Robert Hafner BSc.

Open Badges for Certification of Lecturers and Universities

Master's Thesis

to achieve the university degree of

Master of Science

Master's Degree Programme: Software Engineering and Management

submitted to

Graz University of Technology

Supervisor

Adjunct Prof. PhD for media informatics. Martin Ebner
Institute of Interactive Systems and Data Science (ISDS)

Graz, October 2017

Robert Hafner BSc.

Open Badges zur Zertifizierung von Hochschullehrende und Universitäten

Masterarbeit

für den akademischen Grad

Diplom-Ingenieur

Masterstudium: Software Engineering and Management

an der

Technischen Universität Graz

Begutachter

Priv.-Doz. Dipl.-Ing. Dr.techn. Martin Ebner
Institute of Interactive Systems and Data Science (ISDS)

Graz, Oktober 2017

Diese Arbeit ist in englischer Sprache verfasst.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The document uploaded to TUGRAZonline is identical to the present thesis.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Dokument ist mit der vorliegenden Arbeit identisch.

Date/Datum

Signature/Unterschrift

Abstract

One approach to communicate achievements, skills or interest of people and organisation is Mozilla's Open Badges. These objects are regarded as digital images, which are presented on social media sites. However, Open Badges are much more. It is an elaborate system of issuing, receiving and verifying awards. It consists of an ecosystem and a vocabulary of various data classes which are linked.

The working group, "Open Educational Resources" of Forum Neue Medien in der Lehre Austria, developed a concept to certificate Austrian universities and lecturer for their commitment and engagement to Open Educational Resources with Open Badges. Different parts of requirements such as further training or providing repositories have to be fulfilled, so that levels of certain Open Badges which depends on the numbers of parts can be issued.

This thesis answers the question if Open Badges are usable to certificate Austrian universities and lecturer, is TYPO3 Content Management System the adequate system to meet the technical and conceptual requirements so that this system is a success. The project was separated into two parts. One part was the development of a prototype in TYPO3 to meet the requirements in an easy and effective way. The other part was an analysis of Open Badges in Education to find out what works best so that these results are taken into consideration to decide whether to use TYPO3 or not.

The result is the recommendation not to use TYPO3 for an Open Badge issuer system. Instead, three alternatives were presented: the development by usage of a web application framework such as Zend Framework, Laravel, Spring Framework or Django, the usage of an online service Open Badges Factory or the fork of the issuer system Badgr.io.

Kurzfassung

Eine Möglichkeit, damit Personen oder Organisationen für ihre Leistungen ausgezeichnet werden, ihre Interessen oder Fähigkeiten bekannt geben, ist Mozillas Open Badges. Diese Objekte sind nicht nur einfache digitale Bilder, die man der Öffentlichkeit auf Plattformen von sozialen Netzwerken präsentiert. Nein, sie sind viel mehr, Open Badges sind ein ausgereiftes System um Auszeichnungen auszustellen, zu empfangen und zu verifizieren. Sie bestehen aus einem Ökosystem und einem Vokabular von Daten Klassen, die untereinander verlinkt sind.

Die Arbeitsgruppe „Open Educational Resources“ vom Verein für neue Medien Austria entwickelte ein Konzept, um Open Badges dazu zu benützen, österreichische Hochschulen und Hochschullehrende, die sich dadurch auszeichnen, dass sie sich zu Open Educational Resources bekennen, Ressourcen zur Verfügung stellen Dienste anbieten und für den Bereich engagieren, indem sie sich weiterbilden oder die Metadaten ihrer Lehrmaterialen bekannt geben.

Diese Masterarbeit beantwortet die Frage, ob TYPO3, ein Content Management System, das adäquate Mittel ist, um Hochschullehrende und österreichische Universitäten mit Open Badges auszuzeichnen und somit erfolgreich eingesetzt werden kann. Diese Arbeit wurde in zwei Teile unterteilt. Der erste Teil beschäftigt sich mit der Entwicklung eines einfachen Issuer Systems in TYPO3 als Erweiterung und der andere, mit erfolgreichen Open Badges Systemen in der Lehre. Beide Teile fließen schließlich ein in die Entscheidung, ob TYPO3 angemessen ist oder nicht.

Das Ergebnis ist die Empfehlung nicht TYPO3 zu benützen. Stattdessen wird empfohlen, zwischen drei Möglichkeiten zu wählen: die Neuentwicklung mit einem Web Applikations Framework wie zum Beispiel Zend Framework, Laravel, Spring Framework oder Django, den Service von einem Anbieter wie der Open Badges Factory zu nehmen oder das Forken von dem bestehenden Open Source System Badgr.io.

Contents

List of Figures

List of Listings

Acknowledgements

I would like to express my sincerest gratitude to my supervisor, Dr. Martin Ebner, who has supported me throughout my thesis. He has always been pleasant and honest and given feedback was invariably prompt and valuable. I would like to thank him for taking care of his students, because nowadays that seems to be something exceptional.

I would also like to thank Maria, Markus, Christian and all the other colleagues from the Educational Technology of the Graz University of Technology for their patience and helping hands regarding reading and consulting. No one demanded that in the way they did. Then, I would like to thank my family, who have always stood behind me and encouraged me to pursue my own goals. Without their support, I probably would have never walked that path the way I did.

Robert Hafner
Neuhaus, Austria, Oktober 2017

Chapter 1

Introduction

In the past, there were different ways to present achievements to the public. A well-known physical variation is wearing an insignia on a scarf or on clothes. These items are called badges. A more modern variation of this physical medium is a digital badge. Initially, these items were simple images and as a pioneer, the computer game industry has applied these digital images to reward the players for accomplished tasks. Online platforms such as IMDb[1] or Amazon use badges to reward the customers for creating content [Gibson et al., 2015]. A decisive step forward in the evolution of badges was the insertion of metadata into the image to inform about context, meaning and needed activities to receive the badge.

These certifications which are not received from conventional educational facilities have been a challenge and imperative for years due the need of visibility and acceptance. The standard procedure has been the physical attendance at an educational institution, by taking a combination of courses, examinations and in most cases writing a final thesis, so that a graduate certification can be hung on the wall. Caused by the enormous increase of the cost of tertiary education in the United States, a growing number of non-university institutions offer further education for different people, needs and fees [Young, 2012]. Additionally, through the rapid dissemination of the internet, information about people's competencies can be distributed and accessed globally [Goligoski, 2012].

Until the end of the first decade of the 21st century, digital badges had no commonly used standards. This grievance was identified by the Mozilla Foundation and the MacArthur Foundation and so in 2011 Open Badges was presented. A specification for an open and unified standard defines the participants and interactions in an ecosystem, a vocabulary for data objects and also offers software tools or services for generation, distribution and verification [The Mozilla Foundation, 2011].

Digital teaching and learning material for educational purposes are mostly not limited by the quantity and location where they are stored, but restricted by terms of license. One solution for the problem of this learning material are Open Educational Resources (OER). OER includes different components: learning resources, resources to support teachers and resources which can assure the quality of education and their practice [Ebner and Stöckler-Penz, 2011].

The goal of the working group "Open Educational Resources" as part of Forum neue Medien in der Lehre Austria (fnm-austria)[2], is the endorsement and propagation of Open Educational Resources in the area of tertiary education. Following the recommendations of Ebner et al. [2016] national OER labels should be established. Responsible for this remit will be a Central Authority. Active stakeholders shall be supported by certificating the process of creation, storage and provision of OER. Finally, the exchange of free educational material between the universities as well as interested people should be stimulated.

Based on these requirements a technical concept for a software system was written which allows

[1] http://www.imdb.com/, accessed 20.10.2017

[2] http://fnm-austria.at/home.html, accessed 20.07.2017

the transmission of certifications in form of open badges to the applicant [Ebner et al., 2017]. The logical consequence was the development of this web-based application which awards two essential components of the tertiary education: the lecturer and the university, as juristic persons under public law. A precondition to receive a certification for the university lecturer is:

- the notification of three OER they created by them or within a group.
- the completion of a further training at an Austrian university.

An Austrian university has to fulfil three requirements to receive certifications:

- commitment to OER and establishment of a strategy for OER.
- a number of lecturers which publish the metadata of their OER (see list before)
- provision of a repository to store OER for university lecturer.

Consequently, the question arises whether this web-application can be built. The thesis will give an answer how a software application should be planned, designed so that an Open Badges certification is a success. Additionally a prototype is build within a certain Content Management System (CMS) TYPO3 to test, if this system can be used as an Open Badge Issuer service. The Open Badges Ecosystem (OBE) is described and different studies will be evaluated to give an overview about the reasons why application which issue badges may work or may not. The implementation should lower the barrier to get involved in the topic of OER as well as endorse the idea of OER in the community of lecturer at universities in Austria. However, it should be stated that the application is only a prototype and the product of the first phase of the project. Other, further developer can use this work as a base, source of information as well as a pool for not implemented ideas.

The concept of Open Badges as well as a survey of research at this topic is described in chapter 2. A detailed view on the components of the Open Badges Ecosystem and a description of the interactions between them can be seen. Furthermore, the Open Badges vocabulary which defines several data classes is presented. Chapter 3 represents a brief overview of OER. The different standards for metadata of learning resources are presented, so that a selection can be done for the implementation. The technical concept will be described in the following chapter 4. Objectives, business rules and user cases for the software application are shown in this chapter. Details of the implementation and an explanation of the application are presented in chapter 5. Finally, the reflections (see 6) in form of lessons learned for the TYPO3 application as well as a conclusion and further work is provided (see 7).

Chapter 2

Open Badges

At the beginning of this chapter the origin and forms of badges are described. Later on, the Open Badges Ecosystem with all the macroscopic components and their interactions will be presented. Afterwards, a detailed look into the vocabulary will result in a deeper understanding of different data classes including their mutual dependencies. An explanation of concepts such as Extensions and Endorsement is delivered as well, and finally, information about implementation and verification. The chapter is concluded by certain technical concepts which have to used for creation and verification of badges.

2.1 Forms and Types of Badges

The Cambridge Dictionary defines a badge as

> a small piece of metal, plastic, cloth, etc., with words or a picture on it, that you carry with you or that is fastened or sewn to your clothing, often to show your support for a political organization or belief, or to show who you are, your rank, or that you are a member of a group, etc.: [C. U. Press, 2017]

During the last centuries the badges were frequently used[Halavais, 2012]. Its physical form is well known as insignias of Order of Knights[1] (see figure 2.1a), police badge[2] (see figure 2.1b) or an iron-on badge[3] (see 2.1c).

All the badges, which are shown in figure 2.1, are signs of membership and only imply indirect accomplishment or activity. For certification, the physical badges adhered on sashes (girl-boy scout) or pinned on uniforms of medical personnel (e.g. Certified Registered Nurse Practitioner) are examples for membership and in most cases also a sign of abilities.

2.1.1 Digital Badges

The next step in the development of badges was the transforming into a digital entity. More than the physical objects, the digital artefacts can explain the situation, intention and activities. These badges are images with or without additional information, which are inserted into the object [Gibson et al., 2015].

[1] In the year 1915 the Order of George I was founded by the Greek King Constantine https://commons.wikimedia.org/wiki/File$:Order_of_George_I_of_Greece_(rank_of_Commander)_-_IMG_4977.jpg, accessed 20.03.2017

[2] A famous example of badges is an insignia of a Canadian Peace Officer in British Columbia. https://commons.wikimedia.org/wiki/File:BC_Common_provincial_LEO_badge.png, accessed 20.03.2017

[3] The iron-on badge or also referred to as patch is from a Soviet space mission dating from 1969. https://en.wikipedia.org/wiki/Embroidered_patch#/media/File:Soyuz-5-patch.png, accessed 20.03.2017

(a) This is a badge for the Order of George I.

(b) This is a badge for a peace officer in British Columbia.

(c) An iron-on badge for the members of the soviet space mission in the year 1969.

Figure 2.1: Three examples of physical badges to inform about membership to an organisation.

One of the first sector which endorsed and used digital badges was the digital game industry. They have been creating, awarding and displaying badges to players for their achievements. The badges have been rewarded for reaching a new level or scoring a specific number of points. The purpose of this extrinsic rewards is that the participant wants to continue playing. Nonetheless, there are also critical voices regarding this kind of motivation. Gameful Design which is a concept of designing and building a game criticises this model and opposes that creating intrinsic rewards by positive emotion, relationship, meaning, and accomplishment has a more significant impact so that a game is successful [McGonigal, 2011].

The creation of content is the main task of online platforms such as Amazon[4] and IMDb so that they already have issued badges. Nokia Image Spaces was an application which used digital badges to motivate users to upload photos so that sequence of pictures can create a chronological and spatial scene. A study showed that the positive and adverse effects depend on certain points such as good user experience, appreciation of the product, immediate, and explicit feedback [Montola et al., 2009]. Another example for the usage of badges was FourSquare. Additional to other elements such as points, levels, and leaderboards, badges were issued to motivate players to revisit locations so that they can convert digital entities into real products [Xu, 2011].

However, these early badges did not fulfil the requirements for a learning environment. There were no specifications, no openness, and it was not able to transfer the awards to other systems. In the context of learning or engagement, there are badges needed for different categories such as participation, recognition, achievement, contribution and as certification[5].

[4] Early digital badges are issued by Amazon for create or share reviews or verified representatives of listed items. https://www.amazon.com/gp/help/customer/display.html?ie=UTF8&nodeId=14279681, accessed 21.03.2017

[5] The University of British Columbia rewards students with badges and also curates a collection of open educational practices. http://badges.open.ubc.ca/plan/types/, accessed 03.08.2017

2.2 Badges in Education

People get challenged to get perceived by achievements and interests. On the other side, there is a growing field of interests so that these talents can be seen. An organisation is interested in the competencies, interests or activities of a candidate for a new position and they want to get a better understanding of this person. A certification from a university or a high school diploma, maybe cannot give a satisfying impression, during the date the diploma was issued and the present[6]. The activities on social media network sites can be constructed or at the point of view of the employee private. Also, further education and self-assessments are important aspects during some ones working lifetime. Charity and community work can also be interesting for the social environment. Finally, there is a demand for delivering, controlling, transfer, verification and consuming of all this information [The Mozilla Foundation, 2011]. Digital badges in education have emerged, but in contrast to the usage of badges in digital games or online platforms, the demands are different [Gibson et al., 2015]:

- Learners are stimulated by incentives and change their behaviour in a positive direction.
- The progress in learning, as well as content trajectories, are tracked during the whole time of learning lifetime.
- The meaning, qualification, training and success are clear.

Consequently, there are four areas for digital badges in education: motivation, status recognition, evidence of achievement and research implications.

The use of badges can motivate the learner to acquire skills and knowledge as well as provide continuous engagement. A path consisting of badges can be used as an occasion for further education and in this way help users to remain on track. In fact, a new job, a new position within the company or the work as a volunteer in a social service can be a result of the effort to earn badges. Furthermore, also prestige could be a motivation.

Verification of someone's position is as important as motivation. Badges close the gap between formal degrees and (social) skills and allow for greater flexibility. The status within social media can also be important, and so badges can also be used to build and formalise an identity.

Systems which guarantees evidence of achievement relies on trust. The credibility in the eyes of the other persons or organisations which can follow the development of someone is the most critical issue for proof of achievement. Badges have to fulfil four points so that the credibility increases. First, the design process must be transparent and the selected criteria, for earning badges must be thoughtful. Second, these criteria have to be formulated meticulously so that everyone can understand them. Third, in case that digital learning material is produced, a note within the badge as well as a link to a site hosting the material should be created. Finally, the skills which can be achieved should be aligned to a national or international standard [Buckingham, 2014]. Used within a software system, metadata which are parts of the badges are an accurate solution to store the data necessary for evidence. Consequently, at least the following information must be listed:

- Name and further data such as purpose, contact details of the Issuer.
- The Criteria which have to be accomplished and approved
- The activities which have to be done and the produced artefacts as well as the situations which were experienced.
- Quality of activities, results, and performance.

[6] https://learning.linkedin.com/blog/learning-thought-leadership/what_s-next-in-l-d—experts-revea l-predictions-for-2017, accessed 20.07.2017

The impact of badges is part of research and teaching. Gamification is the term which describes a process to improve services through motivation of an user group by invoking of similar experience or elements of a game [Huotari and Hamari, 2012]. Methods respectively items which are used for Gamification are ranking, high score and virtual assets such as badges. A study, how badges can be used as incentives to steer user's behaviour on an online platform, was made in the year 2013. A model was developed and compared to an analyse of the conduct of million users on StackOverflow[7] [Anderson et al., 2013]. The model and the data set confirmed by four key areas :

1. In case that a certain level of activity is awarded by a certain badge, the user will increase the effort to receive the badge.

2. A certain number of badges create a certain number of options to steer (more badges and more options).

3. The closer a participant is to receiving a badge, the higher the level of steering influence.

4. Steering can be done by shifting the mixture of action a user has to take as well as the overall participation in the medium.

The Carnegie Mellon University's Computer Science Student Network (CS2N) uses Open Badges for the secondary educational purpose [Abramovich et al., 2013]. This online learning system has a variety of application such as intelligent tutoring or learning management systems. A study which used CS2N to test the impact of badges in education has shown that the different types of badges (for participation, mastery of skills) affect the results of the students as well as the learners ability. A Midwestern university endorsed Open badges for co-curricula reasons. Crucial for the success was that one professor motivated other colleges as well as he developed an Open Badge platform [Wu et al., 2015, p.50].

The adoption of OER and other techniques such as Open Badges or xMOOC is a challenge for mainstream universities. Nonetheless, contemporary universities (MIT, Open University) realise that such innovations do not threaten their existing business model. Instead, it can be seen as a complement for personalised learning ecosystem and traditional education [Friesen and Wihak, 2013]. An investigation of the Austrian Massive Open Online Courses "iMooX"[8] confirmed an impact on the participants by using badges[Wüster and Ebner, 2016]. The effect depends on factors such as target groups, obligation, and usability. In fact engagement depends on incentives, and early participation increases the probability of successful completion [Kopp and Ebner, 2017].

An economic approach to analysing the design decisions by rewarding badges to users on a site in a game-theoretic framework was made 2016 [Easley and Ghosh, 2016]. The research question was: How can gamification (by usage of badges) be used to incentive attendees as well as to the effort on an online application which needs contribution by users. The conclusion of the authors was:

1. If badges are awarded for a relative part to the top performers, the size of a top-list were the results are presented should be independent of the numbers of contributors.

2. The designer who defines the optimal effort for achieving a badge needs adequate information of the users but can use either an absolute or relative standards mechanism.

3. The analysis concludes that by rising numbers of Recipient, the value of the badges is decreasing.

Consequently, the decision to publish the information depends on the curvature of the value as a function of winner fractions.

[7] Online platform for a software developer to discuss issues in coding. https://stackoverflow.com/ accessed 03.08.2017
[8] https://imoox.at/mooc/, accessed 20.10.2017

From 2012 to 2014 an extended study was done where the researcher investigated 29 winning Badge Content and Program proposals from a contest organised by the Digital Media and Learning (DML) initiative at the John D. and Catherine T. MacArthur Foundation. As a result, some lessons learned with regards were badges work better were presented [Hickey et al., 2015]:

1. Conclusion number one was that some places work better than other. In fact, conventional institutions had problems to setting up a badge ecosystem.

2. Projects with existing content and technology performed better and they did not fail in establishing a badge ecosystem as projects without a established system.

3. The acceptance of employers and learners decreased by the rise of the formal level of the educational institution which offers the ecosystem.

4. The search for external endorsement to increase the relevance of badges did not work well. An increasing acceptance for internal use was ascertained. Further work to present the real added value is necessary.

5. Ecosystems of digital badges which were competing with alternative information sources did not succeed. The redundancy did not add value.

6. The two projects (a youth sports journalism network and an organisation that issues badges for technology and network management skills) were highly successfully by embedding the badges in an existing system where the learning was social and networked.

In the final analysis, the authors conclude that the most obvious points (external endorsements and awarding formal credit) are the most challenging tasks. The efforts of the Issuer must be increasing and supplemented by the personal and professional relationship within the innovators [Hickey et al., 2015].

Open Badges ought to be introduced in stages, at first a pilot phase is recommended. During this period it is crucial that three points must be observed. First, leadership is necessary so that seniors or lead can argument the how the new technology supports the strategic vision. Second, commitment and investment from the institutions who are involved. Finally, engagement with the persons who are concerned (Recipient of badges) [A. Ramsden and F. F. U. A. Ramsden, 2015].

2.3 Open Badges Ecosystem

In the year 2011 the Mozilla Foundation, the Mac Arthur Foundation and the Peer-to-Peer University published a working draft, which describes the new learning environment of the 21st century. In this paper, the personas which are used as case studies, had different ages, reasons for learning, and social background. Informal, peer-based and self-directed learning was seen as a challenge for learners and organisation who looked for volunteers or employees. The requirement to present relevant skills and competencies in the right granularity was made. Finally, a system which connects the different platforms for education such as the Oregon Tech[9] was needed to makes the achievements more viable, portable and impactful [The Mozilla Foundation, 2011]. Additionally, the principals of openness (open source, cooperation of different groups) were also an important aspect for Mozilla. The result of this requirements were Open Badges and can be described in one sentence [Alliance, 2016a]:

> Open Badges communicate skills and achievements by providing visual symbols of accomplishments packed with verifiable data and evidence that can be shared across the web.

[9] 20 courses for online students uses badges for a credential. http://www.oit.edu/, accessed 23.07.2017

Open Badges consist of an ecosystem, specifications, and software applications as well as software services (Application Program Interfaces). The Open Badges Ecosystem (OBE) which was called in the past Open Badges Infrastructure consists of four elements: Issuer, Recipient, Displayer, and Consumer. Therefore, a typical scenario (see figure 2.2 as used by Erik Knutson[10]) contains the following activities: an Issuer assigns a badge to a Recipient for an achievement. The Recipient transfers the badge to a platform. In order to ensure that the badge is valid, a verification is done by contacting the Issuer. A successful verification makes the badge publicly available, and is visible to Consumers. The tasks and interactions between the components of the ecosystem are described in detail in the following sections as well as in the workflow chart in figure 2.3. Backpack services are not part of the OBE. However, the Mozilla Foundation initialised and implemented the services to promote the Open Badges [The Mozilla Foundation, 2011].

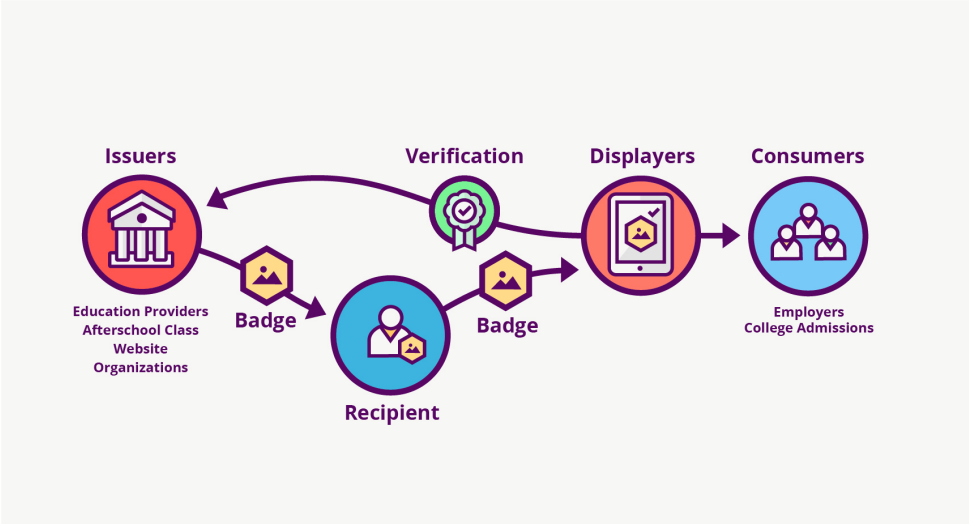


Figure 2.2: The Open Badges Ecosystem describes the four components and the interactions (by Erik Knutson, Concentric Sky, licensed CC-BY).

2.3.1 Issuer

An Issuer is a person or an organisation such as schools, universities, employers, communities, non-profit organisations, government agencies, libraries, museum and further more which certificates achievements, activities or interests. Consequently, everyone can issues badges to pursue different aims. However, a typical scenario for an for an Issuer is [Alliance, 2016b]:

1. Defining general goals to accomplish and design a system in which badges can be used.
2. Deciding the specific competencies which shall be awarded by badges.
3. Designing the badge and defining metadata for the data classes.
4. Completing the metadata and information by inserting into Issuer profile and metaclass for Open Badges (BadgeClass) and publishing these objects.
5. Creating an individual badge and starting delivering.

[10] https://openbadges.org/developers/, accessed 20.10.2017

The usage of badges for awarding effort leads to a one question: Are badges the method of choice? And the answer depends on the organisation and Recipient of the badge. There are cases Open Badges work well and such they do not. Detailed research is necessary before a system will be installed.

2.3.2 Recipient

The first step for an Recipient alias Earner is to decide which skills or competencies correspond at best to personal education, interests or social participation. Afterwards, the course which provides learning or engagement and awards with badges has to be selected, attended and finished. Finally, a decision has been taken whether to publish the badge after receiving or not. Publishing can be done in a variety of ways:

- publishing on the website of the Issuer (in case this service is provided).
- uploading the badges to a Backpack service where storing, publishing or distribution can be done.
- uploading the badges to a Displayer (for example a social media site).

2.3.3 Displayer

A Displayer is the provider of online platforms to create publicity for the Recipient. Famous platforms for displaying are Facebook, LinkedIn, Twitter, Wordpress or about.me. It depends on the settings of the Recipient which parts of the metadata within the badge are displayed. After the upload, a validation and later on a verification of the badge in cooperation with the Issuer is done. This procedure guarantees the correctness of the badge and thus the achievement of the Recipient. This proof is dispensable in case the badge is uploaded to a Backpack service. Some of the providers (e.g. Credly) are Displayer as well as Issuer, and therefore a strict separation, between this roles is not possible.

