aprender através de jogos tem um significado desprezável, especialmente do ponto de vista os sistemas de educação formal.

A incorporação de funcionalidades de jogo em vários domínios não-jogo, tais como *marketing*, saúde e educação tornou-se cada vez mais reconhecido (Lee & Hammer 2011). Este fenómeno chamado *gamification* pode aumentar a motivação dos alunos e aumentar a autoaprendizagem, ajudando os alunos a ficarem entusiasmados e aderirem ao processo de aprendizagem global (Groh 2012). Elementos de jogo são usados em vários contextos. Empresas como a Samsung

atribuem *badges* para motivarem os seus funcionários e serviços como Foursquare atribuem *badges* para os utilizadores que façam *check-ins* em locais (Farber 2013).

Apesar das transformações serem necessárias, estas podem ser demasiado disruptivas. Se os *badges* digitais se tornarem amplamente utilizados, estas alterações irão ocorrer dentro das instituições. O grande problema com *badges* digitais, neste momento, é a perceção do seu valor percebido.

Os *badges* têm vantagens para a aprendizagem, tais como:

- Popularizar a aprendizagem e promover as competências de aprendizagem ao longo da vida.
- Permitir que possam ser usadas ferramentas para desenvolvimento das habilidades metacognitivas a fim de alcançar o sucesso em espaços formais e informais, dando valor ao que está sendo aprendido, suporte a conexões e desenvolvimento de estratégias de negociação e de moldar o ambiente de aprendizagem (Joseph 2012).
- Atuar como uma forma alternativa de avaliação, uma nova maneira de receber *feedback* com base em provas formativas e sumativas (Joseph 2012).
- Promover a motivação e inspiração dos indivíduos. Neste contexto, os *badges* podem expressar os valores de uma determinada comunidade, permitindo uma gratificação autodirigida dentro do grupo e incentivar as aprendizagens participativas (Joseph 2012).

Devido à natureza aberta e distribuída, certas comunidades de Cursos *Online* (MOOCs) estão atualmente a explorar os *Badges* como método de reconhecimento seguro da conclusão do curso. Em parte por estas razões, num relatório sobre o impacto esperado de uma série de tecnologias de aprendizagem, a Open University do Reino Unido concluiu que a utilização dos *badges* apresenta um elevado potencial (Glover & Latif 2013). Segundo os mesmos autores, algumas instituições de ensino superior já implementaram sistemas baseados nos *Open Badges*, tais como a plataforma *Passport* da Universidade de Purdue, representada na Figura 2.

Como é citado no artigo sobre a aplicação de *badges* nos MOOCs (Dona et al. 2014), os *badges* têm a capacidade de servir como um "meio de inspirar ensino e aprendizagem na era digital, confirmando realizações e validação de competências" e como uma excelente forma de motivar

Figura 2 – Representação da plataforma *Passport* (Dona et al. 2014)

os aprendizes como emblemas permitir para a definição de "normas claras que pode capacitar aprendizagem informal, e premiar e reconhecer alunos para a realização deste desafio".

Uma vez que os *Open Badges* podem ser utilizados como indicadores e validadores de realização do aluno, a área da avaliação é onde podem introduzir mudanças mais significativas. Dentro de um MOOC, os questionários e os testes são utilizados para verificar conhecimentos e competências como a memorização de temas enquanto que as tarefas mais complexas, como os trabalhos escritos, necessitam de outras formas de avaliação. Muitas plataformas de MOOC, incluindo o Coursera[7] continuam sem mecanismos de avaliação totalmente automatizados e insistem em usar um formulário de revisão por base humana no próprio processo de pontuação (Dona et al. 2014). No entanto, uma série de MOOCs, tais como os oferecidos por Coursera, tentaram preencher esta lacuna avaliação aproveitando o trabalho em grupo e *feedback* dos pares. Mesmo quando é realizada a *peer evaluation* esta carece de verificação institucional do próprio processo de aprendizagem (Dona et al. 2014). Este é o ponto onde os *badges* podem ajudar a colmatar as lacunas relacionadas com a avaliação, uma vez que podem complementar e apoiar as técnicas de avaliação formais, mantendo os padrões de qualidade através do envolvimento institucional no processo (Sandeen 2013).

Segundo o artigo sobre o alinhamento entre portefólios e os *badges digitais* (Doherty & Sharma 2015), a utilização de Open Badges tem o potencial para reforçar a aprendizagem nas vertentes de envolvimento, motivação e progresso assim como fornecerem evidências de aprendizagem a um nível mais granular. Um dos exemplos apontados pelo autor, é a possibilidade de capturar a aprendizagem "extracurricular" que não deixa de ser importante no processo de aprendizagem. Se pensarmos em *badging* em termos de micro certificação, então o sistema *Open Badges* oferece uma noção temporal da aprendizagem do aluno. Por exemplo, os *badges* digitais poderiam apoiar a introdução de atividades profissionais, através do fornecimento de um meio visível de demonstrar a progressão, passo a passo, para atingir o *badge*.

2.3.1 Projetos

Segundo (Jovanovic & Devedzic 2014), uma série de projetos e iniciativas adotaram os *Open Badges* como base tecnológica para motivar, reconhecer e validar aprendizagens.

No Reino Unido, o projeto DigitalMe OB[8] utiliza uma plataforma *online* para aumentar o valor dos *Open Badges* na aprendizagem e desenvolvimento dos talentos de jovens no contexto educativo, no emprego e na realização pessoal.

A plataforma Makewaves[9] é outro projeto de destaque na área dos *Open Badges*. Trata-se de uma comunidade de milhares de escolas e um ambiente de aprendizagem social onde os jovens

[7] https://pt.coursera.org/
[8] http://www.digitalme.co.uk/badgetheuk
[9] https://www.makewav.es/mwhq

aprendem juntos e partilham recursos. A aprendizagem é concebida com missões de aprendizagem onde as realizações dos alunos são reconhecidos e premiadas com *Open Badges*.

A própria União Europeia reconhece o potencial dos *Open Bagdes*, tendo aprovado e finciado um projeto denominado GRASS[10] que visa apoiar o desenvolvimento, avaliação e classificação das competências transversais dos alunos, tais como a capacidade de resolução de problemas, o pensamento crítico, a colaboração, a comunicação, entre outras.

Nos Estados Unidos da América destaca-se o projeto Mouse Squad[11] com o objetivo de melhorar os conhecimentos dos alunos nas áreas da comunicação e tecnologia digital. Os *Open Badges são utilizados* não só para o reconhecimento dos estudantes nos módulos de aprendizagem, mas também como forma de mapear os percursos dos alunos com possíveis trajetórias de aprendizagem (Ian O'Byrne et al. 2015).

2.3.2 Soluções existentes

Nesta secção, são apresentadas algumas soluções ou protótipos que integram *Open Badges*.

2.3.2.1 Mozilla Discover

O projeto *Mozilla Discover* é um protótipo que visa a conexão dos *Open Badges* com os seus interesses, educação, experiências e personalidade (Mozilla 2016a). Esta combinação permite ao utilizador identificar e construir um percurso profissional ou pessoal de sonho, no contexto da aprendizagem, formal ou informal. Permite mapear as competências e experiências necessárias para o atingir, como se fosse um quadro das competências atuais e as que precisam

Figura 3 – *Screenshot* de um possível percurso profissional na área da saúde com *Open Badges* (Mozilla 2016a)

[10] https://sites.google.com/site/llpgrassproject/
[11] https://mousesquad.org/

de ser adquiridas. Na Figura 3, está representado um *screenshot* da plataforma para um percurso profissional da área da saúde, com a identificação dos *badges* necessários para o atingir.

2.3.2.2 Open Badges Factory

A plataforma *Open Badge Factory*[12] é uma solução em *cloud* que permite centralizar a gestão de *Open Badges* e emiti-los para diversos sistemas. Permite também a certificação como *badge issuer* de formas a garantir a autenticidade e veracidade na emissão de *Open Badges*. É possível monitorizar e avaliar o valor dos *Open Badges*. Após receção dos *badges*, é possível gerar relatórios sobre a receção dos mesmos, utilizados, visualizados e avaliados pelos grupos destino. Compatível com dispositivos móveis, é possível emitir *badges* em qualquer lugar. Possui *plugins* para *displayer*/*issuer* tais como Moodle, *WordPress*.

2.3.2.3 Open Badge Passport

O *Open Badge Passport*[13] é um serviço vocacionado para os *earners*, disponibilizando a receção, arquivo, *display* e partilha dos *Open Bagdes*. Mais do que apenas um repositório, é um local onde os utilizadores podem construir um portefólio com os seus badges.

2.3.2.4 OER – Open Educational Resources

A plataforma Open Educational Resources[14] (OER) tem por objetivo disponibilizar materiais e conteúdos, sem custo associado e de aprendizagem. Ao contrário dos recursos fixos, direitos de autor, a OER tem sido o autor ou então algum dos seus colaboradores. Em alguns casos, é possível descarregar um recurso e partilhá-lo com colegas e alunos. Em outros casos, é possível editar um recurso existente e recoloca-lo na plataforma como recurso revisto/melhorado.

2.3.2.5 Open Badges Europe

O objetivo da plataforma *Open Badges Europe*[15], é chegar aos alunos, cidadãos, educadores, empregadores, autoridades públicas e os formuladores de políticas com o objetivo global de criar as condições para:

- Proporcionar o acesso sistemático para o reconhecimento de toda a aprendizagem, se a não-formal, informal e formal;
- Aumentar a transparência, credibilidade e qualidade do reconhecimento dos conhecimentos adquiridos;
- A capacitação dos indivíduos para mais relações mais equilibradas com instituições e autoridades;
- Criar novas oportunidades de emprego, inclusão social e aprendizagem para todos.

[12] https://openbadgefactory.com/

[13] https://openbadgepassport.com/en

[14] https://www.oercommons.org/

[15] http://www.openbadges.eu/

2.3.2.6 Widgets

Verificou-se a existência de diversos *plugins* e *widgets* para as mais diversas plataformas de desenvolvimento, tais como WordPress, Java, PHP, Django, Moodle, Mobile e outros.[16] Um dos *widgets* a destacar é a ferramenta *Open Badges* Me[17], que tem por objetivo disponibilizar ao utilizador uma ferramenta de edição muito completa e gratuita para criação de *Badges.*

2.4 Infraestrutura OBI

O projeto *Mozilla Open Badges framework*[18] (OBI) disponibiliza um conjunto de ferramentas para emitir, amealhar e apresentar *badges* pelas páginas web e redes sociais, através de uma infraestrutura partilhada, conforme é possível verificar na

Figura 4. O *design* desta *framework* visa tornar o sistema de *badging* mais flexível, de forma a permitir a representação de todo o ecossistema de aprendizagem e experiências *online* e *offline*.

O sistema foi concebido com dois objetivos principais, o de suporte a múltiplos emissores e ao mesmo tempo o de disponibilizar os badges emitidos em diversos ambientes *online* (websites ou redes sociais).

Este sistema divide os atores em três grandes grupos, os *issuers*, os *earners* e os *displayers*. Os emissores são responsáveis pela criação de *badges*, tornando-os disponíveis para os *learners*. Os *learners* submetem uma aplicação para um *badge* (*online* ou *offline*) e após receção dos *badges* a que têm direito, decidem onde querem disponibilizá-los. Os *displayers* são responsáveis por validarem os *badges* antes de os disponibilizar.

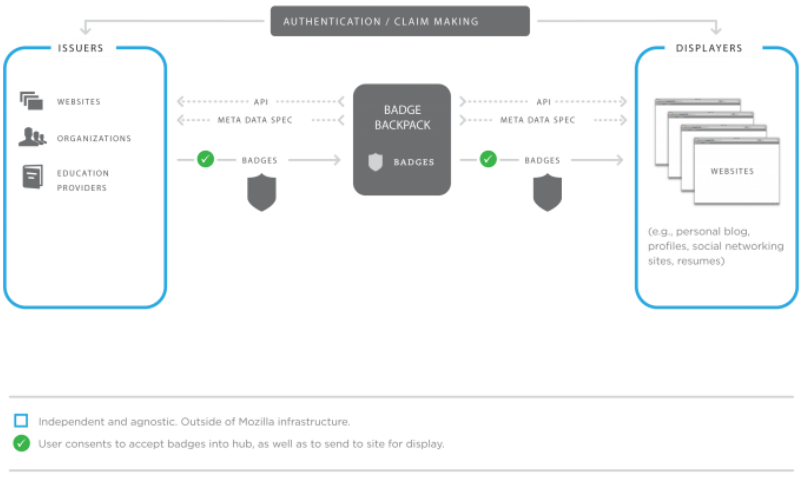

Figura 4 - Infraestrutura *Open Badges* (Mozilla n.d.)

[16] https://github.com/mozilla/openbadges-backpack/wiki/Open-Badges-related-widgets

[17] https://www.openbadges.me/

[18] https://wiki.mozilla.org/Badges/Onboarding-Issuer#Mozilla_Open_Badge_Infrastructure_.28OBI.29

2.4.1 Características técnicas

A infraestrutura de *Open Badges* criada pela *Mozilla* foi desenvolvida maioritariamente em Node.js através da *framework* ExpressJS. Os *badges* são ficheiros no formato PNG embebidos com metadados em JSON. A gestão de identidades é feita através do Mozilla Persona.

2.4.2 Tecnologia OBI

No que se refere à tecnologia, são apresentados as tecnologias mais relevantes da *infraestrutura de Open Badges*.

2.4.2.1 Node.js

Segundo o *website* (NodeJS 2016), o Node.js define-se como uma *framework* orientada para eventos assíncronos, visando a construção de aplicações web escaláveis. A mesma fonte salienta as diferenças com os modelos de concorrência atuais onde são aplicadas *threads*. A justificação para a abordagem disruptiva, prende-se com a relativa ineficiência e dificuldade na utilização desses modelos. Desta forma, os utilizadores da *framework* não precisam de ficar preocupados com problemas de *deadlock* ou de bloqueios por ações de *Input*/*Output*. Todo o conceito do node.js foi pensado para ser escalável, daí a decisão pela arquitetura atual.

2.4.2.2 JSON-LD

O JSON-LD[19], *JavaScript Object Notation – Linked Data*, é um formato JSON de *link* de dados. É baseado no tradicional formato JSON e é um permite tornar os dados em formato JSON interoperáveis para a *web*, sendo o formato ideal para serviços do tipo REST e bases de dados não relacionais (NoSQL) tais como CouchDB e MongoDB.

Os metadados associados a um *Open Badge* estão representados no formato JSON-LD. A especificação dos *Open Badges* consiste em três tipos de objetos base e requer que estejam num formato JSON-LD válido. A escolha do formato JSON como base para guardar a informação dos *badges* está relacionada com a pretendida interoperabilidade. Os três objectos de base são *Assertions*, *BadgeClasses* e *Issuers*. As suas relações estão representadas na Figura 5. Exemplos detalhados da estrutura JSON podem ser consultados no Anexo A.

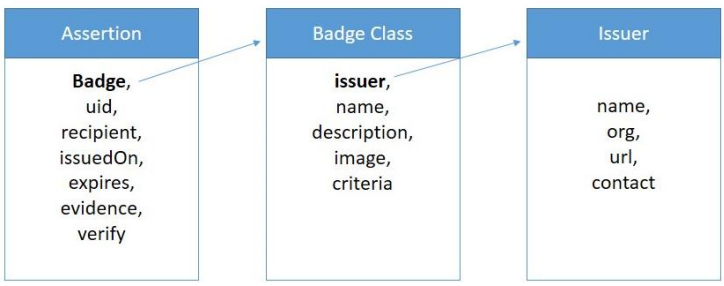

Figura 5 – Esquema dos três objetos constituintes dos metadados de um *Open Badge*

[19] http://json-ld.org/

As *assertions* são a representação em dados do *badge* ganho pelo *earner* e para as quais certas propriedades são de preenchimento obrigatório. Destacam-se as propriedades de contexto, tipo de objeto, recipiente, *badge* (url para o objeto *BadgeClass*), data de emissão, entre outras

Uma das recentes adições ao schema dos metadados dos *Open Badges* (v1.1) é a capacidade de fazer extensões a objetos de forma a adicionar informação extra que não se encontra nas propriedades dos objetos *standard*. Estas extensões permitem incluir informação complementar ao mesmo tempo que disponibiliza a outras entidades emissoras a capacidade de usarem as mesmas extensões de forma similar, com o objetivo da manutenção da interoperabilidade[20].

```
{
  "extension:ExampleExtension": {

"@context":"https://openbadgespec.org/extensions/exampleExtension/context.j
son",
    "type": ["Extension", "extensions:ExampleExtension"],
    "exampleProperty": "I'm a property, short and sweet."
  }
}
```

Figura 6 - Exemplo de extensão de um *Open Badge* (Alliance 2016a)

Existem extensões aceites pela comunidade dos *Open Badges*, que não fazem parte dos objetos *standard* uma vez que não são obrigatórios. Exemplos dessas extensões são nomeadamente o *link* de aplicação ao *badge*, *endorsement* (aprovação/verificação no sentido de credibilizar o *badge*), localização, acessibilidade e criador original.

2.4.2.3 API

Uma API[21], *Application Programming Interface*, é uma Interface de programação de aplicações, que surgiu como resposta à necessidade de separar responsabilidades por grupo de funcionalidades e promoverem a interação entre os diferentes componentes. Assim, quando um componente precisa de aceder a funcionalidades de um outro componente, utiliza um conjunto *standard* de *requests* a APIs que foram definidas pelo componente de destino. Existem dois tipos de API's, as públicas e privadas. Um API pública é disponibilizada livremente a todos os utilizadores enquanto que uma API privada é utilizada no contexto interno das próprias aplicações ou a determinadas aplicações externas devidamente autorizadas.

[20] http://specification.openbadges.org/

[21] http://www.computerworld.com/article/2593623/app-development/application-programming-interface.html

Num contexto *web*, as API's, são normalmente disponibilizadas através de *Web Services* REST (Representation State Transfer). Este tipo de API's, recebe pedidos HTTP do cliente e, através da interpretação do seu conteúdo, são geradas as respostas adequadas, tipicamente em formato JSON (Hunter 2013).

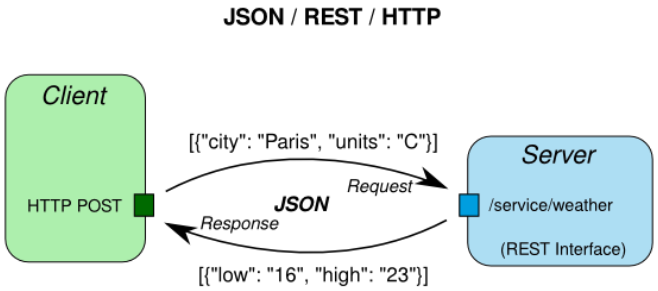

Figura 7 – Diagrama genérico de funcionamento de uma API REST

Este tipo de API tem a vantagem de ser, relativamente, fácil de utilizar e desenvolver, mas onde é necessário ponderar os riscos legais, técnicos e estratégicos associados. Ao implementar uma API, cuidados especiais com a autenticação dos utilizadores e com a segurança dos dados são importantes. *Representational State Transfer* (REST), é um estilo arquitetural não padronizado baseado no *standard* HTTP (*Hypertext Transfer Protocol*). Neste protocolo, tudo é modelado com base em recursos tendo cada recurso tem um URI (*Uniform Resource Identifier*) único associado (Chen et al. 2012). Serviços REST "puros" são completamente stateless (não guardam informação do cliente do lado do servidor), o que faz com que sejam especialmente indicados para o desenvolvimento de uma API pública focada em operações de Create-Read-Update-Delete (Chen et al. 2012).

Estas arquiteturas têm dois componentes essenciais: clientes e servidores. Os clientes fazem pedidos a um determinado recurso localizado no servidor, onde estes são interpretados de forma a gerar a resposta adequada. Os pedidos são efetuados utilizando os métodos:

- GET é utilizado para obter informação de um recurso sem o modificar;
- POST é utilizado para criar e editar informação;
- PUT é utilizado para atualizar informação;
- DELETE é usado para remover informação do servidor.

Um recurso é, essencialmente, qualquer conceito/informação que possa ser endereçada por um URI (Chen et al. 2012). REST suporta variados dados em vários formatos e apresenta ganhos de performance e escalabilidade em relação às soluções concorrentes, já que consegue transmitir o mesmo tipo de informação utilizando uma menor largura de banda.

Os conceitos base associados ao REST são (Jacobson et al. 2012):

- Interface de separação entre cliente e servidor;
- Não é guardada qualquer informação do cliente no servidor (*stateless*);
- A utilização de camadas intermédias para aumentar a escalabilidade e a segurança (o cliente não percebe se está ligado ao servidor principal ou a um intermédio);
- Orientado a recursos.

Uma API REST apresenta grandes vantagens quando o foco é a gestão de dados (CRUD) garantindo o desenvolvimento de aplicações com um consumo de largura de banda reduzida e de elevada performance.

2.4.2.4 Webhooks

O conceito de *webhook*[22] é simples. Um *webhook* é basicamente uma *callback* HTTP que é chamada quando um evento a despoleta, via uma simples notificação por HTTP POST. A aplicação *web* que implemente *webhooks* irá efetuar um *request* via POST para um URL.

Na ótica do utilizador, os *webhooks* são uma forma de receber informação valiosa no instante em que ocorre. Têm um grande potencial e habitualmente são usados de três formas:

- *Push*: Receber data em tempo real
- *Pipes*: Receber dados e transferi-los
- *Plugins*: Processamento de dados com retorno de resposta (extensão de aplicação)

2.4.2.5 *Standards* de imagem

No que se refere às imagens dos *badges,* estas devem estar no formato PNG, com dimensões quadradas e não excederem o tamanho de 256 kb. Outra das recomendações é a não utilização de imagens com uma resolução inferior a 90x90. A imagem é fornecida com um URL para a imagem no servidor do *issuer*, com metadados embutidos na própria imagem.

Ao usar BadgeKit, os emissores podem projetar *badges* dentro do aplicativo Web usando uma ferramenta gráfica, ou podem fazer *upload* de imagens construídas por outra ferramenta externa.

2.4.3 Ferramentas OBI

2.4.3.1 BadgeKit

O BadgeKit[23] é um conjunto de ferramentas criadas pela *Mozilla* para agilizar o processo de desenho, construção e emissão de *badges, com* foco nas organizações emissoras de *badges*. Fornece, entre outras funcionalidades, a possibilidade de utilizar *templates* e modelos tipo

[22] https://webhooks.pbworks.com/w/page/13385124/FrontPage
[23] https://wiki.mozilla.org/Badges/badgekit

milestone. A versão *Beta* desta ferramenta foi lançada em Março de 2014, sendo composta por dois componentes, uma aplicação web e uma API. A aplicação web visa fornecer uma interface gráfica para as entidades emissoras criarem novos *badges*, gerirem os estados de *bagdes* criados e possibilitar a emissão dos mesmos. A API é um componente que permite uma maior flexibilidade e atua como um complemento da aplicação web e/ou interligação/customização com o próprio sistema da entidade emissora. Como exemplo das suas funcionalidades, destacam-se a possibilidade de criar e publicar um *badge* para a aplicação web, obter informação de *badges* criados de forma a poder apresentá-los aos utilizadores de destino, gerar códigos de requisição de um badge, atualizar dados dos badges, submeter aplicações por parte dos utilizadores de destino para requererem o *badge* que lhes é destinado.

Como o *BadgeKit* lida com grande parte do processamento *backend* na aplicação de um *badge*, é possível focar na experiência de emissão e receção por parte dos *earners* propriamente dita.

Os *Issuers* podem usar o *BadgeKit* e a API em conjunto com *website* próprio, da seguinte forma:

- Aplicação web: criar e publicar badges
- API: pesquisar badges disponíveis
    - *Website* do *issuer*: apresentar badges disponíveis aos *earners*
- *Website do Issuer*: Permite aos *earners* candidatarem-se à atribuição de *badges*
    - API: Aplicação de dados
    - Pode incluir evidência por parte do *earner*
- Web app: Rever aplicações (incluindo evidências)
    - Tomar decisões de atribuição de badges
- API: Receber dados de atribuição no webhook
    - Issuer *website*: Interagir com o *earner* de acordo com o *award*
- Passos opcionais (ex: oferecer para publicar num *backpack*)

A API do *BadgeKit* API inclui diferentes *endpoints* de forma a permitir a interação com as várias partes do processo de criação e atribuição de badges. As entidades emissoras de *badges* podem usar o BadgeKit em conjunto com a API para criar badges, listar *badges* disponíveis e publicar badges para aplicações bem-sucedidas.

O BadgeKit lida com grande parte do processo de criação, avaliação e emissão. Permite a integração com *website* próprio proporcionando uma interface customizada aos *earners*.

A API fornece *endpoints* para os quais podemos comunicar com dados de *badges* e *webhooks* que podem ser configurados para serem notificados de eventos de emissão de *badges*. A utilização conjunta da aplicação web e da API, pode permitir controlar a experiência do utilizador enquanto que o BadgeKit lida com a lógica de *backend*.

O acesso à API é dividido em três categorias:

- Listagem

- Emissão
- Avaliação e revisão

A listagem de badges para potenciais interessados é possível de ser obtida pela API do BadgeKit. O BadgeKit permite a criação de badges, pelo que a aplicação de terceiros pode recolher os dados do badge e apresentá-los num ambiente próprio.

A API usa pedidos diretos e retorna dados no formato JSON, que podem ser tratados por diversas tecnologias existentes.

O processo de emissão de badges pode variar da seguinte forma:

- Emitidos diretamente
  - O *earner* fornece o *email* e o *issuer* emite o *badge*
- Emitidos por utilização de códigos de validação
  - O *earner* insere o código de validação para receber o *badge*
- Emitidos por mecanismo automático
  - Um determinado *badge* pode, por exemplo, ser emitido automaticamente após o *earner* obter determinado conjunto de *badges*
- Emitidos após avaliação de um revisor
  - Alguns *badges* requerem a avaliação por parte de elementos externos, denominados revisores que analisam as evidências e as informações de integridade do *badge*

Salienta-se o facto de que na emissão de um *badge*, o *BadgeKit* não procede a qualquer comunicação ao *earner*. Em vez disso, a API envia notificação para o seu URL *webhook* quando um *badge* é emitido. Após receção da notificação, é possível entrar em contato com o *earner*.

O aplicativo BadgeKit Web também inclui uma série de utilitários administrativos para auxiliar processos de emissão de *badges*. Por exemplo, o criador de um *badge* pode incluir uma nota para o revisor. A interface também fornece a capacidade de exibir provas para *badges* e assinaturas de integridade para a revisão dos *badges* do *earner*.

Quando um potencial *earner* vê um *badge* listado que lhe interesse, ele pode apresentar provas para validação das competências. Dentro da aplicação *BadgeKit*, o avaliador pode então validar as evidências contra os critérios do *badge* (por vezes no contexto de uma assinatura), antes de emitir o *badge* ao requerente.

Para usar a API, o emissor tem de ser capaz de comunicar com os processos *BadgeKit*. Para o efeito, a API emite mensagens quando certos eventos do processo de emissão de *badges* ocorrem, tais como avaliações a serem concluídas ou *badges* emitidos. O *website* do emissor pode receber notificação desses eventos, configurando um URL *webhook* para o sistema dentro da instância *BadgeKit*, permitindo-lhe responder e adaptar a experiência de emissão aos seus

earners. A plataforma web do emissor pode, portanto, responder às ações do processo de emissão de *badges* utilizando o *BadgeKit* e integrando-as com a sua próprias funcionalidades. O acesso aos *endpoints* da API do *BadgeKit* requer a introdução de um parâmetro JSON *web token*.

2.4.3.2 Mozilla Backpack

Se o *BadgeKit* é um conjunto de ferramentas destinado a *issuers*, o *Mozilla BackPack*[24] é um conjunto de ferramentas destinado aos *earners* dos *badges.* Com esta ferramenta é possível colecionar, agrupar e organizar os *badges* que o utilizador recebe. O *Mozilla Backpack* visa dar ao utilizador o controlo total sobre os *badges* ganhos. Destacam-se as funcionalidades de permitir que certas entidades emissoras atribuam *badges* para o *backpack* assim como é possível aceitar, um a um, cada *badge* que seja enviado para o *Backpack* do *earner*. Relativamente à organização dos mesmos, é possível organizar, na área de coleções, *por* grupos, como por exemplo, o agrupamento de *badges* por áreas de assunto ou por tipo de competência, até que o utilizador entenda que os seus *badges* possam ser agrupados para a partilha. Na Figura 8 está representada a área de coleções do *Mozilla Backpack*. Nesse momento, o utilizador tem a possibilidade de tornar um grupo de *badges* público para serem visualizados em *websites*, *widgets* ou redes sociais. No entanto é necessário que os websites de partilha conheçam o *email* do *earner* para poderem ser disponibilizados para o público. No que concerne a partilha com redes sociais, existem as opções de partilha com Twitter, Google+ e Facebook, e o *link* para o portefólio público será incluído na altura de publicação nas redes sociais.

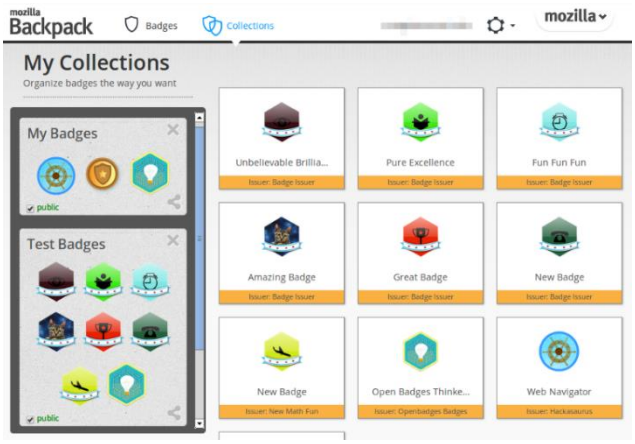

Figura 8 – Área de coleções de *badges* do *Mozilla Backpack* (Mozilla 2016b)

[24] https://github.com/mozilla/openbadges-backpack/wiki/Share-your-Badges-with-the-Mozilla-Backpack

2.4.3.3 Issuer e Displayer API

Através da *Issuer* API[25], é possível publicar badges no *Mozilla Backpack*. A API lida com a obtenção das permissões do *earner* para a publicação no B*ackpack*. Com a *Displayer* API[26] é possível obter badges públicos a partir do *Backpack* para determinado utilizador. Os *earners* podem organizar os seus badges no Mozilla *Backpack* em grupos e controlar se esse grupo é público ou não. Para obtê-los é necessário fornecer apenas o *email* do utilizador ao B*ackpack* e desta forma disponibilizá-lo no próprio *website*, aplicação ou rede social.

2.4.3.4 Connect API

Em alternativa à *Issuer* API, é possível usar o *Backpack Connect*[27] para publicar badges ganhos no *Mozilla Backpack*. A grande vantagem relativamente à *Issuer* API, em que cada utilizador tem de dar permissão para receber os badges no seu Backpack, no *Backpack Connect* é possível configurar uma sessão permanente para a qual não é necessária a permissão do utilizador por cada badge recebido. Assim sendo, após a primeira permissão ser dada, é possível enviar badges para o *Backpack* automaticamente, sem necessidade de autorização. Para esta conexão com o *Backpack*, a *Connect* API utiliza tokens de acesso.

Contudo, a Issuer API é uma alternativa menos complexa de submeter badges para o *Backpack*, requerendo permissão para cada tentativa de submissão de *badge* mas requer ao mesmo tempo muito menos esforço de desenvolvimento no *website* do emissor de badges.

2.4.3.5 Badge baking

Um *baked badge* é basicamente uma imagem com dados embebidos. Os emissores de badges utilizam afirmações para representar os *badges* que premiaram aos *earners*. Cada afirmação num *badge* inclui um ficheiro de imagem e alguns metadados descritivos do recetor, do badge e da entidade emissora. Aplicações conhecedoras das estruturas de Open Badges conseguem extrair os metadados do *badge*.

É possível "cozinhar" (baking) um badge através da *Baker* API[28] do *Mozilla Backpack* ou através da ferramenta openbadges-bakery. Se a entidade emissora utilizar a *Issuer* API para submeter badges para o *Backpack*, o "cozinhar" do badge será efetuado pela própria API.

2.4.3.6 Identity

O componente de validação de utilizadores, mais concretamente na identificação de *earners* é essencial para manter o bom funcionamento do ecossistema OBI. A recomendação por parte da Mozilla é a utilização do produto *Persona*, uma vez que permite a descentralização da gestão da de utilizadores. Funciona através da validação de *email* do utilizador e tem a vantagem de não precisar de nenhum perfil ou informação pessoal do *earner*, apenas o email é suficiente.

---

[25] https://github.com/mozilla/openbadges-backpack/wiki/Using-the-Issuer-API

[26] https://github.com/mozilla/openbadges-backpack/wiki/Using-the-Displayer-API

[27] https://github.com/mozilla/openbadges-backpack/wiki/Using-the-Backpack-Connect-API

[28] https://github.com/mozilla/openbadges-backpack/wiki/Badge-Baking

2.5 Avaliação de soluções existentes

Uma vez que o conceito dos *Open Badges* é recente, existem algumas iniciativas no sentido da adoção ou potenciação dos mesmos. Estas podem ser diferenciadas entre aquelas que são desenvolvidas pela *Mozilla* e *MacArthur Foundation* com base na OBI e as outras que se suportam na mesma infraestrutura mas posicionam-se paralelamente a esta. Relativamente à infraestrutura OBI, verificam-se algumas limitações existentes, nomeadamente em resposta ao problema proposto no âmbito deste projeto, que são na pesquisa e descoberta de *badges*.