2.3.4 Consumer

The information about interests, achievements, etc. is intended to receive an impression of a person. This convenient approach is used to find people who best fit a particular role. In the end, there could be a job, an internship or social participation for the Recipient.[Grant, 2014]. It is recommended to be aware of the consumer and engage the audience early so that in same cases they are involved in the process of defining [Presant, 2016].

2.3.5 Backpack

The Backpack services are an optional component in the OBE and provide reliable means for collecting, administrating and sharing of Open Badges between different platforms[11]. The upload of the badges can be done by the Issuer themselve as well as by the Recipient. This means that most of the Issuer implemented a Backpack API in their software application so that after the earning an upload to the Backpack provider can be done easily.

Currently, the Mozilla Backpack is the main vendor and already stores over 1 million badges[12]. Originally, the intention for this system was to implement a reference system. Backpack has a feature which is called Collections that enables a selective arrangement of the badges. These sets can be shared with other providers such as Credly.

[11] A list of various providers for Backpack services can be found at https://openbadges.org/about/participating-services/#backpacks. In details the providers are different. However, they share the openness of the applications. (accessed 02.01.2017)

[12] https://backpack.openbadges.org/backpack/welcome accessed 22.06.2017

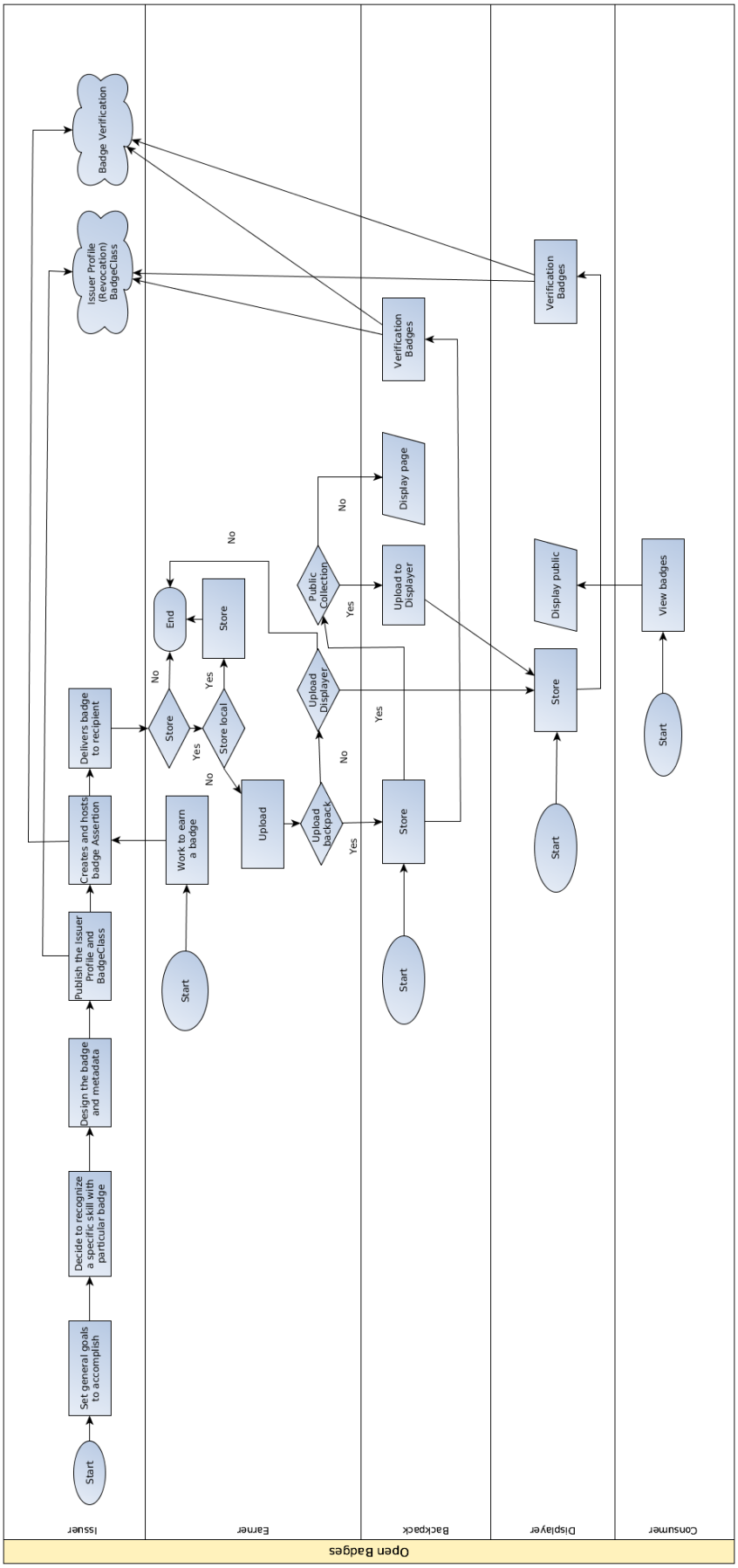


Figure 2.3: Flow chart of the components and tasks of the Open Badge Ecosystem and the backpack service provider. The activities started on the upper left side and ends and the lower left corner (Robert Hafner, CC-BY).

2.4 Open Badges Specification

The Open Badges v2.0 is a collection of specifications that describes an approach to insert (meta)data regarding achievements into an image so that people can be awarded with Open Badges and these accomplishments are verified by software applications.

The Open Badges Vocabulary consist of objects which are needed to implement the Open Badge Ecosystem. Most of these objects are JavaScript Object Notation for Linked Data (JSON-LD) [Sporny et al., 2017]and for further reading see section 2.6.1. Linked Data is an approach to publish and connect structured data on the Internet so that machines can read and track the structures. The method is required to realise the Semantic Web [Berners-Lee, 2006]. This principle is used by linking the core classes of the Open Badges Vocabulary: Assertion, BadgeClass and Issuer. A first impression is gained by the figure 2.4 and presents the connections between the classes of Open Badges v.1.1: The Assertion is a JSON-LD and represents the individual achievement. It is linked to the BadgeClass. The metadata class describes the general requirements that have to be fulfilled and is defined by the Issuer. This object is connected to the profile class Issuer which is also called Profile where the information about the organisation is stored. Additionally, there are also mandatory and optional links to other classes. The internationalist form (International Resource Identifier (IRI) [Dürst and Suignard, 2005]) of the Uniform Resource Identifier (URI) [Daigle et al., 2002] is used, so that the objects can be linked and identified on a server location. Internationalisation is facilitated by the JSON-LD and so Issuers can express badges in language another than English or in more than one languages. The openness of Open Badges allows also the insertion of self-defined properties into the classes.

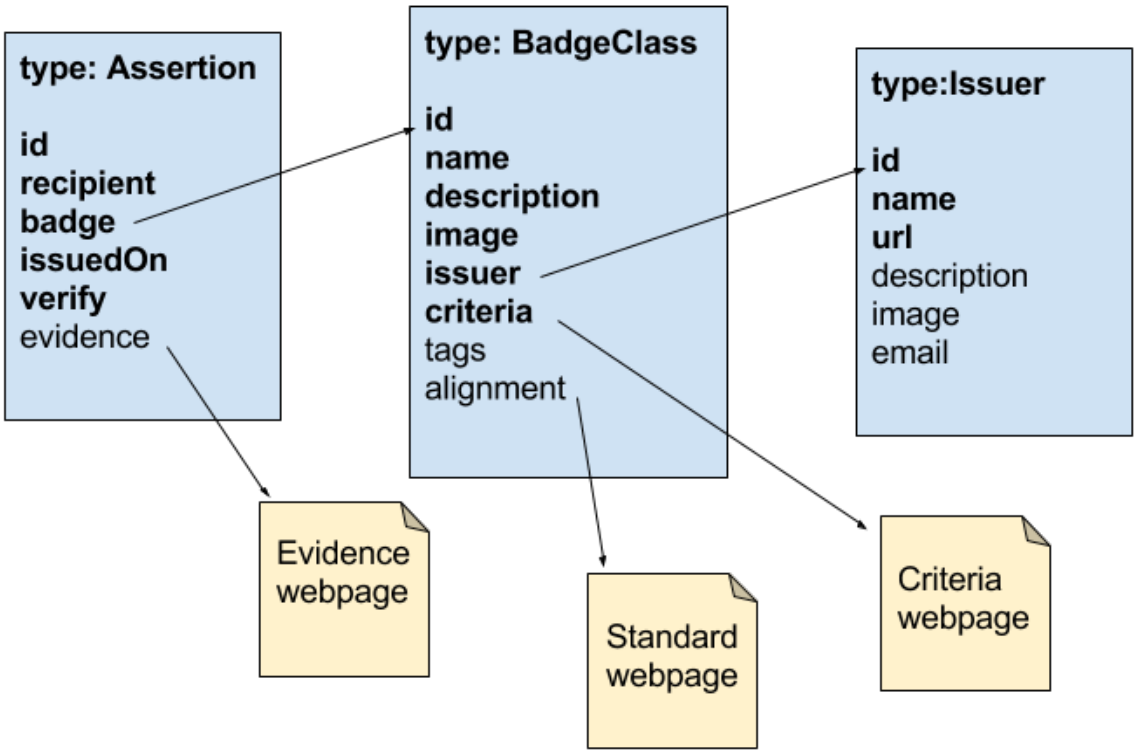


Figure 2.4: The blue objects are the core classes of the Open Badges Vocabulary. Type Assertion is the individual badge, BadgeClass includes the general information, and the Issuer is the organisation which provides the certification. The yellow items are JSON-LD as well as websites.

In 2013 the Open Badges v.1.0 was released [Thompson, 2013]. Two years later, 2015 the version 1.1[13] was published and brought some new features (Extensions, JSON-LD, et Altera). Since the first January 2017 the IMS Global Learning Consortium[14] takes care of the endorsement, maintenance, and further development of Open Badges. The day before, December 31st a new version 2.0[15] was published (now with Endorsements). It is the de facto the standard for digital badges [Ballesté, 2013]. All the specifications can be found on the website of IMS Global and are the primary source for the following section [Consortium, 2017b].

2.4.1 Open Badges Vocabulary

The vocabulary of Open Badges consists of several different data classes. Some of them are mandatory some are optional. Within the classes, there are metadata which are structured as a pair of properties (attributes) and values which indicates whether the elements are mandatory or optional. If the classes are JSON-LD, all the objects have at least the property *@context*. As a result of this, the expected type is an IRI and linked to the Open Badges v2.0 schema[16]. Properties of a data classes which can be found in the schema are considered to be part of the specification.

2.4.1.1 Assertion

An Assertion includes all information about a distinct award and connects a certain Recipient to a particular badge. The standard approach is to inserting this object into an image (for further detail see section 2.5.1). The following key-value pairs are mandatory for a badge:

- **id**. The expected type is an IRI and in the case of hosted verification (see section 2.5.2.2) a URI is used, otherwise, for a signed Assertion (see section 2.5.2.2), it should be a Universally Unique Identifier (UUID) in the format urn:uuid namespace.
- **type** The JSON-LD representation of an Assertion. In most cases a string "Assertion" or an array of strings and URIs or short IRI is the value for the attribute.
- **recipient:** The type is an IdentiyObject (see 2.4.1.4) which contains information about the Recipient of the badge.
- **badge:** The type is an IRI or document and connects the Assertion to the specific linked BadgeClass.
- **verification:** The type is a VerificationObject (see 2.4.1.5). This object contains information about the method to verify the Assertion.
- **issuedOn:** The type is DateTime[17] and date the badge when the Assertion is awarded - for example: “2016-12-31T23:59:59Z“.

The next properties are optional:

- **image:** The type is an Image (see 2.4.1.7). An IRI or document where an image is stored as a representation of achievement.
- **evidence:** The type is Evidence (see 2.4.1.6). An IRI or document for additional information on the positive achievement of the Recipient.

[13] https://www.imsglobal.org/sites/default/files/Badges/OBv2p0/history/1.1-specification.html, accessed 24.07.2017
[14] http://www.imsglobal.org/, accessed 24.03.2017
[15] https://www.imsglobal.org/sites/default/files/Badges/OBv2p0/history/2.0.html, accessed 24.07.2017
[16] The replay of a GET request contains a JSON object which lists all the properties whereby an attribute of a class can be considered as a member of a badge object. https://openbadgespec.org/v2/context.json, accessed 03.08.2017
[17] https://www.w3.org/TR/NOTE-datetime

- **narrative:** A text or markdown text. Only applied at this location if multiple arrays of Evidence are used.
- **expires:** This Date (data object DateTime) stores the information on which the badge expires.
- **revoked:** Boolean value (true, false), whereby false is the default. In case badge is revoked, the value is true.
- **revokationReason:** Additional text for reason of revocation.

An example of a signed Assertion can be seen in listing 2.1. This is an Assertion before the data class is decrypted into a JWS. This Assertion is neither valid nor verified.

```
{
"@context": "https://w3id.org/openbadges/v2",
"type": "Assertion",
"id": "urn:uuid:e79a6c18-787e-4868-8e65-e6a4530fb418",
"recipient": {
"type": "email",
"hashed": true,
"salt": "example",
"identity": "
    sha256$c7ef86405ba71b85acd8e2e95166c4b111448089f2e1599f42fe1bba46e865c5
    "
},
"image": "https://oerbadges.org/lecture-strike-badge.png",
"evidence": "https://oerbadges.org/EbnerMartinTUGraz/lecture-strike-
    badge_1.html",
"issuedOn": "2017-03-31T23:59:59Z",
"badge": "https://oerbadges.org/lecture-badge.json",
"verification": {
"type": "signed",
"url": "https://oerbadges.org/public-key.pem"
}
}
```

Listing 2.1: An example of a signed Assertion

2.4.1.2 BadgeClass

The BadgeClass is a core class object and in general, describes the effort to achieve a badge. Instead of a unique Assertion, which explains the personal effort of a Recipient. For example, a person can get a badge for absolving a course. The details about the course and what the course is about are written in the BadgeClass. It is mostly linked by many Assertions and links to the profile of the Issuer. The mandatory *@id* is linked to a unique URI to the JSON-LD BadgeClass. Most of the Issuer platforms handle only HTTP-based Identifiers. Signed Assertions should be published in a schema such as an urn:uuid instead of using HTTP IRIs for hosted Assertions. The property *type* is either a simple string or an array of BadgeClasses. Additional mandatory components are:

- **name:** The name of the accomplishment.
- **description:** A short textual explanation of the accomplishment.

- **image:** The expected type is an Image (see 2.4.1.7). An IRI (Data URI or URI) of an image representing the accomplishment. The same link as used in the Assertion.
- **criteria:** The expected type is a Criteria (see 2.4.1.8) which is connected to a URI or HTML-Document. Is used for a further description of the effort to achieve a badge.
- **issuer:** The expected type for the URI or Document is the Profile (see 2.4.1.3) class of the Issuer.

For this object, *alignment* and *tags* are optional. For the first property, AlignmentObject (see 2.4.1.9) is the conventional type. As a single element or within an array, an alignment contains information about educational standards. The second property is either a single term or multiple keywords, to describe the type of achievement. An example for a BadgeClass can be seen in listing 2.2.

```
{
"@context": "https://w3id.org/openbadges/v2",
"type": "BadgeClass",
"id": "https://oerbadges.org/lecture-strike-badge.json",
"name": "Lecture produces three OER Badge",
"description": "For publishing three Open Educational Resources.",
"image": "https://oerbadges.org/lecture-strike-badge.png",
"criteria": "https://oerbadges.org/lecture-strike-badge.html",
"tags": ["OER", "Certification"],
"issuer": "https://oerbadges.org/organization.json",
}
```

Listing 2.2: An example of a BadgeClass which describes the efforts (publish three OER) for getting an Open Badge.

2.4.1.3 Profile

This core class can be used for two cases. First, the Issuer is the Recipient of the badge. Second, the person (or organisation) is an Issuer of a badge. In the first case the object has two mandatory properties: *type* and *@id*. By only using these attributes anonymity can be guaranteed. The expected type of the property *type*is string with the value Issuer or Profile. *@id* has a unique URI and links to the Profile JSON-file. In case, this object is used for issuing; the following properties are mandatory:

- **name:** This string is the name of the organisation.
- **url:** The homepage or social media presence of the Issuer to present content and organisation.
- **email:** An email address to contact the Issuer.

Further information can be used to provide a complete picture, and therefore the optional properties can be utilized:

- **telephone:** A phone number of a member of the Issuer.
- **description:** A short textual description of the Issuer.
- **image:** The expected type is Data URI or URL and links to an image which represents the organisation.

- **verification:** Information about the method to verify the Assertions. This URI is linked to a VerificationObject (see 2.4.1.5).

There are two methods to verify an Assertion: signed badge (see 2.5.2.2) and hosted badge (see 2.5.2.2). If the first method is selected the following properties should be used:

- **publicKey:** The expected type of this property is the id of the CryptographicKey (see 2.4.1.11).
- **revocationList:** The IRI to the object RevocationList (see 2.4.1.10) which contains a list of the revoked Assertions.

2.4.1.4 IdentityObject

This data class includes selected information about the earner of a badge. In contrast to the other entities, an IdentityObject is not a JSON-LD (no need for further linking). The following properties are mandatory:

- **identity:** The expected type is a simple text or an identityHash. It is recommended to use a Hash-function (see 2.6.2) so that personal information can not be exposure.
- **type:** The expected type is an IRI. The value defines the method to identify the Recipient. In general, the value *email:* is used by the Issuer. Since v2.0 further properties such as *url* and *phone* are available.
- **hashed:** The boolean value indicates if the identity is hashed or not.

An optional property is *salt*. This variable is an additional parameter for the Hash-function to improve the safety of the method. In listing 2.3 a not hashed IdentityObject shows the email address of the Recipient in plain language. A hashed email address with an additional salt value prevents reading the email address in plaintext (see listing 2.4)

```
{
 "recipient": {
  "type": "email",
  "identity": "robert@oerbadges.org",
  "hashed": false
 }
```

Listing 2.3: An example of an IdentityObject without the usage of hashing. Personal information such as email-address is public.

```
{
"recipient": {
"type": "email",
"hashed": true,
"salt": "example",
"identity": "
    sha256$30593AC17E84D0CA72247A9FC12D60FDA71EF34B240E4F47E2B383C09DBDEF28
    ",
}
```

Listing 2.4: An example of an IdentityObject with hashed identity. Personal information such as email address is protected.

2.4.1.5 VerificationObject

This data object is used to designate the method of the verification as well as additional information for the procedure. A VerificationObject (VO) is a property of an Assertion as well as of the Profile. Consequently, there could be differences in the form cause the VO can exists in different manifestations. Within an Assertion the mandatory property of this object is *type* and the value here can be *hosted* or *HostedBadge*, *signed* or *SignedBadge*. If it is used within a Profile (see listing 2.7.) the VerificationObject includes the following optional elements:

- **verificationProperty:** The expected type is *@id* and only the value *id* is supported.
- **startsWith:** A fragment of the URI which the verification property must start with. All the id within the Assertions must contain this value as a fragment of the id. A single value, as well as multiple entries, are allowed.
- **allowedOrigins:** Part of the URI of an Assertion. In case the id of the Assertions origins with this string, it can be considered as valid.

Future classes of this objects may be used with other methods such as blockchain-based procedures. In contrast to the hosted badge (see listing 2.5, a simple JSON) a signed badge (see listing 2.6) has an additional property **creator** which links to the key used to verify the signature if the Assertion. By the way, the object is also delivered in the format JSON Web Signatures (JWSs) (see 2.5.2.2).

```
"verification": {
   "type": "HostedBadge"
}
```

Listing 2.5: An example of a hosted VerificationObject.

```
"verification": {
"type": "SignedBadge"
"creator": "https://oerbadges.org/publicKey.json"
}
```

Listing 2.6: An example of a signed VerificationObject.

```
 "verification": {
"allowedOrigins": ["oerbadges.org", "two.oerbadges.org"],
"verificationProperty": "id"
}
```

Listing 2.7: An example of an VerificationObject within a Profile. The property allowedOrigins designates two hosts.

2.4.1.6 Evidence

An Evidence is a class within an Assertion consisting of metadata that contains additional information to prove an achievement. There are situations in which the descriptive properties of an Assertion are not sufficient hence Evidence can be used to provide a big picture. Instances within an Assertion only conform to one badge, although a single entry can also describe a set of items. An Evidence can be created in several ways which are caused by complete optionality of the properties.

- **type:** The default format is JSON-LD.
- **id:** An IRI is expected, and the URI to an HTML site refers to the individual achievement of the Assertion.
- **narrative:** Text or Markdown Text which describes evidence and activities to achieve the badge. The property narrative is also an optional part of an Assertion.
- **name:** The title of the evidence.
- **description:** A simple textual description of the evidence. If this property and narrative are used, the narrative provides more in detail.
- **genre:** Description of the type of evidence (e.g. Robotic, Life Long Learning)
- **audience:** This property designates the indented audience for the evidence.

An Evidence, as a property of an Assertion, is the most straightforward manner to use this class. A more detailed object of the class Evidence is the provided in listing 2.8 which contains all the available properties.

```
{
"@context": "https://w3id.org/openbadges/v2",
"id": "https://https://oerbadges.org/university/tu-graz-numbers-badge.
   json",
 "evidence": {
"id": "https://kfuni-graz.at/TUGraz-education-strategy.html",
"name": "TUG further Education and Strategy OER",
"description": "A webpage with the numbers  OER lecturer at TU Graz.",
"narrative": "The Technical University of Graz has a significant number
    of OER lecturer. Further information about people and content can
   be found here",
"genre": "Academic Education"
"audience": "Lecturer"
}
```

Listing 2.8: An example of an Evidence class within an Assertion. Detailed description of achievement related to Recipient and Assertion.

2.4.1.7 Image

A reference to an image is part of all three core classes. It can be used neither as simple property or ImageObject[18]. *@id* is mandatory where a URI or Data URI links to an image. Optional properties for this object are:

- **caption:** The caption of an image.
- **author:** Author of image (e.g. *@id* of Profile, URI).

2.4.1.8 Criteria

According to the open nature of Open Badges Vocabulary, Criteria is a variable way to add descriptive metadata into a BadgeClass. A potential Recipient can read the requirements to achieve a badge within the BadgeClass, instead to open a linked HTTP-URI. The entire class is optional, the type is a JSON-LD and it is composed of two properties maximal:

- **id:** The URI links to a human readable website which contains the requirement(s) to earn a badge.
- **narrative:** Text or Markdown text which describes the requirement(s) contributed to achievement.

2.4.1.9 AlignmentObject

This class refers to the AlignmentObject[19] which in turn is based on the Learning Resource Metadata Initiative[20] As part of the BadgeClass, it is used to connect the resource to an educational framework. The following properties are mandatory:

- **targetName:** The name of the alignment.

[18] http://schema.org/ImageObject, accessed 10 June 2017
[19] http://schema.org/AlignmentObject, accessed 10 June 2017
[20] http://lrmi.dublincore.net/the-specification/alignment-object/, accessed 24.07.2017

- **targetUrl:** The URL of a web page with a description of the objective(s) for the alignment.

Optional properties are:

- **targetDescription:** A textual, short description of the alignment objective(s).
- **targetFramework:** The name of the educational framework.
- **targetCode:** A unique local string to find alignment objectives within the framework.

The example in figure 2.9 is taken from the official IMS Global web page[21] and describes two AlignmentObjects: The first object is used to describe the English language standard for college students and the other one to list the problem-solving skills for the 21st Century.

```
"alignment": [
{ "targetName": "CCSS.ELA-Literacy.RST.11-12.3",
  "targetUrl": "http://www.corestandards.org/ELA-Literacy/RST/11-12/3",
  "targetDescription": "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
     .",
  "targetCode": "CCSS.ELA-Literacy.RST.11-12.3"
},
{ "targetName": "Problem-Solving",
  "targetUrl": "https://learning.mozilla.org/en-US/web-literacy/skills#
     problem-solving",
  "targetDescription": "Using research, analysis, rapid prototyping,
     and feedback to formulate a problem and develop, test, and refine
      the solution/plan.",
  "targetFramework": "Mozilla 21st Century Skills"
}
]
```

Listing 2.9: An example of an AlignmentObject from the IMSGlobal website.

2.4.1.10 RevocationList

The purpose of this data class is, to announce badges which are revoked by the Issuer. The RevocationList is only used, if signed Assertions are issued. Every Assertion id which is found in this file identifies a revoked badge. Consequently, every verification of a signed Assertions includes a validation by requesting this file. The JSON-LD contains three properties, whereby only revokedAssertions is mandatory:

- **id:** The id of the revocationList.
- **issuer:** The id (IRI) of the Issuer.
- **revokedAsssertions:** An Array of revoked badges.

[21] https://www.imsglobal.org/sites/default/files/Badges/OBv2p0/examples/index.html#BadgeClass, accessed 10 June 2017

The revokedAssertions is also a JSON-LD and contains the following elements:

- **id:** The IRI of the revoked Assertion.
- **uid:** Only used in badges for v1.1. It is mandatory. Either *id* or *uid* is used.
- **revoked:** Boolean value, in which default is true.
- **revokationReason:** The optional property designates the reason for revocation.

In the listing 2.10 a RevocationList can be seen. As recommended a uid namespace is used to identify the example.

```
{
{
  "@context": "https://w3id.org/openbadges/v2",
  "id": "https://oerbadges.org/revocationList.json",
  "type": "RevocationList",
  "issuer": "https://oerbadges.org/issuer",
  "revokedAssertions": [
  "urn:uuid:3c344c46-b22f-4f06-erb5-68rew33fsb6fe",
  {
    "id": "urn:uuid:e79a6c18-787e-4868-8e65-e6a4530fb418",
    "revocationReason": "University stopped further education program
       for Open Educational Resources"
  }
  ]
}
```

Listing 2.10: An example of a revoked Assertion.