Neste momento, na infraestrutura atual da Mozilla (OBI), a única possibilidade de pesquisar os *badges* adquiridos pelos utilizadores é estes marcarem-nos no *Backpacks* pessoais como públicos e ser efetuado um pedido, pela entidade que irá efetuar o *display*, à *Displayer* API fornecendo o *email* do utilizador. Este processo é adequado ao *display* em redes sociais por parte do *earner*, mas não é o ideal se pensarmos num ecossistema partilhado para pesquisa de badges e descoberta. Por outro lado o *issuer* tem de comunicar com a *Issuer* API no sentido de emitir o *badge* para um determinado utilizador (*earner*), fazendo com que os *badges* do *issuer* estejam muito segmentados, alojados nos *Backpacks* dos *earners* e disponíveis nos *websites* pessoais ou redes sociais.

Foi efetuado um estudo comparativo, representado na Tabela 1 que ilustra e compara o âmbito de cada uma das soluções apresentada em 2.3.2 e 2.4.3. O que se verifica é que as ferramentas existentes apresentam uma modularidade relevante mas que mesmo combinando-as em diferentes conjuntos de forma a obtermos soluções com todas as responsabilidades necessárias à emissão, recolha e *display* dos *badges*, estas não iriam ser capazes de potenciar ou mesmo responder à descoberta e pesquisa de *badges* porque nenhum componente se foca nesse ponto.

O único componente/solução que visa responder, em parte, à descoberta de *badges* é o projeto *Mozilla Discover*, só que este ainda não está pensado para conexão, por exemplo, com o *Backpack* do utilizador, logo ainda não está incorporado na infraestrutura OBI. Este facto faz com que este projeto seja um bom exemplo de base do que se pretende para a área da descoberta, mas que não passa de um protótipo de teste. Mesmo que esta solução fizesse parte da infraestrutura de base da OBI, a segmentação em termos de aplicações existentes desde o *issuer* até ao *earner* e consequente *display*, estivesse demasiado segmentada.

Uma solução possível seria incorporar as ferramentas num sistema agregador, com *issuers* e *earners*, mas que não fosse exclusivo, permitindo conexão com a tradicional infraestrutura OBI.

Tabela 1 – Quadro comparativo das ferramentas ou soluções existentes mais relevantes

Produto/serviço	**Mozilla OBI**	**Issuer**	**Earner**	**Displayer**	**Custo**	**Pontos fortes**	**Limitações**
Backpack	Sim	Não	Sim	Não	Gratuito	• Organização e publicação de *badges*	• Partilha de *badges apenas* por utilizador
BadgeKit	Sim	Sim	Não	Não	Gratuito	• Flexível • Grande número de funcionalidades	• Focado apenas na emissão de *badges*
OpenBadgesMe	Não	Sim	Não	Não	Gratuito	• Flexível • Editor de *badges* completo • Interoperável	• Focado apenas na emissão de *badges*
Open Badges Factory	Não	Sim	Não	Não	Pago/Trial	• Criação e emissão de *badges*	• Focado apenas na emissão de *badges*
Passport	Não	Não	Sim	Sim	Gratuito	• Upload de badges • Importação de *badges* do Mozilla *Backpack*	• Focado apenas na recolha e *display* de *badges*
Mozilla Discover	Não	Não	Sim	Sim	Gratuito	• Construção de percurso de aprendizagem personalizado • Pesquisa de *badges*/percursos por áreas	• Ferramenta protótipo que usa apenas dados de teste.

2.6 *Standards* de aprendizagem

Um dos principais problemas dos sistemas de aprendizagem em geral é o modo como o conteúdo pode ser distribuído de forma a ser compatível com as diversas plataformas e dispositivos (Behringer 2013). O desenvolvimento *web* nas áreas de programação (HTML e *JavaScript*) possibilita, às experiências de aprendizagem, um grau de interoperabilidade sintática suficiente para que os sistemas de gestão de aprendizagem (LMS) disponibilizassem aos utilizadores um acesso simples a informação. De forma a uniformizar a representação dos conteúdos relacionados com as experiências de aprendizagem, vão sendo propostos *standards* de especificação técnica desse conteúdo.

2.6.1 LRS

A especificação xAPI (ADL 2013) define um LRS[29] como um sistema que persiste informação relacionada com experiências de aprendizagem. Antes da introdução da xAPI, todos os LRS eram LMS[30], no entanto um LRS define-se como um serviço em *cloud* que visa ser um repositório de informação que permite guardar e disponibilizar a mesma. Desta forma, não inclui as funcionalidades diretamente associadas a um LMS, como as de criação e disponibilização de conteúdos, *reporting*, avaliação, performance, ranking, calendários de cursos ou formações, controlo de registo de cursos ou formações. O LRS, devido às suas características de interoperabilidade pode ser integrado em sistemas LMS ou, como tem sido mais comum, disponibilizado como um sistema totalmente independente (Lim 2015).

2.6.2 xAPI

A xAPI foi desenvolvida pela ADL[31] para responder ao desafio do *tracking* de experiências. Esteve projeto teve como ponto de partida o SCORM[32], que é uma especificação que permite a interoperabilidade entre cursos/formações com os LMS, e visou uma modernização e atualização da especificação de forma a responder aos desafios que se avizinham na área da aprendizagem digital. Em 2012, a *Rustici Software* desenvolveu o xAPI, apelidado de projeto *Tin* Can, que culminou no lançamento da primeira versão da xAPI. Na realidade, a xAPI substituiu os protocolos de comunicação de dados e modelos do SCORM, tendo-se baseado na denominada especificação *Activity Stream*. Esta especificação foi desenvolvida pela Google, Facebook, Microsoft e outros com o objetivo de criar um formato *standard* para partilha de experiências sociais. Um dos aspetos chaves nas *activity streams* é que são legíveis por máquinas bem como por seres humanos, o que permite adicionar contexto a ambos os grupos.

[29] LRS – *Learning Record Store*
[30] LMS – *Learning Management Systems*
[31] ADL – *Advanced Distributed Learning* (https://www.adlnet.gov/)
[32] SCORM – *Sharable Content Object Reference*

Os xAPI *statements* são representados em formato JSON e têm por objetivo representar de que forma as atividades que as pessoas realizam, evidenciam as suas experiências de aprendizagem (Lim 2015). Com a xAPI, é possível elevar a interoperabilidade também a nível de dispositivos, permitindo que as experiências de aprendizagem sejam capturadas em dispositivos móveis ou fixos. O xAPI pode ser visto também como um serviço *web* que permite, enviar e guardar em segurança, experiências de aprendizagem num LRS. Essas experiências, capturadas como *statements*, têm um formato base que é constituído por três elementos, o ator, o verbo e o objeto.

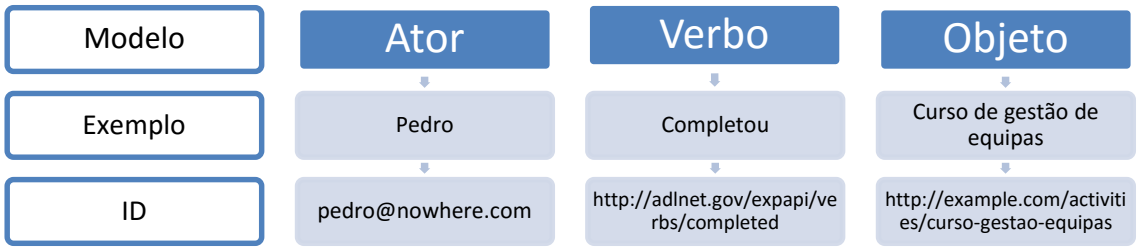

Figura 9 – Elementos básicos e estrutura de um *statement* da xAPI

No exemplo da Figura 9, o "ator Pedro completou um curso de gestão de equipas", que é passível de ser traduzida para a correspondente estrutura JSON como *statement xAPI* válido, Figura 10.

```
{
  "actor": {
    "name": "Pedro",
    "mbox": "mailto:pedro@nowhere.com"
  },
  "verb": {
     "id": "http://adlnet.gov/expapi/verbs/completed",
     "display": {"pt-PT": "completou"}
  },
 "object": {
   "id": "http://example.com/activities/curso-gestao-equipas",
   "definition": {
      "name": { "pt-PT": "Curso de gestão de equipas" }
   }
  }
}
```

Figura 10 – Estrutura JSON representativa de um *statement*

Para além deste tipo de informação base, é possível adicionar mais detalhe utilizando um elemento de contexto, que pode, no exemplo dado, enquadrar o curso de gestão de equipas no âmbito de uma especialização em gestão de empresas ou mesmo conter informação acerca

da nota final. Para além do contexto, existe a possibilidade de inserir informação adicional através de elementos de extensão.

Em resumo, a xAPI tem por objetivo o suporte dos vários tipos de conteúdos relacionados com experiências de aprendizagem, é simples de implementar por usar um modelo interpretável por máquinas e seres humanos em simultâneo e ser apresentado num formato padrão como o JSON. O facto de o conteúdo não ter de ser exclusivo ou proveniente de um LMS e poder ser transportado e enviado das mais diversas formas, permite uma flexibilidade e confere um elevado grau de interoperabilidade entre sistemas que suportem este *standard* (Berking 2016).

2.6.3 A xAPI e outras especificações

Para além da xAPI, existem outras especificações na área de *tracking* de experiências de aprendizagem. Na Figura 11, são apresentadas sete especificações e as suas características.

	Core of the specification	**Educational information**	**Service definition**
NSDL Paradata	Object	No	No
Organic.Edunet format	Object	No	No
xAPI	Event	Yes	Yes
IMS Caliper	Event	Yes	Yes
CAM	Event	No	Yes
Activity Stea.ms	Event	No	Yes

Figura 11 – Comparação entre sete especificações na área das experiências de aprendizagem (Santos et al. 2015)

Como é possível verificar, as especificações xAPI, IMS Caliper, CAM e Activity Strea.ms, definem, para além de um modelo de dados, uma definição dos serviços aplicáveis aos LRS. De salientar também a diferença entre as especificações xAPI e IMS Caliper e as restantes, uma vez que suportam explicitamente informações educacionais tais como a nota e resultado de um curso/formação.

Segundo o artigo sobre *tracking* de dados em ambientes de aprendizagem (Santos et al. 2015), a especificação xAPI apresenta vantagens relativamente às restantes, tais como o facto de ser centrada em eventos, e por este motivo conseguir abranger diversos tipos de ações, enquanto que as especificações do tipo Objeto estão mais relacionadas com ações sociais. Outra característica diferenciadora é fornecer uma definição de serviço e modelo de dados, o que promove a interoperabilidade e permite extensibilidade a nível das arquiteturas analíticas de aprendizagem. E por último, o facto de ser muito popular e ser suportada por serviços de LRS no mercado tais como a *Cloud Scorm* e *Learning Locker*. Estava última característica permite que soluções desenvolvidas com base nesta especificação, possam ser compatíveis com os

sistemas existentes e potenciar o crescimento e adesão de novos elementos do ecossistema de aprendizagem, como os *Open Badges*.

2.6.4 Análise a sistemas LRS

Os LRS, devido à natureza da especificação da xAPI, subentendem um âmbito restrito para a sua utilização, nomeadamente na persistência e disponibilização de *statements* xAPI. Eles podem incluir suporte a análise dos dados, *reporting* e motores de visualização, mas isso não faz parte da especificação base. Se o objetivo de um LRS for a visualização, combinação, agregação e manipulação de dados, é aconselhável a utilização de um LRS que inclua um motor de análise de dados. Como já foi mencionado anteriormente, um LRS pode estar integrado num LMS, o que obviamente não permite ao utilizador escolher o LRS a integrar, sendo este parte do LMS. No entanto, começam a surgir cada vez mais LMS com API de integração com LRS externos. A vantagem destes sistemas são a personalização e a flexibilidade que apresentam, embora a evolução natural e constante da especificação xAPI introduz um fator de risco relacionado com a necessária atualização da API de integração do LMS.

Na Tabela 2 são apresentados alguns dos mais comuns LRS do mercado bem como as suas características (Berking 2016).

Tabela 2 – LRS existentes no mercado e suas funcionalidades

LRS		**Funcionalidades**			
		Persistência de *statements*	Motor de análise de dados	LMS com LRS integrado	LMS com API de integração com LRS
ADL LRS (open source)	https://github.com/adlnet/ADL_LRS	x	-	-	-
Wax LRS (versão base)	http://www.saltbox.com/wax-learning-record-store.html	x	-	-	-
GrassBlade	http://www.nextsoftwaresolutions.com/grassblade-lrs-experience-api/	x	x	-	-
Learning Locker (*open source*)	https://github.com/LearningLocker/learninglocker	x	x	-	-
Watershed LRS	http://site.watershedlrs.com/				
Wax LRS (versão paga)	http://www.saltbox.com/wax-learning-record-store.html	x	x	-	-
Skillanalyzer	http://skillaware.com/en/skillanalyzer/	x	x	-	-
1xHive	http://www.brightcookie.com/products/	x	x	x	-

Brindlewaye Dac	http://brindlewaye.com/all-about-design-a-course/	x	x	x	-
ChallengeMonitor	http://www.e-teach.ch/eteachServer.php	x	x	x	-
LearnUpon	https://www.learnupon.com/	-	-	-	x

2.6.5 xAPI e os Open Badges

Numa primeira análise, tanto a xAPI como os *Open Badges* apresentam especificações próprias para documentação e partilha de aprendizagens pelos mais variados ambientes de aprendizagem. Desta forma, poderiam ser considerados incompatíveis, contudo com uma análise mais aprofundada é possível verificar que cada conceito só lida com uma parte desta deste processo. A Figura 12 ilustra a forma como a especificação xAPI pode ser utilizada para associar experiências de aprendizagem e portefólios digitais.

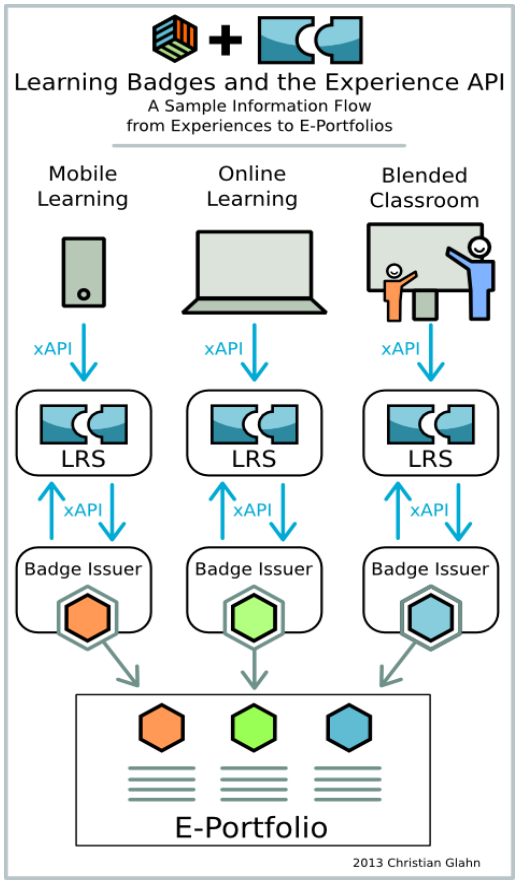

Figura 12 – *Open Badges* e xAPI (Glahn 2013)

A xAPI e o LRS têm a função de documentar e disponibilizar o acesso a experiências de aprendizagens de várias e para várias fontes. Os *Open Badges*, ao contrário da especificação

xAPI, apresentam limitações para interpretação dos mesmos por sistemas automáticos, de forma que a utilização da xAPI pode colmatar esta limitação. Outro dos pontos a favor da xAPI em conjunto com o *Open Badges* está relacionado com a possibilidade de associar os emissores de *badges* com os LRS, através de *tracking* das experiências de aprendizagem que permitem que os LRS detetem uma nova atividade de aprendizagem e criem um xAPI *statement* em concordância. Desta forma, o emissor saberá que foi atingido um objetivo de aprendizagem e emitir um *badge* para o portefólio pessoal do utilizador.

Por outro lado, este processo permite a partilha e pesquisa mais eficaz das experiências de aprendizagem, uma vez que, se os *statements* representativos de *Open Badges* coexistirem com outros tipos de *statements*, é possível, pela natureza do padrão xAPI, que os *Open Badges* estejam em ecossistemas de maior dimensão. O desenvolvimento de soluções nesta área pode ser chave para a adoção dos *Open Badges* numa maior escala.

Têm sido realizados desenvolvimentos por grupos de trabalho neste sentido, tendo sido proposta, pelo grupo de trabalho denominado *xAPI Open Badges*, uma receita para integração da xAPI com os *Open badges*. Este grupo é formado por elementos ativos das comunidades dos *Open Badges* e *Tin Can* (xAPI). Como resultado desse trabalho conjunto, foi criada uma "receita" para mapeamento entre a especificação de um *Open Badge* e a especificação xAPI. Na Tabela 3, são apresentados os identificadores propostos para esse mapeamento.

Tabela 3 – Identificadores dos *Open Badges* utilizados na adaptação à especificação xAPI
(*Adaptado de https://github.com/ht2/BadgesCoP/blob/master/earning/vocab.md)*

Propriedade xAPI	**Identificador**	**Descrição**
Verb.id	http://specification.openbadges.org/xapi/verbs/earned	Evidencia que o ator arrecadou o objeto.
context.contextActivities.category.N.id	http://specification.openbadges.org/xapi/recipe/base/0_0_2	Evidencia que o *statement* é referente à receita dos *Open Badges*.
attachments.usageType	http://specification.openbadges.org/xapi/attachment/badge	Evidencia que o anexo é uma imagem de um Open Badge.
object.definition.type	http://activitystrea.ms/schema/1.0/badge	Evidencia que o objeto é um Open Badge

Na Tabela 4, são referidos as propriedades xAPI que contêm o mapeamento da informação acerca das *assertions* e das *badge classes*, referidas em 2.4.2.2.

Tabela 4 – Extensões dos *Open Badges* adaptados à especificação xAPI
(*Adaptado de https://github.com/ht2/BadgesCoP/blob/master/earning/vocab.md)*

Propriedade xAPI	**Extensão**	**Descrição**
result.extensions	http://specification.openbadges.org/xapi/extensions/badgeassertion	Contém um objeto xAPI representativo das *assertions* dos *Open Badges*.

No exemplo seguinte, Figura 13, ilustra-se a estrutura base representativa de um *Open Badge* adaptado para a especificação xAPI. A integração da xAPI e LRS com os *Open Badges* elimina a necessidade de um *Backpack* como repositório de *badges*, uma vez que através da xAPI é

possível coexistirem com outro tipo de elementos de aprendizagem. A natureza da especificação xAPI ainda apresenta outra vantagem, a nível de língua, uma vez que suporta diferentes localizações num mesmo *statement*.


```
{
  "actor": {
    "name": "Example Earner",
    "mbox": "mailto:earner@example.com"
    }
  },
  "verb": {
    "id": "http://specification.openbadges.org/xapi/verbs/earned",
    "display": {
      "en-US": "earned"
    }
  },
  "result": {
    "extensions": {
      "http://specification.openbadges.org/xapi/extensions/badgeassertion":
{
        "@id": "http://www.example.com/assertion/1"
      }
    }
  },
  "context": {
    "contextActivities": {
      "category": [{
        "id": "http://specification.openbadges.org/xapi/recipe/base/0_0_2",
        "definition": {
          "type": "http://id.tincanapi.com/activitytype/recipe"
        },
        "objectType": "Activity"
      }]
    }
  },
  "object": {
    "id": "http://www.example.com/badgeclass/1",
    "definition": {
      "extensions": {
        "http://specification.openbadges.org/xapi/extensions/badgeclass": {
          "@id": "http://www.example.com/badgeclass/1",
          "image": "http://www.example.com/badgeimage/1.png",
          "criteria": "http://www.example.com/criteria/1",
          "issuer": "http://www.example.com/issuer/1"
        }
      },
      "name": {
        "en-US": "Name of the badge"
      },
      "description": {
        "en-US": "Description of the badge."
      },
      "type": "http://activitystrea.ms/schema/1.0/badge"
    },
    "objectType": "Activity"
  }
}
```


Figura 13 – Exemplo de um *Open badge* representado como um *statement* xAPI

Segundo Downes (Downes 2015), assinalam-se em seguida as grandes vantagens desta abordagem:

- Partilha de *Badges* entre diferentes sistemas
- Partilha de metadados associados aos emissores em diferentes sistemas
- Definir critérios e evidências dos *Open Badges* interpretáveis automaticamente (máquina)
- Emitir *badges* automaticamente, baseados nos *statements* xAPI
- Utilização dos LRS como um *backpack*
- Localizar a definição de um *badge*
- Descobrir *badges* que de outra forma não eram passíveis de serem encontrados

3 Valor, ameaças e oportunidades

Neste capítulo, é realizada uma análise *SWOT*[33] da área de negócio dos *Open Badges*, são identificados parceiros e por fim uma análise de valor à solução proposta.

3.1 Parceiros e Stakeholders

As atividades recentes fizeram com que parceiros e *stakeholders*, não tradicionais no ecossistema educativo, encontrassem motivos para discutirem novas formas de colaboração (Finkelstein et al. 2013). Por exemplo, áreas como a gestão financeira, a saúde, o desporto, entre outros, começam a compreender o benefício da formação, financiamento e assistência de marketing em troca da emissão de *Open Badges* para reconhecimento de competências e aptidões. Nesse sentido, os potenciais *earners* de *badges* devem ser aconselhados a partilhar que tipo de *badges* lhes interessarão ou recompensarão mais e as potenciais recompensas que estes lhes poderão trazer.

Um dos parceiros e *stakeholders* a destacar é a *Badge Alliance*. Esta organização, formada no ano de 2014, tem por objetivos coordenar a utilização e implementação dos *Open Badges* nos diversos contextos ao mesmo tempo que se dispõe a assegurar e manter um ecossistema de sustentabilidade para os Open Badges (Casilli & Hickey 2016).

[33] SWOT – Strenghts, Weaknesses, Opportunities and Threats

3.2 Valor e Impacto

Reconhecer a aprendizagem e sucesso como parte da vida de um individuo, incluindo conquistas formais e não formais, abre a possibilidade de pessoas de todas as idades partilharem o seu percurso pessoal de aprendizagem. Indivíduos ou organizações com experiência e vontade de apostar, podem atribuir *badges* àqueles que obtenham competências, conhecimentos ou conquistas de valor para essa organização (Finkelstein et al. 2013). A natureza visual dos *badges* facilita na compreensão do progresso efetuado, o que pode ser um fator extra de motivação para o aluno. Consequentemente, os *badges* podem ajudá-lo a construir o seu caminho de aprendizagem. Todas as organizações que emitem *badges* digitais aumentam o seu potencial e área de atuação, uma vez que chegam novos grupos e promovem oportunidades de aprendizagem que podem ser reconhecidas.

O procedimento de oferecer *badges* a quem participa em cursos *online* está a atrair a curiosidade e a fomentar a partilha de opiniões sobre os mesmos nos últimos anos (Cross et al. 2014). Descrito como uma "credencial digital que representa as competências, interesses e conquistas obtidas por um indivíduo através de projetos específicos, programas, cursos e outras atividades," (Mozilla 2013), o próprio *badge* não é considerado um método de avaliação por si só, mas em vez disso representa um resultado da avaliação e apresenta informações sobre a avaliação.

Uma grande parte do valor de um *badge* advém da sua natureza eletrónica e do facto de conter benefícios inerentes à sua composição digital (Finkelstein et al. 2013). Os *Badges* podem ser distribuídos mais facilmente do que um diploma tradicional, isto significa que eles podem ser oferecidos com maior frequência ou para fins mais granulares. A tecnologia para a emissão de *badges* também está disponível para qualquer entidade, elevando a probabilidade de que grupos que não emitam tradicionalmente *badges*, o venham a fazer no futuro, contribuindo para aumentar a credibilização de todo o sistema. A capacidade de dar reconhecimento formal não será, no futuro, apenas do domínio das instituições de ensino ou programas de certificação. Os indivíduos ou organizações com experiência podem tornar-se emissores de *badge* e, desta forma, mais atividades e diferentes demonstrações de capacidade se tornam sujeitas reconhecimento.

A natureza visual dos *badges*, incluindo características como forma, cor, tamanho, texto e iconografia e sua forma relativa torna-os veículos muito apropriados para ilustrarem o progresso atual ou futuro para alcance dos objetivos de aprendizagem definidos. Os *badges* envolvem muitos dos princípios conhecidos para envolver e motivar as pessoas em jogos. Há um grande interesse em explorar esse elemento de "mecânica de jogo" de *badges* no sentido de aumentar o envolvimento e motivação nos contextos de aprendizagem.

Numa ótica de avaliações, estas podem contribuir para a aprendizagem e compreensão adequada através do fornecimento de *feedback* formativo necessário, bem como outras formas

de medir a aprendizagem do aluno (Abramovich et al. 2013). Mas sobreavaliação e avaliação imprecisa têm consequências negativas na motivação (Stiggins 2002). Os alunos que estão sobreavaliados podem tornar-se motivados a procurar a especialidade em passar a exames em vez de experimentarem uma verdadeira aprendizagem. Avaliação imprecisa é perigosa, pois pode causar uma queda na motivação para aprender para um aluno que, em teoria, estaria a aprender. A motivação é a chave para a aprendizagem, pelo que a vantagem potencial de uma avaliação é determinada pela sua capacidade de manter a motivação tanto na aprendizagem em si como na comunicação da mesma com o aluno (Abramovich et al. 2013).

3.3 Ameaças

As ameaças aos *Open Badges* estão maioritariamente relacionadas com o sistema de validação e integridade da informação.

Segundo é citado no relatório *Aligning the Use of Portfolios with Digital Badging* (Doherty & Sharma 2015), enquanto os *badges* têm valor para o processo de aprendizagem e para o desenvolvimento profissional contínuo, existem potenciais aspetos negativos no processo de *badging*. Exemplos desses aspetos incluem a sobrecarga de currículo, as pressões sobre os empregadores para avaliarem o verdadeiro mérito do *badge* e a procura de *badges* como se fosse uma mercadoria, tendo os alunos foco na procura de *badges* ao contrário de procurarem por área de interesse.

De forma a minimizar os efeitos dos aspetos mencionados anteriormente, é necessário que os alunos compreendam o processo de *badging* e que usem os *badges* no seu currículo de uma forma criteriosa, colocando aqueles que são realmente relevantes para o seu percurso académico e/ou profissional. O valor de um *badge* está relacionado com a credibilidade da instituição que o concedeu e os empregadores já estão sensibilizados para essa situação. Finalmente, a questão dos estudantes colecionarem *badges* em detrimento da aprendizagem, pode ser resolvido através de um *design* adequado de aprendizagem, que garanta que os alunos estão motivados para a aprendizagem, e não apenas para a aquisição de um *badge*.

Os alunos modernos estão envolvidos na educação, formação e desenvolvimento pessoal de novas e emocionantes formas, registando o seu progresso, competências e realizações usando tecnologias diferentes e inovadoras (Pearson Learning Solutions 2013). Com tantas atividades em tantos sistemas, aplicações e locais diferentes, pode ser difícil de obter reconhecimento para as competências que os alunos vão adquirindo e demonstrando - e ainda mais difícil obter esse reconhecimento num formato que os alunos possam usar e partilhar.

A interoperabilidade entre ferramentas de *e-portfolio*, Sistemas de Gestão da Aprendizagem e Sistemas de Informação de Gestão estão a aumentar ao mesmo tempo que surgem formatos *standard* para a transferência de dados, mas estes, na maioria dos casos, requerem um sistema compatível para importar esses registos. Também é fácil, ao exportar e importar dados de

realização, perder os detalhes de validação da conquista e, portanto, a integridade da reivindicação.

Segundo o relatório sobre os *badges* digitais na aprendizagem extra escola (Davis & Singh 2015), os *badges,* como medida de conhecimento eficaz, devem ser reconhecidos por todas as partes interessadas, desde estudantes e professores para admissões oficiais e potenciais empregadores. O facto de a credibilidade ter surgido como o desafio mais importante para os *Open Badges*, reflete o reconhecimento e o potencial sobre os *badges* que ainda não tinham sido realizados.

Estas reflexões sobre os desafios de estabelecer credibilidade são consistentes com o relatório de Rughinis (Rughinis 2013) de que *badges* digitais vão implicar um trabalho de interpretação considerável para funcionarem como objetos de fronteira através dos contextos. O mesmo autor observou que os metadados associados com *badges* fornecem informações com mais diferenças do que as credenciais tradicionais. Ao mesmo tempo, estas diferenças requerem mais esforço por parte do público externo, que deve avaliar e pesar os metadados do *badge* para fazerem uma avaliação da natureza da aprendizagem adquirida.

3.4 Oportunidades

Quando se falam em oportunidades para os *Open Badges*, uma das lacunas centra-se na creditação das entidades emissoras. Segundo (Finkelstein et al. 2013), na disponibilização e partilha de formas tradicionais de credenciação, deve ser introduzido um novo elemento na decisão de quando e onde as conquistas ficarão visíveis.

No que se refere à colaboração entre a comunidade, há ainda um grande caminho a percorrer. A colaboração está incluída neste quadro de oportunidade no sentido de permitir que os alunos tenham momentos para se articularem e se envolverem uns com os outros, em experiências de construção de significado partilhado (Gamrat & Zimmerman 2015).

Um dos fatores a ter em conta no futuro dos *Open Badges* prende-se com a componente de personalização da aprendizagem. Segundo (Gamrat & Zimmerman 2015), o subcomponente de personalização é alto quando um aluno é capaz de ajustar e tomar decisões sobre a sua própria educação, como temas de interesse e definição de objetivos. A dimensão de personalização permite que o *earner* selecione o caminho que ele procura e que vai de encontro aos seus objetivos pessoais e profissionais.

Segundo o mesmo autor (Gamrat & Zimmerman 2015), a contextualização das aprendizagens é um componente importante na motivação dos alunos (*earners*), uma vez que quando um aluno é exposto a situações de aprendizagem que são concebidas para serem realistas ou relevantes para a sua vida pessoal ou profissional, o mesmo consegue ter maior perceção do valor da competência para o seu dia-a-dia.

Apesar de incentivar a aprendizagem autodirigida é antes de tudo um método de ensino e aprendizagem e, portanto, pode ser fomentado nos mais diversos ambientes e por uma ampla gama de ferramentas. Muitas das características concretas que definem *badges* digitais tornam-nos adequados para estimular a procura de percursos individualizados de aprendizagem.

Relativamente ao tema da validação e autenticidade, um *badge* é, essencialmente, um certificado *online* oferecido ou endossado pelo emissor que confirma a veracidade dos dados. A validação diz respeito a três partes *earner*, emissor e observador, fomentando um sentimento de confiança no processo pelo qual os *badges* são concedidos e recompensados. Qualquer perceção de que um sistema carece de validação adequada pode pôr em causa a integridade de cada conquista.

Em ambientes modernos e flexíveis, baseados em tecnologia de ensino os alunos são capazes de escolher a hora, local e local de seu evento de aprendizagem (Elliott et al. 2014). Enquanto os educadores têm de fornecer todo o material e estruturar as atividades de aprendizagem para garantir que os alunos têm todas as condições para aquisição de competências, a responsabilidade final de as atingir é transferida do instrutor para o aluno. Nesses ambientes mais personalizados alunos precisam de ser mais auto-motivados e auto-dirigidos (Elliott et al. 2014). Um critério fundamental para o sucesso desses ambientes de aprendizagem é o fornecimento de *feedback* válido, para motivação dos alunos nas suas atividades. *Os badges* poderiam também desempenhar um papel fundamental no reconhecimento da aprendizagem prévia, o que forneceria um percurso de aprendizagem mais eficiente.

As micro-certificações e distintivos digitais estão cada vez mais a ser usados como indicadores de realização, competência, conhecimento ou interesse. A micro-certificação digital tem de se manifestar no *badges* para esclarecerem, validarem, descreverem e definirem as competências, conhecimentos e capacidades que os alunos adquiriram (Elliott et al. 2014). O uso de micro-certificações associadas a *badges* digitais está a ganhar força em todo o mundo com muitas instituições a adotarem um quadro *badge* digital para seus alunos (Elliott et al. 2014). *Os badges* e as micro-certificações são projetados para indicarem as competências específicas, conhecimento e capacidades adquiridas pelo indivíduo. Portanto, como, quando e onde eles são exibidos é de importância crítica, logo um portefólio digital fornece um mecanismo útil de estruturação e exposição a expor das suas competências. Em resumo, permitem que os alunos possam ilustrar aos colegas e outros interessados, o seu progresso, assim como indicarem o seu conjunto de competências.

As oportunidades de aprendizagem desbloqueadas pelo aparecimento dos *Open Badges* é um forte trunfo para lidar com os desafios futuros (Doherty & Sharma 2015). Tradicionalmente, um certificado é sempre tratado como uma questão de tudo ou nada, ou se recebe ou não. A introdução dos *badges*, permitiu o aparecimento do conceito das micro-certificações, ou seja, os alunos vão obtendo *badges* à medida que vão adquirindo mais conhecimento e competências em determinadas áreas de estudo. Por exemplo, um aluno que esteja interessado

em tirar um curso de programação *web*, pode querer focar-se apenas em módulos relacionados com as linguagens HTML, Javascript e CSS, e obter os *badges* correspondentes. Desta forma, o aluno pode construir um portefólio *online* com os *badges* amealhados.