2.4.1.11 CryptographicKey

Based on the Key class published by W3C[22] this object contains data which is used to verify the signature. The type is a JSON-LD containing the value CryptographicKey. Additional, the following properties are used:

- **id:** A IRI to locate the key.
- **owner:** The IRI of the Profile which is the owner of the key. A two-way connection is established from this link and the property publicKey of the Profile.
- **publicKeyPem:** This is the public key which is needed to verify the signature (for technical detail see 2.6.3).

It is recommended to use the HTTPS protocol for the Profile and the CryptographicKey and an example can be seen here in listing 2.11.

[22] https://web-payments.org/vocabs/security, accessed 12 June 2017

```
{
  "@context": "https://w3id.org/openbadges/v2",
  "type": "CryptographicKey",
  "id": "https://oerbadges.org/publicKey.json",
  "owner": "https://oerbadges.org/organization.json",
  "publicKeyPem": "-----BEGIN PUBLIC KEY-----\nBG0BA...\n-----END
     PUBLIC KEY-----\n"
}
```

Listing 2.11: An example of a CryptographicKey contains all the needed information to verify a signed Assertion.

2.4.2 Extension

Additional properties for data classes in the vocabulary are regulated by the JSON specifications (see 2.6.1), and not by properties and values. This means that the standard allows the Issuer to add metadata if they want. However, an approach for a well-formatted object with additional metadata to inform other participants of the Open Badge Ecosystem, about the context and also to enable a reusability was introduced by Open Badges Specifications v.1.1: Extensions. Each core class can be enhanced by Extensions. Every data structure of an Extension should be described by a new hosted JSON-LD (e.g. context file) which further can be used to define a JSON-schema[23]. All implementations of this particular Extension should pass a validation by the context file. This object is linked to the value of the *@context* property and stored separately. The mandatory properties for an Extensions are:

- **@context:** URL for JSON-LD context file.
- **type:** Array of IRIs to describe the type of data. It must include the term "extension".
- **anyProperties:** The properties which are defined in the schema with a valid value.

A validation of an Extension, as well as the other Open Badges objects, is intended in two ways: TypeValidation and FrameValidation. The first approach matches the schema located at the link which is designated in the property *validationSchema* and designated by the type *TypeValidation* against the object designated in the *validatesType*. Currently, only TypeValidation[24] is implemented and included into the Open Badges Vocabulary.

The Issuer has to submit a pull request of the schema to the GitHub repository[25], to inform the community about a new extension. The listing 2.12 is an example of an Extension and created by Nate Otto to license a BadgeClass and listing 2.13 is the context file which can be used to validate an extension by mapping the syntactic requirements.

[23] http://json-schema.org/ accessed 12 June 2017

[24] The context file of the Open Badges Specification can be used for a syntactical verification for certain properties of the data classes. https://www.imsglobal.org/sites/default/files/Badges/OBv2p0/v1/context.json, accessed 10.08.2017

[25] https://github.com/openbadges/openbadges-specification/blob/master/extensions/index.md, accessed 12 June 2017

```
{
  "@context": "https://w3id.org/openbadges/v2",
  "type": "BadgeClass",
  "name": "Licensed Badge",
  "...": "...",
  "schema:license": {
    "@context": "https://openbadgespec.org/extensions/licenseExtension/
       context.json",
    "type": ["Extension", "extensions:LicenseExtension", "cc:License"],
    "id": "CC-BY",
    "name": "Creative Commons Attribution",
    "legalCode": "http://creativecommons.org/licenses/by/4.0/legalcode"
  }
}
```

Listing 2.12: Example of Nate Class to add the license of a BadgeClass.

```
{
{
  "@context": {
    "obi": "https://w3id.org/openbadges#",
    "cc": "http://creativecommons.org/ns#",
    "legalCode": "cc:legalcode",
    "License": "cc:License",
    "CC-0": "https://creativecommons.org/publicdomain/zero/1.0/",
    "CC-BY": "http://creativecommons.org/licenses/by/4.0/",
    "CC-BY-SA": "http://creativecommons.org/licenses/by-sa/4.0/",
    "CC-BY-NC": "http://creativecommons.org/licenses/by-nc/4.0/",
    "CC-BY-NC-SA": "http://creativecommons.org/licenses/by-nc-sa/4.0/",
    "CC-BY-ND": "http://creativecommons.org/licenses/by-nd/4.0/",
    "CC-BY-NC-ND": "https://creativecommons.org/licenses/by-nc-nd/4.0/"
  },
  "obi:validation": [{
    "@type": "obi:FrameValidation",
    "obi:validationFrame": {"@context": "https://w3id.org/openbadges/v1
       ", "@type": "extensions:LicenseExtension"},
    "obi:validationSchema": "https://openbadgespec.org/extensions/
       licenseExtension/schema.json"
  }]
```

Listing 2.13: Context file of the licenseExtension (Nate Otto).

2.4.3 Endorsement

On the Consumer side, the confusion increases by the increasing number of badges and Issuer. Does it manifest itself in questions such as: Is this badge interesting to me? Are the Profile data verified or not? Due to this uncertainty for Consumer and Displayer, a method which re-establish confidence is needed. For this purpose, the data class Endorsement was provided by v1.1. This object similar to an Assertion contains a property "claim" to create a specific claim to a certain entity of the three core classes, Assertion, BadgeClass or Profile. The following properties are mandatory:

- **id:** A unique IRI for the Endorsement entity.
- **type:** JSON-LD is the expected type and the value is Endorsement[26].
- **claim:** An Object which is identified by @id so that an endorser can make a claim.
- **issuer:** The Profile of the Endorsement Issuer.
- **issuedOn:** The timestamp (DateTime) of publishing the endorsement.
- **verification:** The VerificationObject contains the information how to verify the Endorsement (signed or hosted).

An optional property is endorsementComment which can be used to add an additional comment into the Endorsement. A simple example of this object which demonstrates the structure and purpose can be seen in listing 2.14.

```
{
"@context": "https://w3id.org/openbadges/v2",
"type": "Endorsement",
"id": "https://anotherorganization.org/endorsement-12.json",
"issuer": "https://anotherorganization.org/issuer.json",
"claim": {
"id": "urn:uuid:e79a6c18-787e-4868-8e65-e6a4530fb418",
"endorsementComment": "Great OER material for higher education. Keep
   calm and carry on."
},
"verification": {
"type": "hosted"
}
}
```

Listing 2.14: An example of an Endorsement to claim a Recipient of an Open Badge for creating three OER.

[26] This type is a subclass of Verifiable Claims Task Force Credential. https://www.w3.org/2017/vc/ , accessed 10.08.2017

2.5 Implementation

There are three approaches to issue badges. First, is the usage of an existing online services[27] such as BadgeList[28] or Badgr[29]. Second, install and configure a software such as BadgeKit[30], BadgeOS[31] or Moodle[32] on a server.

Third, developing an application according to the Open Badges Specifications (see section 2.4). The minimum requirements for such a self-implemented system are:

1. A webserver to publish objects. These entities are special files (see section 2.6.1) and should be stored in a stable manner.

2. A identity management for the Recipient so that they can be awarded with badges and if anonymity is needed, this service can be done. For this purpose mostly email-addresses are used. However, since v.2.0 (see section 2.4.1.4) new types are available.

3. optional: Public/Privat-Key-Application which is only needed in case of signed badges. For further information see section 2.6.3.

These two, respectively three points are important parts of the process to verify a badge (see 2.5.2)

The delivery of the badges to the Recipient takes place in two ways. First, the signed or hosted Assertion is sent by the Recipient, from the website of the Issuer to a Backpack service. For example, the Issuer API from the Mozilla Backpack Service[33] can be used to upload the badge. Second, the Assertion is baked (see paragraph 2.5.1) into an image, and the Issuer enables the download of the file.

2.5.1 Badge Baking

An Open Badge is an image and embedded metadata [Consortium, 2017a]. This two components image and Assertion are combined, and later on, the metadata can be extracted automatically, and a consumer can read and understand the context. By specification only two formats can be used: Portable Network Graphics (PNG) [Boutell, 1997] and Scalable Vector Graphics (SVG)[W3C, 2008]. The image should not be greater than 90 x 90 pixels and the file size needs to be less than 265 kB.

PNG It is mandatory to insert an iTXt chunk[34] with the keyword *openbadges* into the file. Only one chunk with the keyword can be inserted. Just in case the keyword *openbadges* is part of an image, it must be overwritten. For hosted badges, the JSON-LD for signed badges, JWS can be used as text, without compression. For extraction, the file is parsed until the keyword is reached.

SVG Unlike the PNG, the processing of an SVG is more complicated. First, a xmlns:openbadges attribute with the value "http://openbadges.org" has to be inserted into the <svg> tag. This element is followed by the tag <openbadges: assertion> and the verifying attribute. If SignedBadges is used, the

[27] A complete list of the service provider can be seen online at https://openbadges.org/about/participating-services/#issuing-services, accessed 23.06.2017

[28] https://www.badgelist.com/, accessed 23.06.2017

[29] https://info.badgr.io/, accessed 23.06.2017

[30] A BadgeKit instance can be built from a GitHub repository and hosted on an own server. There is no active maintenance. https://github.com/mozilla/openbadges-badgekit/wiki/BadgeKit-Self-Hosting-Guide, accessed 23.06.2017

[31] http://badgeos.org/, accessed 23.06.2017

[32] The learning platform also offering digital badges so that providers can award students. https://docs.moodle.org/31/en/Badges, accessed 23.06.2017

[33] Github repository explains in detail the usage of the Issuer API to push the Assertion to Mozilla Backpack. https://github.com/mozilla/openbadges-backpack/wiki/Using-the-Issuer-API, accessed 10.08.2017

[34] https://www.w3.org/TR/PNG/#11iTXt, accessed 23.06.2017

value for verifying is the entire JWS file, and the tag should be self-closed otherwise, only the URL of the hosted Assertion is stored as value. Additionally, the JSON-LD is wrapped by <![CDATA[...]]>. Same as the PNG, only one keyword has to be applied, and if it already exists it should be overwritten. For extraction, the file is parsed until the keyword is reached and depending on the verification method selected values will be taken.

2.5.2 Verification and Validation

As discussed in the section 2.3, a verification is an integral part of the OBE. A certification system has to guarantee the truthfulness of the data about an achievement (see 2.2). Hence, a system which fulfils this requirement has to be transparent and it is easy to use and to implement methods [Education, 2013]. This process consists of two steps, first validation and second verification. Depending in detail on the selected method, SignedBadge (2.5.2.2) or HostedBadge (2.5.2.2) Verification. The review of an Assertion is not only the inspection of a single object [Smith and et altera, 2014]. It is also the validation of all Open Badges objects which are linked to the certificate [Consortium, 2017b].

2.5.2.1 Validation

This procedure is done by the Backpack service as well as the Displayer and includes several tests to ensure the validation. The following steps are mandatory:

- All the mandatory objects are correctly linked and available (optional is a HTTP Replay *200 OK*).
- Each optional class is linked and available.
- The Open Badge Objects have the valid form (e.g. JSON-LD).
- The Open Badges Objects encloses all expected properties.
- The Open Badges Objects are fulfilled with the expected data type which are defined in the Open Badges Vocabulary.

Online services[35] as well as applications[36] perform a validation. However, it is not granted for Open Badges v2.0.

2.5.2.2 Verification

A Verification not only includes valid data, but it also grants that the information can be trusted. Therefore, mandatory checks ensure that:

1. The validation of all Open Badges objects is successful.
2. All the Open Badges Objects are created by an Issuer which is verified by Profile, according to the selected properties.
3. The Recipient of an Assertion is valid by the chosen type (e.g. email) in the IdentityObject.
4. The Issuer of the Assertion is authorised to issue the badge of the declared BadgeClass.
5. The Assertion is not expired.
6. The Assertion is not revoked.

[35] The http://validator.openbadges.org/, accessed 05.07.2017
[36] https://github.com/mozilla/openbadges-validator, accessed 05.07.2017

Hosted Verification A hosted Assertion (see listing 2.1), is a valid JSON-LD with content type application/ld+json or application/json and is stored on a stable web-server which can be identified and located by a URI. A redirection to another server is possible and accepted. The scope of the Assertion is defined within the VerifcationObject (see 2.4.1.5). Several steps are necessary to verify the hosted Assertion:

1. Start calling the URL which is stored in Assertion and test the availability. The GET request must retrieve a 200 status [Belshe et al., 2015]. If HostedBadge is selected use the retrieved data for further verification.
2. Testing whether Assertion is expired.
3. Validation of specified structure (property types confirm to specification).
4. Ensure the IdentityObject is correct (mostly by check the email address).

In case all checks are passed, the Assertion can be trusted. It is recommended to return an HTTP Status 410 for a revoked hosted Assertion. Additionally, the body of the response contains the property "revoked" and the value true. In the listing 2.15 a compact JSON-LD meets the specifications.

```
{
"@context": "https://w3id.org/openbadges/v2",
"id": "https://oerbadges.org/university-strategy-badge.json",
"revoked": true,
"revocationReason": "University revoked their stratey to promote Opern
   Educational Resources."
}
```

Listing 2.15: An example of a revoked Assertion. The properties context, id, revoked are mandatory. The attribute revocationReason is optional.

Signed Verification It is highly recommended to publish a signed Assertion as JSON Web Signature. In this section, it is only necessary to know that such a file is separated by two dots into three parts: header, payload and signature (further reading see 2.6.1). The payload contains the Assertion. Furthermore, several steps are necessary to verify the signed Assertion:

1. Separate the JWS object into three parts and decode it into a string by using Base64.
2. Convert the string into a JSON. In the case of an error, the object is not valid.
3. Decoded Object and linked entities (BadgeClass, Profile) are verified for structure and property.
4. Determine the signed key for further testing against attributes of the VerificationObject and the Issuer Profile (both contains a URL to the public key).
5. Extracted URLs are used for a GET request. A correct retrieval (HTTP STATUS 200 OK) delivers the private key which will be stored.
6. Every trusted key is used to decode and verify the JWS object. If verification fails, the Assertion is not valid.
7. Controlling of the revocation list against the id of the badge. Validation can be assumed successfully if the id of the award does not appear in the list.

2.6 Selected technical Methods used for Open Badges

In this section, some technical concepts which are relevant to issue badges are explained. At first, metadata stored in JavaScript Object Notation, followed by the hash method and finally, the public-key cryptography which is used for signed Assertions are presented.

2.6.1 JavaScript Object Notation and Variants

Metadata is defined as structured data about data. There exist different types, structures, standards, usage, creation and maintenance [N. Press, 2004] [Riley, 2017]. Famous formats for interchange metadata are Extensible Markup Language(XML)[Group, 2008], Resource Description Framework(RDF)[W3C, 2014], Relative Expression Based Object Language[37] or JavaScript Object Notation.

JSON is easily-readable for humans can be computed by a lot of programming languages and a compact data format which is used to send between software applications. It is based on the ECMAScript Programming Language Standard [Bray, 2014]. A JSON file contains a text with a sequence of tokens that includes six structural characters (left square bracket, left curly bracket, right square bracket, right curly bracket, colon, comma), strings, numbers, and three literal names. The serialised values consist of an object, array, number, string or three literal names (true, false, null). A parser is used to transform the text into another object or data types.
An enormous amount of documents or web pages is distributed over the Internet. An approach to form a network made by machines is Linked Data [Berners-Lee, 2006]. JavaScript Object Notation Linked Data is the variation in JSON and fulfils the requirements of Linked Data [Sporny et al., 2017]. In addition to the predecessor, there are more features for JSON-LD such as:

- A mechanism to identify JSON objects by IRI.
- Common used key by different JSON objects mapped to an IRI.
- A method to refer to an attribute of another JSON object.
- Annotation of property with their language.
- Association of data types with specific values.
- Opportunity to express one or more directed graphs within a document.

JSON-LD is applied for Representational State Transfer (REST) Web services and NoSQL databases such as CouchDB and MongoDB[38].

JSON Web Signature has a JSON structure and is used to secure content using a signature or Message Authentication Code (MAC). The mechanism guarantees the integrity of the data by using cryptographic methods [Jones et al., 2015]. A JWS consists of three components, which are separated by two dots: JOSE Header, JWS Payload and JWS Signature. The header contains the information about the cryptographic operations and parameters. A signed message is a payload, and the signature is the digital entity or MAC over the two other components. The first step to create a JWS is to encode the original text of the Header and the payload using base64url [Josefsson, 2006]. Afterwards, the JWS signature which is created by a cryptographic algorithm (e.g. RS256 for Open Badges) is also encoded by base64url. Finally, the three components are concatenated with "."

[37] http://www.rebol.com/, accessed 15.07.2017
[38] https://json-ld.org/, accessed 24.05.2017

2.6.2 Hash Function

Hashing is a mathematical function $H()$ which is maps a data D of any size into a data of a fixed size F. The mathematical expression is: $F = H(D)$. This is a One Way Function, and the result is called hash value or message digests [Bakhtiari et al., 1995]. A few points are important in creating a "good" hash function [Rogaway and Shrimpton, 2004]:

- preimage-resistance - that means it is not possible to compute the input from the hash value.
- collision resistance - within a good hash-function, it is not feasible to compute the same hash value with different input.
- determinism - the given input always computes the same hash value.
- uniformity - every hash value is computed with the same probability.

The application areas for hash functions are:

- Databases - computing an index for a table by hashing the value.
- Checksum - computing a hash value for a dataset and send both objects. By repeating the hash function over the data, a manipulation can be proven if the result of the hash is different to the checksum.
- Cryptographic - Signature for a message and integrity test.

2.6.3 Asymmetric Cryptography

Asymmetric cryptography (also called public-key cryptography) is a technique which is used for secure communication as well as digital signature [M. E. Hellman, 2002]. For signed Assertions, this method is used so that the achievement of the Recipient can be verified. Two applications and same theoretical aspects are described in this section.

2.6.3.1 Public-key Encryption

Public-key encryption is a technique were communication partners send messages over insecure channels. It is not necessary to share a secret (key) with the opposite partner, in comparison to the symmetric cryptographic [Diffie and M. Hellman, 1976].

For further action, two entities which are dissimilar are needed. These two keys are created by a key-generated function. One of them the public-key K_E is uploaded and so publicly accessible. The other one, the so-called private key K_D is kept secure. Basically, the public key is a product of two large randomly selected prime numbers. The private key is the two primes. This approach is secure because it is a great mathematical effort (exponential) to find the private key from the public or finding the factors of a large number. A second person can use the public key in an encryption-function E to process a message (M) and forwarding the ciphertext (C). After the C reaches the destination, only the owner of the private key (in case that the procedure and the key are still secure) can decrypt with the function D the message.

Encrypt: $C = E_{Ke}\{M\}$

Decrypt: $M = D_{Kd}\{C\}$

For a communication two sets of keys (every partner has a private as well as a public key which is not created by the same function call) are compulsory. Public-key encryption only grants confidentiality (not integrity or authentication). A disadvantage of the asymmetric encryption is the limited size of the message. The reason, therefore, is that the key has to grow linear to the message. And so a practical approach is to combine asymmetric and symmetric cryptographic. Only the key which is used to cypher the message is asymmetric encrypted and sent over the channel, additional to the encrypted message [A. J. Menezes et al., 1996, p.512]. Important methods are Rivest, Shamir und Adleman(RSA) Algorithm, Elliptic Curve Cryptography (ECC) and El Gamal [Rivest et al., 1978] [Jurišic and A. Menezes, 1997] [ElGamal, 1985].

2.6.3.2 Digital Signature

The signature of a digital entity can be obtained by the public-key cryptographic [Rivest et al., 1978]. The reasons for using is that integrity (message is not modified during transmission), authentication (origin of the message) and non-repudiation (at a later time the signature cannot deny). This digital signature scheme consists of two parts: the signature generation algorithm and the signature verification algorithm [A. J. Menezes et al., 1996, p.426].

First, a message M is hashed so that the result is $\tilde{M}$: $\tilde{M} = H(M)$. In comparison to the application discussed before, the keys are used in the opposite direction. The private key K_s is used in a signature function S to create a signature S*: S* = $S_{K_s}(\tilde{M})$. The public key K_v is needed to verify the signature. Second, the message is hashed by the same method so that $\tilde{M} = H(M)$. Afterwards $\tilde{M}$ and the signature S* are taken as parameters for the verification function V which returns the result R, whereby $R \in TRUE, FALSE$. And so only if the result $R = V_{K_v}(\tilde{M}, S*)$ is TRUE the message respectively the signature should be accepted.

Chapter 3

Open Educational Resources

This chapter outlines the origin, definitions, categorisations, types, formats, areas of application in education and meta data which can be used to describe the entities for Open Educational Resources.

3.1 Origin and Definitions

In 2002 after a meeting of the UNESCO about the impact of open course material for higher education, the term Open Educational Resource appears as a recommendation for a service previously known as open courseware [Dahbi et al., 2002]. This conference was the starting point for further evolving as well as different definitions. Given the fact that there are more definitions[1], only a few examples are enumerated here:

The origin UNESCO definition which is created after the meeting is[2]:

> Open Educational Resources are defined as technology-enabled, open provision of educational resources for consultation, use and adaptation by a community of users for non-commercial purposes. They are typically made freely available over the Web or the Internet. Their principal use is by teachers and educational institutions support course development, but they can also be used directly by students. Open Educational Resources include learning objects such as lecture material, references, and readings, simulations, experiments and demonstrations, as well as syllabi, curricula, and teachers' guides.

Another definition is from the OECD [OECD, 2007]:

> OER are digitised materials offered freely and openly for educators, students, and self-learners to use and reuse for teaching, learning, and research. OER includes learning content, software tools to develop, use, and distribute content, and implementation resources such as open licences.

Finally, the William and Flora Hewlett Foundation[3] describes Open Education Resources as:

> OER are teaching, learning, and research re- sources that reside in the public domain or have been released under an intellectual property license that permits their free use and re-purposing by others. Open educational resources include full courses, course materials, modules, textbooks, streaming videos, tests, software, and any other tools, materials, or techniques used to support access to knowledge

[1] https://wiki.creativecommons.org/wiki/What_is_OER%3F, accessed 15.07.2017
[2] https://opencontent.org/blog/archives/247, accessed 15.07.2017
[3] http://www.hewlett.org/strategy/open-educational-resources/, accessed 15.07.2017

The area of this topic is now global, but at the beginning was the focus on nations which are less developed [Johnstone, 2005]. The learning material should support local teachers. The potential was also recognised by the well-developed countries, and so private initiative, starting with the USA, have been involved [Atkins et al., 2007]. However, the topic swapped over to Europe as well as Germany [Deimann et al., 2007]. A link between Austria and Germany, and as an analysis of the German-speaking area is the work of [Ebner et al., 2015]. In this paper, an overview about OER in secondary and higher education, but also for professional and further training is given.

3.1.1 Legal Aspects

Consequently, a closer look to OER under the aspects of rights is required. In the light of the definitions presented at the beginning (see 3.1), a German legal expert [Kreutzer et al., 2013, p. 10] has concluded:

- There is no uniform definition of OER. The essential difference is the question: should the material be distributed free of charge?
- The material is not free of cost, but the usage of it.
- Compared to other teaching and learning material the license makes the difference. Only by using an open content license (Creative Commons (CC), GNU General Public License) the resources are open.
- It is possible to establish an OER-strategy by the current copyright law in Germany.
- Open licenses allow the copyright owner to use the material in a simple manner.
- Without fees for the license, it is possible to create a model (business, publication, income) which enables to generate revenue.
- Depending on the strategy for OER, the current license model by Creative Commons covers most cases.

3.1.2 Terminology of Open Educational Resources

The term Open Educational Resource was not randomly selected, every word is meaningful, and so there are three major concepts which are explained in the next sections.

3.1.2.1 Open

The German translation of open, not only means open for everybody, but it is also a synonym for free [Deimann et al., 2015, p. 10]. Which does not mean that the material is free of charge [Ebner et al., 2015, p.11]. However, there is also a discussion, about the software that is used to create the OER as well as the format of the material. The purist recommends to use software which is Free and Open Source Software (FOSS) as well as the formats of the OER should also be open. Not in all definitions enumerated before (see section 3.1), this openness includes the access to the material for all groups of persons. However, as the central point, the license has to be considered and should guarantee the points which are described by the five Rs [Wiley, 2014]. These are the rights to:

- create a copy (Retain).
- reuse the content (Reuse).
- modify the content (Revise).

- combine the material with other material (Remix).
- share original and modified content (Redistribute).

3.1.2.2 Educational

Education is a wide area, and so it has been discussed: is OER, only for educational purpose? Depending on the definition (Hewlett Foundation), it includes also material which is created for another reason. Additionally, there is also the question: How should „Open Access“ (results of research eg. publications and datasets) be included into the topic OER. [Ebner et al., 2015, p.90]. Synergism is expected, under the umbrella of open access to free knowledge [Deimann et al., 2015, p.33]. It is recommended by Heise to close the gap between this areas of interest [Ludwig et al., 2013].

3.1.2.3 Resource

OER are mainly digital content. Although, some of the definitions do not exclude analogue material such as worksheets, books et al. However, not only the learning objects are part of OER. Consequently, there is also software for learning, content management, content development, the standards for OER and finally the tools for licensing to publish [Peña-López et al., 2007, p. 30] and so a conceptual map of OER can look like:

- Learning content: Material for which is published for learning (resources/objects) or reference (collections: Internet archive, Google Scholar)
- Tools: Open source software for development and distribution - Content Management Systems, Development tools, Social Software (example: wikis), Learning Management Systems (example: Moodle).
- Implementation resources: Licensing tools (creative commons, GNU free Documentation), Best practices (CMU design principles), Interoperability (IMS, SCORM, OKI).

Further information, especially in context of creator, structure, types can be found in the next section 3.2

3.2 Classification/Categorization

There is not only a legal aspect when it comes to OER. Consequently, there are several ways to classify OER. [Camilleri et al., 2014]. A separation can for example be made by the structure [Commons, 2011], by the participants [Weller, 2010] and by the types (interactivity, learning resource).

3.2.1 Classification by Structure

An approach for classification is to separate the material regarding its structure, complexity and variability. This categorisation is a more focused point of view at the resource:

- Individual OER: In generally so called learning objects are part of this group. These objects are defined as digital artefacts that can be used to support learning [Wiley and Edwards, 2002]. Typical forms are images, music, videos.
- Semi-structured OER: a higher rate of the structure. Typical applications are an encyclopaedia and digitised library collections.

- Highly structured OER: Learning objects collected in a textbook or an online course are included in this group. There is a strong “as-is“relationship. Combination, adoption, modification, and reconfiguration of parts are possible.

3.2.2 Classification by Producer

Finally, a classification in consideration of the producers of the material can be made. In this case, there are two ways to classify. Since 2009 one of these approaches is known as Big and Little OER [Weller, 2010, p.2]:

- Big OER: This material is generated within an institution. The result is teaching units or courses at a high level and distributed by special platforms.
- Little OER: The objects are generated by individuals. These persons are not working explicitly in the education sector. The quality and the costs of the products may not be high.

Also, this classification is known as organizationally-produced respective user-generated resources. The other option is the numbers of people which are producing the resources. In case a community has authored the material it is called peer-produced respectively crowd-sourced (as an example: Wiki). The work of a single person is designated as individually-authored resource.

3.2.3 Classification by Types

Two types of categorisation for learning material can be found in the data model for Learning Object Metadata (LOM) (see section 3.3.2). A differentiation is made by kind of document and the pedagogical method of the resource. The next list is from the vocabulary of the Swiss version of LOM[4]:

- Animation; Mobile Application; Audio; Bibliography; Picture/Graphic; Guide; Map; Sheet music; Reference book; Resource for interactive White-boards; Software; Text-document; Video, Website; Not specified;

Like the enumeration before, also this the pedagogical methods are designated by the Swiss Specialist Agency for ICT and Education on the forum educa:

- Demonstration; Exploration; Experiment; Case study; Formative evaluation; Free activity; Information research; Educational game; Methodological tool; Teaching scenario; Presentation; Project; Role playing; Self evaluation; Simulation; Summative evaluation; Tutorial; Exercise; Workshop;

3.3 Metadata of Learning Resource

A short introduction about metadata was made in the section 2.6.1 to explain the usage of JSON. Depending on the requirements there are different forms of metadata for resources (XML, JSON) [Rensing, 2013]. The most important organizations which have been defining the standards for learning resources are: IMS Global Learning Consortium, IEEE Learning Technology Standards Committee[5], ISO/IEC[6], and Advanced Distributed Learning Initiative (ADL). The decisive factor depends on the use case. One objective here for is to ensure compatibility between different systems. As a result, it is easier to find the resource by a search query for an attribute of the data model, defined in the standard. The following subsections present three standards: IEEE Learning Objects Metadata, Dublin Core, and Learning Resource Metadata Initiative (LRMI)[Ziedorn et al., 2013].

[4] http://www.educa.ch/de/online-zugang/lom-ch, accessed 27.07.2017
[5] http://grouper.ieee.org/groups/ltsc/wg12/, accessed 29.07.2017
[6] https://www.iso.org/home.html, accessed 29.07.2017

3.3.1 Dublin Core

The origin of this initiative was a workshop in Dublin, Ohio, in the year 1995[7]. A solution for the problem to address resources in the World Wide Web (WWW) was the original reason for this initiative. The fifteen elements are the basic vocabulary to account the properties of a resource:

- Contributor: An entity responsible for making contributions to the resource.
- Coverage: The spatial or temporal topic of the resource, the spatial applicability of the resource, or the jurisdiction under which the resource is relevant.
- Creator: An entity primarily responsible for making the resource.
- Date: A point or period associated with an event in the lifecycle of the resource.
- Description: An account of the resource.
- Format: The file format, physical medium, or dimensions of the resource.
- Identifier: An unambiguous reference to the resource within a given context.
- Language: Language of the resource.
- Publisher: An entity responsible for making the resource available.
- Relation: A related resource.
- Rights: Information about rights held in and over the resource.
- Source: A related resource from which the described resource is derived.
- Subject: The topic of the resource (tags).
- Title: The name of the resource
- Type: The nature or genre of the resource.

In case extensions or specifications are needed, there are additional terms in the property[8] or element refinements.

3.3.2 IEEE Learning Objects Metadata

The Learning Objects Metadata is used to describe teaching and learning material. Published in the year 2002 it defines a data model which contains descriptive properties, used data types and the vocabulary for selected properties [Duval, 2002]. The original standard has nine categories and about 60 fields of attributes. The categories are separated into the areas: general, lifecycle (history and current state), meta-metadata (catalogue, date etc.), technical (information), educational (information), rights, relation (to other resources), annotation and classification. A detailed overview of all the attributes of the standard for LOM is provided by figure 3.1. Some countries such as Switzerland or France have defined their standards based on LOM [Adrian et al., 2017]. The ADL uses this standard as the Sharable Content Object Reference Model (SCORM)[9]. However, in practice, this standard is not used frequently by users

[7] http://dublincore.org/documents/dces/, accessed 29.07.2017

[8] http://dublincore.org/documents/dcmi-terms/#H2, accessed 29.07.2017

[9] An initiative from the US government to support learning through the use of technology. http://www.adlnet.gov/adl-research/scorm/scorm-2004-4th-edition/, accessed 10.08.2017

to describe their resources. The numbers of attributes are too high for practices, and so Dublin core finds a wider use. [Rensing, 2013].

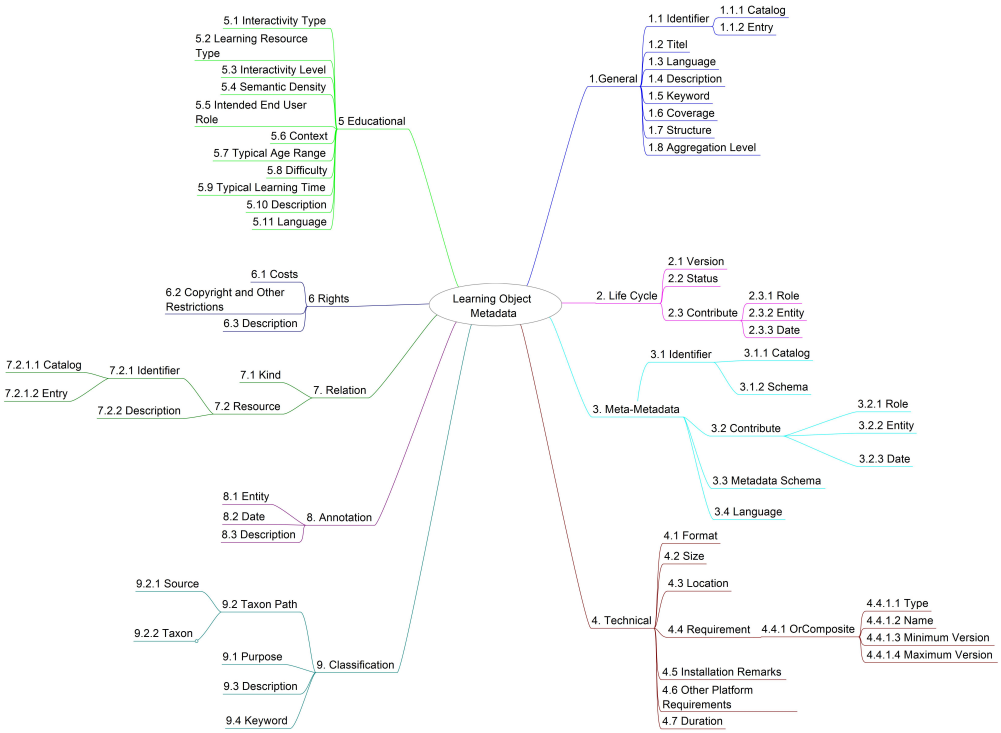


Figure 3.1: The data model of the learning object metadata version 1484.12.1-2002 - IEEE Standard (by Robert Hafner, licensed CC-BY).

3.3.3 Learning Resource Metadata Initiative

The goal of Learning Resource Metadata Initiative (LMRI) is the publishing, finding and distribution of educational resources on the internet. Funded by the Bill&Melinda Gates Foundation and the William and Flora Hewlett Foundation, the Association of Educational Publishers and Creative Commons published 2012 a new metadata standard for educational material [Barker and Campbell, 2015]. Since 2014 it is part of the Dublin Core Metadata Initiative[10]. Similar to LOM also attributes for educational purposes are defined, but commonly, the user has less effort to entering the data. The collection of classes and properties are build up to the vocabulary of some objects of *schema.org*: CreativeWork, AlignmentObject and EducationalAudience.

- **CreativeWork**: educationalAlignment, educationalUse, timeRequired, typicalAgeRange, interactivityType, learningResourceType, useRightsUrl, isBasedOnUrl, name, about, dateCreated, author, publisher, inLanguage, accessibilityAPI, accessibilityControl, accessibilityFeature, accessibilityHazard.

- **AlignmentObject**: alignmentType, educationalFramework, targetDescription, targetName, targetUrl.

- **EducationalAudience**: educationalRole

[10] http://dublincore.org/dcx/lrmi-terms/1.1/, accessed 10.08.2017

Chapter 4

Concept for OER Certification on Austrian Universities

This chapter provides in the first section a detailed insight into the concept of the OER certification. Afterwards, the requirements are transferred to a project so that the most significant aspects such as stakeholders, business rules, deliverables, use cases, non-objectives and mockups can be read respectively seen.

4.1 Current situation

The society *Forum neue Medien in Austria* referred to as fnm-austria is an inter-university network which represents their interests for developing and distribution of interinstitutional measures and models in the field of (technology supported) educational services. There are three missions statements. First, the society is the single nationwide E-learning network in the German-speaking area. Second, the society is the single association which campaigns for the use of new media. Finally, the society is a platform for Austrian universities, corporate partners and interests representatives [fnm-austria, 2012].

The working group "Open Educational Resources" of the society fnm-austria, published a document which relates to the claim of integration Open Educational Resources at Austrian universities [Ebner et al., 2016]. In this document, there is one item in an agenda with regards to the recommendation to create and establish a national label for OER. A national authority is supposed to be issuing the certifications for OER to support the intra-university exchange and distribution of OER. The underlying idea is that the creation and distribution of OER should be certificated. Nevertheless, it is not the task of the national authority to ensure the quality of the resource as well as legal certainty. It may be conjectured that lecturer at a university can decide on the correctness of their self-made learning material. Equally important is the support of the universities in the form of further education and repositories for OER [Ebner et al., 2017].

4.1.1 Specification of the Concept for OER-Certification at Austrian Universities

There are two fields of certification: first, the university lecturer and second, the universities themselves. The national authority which is responsive for the certification is staffed by representatives of the universities. Thereby, expertise and objective assessment are granted.

4.1.1.1 University Lecturers

The procedure for university lecturer contains two parts. First, the person has to create and publish three OER-objects (Lecturer Part 1 (LP1)). Second, the person has to prove the attendance of further training for OER (Lecturer Part 2 (LP2)). For the certification of level 1, it is compulsory to prove part 1 or part 2. For level 2 both parts are needed (Lecturer Level 1 (LL1), Lecturer Level 2 (LL2)). The lecture applies the certification at the national authority by the presentation of evidence. Both certifications cannot expire and remain valid after the lecturer moves to a different Austrian university. The certification will be stored in a database of the national authority as well as in the form of an Open Badge by the lecturer. If a learning material is created by more than a person, each originator can apply for an Open Badge. The evidence for the OER-objects is the storing in a repository of the Austrian universities or a suitable platform selected by the universities or an accredited platform (listed in a central register). It is recommended to use metadata for a closer description of the material. Furthermore, the usage of the free license CC-0, CC-BY or CC-BY-SA is recommended.

4.1.1.2 Universities

This section describes the further training for lecturer as well as the task for the universities. The further training for lecturer shall be offered by all Austrian universities in comparable modes as well as the same workload and moduls. The project "Open Education Austria" offers a proven example for such a course. This seminar is based on the concept of blended learning [Garrison and Kanuka, 2004] and starts with a three-hour presence workshop. Afterwards, a 10-hour online course (MOOC) and finally, a 6-hour presence course. Additional, there are 6 hours of self-study for the participant. Overall the recommended syllabus requires 25 hours. The respective educational institute plans and conducts the course as well as issues the certification for the further training.

The certifications for the universities themselves is the second area of the OER-certifications and similar to the lecturer the certification approach, which consists of parts and levels. This evidences of achievements are in the field of experience and competence of open learning and teaching material. The area is separated into numbers of lecturer which are awarded by an OER-certification and provision of further training as well as infrastructure. Additional, the strategic approach is also firmly established by an OER strategy.

The certification for universities is separated into three parts. First, the university offers further training for OER and commits to Open Educational Resources at universities (University Part 1 (UP1)). Second, the university has a certain number of lecturers (University Part 2 (UP2)). Finally, the university (or in cooperation with other universities) provides a repository for the lecturer to store their OER (University Part 3 (UP3)).

As previously taught by the lecturers there are levels for universities. Instead of two, for the universities are three levels. The first level (University Level 1 (UL1)) is completed by fulfilling one of the three parts. For level two (University Level 2 (UL2)), two parts have to be fulfilled. Finally, level three includes to accomplish all three parts (University Level 3 (UL3)).

The modus operandi is that the certification is applied at the national authority by the universities or representatives by proving certain criteria. The Open Badge expires after three years and shall be proposed again. If a new level is reached during the regular time-frame, the period starts again beginning with the new level. The certification is an Open Badge and is stored by the recipient as well as by the national authority. Additional information for the part 2 of certification at the university is listed here (numbers of OER-lecturers in relation to numbers of student per university):

- university with less than 1000 students - number of OER certified lecturers 5
- university with more than 1000 and less than 3000 students - number of OER certified lecturers 10

- university with more than 3000 students and less than 5000 - number of OER certified lecturers 20
- university with more than 5000 students - number of OER certified lecturers 40

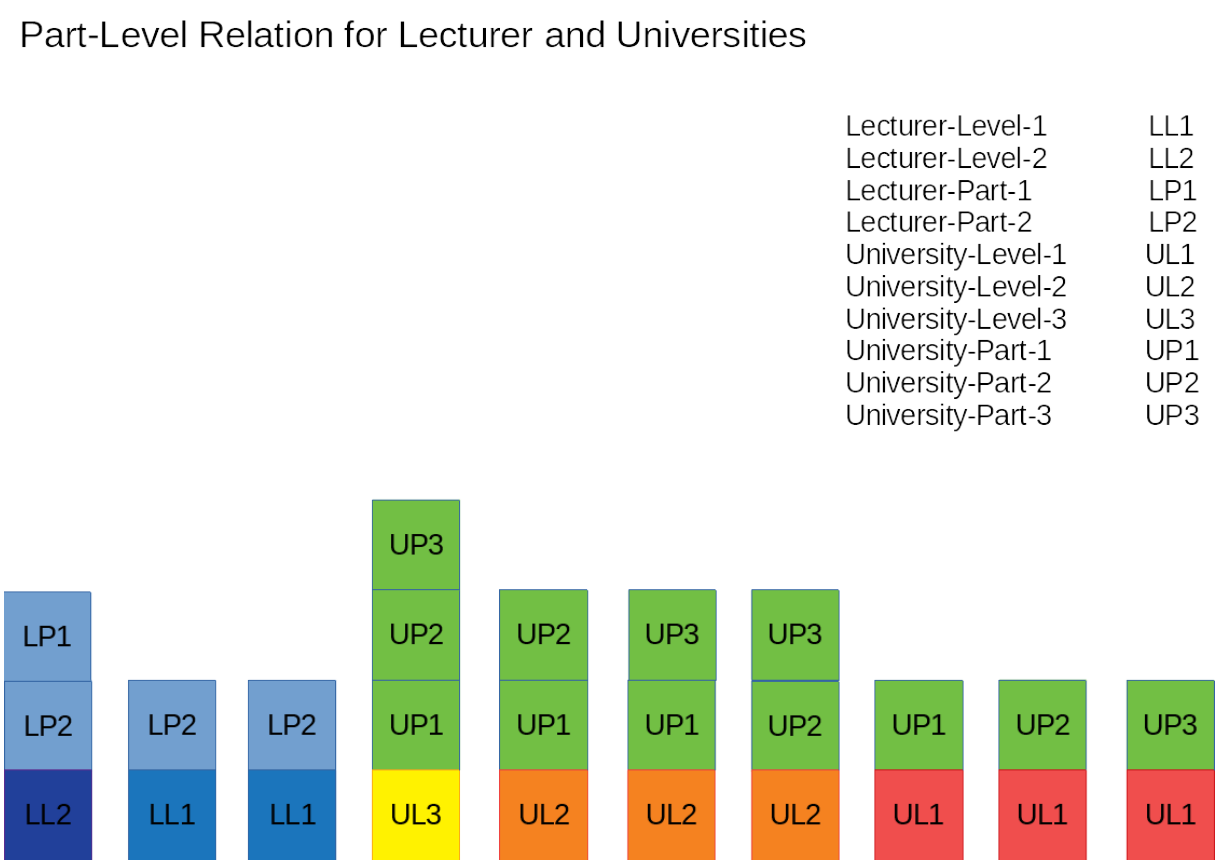


Figure 4.1: This diagram should show the relations between parts and levels (Robert Hafner CC-BY).

4.2 Project

The concept for OER-certification at Austrian universities is the foundation for a software application. Before the implementation can be started a detailed presentation about stakeholders, actors (rules), business rules, non-objectives, deliverables, phases, user stories and non-functional requirements are required. In the following part, project is used as a synonym for work respectively thesis.

4.2.1 Project Owner, Sponsors and Stakeholder

Stakeholder:

1. Central Authority: The Central Authority is the project owner and takes responsibility for the project and the business rules.
2. Universities of Austria
3. University Lecturer: person with teaching responsibilities at an Austrian university.

4. OER-representative of a university: is the link between the university and the Central Authority.
5. Administrator/Editor: executive person of the Central Authority. Verifies the data of the recipient and maintenance the software application.
6. System-administrator for Content-Management System: maintenance (e.g update, migration)
7. Developer
8. External Developer
9. Ministry of Science, Research and Economy[1]
10. fnm-austria
11. public
12. Consumer to the website

4.2.2 Actors

Out of the active stakeholders, this section describes the actors of the software application. Later on, these actors are needed for the use cases of the software application (see 4.2.6).

- University lecturer: employees of an Austrian university
- Representative of the participating University: responsible for the OER-agenda.
- Administrator/Editor: employee of the Central Authority who administrates the content of the Content Management System.
- Visitor of the website

4.2.3 Business Rules

The concept designates same specific requirements and objectives for the software application. In this sections, the business rules are named and explained in detail so that the application can be implemented without inconsistency. Business rules can be used, to avoid contradictions. For this work as a template, the IBM WebSphere ILOG JRules were used to define and declare the entities in detail. IBM WebSphere ILOG JRules is part of a business rule management system (BRMS) which is a tool for non-developer to understand and configure an application [Boyer and Mili, 2011]. These rules are defined as if-else constructs with variables and entities. Underneath the definitions, all the entities of a variable are named, so that a specific entity cannot be mistaken by another. For example 'Open Badge v.2.0 nonexpired Level 1 1' is a specific value (entity) of the variable Open Badge v.2.0 nonexpired Level 1. During "runtime" only this entity exists and can be identified. The awards are Open Badges of the v2.0, and as defined in the concept (see 4.1.1.1 and 4.1.1.2) they can expire or not. Between the two user groups, this is the most significant difference and will be discussed for the university in section 4.2.3.2.

4.2.3.1 Lecturers

The lecturers are one of the essential user groups, and therefore there are specific rules and requirements for the implementation. The first business rule for university teacher who receives an Open Badge v2.0

[1] https://www.en.bmwfw.gv.at/Seiten/default.aspx, accessed 20.10.2017

non-expired Level 1 can be seen in listing 4.1. The "university lecturer" is a variable and "university lecturer 1" is a value. The "university lecturer identity" is the property for identification. The lecturer could be identified by firstname, lastname and the email address. In this case, it is the email address of the lecturer which identifies also the affiliation to the university. This business rules presents exactly the behave of the system that the insertion of exact the same metadata three times leads to the award of a badge. It seems to be a better idea to change this rule. As a result of a change of the employer, the new affiliation is stored and has effects on the second part of university certification (see 4.6). The second business rule handles the issuing of an Open Badge v2.0 non-expired Level 2 for university lecturer (see listing 4.2). Essential for the second rule is that in case a lecturer has done a further training and each three time entering of OER metadata issues an Open Badge v2.0 non-expired Level 2. Another upload of a further training and three OER-objects is not necessary to fulfil business rule 2. It is recommended to extend the rules so that another level will be created if the numbers of non-equal OER metadata are published. Overall, it seems to be necessary to inform the Recipient after the entering of the data, what the Recipient needs in order to complete the part (entering still x numbers of OER) or to achieve the next level.

```
definitions
set 'Open Badge v.2.0 nonexpired Lecturer Level 1 1' to a Open Badge v2
    .0 nonexpired Lecturer Level 1;
set 'austrian university 1' to an austrian university;
set 'further training OER 1' to a further training OER;
set 'OER metadata 1' to a OER metadata;
set 'OER 1' to a OER;
set 'university lecturer 1' to a university lecturer;
set 'university lecturer identity 1'to a university lecturer identity;
if
'university lecturer 1' of an  'austrian university 1' enters the 'OER
    metadata 1' of an self-made or participated 'OER 1' three times

or

'university lecturer 1' of an  'austrian university 1' verifies the
    certification of a 'further training OER 1'  at an Austrian
    university

then

'university lecturer 1' identified by 'university lecturer identity 1'
    applies an
'Open Badge v.2.0 nonexpired Lecturer Level 1 1'
```

Listing 4.1: The business rule to achive an Open Badge v.2.0 for universitiy lecturer of level 1.

```
definitions
set 'Open Badge v.2.0 nonexpired Lecturer Level 2 1' to a Open Badge v2
    .0 nonexpired Lecturer Level 2;
set 'Austrian university 1' to an austrian university;
set 'further training OER 1' to a further training OER;
set 'OER metadata 1' to a OER metadata;
set 'OER 1' to a OER;
```

```
set 'university lecturer 1' to a university lecturer;
set 'university lecturer identity 1'to a university lecturer identity;
if
'university lecturer 1' of an  'austrian university 1' enters the 'OER
   metadata 1' of an self-made or participated 'OER 1' three times

and

'university lecturer 1' of an  'austrian university 1' verifies the
   certification of a 'further training OER 1'  at an Austrian
   university.

then

'university lecturer 1' identified by 'university lecturer identity 1'
   applies an 'Open Badge v.2.0 nonexpired Lecturer Level 2 1'
```

Listing 4.2: The business rule to achive an Open Badge v.2.0 for universitiy lecturer of level 2.

4.2.3.2 Austrian Universities

The Austrian universities are the other essential user groups of the application. The common rules for achieving this certification see section 4.1.1.2. There are three business rules for the universities to achieve Open Badges v2.0 expired Level 1, Level 2 and Level 3. The three business rules have in common that at first the input of the data is only a request for a specific certification. The listing 4.3 describes the rules for an Open Badge v.2.0 expired University Level 1. As can be seen in the listing only one of the parts have to be fulfilled, by one Austrian university with a particular identity. Add this phase of the project only a single email address should be used. The next listing 4.4 describes the rules for an Open Badges v.2.0 expired University Level 2. Here, it is necessary to fulfil two of the university parts (1 and 2, 1 or 3, 2 or 3). Finally, all three parts for the universities have to be completed, so that a Open Badges v.2.0 expired University Level 3 can be issued (see listing 4.5).

Strictly speaking, University Part 2 is done by the administrator, cause the information about the numbers of OER-lecturers by a particular university, is hold by them. In this phase of the project it is not indented to implement a view for an overview of the numbers and/or names of the OER-lecturers. For more details about this business rules see section 4.2.3.3.