3.5 Análise de Valor

A proposta de valor é o elemento base na modelação do negócio, uma vez que sem ela não é possível saber se e quando o negócio vai gerar receita, que parceiros são precisos, a natureza das principais operações e como e de que forma se vão capturar e reter os clientes (District 2012). Segundo o mesmo autor, a proposta de valor deve identificar, de forma rápida, a razão pela qual um cliente deve optar por uma empresa e não pela concorrência. A mesma é constituída por quatro componentes chave, nomeadamente:

- Como se descreve o produto;
- Que tipo de valor ou benefício está associado com essa oferta;
- Quais os clientes alvo;
- De que forma o produto é único.

Apesar de estes componentes serem essenciais, é necessário ter em consideração o valor percebido. Este é, na sua essência, a avaliação geral que o cliente faz relativamente à utilidade do produto, baseado na comparação da oferta do que é efetivamente recebido (Zeithaml 1988). Na prática, este valor é o que o cliente espera obter com o produto ou serviço enquanto que o valor percebido é a apreciação que o cliente faz do benefício em contrapartida com o sacrífico que irá fazer com a escolha do produto.

A concretização de uma proposta de valor ajuda a definir clientes alvo, bem como a identificar e segmentar esses clientes de forma a delinear diferentes estratégias de marketing para os potenciais interessados. Da mesma forma, é possível identificar também os clientes ou segmentos de mercado que seriam de evitar pela falta de identificação com a oferta da empresa.

No projeto *Discovering Badges*, pretende-se desenvolver uma solução para a problemática da descoberta de *badges* para que os utilizadores consigam, no papel de *earner*, encontrarem *badges* relevantes para a construção dos seus percursos personalizados de aprendizagem. Este sistema contemplará uma rigorosa validação e verificação na emissão e angariação de *badges*.

Assim sendo, é possível identificar alguns tipos de valor associados à utilização da solução proposta. Entre eles destacam-se a **inovação**, a **customização**, a **integridade** e **validade dos dados**, o **tempo**, as **características** do produto e o **preço**.

Considerando que o conceito de *Open Badges* é muito recente e que estes surgiram no sentido de colmatar uma falha no sistema de certificação de aprendizagem formal e informal, disponibilizando uma ferramenta agregadora e complementar à Infraestrutura dos *Open*

Badges. Como os objetivos são o de promover as entidades emissoras de *badges* junto dos *earners* mas em primeiro lugar, o de criar uma espécie de comunidade global no âmbito da aprendizagem, torna ilimitadas as possibilidades de construção de um percurso personalizado e de encontro aos interesses dos utilizadores.

Na Tabela 5 são enquadrados, numa perspetiva longitudinal, os benefícios e os sacrifícios do valor para o cliente.

Tabela 5 – Representação dos benefícios e sacrifícios da proposta de valor

	Produto/Serviços	**Relacionamento**
Benefício	Inovação Customização Qualidade Preço	Integridade Segurança
Sacrifício		Tempo/esforço Conflito

Como se pode verificar na tabela anterior, salientam-se como sacrifícios o **tempo** ou **esforço** dedicados na utilização do produto, tanto na pesquisa, angariação, organização de *badges* tanto como na construção e edição do percurso de aprendizagem. O **conflito** é um sacrifício em destaque uma vez que apesar de haver possibilidade de coexistência com as soluções já existentes no mercado (infraestrutura OBI), pode ocorrer algum atrito à mudança para utilizadores que já utilizam as outras ferramentas.

Apesar do conceito não ser totalmente novo, a possibilidade de complementar as soluções existentes no mercado com uma solução interoperável e particularmente orientada para o utilizador no sentido de o ajudar a construir o seu percurso de aprendizagem (**customização**) com um vasto leque de opções, torna a solução realmente **inovadora**.

A realidade atual dos sistemas de informações identifica potenciais riscos na utilização inadequada da solução, mais concretamente na tentativa de se aproveitarem do sistema para distribuírem ou angariarem *badges* de forma inapropriada. Nesse sentido, a plataforma pretende incorporar um mecanismo de validação e análise de *badges* emitidos bem como a possibilidade de revisão e validação dos mesmos por entidade competentes. Tudo isto no sentido de transmitir e assegurar ao utilizador a **integridade** e **validade dos dados** da solução.

A solução pretende fornecer, **gratuitamente**, um conjunto de funcionalidades que permitam responder às necessidades do utilizador. A eventualidade da existência de contas de utilizador pagas, visam mais concretamente oferecer novas funcionalidades ou extensões às já existentes.

Do ponto de vista da análise de negócio do projeto, entende-se que o objetivo da negociação é na cooperação para maximizar os benefícios, logo o cenário de negociação será o integrado, *win-win*, em que ambas as partes ganham e tendem a ver-se como parceiros na persecução de um objetivo comum.

Alguns dos procedimentos a considerar no processo de negociação do tipo *Win Win* (integrado), são nomeadamente:

- Definição clara das expectativas e objetivos;
- Identificação de pontos indiscutíveis;
- Previsão de contraofertas que possa fazer ou receber;
- Conhecimento de todos os pormenores e todos os assuntos envolvidos;
- Antecipação do que a outra parte deseja;
- Definir qual é o máximo/mínimo que vai receber ou dar;
- Estar preparado para explicar o porquê desse máximo / mínimo que vai receber ou dar.

No que concerne a ideia de negócio do projeto *Discovering Badges*, e seguindo a um processo de modelação de negócio, é necessário que o mesmo responda a diversas questões, nomeadamente:

- Quem são os nossos clientes?
- O que valorizam?
- Como vamos chegar até eles?
- Quais as competências necessárias?
- Que tipo de parceiros devemos ter?

Para responder a estas questões, uma forma sistematizada e prática é utilizar o modelo canvas de criação de Modelos de Negócio. Este modelo compreende nove áreas, nomeadamente, Segmentos de clientes, Relacionamento com os clientes, Canais, Proposições de valor, Fluxos de Receita, Principais parceiros, Principais atividades, Principais Recursos e Estrutura de custos.

Em seguida, serão explicadas algumas das áreas para melhor compreensão do modelo de negócio. Relativamente à área de Segmento de Clientes, eles representam os potenciais utilizadores da solução proposta, nomeadamente os *earners* (utilizadores que pretendem angariar *badges*), as entidades emissoras de *badges* e as entidades revisoras. Na área de Proposições de Valor, encontram-se descritas as propostas de valor já mencionadas em cima. No que refere ao Relacionamento com os clientes, é intuito deste projeto o estabelecimento de uma relação de proximidade com os clientes, de forma fornecer um serviço focado nos interesses dos clientes e que essa proximidade com a comunidade ajude a desenvolver ou melhorar processos ou especificações no ecossistema da emissão e angariação de *Open Badges*.

Relativamente aos principais parceiros, destacam-se as organizações que suportam, divulgam ou potenciam a utilização dos *Open Badges*, tais como a *Mozilla Foudation*, *MacArthur Foundation*, *Badge Alliance, Khan Academy*, *P2P University*, HASTAC (*Humanities, Arts, Science, and Technology Alliance and Collaboratory*). Ao mesmo tempo, identificam-se os parceiros que fornecem um serviço como os *IaaS Providers* (*Infrastructure as a Service*) que fornecem serviços

na *cloud* e destacam-se também os fornecedores de tecnologia *Open Source*, onde nomeademente, a Infraestrutura de base dos *Open Badges* está assente.

Nas principais atividades que o projeto deve fazer de forma a fazer que o modelo de negócio funcione está o desenvolvimento e manutenção da plataforma *web* que responde às necessidades da solução proposta para o projeto. Apresenta-se, na Figura 14, o modelo de negócio de *Canvas* elaborado.

No que se refere à análise/modelação e quantificação da criação de valor e considerando o conteúdo do módulo de análise de valor, entende-se que o método analítico mais adequado seria o do apoio multicritério, mais concretamente o AHP. Este processo é uma técnica estruturada para lidar com decisões complexas. Para o utilizar, é necessário definir uma árvore hierárquica constituída por objetivo, critérios e alternativas. Considerando o âmbito do projeto *Discovering Badges*, determinamos o objetivo que será a plataforma *web* agregadora do sistema OBI e os critérios que serão a customização, o preço, a integridade de dados, o grau de confiança/segurança e as funcionalidades existentes. As alternativas são as soluções relevantes do mercado, mesmo que só se abranjam um *subset* das funcionalidades pretendidas para o projeto. Relativamente ao processo de tomada de decisão, teriam de ser previamente efetuadas comparações par a par de todos os critérios do género Customização *versus* Preço, por exemplo. Após a comparação baseada nos *inputs* do utilizador, são calculadas as pontuações de preferência para cada critério. Sobre esta estrutura é aplicado uma fórmula que determina o coeficiente que indica a alternativa vencedora. Este processo permite assim posicionar o projeto *Discovering Badges* relativamente às soluções identificadas no mercado.

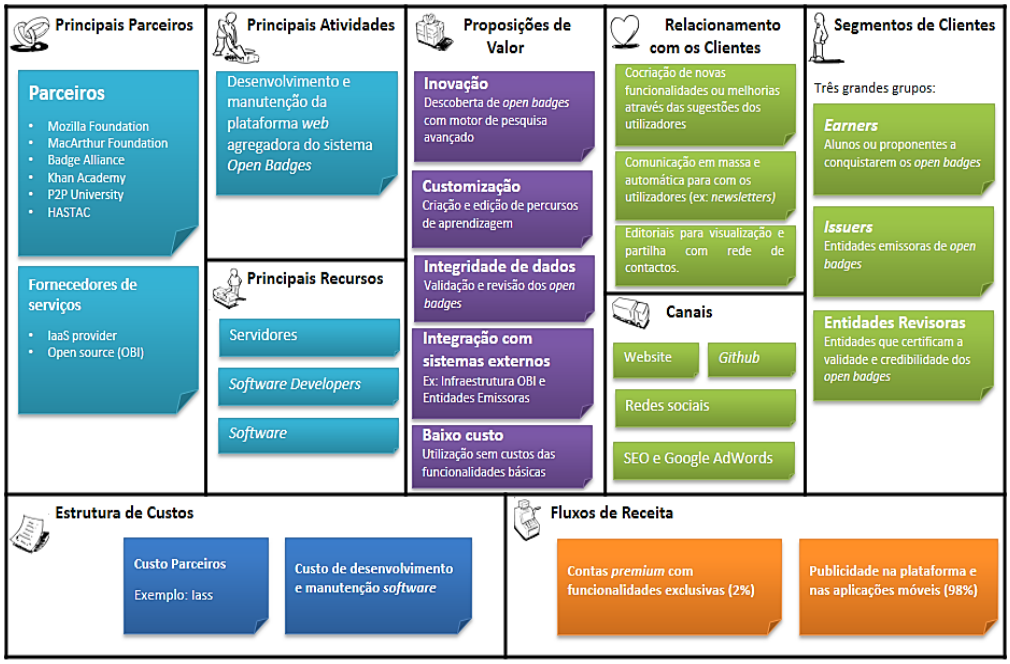

Figura 14 - Canvas do modelo de negócio do projeto *Discovering Badges*

4 Arquitetura da Solução

A fase de definição da arquitetura de uma solução a desenvolver é essencial para o sucesso do projeto, pelo que teve de ser realizada com especial atenção aos detalhes, nomeadamente ao tipo e formato de dados que é suposto a plataforma lidar. Nas seções seguintes, é demonstrado o culminar desse trabalho através da proposta de uma arquitetura base para a solução a implementar.

4.1 Conceitos

Segundo Newman (Newman 2015), os microserviços são pequenos agentes autónomos que trabalham em conjunto e que se vêm tornando cada vez mais utilizados nas mais recentes arquiteturas de software. Algumas empresas como a NetFlix, SoundCloud e Twitter têm apostado em explorar este conceito nas suas soluções e stêm-se destacado pela evolução das suas arquiteturas para microserviços. Autores como Lewis e Fowler ((Lewis & Fowler 2014)), descrevem os microserviços como uma forma de desenvolver *software* como um conjunto de serviços independentes, reutilizáveis e com funcionalidades muito bem definidas. Cada serviço é executado num processo independente, que pode ser implementado como um serviço isolado, uma plataforma para serviços ou mesmo um próprio processo do sistema operacional (Newman 2015). A comunicação entre os serviços pode ser efetuada com recurso a APIs e recorrendo ao protocolo HTTP. Para justificar a adoção de uma arquitetura por microserviços, é importante comparar com as arquiteturas monolíticas que atuam como um bloco único e, caso a caso, determinar qual a melhor abordagem. Esta última é adequada para pequenas aplicações, pois o desenvolvimento e teste é relativamente simples. No entanto, para

aplicações de maiores dimensões e complexidade, a arquitetura monolítica torna-se um obstáculo ao desenvolvimento, introdução de novas funcionalidades e tecnologias. Neste tipo de arquiteturas, qualquer alteração ao sistema, envolve a criação e implementação de uma nova versão do *software*, o que terá um impacto significativo em toda a aplicação, dificultando o lançamento de novas versões (Lewis & Fowler 2014). A adesão a uma arquitetura por microserviços, propõe a decomposição de sistemas monolíticos num conjunto de serviços autónomos. Com esta aborgadem, é possível alterar um serviço de uma forma isolada, logo sem dependências o que confere características de descentralização ao sistema. Os microserviços são também mais fáceis de compreender porque o seu âmbito está diretamente ligada a um domínio da aplicação. Na Figura 15, é representada a diferença entre um sistema monolítico e o sistema dividido em microserviços.

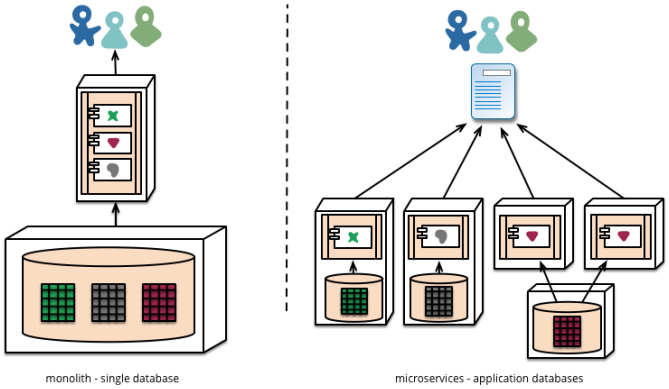

Figura 15 – Comparação entre um sistema monolítico e um orientado a microserviços (fonte: http://martinfowler.com/articles/microservices.html)

4.2 Arquitetura proposta

O que se propõe é uma arquitetura de microserviços com uma API pública que permita a comunicação bidirecional e em que a plataforma proposta seria um cliente dessa API, no sentido de criar a rede global e permitir, tanto a *issuers* como *earners*, beneficiarem da partilha e recolha de informação. Os *badges* dos *issuers* iriam ser mais divulgados, tanto a nível de partilha pelo *issuer* como partilha pelo *earner*, bem como para os *earners* que iriam poder construir *pathways* relevantes para os seus interesses de aprendizagem bem como descobrir *badges* de outros utilizadores e/ou *issuers*. A correspondente arquitetura está representada na Figura 16.

O *Recommendation Service*, como o nome o indica, pretende disponibilizar um serviço de recomendações de *badges* para complementar a descoberta de *badges*. Teve subjacente à sua implementação algoritmos de recomendação baseados na amostra de dados existentes no catálogo de *badges* para calcular possíveis *badges* com valor para o utilizador.

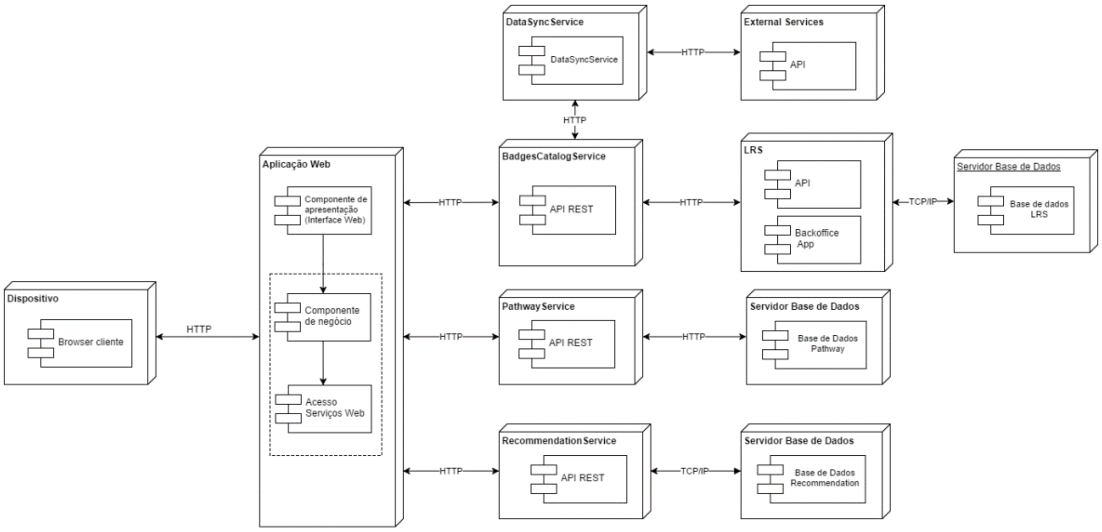

Figura 16 - Diagrama UML da Arquitetura da Solução proposta

O *CatalogSync Service* visa comunicar com LRS externos e outros elementos no sentido de sincronizar *statements* e centralizar a informação num repositório central. O objetivo deste serviço é atuar como intermediário entre LRS/LMS e outros sistemas, encapsulando toda a lógica inerente de transformação, se necessário, do modelo de dados proprietários para a especificação xAPI.

Numa representação mais a nível conceptual, pode visualizar-se na Figura 17 a disposição e distribuição dos microserviços pela correspondente área de atuação.

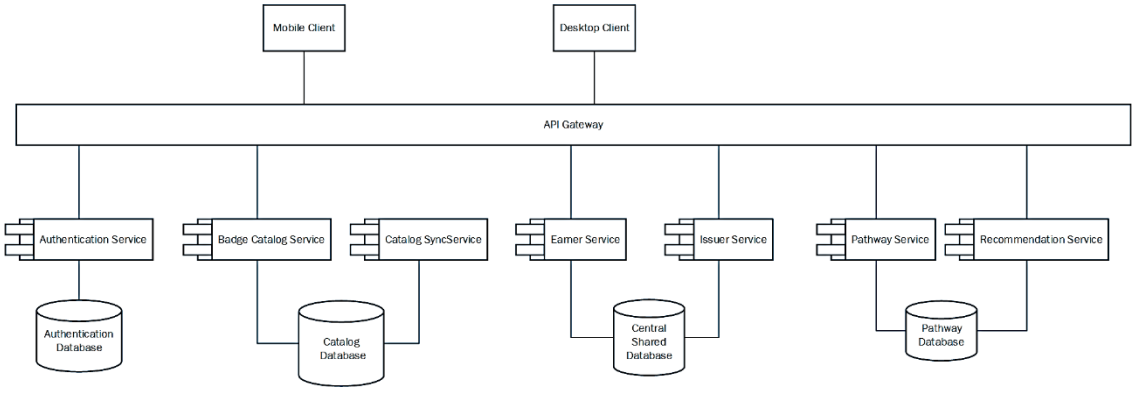

Figura 17 – Diagrama de microserviços para a solução proposta

Para os microserviços e aplicações *web* entende-se que, apesar das diferenças de contexto, devem ser desenvolvidos com base numa estrutura comum, podendo ter ou não, por exemplo, a camada de apresentação. A separação das diferentes aplicações por camadas tem por objetivo a redução da complexidade e potenciar a reutilização de código, sendo por estes motivos, considerada uma boa prática no desenvolvimento de *software*. A arquitetura proposta seguiu os princípios do DDD (*Domain Driven Design*), que se rege por 5 camadas fundamentais:

- Camada de apresentação, como interface ao utilizador e interação com o mesmo;

- Camada aplicacional, que tem por objetivo de ser uma camada intermédia entre as camadas de apresentação e de domínio. Tem as responsabilidades de coordenação das transações com camadas inferiores, validação de dados e questões de segurança, devendo por isso armazenar o estado do progresso das transações sem nunca apresentar ou manter regras de negócios.

- Camada de domínio, que inclui as entidades de domínio e as suas regras. É responsável pela representação das entidades de negócio e pela implementação das consequentes regras de negócio.

- Camada de dados disponibiliza acesso a dados no contexto do componente bem como a dados exteriores ao componente, tal como serviços *web* ou sistemas externos.

- Camada infraestrutural, que suporta as camadas superiores com as suas competências transversais. Tem a responsabilidade de disponibilizar operações de autenticação, autorização, *cache*, gestão de exceções, *logging*, IoC. Contudo, uma camada extra pode ser necessária, como a Camada de Serviços distribuídos, como neste caso através de ferramentas como a ASP.NET Web API para fornecer esta camada.

Na Figura 18 é apresentado o diagrama UML para a divisão de camadas indicada anteriormente. Este diagrama serviu de apoio para a implementação dos microserviços, como uma boa prática de Engenharia de *Software*.

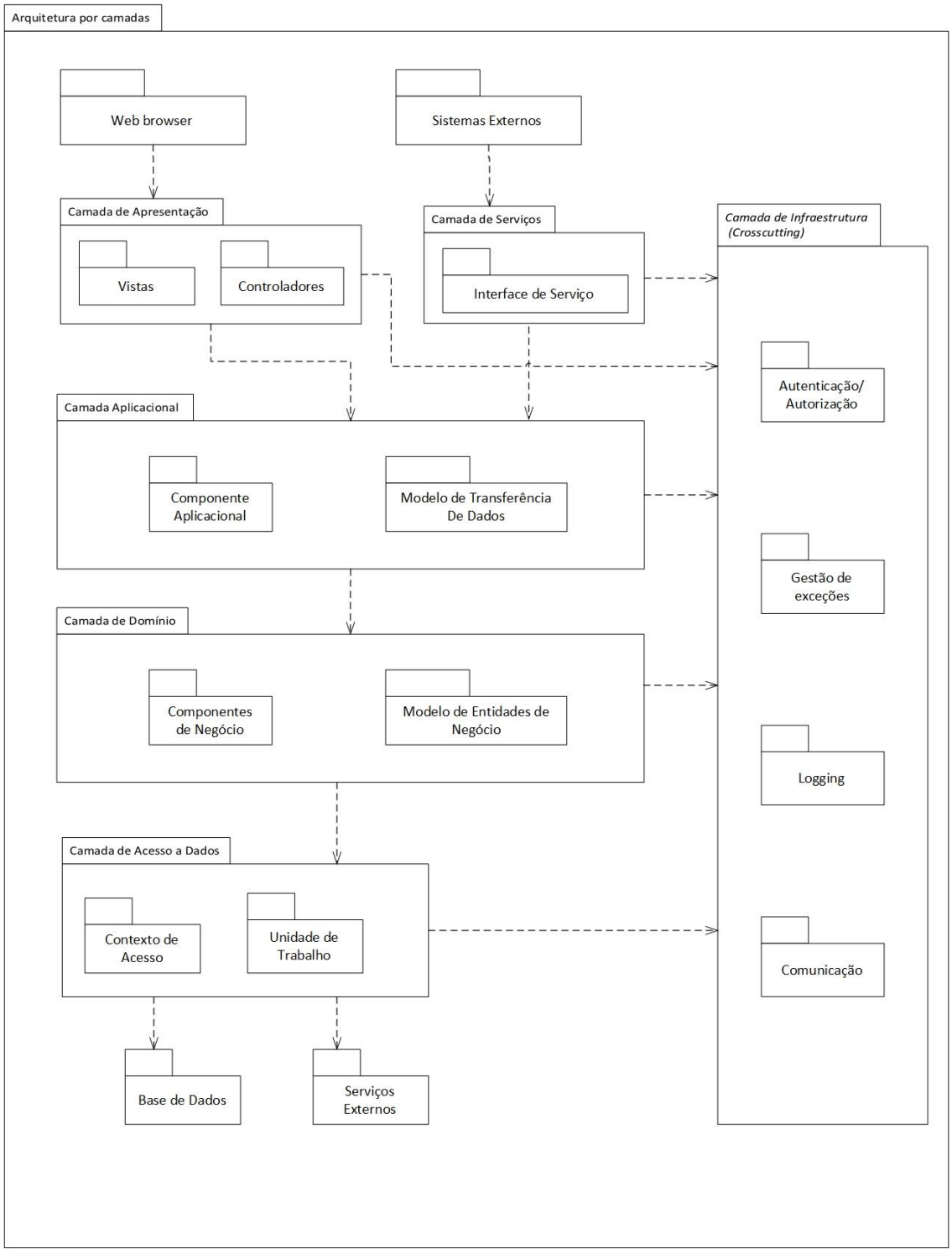

Figura 18 - Diagrama UML da arquitetura por camadas

4.3 Padrões arquiteturais

Algumas formas de resolver problemas plenamente documentadas e boas práticas de Engenharia, foram utilizados racionalmente e com propósitos bem definidos no âmbito deste projeto.

Na API pública, utilizou-se um padrão denominado API Gateway. Este padrão visa retirar dos microserviços as responsabilidades de roteamento, autenticação, transformação de dados. De forma, ao retirar complexidade dos serviços e aumentar a consistência de implementação dos mesmos, atribuem-se estas responsabilidades a um componente denominado *gateway*. Esta, permite centralizar, gerir e monitorizar aspetos relacionados com os requisitos não funcionais da solução proposta bem como reduzir *round trips* às diferentes aplicações *web*.

Ao nível da arquitetura de camadas proposta na Figura 18 esta foi baseada no *Single Responsability Principle* que determina que cada módulo ou classe deve ter responsabilidade por uma parte bem definida das funcionalidades disponibilizadas pela aplicação. Para se atingir essa separação de responsabilidades, recorreu-se ao padrão o *N-Layer*. Este padrão assenta em diversos princípios como a abstração, o encapsulamento, a alta coesão, o baixo acoplamento e a disponibilidade de reutilização de componentes.

Outros dos padrões arquiteturais utilizado foi a Inversão de Controlo, também denominada de *Dependency Injection*, uma vez que este padrão permite tornar uma aplicação desenvolvida mais modular e por isso mais fácil de testar unitariamente.

4.4 Camada de Acesso a Dados

Esta camada de *software* tem a principal e fundamental responsabilidade de persistir a informação do sistema. Por este motivo, foram definidos alguns pontos-chave:

- Independência do Sistema de Base de dados a utilizar
- Segurança
- Estabilidade
- Rapidez

O primeiro ponto referido assenta como base de todo o desenvolvimento. Havia a necessidade de tornar o acesso/leitura/escrita de dados independentemente da forma como são de facto persistidos.

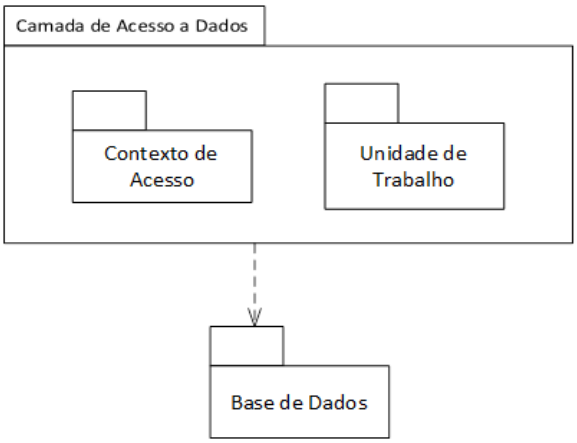

Figura 19 - Diagrama de componentes UML da camada de acesso a dados

Através da observação da Figura 19 é possível verificar que a camada de acesso a dados é composta por dois componentes principais: Contexto de Acesso e Unidade de Trabalho. O pacote Contexto de Acesso representa uma ligação ativa à base de dados em uso, permitindo leituras em *lazy load* e transações de dados. O pacote Unidade de Trabalho representa um conjunto de repositórios de informação responsável por efetuar a persistência de dados utilizando o Contexto de Acesso. Salienta-se ainda que o conteúdo deste componente resulta da adoção de um conjunto de padrões de *software* (*Adapter*, *Factory*, *Repository* e *Unit Of Work*). O resultado desta combinação é uma estrutura eficiente e dinâmica, ideal para a solução pretendida. A utilização destes padrões é gradualmente detalhada ao longo das secções seguintes.

4.4.1.1 Interfaces de Repositórios

A necessidade da existência de interfaces de código que permitisse realizar operações de persistência de informação de forma independente do sistema de base de dados em utilização, Para separar responsabilidades e separar conceitos de negócio ou entidades, foi aplicado o

padrão *Repository*[34]. Com esta implementação é criada uma interface para cada entidade do modelo de dados. São assim garantidas as operações CRUD[35] para cada tipo de entidade.

4.4.1.2 Repositórios

A implementação dos repositórios de dados faz uso do Contexto de Informação para aceder e manipular informação. Compete à Unidade de Trabalho, no momento da instanciação, fornecer o contexto de acesso. Com este último, o repositório irá realizar todo o trabalho necessário. Na Figura 20 é apresentado um diagrama de sequência ilustrativo da utilização destas implementações e a utilização do Contexto de Informação por parte das mesmas.

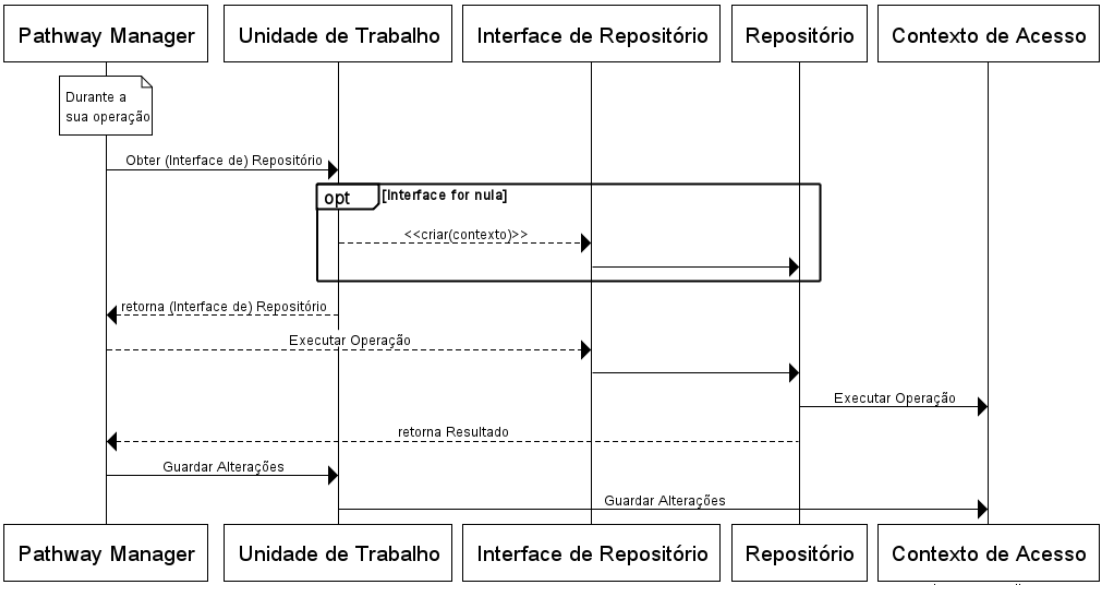

Figura 20 - Diagrama de sequência UML representativo da implementação de um Repositório

4.4.1.3 Unidade de Trabalho (*Unit of Work*)

Com a necessidade da existência de um mecanismo conjugador e transversal a todas as interfaces, surge assim o padrão *Unit Of Work*[36]. Na Figura 21 está representada a relação de associação entre os Controladores, o Gestor de Contexto de Acesso e a Unidade de Trabalho (classe implementadora do padrão *Unit Of Work*). A mesma tenta ilustrar a intrínseca relação entre as classes Coordenador e a Unidade de Trabalho. Todas as classes Coordenador fazem uso do Gestor de Contexto de acesso que por sua vez utiliza uma instância da classe Unidade de Trabalho para permitir efetuar as operações de leitura e escrita na base de dados.

A Unidade de Trabalho torna este processo possível através do conjunto de interfaces de repositórios a si associados. Desta forma, é garantido o encapsulamento tendo a responsabilidade de instanciar a implementação de interface correta. Para os Coordenadores é garantida independência lógica da implementação e entre camadas lógicas com o Gestor de

[34] Repository – Padrão de Software
[35] CRUD – *Create*, *Read*, *Update* e *Delete*
[36] *Unit Of Work* – Padrão de Software

Contexto. Na sequência da funcionalidade do Gestor de Contexto de Acesso verificou-se que esta classe deveria representar uma transação para com a base de dados. Assim sendo aqui são também instalados mecanismos de gestão de sessão de conexão, que são abordados na secção seguinte, de forma a poder guardar todas as alterações no seu conjunto ou reverter todas essas mesmas alterações.