A different situation is the fact that a new level is reached by the time a badge is not expired. For this case, the badge with the lower level is not revoked. By upgrading from a lower level to a higher level the timeframe until the new badge is valid starts again by three years. It is not exactly defined what is happening to the badge with a lower level as well as the situation if the older part is not valid anymore. Formal the requirements are still fulfilled. That means that an expired badge with a lower level does not revoke a badge with a higher level. A correct procedure is to check every badge with a higher level of a badge with a lower level is expired, and the badge with the higher level have to be revoked until the requirements are again fulfilled, and a new badge with the higher level can be issued. However, this topic has to be discussed.

```
definitions
set 'Open Badge v.2.0 expired University Level 1 1' to a Open Badge v2
   .0 expired University Level 1;
set 'austrian university 1' to an austrian university;
```

```
set 'further training OER 1' to a further training OER;
set 'OER strategy 1' to a OER strategy;
set 'OER number of lecturer 1' to a OER number of lecturer;
set 'OER repository 1' to an OER repository;
set 'university lecturer 1' to a university lecturer;
set 'university identity 1'to a university identity;

if
'university 1'  offers 'further training OER 1' and commits to 'OER
   strategy 1'

or 'univeristy 1' has 'OER number of lecturer 1'

or 'OER repository 1' provided by 'university 1' or 'univerity 1' is
   partof 'university 1 - n'

then
'university 1' identified by 'university identity 1' can requests for
   an 'Open Badge v.2.0 expired University Level 1 1'
```

Listing 4.3: The business rule to achive an Open Badge v.2.0 for Austrian universities level 1.

```
definitions
set 'Open Badge v.2.0 expired University Level 2 1' to a Open Badge v2
   .0 expired University Level 2;
set 'austrian university 1' to an austrian university;
set 'further training OER 1' to a further training OER;
set 'OER strategy 1' to a OER strategy;
set 'OER number of lecturer 1' to a OER number of lecturer;
set 'OER repository 1' to an OER repository;
set 'university lecturer 1' to a university lecturer;
set 'university identity 1'to a university identity;
set 'university part 1' = 'univeristy 1'  offers 'further training OER
   1' and commits to 'OER strategy 1';
set 'university part 2' = 'univeristy 1' has 'OER number of lecturer
   1';
set 'university part 3' = 'OER repository 1' provided by 'university 1'
    or 'univerity 1' is partof 'university 1 - n'

if
'univerity part 1' and 'univerity part 2'

or 'univerity part 1' and 'univerity part 3'

or 'univerity part 2' and 'univerity part 3'

then
'university 1' identified by 'university identity 1' can requests for
   an 'Open Badge v.2.0 expired University Level 2 1'
```

Listing 4.4: The business rule to achive an Open Badge v.2.0 for Austrian universities level 2.

```
definitions
set 'Open Badge v.2.0 expired University Level 3 1' to a Open Badge v2
    .0 expired University Level 3;
set 'austrian university 1' to an austrian university;
set 'further training OER 1' to a further training OER;
set 'OER strategy 1' to a OER strategy;
set 'OER number of lecturer 1' to a OER number of lecturer;
set 'OER repository 1' to an OER repository;
set 'university lecturer 1' to a university lecturer;
set 'university identity 1'to a university identity;

if
'univeristy 1'  offers 'further training OER 1' and commits to 'OER
    strategy 1'

and 'univeristy 1' has 'OER number of lecturer 1'

and 'OER repository 1' provided by 'university 1' or 'univerity 1' is
    partof 'university 1 - n'

then
'university 1' identified by 'university identity 1' can requests for
    an 'Open Badge v.2.0 expired University Level 3 1'
```

Listing 4.5: The business rule to achive an Open Badge v.2.0 for Austrian universities level 3.

For the UP 2 a clear rule has to be defined in case a OER-lecturer is leaving an Austrian university or switching to another Austrian university. It is recommended that as long as a lecturer is not publishing another OER with a new identity (in most of the cases it will be a new email address with the domain of the new employer) the lecturer is "counted" for the university which was the last employer. At the moment a new identity (email) is named by an application the lecturer increase the number of OER lecturer of the new university and the number of OER for the former employer decreases. A detailed listing can be seen in 4.6. Additional it is recommended to store former identity of the OER-lecturer for further analyse.

```
definitions
set 'OER number of lecturer university 1' to a OER number of lecturer
    university;
set 'OER number of lecturer university 2' to a OER number of lecturer
    university;
set 'Open Badge v.2.0 non-expired Lecturer Part 2' to a Open Badge v
    .2.0 non-expired Lecturer Part 2
set 'university lecturer 1' to a university identity;
set 'university lecturer identity 1' to a university lecturer identity;
set 'university lecturer identity 2' to a university lecturer identity;

if 'university lecturer 1' is changing 'university lecturer identity 1'
to  'university lecturer identity 2'

and  'university lecturer 1' owns 'Open Badge v.2.0 non-expired
    Lecturer Part 2 '
```

```
then

increase  'OER number of lecturer university 2'

decrease  'OER number of lecturer university 1'
```

Listing 4.6: The business rule for Administrator of the application for the issue of an 'Open Badge v.2.0 expired University Level *'.

4.2.3.3 Administrator-Editor

The administrator of the application is the representative of the Central Authority and therefore this person has to complete different tasks. The person has to fulfil the tasks of an Issuer (see section 2.3.1) and supervise all the requests from the people or organisations who apply an Open Badge v2.0 * to meet the formal criteria (see section 4.1.1.1 and 4.1.1.2). Except for the UP 2, to get a certification, a request of the Recipient must be submitted on the website of the national authority. As a result of this, it is not the task of the university to submit a request for the certification. It is not the responsibility of the administrator to control the list of the lecturer and the affiliated university so that the Open Badge can be issued. Instead, it is the task of the Content-Management-System to apply the request for a particular university if the requirements are fulfilled. It is the duty of the administrator to enter the criteria so that a university can receive a certification. If the identity of the university (email address) is unknown, the administrator shall contact the university so that they can receive the badge. There are particular cases for badges which can expire.

```
definitions
set 'Open Badge v.2.0 nonexpired Level 1 1' to a Open Badge v2.0
   nonexpired Level 1;
set 'Open Badge v.2.0 nonexpired Level 2 1' to a Open Badge v2.0
   nonexpired Level 2;
set 'Open Badge v.2.0 expired Level 1 1' to a Open Badge v2.0 expired
   Level 1;
set 'Open Badge v.2.0 expired Level 3 1' to a Open Badge v2.0 expired
   Level 3;
set 'austrian university 1' to an austrian university;
set 'further training OER 1' to a further training OER;
set 'OER strategy 1' to a OER strategy;
set 'OER repository 1' to an OER repository;
set 'applier 1'to a applier;

if
'Open Badge v.2.0 nonexpired Level 1 1'

or 'Open Badge v.2.0 nonexpired Level 2 1'

or 'Open Badge v.2.0 expired Level 1 1'

or 'Open Badge v.2.0 expired Level 2 1'

or 'Open Badge v.2.0 expired Level 3 1'

is applied
```

```
then

if
condition for Open Badge v.2.0 nonexpired* is true #common case

then
(Open Badge v.2.0*) is issued to applier
else
(Open Badge v.2.0*) is not issued
```

Listing 4.7: The business rule for Administrator of the application for the general issue of all badges.

```
definitions
set 'Open Badge v.2.0 expired Level 2 1' to a Open Badge v2.0 expired
   Level 2;
set 'OER number of lecturer 1' to a OER number of lecturer;
set 'Students of univeristy 1' to a Students of univeristy;
set 'university identity 1' to a university identity;
set 'university 1' to a university;
set 'range number students university 1' to range number student
   university;

if
'OER number of lecturer 1' is 'range number students university'

then
if
'university identity 1' is true # verify identity
then
'university identity 1' receives an 'Open Badge v.2.0 expired Level 2
   1'
else
contact 'university 1' for   'university identity 1'
```

Listing 4.8: The business rule for Administrator of the application for the issue of an 'Open Badge v.2.0 expired Level 2 1'.

4.2.4 Deliverable

For the certification of university lecturer and the universities a software prototype is to be developed, implemented and tested. The following Components are to be delivering:

- Source code of the Application at a Git-repository.
- Documentation for maintenance and further developing.
- Software application

4.2.5 Phases

This project is separated into two phases. In the first phase the version 1.0, is mapping all the necessary functional requirements of the application. During the first step, the decisions about architecture, and design must be considered in regard of further development. Recommendations for further development will be described in section 7.

Phase 1: At this point, the focus is on the general functionality for the lecturers and the representative of the university as well as the tasks of the administrator. These entities are identified by the name and the email address.

Phase 2: Generally speaking, automations for the tasks of the administrator and usability should be considered.

4.2.6 Use Case

The use cases are separated into the different user groups. The enumerations follows the schema: UC is the abbreviation of user case, the following number is the actor (see 4.2.2), and finally the last number is a simple enumeration. Only the simple questions who, what, and why shall be answered, so that an agile software developing can be done. An overview about the actions and interactions of the actors can be seen in figure 4.2.

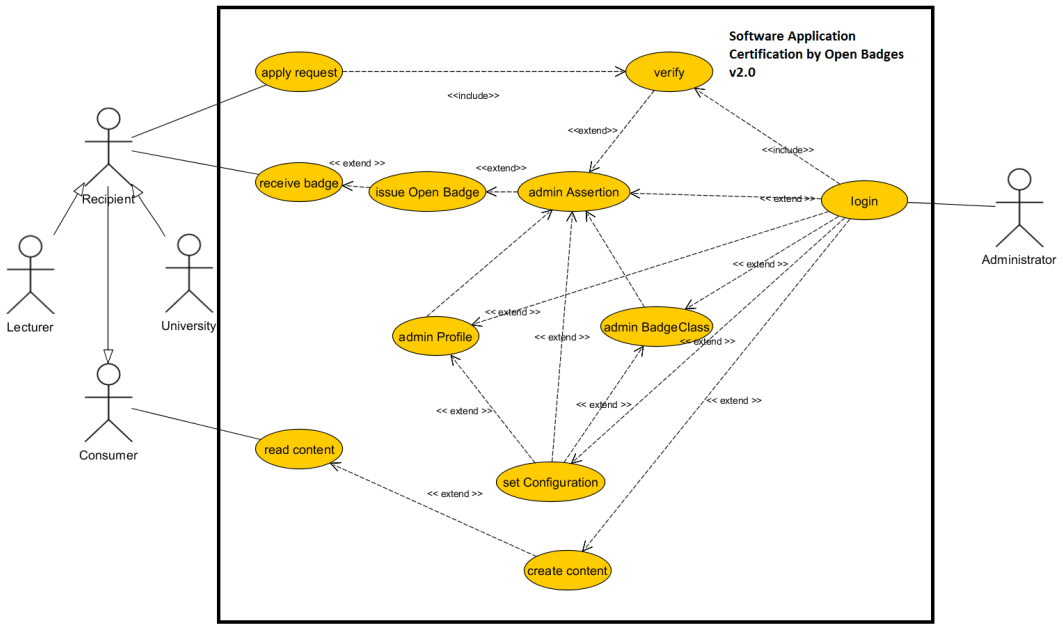


Figure 4.2: A use case diagram shows the dependencies between the actors and the system (Robert Hafner CC-BY).

4.2.6.1 Phase 1 User Group: University lecturer

UC.1.1 Enter the metadata of the created OER: a simple form is used to insert data about the Open Educational Resource. This is linked to UC.2.2. A variety of forms are used to insert the metadata.

This forms shall be chosen by the lecturer. The specified email address is used to identify the recipient.

UC.1.2 Certification of further training: The evidence can be done by the upload of the written confirmation. The specified email address is used to identify the recipient.

4.2.6.2 Phase 1 User Group: OER-Representative at Austrian universities

UC.2.1 Commitment and further training: The announcement of an OER further training and public commitment to open teaching and learning material. The OER Representative confirms the both requirements by enter the URL of the website for OER further training as well as the public commitment to OER. The specified email address is used to identify the recipient.

UC.2.2 Numbers of OER-lecturer: For this part, only the specified email address of the OER-representative is needed. It is a requirement of the backend and the administrator needs to verify the number as well as to issue the badge.

UC.2.3 The announcement of the repository: the OER-representative enters the email address and the URL of the repository or the website of notification for cooperation.

4.2.6.3 Phase 1 User Group: Administrator-Editor of national authority

UC.3.1 Login: Administrator logs into the backend to fulfil the prescribed tasks.

UC.3.2 Configuration setting for the system: The administrator has a dashboard to enter the main configuration input such as domain, directory of the private key etc.

UC.3.3 Verification of the requests. The entries of the universities as well as the lectures must be verified so that a badge can be issued.

UC.3.4 Condition: The mapping of the conditions by creation, modification, and deletion of potential recipient entities.

UC.3.5 Editorial content generation: If the Central Authority will changes some content of the website such as instruction for us as well as new informations about a new policy.

UC.3.6 Overview and maintenance (create, read, update, delete = CRUD) of Open Badge data class BadgeClass.

UC.3.7 Overview and maintenance (CRUD) of Open Badge data class Profile.

UC.3.8 Overview and maintenance (CRUD) of Open Badge data class Assertion.

4.2.6.4 Phase 1 User Group: Consumer/Visitor

UC.4.1 Consumer view: Person who is interested in OER-certification and work of national authority.

4.2.7 Mockups

Part of this phase of the work was the design of a few mock-ups to provide a first impression of the layout. In figure 4.3 a first design of the root page can be seen. Another mockup which can be seen in figure 4.4 presents a form so that a lecturer can enter the metadata of OER (UC.1.1) When building

the prototype, it was more important to implement and test the functionality, and therefore, the mockups were only used as a rough template. The mockups were created using Pencil[2].

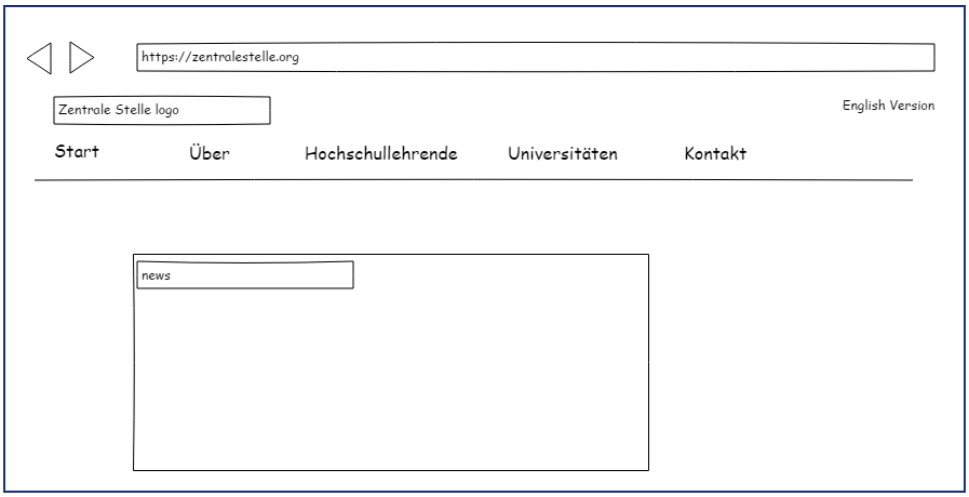


Figure 4.3: Mockup of the start page (Robert Hafner CC-BY).

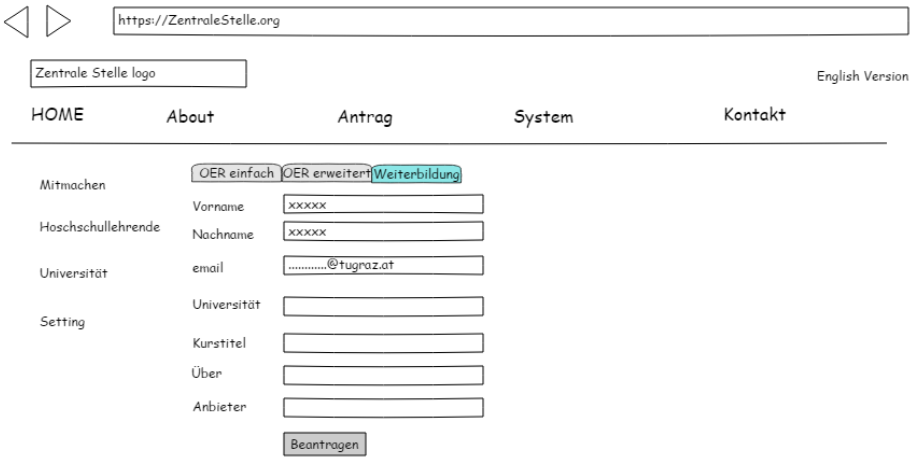


Figure 4.4: Mockup of the page for the input field of lecturer to publish the further training (UC1.2) (Robert Hafner CC-BY).

4.2.8 Non-Objectives

As important as the objectives of a project, are the non-objectives. They are listed in this section to avoid misunderstandings.

The input from a lecturer into the CMS should not automatically verified:

- the legality of the OER.

[2]Open-source tool to create mockups for different platforms. https://pencil.evolus.vn/, accessed 28.09.2017

- the correctness respectively validity of the URL.
- the re-enter of the OER by the lecturer as well as another person.
- the identity of the lecturer (if the person is still working for university)

The input from a OER-representative of a university into the CMS should not automatic verify:

- the validity of the URL regarding the OER strategy of a university.
- the validity of the URL regarding the further training.
- the validity of the URL regarding the repository.

4.2.9 Non-functional Requirements

As important as the functional requirements are for a software project, also the non-functional requirements have to be presented. And so in the following section some points are described,"not what the software will do, but how a software will do it" [Chung et al., 2012, p.2]. A detailed review of the TYPO3 non-functional requirements can be read in the chapter 6.

- **Portability:** The Content Management System should run on every platform, e.g. Unix-Systems, Windows as well as Mac OS. Depending on the CMS a web server, e.g. Apache HTTP Server should run so that PHP is supported. The Clients for entering the input, or reading the content should run on Personal Computer (portable, stationary) and mobiles (e.g. Android, iOS).
- **Reliability:** The system should behave consistently in a user-acceptable way so that the operations with the environment are working intended. Therefore it is necessary that the CMS enables: coding standards, unit tests, configuration management, software metric and software models.
- **Efficiency:** some characterisations how a system uses scarce computational resources such as capacity (e.g. max. numbers of users), degradation of service (system handle load, system crash) or time constraints.
- **Usability:** The system should be designed with usability and ease of use in mind by a simple user-interface for the potential recipient as well as for the administrator.
- **Dependability:** The dimensions such as availability, privacy ,safety (quality, concept, et al.), and security (threats such as XSS, SQL-injection et al.) should be a constraint too.

Chapter 5

Implementation

This chapter describes some aspects of the implementation of an Open Badges Issuer System in a Typo3 framework. The details about the Content Management System respectively TYPO3, and the development environment is presented. Afterwards, the software is divided into the two primary parts, front end and back end, and the aspect functionality and implementation are shown in detail.

5.1 Content Management System

A Content Management System (CMS) is a computer application that makes it possible to manage, create, store, edit and distribute content (files, images and media) on the internet [Youseck Yang and Lim, 2016]. Traditionally, it is used by people without technical skills, so that it becomes prevalent in the area of web applications [López et al., 2011]. The advantage of this system is the bridge between the programmer of the application and the content producer. In most cases, the system is divided into two parts: the content management application which is the front end user interface to process content by a user and the content delivery application which is used to compile and update the website [Mauthe and Thomas, 2004]. Since smartphones are favourite most of the CMS can satisfy the demand for web accessibility as well as to mobile accessibility.

Most of the CMSs are built on a LAMP (Linux, Apache, MySql and PHP) stack. Additionally, all of them provide the option to download or program modules or extensions so that the original functionality can be modified and enhanced. It is remarkable that the majority of the systems are open source [Nirgude and Giri, 2009]. World Wide Web Technology Surveys gives an overview about the market share of CMS. Wordpress[1] is in the first position with a market share of 59.6 %. Second is Joomla![2] with 6.7 % followed by Drupal[3]. TYPO3 is on the seventh place with 1.5 % market share[4].

5.2 TYPO3

TYPO3 offers different products for Content Management Systems. Besides, to the TYPO3 CMS, there are also the products Extbase, Fluid, and Surf. TYPO3 CMS is the mainframe and provides the basic functionality to process content. Extbase (see 5.2.4) is a framework to build extension. Fluid is the templating engine to render the user interface. Surf can be used for automatic deployment, and therefore maintain a continuous integration. For this project, only the first three products were used.

[1] https://de.wordpress.com/, accessed 20.10.2017

[2] https://www.joomla.de/, accessed 20.10.2017

[3] https://www.drupal.org/, accessed 20.10.2017

[4] W3Techs provides a detailed overview of the usage of various technologies on the web. https://w3techs.com/technologies/history_overview/content_management, accessed 28.09.2017

The TYPO3 Association is maintaining and developing four different Long Term Systems (as can be seen in the figure 5.1) [Association, 2017b]: TYPO3 v6, TYPO3 v7, TYPO3 v8 and TYPO3 v9. Currently, for TYPO3 v7.6.18 only bug-fixing is done, and TYPO3 v8.7.x is in regular maintenance. The changes between the versions were significant and now v8.7.x only uses PHP version 7. The MySQL version was upgraded from 5.5 to 5.7, and a significant reduction of Line of Codes was done[5]. The result is a serious acceleration of the CMS.

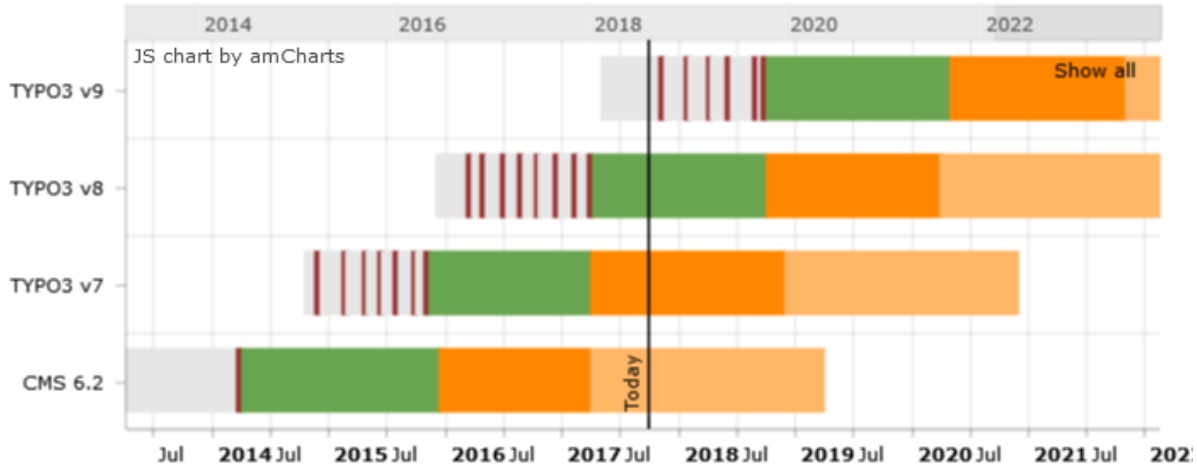


Figure 5.1: The roadmap of the Long Term Support versions of TYPO3 (source: typo3.org).

As presented in section 5.1 a CMS can be divided into two parts the front end and the back end. Since TYPO3 v 8.7.x the back end is now responsive and so the CMS can be admitted by mobile or tablet[6].

5.2.1 Typo3 Certification

By the increasing proliferation on the market, it was required to install a global standard for certification of the four actors of the CMS:

- **Editor:** This person has common knowledge about the Back end, especially pagetree, working with files, important element content types, working with lists, access rights etc.
- **Integrator:** This Person has common knowledge about TypoScript (a script language for configuration), templating with Fluid, page, page content, access control, localisation, internationalisation, install extensions, internals, troubleshooting.
- **Developer:** This person has common knowledge about architecture, Fluid, Repository, security, exception handling, debugging, and internals.
- **Consultant:** This person has in general a deeper knowledge, about terminology, possibilities, administration, project management, third-party services, search engines, maintenance etc.

In the following section the terms for the actors of TYPO3 will be taken, and so the formerly named administrator is now the editor. The role of the developer of this application includes now the role of the integrator and the developer.

[5]Detailed changes can be seen in https://typo3.org/download/release-notes/whats-new/, accessed 28.09.2017

[6]An overview of the most significant changes between v8.7.x and v7.6.18 can be read on this official website. https://typo3.org/download/release-notes/typo3-v8-release-notes/, accessed 29.09.2017

5.2.2 Front end

Usually, the front end is used to consume the content (e.g. blog entry, file) by the user. However, it is also possible to create content, and this input is sent to the back end so that it can be stored persistently in a MySQL database or a directory of the web-server. However, the primary approach by TYPO3 is to create content by a login into the back end as editor.

5.2.3 Back end Dashboard

The back end of TYPO3 can be used by at least three user groups: editor, integrator, and developer. First, the editor can publish distinct content such as header, text, images, file links and plugins or upload some content. Second, the integrator selects additional extensions or controls the access to distinct areas. Figure 5.2 presents a typical TYPO3 back end. The layout of this view is divided into four parts. The red framed box in the header is the menu for logout, error messages etc. On the left side, the green framed box contains the modules. These buttons open a system extension or an extension produced by a third party. By clicking on the icons on the left side of the module one or two areas appear (that depends on the selected module). The blue framed box presents the pagetree of the homepage and is used for orientation and as interaction element. The yellow framed box shows the detail area. In this image, the dashboard to select a plugin for generating content is presented.

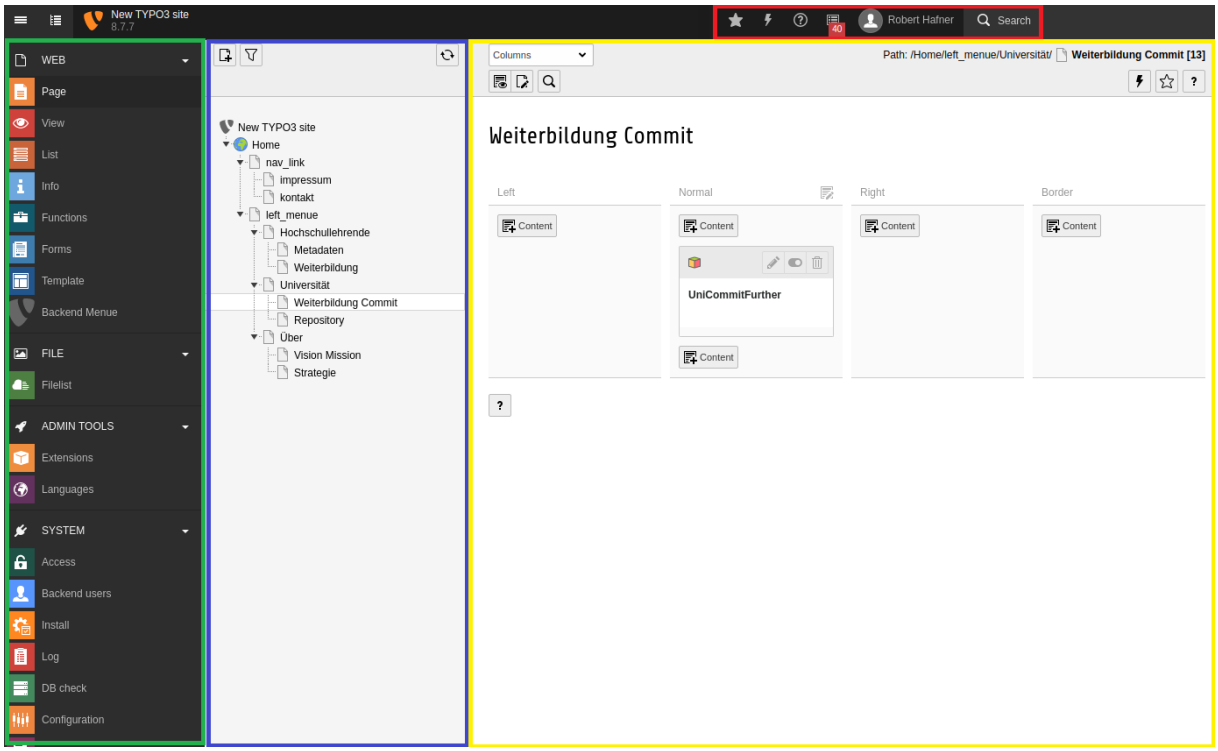


Figure 5.2: Screenshot of the back end of the TYPO3 v8.7.7 (Robert Hafner CC-BY).

5.2.4 Extension

This piece of software is used to extend or alters the functions of the CMS. The users of TYPO3 CMS is able to modify their System by choosing between 6000 Extensions. These extensions can be downloaded in the Extension Manager from an online repository or uploaded by a Zip file. In case an extension is

built the frameworks Extbase and Fluid are needed. [Rau et al., 2013]. TYPO3 recommends to using the following principles which by the way are system immanent:

- **Object Oriented Programming (OOP):** This programming paradigm is widely used by Extbase and so it is recommended to be used by the developer of an extension. The reason, for using it, is encapsulation, reuse, refactoring and avoiding of Spaghetti Code.
- **Coding Standard:** There are detailed guidelines for coding for the core as well as for the extensions[7].
- **Domain Driven Design (DDD):** A paradigm to develop a model which represents the reality as precisely as possible. The reason, for using, is a better understanding of the relevant problems so that the programmer works structured and efficient. A transient object is transformed into persistence by a so-called Repository. This class connects the object to the MySQL database so that a reconstitution (persistence into a transient object) is possible.
- **Model View Controller (MVC):** The Domain Driven Design and the Object Oriented Programming is completed by the Model View Controller pattern. OOP delivers the fundamental paradigm. DDD creates and stores the objects and the MVC is providing the class schema. The View encapsulates the output logic. For this purpose, Fluid provides the templates. The controller class contains the functions which can be called after a user interaction. This object coordinates and controls the actions between the view and the model. Finally, the model encapsulates the user logic and data.
- **Test Driven Development (TDD):** TTD is supported within TYPO3 by PhPUnit[8] and Test Runner[9]. The principle of TDD is to first write a unit-test and afterwards write the function. The result is that every function and not only an entire feature is tested. As a result a TDD developed system is testable by nature and instant deliverable. However, it requires discipline and mastery so that developers become more efficient [Erdogmus et al., 2010].

A feature of the extension is to implement a front end plugin as well as a back end module[10]. The plugin is an interface for the front end- and the back end-user to CRUD content by using Fluid elements and linking them to assigned controller functions. A similar concept is the back end module. Here it is only possible to operate the extension as back end user. A significant sign that a back end module is used is a small gif[11] at the left side at the module dashboard (see figure 5.2 green framed box). An extension consists of three parts[12]. :

1. A directory which is named after the extension key (a unique name that identifies the extension).
2. Standard files with a reserved name.
3. An arbitrary number of additional files.

5.2.4.1 Structure of an Extension

The structure of the extension reflects the principles for programming extensions, and it is well-established. If a new extension has to be programmed, it is recommended to us the structure which is presented in

[7] https://docs.typo3.org/typo3cms/CodingGuidelinesReference/Index.html, accessed 28.09.2017
[8] https://phpunit.de/, accessed 20.10.2017
[9] https://www.soapui.org/test-automation/running-from-soapui/functional-tests.html, accessed 20.10.2017
[10] There are four other kinds to extend functionality. However, only the front end plugin and back end module is used.
[11] https://www.w3.org/Graphics/GIF/spec-gif89a.txt, accessed 20.10.2017
[12] Website with further details about mandatory and optional files respectively directories for an extension. https://docs.typo3.org/typo3cms/CoreApiReference/ExtensionArchitecture/FilesAndLocations/Index.html#extension-files-locations , accessed 29.09.2017

figure 5.3. Every extension which is made by the extension_builder follows this structure. A short list of the files and directories with a short explanation are listed here:

- **ext_emconf.php:** Is the only mandatory file and is needed to describe the extension for the CMS. It contains title, description, visions, constraints (e.g. dependencies, conflicts), state, other metadata (author, author_mail), and autoloads information. The Extension Manager cannot find and install the extension as long as this file is not present
- **ext_localconf.php:** Contains hook and plugin configuration.
- **ext_tables.php:** Contains declarations of modules and back end styles.
- **ext_tables.sql:** SQL file which contains the extension tables. Maybe this file is not "dumpable" so that it can be inserted into a MySQL database.
- **Classes:** Contains the directories Controller and Domain. The standard action functions are: list, new, create, delete, update, show. The domain contains Model (the M from MVC) and Repository (files are the connector to the DB) directory.
- **Resources:** Contains the templates and partials of Fluid files to build a form.

Figure 5.3: A overview about the directories of an extension.

The Domain Model The modelling of a domain and consists of multiple step and is mostly the first step of the development of an extension. It starts by describing the application domain with terms that roles a system. As a next step the located rules and operations between the terms have to be defined. Afterwards, by the creation of a domain model, the relation of the objects (terms) to each other are defined. The domain model describes the association of at least two elements in the way that one object can described by a has-a object (the usage of an is-a-relation should be avoided) and how this hierarchical

relationship can be displayed as 1:1, 1:n (n:1), and m:n relation. Every object is a database table, and also the relations are mapped into the object table or another table as far as possible. As a result of this modelling, an object has to be selected as the root so that the aggregation can be accessed. If an object is chosen as root, this entity has more than the obligatory model class and a database table.

Additional, there is a Controller class which controls the flow, and a Repository class which is used as an interface to the database. Only by the Controller, this repository should be accessed. The repository is derived from an abstract class, and so standard queries are implemented. However, individual queries can be created in this class too. The access to the objects which are bottom down in the model is done by the getter and setter functions of the model. The inheritance of classes as an object-oriented paradigm has to be mapped into a relational database. Unfortunately, there is a mismatch between these two concepts, and so a compromise is needed. There are two options for a solution. Every concrete class has a table (Concrete Table Inheritance), or all classes of the hierarchy are combined in one table (Single Table Inheritance). A Class Table Inheritance which is matching the classes and the tables by their properties is still not implemented [Association, 2009].

The Controller links the View and the Model by accepting the Request and passing of the Response objects. These elements are transient and are a domain model object as well as a combination of objects. The check of the incoming data, the calling of the correct function, the preparation of the incoming data and the initialising of the rendering process is also part of the task scope. The last named task, the rendering is started by assign an object to the View.

So far only Extbase has been used (Controller, Model), from now on Fluid takes over (View). Fluid is the template engine for displaying the content on the website. The template is a special file with placeholders which will be replaced by content. The Object Accessors access the variables of the content. Viewhelper which are simple PHP files provides functionality such as loops and Arrays assign hierarchical values to this helpers.

5.2.4.2 Extension Builder

The extension_builder[13] can be used to generate the domain model for an extension within a graphical user interface. This extension can be used to distinguish between entity and value object and the associations between the objects. It is recommended to install it within the Extension Manager, but only use it under TYPO3 v7.8.18 LTS. When using TYPO3 v8.7.x as productive system, it is recommended to create the files under 7.6.18 and install it afterwards.

5.3 Development Environment

The focus of the TYPO3 association is on the TYPO3 v8, and so this version was used as the mainframe for the development of the extension. The level of reliability was sufficient and given by the possible later time of productive use.

5.3.1 Installation

The operation system used for the installation was Ubuntu 16.04 and two version of TYPO3 were installed locally. Both of them TYPO3 v8.7.x and TYPO3 v7.6.18 were provided by Bitnami[14]. For the generation of the Domain Model the extension_builder installed on TYPO3 v7.6.18 was used. Afterwards, the extension was tested on TYPO3 v8.7.x. During development the version increases and the

[13] A proven way to build an extension by Domain Driven Design. https://extensions.typo3.org/extension/extension_builder/, accessed 28.09.2017

[14] Bitnami provides packaged applications for any platforms so that ready-to-run server applications can be used. https://bitnami.com/stack/typo3/installer, accessed 28.09.2017

final product was v.8.7.7. Finally, the bitnami system TYPO3 v8.7.7 was installed on Windows 10 to test the extension on another operation system.

5.3.2 Browser

The application was developed and tested by using different Browsers, mainly with Google Chrome version 61.x and higher. Mozilla Firefox and Opera were used as well, to verify functionality and correctness of the application in a Linux system. The browser Edge was used to test the application under Windows 10.

5.3.3 Integrated Development Environment and Version Control

The Integrated Development Environment (IDE) PhpStorm was used for programming[15]. Git[16] was used as the version control system. The development model was influenced by Vincent Driessen[17] The remote server was `https://git.tugraz.at`.

5.4 Modularization

During the phase of planning as well as the implementation, it was clear that the application can be divided into four modules. This program parts were development independently and linked by the editor or interfaces:

1. the content management application to generate and maintenance the content
2. a module for the input of user data
3. a module to verify and maintenance the conditions
4. a module for Open Badges issuing

These splitting combines certain use cases for the administrator, the lecturer, and the university and reflects the interactions between actors and system. In this paper only point one and two are displayed at the front end. All the settings for configuration and creation of the front end are made in the back end or by the files created in the integrated development environment.

A back end module was created to handle the verification of the input and the conditions and the specific settings for the Open Badges issue system. Another reason for this fragmentation is the opportunity to combine or recombine this parts with another system so that a final product can use or reuse same parts of the prototype. All the parts are connected to each other directly or indirectly by interfaces. A detailed explanation follows in the particular sections. The extension which handles three of this four functional parts is called OBOER (extension key: oboer): Open Badges for Open Educational Resources.

5.5 Front end Content

The front end is the presentation layer of a software. The basic structure of a website or more precisely the site map is constructed in TYPO3 back end by the page module where the pages such as home, the menu or other navigation elements are generated at first as template. The behaviour of these pages are configured using TypoScript.

[15]The IDE version for developing with PHP is PhpStorm. https://www.jetbrains.com/phpstorm/, accessed 29.09.2017

[16]https://git-scm.com/, accessed 29.09.2017

[17]An early model to use git in a comfortable and proven way. http://nvie.com/posts/a-successful-git-branching-model/, accessed 02.04.2017

5.5.1 Pages and Templates

The front end consists of pages and content. A template of a HTML-page is created by a simple click on the icon with a file symbol. Afterwards, by dragging and dropping of pages and linking to the page tree, the construct is growing. In figure 5.2 the blue boxed frame shows the page tree. For this simple example, the root element is Home. This page is the parent, and the following sites are the children or grand-children. On the right upper side are the two pages impressum and contact. On the left side of the front end is the main navigation with the three (four) children: Home (redirection page to the root) About, Lecturer and University. About is divided into Vision-Mission and Strategy, Lecturer into Metatdata and Further Training, and University into Further Training - Commit and Repository.

The connection between design and content is made in TYPO3 using TypoScript [Stöckl and Bongers, 2005]. This configuration language defines the behaviour of the elements. It connects the basic HTML file(s), the CSS file(s), the page tree, menu(s) and the content. This script does not only configure the user interface of the consumer it is also responsible for the the back end dashboard. The listing 5.1 shows the most significant parts of the TypoScript for the prototype. This files describes the origin and simple use for TYPO3 systems.

```
{
    page = PAGE
    page {
    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
    10 = TEMPLATE
    10.template = FILE
    10.template.file = fileadmin/maintemplate/home/template.html  # add
         the root home page
    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
    includeCSS{
    screen = fileadmin/maintemplate/home/style.css
    screen.title = display
    screen.media = screen
    }                                                            #
       include CSS file
    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
    LOGO = IMAGE
    LOGO.file = fileadmin/images/logo/logo_v1.2.png
    LOGO.file.height = 115                                        #
       add a logo
    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
    CENTER = CONTENT
    CENTER {
    table = tt_content
m    }                                            # load the created
   content
}
```

Listing 5.1: Parts of the TypoScript. The comments describe the effect of the code

5.5.2 Editorial Content

The primary task of a CMS is to create editorial content for the consumer. For this project, this means: create content to inform the public about mission, vision, strategy, and usage of the front end. The textual

creation is done in the TYPO3 back end. Editorial content is created as follows: First, the editor has to click on the Page module. Second, the Columns have to be selected from the drop-down menu called QuickEdit. Third, a selection of the area where the content should be located has to be done, and the icon with the side symbol and the label Content has to be clicked (see the yellow boxed frame in figure 5.2). In case a Text and Media content is selected a dashboard appears where the text can be edited (see figure 5.4). Finally, the modifications have to be saved. Otherwise, all the changes disappear. The content is stored in the database table tt_content. The website from an early stage of the development can be seen in figure 5.5.

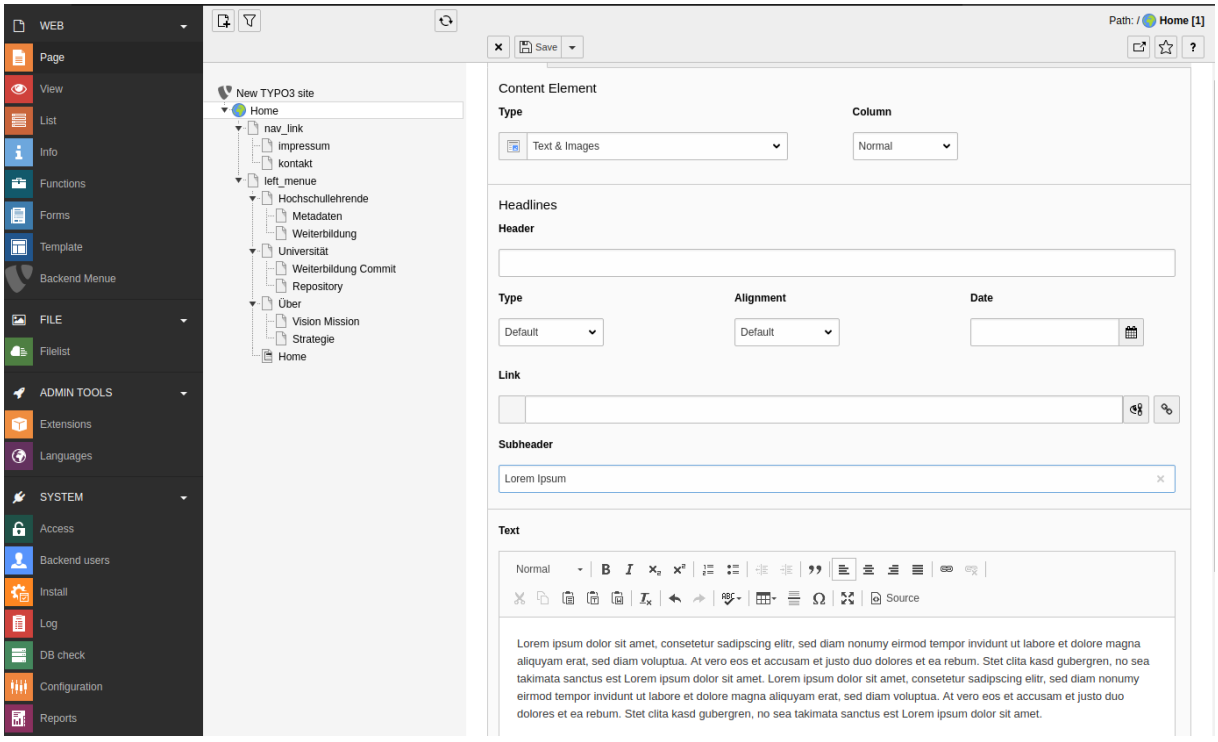


Figure 5.4: The backend dashboard creates the textual content by a (Robert Hafner CC-BY).

Figure 5.5: Screen-shot of the back end of the TYPO3 v8.7.7 (Robert Hafner CC-BY).

5.6 Data Input by Forms

Depending on the installation there are two approaches to get the data of the user groups into the system. For TYPO3 v 8.7.x the domain model and the new system extension Form offer a form to insert text. For the predecessor only the domain model is available.

5.6.1 Form by Domain Model

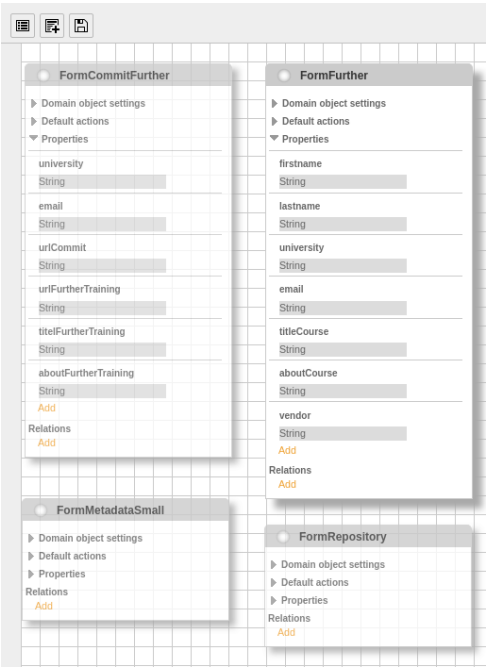
The domain model objects were created by the extension_builder, and the following section describes the usage of the builder. The creation of the domain model starts in the Domain Modelling view by dragging the icon with the label New Model Object and dropping it on a free space of the Canvas which occupies most of the dashboard. Afterwards, the properties are added by selecting a type (bool, string, integer etc.) and a name.

Depending on the needed information, different types and names were chosen for the various forms. For the lecturer first name, last name, university and email is mandatory. The first form to request the metadata of OER uses the attributes of the Dublin Core. The form which can be seen in figure 5.6a contains a reduced input field. Only the additional input fields Type, Format, URL of the repository is used to collect the OER metadata. The form of the further training is completed by the title, a short description and the name of the university which provides the course.

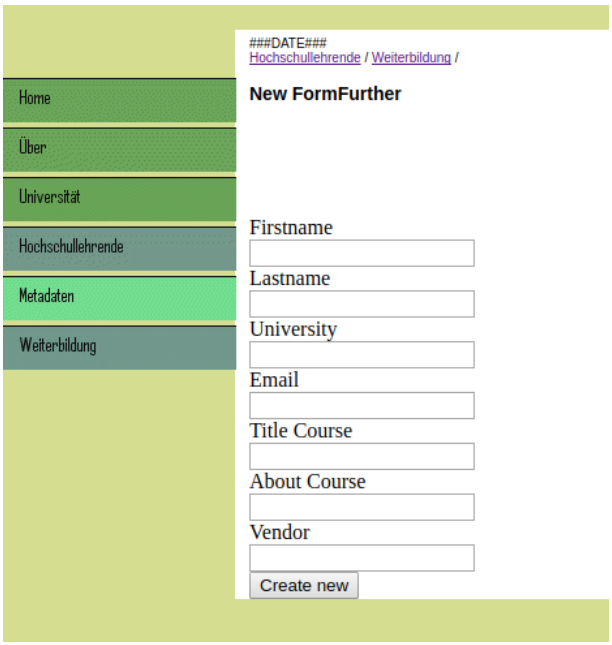
For the universities forms, name and email are mandatory and used for both forms. The form asks for URL of the commitment and the further training, the title and a description of the course. The form which can be used to get the data for the repository has two additional input fields, one can be filed with the URL and other with the name of the vendor of the archive.

It is recommended that the new object is set as root element and the available prototypes of the controller functions are set so that they were created automatically. A usage of this objects is only possible by creating a front end plugin and naming of the controller function name which have to be

used. In fact this means that the controller functions are written into the ext_tablesconf.php file. For the application four front end plugins were created but only three controller functions are used: new, create, confirm. The reason for this minimized usage of the controller functions is that the form should only be used to add new data not to list, update or delete them. This functions only were used by the administrator (see section 5.7). Finally, the data is stored into a database table which is named after the convention: tx_ "name of extension" and "name of the domain model object".



(a) These four elements are the domain model objects for the form which can be used by lecturer as well as the universities. The picture shows parts of the extension_builder dashboard.



(b) This is the view made by the domain model.

Figure 5.6: The beginning of a form starts with the modelling of the domain in the extension_builder. Finally, the properties can be seen in the view

5.6.2 EXT:Form

Since TYPO3 v8.5 the complete rewritten system extension EXT: form can be used to create forms by simple use of the Form Manager[18]. A significant advance of this new application is that editors can also create a form without knowledge of DDD or OOP. After starting the back end module Form, the first step to create a new form is by click one "add a button" with a plus. Afterwards, a pop-up appears, and it is necessary to insert a name for the new form. The dashboard of this system extension appears, and the form editor can be used to create and label different elements: text, text area, password, checkbox, radio button file upload et al. For all these elements names, placeholders, default value, a checkbox for required fields as well as a validator can be chosen. By default, the file which is used to store the form is a YAML [Ben-Kiki et al., 2009] and saved into the user_uploads directory which is also used to store all

[18]The TYPO3 Group Munich represented by Peter Kraume presents the rewritten extension EXT: form https://de.slideshare.net/pk77/frontend-formulare-in-typo3-8-lts, accessed 28.09.2017

the other uploaded files such as images or PDFs. There are several methods to store or redirect the input of the form by using a simple finisher. A persistence storage is created by the SaveToDatabase finisher which is defined within the YAML file of the form or by a finisher class. It is also possible to forward the input of the form to an email or another web server [Association, 2017a]. In contrast to the domain model, the upload of certification for lecturer and further training was implemented.

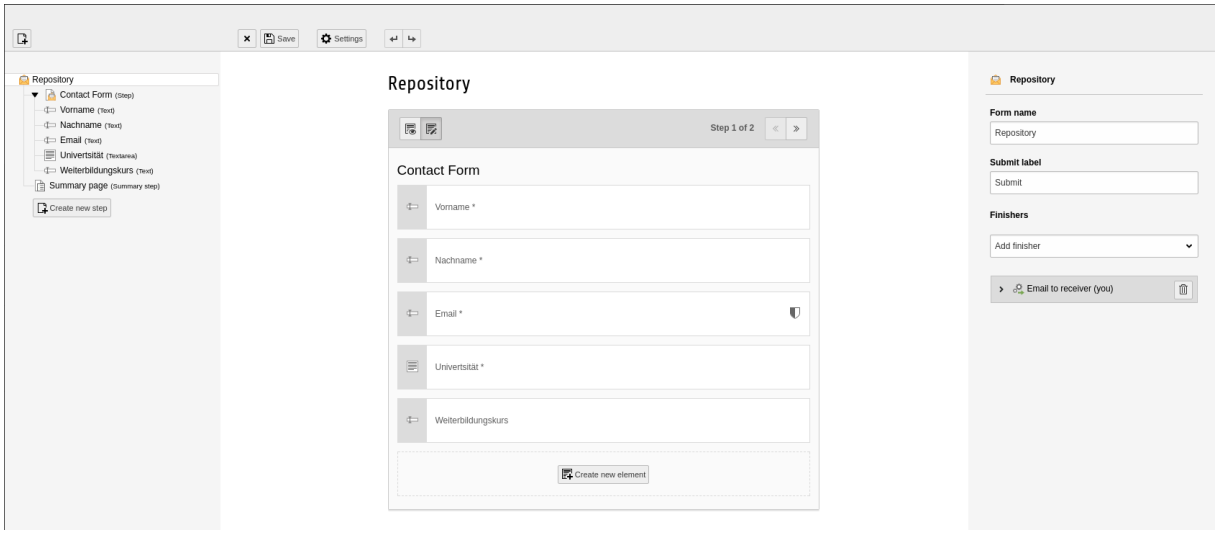


Figure 5.7: The rewritten system extension EXT:form has a new Form Editor which enables easy creation of forms by editors, integrators or developer (Robert Hafner CC-BY).

At this point all the use cases that involves the actors universities, lecturer, and consumer are implemented. The result of the efforts can be seen in chapter 6.