4.4.1.4 Contexto de Informação

O Contexto de Informação representa uma implementação concreta para a estrutura abordada anteriormente (*Adapter* + *Repository* + *Unit Of Work*). A sua função é garantir um contexto (sessão) de conexão à base de dados para pedidos de leitura e/ou escrita. É também a classe responsável pela implementação e mapeamento do modelo de classes para modelo de dados e implementará uma outra classe do tipo *"Database Context"*, classe constituída por código de acesso a base de dados e comandos gerais de acesso a dados. As suas propriedades são do tipo *"Database Set"* e cada uma destas representa uma tabela de dados.

Numa perspetiva global, este *core* é constituído segundo a Figura 21. Segundo esta, o Contexto de Informação é utilizado pela Unidade de Trabalho para instanciar os repositórios de acesso. Por sua vez, estes utilizam o Contexto de Acesso para executarem as suas operações. No final, compete ao *Manager* o pedido à Unidade de Trabalho sobre o Contexto de Informação para guardar todas as alterações efetuadas.

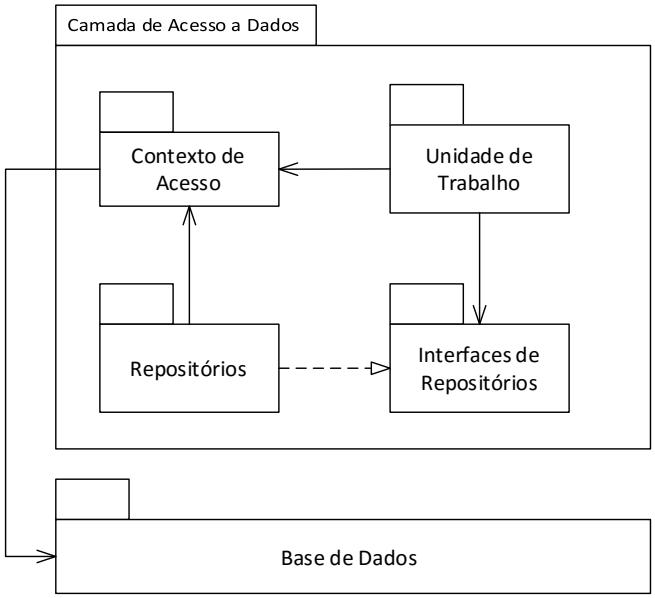

Figura 21 - Visão geral da Arquitetura implementada para o *core* de acesso a dados nos serviços *web*

5 Requisitos

No presente capítulo é descrita a fase de conceção da solução a desenvolver, identificando as diferentes abordagens consideradas, os requisitos funcionais e não funcionais, o modelo conceptual e uma visão arquitetural da solução proposta.

5.1 Solução pretendida

O projeto *Badges For All* visa responder ao problema da descoberta de *badges* relevantes, pretendendo-se o desenvolvimento de um sistema compatível com a OBI e que garanta a integridade e relevância dos metadados dos *badges*, permitindo a pesquisa eficaz de *badges* por parte dos utilizadores na construção do percurso do conhecimento.

Os constrangimentos arquiteturais da OBI no âmbito do problema deste projeto impedem que, de momento, se crie uma rede global de emissão e descoberta de badges. O sistema ideal tem de ser flexível ao ponto de permitir que se utilizem os processos anteriores, tanto do *issuer* como do *earner*, mas que ao mesmo tempo permita recolher essa informação para construção da rede global. A segmentação existente nas soluções da *Mozilla OBI* e de outras do mercado dificulta a resposta ao problema da pesquisa e descoberta de *badges.*

Assim, pretende-se desenvolver uma solução baseada na especificação técnica dos Open Badges e na infraestrutura OBI mas que permita um ambiente integrador, de forma a centralizar, estruturar e relacionar a informação dos *badges* simplificando a pesquisa e descoberta dos mesmos. Para esse efeito, tem-se por objetivo experimentar a utilização da especificação xAPI e do seu ecossistema para se atingirem os objetivos apresentados.

5.2 Requisitos

5.2.1 Atores do Sistema

Consideram-se três os atores possíveis do sistema e que são os intervenientes da solução pretendida, nomeadamente o *issuer*, o *earner* e o revisor. Em seguida são identificados os papéis gerais de cada um deles:

- Emissores, que criam os *Open Badges* para a realização de tarefas e atividades, bem como a demonstração de habilidades e competências.
- *Earners*, que são indivíduos que se propõe a completar as tarefas, demonstram competências e consequentemente amealham *Open Badges.*
- Revisores, que são as entidades ou indivíduos que atuam como *endorsement* dos *badges* e asseguram a integridade dos mesmos

5.2.2 Requisitos funcionais

Considerando os três atores do sistema, *issuer*, *earner* e revisor descrevem-se nesta subsecção, as funcionalidades a desenvolver no âmbito deste projeto com foco no *earner*. Contudo, no

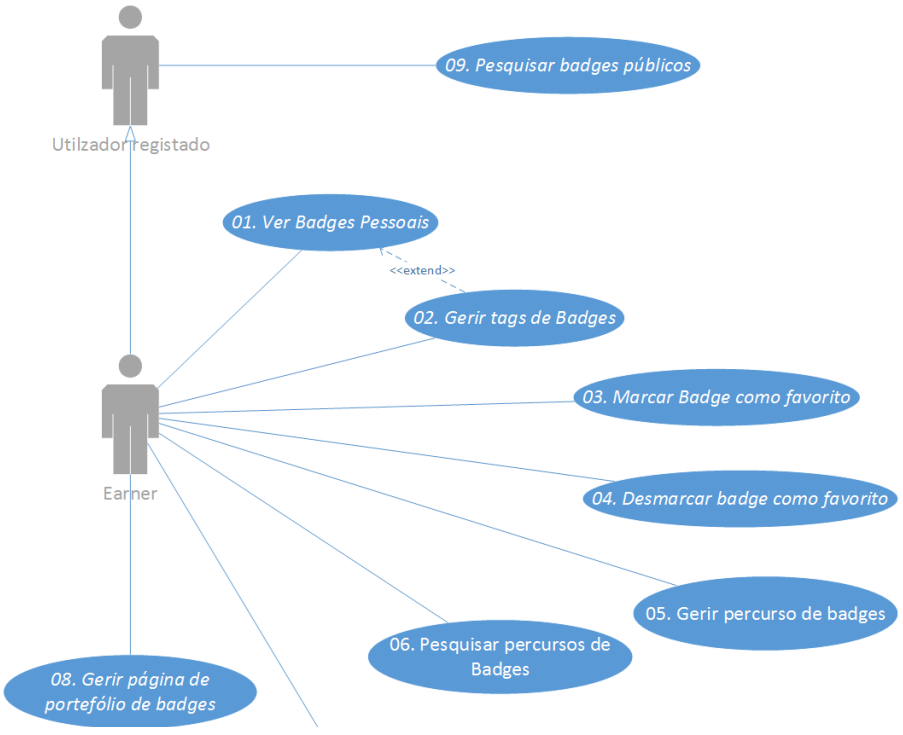

Figura 22 - Diagrama de casos de uso para a solução proposta

Anexo B, está representada a listagem completa de funcionalidades previstas para a solução completa. O subconjunto de casos de uso descritos na Figura 22, foram identificados como os essenciais para a exploração e implementação do protótipo de resposta ao problema da descoberta de *badges* com foco no *earner*, como o ator com maior impacto direto.

Na Tabela 6 estão representados os casos de uso por ator com uma breve descrição do mesmo.

Tabela 6 – Descrição dos casos de uso para a solução proposta

UC	Ator	Descrição
01	*Earner*	Ver *badges* pessoais
02		Adicionar e remover tags dos *badges*
03		Marcar *badge* como favorito
04		Desmarcar *badge* como favorito
05		Gerir percurso de aprendizagem, com criação e edição do mesmo
06		*Pesquisar percursos de aprendizagem públicos*
07		Ver sugestões de *badges*
08		Gerir página de portefólio de *badges*
09	Utilizador registado	Pesquisar *badges públicos*

5.2.3 Requisitos Não funcionais

Os requisitos não funcionais, apesar de não estão diretamente relacionados com as funcionalidades a implementar, são de extrema importância para a solução a desenvolver, uma vez que caracterizam o sistema do ponto de vista de segurança, fiabilidade, usabilidade, entre outros, que são fatores essenciais na determinação da qualidade da solução desenvolvida.

Desta forma, são apresentados na Tabela 7 os requisitos não funcionais identificados:

Tabela 7 – Requisitos não funcionais

Objetivo de negócio	Requisito	Atributos	Cenário de Qualidade	Prioridade
Suportar utilizadores de todo o Mundo	Suporte a novas linguagens	Modificabilidade Escalabilidade	Uma versão da aplicação com suporte a uma nova linguagem está disponível em 5 dias.	Alta
Suportar novos formatos	Extensibilidade do sistema para lidar com diferentes	Modificabilidade Expansibilidade	Uma versão da aplicação com suporte a novos	Média

	especificações de *statements*		formatos de *statements* está disponível em 15 dias.	
Pesquisa avançada com boa performance	Suporte a sistemas de *caching* e indexação no servidor e base de dados	Desempenho Eficiência	Uma versão da aplicação com suporte e configuração de sistema de *caching* e indexação está disponível em 15 dias.	Alta
Suporte a novos sistemas de *Login*	Suporte a sistemas de *Login* por sistemas externos (Google+, Facebook, Twitter,…)	Modificabilidade Usabilidade	Uma versão da aplicação com suporte a novos tipos de *Login* estará disponível em 15 dias por cada sistema de *Login*.	Média

6 Desenho da solução

No presente capítulo é descrita conceção da solução a desenvolver, com consideração dos requisitos anteriormente apresentados. Apresentam-se o modelo conceptual, pretende-se demonstrar a estrutura de dados a ser implementada, descrevem-se o funcionamento dos serviços e da aplicação *Web*.

6.1 Modelo Conceptual

No que se refere ao modelo conceptual para a solução proposta, idealiza-se que o sistema deva ser modular, flexível e interoperável. Nesse sentido, modulou-se o sistema com base nestes princípios. Na Figura 23 está representado esse conceito.

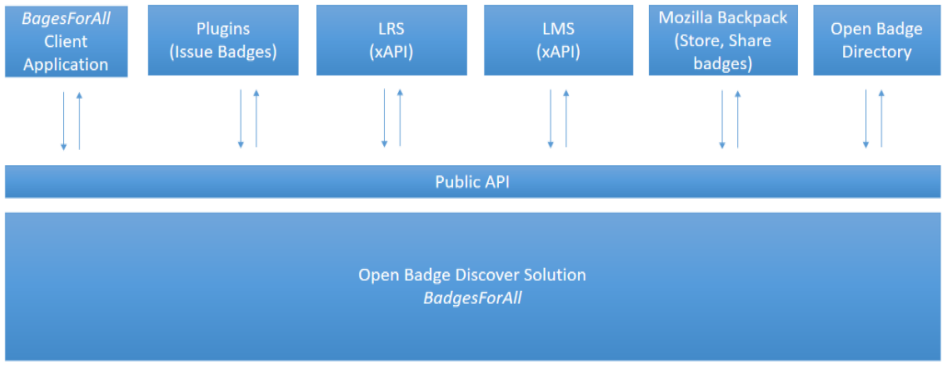

Figura 23 - Modelo Conceptual da solução proposta

Como é possível observar na figura anterior, o objetivo do sistema é usar um conceito de *multi-tenancy*. Este conceito promove a uma maior eficiência dos sistemas e permite, entre outras coisas, que um grupo de utilizadores comuns (aplicações cliente) comunique com o sistema de

forma completamente diferente comparativamente a um outro grupo. As responsabilidades inerentes são diferentes de grupo para grupo e consiste em instâncias independentes operarem num ambiente partilhado. No modelo desenhado, entende-se que através de uma API pública seja possível, num ambiente integrado, diferentes aplicações operarem sobre o sistema. Esta partilha tem benefícios evidentes na centralização de dados de *badges* mas ao mesmo tempo na separação lógica e modular das diferentes aplicações.

A própria aplicação de *frontend* de resposta aos requisitos funcionais descritos, atuaria, ela própria, sobre o sistema como uma instância do ambiente partilhado.

6.2 Modelo de Domínio

Na Figura 24 está representado o modelo de domínio base que foi identificado para os *Open Badges* no contexto da xAPI, v1.0.1. Este, demonstra claramente a representação de um *Statement* como um trinómio "Ator /Verbo / Objeto". O Ator é a pessoa que adquiriu o *badge*, o Verbo descreve a ação que foi executada sobre o Objeto que é por sua vez uma atividade de algum tipo no domínio da aprendizagem, formal ou informal.

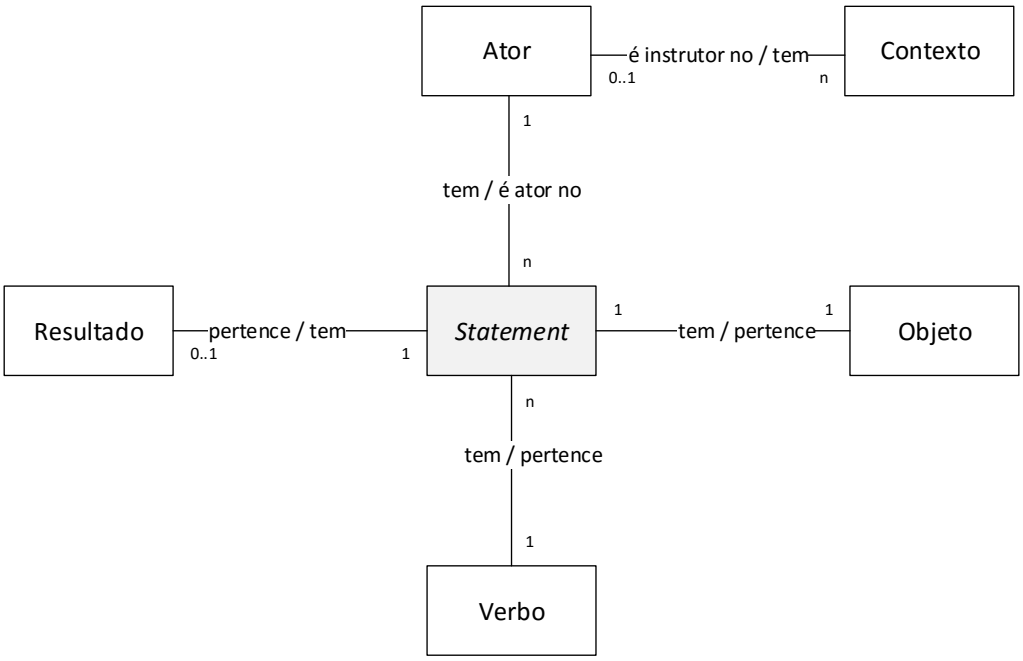

Figura 24 – Modelo de Domínio base dos *Open Badges* no contexto xAPI

6.3 Modelo de dados

Relativamente ao modelo de dados, destaca-se o modelo de domínio associado aos elementos representadores da aprendizagem, os *Open Badges*, que através da utilização da receita para adaptação ao modelo xAPI, secção 2.6.5, resulta no diagrama de classes da Figura 25 para os S*tatements*.

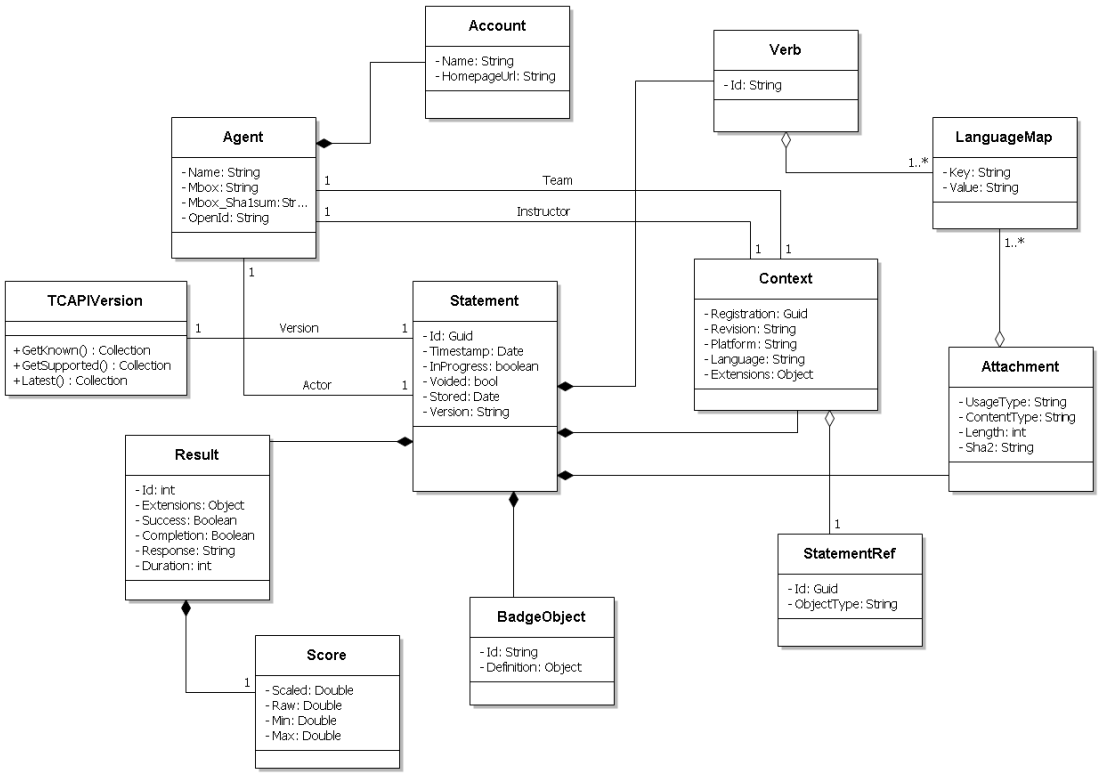

Figura 25 – Diagrama de Classes UML para a representação dos *Statements*

Este modelo foi desenvolvido tendo em conta especificação de *Statements* descrita pela ADL[37], e será a base para validação e mapeamento JSON para o serviço de catálogo de *Badges* no formato xAPI. Destacam-se as entidades *Statement* como elemento representativo da conjunção da informação, do *BadgeObject* que representa a definição do *bade* e onde se encontrará maior parte da informação relacionada com a especificação dos *Open Badges*, A entidade *Agent* que visou representar Atores e Emissores e o *LanguageMap* cuja importância está relacionada com o facto de ser possível inserir descrição a determinados elementos de informação em várias línguas.

[37] https://github.com/adlnet/xAPI-Spec/blob/1.0.1/xAPI.md#statement

6.4 Bases de Dados

Relativamente à base de dados, e seguindo a estratégia dos microserviços, é possível escolher o tipo de base de dados de acordo com as necessidades de cada área. No que se referem aos diferentes tipos de base de dados, destacam-se as bases de dados do tipo Relacional (exemplo: tradicionais base de dados SQL), do tipo Documental (ex: *MongoDB*) e do tipo Grafo. O tipo de base de dados a escolher para cada caso está relacionado com os pontos fortes de cada um desses tipos. Na Tabela 8 estão representados os tipos de bases de dados escolhidos para a solução proposta.

Tabela 8 – Tipos de Bases de Dados para a solução proposta

Base de Dados	**Tipo**
Authentication Database	Relacional
Badge Catalog Database	Documental
Central Shared Database	Relacional
Pathway Database	Relacional/Grafo

Apesar de as bases de dados do tipo relacional serem as mais comuns e mais utilizadas tradicionalmente, existem contextos onde outros tipos são mais adequados. Por exemplo, as bases de dados do tipo documental são as ideais para catálogos de conteúdos e consequentemente permitem leituras mais rápidas.

Por outro lado, as bases de dados do tipo Grafo são ideais quando se pretende representar os dados e as suas conexões, podendo ser desta forma uma boa representação dos *badges* e da relação com *issuers*, *earners* e *pathways*. Permite assim ser útil na construção de um mecanismo de sugestões para o *earner* na construção do seu percursos de aprendizagem (*pathway*).

6.5 Serviços *Web*

Uma API REST é um serviço que pode trazer grandes vantagens a nível de extensibilidade e escalabilidade, dado que permite uma grande facilidade na integração com sistemas internos e externos. Para garantir escalabilidade é necessária uma arquitetura por camadas tal como as aplicações *web*, ou seja, pode existir uma camada responsável por receber os pedidos de clientes e outra para aceder a um ou vários recursos de forma a fornecer a resposta esperada (Subbu Allamaraju 2011).

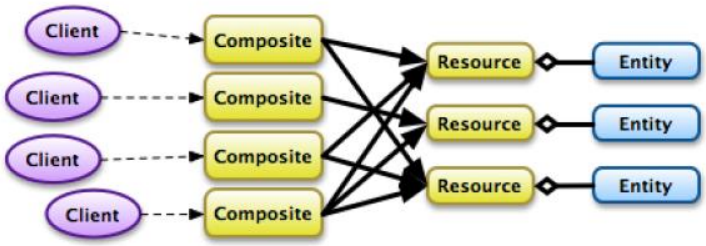

Figura 26 – Diagrama de uma API REST com múltiplas camadas (Subbu Allamaraju 2011)

Se pensarmos numa API como uma aplicação web comum, a alta escalabilidade pode ser garantida utilizando uma aproximação ao padrão arquitetural Model-View-Controller.

Uma API REST deve ser sempre acompanhada de uma manual de utilização ou documentação, onde são explicados quais os métodos existentes e a forma de os invocar, como é possível ilustrar na Figura 27. Assim sendo, do ponto de vista dos utilizadores finais, a utilização de uma API é um processo bastante simples e de fácil utilização que consiste na realização de pedidos HTTP e interpretar/tratar as respostas do servidor. Apesar de ser uma abordagem de fácil utilização, esta apenas fornece a possibilidade de realização de operações de gestão sobre a informação (leituras e escritas), o que significa que os restantes componentes aplicacionais, como a interface, têm de ser garantidos e preparados pelos utilizadores/sistemas.

Segundo a Yahoo (Yahoo 2013) a técnica a ser utilizada para melhorar a performance é minimizar o número de pedidos HTTP. Pelo facto de a utilização das API's REST ser baseada em constantes pedidos HTTP ao servidor, conclui-se que este princípio é não é assegurado o que é prejudicial à performance dos serviços ou mesmo de toda a infraestrutura aplicacional.

Com o crescimento de utilização da API, o número de aplicações desenvolvidas através da mesma irá, consequentemente, aumentar levando assim ao aumento do número de pedidos diários recebidos.Com o aumento do número de pedidos diários, a carga do servidor web e da Base de Dados aumenta. Se nenhuma ação for tomada no sentido de balancear esta carga, o tempo de resposta irá consequentemente crescer.

As API's REST são muito utilizadas no contexto web, mas têm ganho muita popularidade no desenvolvimento de aplicações para dispositivos móveis, pelo facto de os dados serem passados em formatos leves com um baixo consumo de largura de banda (JSON).

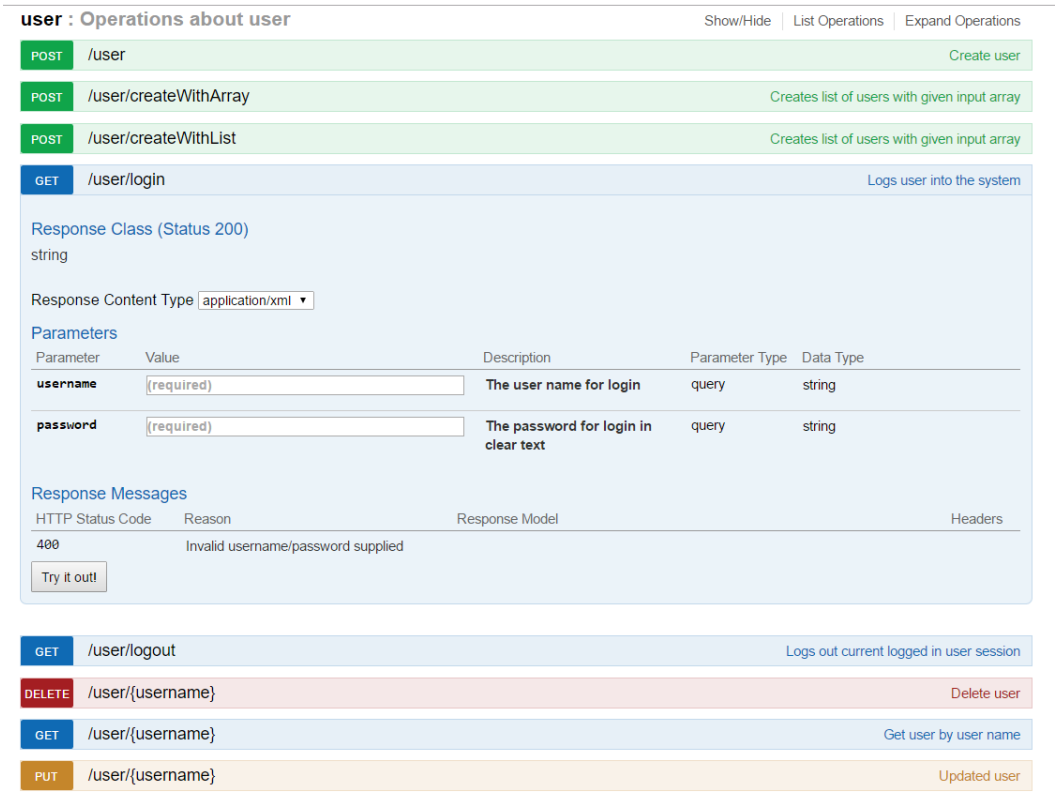

Figura 27 – Documentação típica de uma API REST (fonte: http://petstore.swagger.io/)

6.5.1 Badge Catalog Service

O *Badge Catalog Service* surge, no contexto do problema, como um componente com os objetivos de dar suporte à pesquisa e submissão de *Open Badges*. Uma vez que foi escolhida a adaptação dos *badges* à especificação xAPI, este serviço apenas permitirá comunicação de dados neste formato. A escolha deste formato, permitirá que este sistema garanta interoperabilidade com outros repositórios de aprendizagens usando a mesma especificação ou mesmo um mapeamento mais simples com sistemas de especificação diferentes.

Este serviço será um serviço *web* público com chave partilhada e surgirá como componente intermédio da comunicação com o LRS do sistema. A comunicação com o LRS seria possível efetuar de uma forma direta, no entanto, no âmbito dos *Open Badges* este componente foi idealizado para permitir maior flexibilidade no futuro, tanto por substituição do LRS por outro ou por um sistema diferente dos LRS tradicionais, um sistema de base de dados próprio ou mesma outra tecnologia. Desta forma, é possível verificar na Figura 28, um diagrama a título de exemplo da forma de comunicação entre o serviço e o LRS.

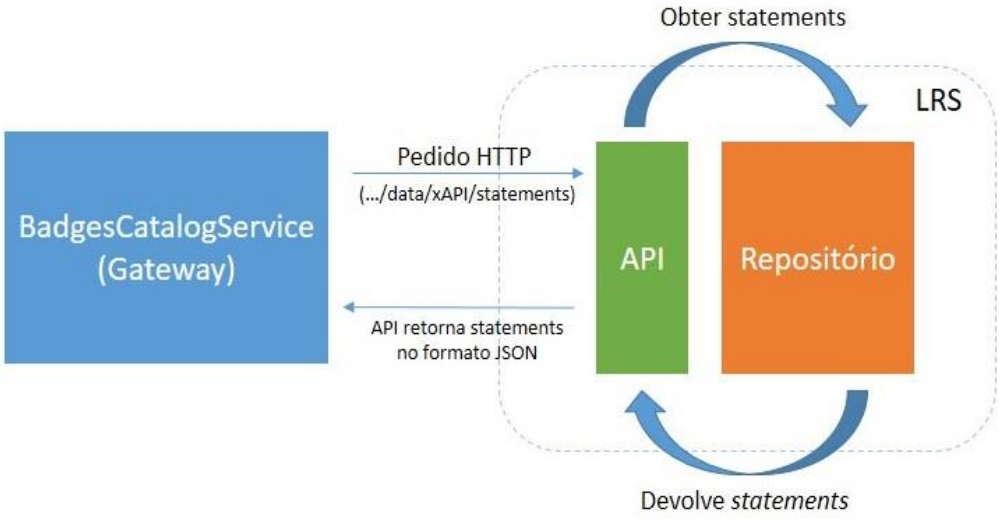

Figura 28 - Diagrama de comunicação entre a *gateway* do serviço e o LRS externo

A título de exemplo, é possível verificar na Figura 29 o funcionamento do serviço para a criação e pesquisa de *statements*. Na criação de *statements,* representação xAPI de um *Open Badge*, é possível verificar, na Figura , a sequência de validação da execução do pedido de criação que o serviço irá realizar. Essas validações têm contextos distintos, sendo a primeira associada à existência de autenticação válida no pedido ao serviço e a segunda associada à integridade e validade dos dados inseridos (xAPI *compliant)*. Estas validações garantem a segurança e integridade do sistema.

Após a fase validação são enviados os dados para o destino, LRS ou outro serviço externo. O identificar do serviço de destino está relacionado com o parâmetro do pedido, que se irá denominar *Tenant*, e que não é mais do que uma entidade interna ou externa relacionada com o sistema. A entidade por defeito é a do sistema desenhado, mas outros serviços externos podem "viver" no mesmo e permitir que os seus utilizadores utilizem as ferramentas desenhadas. O objetivo é, acima de tudo, garantir a interoperabilidade.

De forma a garantir a comunicação entre a *gateway* e um serviço externo, LRS ou outro serviço externo, resolveu-se desenhar esse componente de integração utilizando o padrão *Adapter*. Desta forma, para o *tenant* associado ao pedido, é redirecionado para a implementação da *gateway* que responde a esse pedido. A resposta, no entanto, é uniformizada de acordo com a especificação xAPI. Desta forma, garante-se que é possível comunicar através do formato xAPI, mas também é possível integrar com outros sistemas externos que não suportam a mesma especificação, como por exemplo API's tradicionais de *Open Badges*. Na Figura 30, é apresentado um diagrama a título de exemplo do esperado na implementação deste componente do serviço.

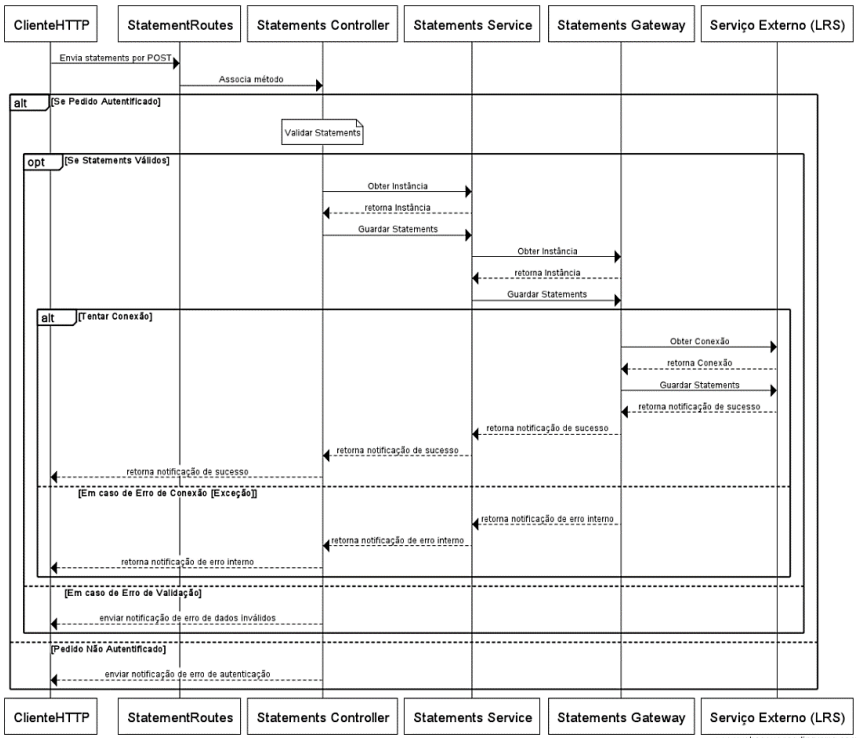

Figura 29 - Diagrama de sequência UML para a criação de *Statements*

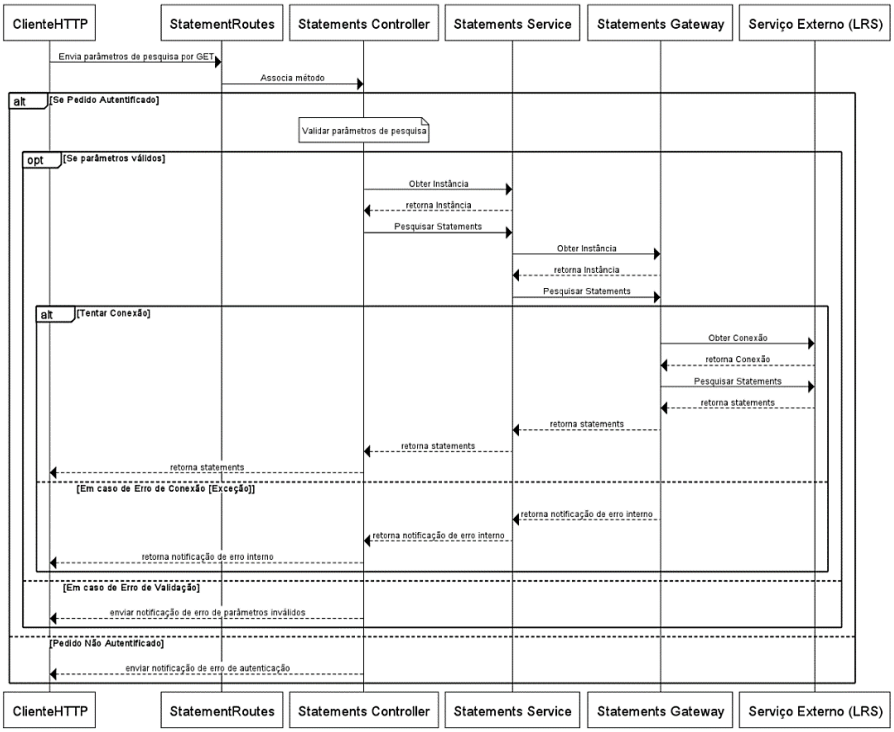

Figura 30 - Diagrama de sequência UML para a pesquisa de *Statements*

Na

Tabela 9, estão representadas as rotas definidas para os pedidos ao serviço de catálogo,

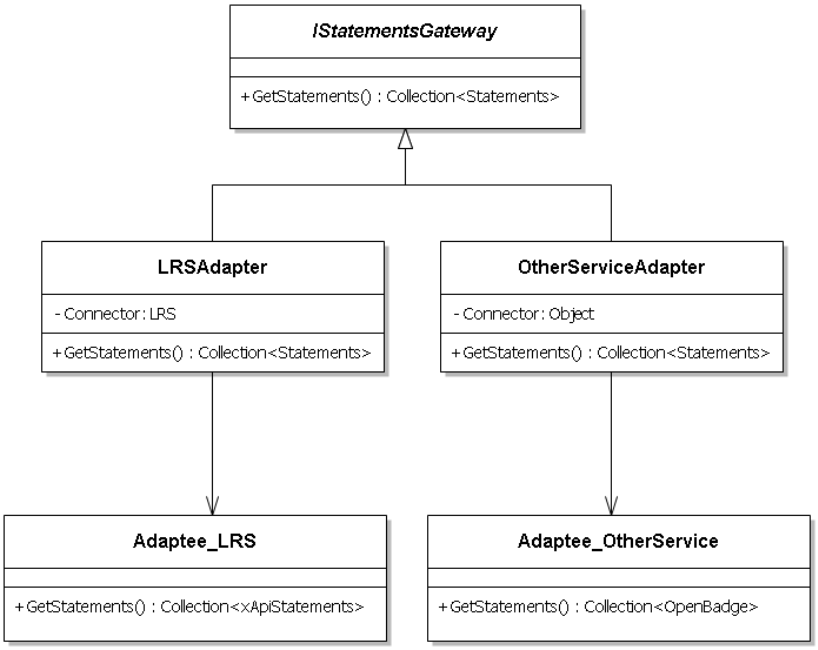

Figura 30 – Diagrama de classes UML de aplicação do padrão Adapter

juntamente com os métodos HTTP associados ao mesmo.