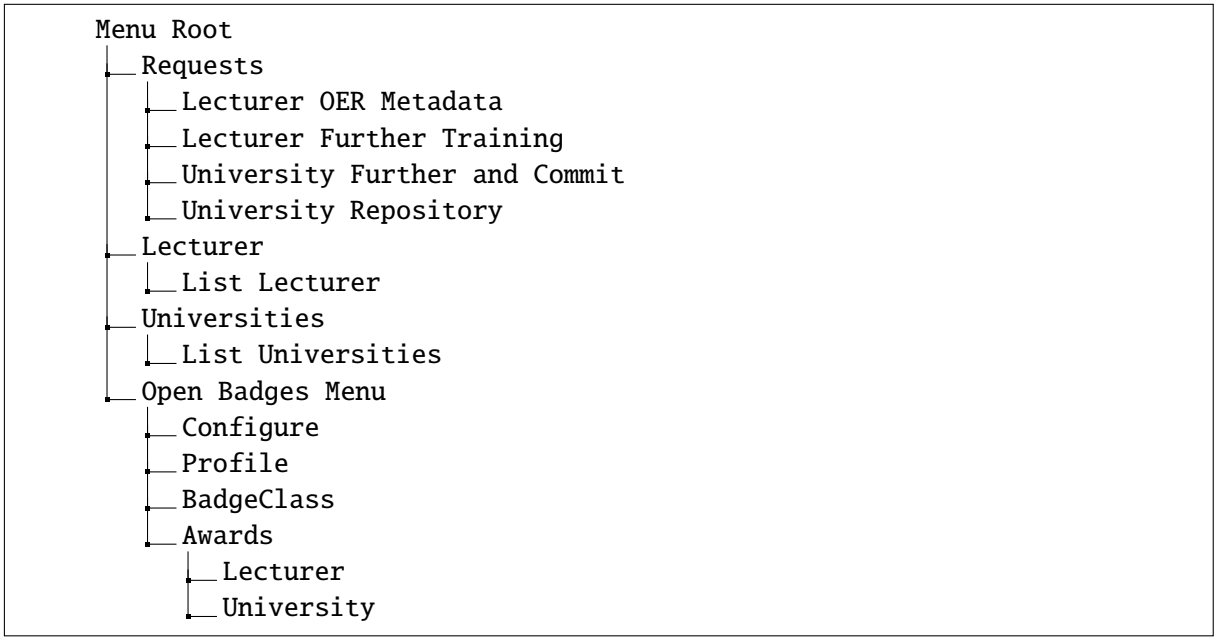
5.7 Back end View

This section presents the functionality and background of the back end module of OBOER. Only the software developer, the integrator and the editor of the Central Authority have the right to login into the CMS back end.

Similar to a front end plugin (see section 5.6.1), a back end module can be created by the extension_builder or by programming a new module. It works as the front end, by using the Extbase and Fluid. The registration of the module is done within the ext_tables.sql file, and the functions of the Controller can only be used unless they are declared in this file.

The menu of the back end module is divided into four main parts. Requests is the link to the four pages that handle the input of the potential Recipients. By clicking Lecturer, a list (table) of all the lecturer can be seen. For a list of all the educational institutions, the link "Universities" has to be clicked. The Open Badges Menu has additional sub-links: Configure, Profile, BadgeClass and Awards which is separated into Lecturer, and Universities.

Figure 5.8: The menu of the backend module as a page tree



5.8 Conditions and Data Input

This section describes how the software application and the primary editor handles the input of the universities and the lecturer, and stores and proves the condition to create a request for an Open Badge. The reason for dividing the application at this point is that the editor should also have the option to insert the data of the potential Recipient by hand.

5.8.1 Domain Model Objects for the Conditions

There are two domain model objects which are used to store and later on prove that the conditions for an Open Badge are met. Both of them are made by the extension_builder and can be seen in the figure 5.9. The model also describes the relation to the requests (application). The Institution (University) has a 1:n relation to a lecturer as well as a university can have n numbers applications. The Lecturer has an n:1 relation to University. This is also shown by the fact that the lecturer has only one email address. Additional, this person has n objects of Metadata but only one FurtherTraining object (1:1 relation).

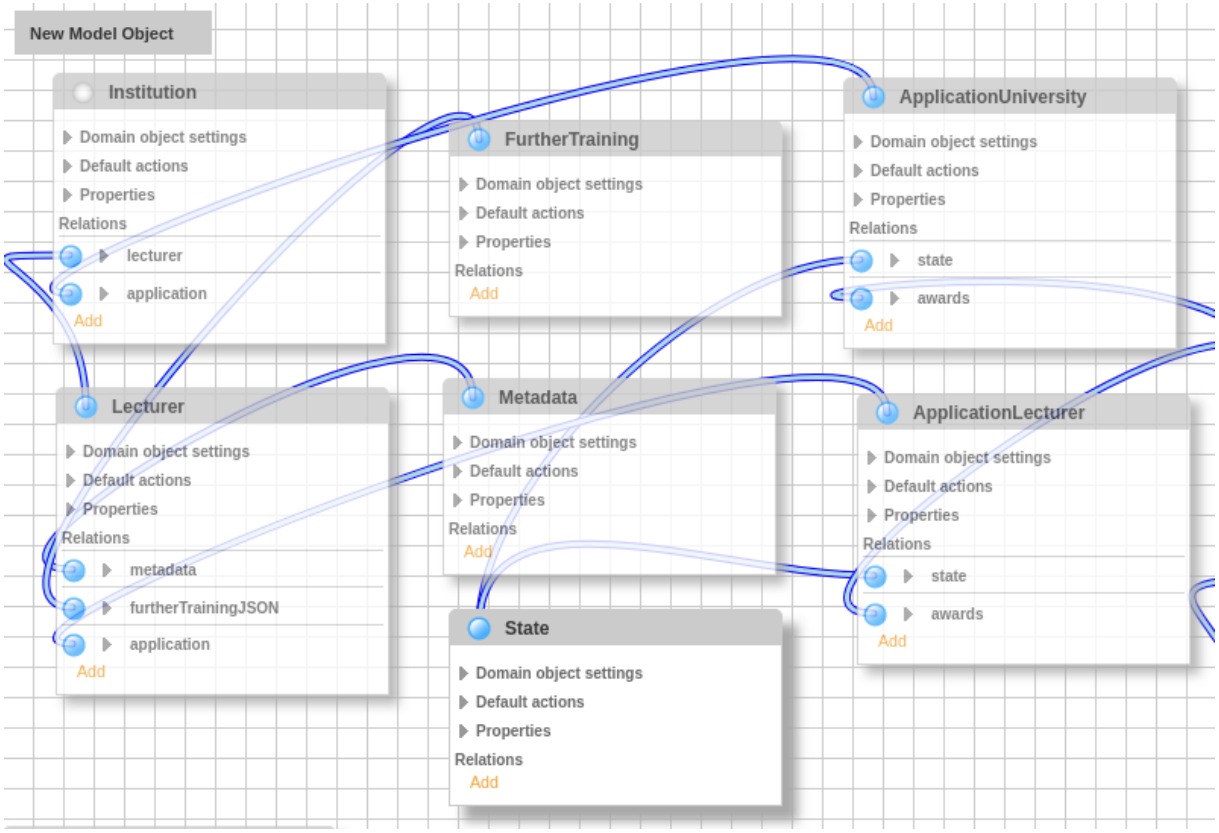


Figure 5.9: This is the domain model which stores and proves the conditions for a badge as well as the links to the request to get issued (Robert Hafner CC-BY).

5.8.2 Back end for the Input of the Lecturer

Same as the four properties of the forms as well as the domain model object of a lecturer contain first-name, lastname, email address and university. In addition, this object stores the information of numbers of OER as an integer and the metadata itself are stored in the object Metadata. This Metadata object has a n:1 relation to the lecturer, and at this point, the only property is the JSON object as a string. The extension_builder cannot store generically JSON, and so later on an array is constructed and filled with the data, and finally, this object is JSON encoded. The fact that a person has finished a further training for OER is stored as a boolean value in the domain model object. Additionally, the data about the further training is stored in a separate object. Same as the other metadata also this information is stored into a JSON-object. The reason that the information of the further training is not only a simple relation to a university is that it cannot be ruled out that other further training courses may also be credited.

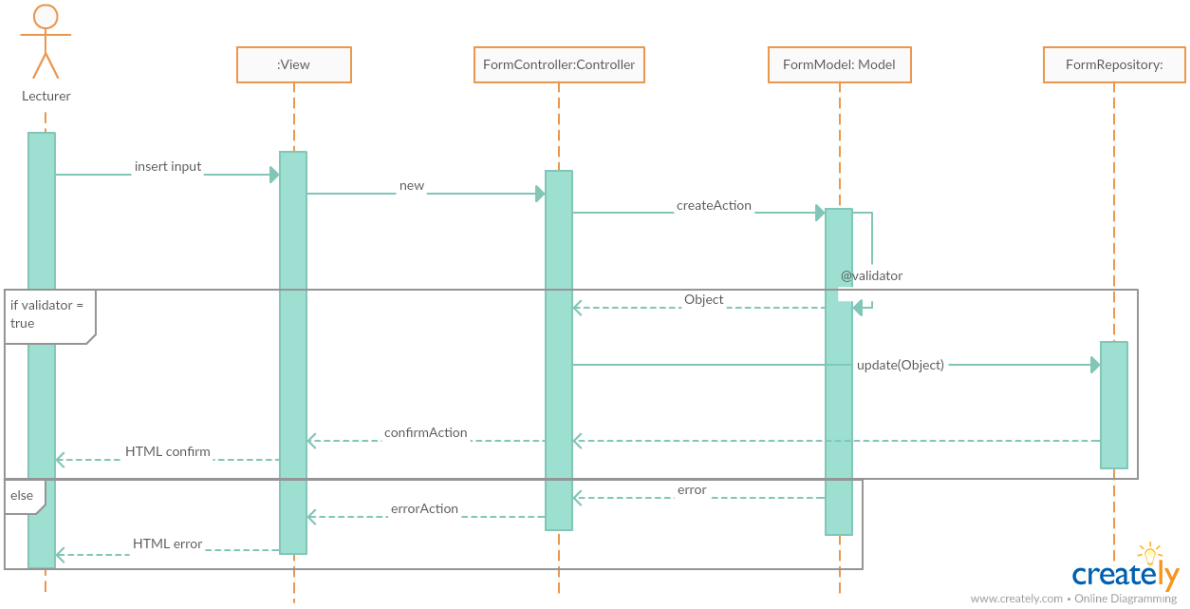


Figure 5.10: This is the sequence diagram of the lecturer inserting the data about the metadata as well as the further training (Robert Hafner CC-BY).

After the information is sent by the lecturer, it is the task of the editor to prove that the input is correct. It is the task of the application to check the identity input (firstname, lastname, email-address, employer) of the lecturer, in case there is a database entry about this person. If the properties do not exactly match the attribute value of the table, messages are sent to the editor, so that this person can react to guarantee the safety of the database. A simple list provides an overview of the data of the lecturer. In case the dataset is valid the editor accepts the input (see figure 5.11), and the lecturer will be updated (see first half of figure 5.13).

Figure 5.11: This is a single view of input for a further training (Robert Hafner CC-BY).

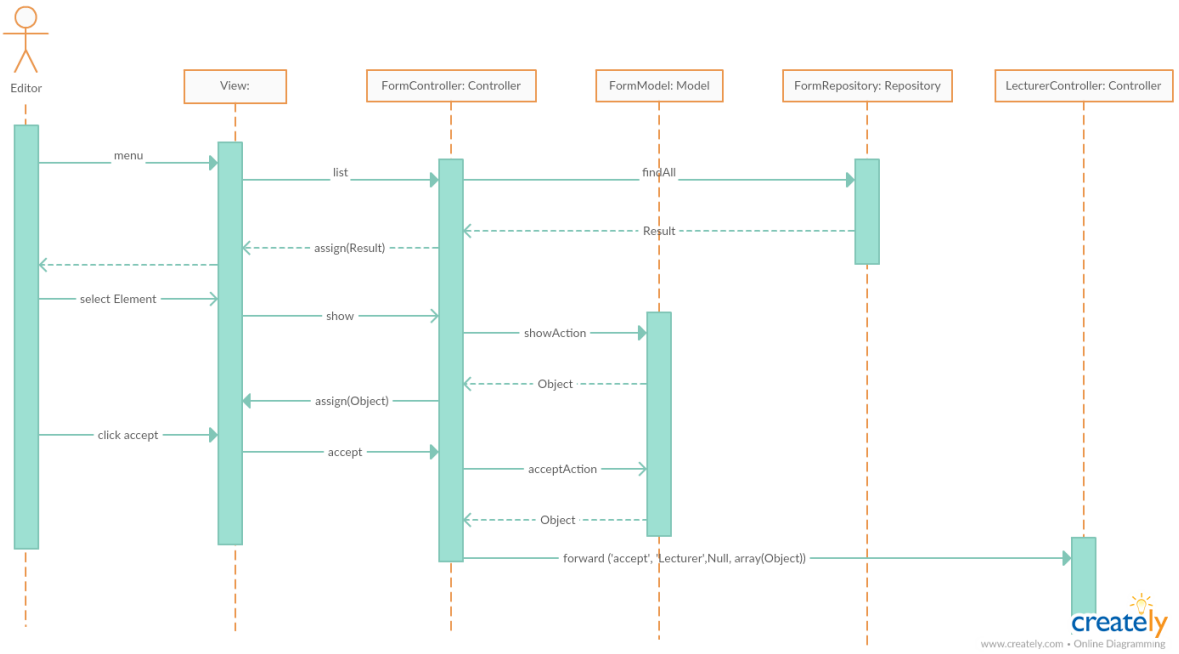


Figure 5.12: This is the sequence diagram of the editor which is controlling and later on accepting the announced data of the lecturer (Robert Hafner CC-BY).

5.8.3 Back end for the Input of Universities

The mandatory input of the university forms is the name and email address. This fact is also reflected in the domain model object of a university, and so the first two properties are also name and email address. Afterwards, the number of students can be stored as an integer. The numbers of the lecturers is the 1:n relation to the domain model object Lecturer. For the further training, the commit and the repository only boolean is used in each case. After the input of the university is send and reviewed, the tasks of the editor are:

1. To add the name of the university into the field so that only the official name is stored.
2. To add an email address into the field to define the identity of the Recipient.
3. To add the number of students into the field so that later on the condition for Open Badges v2.0 expired Part 2 can be checked.

5.8.4 Met the Conditions by Strategy Pattern

At this point of the workflow, the control of further actions is transferred from the editor to the application. Every change on the entities of the objects Lecturer and Universities by the editor has a direct effect on this module. If a new level has been reached an application for a badge starts as well as a reaction is needed if a part is not fulfilled any more. A module which is inspired by the strategy pattern[19] handles the verification of the conditions. The sequence diagram which shows the the interactions between the involved classes in case of the lecturer can be seen in figure 5.13. The approach works for the object Lecturer as well as the University.

After the editor has accepted the input of the lecturer (see the last sequence of figure 5.12) and the controller function accept of the Lecturer is called, the function serialise of the class *SerializeObject*is

[19]https://www.tutorialspoint.com/design_pattern/strategy_pattern.htm, accessed 20.10.2017

called, and the Form is passed as a parameter. After filtering and serialisation, two objects are returned by *SerializeObject*: an entity of a Lecturer as well as a simple Object of the additional data. In principle, both potential Recipients are treated equally at this point. The difference is that the update of the university is done manually and not by the system itself. Now a submodule to verify the conditions for Parts and the Level is used (see 4.2.3) so that an Application (request) for an Open Badge can be submitted to the next submodule, the Open Badge Issuer subsystem.

The class *StrategyCertification* is the interface to verify the level of a lecturer and a university. By bypassing the potential Recipient to the class *PartOne** to *PartTwo**, respectively *PartThree**[20] these classes tests if the objects fulfil the conditions by comparing the potential with the private attributes. These classes return true in case the conditions are fulfilled, and all these values are compared in various 2^n combinations (n numbers of Part classes) in the caller function. A switch case statement returns the Level class which matches the conditions of the potential Recipient or an error message in case no condition (e.g. only 1 or 2 OER metadata are named) for a Level matches. This Level class and Lecturer or University is now used as a parameter for an application.

The issue with the badges for the universities which can be expired has to be handled in this section of the code (see 4.2.3.2). It is recommended to us Cron[21] as a solution. This job scheduler can be used for tasks that appear repeatedly. This task should be an automatic reminder which is sent to the Recipient so that a reawarding of a badge of a lower level can be done soon enough.

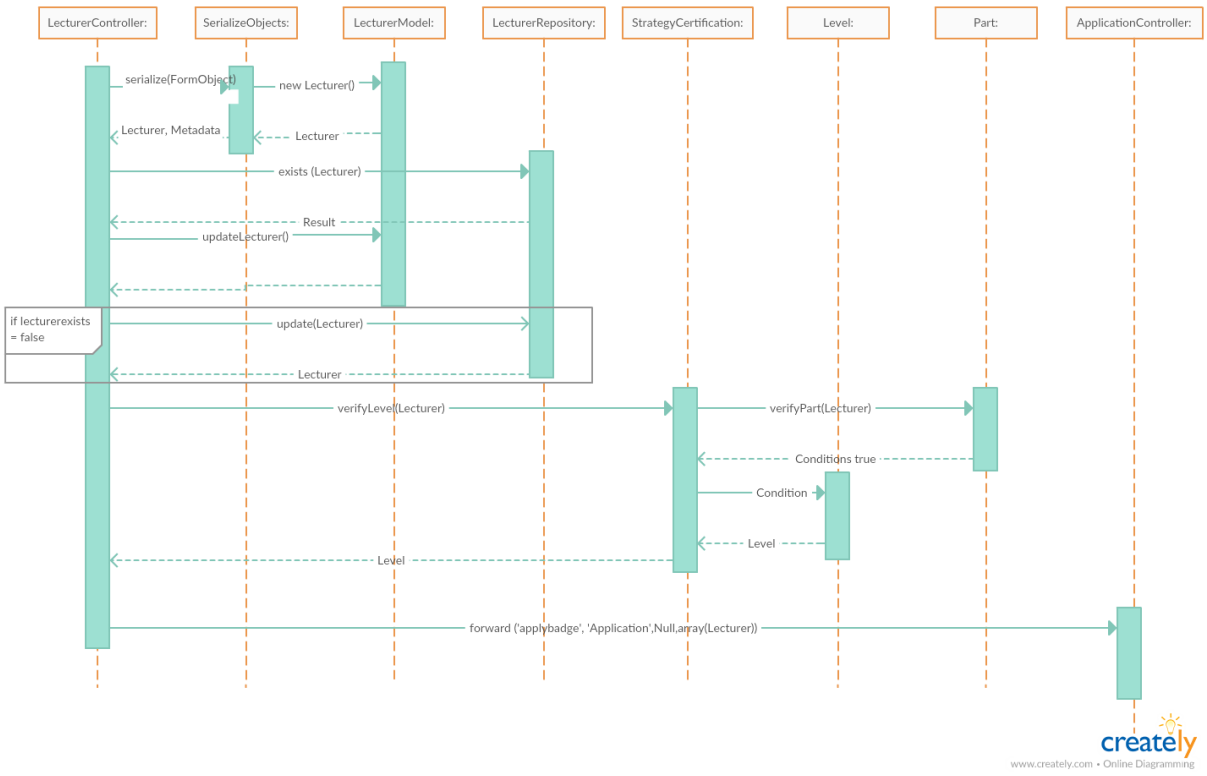


Figure 5.13: This is a sequence diagram that describes the interactions between the classes which are involved (Robert Hafner CC-BY).

At this phase of the project, only a developer can change the settings for the conditions by changing

[20] * can be replaced by Lecturer or University

[21] Job Scheduler for Linux systems. http://pubs.opengroup.org/onlinepubs/9699919799/utilities/crontab.html, accessed 20.10.2017

or inserting new Part or Level Classes.

5.9 Open Badges Issuer System

The issuer system is the central part of this software. The interface to the former part is the application of a lecturer or a university. The editor has still a chance to reject the application or to accept it. In case the editor agrees with the application, by default an award is created and send to the email address which is named.

5.9.1 Domain Model Issuer System

As described in the chapter 2 the issuing of badges is made within the Open Badges Ecosystem by the Issuer and by usage of the Open Badge Vocabulary. In this section, the data classes of the vocabulary were mapped to the domain model. Not all classes of the vocabulary are used in this phase of the project; only the mandatory classes are used. The Applications of the lecturer and universities can be seen in the figure 5.9 and the last sequences of the interaction with the system can be seen in the diagram 5.13. The next module of the entire domain model for the open badges issuer module can be seen in figure 5.14.

A short overview of the relation is described here: a lecturer or a university has 0 or n numbers of applications. If an application changes the state from send to approved the 1:1 relation exists between an Award and the Application. This Award is linked to only one BadgeClass. The BadgeClass is connected in a 1:1 relationship to the Profile which just exits one time in the application. The other relations are linked to Criteria and Image. The relation between the BadgeClass and the object Tags is 1:n. The RevocationList, as well as the CryptographicKey, is linked to the Profile. The List of the revoked badges is connected to the Award so that an m:n relation is used to store this annulled awards.

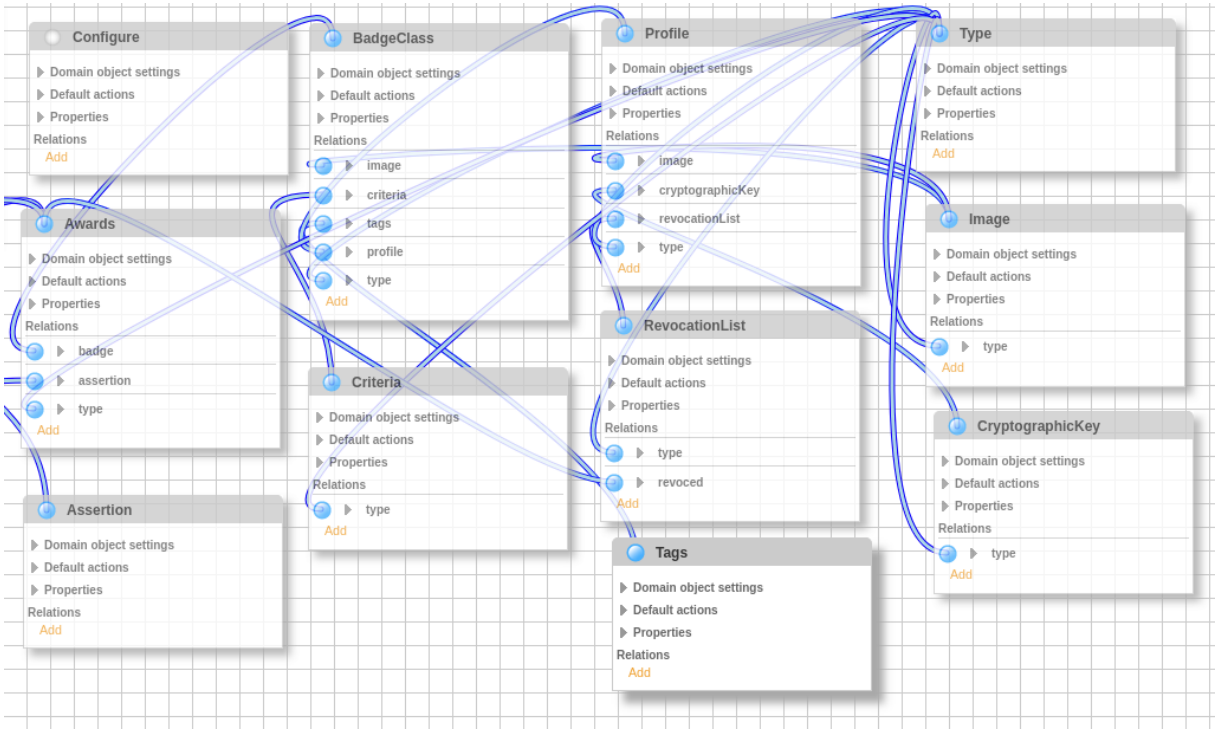


Figure 5.14: This is the Domain Model of the Open Badge Issuer system at phase 1 (Robert Hafner CC-BY).

5.9.2 Configuration

The configuration of the system is a task for the editor, the integrator, and the developer. The first step towards is to initialise the system in the menu Configuration. Currently, only the domain name has to be inserted so that the all the *ids* of the data classes can be concatenated correctly.

This steps are essential to start the system:

1. create a pair of RSA keys with openssl
2. set the values for the Configuration
3. set the values and initialize the Profile
4. set the values and initialize the BadgeClasses

Only, if all the steps are fulfilled, the application can be used to issue the badges. It is recommended to follow the order of activities cause the objects which are created contains linked data for the verification. For specific configurations which have to be done by the developer to control the application the file *configure.php* is used. In this file, the locations (directories), and titles for the images and the JSON-LD files, are stored. Further information can be found in the file as a comment.

Configuration of the System-RSA PK The RSA keys for the digital signature to sign and verify the assertions are created by openssl. There are two commands which are required to create the keys. The first command creates the private key *$ openssl genrsa - out private-key.pem 2048* and the other one extracts the public key from the private key. *$ openssl rsa -pubout -in private_key.pem -out public_key.pem*. Both keys are stored in the upload folder fileadmin\keys of TYPO3. Every time a signed assertion is created this private key is used and called within a function of the class *BadgeFactory* so that the JWS can be created. The productive system shall not store the private key within the CMS. It is recommended to store it in a certain directory of the operating system.

5.9.3 Publishing for Verification

The concept of linked data is essential for the Open Badges Ecosystem. This approach to verify and validate the awards and the organisation is enabled by files which are stored on the web-server. A request from Displayer and Backpack services to an URL which is inserted into an at least Assertions, BadgeClass or Profile starts or finish the process of verification. This requirement for this process is separated into two parts. First, the extension provides functionality by different classes to create JSON files and second the requests from the Displayer or the Backpacker have to replied by storage and usage from extensions of other vendors.

JSON Objects for Data Classes The Domain model objects for the Open Badge Issuer service stores the prosperities which have to be inserted by the editor and automatic by the system. In details, there are differences between the objects, Assertion, BadgesClass, Profile, CryptographicKey, and RevocationList. However, they have in common that the classes with the suffix *Factory (replace * by the name of the data classes named before) creates the objects which are needed. All of the essential objects (values) are passed as a parameter to the factories and an object ready to use is returned. In detail, an array which is corresponding to the data classes is created, and this object is returned. These functions are called in the controller classes after an act of confirmation is done (mainly by clicking a button). The function publish of the singleton class *PublishJson* passes the array as a parameter, creates a JSON file and currently stores the object on the web-server.

Request for Verification The URLs of TYPO3 systems are the so-called non-speaking identifier. That means that the locators are not classical readable links with a .html suffix. An example of such a non-speaking link is */index.php?id=12345&tx_news[news_id]=12345*. However, links which are needed for linked data should be readable.