Tabela 9 – Tabela de recursos do serviço de Catálogo e correspondentes métodos

Recurso	**URI**	**Método HTTP**	
Statement	/api/{tenantId}/statements/	GET	getStatements
	/api/{tenantId}/statements/	POST	createStatements
	/api/{tenantId}/statements/findbyactor	GET	getStatementsByActor
	/api/{tenantId}/statements/find	GET	findStatements
	/api/{tenantId}/statements/{statementId}	PUT	putStatements
	/api/{tenantId}/statements/{id}	DELETE	deleteStatement
Favoritos	/api/{tenantid}/favorites/{email}	GET	getfavoritesByEmail
	/api/{ tenantid}/favorites/{email}/{id}	POST	createFavorite
	/api/{tenantid}/favorites/{email}/{id}	DELETE	deleteFavorite
Actividades	/api/{tenantId}/activities/{id}	GET	getActivity

6.5.2 Pathway Service

O *Pathway* Service é um serviço criado para gerir a lógica de negócio associada à construção de percursos de aprendizagem, formal ou informal. Será construída com base na mesma estrutura arquitetural padrão dos serviços *web* como referido na secção 4.2.

Relativamente aos pedidos efetuados, este serviço não será, numa primeira fase, *multi-tenant*, uma vez que esse conceito não existe em grande parte dos serviços externos e será desenhado em particular para o sistema em questão e contexto associado. O objetivo deste serviço é dar suporte à construção de percurso de aprendizagem com base em *Open Badges*.

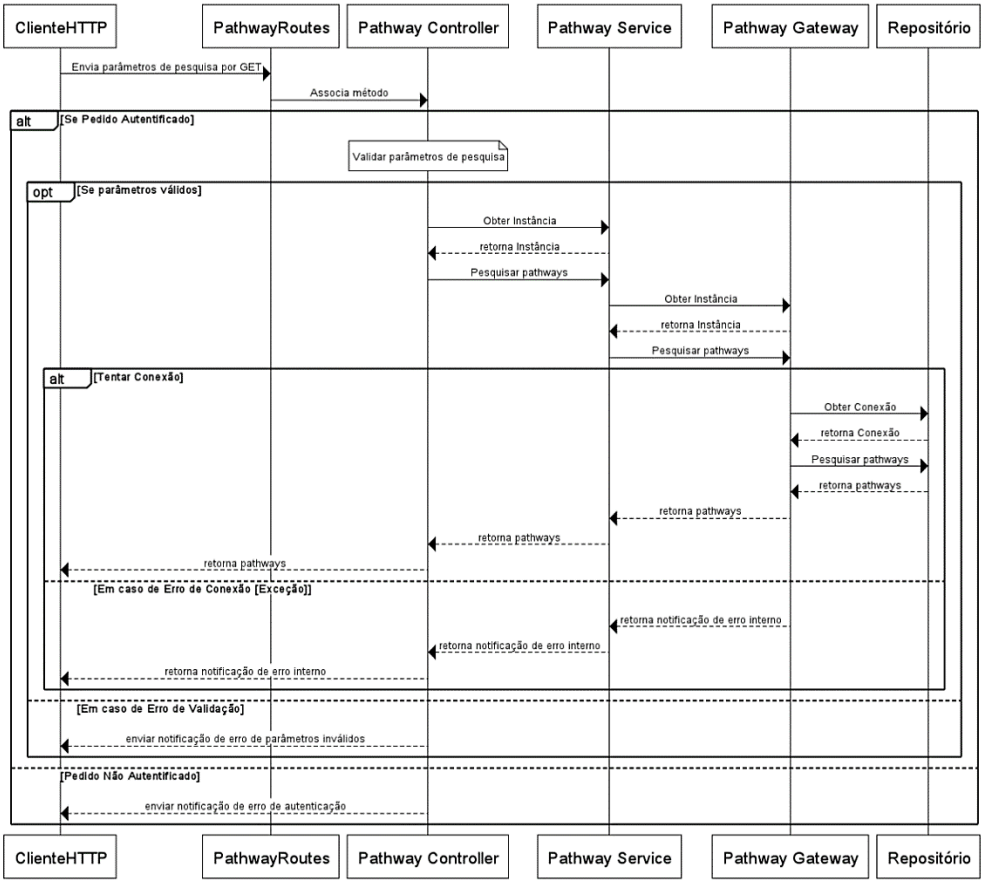

Figura 31 - Diagrama de sequência UML para a pesquisa de *Pathways*

Na Tabela 10 estão representadas as rotas definidas para os pedidos ao serviço de *Pathway*, juntamente com os métodos HTTP associados ao mesmo.

Tabela 10 - Tabela de recursos do serviço *Pathway* e correspondentes métodos

Recurso	**URI**	**Método HTTP**	
Pathways	/api/services/get	GET	getPathway

	/api/services/getall	GET	getPathways
	/api/services/post	POST	createPathway

6.5.3 Recommendation Service

O *Recommendation Service* é um serviço desenhado para sugerir badges para um utilizador registado no sistema. A nível de componentes, realça-se o componente de sugestão de *badges*, para o qual se pretende implementar o sistema de mecanismos de sugestão utilizando um padrão *Strategy*. Desta forma, será possível implementar no futuro diferentes algoritmos de recomendações.

De acordo (Zhang et al. 20116) o interesse pessoal está diretamente relacionado com os *badges* que os utilizadores adquiriram no passado. Ou seja, num cenário em que um determinado utilizador tenha um percurso de aprendizagem centrado nas tecnologias móveis, é possível demonstrar que outras formações ou cursos nessa área possam ser do seu interesse. Assim sendo, é possível avaliar no sistema proposto a utilização do cálculo de interesse pessoal para sugestão de *badges* aos utilizadores. A fórmula a utilizar é ilustrada na Figura 32, em para um dado utilizador u_i, H representa o conjunto de *badges* obtidos pelo utilizador no passado, $s(b_j, b_k)$ representa o grau de similaridade entre um *badge* b_j e b_k e $v^{pi}(u_i, u_k)$ o valor de interesse pessoal do *badge* b_k para o utilizador.

$$v^{pi}(u_i, b_j|\mathcal{H}) = \frac{\sum_{b_k \in \mathcal{H}} s(b_j, b_k) v^{pi}(u_i, b_k)}{|\mathcal{H}|},$$

Figura 32 – Fórmula para cálculo do interesse pessoal do utilizador (Zhang et al. 20116)

Numa segunda abordagem, irá ser usado uma *framework* com um mecanismo automático baseado no *Collaborative Filtering* disponibilizado pela NReco[38], que permite de uma forma simples obter recomendações em tempo real.

[38] https://www.nrecosite.com/recommender_net.aspx

O objetivo destas duas abordagens é obter um primeiro ponto de partida para um mecanismo de sugestões, de forma a disponibilizar ao utilizador um motor de sugestões que vá de encontro às suas necessidades. Na Figura 33 é apresentado o respetivo diagrama de classes de apoio a este mecanismo.

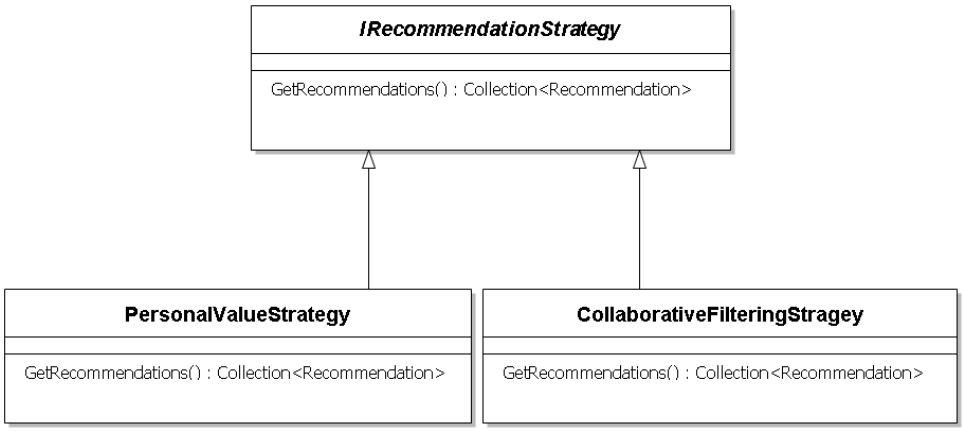

Figura 33 - Diagrama de classes UML representativo do padrão *Strategy* a ser implementado neste serviço

No diagrama da Figura 34 é possível verificar o mecanismo idealizado para este serviço, salientado a existência de um componente de *caching* que foi introduzido para tentar minimizar os tempos de obtenção das recomendações uma vez que os cálculos associados são complexos e necessitam de tempo para serem executados.

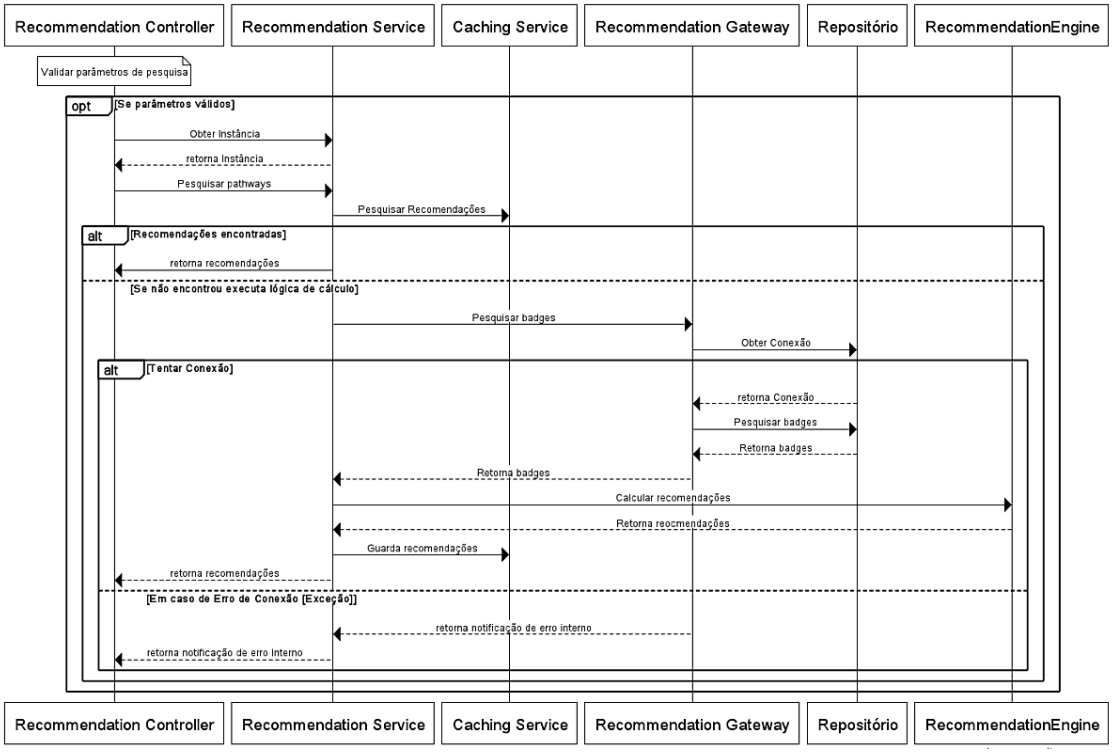

Figura 34 – Diagrama de sequência UML representativo do mecanismo de recomendações

Na Tabela 11 é apresentada a rota definida para os pedido ao serviço de *Recommendation*, juntamente com o método HTTP associado ao mesmo.

Tabela 11 - Tabela de recurso do serviço *Recommendation* e correspondente método

Recurso	**URI**	**Método HTTP**	
Recommendations	/api/recommendations/get	GET	getRecommendations

6.5.4 Catalog Sync Service

O *Catalog Sync Service* é um serviço que foi desenhado para efetuar uma integração com sistemas externos no sentido da partilha de informação, mas acima de tudo propor um possível sistema de integração entre sistemas LRS e não LRS. No caso específico dos O*pen Badges* e da sua descoberta, esta partilha com outros sistemas visou aumentar o catálogo de *badges* disponível. Na Figura 35 estão descritas as diferentes possibilidades para integração com LRS externos. Uma vez que se pretende assegurar a interoperabilidade entre sistemas, a opção mais adequada foi do *man-in-the-middle*, cujo conceito é o de potenciar a partilha de *badges* recorrendo a um componente que atua como intermediário dos pedidos e respostas dos LRS e sistemas externos. Tendo em conta que se pretende permitir uma conexão futura com sistemas

externos não era aconselhável considerar mais nenhuma das opções, tendo-se optado pela ferramenta de partilha centralizada de *badges*.

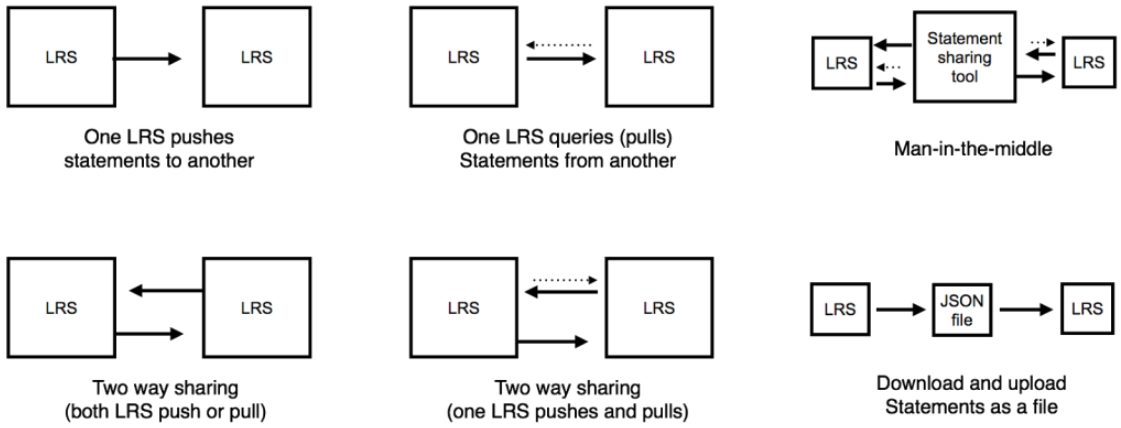

Figura 35 – Estratégias para partilha de *statements* xAPI (fonte: http://tincanapi.com/sharing-statements/)

6.6 Aplicação Web

A aplicação *web* a desenvolver, pretende disponibilizar ao utilizador, através de uma interface gráfica, as funcionalidades descritas na secção 5.2.2. Para além dos requisitos funcionais identificados, terá de respeitar também os requisitos não funcionais descritos na secção 5.2.3.

No que respeita à arquitetura da aplicação, ela seguirá os padrões e estrutura já referidos na secção de Arquitetura, secção 4. Uma vez que os microserviços possuem toda a lógica de negócio, validação e autenticação, bem como repositório internos ou externos de informação, a aplicação *web* será apenas um consumidor desses serviços, não tendo portanto necessidade de ter uma base de dados própria. A aplicação deverá ser executável em qualquer plataforma e dispositivo, com recurso a *web browser* e deve ajustar-se visualmente à área gráfica disponibilizada.

6.6.1 Estrutura e funcionalidade

Para garantir os requisitos identificados, desenvolveu-se uma estrutura elaborada por componentes independentes que suportam os requisitos funcionais e não funcionais como por exemplo, suporte a diferentes linguagens. Tal como acima referido a aplicação *Web* terá toda a sua lógica assente na comunicação efetuada com os serviços da plataforma. Desta forma, terá acesso a todas as funcionalidades e abstrai-se, ao mesmo tempo, da lógica de negócio inerente que estará implementada nos serviços, garantindo a separação de responsabilidades e escalabilidade. Esta aplicação será baseada no padrão de *software Model-View-Controller*

(Modelo – Vista – Controlador) e irá comunicar com a camada aplicacional que encapsula o acesso aos componentes de modelo de negócio, como pode ser verificado pelo diagrama da Figura 36.

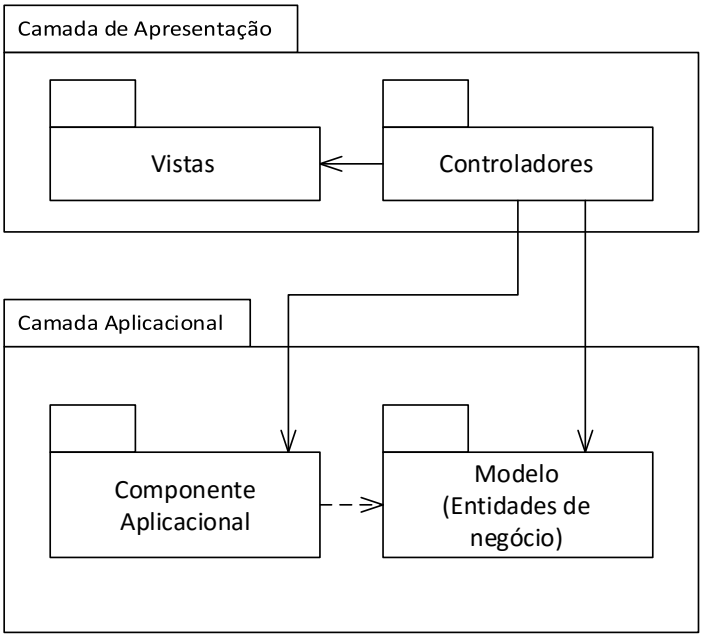

Figura 36 - Diagrama de componentes UML de colaboração entre os controladores e camada aplicacional

Nesta sequência de colaboração surge uma dinâmica particularmente eficiente. Ao Controlador compete receber o pedido do cliente, moldar e redirecionar esse pedido para o Componente Aplicacional, aguardar resposta, e emitir a Vista que irá ser retornada para o utilizador. Esta conceção permite um baixo acoplamento e uma alta coesão entre as suas partes integrantes, uma vez que fomenta a distribuição e atribuição de responsabilidades às entidades especialistas: O controlador, responsável por receber informação do cliente e determinar qual a operação a executar se os dados recebidos forem válidos; o Modelo que representa a estrutura de dados associadas aquela área e por último, a Vista, responsável por representar os dados resultantes numa componente gráfica que será apresentada ao utilizador. O processo desenrola-se de acordo com o ilustrado com o diagrama da Figura 37.

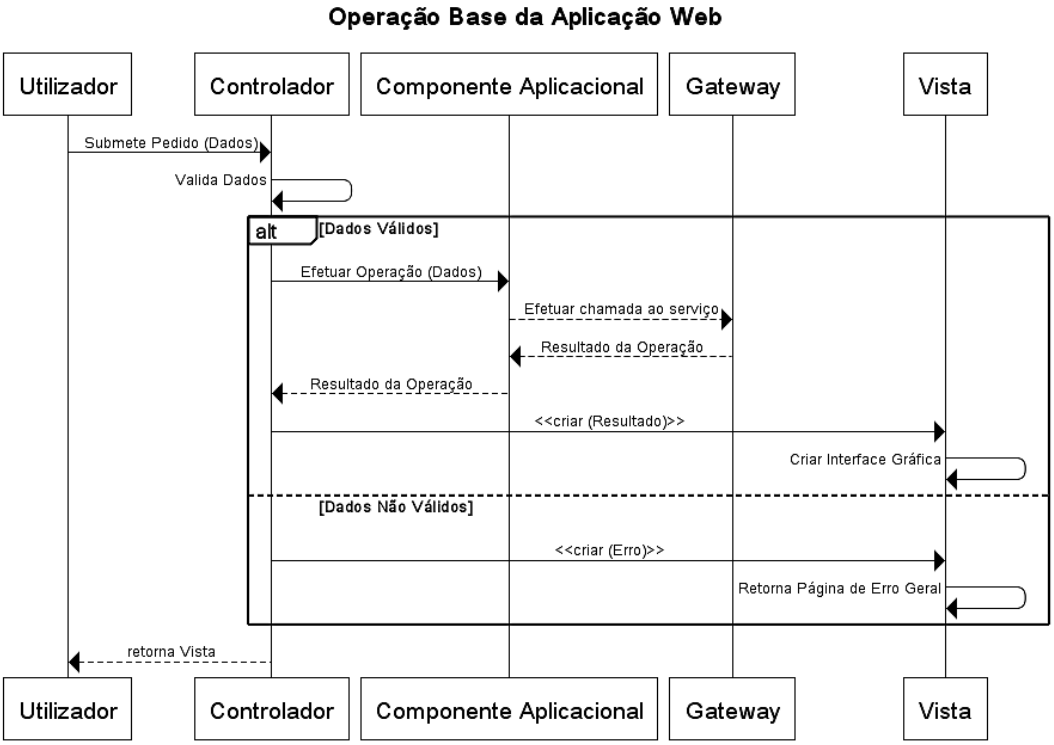

Figura 37 – Diagrama de sequência UML de uma operação base da aplicação *Web*

6.6.2 Controladores e Vistas

Sendo estas estruturas partes essenciais do padrão de *software* em utilização, os controladores e vistas revelam uma importância fundamental para se desenvolverem as funcionalidades previstas. Relacionados intrinsecamente com os casos de uso desta solução, estes componentes assumem uma disposição aproximada à funcionalidade – controlador, em que cada controlador representa uma funcionalidade, e possui um conjunto de vistas associadas para expressão da funcionalidade ou resultado dessa mesma operação. Assim sendo, existirá um controlador para cada conjunto de casos de uso associados ou funcionalidade, como pode ser verificado na Tabela 12.

Tabela 12 – Associação entre controladores, vistas e casos de uso

Controlador	**Vistas**	**Caso de Uso**
BadgesController	Index	UC09 - Pesquisar *badges* UC03 - Marcar *badge* como favorito UC04 - Desmarcar *badge* como favorito
	Personal	UC01 - Ver *badges* pessoais UC08 - Gerir página de portefólio de *badges*
	Detail	UC01 - Ver *badges* pessoais UC02 - Adicionar e remover *tags* dos *badges*
PathwayController	Index	UC05 - Gerir percurso de aprendizagem

	Search	UC06 - Pesquisar percursos de aprendizagem públicos
RecommendationController	Index	UC07 – Ver sugestões de *badges*

Analisando a Tabela 12 é possível concluir que, normalmente, cada controlador está associado a uma entidade do modelo de negócio, dispondo de vistas para gerir essas mesmas entidades associadas.

6.7 Abordagens

Tendo em consideração a tipologia da informação a persistir e a estruturar no projeto, será necessário avaliar as diferentes hipóteses para a pesquisa e estruturação de *badges*.

No caso da pesquisa, optou-se por fazer um estudo comparativo na pesquisa de *badges* a dois tipos de LRS diferentes no sentido de avaliar a que apresenta a melhor flexibilidade na pesquisa. Nesta comparação serão utilizados dois tipos diferentes de LRS, um instalado internamente no sistema e um outro externo, fornecido por uma entidade também externa.

A hipótese levantada com esta experiência estava relacionada com o facto de o LRS interno poder apresentar um comportamento mais flexível e moldável. As variáveis independentes são os dois repositórios e as amostras de dados, enquanto a variável dependente será o número de pesquisas possíveis. O teste estatístico a utilizar nesta experimentação será o *Wilcoxon Signed Rank Test* uma vez que os dados serão emparelhados mas a amostra será aleatória e poderá não seguir uma distribuição normal.

No que se refere ao mecanismo de recomendações de *badges*, entende-se que deverá ser feito um estudo entre dois mecanismos de recomendação, o *Collaborative Filtering* e um proposto para cálculo de interesse pessoal na sugestão de *badges* (Zhang et al. 20116), denominado *Personal Value*. O *Collaborative Filtering* assenta no conceito de relação entre utilizadores e agrupa-os pelos seus interesses. O utilizador pode assim receber recomendações a itens que ainda não avaliou porque foram avaliados positivamente por algum utilizador do grupo (Asanov 2011).

A hipótese que se pretendia testar era de que as recomendações obtidas por *Collaborative Filtering* seriam mais adequadas que o mecanismo de cálculo do interesse pessoal. As variáveis independentes serão os mecanismos a utilizar e a variável dependente a ser avaliada será o número de recomendações relevantes. O teste estatístico a ser utilizado será o *Wilcoxon Signed Rank Test* um vez que os dados a analisar não são paramétricos e o domínio é singular.

7 Implementação

7.1 Tecnologias

Nesta secção são apresentadas as tecnologias mais relevantes para o desenvolvimento do projeto. Serão divididas por tipos e áreas de atuação, bem como enquadradas no âmbito da solução desenvolvida.

7.1.1 Infraestrutura

Relativamente ao sistemas operativos destaca-se a utilização do sistema operativo Windows[39] da versão 10 para sistema de suporte ao desenvolvimento da aplicação e serviços *web*, enquanto que para o *Learning Record Store*, irá ser usado o sistema Operativo Ubuntu[40] de versão 14.04.2.

O IIS[41] é um servidor *web* para o sistema operativo Windows e foi responsável pelo alojamento da aplicação e serviços *web*. Para o *Learning Record Store*, este foi alojado num Servidor Apache no ambiente Ubuntu.

Como ferramenta de desenvolvimento principal, foi utilizado o Visual Studio 2015[42], uma vez que a *stack* de base escolhida para o projeto foi a da Microsoft. Esta, é constituída pelo Windows, IIS, SQL Server e C# ou VB.NET.

[39] https://www.microsoft.com/pt-pt/windows/
[40] https://www.ubuntu.com/
[41] https://www.iis.net/
[42] https://www.visualstudio.com/

7.1.2 .NET

A framework .NET[43] foi escolhida como base de desenvolvimento por apresentar ferramentas de desenvolvimento que permitem uma prototipagem rápida, tais como o *Visual Studio , Nuget* e o *Entity Framework*, apresentam uma excelente performance tanto a nível aplicacional como a nível de serviços, e agora com a chegada do .NET Core, torna-se uma framework que permite desenvolver aplicações compatíveis com vários sistemas operativos, como *Windows*, *Linux* e *Mac*.

7.1.2.1 Entity Framework

Esta *framework* facilita o acesso aos dados e abstrai o programador das regras e validações da base de dados, garantindo a sua integridade e exactidão dos dados. O Entity Framework, como uma ferramenta ORM, responde ao problema de compatibilidade e mapeamento entre os modelos Orientado a Objetos e Entidade-Relacionamento. O primeiro é adotado pelas linguagens de programação e possui uma série de características como a herança, o polimorfismo, entre outras, que os programadores não abdicam facilmente. Por outro lado, o segundo modelo é usado pelos sistemas de persistência de dados (SGBDS) que revela ser mais eficaz na manipulação de grandes volumes de informação.

Para além disso, esta *framework* disponibiliza uma série de recursos que aumentam a produtividade no desenvolvimento de aplicações. A ADO.NET Entity Framework utiliza uma variante da linguagem SQL, denominada Entity SQL, que é responsável por codificar consultas declarativas e actualizações sobre entidades e as suas relações, no nível conceitual. É distinto do SQL na medida em que não tem construções explícitas para junções porque o EntityData Model (EDM) foi desenhado para abstrair o particionamento dos dados nas tabelas. Os arquitetos e os responsáveis pelo desenvolvimento das aplicações têm necessidade de alcançar dois objetivos distintos, por um lado a modelação de entidades, relacionamentos e problemas do modelo de negócio e por outro o trabalhar com sistemas de armazenamento de dados para persistência e consulta de dados. O problema reside no facto de esses dados poderem estar distribuídos em múltiplos sistemas de armazenamento, cada um com as suas características próprias e protocolos específicos. Mesmo que se use um único sistema de armazenamento, ainda existe a necessidade de equilibrar os requisitos de armazenamento com os requisitos de codificação.

Nesse sentido, o *Entity Framework*[44] permite trabalhar com os dados na forma de propriedades e objetos de domínio, sem ter que relacioná-los com as tabelas da base de dados. Tal é possível devido ao elevado nível de abstração que a *framework* oferece.

[43] https://www.microsoft.com/net/intro
[44] http://msdn.microsoft.com/en-us/library/bb399567.aspx

Contudo, apesar das vantagens de utilização desta *framework* é necessário ter em conta as suas desvantagens. Estudos realizados[45] permitiram concluir que o aumento de *overhead* é uma realidade que não pode ser ignorada. No entanto, em soluções de menor dimensão as vantagens superam largamente as desvantagens. Para além das vantagens apresentadas, um dos motivos que levaram à escolha desta tecnologia está relacionada com um dos objetivos do projeto: o desenvolvimento de uma solução de alguma complexidade num reduzido espaço de tempo.

Na Figura 38, é apresentada a arquitetura do *Entity Framework* para acesso a dados. Uma análise cuidada permite verificar que a camada composta pelo *EntityClient Data Provider* é o elemento central da arquitetura, sendo responsável pela gestão de conexões, tradução de consultas a entidades para consultas específicas à base de dados e retorno de *data readers* que serão posteriormente usados para mapear dados de entidades para objetos.

Neste projeto irá recorrer-se a esta *framework* para a aplicação *web* e para o microserviços *BadgeCatalogService* e Autenticação.

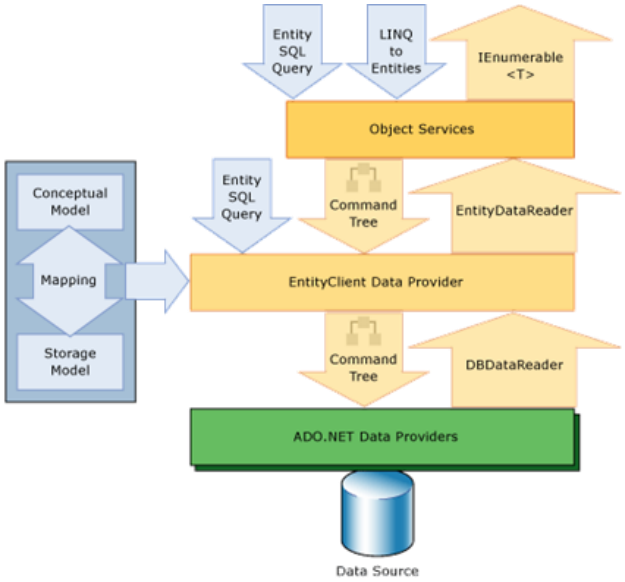

Figura 38 - Arquitetura do *Entity Framework* para acesso a dados

7.1.2.2 ASP.NET MVC 5

O padrão de arquitetura *Model-View-Controller* (MVC) separa a aplicação em três grandes componentes: o modelo, a vista e o controlador (Figura 39).

[45] http://blogs.msdn.com/b/adonet/archive/2008/03/27/ado-net-entity-framework-performance-comparison.aspx

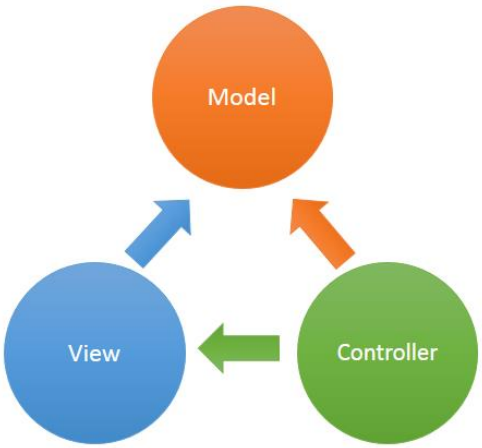

Figura 39 - Esquema do padrão MVC

O modelo (*Model*) é o componente da aplicação que implementa a lógica de domínio, as regras de negócio e o modelo de domínio. Regra geral, os objetos do domínio persistem numa base de dados.

As vistas (*View*) são os componentes que apresentam a interface da aplicação ao utilizador para interação. Tipicamente, a interface de utilizador (UI) é criada tendo por base a informação presente no modelo. Apresenta somente informação.

Os controladores (*Controllers*) são responsáveis por responder às ações do utilizador, trabalham com o modelo e selecionam uma vista para a apresentação ao utilizador. Numa aplicação *MVC*, o controlador processa e responde a pedidos e interações do utilizador.

O *ASP.NET MVC* permite criar aplicações *Web* baseadas no padrão *MVC* que possui vantagens como as de separar as regras e lógicas de negócio da camada de apresentação, permitir um maior controlo da aplicação e permitir a integração com bibliotecas *Javascript* como o *Jquery e Bootstrap*[46]. Apesar das vantagens apresenta também algumas desvantagens tais como a aprendizagem mais lenta e o requerer um conhecimento elevado de HTML e *JavaScript*.

Como *framework*, esta revela ser leve e com uma estrutura de apresentação altamente testável, totalmente integrado com os recursos existentes do *ASP.NET*, como páginas mestre e associações baseadas em autenticação. Relativamente à versão escolhida, optou-se pelo *ASP.NET MVC 5* por ser a última versão completa da framework *.NET*.

7.1.2.3 Json.NET

O JSON.NET é uma biblioteca para .NET que responde aos problemas de serialização de JSON quando este se consegue mapear facilmente para uma classe .NET. Apresenta várias funcionalidades úteis na manipulação de informação em formato JSON como por exemplo, a

validação de JSON através de schemas, serialização e deserialização, querying ao JSON sem recorrer a classes e através de a uma API semelhante a LINQ para permitir estas operações.

7.1.2.4 ASP.NET WebApi

A ASP.NET Web API[47] é uma *framework* que agiliza o processo de desenvolvimento de serviços HTTP para dispositivos móveis e fixos. É a plataforma ideal para construir serviços REST no âmbito da *framework .NET*.

O protocolo HTTP não serve apenas para disponibilizar páginas *web*, mas também para construir APIs para disponibilização de serviços e dados. Tem características únicas como a simplicidade, flexibilidade e ubiquidade. Praticamente todas as plataformas suportam http, por isso serviços baseados no http conseguem chegar a uma grande quantidade dispositivos, fixos ou móveis.

7.1.2.5 AutoMapper

O *AutoMapper*[48] é uma biblioteca que tem por objetivo, evitar a codificação excessiva e desnecessária de mapeadores entre objetos equivalentes mas de diferentes camadas, ou serviços. Possui uma API de configuração com estratégias de mapeamento que utiliza um algoritmo para conjugar os valores origem com os de destino. Adequa-se em grande medida a objetos complexos e simples

A sua utilização neste projeto visa, por um lado, reduzir a quantidade de código e a legibilidade do mesmo, bem como acelerar o processo de desenvolvimento recorrendo a mapeadores automáticos entre entidades de camadas arquiteturais adjacentes.

7.1.2.6 Hangfire

O *Hangfire*[49] é uma *framework open-source* utilizada para criar, processar e gerir tarefas em *background* em aplicações .NET. Tem a grande vantagem de não necessitar de nenhum serviço Windows ou processo paralelo para poder ser executado.

É principalmente aconselhada a sua utilização em serviços de notificações/*newsletters*, *upload* de ficheiros por *batch* (XML, CSV, JSON), execução de *web hooks*, processamento de imagem e vídeo, limpeza de pastas, *reporting* e manutenção e sincronização de bases de dados. A persistência de tarefas entre reinicializações das aplicações é possível recorrendo a um sistema de base de dados, tais como Redis, MongoDB ou mesmo SQL Server.

No âmbito deste projeto, a sua utilização estará relacionada com o serviço *CatalogSyncService* que necessita que a sua lógica de implementação assente em tarefas em *background* e recorrentes. Uma vez que o *Hangfire* se adequa às mais diversas operações, curtas ou de longa

[47] https://www.asp.net/web-api
[48] https://github.com/AutoMapper/AutoMapper/wiki
[49] http://hangfire.io/

duração, intensivas em termos de CPU e I/O ou não, de execução única ou recorrente, torna-se uma escolha óbvia para a sua inclusão neste projecto.

7.1.2.7 WebApiProxy

O WebApiProxy[50] é uma biblioteca que estende o ASP.NET Web API com um *proxy endpoint*, uma espécie de SDK da API, disponibilizando aos clientes da API os métodos e os tipos de objetos usados de uma forma explícita. Assim sendo, é possível conhecer em cada instante os detalhes do serviço de forma automática e identificar as prováveis alterações ao longo do tempo. Por outro lado, torna extremamente simples a implementação das chamadas aos serviços uma vez que oferece uma interface para essa comunicação.

Esta biblioteca vai ser utilizada na comunicação da aplicação *web* com os serviços e mesmo entre serviços, permitindo a uma implementação mais uniforme e estruturada na conexão

7.1.2.8 Swagger

O *Swagger* [51] é um *standard* que, independentemente da linguagem, estabelece uma especificação formal de uma REST API que é interpretável, ao mesmo tempo, por humanos e computadores e permite descobrir os detalhes de um serviço sem ser preciso aceder ao código fonte ou documentação. Apresenta diversas funcionalidades e ferramentas que permitem desde especificar o serviço a disponibilizar detalhes do mesmo numa interface gráfica. No contexto do trabalho em questão irá usar-se apenas o componente *Swagger UI* que permite explorar a API de um serviço diretamente sobre o *endpoint* do mesmo, quer para conhecer os detalhes do serviço como parâmetros de entrada, tipologia e assinatura dos métodos, quer para experimentar as chamadas ao serviço.

7.1.2.9 Moq

A biblioteca *Moq*[52] é dedicada ao ambiente *.NET* e permite, através da sua API, fazer *mocking* de Métodos e objectos de uma forma simples e intuitiva, sendo por este motivo utilizada no desenvolvimento de testes unitários. O *mocking* não é mais do que simular respostas de métodos ou comportamentos no caso dos objectos. A utilização desta tecnlogia permite desenvolver testes unitários ao código de uma forma mais uniforme e estruturada, isolando comportamentos e testando apenas pedaços de código independentemente do contexto onde se inserem.

[50] http://reynders.co/introducing-webapiproxy-providing
[51] http://swagger.io
[52] http://www.agile-code.com/blog/mocking-with-moq/

7.1.2.10 Specflow

O *SpecFlow*[53] é um projeto de código aberto cujo objetivo é criar uma associação entre regras de negócio e o código. O mesmo descreve critérios de aceitação para, nomeadamente, casos de uso, através da sintaxe Gherkin[54]. É utilizado através da descrição de cenários de teste em ficheiros denominados *feature*, como é possível verificar na Figura 41.

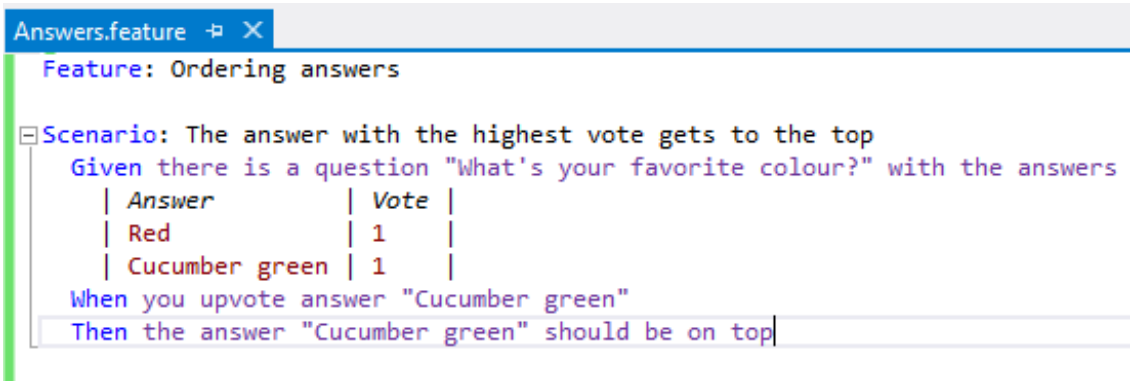

Figura 41 – Exemplo de ficheiro *feature* com a descrição do cenário de teste (fonte: http://www.specflow.org/)

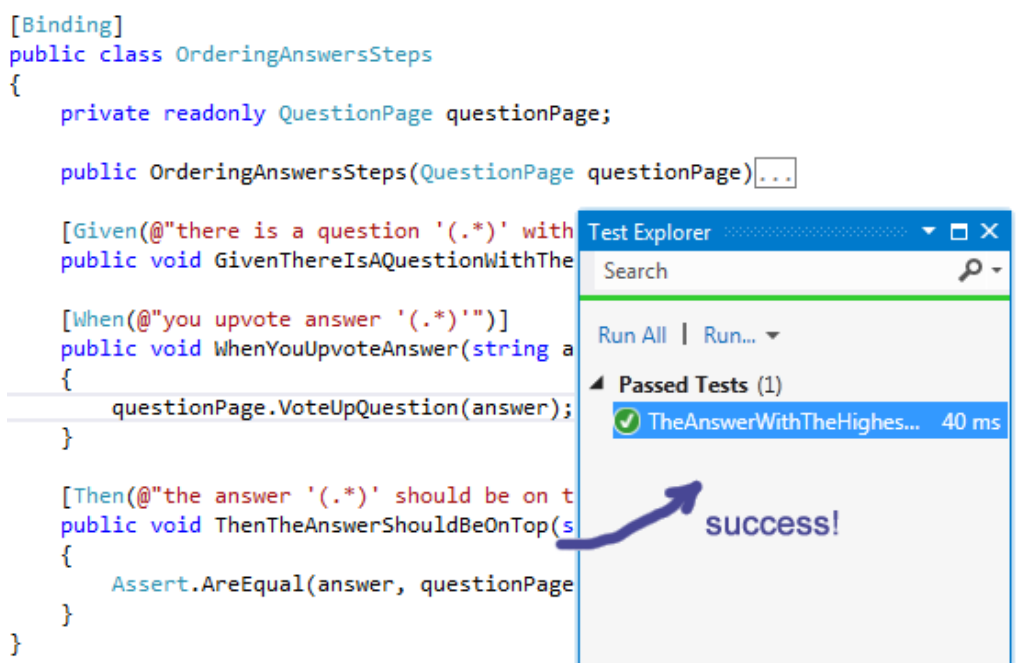

Figura 40 – Exemplo de uma classe de teste do cenário definido na correspondente *feature* (fonte: http://www.specflow.org/)

[53] http://www.specflow.org/
[54] https://github.com/cucumber/cucumber/wiki/Gherkin

Por outro lado, terá de existir associação entre esta descrição do cenário de testes e a implementação em código. Para tal efeito, é criada uma classe de código representativa dos passos descritos no cenário, como é possível ver na Figura 40.

Juntamente com o *Specflow* foi utilizado o *NUnit*[55], que é uma *framework* de testes unitários cujo objetivo foi a validação do resultado dos cenários de teste.

7.1.2.11 TinCan.NET

A biblioteca *TinCan.NET*[56] visa auxiliar na comunicação com *Learning Record Stores* (LRS) uma vez que estes sistemas apresentam um padrão comum na especificação dos métodos das API's, logo não é necessário duplicar código para comunicar com os LRS; basta usar as bibliotecas que foram criadas para este fim. Salienta-se que foi desenvolvido pela empresa *Rustici Software*[57] , empresa encarregue pela primeira versão da xAPI.

7.1.3 Bases de Dados

7.1.3.1 SQL Server e SQL Server Management Studio

O SQL Server[58] é uma plataforma de base de dados altamente escalável e será utilizada para bases de dados relacionais de apoio aos serviços *web*, nomeadamente ao *AuthenticationService*, *BadgeCatalogService* e *PathwayService*.

Para gestão das bases de dados relacionais será utilizado o *SQL Server Management Studio* 2014[59], que permite fazer gestão de tabelas bem como de permissões e questões de segurança da base de dados.

7.1.3.2 MongoDB

O MongoDB (MongoDB n.d.) enquadra-se no contexto das bases de dados *NoSQL* e tem-se tornado muito popular pela facilidade de utilização oferecida pelo sistema MongoDB. Não é só extremamente bem documentado, como ainda é suportado por uma grande comunidade. Ao mesmo tempo revela-se bastante amigável para programadores que possuam experiência com sistemas relacionais e linguagem SQL, tornando-o num sistema especialmente indicado para utilizadores inexperientes no mundo NoSQL (Wilson 2013). MongoDB é um sistema NoSQL, orientado para o armazenamento sob a forma de documentos e desenvolvido em C++. Os documentos neste sistema são serializados no formato JSON e armazenados fisicamente recorrendo à codificação binária do formato JSON, denominada de BSON. O sistema MongoBD utiliza um sistema de indexação parecido com aquela a que recorrem as bases de dados relacionais. Cada documento é identificado por um campo "_id" e nesse campo é criado um índice único. A indexação é um processo importante para a eficiência das operações de leitura,

[55] http://www.nunit.org/
[56] http://rusticisoftware.github.io/TinCan.NET/
[57] https://github.com/RusticiSoftware
[58] https://www.microsoft.com/pt-pt/server-cloud/products/sql-server/overview.aspx
[59] https://msdn.microsoft.com/en-us/library/mt238290.aspx

no entanto isso pode revelar-se como um fator que impacta negativamente a performance de operações de inserção de dados. O sistema MongoDB oferece nativamente vários controladores, que permitem que interações com o sistema, sejam realizadas por aplicações desenvolvidas em várias linguagens, tais como C, C++, Java, .NET, JavaScript e PHP. Muitas organizações de grande dimensão a nível mundial, possuem nos seus repositórios de dados, um sistema MongoDB.

7.1.3.3 REDIS

O Redis[60] significa *Remote Dictionary Service* e é uma base de dados NoSQL de armazenamento do tipo chave-valor. Este projeto, lançado em 2009, foi implementado recorrendo à linguagem C e é um projeto de fonte aberta (Cattell 2010). Este sistema trabalha em memória, oferecendo assim uma grande performance. Ao mesmo tempo, fornece uma boa capacidade de replicação e um modelo de dados único para a construção de aplicações orientadas para a resolução de problemas (Carlson, 2013). Este sistema tem a seu favor a facilidade de utilização e um conjunto sofisticado de comandos para interação com o sistema. Para além destes fatores, o sistema Redis é especialmente renomeado pela sua rapidez. De acordos com indicadores obtidos em testes de performance, as leituras neste sistema são bastante rápidas e as escritas ainda mais. O sistema é capaz de executar e gerir mais de cem mil operações por segundo (Redmond & Wilson 2012). Esta velocidade deve-se, em grande parte, ao facto de este sistema manter o conjunto de dados a utilizar em memória. É também capaz de oferecer persistência de dados, porque possui um mecanismo que, de uma forma assíncrona, escreve as alterações realizadas, em disco (Haines 2009). É possível configurar de quanto em quanto tempo, ou depois de quantas transações é que o sistema deve guardar os dados. O ponto fraco do sistema Redis é o tamanho do conjunto de dados a utilizar uma vez que não pode aproximar-se do tamanho máximo da memória RAM disponível no sistema (Haines 2009), estando limitado nesse aspeto em relação a outras soluções *NoSQL*, e mais especificamente, a outros sistemas de armazenamento do tipo par chave-valor.

7.1.4 Repositórios

Nesta secção são apresentados os repositórios alvo de estudo neste projeto. Estes foram escolhidos tendo em consideração duas realidades distintas, a dos *Open Badges* no contexto da especificação xAPI e a realidade dos *Open Badges* como especificação particular e independente da xAPI.

7.1.4.1 Learning Locker

O *Learning Locker*[61] é um LRS *open source* desenvolvido pela empresa HT2[62] e, como já foi referido anteriormente é um tipo de repositório adequado a guardar *statements* de atividades de aprendizagem gerados pela especificação xAPI. Este sistema suporta a versão 1.0 da

[60] http://redis.io/
[61] https://learninglocker.net/
[62] https://www.ht2labs.com/

especificação anterior e tem a limitação de não suportar versões inferiores a esta. Para além de uma API uniforme relativamente aos outros LRS no mercado, possui também autenticação por HTTP Basic ou mesmo OAuth. Este sistema permite também criar várias instâncias de LRS para servir as mais diversas necessidades, tanto de separação de dados entre aplicações ou grupos de utilizadores ou mesmo por questões de segurança. Para além da API possui um *backoffice* que permite gerir contas de clientes para comunicação autorizada com a API e respetivo LRS, ferramentas de *reporting* e de exportação de dados. Um dos pontos fortes desta ferramenta é a integração nativa da API de agregação de MongoDB, o que permite, em termos de leitura de dados, combinações complexas de pesquisa de uma forma semelhante ao que é feito em bases de dados relacionais com o SQL.

Este LRS será o escolhido como LRS interno da plataforma a desenvolver por apresentar um suporte completo à especificação xAPI, apresentar funcionalidades importantes como disponibilizar uma API muito completa para inserção e pesquisa de statements, e por se tratar de uma ferramenta completamente *open source*, e por isso extremamente flexível e sem custos associados.

7.1.4.2 WaxLRS

O produto WaxLRS[63] é um LRS pago com possibilidade de registo de conta com limitações na quantidade de *statements* inseridos por mês. Este LRS possui funcionalidades de *reporting, querying analytics* em *Big Data,* suporte a múltiplas contas de utilizadores, múltiplos plugins para conexão (ex: Moodle, Wordpress, Adobe,…), uma API dedicada e bem documentada para integração com sistemas externos.

7.1.4.3 Watershed LRS

O Watershed LRS[64] é um LRS pago que possui funcionalidades de agregação, em tempo real, de dados no formato xAPI através de uma API dedicada, exportação de dados em CSV e JSON e ainda integração com fontes de dados com especificação diferente da xAPI. Possui também funcionalidades de análise de dados e *reporting* avançadas.

7.1.4.4 Open Badges Directory

O projeto *Open Badges Directory*[65] é um protótipo de repositório para guardar e disponibilizar *Open Badges*. Tem funcionalidades de pesquisa avançadas, por *issuer*, *tags*, texto livre, nome do *badge*. Para além das funcionalidades de pesquisa permite adicionar *badges* ao repositório. Ao contrário dos LRS anteriores, esta solução não apresenta os dados num formato padrão como o xAPI, mas apenas no formato *Open Badge*.

[63] http://www.saltbox.com/wax-learning-record-store.html
[64] https://www.watershedlrs.com/
[65] https://badgealliance.github.io/openbadges-directory/

7.1.5 UI

7.1.5.1 Bootstrap

O *Bootstrap*[66] é uma *framework* UI para desenvolvimento de interfaces gráficas de aplicações *web* de uma forma padronizada e recorrendo a boas práticas de design e *scalling*. Com esta tecnologia é possível desenvolver interfaces gráficas que se adaptam aos mais diversos dispositivos e resoluções de ecrã (*responsive*).

No contexto deste projeto, será utilizada no desenvolvimento da aplicação *web*, permitindo desenvolver os diferentes componentes de uma forma padronizada e com um *aspeto gráfico* semelhante assim como permite que se partilhe a mesma estrutura base ao longo dos mesmos.

7.1.5.2 JQuery

O *JQuery* é uma biblioteca *Javascript* totalmente livre e de código aberto cuja principal missão é tornar o desenvolvimento web mais rápido foi, desde o primeiro momento, uma das escolhas tecnológicas mais relevantes. Relativamente aos seus pontos fortes destacam-se a capacidade de manipulação gráfica, como animações e efeitos, a gestão de eventos, o suporte a comunicações *Ajax* e acima de tudo o facto de apresentar uma arquitetura baseada em *plugins* que lhe conferem uma grande extensibilidade. Esta linguagem foi utilizada no desenvolvimento da plataforma web na camada de apresentação.

7.1.5.3 MvcGrid.NET

A biblioteca *MvcGrid.NET*[67] tem por objetivo fornecer as funcionalidades básicas para listar coleções de objetos de uma forma que se possa reutilizar o mesmo comportamento em toda a aplicação. Possui funcionalidades paginação, ordenação, aplicação de filtros, exportação para csv e ainda permite estender as já existentes. A nível gráfico, baseia-se no *Bootstrap* o que se revela mais uma vantagem na sua adoção para este projeto.

7.1.6 Dados de teste

No desenvolvimento de qualquer aplicação, é necessário testá-la em condições semelhantes ao de um ambiente de produção e com vários utilizadores e possivelmente elevado número de processos de escrita e leitura de dados. Neste contexto surgem os dados de teste que devem ser criados no sentido de criar um ambiente o mais completo e realista possível.

A plataforma *Mockaroo*[68] é um serviço que fornece gratuitamente dados de teste para aplicações para os mais diversos propósitos. Trata-se de uma ferramenta muito útil para criar uma grande quantidade de elementos de teste, de forma a aproximar os ambientes de desenvolvimento e teste muito semelhantes a um eventual ambiente de produção. Desta forma, foi utilizada no âmbito deste projeto para criar uma grande quantidade de dados de teste para

[66] http://getbootstrap.com/
[67] http://mvcgrid.net/
[68] https://www.mockaroo.com/

a alimentar o repositório de *badges*. Em primeiro lugar foram utilizados cursos de formação retirados das APIs públicas do eDX[69] e do PluralSight[70], no sentido de dotar a base de dados de teste com cursos reais e atualizados, para uma simulação mais próxima da realidade. O *Mockaroo* permitiu criar, aleatoriamente, nomes e emails para utilizadores da plataforma. Através de uma distribuição normal foi possível associar para esses cursos os utilizadores gerados e construir dados fictícios para *Open Badges*. Foram criados assim 979 *badges* que foram submetidos diretamente no serviço de catálogo para persistir no catálogo interno.

7.2 Metodologia

Relativamente à metodologia utilizada para desenvolver a aplicação e serviços *web*, utilizou-se a abordagem TDD, *Test Driven Development*, para codificação de todos os componentes. Esta metodologia tem por base o escrever os testes unitários antes de se implementar os métodos. Revela-se uma técnica bastante eficaz no sentido em que ajuda a que se teste melhor a aplicação e ao mesmo tempo se tenha uma análise crítica ao modelo de classes implementado. Como é óbvio, o TDD não justifica que seja utilizado sempre, uma vez que existem alguns casos em que tal não é tão vantajoso, como por exemplo quando se implementa uma classe de acesso à base de dados. Na Figura 42 está representado o fluxograma idealizado para este desenvolvimento.

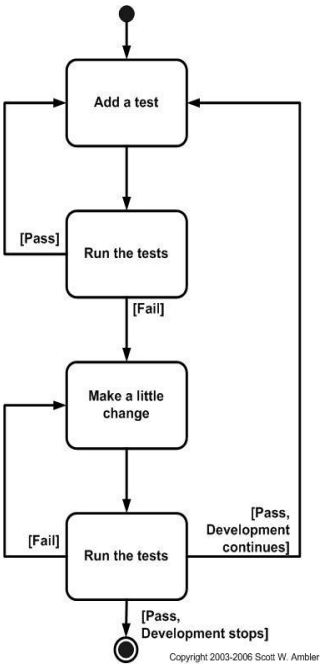

Figura 42 – Fluxograma representativo de uma metodologia TDD (fonte: http://agiledata.org/essays/tdd.html)

[69] https://www.edx.org/
[70] https://www.pluralsight.com/

7.3 Prova de conceito

7.3.1 Instalação LRS

De forma a ser possível desenvolver e implementar o sistema proposto, o primeiro passo seria configurar o servidor LRS para o projeto. O LRS escolhido foi o *Learning Locker* pelos motivos descritos na secção 7.1.4.1.

Tabela 13 – *Setup* de instalação do *Learning Locker*

Ferramentas	**Versão**
Ubuntu	14.04.2
MongoDB	2.6.6
Apache	2
PHP	5
Learning Locker	1.13.3

Na Figura 43 está representado o ecrã de administração das instâncias LRS do *Learning Locker*, no qual é possível criar novas instâncias de LRS, Figura 45, para diferentes ambientes de teste por exemplo. No ecrã de gestão de clientes, Figura 46, é possível criar clientes para os serviços usarem nos pedidos à API. Neste caso, pedidos efetuados pelo *BadgesCatalogService* uma vez que o LRS fica encapsulado por esse serviço.

Na Figura 44 está representado ecrã do *Learning Locker* que permite visualizar os registos de *statements* guardados no LRS.

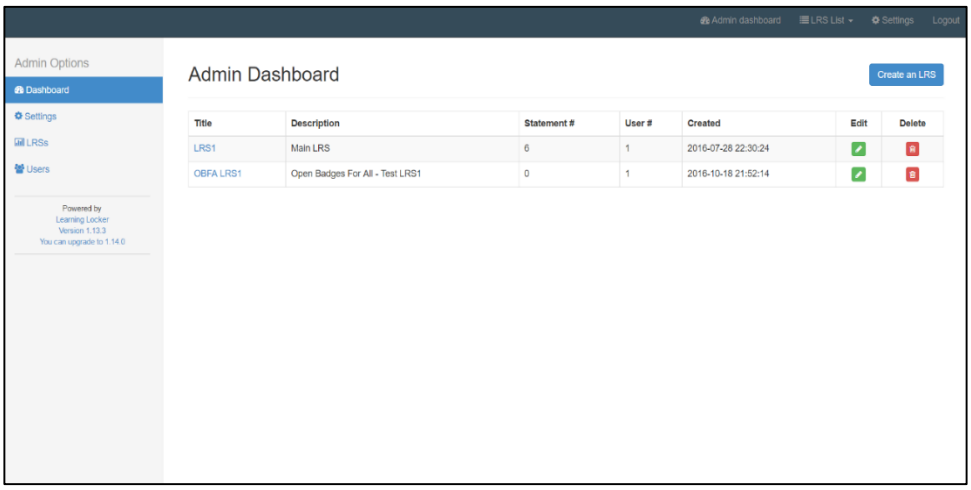

Figura 43 – Painel de Administração do *Learning Locker*

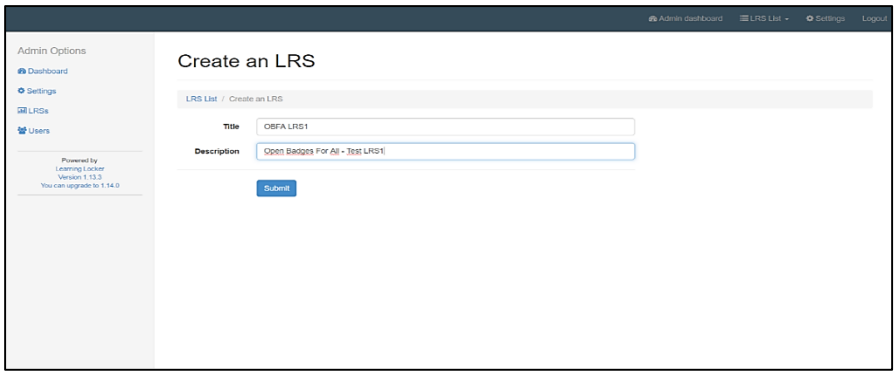

Figura 45 – Ecrã de criação de uma instância LRS no *Learning Locker*

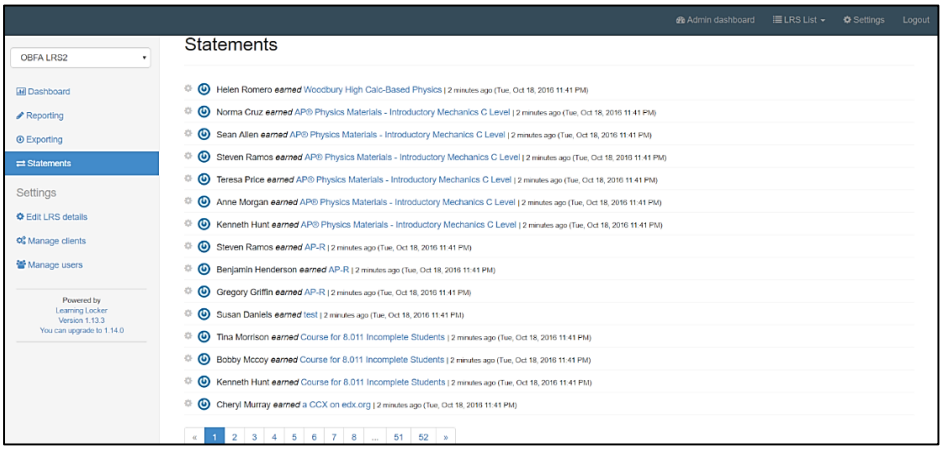

Figura 44 – Ecrã de visualização de *statements* para determinada instância

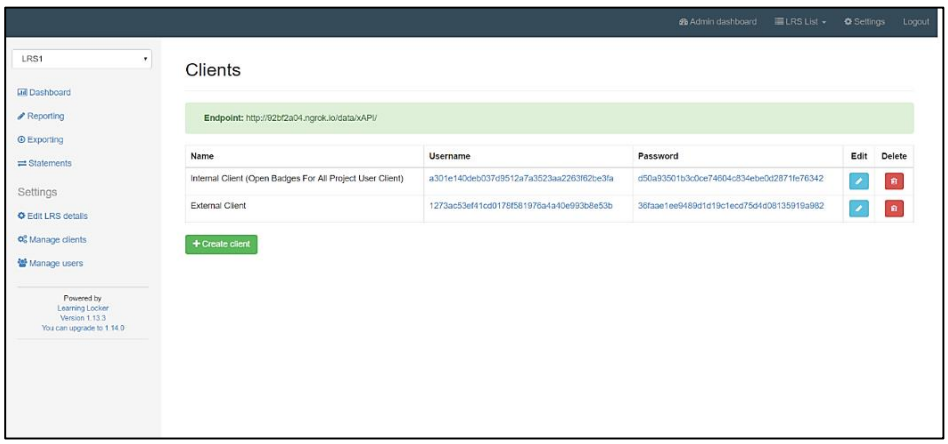

Figura 46 – Ecrã de visualização e gestão de clientes

7.3.2 Microserviços

Os microserviços *BadgesCatalogService*, *PathwayService*, *RecommendationsService* e *CatalogSyncService* foram implementados seguindo as mesmas orientações arquiteturais e o documento de *design* proposto.

7.3.2.1 Badges Catalog Service

Para este serviço, é possível visualizar na Figura 47 as rotas implementadas, tendo sido disponibilizado também um cliente através do *WebAPIProxy* para comunicação com este serviço.

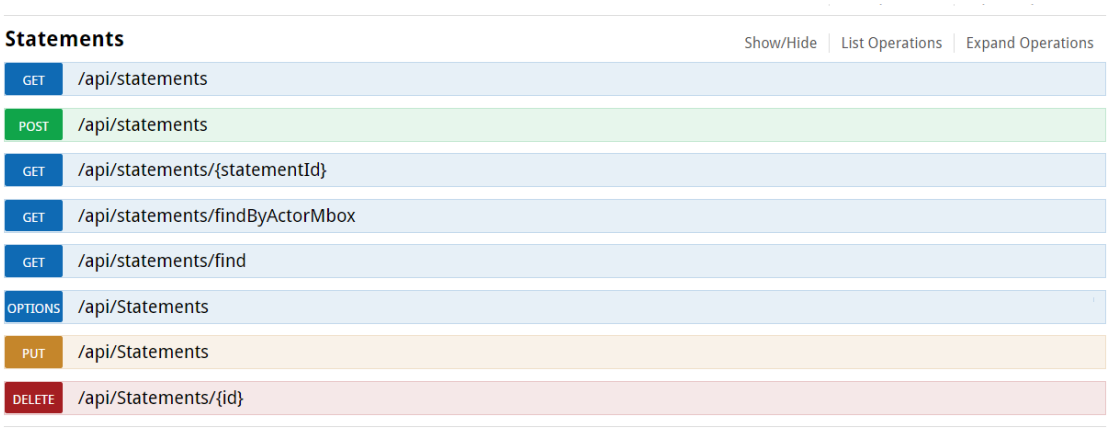

Figura 47 – Rotas disponibilizadas pelo serviço para *Statements* (Swagger UI)

Salienta-se ainda para este componente, o mecanismo de comunicação com os repositórios, diferenciados pelo *Tenant Id* e que permite a pesquisa de *statements (badges)*. Este método suporta filtros e paginação, para otimizar as chamadas aos repositórios. No Código 1 e Código 2 estão representados os métodos principais para pesquisa de *statements*.

```
public StatementsQueryResult GetStatements(int tenantId, StatementQuery
query, Paging paging)
        {
            var lrsQuery = BuildLRSQuery(query, paging);
                return new StatementsQueryResult
                {
                    Statements = this.GetStatements(tenantId, lrsQuery),
                    Total = CountStatements(tenantId)
                };
    }
```

Código 1 – Método responsável pela pesquisa de *Statements*

```
private IEnumerable<XAPI.Domain.Statement> GetStatements(int tenantId,
StatementsQuery query)
        {
            var statements = this.GetLRSStatements(tenantId, query);

            if (statements != null && statements.Any())
            {
                var statementsList = new List<XAPI.Domain.Statement>();

                foreach (var statement in statements)
                {

statementsList.Add(Mapper.Map<XAPI.Domain.Statement>(statement));
                }

                return statementsList;
            }

            return null;
        }

private IEnumerable<TinCan.Statement> GetLRSStatements(int tenantId,
StatementsQuery query)
        {
            var target =
StatementsTargetFactory.GetStatementsTarget(tenantId);
            var response = target.QueryStatements(query);

            if (response.success)
            {
                return response.content.statements;
            }

            return null;
        }
```

Código 2 – Métodos privados de conexão com os LRS

7.3.2.2 Pathway Service

Relativamente ao *Pathway Service*, este possui também um estrutura arquitetural tradicional, *N-Layer*, que permite criar percursos de aprendizagem bem como fazer pesquisas sobre o repositório. No Código 3 é possível visualizar parte da classe de serviço na qual se insere a lógica de negócio da criação e pesquisa de *pathways*.

```
    public class PathwayService :
PathwayServiceAppServiceBase, IPathwayService
    {
        private readonly IPathwayManager pathwayManager;
        private readonly IRepository<Pathway, Guid>
pathwayRepository;

        public PathwayService(
            IPathwayManager pathwayManager,
            IRepository<Pathway, Guid> pathwayRepository)
        {
            this.pathwayManager = pathwayManager;
            this.pathwayRepository = pathwayRepository;
        }

        public async Task CreateOrUpdate(PathwayDTO pathway)
        {
            Pathway newPathway;
            if (pathway.Id == Guid.Empty)
            {
                newPathway = Pathway.Create(pathway.Name,
pathway.Description, pathway.OwnerMBox);
                await pathwayManager.CreateAsync(newPathway);
                await CurrentUnitOfWork.SaveChangesAsync();
            }
            else
            {
                newPathway = await
pathwayManager.CleanPathway(pathway.OwnerMBox);
            }

            if (newPathway != null && pathway.Badges != null
&& pathway.Badges.Any())
            {
                var createdPathway = await
pathwayManager.GetAsync(pathway.OwnerMBox);

                foreach(var badge in pathway.Badges)
                {
                    var badgePathway = await
pathwayManager.CreatePathwayBadgeAsync(createdPathway,
badge.x, badge.y, badge.ReferenceBadge);
                }
            }

            await CurrentUnitOfWork.SaveChangesAsync();
        }

    }
```

Código 3 – Excerto da classe de serviço responsável pela lógica de negócio da criação e pesquisa de *pathways*

7.3.3 Aplicação *Web*

A aplicação *Web* é a interface primária de toda a plataforma. Tem como objetivo disponibilizar uma interface simples e eficaz ao utilizador, assim como oferecer potencialidade em termos de desenvolvimento e adição de novas opções e funcionalidades. Em termos de arquitetura, como referido no capítulo de análise deste relatório, esta interface é baseada na utilização do padrão Modelo – Vista – Controlador. Para garantir conformidade e organização, recorre-se a um modelo de visualização que baseado numa entidade de modelo, revela ou oculta caraterísticas deste trazendo vantagens específicas em cada caso de utilização.

7.3.3.1 Interface Gráfica

A interface gráfica foi desenvolvida com recurso às tecnologias Razor do ASP.NET, HTML, CSS 3 e *Javascript*, com auxílio dos *plugins JQuery*, *Bootstrap* e *MVCGrid*. Recorrendo à tecnologia AJAX a interface está desenhada segundo um conceito em que apenas parte da página é atualizada, simulando o conceito de *single page application*. A esta última acresce ainda o facto de diminuir a quantidade de dados que tem de ser transferida entre Servidor e Cliente, e consequentemente aumentar a velocidade das operações.

Numa altura em que dispositivos móveis e computadores tradicionais operam praticamente da mesma forma e apresentam as mesmas funcionalidades, a interface gráfica da aplicação teve de ser uniformizada de modo a que fosse utilizável independentemente do tamanho do ecrã.

Neste seguimento surgiu então uma camada de interface para com o utilizador que pode ser usada em qualquer dispositivo e/ou em qualquer sistema operacional. Um exemplo deste casos são as Figura 48 e Figura 49, que demonstra, respetivamente, o ecrã de login para a versão computador pessoal (*desktop)* e *mobile*.

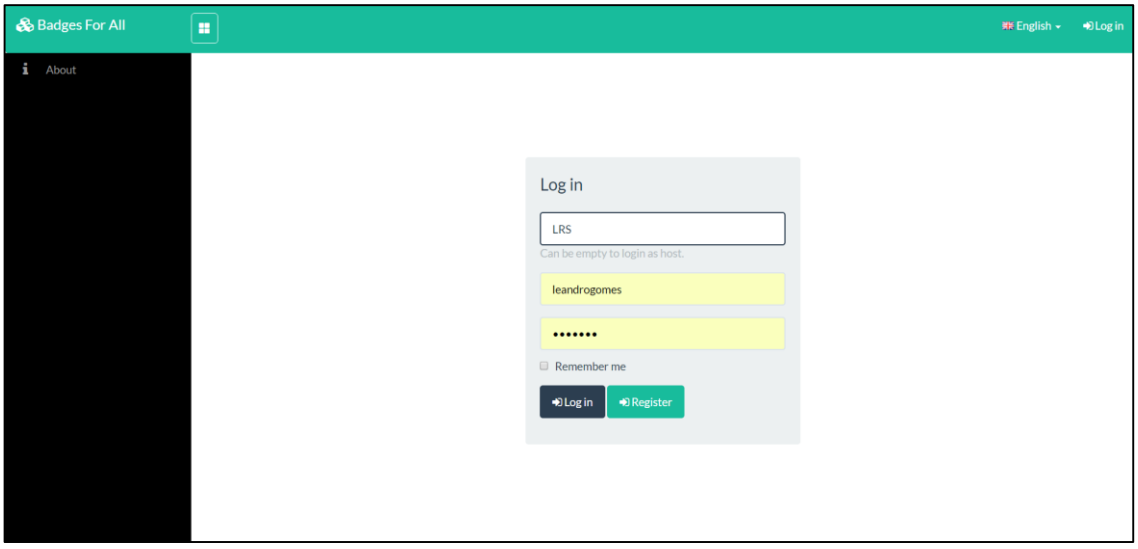

Figura 48 – Página de *login* de utilizador numa resolução *desktop*

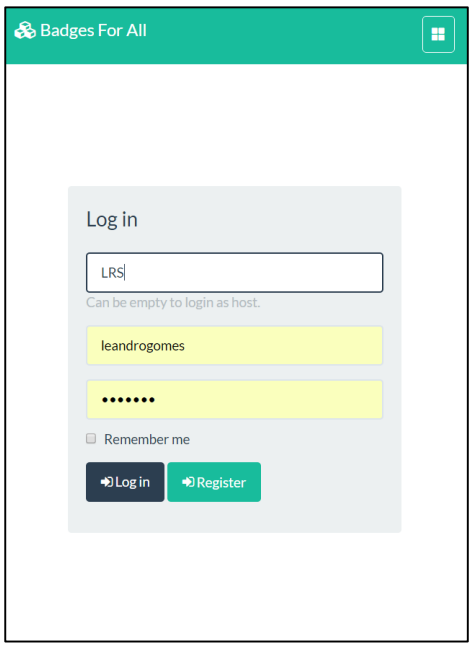

Figura 49 – Ecrã de login de utilizador para ecrã com dimensões *mobile*

7.3.3.2 Página pessoal de *Badges*

Na página pessoal do utilizador, é possível visualizar os *badges* que lhe pertencem.

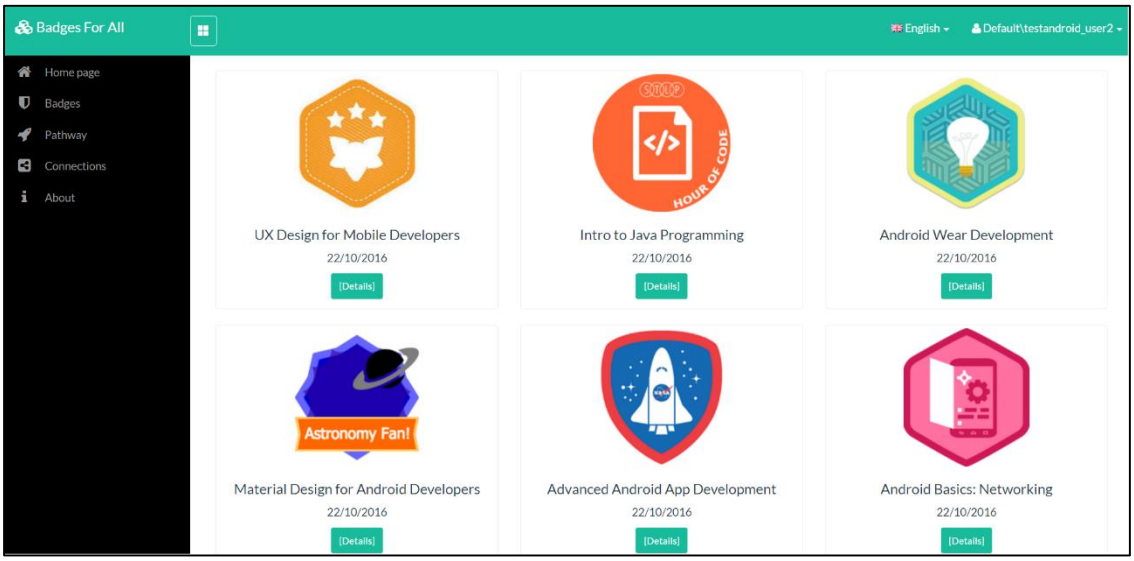

Figura 50 – Exemplo de ecrã de portefólio pessoal de *Open Badges*

Nesta mesma página é possível visualizar, numa *modal box*, o detalhe de cada um dos *badges*, como é possível ilustrar na Figura 51 – *Modal box* de detalhe de um *badge*

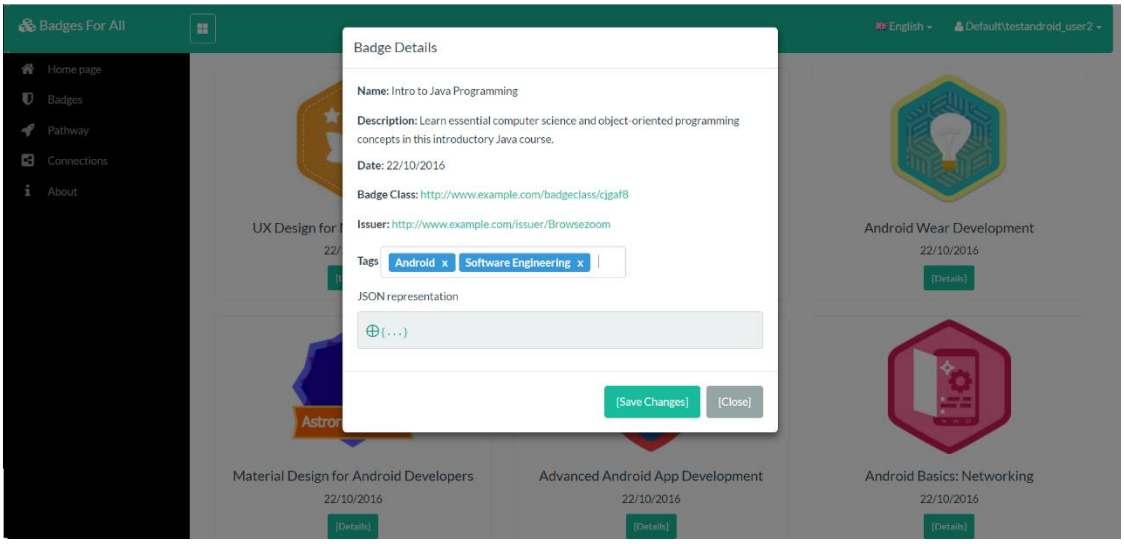

Figura 51 – *Modal box* de detalhe de um *badge*

7.3.3.3 Página de detalhe de *Badge*

Relativamente aos requisitos funcionais, é possível verificar na Figura 53 e na Figura 52 a visualização de detalhe de um *badge*, com foco para o motor de renderização JSON na própria página de detalhe. É neste ecrã que no caso de se tratar de um utilizador a visualizar os seus *badges*, este pode acrescentar *tags* ao mesmo. Esta funcionalidade de edição de *tags* é particularmente relevante para os filtros de pesquisa a aplicar na procura de *badges*.

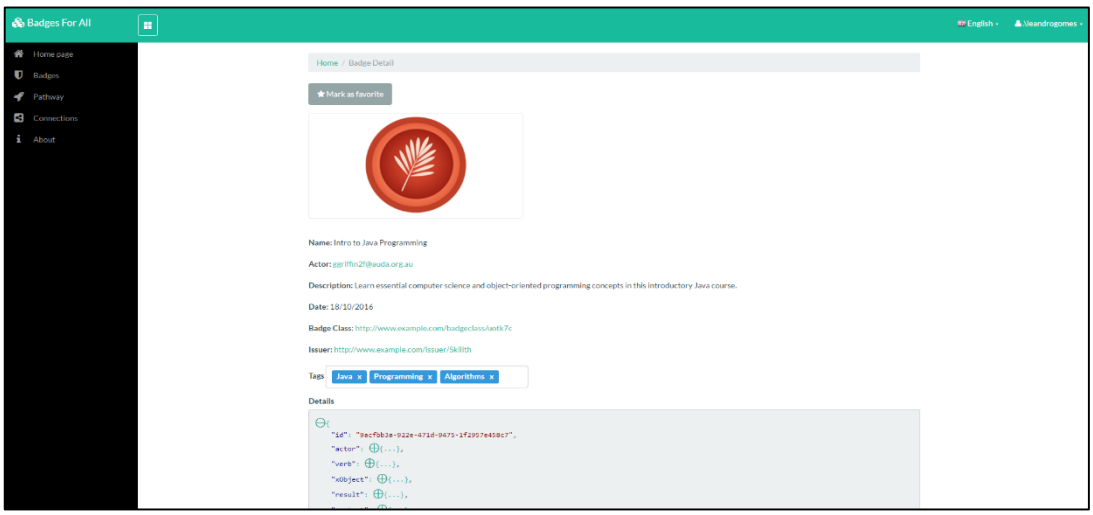

Figura 52 – Página de detalhe de um *badge*

Figura 53 – Página de detalhe de um *badge* com foco no visualizador de documento JSON no formato xAPI

7.3.3.4 Página de listagem de *Badges*

Uma das particularidades de qualquer sistema atualmente, reside no facto de estes possuírem grandes quantidades de informação. É requisito funcional a existência de uma forma de visualizar todos os dados de uma forma organizada, avaliando e analisando toda a informação em pequenas porções. Esta estratégia é a aplicação de filtros e paginação, dois conceitos muito usados para pesquisas orientadas de dados. Para tornar itso possível recorreu-se ao *plugin MVCGrid*, um componente desenvolvido em *jQuery* (*Javascript*) e *Razor* que através de pedidos a um servidor com respostas em formato JSON, preenche assincronamente uma tabela de informação. Esta aplicação torna-se de facto útil quando pensamos em listar todos *badges* dos utilizadores. Estas grandes listagens apresentam um conjunto de colunas ordenáveis, assim com botões para navegação entre páginas ou ainda a existência de filtros para uma pesquisa mais avançada. A Figura 55 ilustra a área de filtragem onde se destacam campos como o tipo de catálogo a pesquisar, o *Issuer*, datas e *tags*. Relativamente às opções de pesquisa no catálogo, estas são baseadas nos repositórios escolhidos para este projeto, referenciados na secção 7.1.4. O valor por defeito é o *Learning Locker* que é o LRS interno da plataforma. O *issuer* pode ser um campo bastante útil numa pesquisa uma vez que permite encontrar a coleção de *badges* emitidos por um *issuer* o que pode aumentar a probabilidade de um utilizador encontrar aquilo que procura. As *tags* por sua vez permitem definir um contexto de pesquisa, em que só aparecerão resultados que estejam associados a essas *tags*. O potencial deste campo de filtragem é imenso, uma vez que permite reduzir o número de elementos na listagem e por sua vez aumentar a probabilidade de um utilizador encontrar aquilo que procura. O resultado de uma filtragem por aplicação das *tags* da Figura 55 está representada na Figura 54.

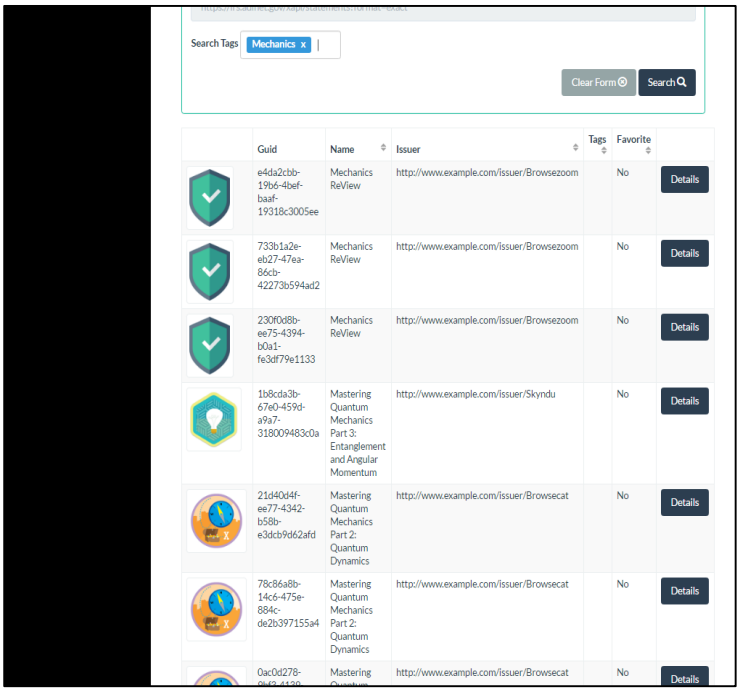

Figura 54 - Página de pesquisa de *badges* com foco na área de listagem

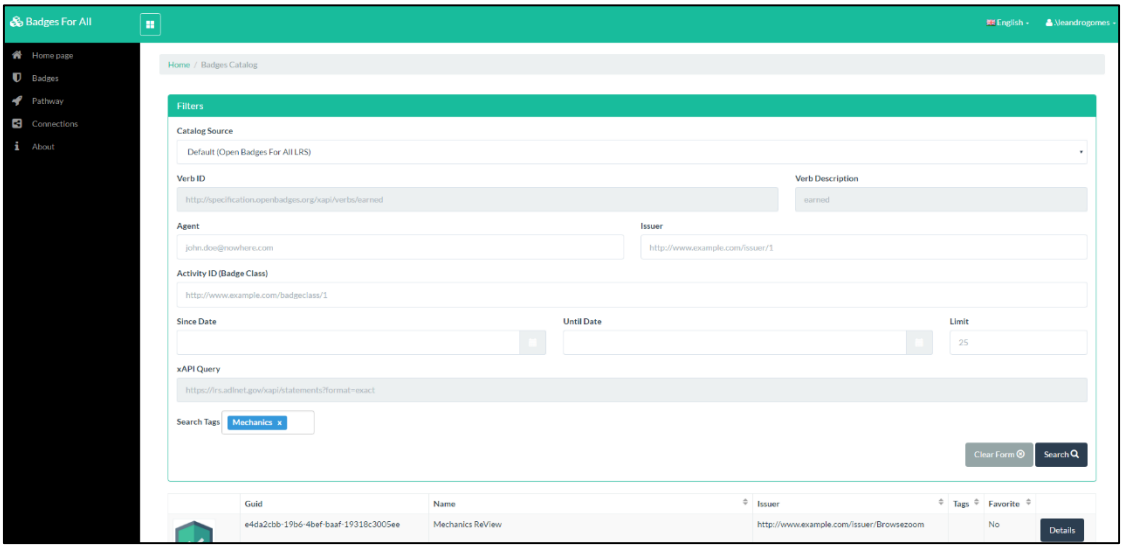

Figura 55 – Página de pesquisa de *badges* com foco na área de filtros

7.3.3.5 Página de recomendações

A página de recomendações é basicamente a página de entrada do utilizador, onde lhe são recomendados *badges* que lhe possam ser relevantes. Na Figura 56, é apresentada a página de recomendações para um utilizador registado. As recomendações apresentadas provêm do serviço de recomendação, que calcula as recomendações em tempo real para cada utilizador que acede à página principal da aplicação.

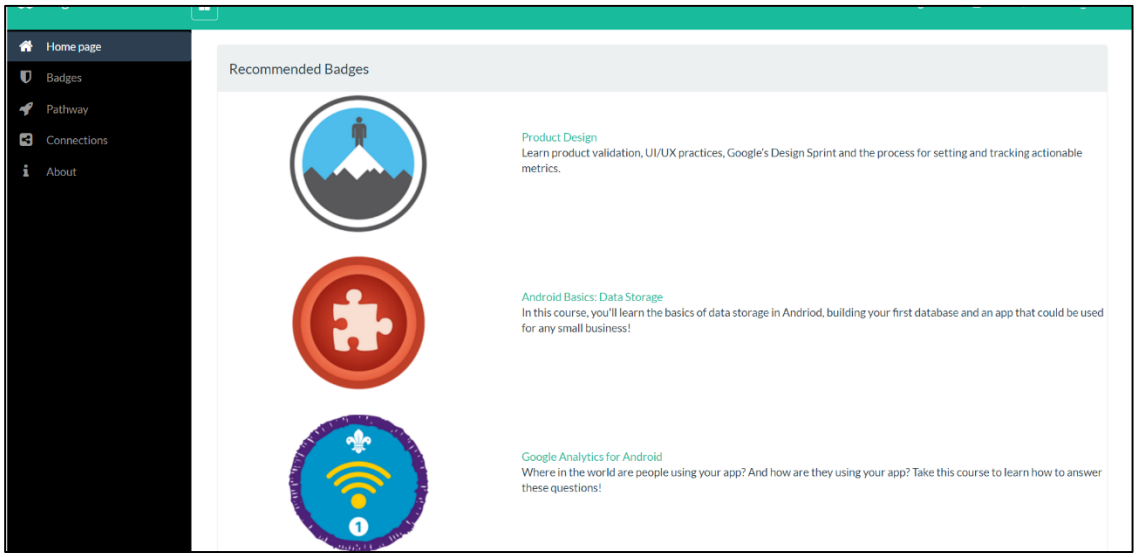

Figura 56 – Página de entrada com recomendações de *badges*

7.4 Testes

Os testes de *software* são uma forma de garantir a qualidade de um desenvolvimento e de um produto final. Testar um *software* é, na prática, realizar um conjunto de atividades planeadas e executadas sistematicamente, que devem seguir alguns critérios como assegurar que os testes se iniciam a um nível modular até aos níveis de integração e sistema (Ahamed 2009).

Como uma atividade integrante do desenvolvimento de *software*, os testes requerem um planeamento detalhado. Durante todo o processo, a elaboração de uma documentação bem estruturada melhora a visibilidade das fases de teste. O padrão IEEE 829-2008 (Society, 2008) descreve um conjunto de oito documentos básicos de teste de *software*. Esta documentação abrange desde a fase de preparação até ao registo de resultados. A metodologia de execução dos testes que é descrita em seguida é baseada no padrão acima mencionado. As atividades relacionadas com cada fase de teste têm como objetivo a produção e manutenção da documentação.

Este padrão divide as atividades de teste em três fases: preparação, execução e registo de teste. Nesse sentido são então criados os seguintes documentos:

- Plano de Testes
- Especificação do projeto de teste
- Especificação dos casos de teste
- Especificação dos procedimentos de teste
- Relatório de encaminhamento dos itens de teste
- Diário de teste
- Registo de incidentes de teste
- Relatório resumo de teste

Apesar de definir um conjunto de documentos de testes de *software* básico este não especifica o conjunto necessário e essencial de documentos, podendo desta forma serem produzidos na totalidade e ou através da simplificação dos mesmos mediante o projeto.

7.4.1 Plano de Testes

No âmbito do planeamento, manutenção e monitorização das atividades relacionadas com o projeto, especificou-se que o planeamento das fases de testes devia incluir:

- A estratégia de implementação e as diferentes fases de testes
- A definição dos testes a realizar
- A definição de critérios de aceitação
- A avaliação dos resultados dos testes
- O modo de documentação, acompanhamento e resolução de problemas durante a realização dos testes

Nesse sentido foi elaborado um plano de testes com o objetivo de diminuir o tempo de manutenção e correção de eventuais erros de implementação ou configuração. Segundo Ron Patton (Patton 2005), os custos de correção de um problema aumentam exponencialmente no decorrer do projeto. Quanto mais tarde for detetado o problema, menor é a possibilidade da existência de uma solução viável que ao mesmo tempo cause pouco impacto, utilize recursos reduzidos e se adapte ao tempo disponível para correção.

A título de exemplo, um erro que seja identificado logo após a fase de definição de requisitos, pode ter como correção somente alguns ajustes na documentação dos requisitos e no planeamento do projeto. Se este mesmo erro apenas for detetado após uma fase de implementação, pode ter como consequência mudanças significativas na própria arquitetura da solução.

7.4.2 Características

A classificação dos tipos de teste é feita tendo como referência a característica da qualidade que se deseja alcançar. Existem testes específicos para se atingir cada uma das características relacionadas pela norma ISO/IEC 9126-1[71]. Na Tabela 14 - Tipos de Características dos Testes, são apresentados os tipos de características de qualidade propostas pela norma referida anteriormente e a sua descrição.

Tabela 14 - Tipos de Características dos Testes

Característica	Descrição
Funcionalidade	Evidenciar que o conjunto de funções atende às necessidades explícitas e implícitas para a finalidade a que se destina o produto.
Confiabilidade	Evidenciar que o desempenho se mantém ao longo do tempo e em condições estabelecidas.
Usabilidade	Evidenciar a facilidade para a utilização do *software*.
Manutenibilidade	Evidenciar que há facilidade para correções, atualizações e alterações.
Portabilidade	Evidenciar que é possível utilizar o produto em diversas plataformas com pequeno esforço de adaptação.

7.4.3 Tipos de Teste

A quantidade de tipos de teste existente é bastante grande e tem um relacionamento direto com o tipo de produto de *software* que está a ser testado, e com o ambiente onde será utilizado. Relativamente às fases de teste, dividiram-se em sete tipos, unitários, de integração, de sistema, de usabilidade, de desempenho, de segurança e de aceitação.

7.4.3.1 Testes Unitários

Nesta fase foram aplicados testes aos componentes de código sob a forma de testes unitários. Numa abordagem TDD, os testes unitários são desenvolvidos antes da implementação de um método e permitem, quando terminados, que se detenha o chamado *self-testing* code. Segundo Fowler (Fowler 2014), ao atingir-se este estado existe confiança no código e que este está livre de erros potencialmente graves. A qualquer momento é possível conhecer o estado do código através da execução de uma série de testes automáticos sobre o código base. O desenvolvimento recorrendo à programação orientada à interface auxilia na criação de testes unitários, uma vez que reduz o número de dependências e permite escrever o teste antes da implementação dos métodos.

Relativamente ao desenvolvimento dos testes unitários, estes foram criados usando Moq e recorrendo à notação AAA, *Arrange-Act-Assert*. Um exemplo de teste unitário desenvolvido é ilustrado no Código 4.

[71] http://www.cse.unsw.edu.au/~cs3710/PMmaterials/Resources/9126-1%20Standard.pdf

```
    [TestClass]
    public class StatementGatewayTests
    {
        [TestInitialize]
        public void Initialize()
        {
            AutoMapperConfiguration.Configure();
        }

        [TestMethod]
        [ExpectedException(typeof(TenantTargetNotFoundException))]
        public void GetStatements_WhenInvalidTenantId_ExpectToReturnNull()
        {
            //Arrange
            var gateway = new StatementGateway();

            Mock<IStatementsTarget> target = new Mock<IStatementsTarget>();
            target.Setup(x =>
x.AggregateCountStatementsQuery(It.IsAny<AggregateCountQuery>()))
                .Returns(new StatementsCountResultLRSResponse
                {
                    Count = 0
                });

            target.Setup(x =>
x.QueryStatements(It.IsAny<StatementsQuery>()))
                .Returns(new StatementsResultLRSResponse());

            //Act
            var result = gateway.GetStatements(0);

            //Assert
            Assert.IsNull(result.Statements);
            Assert.AreEqual(0, result.Total);
        }

        [TestMethod]
        public void
GetStatements_WhenValidTenantId_AndEmptyTarget_ExpectToReturnNull()
        {
            //Arrange
            var gateway = new StatementGateway();

            Mock<IStatementsTarget> target = new Mock<IStatementsTarget>();
            target.Setup(x =>
x.AggregateCountStatementsQuery(It.IsAny<AggregateCountQuery>()))
                .Returns(new StatementsCountResultLRSResponse
                {
                    Count = 0
                });

            var mockAppSettings = new Mock<IConfigurationReader>();
            mockAppSettings.Setup(x =>
x.GetAppSetting(It.IsAny<string>())).Returns("test_key");

            target.Setup(x =>
x.QueryStatements(It.IsAny<StatementsQuery>()))
                .Returns(new StatementsResultLRSResponse());
```

```
            //Act
            var result = gateway.GetStatements(1);

            //Assert
            Assert.IsNull(result.Statements);
            Assert.AreEqual(0, result.Total);
        }

        [TestMethod]
        public void
GetStatements_WhenValidTenantId_AndValidTarget_ExpectToReturnResultWithStat
ements()
        {
            //Arrange
            var gateway = new StatementGateway();

            var mockAppSettings = new Mock<IConfigurationReader>();
            mockAppSettings.Setup(x =>
x.GetAppSetting(It.IsAny<string>())).Returns("test_key");

            Mock<IStatementsTarget> target = new Mock<IStatementsTarget>();
            target.Setup(x =>
x.AggregateCountStatementsQuery(It.IsAny<AggregateCountQuery>()))
                .Returns(new StatementsCountResultLRSResponse
                {
                    Count = 1
                });

            target.Setup(x =>
x.QueryStatements(It.IsAny<StatementsQuery>()))
                .Returns(new StatementsResultLRSResponse
                {
                    content = new StatementsResult
                    {
                        statements = new List<Statement>
                        {
                            new Statement
                            {
                                actor = new Agent
                                {
                                    name = "Test Actor",
                                    mbox = "mail@nowhere.com"
                                },
                                result = new Result
                                {
                                    score = new Score
                                        {
                                            min = 10,
                                            max = 100,
                                            raw = 88
                                        },
                                },
                                verb = new Verb
                                {
                                    display = new LanguageMap(new
Dictionary<string, string>())
                                }
```

```
                        }
                    }
                }
            });

            //Act
            var result = gateway.GetStatements(1);

            //Assert
            Assert.IsTrue(result.Statements.Any());
            Assert.AreEqual(1, result.Total);
        }
    }
}
```

Código 4 – Exemplo de Teste Unitário efetuado ao componente *StatementsGateway* do serviço *BadgesCatalogService*

7.4.3.2 Testes de Integração

Relativamente aos testes de integração, foram testadas as interações entre componentes, nomeadamente para a integração com os diversos microserviços. Para execução desta fase, foi necessário que os respetivos componentes tivessem uma elevada cobertura de testes unitários. Assim sendo, estes testes foram criados à medida que cada microserviço apresentava novos *endpoints* na respetiva API. A esta técnica dá-se o nome de *Behavior Drive Development* (BBB), que visa integrar regras de negócio com as linguagens de programação.

Nesse sentido foi criado um projeto de testes automáticos às APIs dos microserviços, recorrendo à utilização do *Specflow* e do NUnit. Através do *Specflow* foi possível descrever em formato de teste as regras de negócio de uma forma simples. Estes ficheiros *feature* têm a

```
Feature: BadgeStatementsFeature

@statements

Scenario Outline: Get Badge Statements
  Given I want to get badge statements <TenantId>
  Then The API should return statements

Examples:
| TenantId |
| 1        |

Scenario Outline: Get Badge Statements with Filters
  Given I want to get badge statements with filters <TenantId> <Verb>
<Take>
  Then The API should return the expected number of statements

Examples:
```

Código 5 – Excerto de uma *feature* em *Specflow* para a validação de integração com o *BadgesCatalogService*

grande vantagem de serem também uma forma de documentação. No Código 5 é possível observar um excerto de uma *feature* de teste à API do *BadgesCatalogService*

Para cada *feature*, foi criado um ficheiro de *steps* para teste das condições propostas. No Código 6 é apresentado um excerto da classe de *steps* correspondente à *feature* apresentada no Código 5.

```
[Given(@"I want to get badge statements with filters (.*) (.*) (.*)")]
        public void GivenIWantToRetrieveStatements(int tenantId, string
verb, int take)
        {
            var query = new StatementQuery
            {
                Verb = verb
            };

            var paging = new Paging
            {
                Skip = 0,
                Take = take
            };

            NumberOfExpectedStatements = take;

            var request = new HttpRequestWrapper()
                            .SetMethod(Method.GET)
                            .SetResourse("/api/statements/")
                            .AddParameters(new Dictionary<string,
object>() {

{ "tenantId", tenantId },

{ "query", query },

{ "paging", paging }

});

            restResponse = new RestResponse();
            result = request.Execute<StatementsQueryResult>();
        }

        [Then(@"The API should return the expected number of
statements")]
        public void ThenTheAPIReturnedTheExpectedNumberOfStatements()
        {
            Assert.That(() => result.Statements.Count() ==
NumberOfExpectedStatements);
```

Código 6 – Excerto da Classe *steps* para a *feature* de validação de integração com o *BadgesCatalogService*

7.4.3.3 Testes de Sistema

No que se refere aos testes de sistema, estes são aplicados à aplicação em concreto, sendo testadas todas as suas funcionalidades. É também objetivo desta fase de testes, a verificação da conformidade com os requisitos através da simulação de um ambiente de produção real.

8 Avaliação da solução

8.1 LRS

Os LRS escolhidos para a avaliação da solução desenvolvida foram o *Learning Locker* e o *Wax LRS*. Numa primeira abordagem efetuou-se uma análise comparativa das limitações de cada pesquisa para cada um dos repositórios.

Tabela 15 – Listagem de suporte a parâmetros de pesquisa pelo *Learning Locker* e *WAX LRS*

Pesquisa (parâmetros)	***Learning Locker***	***WAX LRS***
StatementId	Sim	Sim
Actor	Sim	Sim
Verb	Sim	Sim
Activity	Sim	Sim
Since	Sim	Sim
Until	Sim	Sim
Paging	Sim	Sim
Attachments	Não	Sim
Assertion	Sim	Não
BadgeClass	Sim	Não
Issuer	Sim	Não
Image	Sim	Sim
Tags	Sim	Não
Criteria	Sim	Não

Como é possível verificar na tabela anterior, o *Learning Locker* apresenta uma grande flexibilidade na sua pesquisa, e grande parte advém da utilização de base de dados Mongo e ter

suporte a *aggregation pipeline*. Na realidade, a especificação xAPI e a adaptação dos *Open Badges* faz com que algumas propriedades relevantes para representação de um *badge* fiquem bastante hierarquizadas no documento. Sem a possibilidade de executar as pesquisas numa *pipeline*, torna a utilização e um LRS pouco vantajoso para os *Open Badges*.

8.2 Mecanismo de Recomendações

Relativamente à avaliação dos mecanismos de recomendações abordados, foi efetuado primariamente um levantamento das variáveis de dependentes e independentes. Neste caso, a variável independente são os mecanismos de recomendação, *Collaborative Filtering* e o cálculo do *Personal Value*. Para realizar esta análise, foi efetuada uma análise comparativa entre os dois mecanismos para três áreas distintas e com amostras também distintas, nomeadamente, *Android*, *Web Development* e *Data Science*. Desta forma, a população utilizada para esta análise apresentava um total de 979 *badges*, dos quais 115 utilizadores possuíam *badges* relacionados com a área de *Android*, 155 da área de *Web Development* e 81 da área de *Data Science*. Na Figura 57 são apresentadas as proporções destas áreas para o total da população de teste.

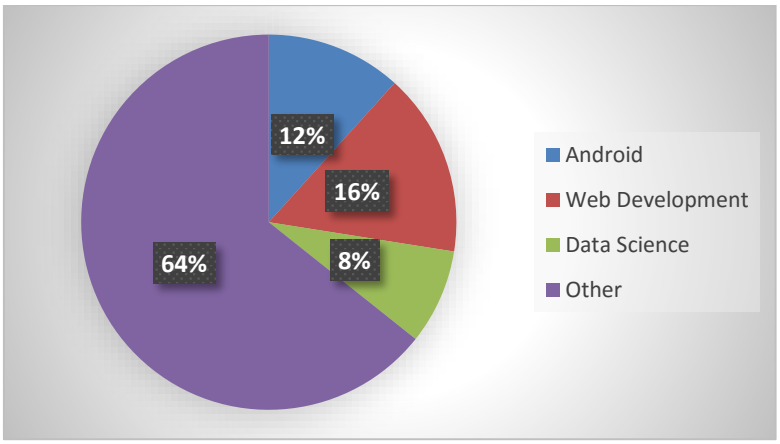

Figura 57 - Distribuição das áreas em estudo pela população

O objetivo desta análise era comparar a exatidão das recomendações para um contexto restrito de cada área. Os utilizadores de teste tinham *badges* dessas mesmas áreas e pretendia-se verificar qual o mecanismo que, no geral, apresentava as cinco recomendações mais relevantes para a área de interesse do utilizador. Como teste controlo, foi escolhido um utilizador do sistema sem interesse particular por nenhuma dessas áreas. O objetivo deste primeiro teste era verificar que os algoritmos subjacentes a cada mecanismo apresentavam um comportamento neutro. Cada teste a uma determinada área foi interpretado como um ensaio comparativo entre os mecanismos. Na Tabela 16 são apresentados os resultados obtidos pelo ensaio controlo.

Para cada área, foram criados dois testes com utilizadores diferentes mas ambos com interesse na mesma área. Nas tabelas 17 a 22, são apresentados os resultados dos testes obtidos.

Tabela 16 – Resultados do teste para o utilizador controlo

Utilizador	***Badges Pessoais***	***Collaborative***	***Personal Value***
	Badge	***Badge***	***Badge***
Test User Control (jlynchr@ wunderground.com)	Data Wrangling with MongoDB (*Software Engineering, Data Science)*	Offline Web Applications (Web Development)	Android TV and Google Cast Development (Android)
	Rapid Prototyping (Non-Tech)	Origins of the Human	Advanced Design and Analysis of Algorithms (Alghoritms)
	Fundamentals of Programming (Algorithms)	Principles and Practice of Computer Aided Translation	International Political Economy
	Ancient History of China (History)	Historical Relic Treasures and Cultural China: Part2 (History)	App Marketing (Non-Tech)
	Chinese History From Warring States to the Tang Dynasty (History)	Fun with Prime Numbers: The Mysterious World of Mathematics	Visualizing Algebra (Non-Tech)
	Tales from the Genome (Non-Tech)		
	Intro to Physics (Non-Tech)		

Tabela 17 – Resultados do primeiro teste na área *Android*

Utilizador	***Badges Pessoais (Android)***	***Collaborative***	***Personal Value***
		Badges	***Badges***
Test Android User 1	Android Wear Development	How to Use Git and GitHub (Web Development, Software Engineering)	Software Debugging (Software Engineering)

	Android Basics: Multi-screen Apps	A/B Testing (Data Science)	**Google Analytics for Android** (Android)
	Android Basics: Networking	**Intro to Java Programming** (Software Engineering, Android)	The Extremes of Life: Microbes and Their Diversity (Non-Tech)
	Android Wear Development	**Advanced Android App Development** (Android)	Intro to Hadoop and MapReduce (Data Science, Software Engineering)
	Android Basics: Multi-screen Apps	**Android Basics: User Interface** (Android)	Intro to Inferential Statistics (Data Science)
	Android Basics: Networking		
	Android Wear Development		

Tabela 18 - Resultados do segundo teste na área *Android*

Utilizador	***Badges Pessoais (Android)***	***Collaborative Badges***	***Personal Value Badges***
Test Android User 2	UX Design for Mobile Developers	Machine Learning (Data Science)	Product Design (Non-Tech)
	Intro to Java Programming	**Android Basics: Multi-screen Apps** (Android)	**Android Basics: Data Storage** (Android)
	Android Wear Development	Mechanics: Momentum and Energy	**Google Analytics for Android** (Android)
	Material Design for Android Developers	Data Structures	Segmentation and Clustering
	Advanced Android App Development	Making Biologic Medicines for Patients: The Principles of Biopharmaceutical Manufacturing	Global Warming Science
	Android Basics: Networking		
	Android Basics: User Input		

Tabela 19 - Resultados do primeiro teste na área *Web Development*

Utilizador	***Badges Pessoais (Web Development)***	***Collaborative Badges***	***Personal Value Badges***
Test Web Development User 1	Mobile Web Development	**Configuring Linux Web Servers** (Web Development)	**Responsive Images** (Web Development)
	Full Stack Foundations	**Responsive Images** (Web Development)	**Full Stack Foundations** (Web Development)
	Intro to Relational Databases	**Offline Web Applications** (Web Development)	College Algebra
	How to Use Git and GitHub	**Intro to jQuery** (Web Development)	**Website Performance Optimization** (Web Development)
	Front End Frameworks	**Web Tooling & Automation** (Web Development)	Differential Equations in Action
	Full Stack Foundations		

Tabela 20 - Resultados do segundo teste na área *Web Development*

Utilizador	***Badges Pessoais (Web Development)***	***Collaborative Badges***	***Personal Value Badges***
Test Web Development User 2	Website Performance Optimization	**Mobile Web Development** (Web Development)	**Mobile Web Development** (Web Development)
	Configuring Linux Web Servers	**Web Tooling & Automation** (Web Development)	Artificial Intelligence for Robotics (Software Engineering,Georgia Tech Masters in CS)
	JavaScript Testing	Technical Interview	C++
	Developing Scalable Apps in Python	Xcode Debugging (iOS)	Programming Languages (Software Engineering)
	Responsive Images	Intro to Machine Learning (Data Science)	Data Structures and Algorithms, Part 2

	Responsive Web Design Fundamentals		
	How to Use Git and GitHub		
	Offline Web Applications		

Tabela 21 - Resultados do primeiro teste na área *Data Science*

Utilizador	***Badges Pessoais (Data Science)***	***Collaborative Badges***	***Personal Value Badges***
Test Data Science User 1	Machine Learning	**Model Building and Validation** (Data Science)	**Model Building and Validation** (Data Science)
	Reinforcement Learning	**Intro to Inferential Statistics** (Data Science)	Linux Command Line Basics (Web Development)
	Intro to Descriptive Statistics	Advanced Android App Development (Android)	Principles of Electric Circuits: Part 2
	Data Analysis with R	Android Basics: Networking (Android)	Relics of Chinese History - Part 2: Astronomy and Medicine (History)
	Intro to Data Science	Android Basics: User Input (Android)	Advanced Android App Development (Android)
	Intro to Hadoop and MapReduce		
	Intro to Computer Science		
	Machine Learning		

Tabela 22 - Resultados do segundo teste na área *Data Science*

Utilizador	***Badges Pessoais (Data Science)***	***Collaborative Badges***	***Personal Value Badges***
Test Data Science User 2	Model Building and Validation	How to Use Git and GitHub (Web Development,Software Engineering)	**Intro to Computer Science** (Data Science)

	Intro to Inferential Statistics	**Machine Learning** (Data Science)	Linux Command Line Basics (Web Development)
	Intro to Descriptive Statistics	**Reinforcement Learning** (Data Science, Georgia Tech Masters in CS)	**Data Analysis with R** (Data Science)
	A/B Testing	**Data Analysis with R** (Data Science)	Intro to Relational Databases (Web Development)
		Intro to Data Science (Data Science)	**Intro to Hadoop and MapReduce** (Data Science)

Como é possível verificar nas tabelas anteriores, o mecanismo *Collaborative Filtering* conseguiu apresentar recomendações mais relevantes do que o mecanismo *Personal Value*. Na Tabela 23 são apresentados os resultados sob a forma de percentagem das recomendações da área pelo total de recomendações.

Tabela 23 – Resultados dos ensaios realizados por área

Ensaio	**Percentagem de recomendações média por ensaio**	
	Collaborative	***Personal Value***
1 – Android	40%	30%
2 – Web Development	70%	20%
3 – Data Science	60%	40%

Para comprovar esta conclusão, aplicou-se o teste estatístico de Wilcoxon Signed-Rank, tendo-se obtido o valor de p-value de 0,00939 que é inferior ao nível de significância 0,05, logo a hipótese formulada pode ser considerada válida. O mecanismo de *Collaborative Filtering* será então o escolhido como ponto de partida para o possível teste com utilizadores reais. O objetivo é evoluir o sistema de recomendações para que este seja o mais eficaz possível e consiga apresentar ao utilizador *badges* relevantes.

9 Conclusões

Neste capítulo é resumido o trabalho efetuado destacando as suas principais contribuições. São respondidas as questões elaboradas no início da dissertação e, no final, descreve-se o trabalho futuro a ser concretizado para a implementação e disponibilização da solução com foco em todas as entidades envolvidas no processo de emissão, certificação e receção de *Open Badges*.

9.1 Trabalho realizado

O presente trabalho revelou-se muito aliciante, na medida em que foi possível explorar uma nova forma de atestar competências adquiridas e perceber um pouco o caminho que está a ser desenvolvido a nível de sistemas de certificação de aprendizagens, formais ou informais.

Este trabalho teve uma componente importante na análise de soluções e ferramentas no contexto dos *Open Badges* e permitiu com o devido enquadramento, identificar as limitações existentes na descoberta de *Open Badges*, nomeadamente:

- Segmentação das ferramentas dos *Open Badges* e orientação para apenas uma entidade do sistema
- Desconhecimento das entidades emissoras e de *Open Badges* disponíveis
- Ecossistema relativamente pequeno
- Pouco reconhecimento do valor associado a um *Open Badge*

Foi necessário entender a arquitetura geral do ecossistema existente, os processos associados à mesma e a metodologia de desenvolvimento utilizada (quais os seus conceitos principais e de que forma esta poderia trazer benefícios para a solução a desenvolver). O objetivo deste trabalho incidiu na proposta de uma solução ao problema da descoberta de oportunidades de aprendizagem. Após análise do estado da arte e dos sistemas e infraestrutura existentes, foi

possível desenhar uma solução para responder ao problema levantado. Este desenho surgiu das escolhas efetuadas de acordo com as suas principais características, dos conceitos teóricos associados, das principais vantagens e desvantagens e de quais os contextos em que determinada solução é escolhida em detrimento de outra.

O projeto *Open Badges For All* agrupa um conjunto de funcionalidades de foco no *earner*, recetor de *badges*, e disponibiliza a pesquisa de *badges* com filtros avançados, a construção de um percurso de aprendizagem, bem como fornece recomendações de *badges* relevantes.

A solução implementada baseou-se numa arquitetura de microserviços, modular e flexível, de tal forma que é possível, através de um sistema externo, integrar apenas parte dos componentes ou serviços. Na realidade, um dos problemas da área de aprendizagem formal ou informal, prende-se com o facto de existir uma diversidade de especificações e empresas que utilizam os seus próprios protocolos e processos, o que dificulta a difusão e crescimento, tanto de *Badges* como da especificação xAPI. Este projeto teve por grande objetivo, a interoperabilidade entre sistemas e especificações. O projeto como prova de conceito, permitiu atingir uma solução mais transversal, através da adaptação da especificação *Open Badges* à xAPI. Esta adaptação permite contribui para o fomentar o crescimento dos *Open Badges*, uma vez que assim é possível que os *Open Badges* coexistam juntamente com outro tipo de experiências de aprendizagem.

9.2 Trabalho Futuro

Poderá ser necessário ajustar vários aspetos teóricos para melhor corresponder às necessidades dos utilizadores e permitir uma melhor experiência de utilização, bem como acrescentar funcionalidades relevantes.

Um dos objetivos principais será também a disponibilização do código fonte numa plataforma de partilha de projetos como o Github, para, se possível, arranjar colaboração para posteriores desenvolvimentos e testes da plataforma.

Perfila-se muito trabalho pela frente para concluir e disponibilizar a solução desenvolvida durante a dissertação. Os resultados, a arquitetura apresentada e o protótipo desenvolvido representam um bom ponto de partida para a implementação da solução completa para *issuers*, *earners* e revisores. Questões de performance, otimizações e ajustes nos serviços poderão ter de ser equacionados.

Referências

Abramovich, S., Schunn, C. & Higashi, R.M., 2013. Are badges useful in education?: It depends upon the type of badge and expertise of learner. *Educational Technology Research and Development*, 61(2), pp.217–232.

ADL, 2013. Experience API. Available at: https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-About.md#partone [Accessed August 15, 2016].

Ahamed, S.S.R., 2009. STUDYING THE FEASIBILITY AND IMPORTANCE OF SOFTWARE TESTING: AN ANALYSIS. *Internatinal Journal of Engineering Science and Technology*, 1(3), pp.119–128.

Alliance, B., 2016a. Open Badges Extensions. Available at: https://openbadgespec.org/extensions/ [Accessed February 1, 2016].

Alliance, B., 2016b. Open Badges Technical Specification. Available at: http://specification.openbadges.org/ [Accessed February 6, 2016].

Asanov, D., 2011. *Algorithms and Methods in Recommender Systems*, Berlin, Germany. Available at: http://www.uic.edu.hk/~alex/Algorithms and Methods in Recommender Systems.pdf.

Association for Project Management, 2014. *Introduction to Gamification*, Available at: https://www.apm.org.uk/sites/default/files/gamification - epdf.pdf.

Behringer, R., 2013. Interoperability Standards for MicroLearning.

Berking, P., 2016. *Choosing a Learning Record Store (LRS)*, Available at: https://www.adlnet.gov/.

Casilli, C. & Hickey, D., 2016. Transcending conventional credentialing and assessment paradigms with information-rich digital badges. *The Information Society*, 32(2), pp.117–129. Available at: http://www.tandfonline.com/doi/full/10.1080/01972243.2016.1130500.

Cattell, R., 2010. Scalable SQL and NoSQL Data Stores.

Chen, Y., Xu, X. & Zhu, L., 2012. *Web Platform API Design Principles and Service Contract - Pesquisa Google*, Available at: https://www.google.pt/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=Web+Platform+API+Design+Principles+and+Service+Contract.

Cross, S., Whitelock, D. & Galley, R., 2014. The use, role and reception of open badges as a method for formative and summative reward in two Massive Open Online Courses. *International Journal of e-Assessment*, pp.1–16. Available at: http://oro.open.ac.uk/40593/1/__userdata_documents_sc8457_Documents_Assessment_Journal Paper 2014_Cross2014_UseRoleReceptionOfOpenBadges.pdf.

Davis, K. & Singh, S., 2015. Digital badges in afterschool learning: Documenting the perspectives and experiences of students and educators. *Computers & Education*, 88, pp.72–83. Available at: http://www.sciencedirect.com/science/article/pii/S0360131515001128.

Deterding, S. et al., 2011. From game design elements to gamefulness. *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11*, pp.9–11. Available at: http://doi.acm.org/10.1145/2181037.2181040\nhttp://dl.acm.org/citation.cfm?doid=2181037.2181040.

District, M.D., 2012. Management Crafting Your Value Proposition. *MaRS FUndamentals of Entrepreneurial Management*, (November), p.25.

Doherty, I. & Sharma, N., 2015. Aligning the Use of Portfolios with Digital Badging. *British Journal of Hospital Medicine*, 76(10), pp.596–598. Available at: http://dx.doi.org/10.12968/hmed.2015.76.10.596.

Dona, K.L. et al., 2014. Badges in the Carpe Diem MOOC Authors : The Carpe Diem MOOC : The Design. , pp.1–10.

Downes, A., 2015. Andrew Downes - 3/7 - Tin Can API. Available at: https://tincanapi.com/author/andrewdownes/page/3/.

Elliott, R., Clayton, J. & Iwata, J., 2014. Exploring the use of micro-credentialing and digital badges in learning environments to encourage motivation to learn and achieve. *Rhetoric and Reality: Critical perspectives on educational technology. Proceedings ascilite Dunedin 2014*, pp.703–707.

Farber, M., 2013. Gamifying Student Engagement | Edutopia. Available at: http://www.edutopia.org/blog/gamifying-student-engagement-matthew-farber [Accessed February 11, 2016].

Finkelstein, J., Knight, E. & Manning, S., 2013. The Potential and Value of Using Digital Badges for Adult Learners: DRAFT for Public Comment. Available at: https://lincs.ed.gov/professional-development/resource-collections/profile-716.

Fowler, M., 2014. SelfTestingCode. Available at: http://martinfowler.com/bliki/SelfTestingCode.html [Accessed July 21, 2016].

Frith, J., 2013. Turning life into a game: Foursquare, gamification, and personal mobility. *Mobile Media & Communication*, 1(2), pp.248–262.

Gamrat, C. & Zimmerman, H.T., 2015. An Online Badging System Supporting Educators STEM Learning. *2nd International Workshop for Open Badges in Education*, 1358, pp.12–23. Available at: http://ceur-ws.org/Vol-1358/.

Glahn, C., 2013. xAPI, Open Badges and E-Portfolios - Christian Glahn. Available at: https://lo-f.at/glahn/2013/07/xapi-open-badges-and-e-portfolios.html.

Glover, I. & Latif, F., 2013. Investigating perceptions and potential of Open Badges in formal higher education. *World Conference on Educational Multimedia, Hypermedia and*

Telecommunications 2013, pp.1398–1402. Available at: http://shura.shu.ac.uk/7173/.

Groh, F., 2012. Gamification: State of the Art Definition and Utilization. *Research Trends in Media Informatics*, pp.39–46. Available at: http://d-nb.info/1020022604/34/#page=39.

Haines, K., 2009. To Redis or Not To Redis? (Key-Value Stores Part 4). Available at: https://blog.engineyard.com/2009/key-value-stores-for-ruby-part-4-to-redis-or-not-to-redis [Accessed July 21, 2016].

Hickey, D. et al., 2015. From Learning Evidence to Learning Analytics. *2nd International workshop on open badges in education (OBIE 2015)*, (Obie), pp.392–393.

Hickey, D.T., Iii, J.E.W. & Quick, J.D., 2015. Where Badges Work Better. , (June), pp.1–13.

Hunter, T., 2013. Principles of good RESTful API Design - Code Planet. Available at: https://codeplanet.io/principles-good-restful-api-design/ [Accessed August 12, 2016].

Ian O'Byrne, W. et al., 2015. Digital Badges - Recognizing, Assessing, and Motivating Learners In and Out of School Contexts. *Journal of Adolescent & Adult Literacy*, 58(6), pp.451–454. Available at: http://www.readcube.com/articles/10.1002%2Fjaal.381?r3_referer=wol&tracking_action=preview_click&show_checkout=1&purchase_referrer=onlinelibrary.wiley.com&purchase_site_license=LICENSE_DENIED_NO_CUSTOMER [Accessed February 19, 2016].

Ian Quillen, 2013. How Mozilla's Open Badges May Work In the Real World. Available at: http://ww2.kqed.org/mindshift/2013/03/25/how-mozillas-open-badges-may-work-in-the-real-world/ [Accessed February 6, 2016].

Jacobson, D., Bail, G. & Woods, D., 2012. *APIs A Strategy Guide*, O'Reilly Media, Inc. Available at: http://droppdf.com/v/kTWdS.

Joseph, B., 2012. Six Ways to Look at Badging Systems Designed for Learning | Online Leadership Program. Available at: http://www.olpglobalkids.org/content/six-ways-look-badging-systems-designed-learning [Accessed February 11, 2016].

Jovanovic, J. & Devedzic, V., 2014. Open Badges: Novel Means to Motivate, Scaffold and Recognize Learning. *Technology, Knowledge and Learning*, 20(1), pp.115–122.

Kriauciunas, Nerijus , Ragauskas, L., 2013. Unique learning Badges to recognise non-formal learning in european Youth Work. , pp.10–11.

Lee, J.J.J. & Hammer, J., 2011. Gamification in Education: What , How , Why Bother? *Academic Exchange Quarterly*, 15(2), pp.1–5. Available at: http://dialnet.unirioja.es/servlet/articulo?codigo=3714308.

Lewis, J. & Fowler, M., 2014. Microservices. Available at: http://martinfowler.com/articles/microservices.html.

Lim, K.C., 2015. Case Studies of xAPI Applications to E-Learning. *The Twelfth International Conference on eLearning for Knowledge-Based Society,*, pp.11–12.

Loi, R. et al., 2015. Badge It! *Journal of Managerial Psychology*, Vol. 30 No, pp.645–658.

MongoDB, Agile Development Environment | MongoDB. Available at: https://www.mongodb.com/scale/agile-development-environment [Accessed July 15, 2016].

Mozilla, Badges - MozillaWiki. Available at: https://wiki.mozilla.org/Badges/ [Accessed February 6, 2016].

Mozilla, 2016a. Discover Open Badges. Available at: http://discover.openbadges.org/ [Accessed February 6, 2016].

Mozilla, 2015. O que é o projeto Open Badges da Mozilla? | Ajuda de Open Badges. Available at: https://support.mozilla.org/pt-PT/kb/o-que-e-o-projeto-open-badges-da-mozilla [Accessed February 6, 2016].

Mozilla, 2011. Open Badges for Lifelong Learning. *White Paper*, pp.1–14. Available at: https://wiki.mozilla.org/File:OpenBadges-Working-Paper_092011.pdf.

Mozilla, 2016b. Share your Badges with the Mozilla Backpack. Available at: https://github.com/mozilla/openbadges-backpack/wiki/Share-your-Badges-with-the-Mozilla-Backpack [Accessed February 12, 2016].

Mozilla MacArthur Foundation, 2016. About | Open Badges. Available at: http://openbadges.org/about/ [Accessed February 6, 2016].

Newman, S., 2015. *Building Microservices*, O'Reilly Media Inc.

NodeJS, 2016. About | Node.js. Available at: https://nodejs.org/en/about/ [Accessed February 7, 2016].

Patton, R., 2005. *Software Testing* 2nd., Sams Publishing.

Pearson Learning Solutions, 2013. Open Badges for Higher Education. Available at: http://www.pearsonlearningsolutions.com/blog/wp-content/uploads/2013/12/Open-Badges-for-Higher-Education.pdf.

Redmond, E. & Wilson, J.R., 2012. *Seven Databases in Seven Weeks A Guide to Modern Databases and the NoSQL Movement* 1st ed., Dallas, Texas, North Carolina: Pragmatic Programmers, LLC.

Rughinis, R., 2013. Talkative Objects in Need of Interpretation. Re-Thinking Digital Badges in Education.

Sandeen, C. (American C. on E., 2013. Assessment's Place in the New MOOC World. *Research and Practice in Assessment*, 8(Summer), pp.5–12.

Santos, Carlos, Pedro, Luís, Almeida, Sara, Aresta, M., 2012. Decentralized badges in educational contexts: the integration of open badges in sapo campus. , (December), pp.1–10.

Santos, J.L. et al., 2015. Tracking Data in Open Learning Environments.

Stiggins, R.J., 2002. Assessment crisis: The absence of assessment for learning. *Phi Delta*

Kappan, 83(10), p.758. Available at: http://search.ebscohost.com/login.aspx?direct=true&db=afh&AN=6755928&site=ehost-live\nhttp://content.ebscohost.com/ContentServer.asp?T=P&P=AN&K=6755928&S=R&D=afh&EbscoContent=dGJyMNHr7ESeqLM4v+vlOLCmr0uep7ZSr6a4S7eWxWXS&ContentCustomer=dGJyMPGptky2q7NNueP.

Subbu Allamaraju, 2011. Performance of RESTful Apps – Subbu Allamaraju. Available at: https://www.subbu.org/blog/2011/03/performance-of-restful-apps.

Wilson, M., 2013. *Building Node Applications with MongoDB and Backbone* 1st ed., Sebastapol, California,: O'Reilly Media, Inc.

Yahoo, 2013. Best Practices for Speeding Up Your Web Site - Yahoo Developer Network. Available at: https://developer.yahoo.com/performance/rules.html#num_http= [Accessed September 15, 2016].

Zeithaml, V.A., 1988. Consumer perceptions of price, quality, and value: a means-end model and synthesis of evidence. *The Journal of marketing*, pp.2–22.

Zhang, J., Kong, X. & Yu, P.S., 20116. Badge System Analysis and Design.

Anexo A

Na figura seguinte, é apresentado um exemplo de um objeto *Assertion no* formato do JSON-LD aplicado no contexto dos *Open Bagdes*.

```
{
  "@context": "https://w3id.org/openbadges/v1",
  "type": "Assertion",
  "id": "https://example.org/beths-robotics-badge.json",
  "uid": "f2c20",
  "recipient": {
    "type": "email",
    "hashed": true,
    "salt": "deadsea",
    "identity":
"sha256$c7ef86405ba71b85acd8e2e95166c4b111448089f2e1599f42fe1bba46e865c5"
  },
  "image": "https://example.org/beths-robot-badge.png",
  "evidence": "https://example.org/beths-robot-work.html",
  "issuedOn": 1359217910,
  "expires": 1609458300,
  "badge": "https://example.org/robotics-badge.json",
  "verify": {
    "type": "hosted",
    "url": "https://example.org/beths-robotics-badge.json"
  }
}
```

Fonte: https://openbadgespec.org/examples/#Assertion

No que se refere ao objeto *BadgeClass*, este possui detalhes sobre a obtenção do *badge* e possui propriedades tais como o nome da conquista, a descrição da mesma, a imagem representativa, o critério, o emissor, entre outras. A figura seguinte representa um exemplo do objeto *BadgeClass*.

```
{
  "@context": "https://w3id.org/openbadges/v1",
  "type": "BadgeClass",
  "id": "https://example.org/robotics-badge.json",
  "name": "Awesome Robotics Badge",
  "description": "For doing awesome things with robots that people think is
pretty great.",
  "image": "https://example.org/robotics-badge.png",
  "criteria": "https://example.org/robotics-badge.html",
  "tags": ["robots", "awesome"],
  "issuer": "https://example.org/organization.json",
```

```
  "alignment": [
    { "name": "CCSS.ELA-Literacy.RST.11-12.3",
      "url": "http://www.corestandards.org/ELA-Literacy/RST/11-12/3",
      "description": "Follow precisely a complex multistep procedure when
      carrying out experiments, taking measurements, or performing technical
      tasks; analyze the specific results based on explanations in the text."
    },
    { "name": "CCSS.ELA-Literacy.RST.11-12.9",
      "url": "http://www.corestandards.org/ELA-Literacy/RST/11-12/9",
      "description": " Synthesize information from a range of sources (e.g.,
      texts, experiments, simulations) into a coherent understanding of a
      process, phenomenon, or concept, resolving conflicting information when
      possible."
    }
  ]
}
```

Fonte: https://openbadgespec.org/examples/#BadgeClass

O objeto *Issuer* representa os metadados da entidade emissora do *badge* e encontra-se referenciado no objeto *BadgeClass* na propriedade *Issuer* como um URL. Na figura seguinte está representado um exemplo do objeto *Issuer*.

```
{
  "@context": "https://w3id.org/openbadges/v1",
  "type": "Issuer",
  "id": "https://example.org/organization.json",
  "name": "An Example Badge Issuer",
  "image": "https://example.org/logo.png",
  "url": "https://example.org",
  "email": "steved@example.org",
  "revocationList": "https://example.org/revoked.json"
}
```

Fonte: https://openbadgespec.org/examples/#Issuer

Anexo B

O *issuer* (emissor) terá as seguintes ações perante o sistema:

- Criação de *badges* via editor na plataforma
- Acesso à biblioteca de *badges* públicos, podendo reutilizar *badges já existentes*
- Tornar *badge público* (publicar)
- *Upload* de *badges* para o sistema
- Emitir *badge* para o *earner* (código de obtenção por email)
- Criação de metadados (Evidências e critérios)
    - Criação de links para páginas *web*
    - Incluir metadados para serem compatíveis com a OBI
    - Adicionar novos metadados e extensões aos mesmos
    - Adicionar categorização aos metadados
- Ligar *badge* a outros *badges* (quando um conjunto de *badges* dá origem a um *badge* global)
- Criar set de *badges* (2 ou mais *badges* que em conjunto dão origem a um *badge* global)
- Atribuição de tags aos *badges* para pesquisa
- Atribuir pontuação aos *badges*
- *Badge baking (automático na criação)*
- Organizar *badges* em folders
- Atualizar e/ou remover *badges* (com log de histórico)

O *earner* terá as seguintes ações perante o sistema:

- Aceitar/recusar *badges*
- Adicionar metadados ao *badge*
- Adicionar *tags* para pesquisa
- Publicar *badge* (exterior ou página interna, fica visível para os outros utilizadores se for marcado como público)
- Organizar *badges*
- Criar *link* para partilha
- Pesquisar *badges* públicos
- Marcar badges como favoritos
- Listar *badges* possíveis de ganhar (pesquisa por metadados e categorização anexada) com detalhe do issuer, critérios, evidências e processo de submissão
- Construir e visualizar árvore de *badges (percurso de aprendizagem com os já obtidos e os que faltam atingir)*

- Ver *badges* recomendados com base nos seus interesses

O revisor terá a função de *endorsement* perante os *badges*:

- Validar e "assinar" o *badg*e se solicitado antes da emissão para o *earner*
- Pesquisar *badges emitidos por entidade emissora*

Salientam-se também algumas funcionalidades transversais, tais como:

- Página de perfil (Se for *issuer* possuir um tipo (Individual, Empresa, Universidade, Colégio, Centro de formação, Departamento, Curso, etc…)
- Sugerir *badges* (possibilidade de enviar email ou notificação para issuers que possam estar na mesma área do *badge sugerido)*
- Mecanismo de autenticação de contas e verificação de permissões nas interações com o utilizador