Realurl[22] must installed, and TYPO3 have to be configured as well as the Apache 2 server so that the redirection is working and files with with a .html, .json, or .png suffix can be found.

5.9.4 Profile

A Profile is a link menu item and a domain model object to insert and store particular values for the data class. After entering the required input, the object is stored persistently in a Repository, created, and uploaded as a JSON file. The mandatory values for properties id, type as well as the optional properties name, url, email, description are used. In the single view of Profile (see figure 5.15) only the optional properties can be viewed and edited. The reason is that the system automatically handles this requirement and should not be modified by the editor so that a manipulation of this mandatory properties can be avoided. However, it is not redundant to insert the domain into the configure as well as in the Profile. The URL of the profile can also be linked to a social media site if a differentiated communication strategy is applied.

Figure 5.15: The single view of the Profile contains certain values of the domain model object (Robert Hafner CC-BY).

5.9.5 BadgeClass

The BadgeClass is a link menu item as well as domain model object and used to edit and store the five metadata classes. In the prototype, the mandatory properties id, type, image, criteria are used, but inserted by the application calling the file *configure.php* and not from the editor. The task of the editor is the inserting of the mandatory properties name and description and the optional tags.

The png-files for the badges are linked to the BadgeClass and the Assertions also, but only stored once in the fileadmin\image directory. The reference URL for an image request is currently composed of the domain the directory fileadmin/user_upload \images and the file name. As an example identifier is the domain *http://www.oboer.org/* plus *fileadmin/user_upload/images* plus the filename *lecturer-level-1.png*. The five images for the badges can be seen in figure 5.16

[22] RealUrl is one of the most (source `https://www.liquidlight.co.uk/blog/article/essential-free-extensions-in-typo3/, accessed 05.10.2017` or `http://www.nitsan.in/blog/post/top-most-used-typo3-extensions-from-ter/, accessed 05.10.2017` useful extension when working with TYPO3. https://extensions.typo3.org/extension/realurl/, accessed 05.10.2017

Figure 5.16: This five png-files are used as image for the badges. Every single file is linked to a BadgeClass and so finally to a awarded badges (Robert Hafner CC-BY).

CryptographicKey This domain model object of this data class mainly consists of a string which contains the public key. There is a redundancy cause the key is also stored in the fileadmin directory.

5.9.6 The Awarding Process

It is quite simple to award a lecturer or a university with a badge. After the verification of the correct level and Recipient, only a simple click on the accept button in the single view of the application has to be done. The application is moving form the list and inserted into the list of the Awards. The view of the application of a lecturer can be seen in the figure 5.17.

Figure 5.17: An application for an Open Badge Lecturer Level 1 (Robert Hafner CC-BY).

5.9.7 Awards-Assertion

Contrary to the other linked data classes the decision to award signed badges, caused that these objects do not have to be published on the server. The *BadgeFactory* did the task to create an unsigned badge in the form of an array. The class *SignBadge* does the digital signature.

Unlike the other data classes which have a URI in the id, a signed Assertion is identified by a UUID. This identifier is currently generated by the PHP uniqid function, and openssl_random_pseudo_bytes () is used to generate a more secure token. The value for the issuedOn is set at the moment the editor is accepting the application. The badges for the university can expire, and so these awards have an additional attribute expiresOn which is part of all award objects (this property is not used for the badges of lecturer).

For the generation of the *IdentityObject* the property type and hashed are constant settings within the *BadgeFactory*. Only the salt value is stored in the database as a property of the Awards cause this value is always created anew. The digestion is done by the sha256 hash function.

A short explanation how a JWS object is created can be found in section 2.6.1. In the implementation to create a signed Assertion the JavaScript Object Signing and Encryption JOSE PHP library[23] from ritou was used as a template so that the requirements can be met. Further detail can be found in the comment of the file *SignBadge.php*.

The domain model object Assertion is only used to store the JWS-file as a backup. The baking of the badges is done by the class *BadgeBaking*. As arguments, the JWS file and the image is taken. As defined in the standard the text of JWS file is inserted into the image after the keyword *openbadges*.

By the decision of the working group to award also badges that can be expired two user interfaces for the Recipients are made. First for the lecturer, second for the universities. The view Lecturer presents the UUID, firstname, lastname, email, and the date the badge is awarded. In contrast, the university view presents the UUID, the name, the email, the date the badge was issued (label issuedOn), and the date the award will expire (label expires) and also enables a re-awarding by a simple click on the reaward button. The single view of an award can be seen in figure 5.18.

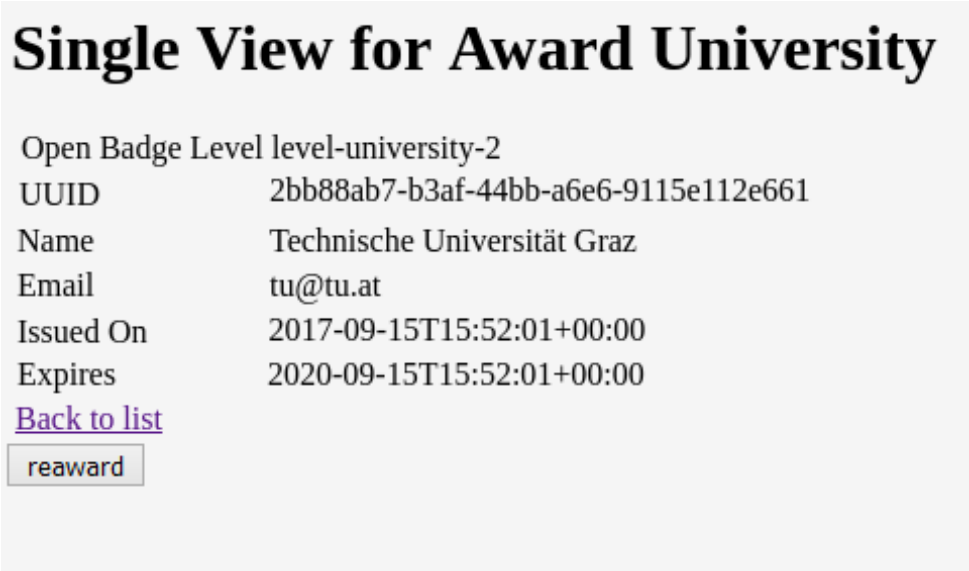


Figure 5.18: An application for an Open Badge Lecturer-Level-1 (Robert Hafner CC-BY).

5.9.8 Revocation

The revocation on a Badge is quite simple. A click on the edit link in the list of the Awards and a user interface appears. Now, by use of the checkbox revoke, and a click on the save-button revokes the badge. An update of the RevocationList is done by concatenating the UUID into the file. Just in case a badge is revoked the entry is not deleted, but the controller function list does not display the database tuples for this entry anymore. This structure enables a re-revocation of an award.

[23] https://github.com/ritou/php-Akita_JOSE, accessed 05.10.2017

Chapter 6

Reflections

In this chapter, a final review of the CMS, the concept, the implementation, the non-functional requirements which have to be considered are presented, and alternatives are shown so that a decision can be made to accept or reject TYPO3 as an issuer system for Open Badges.

6.1 Lessons Learned

For the further development this chapter is a milestone. All the aspects of the development are collected in this chapter and with some points in the section 7.2 a decisions can be made.

6.1.1 Concept

During the writing of the concept paper, the author of this thesis was involved [Ebner et al., 2017]. However, the focus of the working group was to find a compromise for all the stakeholders on a macroscopic point of view. As a result of this process, the implementation of software has some inconsistencies in the business rules and so assumptions were made by the developer. Further on, it is inevitable that the workflow of the application has to be revised in close cooperation with the client.

6.1.1.1 Business Rules

The usage of business rules in combination with the creation of the domain model is essential. The presentation of the business rules in the form of an Action Rule is an interesting and satisfying solution for a developer. The great advantage is the deterministic and abstraction level for verifying the behaviour of the software application. However, for a client, this kind of description is too technical, and can only be presented with an explanation.

6.1.2 TYPO3

The learning curve of TYPO3 is quite high. It is not recommended to start such an application (extension) without experience based on projects made before. It seems to be a framework with a lot of opportunities as well as obstacles. The documentation is detailed but also confusing. In case there are errors or questions the numbers of internet forums, and the finding of solutions, as well as the numbers and quality of answers, is patchy.

The result of the policy to create extensions is that reasonable extensions are not part of the core and so the developing and evolving is made by private persons or companies which are interested to perpetuate their business model. So there is no guarantee that a needed extension is further developed or

maintained. As an example, the extension_builder (still working only under TYPO3 v7.6.18) and the RealUrl have to be named. It seems to be a design decision to transfer essential functionality from the core into the hand of the community as an extension. It seems to be a trend to insert functionality into CMS systems so that they can be used for use cases they are not developed originally[1].

This concept worked well in case only a CMS system and the core functions are needed for the project. Reference projects such as the website of the Graz University of Technology[2], Technical University of Berlin[3] or the Deutsche Fußball-Bund[4] send a clear signal and also presents the strength of TYPO3.

6.1.2.1 Implementation

Neither the implementation of the system was challenging. The most significant tasks were to understand how TYPO3 is build up and working, the reading of TYPO3 developer book [Rau et al., 2013], the search for examples of other developers and to get answers in internet forms.

The extension_builder is an excellent tool for beginners as well as for experienced users. However, there are also problems: the relations were not displayed correctly, and the system had issues to store changes. If great changes are made in the database schema or tables, these changes were not or not correctly adopted. These modifications have to be removed by hand or by the database wizard. If objects were changed entirely or new objects with the same name were created the system does not clean the former entries completely, and so errors occur. The result of an increasing complexity of the domain model was an observable slow ability within the interactions on the dashboard. It was not clear what caused this behaviour.

The only approach to implement an issuer system for Open Badges for TYPO3 or to to be more accurate for Flow and Neos[5] was made 2014 by a German company[6].The project is still alpha. It seems to be that the task is no longer being pursued. Existing systems have been developed with other programming language as well as frameworks (see section 6.2).

6.1.2.2 Editorial Content

The functionality and usability to create content for the consumer guarantee the system on a satisfied level. The reason, therefore, is that this is the primary task of a CMS.

6.1.2.3 Data Input

Just in case the system extension EXT: Form is used to create forms, and so that the input of the potential Recipient can be inserting into the system, it will be necessary to develop a wrapper class so that the data can be stored persistence into the database. For the editor, it is an advantage to create a form without contacting the developer.

However, the frequency of error messages indicates that the extension is still work in progress. The efforts to get this extension running was finally stopped. Anyway, this extension is very promising.

The creation of forms as a domain model object should only be made for small forms. In case a complex form is selected (LOM), it is recommended to create a simple form without relations.

[1] 7907 extensions can be found for Joomla, without some to issuing Open Badges. https://extensions.joomla.org/, accessed 15.10.2017

[2] https://www.tugraz.at/home/, accessed 18.10.2017

[3] http://www.tu-berlin.de/menue/home/, accessed 18.10.2017

[4] https://www.dfb.de/index/, accessed 18.10.2017

[5] Web framework to develop mid or large-scaled web applications, https://flow.neos.io/, accessed 18.20.2017

[6] The TYPO3 Flow is a web application framework to develop mid- or large scaled web applications. https://github.com/networkteam/Networkteam.OpenBadges, accessed 10.10.2017

6.1.2.4 Conditions

The validation of the conditions made by the domain model objects lecturer respectively university works well. As root element to link the applications and further on the awards is sustained and can be used later on. The strategy pattern to verify the conditions also works well and can be used further on. The issue to handle different timeframes at badges which build upon each other can be controlled by informing the Recipient timely so that an update can be made and the levels keep valid.

6.1.2.5 Open Badges Issuer System

The Open Badges issuing software in TYPO3 does not fulfil the hole requirements of an Open Badges Issue service. The process of verification is not done in a way without obstacles. In general, the MVC pattern is a great idea. However, the implementation as an extension in the CMS is not practical without the extension_builder but too cumbersome to be handled within an IDE. Such growing systems sometimes have the claim to enable everything.

Publishing of Data for Verification The most significant disadvantage of this system is that an extension is needed to publish readable links. For this kind of requirement an out-of-the-box solution without the goodwill of a third party is preferable. The usage of the realurl caused same significant errors within the TypoScript so that only the text was presented without effects of CSS.

6.1.3 Non-functional Requirements

The definition of the behaviour of a system is as important as what a system should do. So these no-functional requirements about the CMS has to be discussed in more detail. It is recommended that a professional TYPO3-hoster should be used for a productive system and most of the following points can be used as part of the service-level-agreement.

6.1.3.1 Portability

As described in the section 5.3.1, TYPO3 can installed on Unix-systems, Windows, and Mac OS. TYPO3 is running not only on an Apache HTTP server also a NGINX server can be used. As well as the server even clients which are running on different devices and operating systems can be used. The only requirement is a browser. Responsive design[7] is currently standard for creating websites so that different devices of clients can be used. TYPO3 provides various options to install responsive design.

6.1.3.2 Reliability

TYPO3 is managed by the TYPO3 Association[8]. In cooperation with the Association, adequate numbers of private supporters, as well as companies (e.g. Mittwald[9]) guarantees current and future development.

Currently, TYPO3 v8.7.x is in regular maintenance, and the priority bug fixing is guaranteed until March 2020 (extended support is only planned for March 2022). The support for the previous version, v7, is ending at December 2018. TYPO3 v9 is in the Sprint Phase and will be published in April 2018. This roadmap shows that the support for the CMS is guaranteed for a sufficient timeframe. However, the support for the needed extensions is not secured, so that a further development and hence the usage cannot be ensured (see section 5.2).

[7] Another view of this topic can be read here. https://www.mobile-zeitgeist.com/warum-responsive-webdesign-schrott-ist/, accessed 15.10.2017

[8] https://typo3.org/association/, accessed 20.10.2017

[9] https://www.mittwald.de/, accessed 20.10.2017

A great advantage of TYPO3 is the systemic immanence of coding standard, unit tests, and configuration management. At least for the last point, a stunning number of settings is possible. The model driving engineering (software model) by creating of domain models is system immanent. Some software metrics can be found on the website of the organisation[10]. Another source that compares projects of companies and FOSS attested TYPO3 a high quality over all the processes started by the planning (architecture), implementation, testing and releasing. [Bergmann and Priebsch, 2013].

Information about such things as Mean Time to Failure (MTTF) or Mean Time to Recovery (MTTR) was not available.

6.1.3.3 Efficiency

TYPO3 is a system which provides developers and integrators with a lot of opportunities to increase the performance. Starting with the development and also installation of an extension, configuration by TypoScript and other tools.

Depending on the number of page views (less than 150,000) it is recommended to use the service of specific providers, otherwise, a local server can be used [Meyer and Helmich, 2016]. The expecting frequency of usage by the clients[11] in relation to performance, response time, processing time, query and reporting time are included in the consideration. However, depending on the selection of the TYPO3 version as well as other parameters such as hardware or provider a detailed planning should be done at first in phase 2.

6.1.3.4 Usability

The primary focus was, how TYPO3 could handle the requirements of an Open Badges issue system and not the usability for the editor or the consumer. The blog[12] from the Netzialisten describes the main problems of editors in TYPO3: information overload, too many icons, one dashboard for all, error management, get in contact with the client. However, another demand for this section is localisation (internationalisation) which can be made without significant effort. Another critical point is the accessibility of the CMS. An article from 2008 showed an example and proved their functioning [Arens, 2008].

6.1.3.5 Dependability

The dependability of a software application is related to different aspects such as availability, recoverability, robustness and security.

Availability The availability of a TYPO3 CMS depends on the selected provider of services as well as the hardware. Hours of operation, maintenance times etc. are part of the service-level-agreement.

Recoverability The recovery process of TYPO3 is quite simple. In TYPO3 a backup is made by dumping the database and copying of specific numbers of directories. A new installation and the import of the data is done in a manageable time frame. However, the disaster recovery so that the business continuity can be guaranteed can only be made by the integrator of the service provider

[10] The TYPO3 core team presents issues tracker etc. https://forge.typo3.org/projects/team-metrics, accessed 15.10.2017

[11] 36,495 is the number of lecturer and a scientific person in the year 2015 at Austrian universities. source: uni:data https://oravm13.noc-science.at/apex/f?p=103:6:::NO::P6_OPEN:N, accessed 20.10.2017

[12] Already old but still up to date. https://die-netzialisten.de/allgemein/5-usability-probleme-im-typo3-backend/, accessed 15.10.2017

Robustness The ability of the system to resist change without a adapting is initial a stable configuration. Again the reference to the service provider, and the service-level-agreement. Within this section also the archivability should be mentioned and so a study attested only mediocre for TYPO3 [Banos and Manolopoulos, 2015].

Security A comprehensive study on the CMS was conducted in 2013 and concluded that three aspects are important for a safe operation of a CMS [Breitenstrom et al., 2013]:

1. security of the used software
2. appropriate configuration
3. appropriate system management (including patch management)

The following was highlighted for TYPO3. The structure and information content of vulnerabilities in TYPO3 as well as Drupal, Joomla!, and Plone are exemplary, web-services were not secured after WS-Security[13] and the documentation of safety-relevant settings on the application TYPO3 can be improved.

Another source [Grothaus, 2016] noted that the vast developer community discovers early security problems, the TYPO3 Security Team guarantees a high-security standard and the TYPO3 core developer have been established a framework to sensitise the developer community to security concerns.

6.1.4 Costs

TYPO3 is a FOSS, and so there are no license charges. However, the system has to be developed as well as maintained. A detailed list of costs based on the prototype for the development as well as further costs for maintenance will be the first activity of phase 2 after a more detailed selecting.

6.2 Alternatives

As described in the section 2.5 there are three ways to issue Open Badges. Service by a vendor, installation of an application or development with other CMS or as web application.

6.2.1 Service

Credly offers a platform to create, issue and manage badges, and the recipients can display and share the badges to other systems. The service costs depends on the numbers of earners.

Open Badge Factory[14] is a cloud-based service to design, create, manage and to issue badges. There are plugins to other systems like Moodle or WordPress which can be found on Github, but also developed for another third party. The prices depend on the numbers of badges which will be issued as well as selected features (e.g. reports, helpdesk, user accounts) started by 0 €, up to 600 €. The service is linked to the Open Badge Passport (Backpack).

6.2.2 Installation

Badgr Server [15] is a web application for issuers, earners, and consumers written in python. Application Program Interfaces (API) can be used to enable the functionality of the issuer service.

Moodle (see 2.5) is an learning management system that issues Open Badges by default.

[13] https://www.w3.org/2002/ws/, accessed 20.10.2017

[14] https://openbadgefactory.com/, accessed 20.10.2017

[15] An open source alternative to install on the linux system such as Ubunutu or CentOS https://github.com/concentricsky/badgr-server

6.2.3 New Development

In case the development of a new system should be considerate another CMS, and web frameworks are alternatives.

As an alternative CMS, several providers can be considered: Wordpress, Joomla!, and Drupal. All of them provides the opportunity to extend the functionality by extensions respectively plugins. An Open Badge plugin for Wordpress is available[16]. A comparative analysis with other CMS shows most significant differences [Mirdha et al., 2014]. TYPO3 had advantages in the area of performance and data management.

The complete new development with a web application framework that supports the MVC pattern is an also an alternative. Three systems can be selected in the shortlist:

- ASP.NET: pattern-based web application framework[17] to build dynamic websites[18]. TDD, OOP, MVC is a standard feature.
- Django: Is a web framwork[19] written in Python. TDD, OOP, MVC is a standard feature.
- Laravel: Open source web framework[20] written in PHP. MVC framework with built-in support for authentication and authorization.
- Spring Framework: FOSS application framework[21] written in Java. Several modules provide a variety of services such as MVC, Data access, aspect-oriented programming, and TDD.
- Zend Framework: Open source web application framework[22] written in PHP 7. MVC, OOP, and TDD are standard features.

[16] Does not create or manage the badges itself is only an interface to BadgeOS. https://de.wordpress.org/plugins/badgeos/, accessed 20.10.2017

[17] https://www.asp.net, accessed 20.10.2017

[18] A list of examples can be seen here. https://www.danylkoweb.com/Blog/top-10-websites-written-using-aspnet-mvc-JK, accessed 20.10.2017

[19] https://www.djangoproject.com/, accessed 20.10.2017

[20] https://laravel.com/, accessed 20.10.2017

[21] https://spring.io/, accessed 20.10.2017

[22] https://framework.zend.com/, accessed 20.10.2017

Chapter 7

Conclusions

In this thesis, a realised concept to certificate lecturer and Austrian universities for creating and supporting of Open Educational Resources with Open badges has been presented. The history of badges, the usage in entertainment and education, the Open Badges Ecosystem, especially their vocabulary of the Mozilla Foundation have been presented in chapter 2. Definitions, classification and standards of metadata for Open Educational Resources have been presented in chapter 3. Chapter 4 have been described the concept of fnm-austria for a certification of lecturer and Austrian university in the area of Open Educational Resources. A project was developed from the specifications of the concept to identify stakeholders, user groups, business rules, non-objectives, use cases. The Content Management System TYPO3 and the implementation of a prototype to issue Open Badges has been presented in the chapter 5. An reflection with lessons learned and a detailed reflection of the non-functional requirements have been presented in chapter 6. There is no evaluation of the non-functional requirements for TYPO3 at this point and it is referred to section 6.1.3.

7.1 Discussion

A prototype was developed, but not tested in use. In this phase, it should be found out whether TYPO3 is the right choice for an issuer system. During the implementation, particular factors come abroad, and a lot of parts of this work can be reused. The primary focus to decide whether TYPO3 is the right choice for an issuer system should depend on how well the CMS handles the four different workspaces:

1. content creation and presentation
2. form user input
3. checking the conditions for a badge
4. allocation of open badges

At point one: The presentation of content is the primary task of a CMS and TYPO3 is excellent. Uploading images or entering text using the What You See Is What You Get (WYSIWYG) editor works flawlessly and relatively effortlessly. Training for the editors may be necessary. Nevertheless, this task is easily accomplished. Of course, the question arises here how often this feature is needed and it must be said: rather rarely. A half point out of one for TYPO3.

At point two Entering user data using forms is a simple task and TYPO3 does a good job. The new approach of EXT: form is a good idea and should be pursued further. However, some corrections are still

necessary, and certain adjustments have to be made. The use of domain model objects seems to be a bit overhead, especially if the objects are to be mapped relationally in databases and above all, a metadata standard like LOM is used. It is recommended to use EXT: form and save the information as a JSON file to the database using databasefinisher, regardless of the fact that this system extension needs some time to develop. Of course, the question arises here how often this feature is needed by the editor and it must be said: rather rarely. A half point out of one for TYPO3

At point three: Checking whether a part and thus a level for a badge is fulfilled is a module that is not specifically dependent on TYPO3. It is important that the developer is experienced or examples can be found, and an IDE is supporting the developing process. Of course, the question arises here how often this feature is needed and it must be said: rather rarely. Only important for the developer or if the conditions will be changed and so also the editor can be involved. A half point out of one for TYPO3, cause it does not depend on the CMS.

At point four: The usage of TYPO3 leads to severe restrictions. One is that Extbase and Fluid must be used. Fluid could be extended by using JavaScript. However, the use of this products is therefore a limitation for developers. In addition, there are also specifications by the CMS, such as the use of extensions that have to be installed, and are developed by third parties. The task of extensions such as realurl can be done by other frameworks simply by calling a function or editing of a configuration file. This concept seems to be an overhead for the project. For the editor this most important part, the configuration and usage of the Open Badges Issuer system should be simple and clear. It seems to be an overhead to login into the TYPO3 back end with all the functionality. Admittedly, the access management can be restricted by the integrator so that only the most necessary information is displayed. During the planning, a front end plugin for the editor was discussed. However, this idea was abandoned by the argument that there is an existing back end and already a login. Of course, the question arises here how often this feature is needed, and it must be said: often. This is the primary task for this project and the most elaborate one. A half point out of one for TYPO3, cause depends on the CMS. So finally there is 2 out of 4.

However, there is still the question what is needed that an Open Badge certification will be a success. Gamification can be used. A login and a dashboard will be necessary to see the individual position and also the next goal respectively the next badge. The prototype only provides a plugin without a login so that there are no obstacles to enter metadata. A compromise is needed. A well-considered ranking should be used so that a healthy competitive situation can arise. A pilot project should also be considered. The acceptance increases in the group of Recipients and not from outside. For this reason, the use of advertising or decree from administration should be handled with care. Problems can arise by the fact that conventional institutions have issues to setting up an OBE, on the other side a success factor for an issuer service is that no other certification systems exist. Overall, in the next phase the decision to design and provide an Open Badge Issuer service needs a meeting of all necessary stakeholders to talk about the requirements that the certification with Open Badges will be a success.

In the end, the technical and conceptual requirements that a certification with Open Badges will be a success are too specific so that TYPO3 cannot meet them in a way that other systems (Keep it simple, stupid) can make better.

In the medium term the future for web applications is a high concentration on JavaScript frameworks (e.g jQuery[1], D3[2]) for the front end, with a back end such as Django, Zend Framework, Spring Framework or .ASPnet. However, all the service provider of Open Badges Issuer system does not use a CMS to issue badges instead they have cloud services as well as web frameworks (Django) or learning management systems (moodle) (see section 6.2). So, in my opinion, the project should make a step back and

[1] https://jquery.com/, accessed 20.10.2017

[2] https://d3js.org/, accessed 20.10.2017

make a decision out of three:

1. A web application framework such as ASP.NET, Django, Laravel,Spring Framework, Zend Framework can be used. Migrate the concept and knowledge of this work so that a new prototype can be developed faster.

2. As an alternative, software-as-a-service by Open Badge Factory can be ordered. Is specialized on the issue service can be connected by a self-developed plugin with a CMS system. The costs for the services are acceptable, and the quality of the issuing is proven by a sufficient number of clients.

3. Finally, there is the following alternative: fork the Badgr Server and complete the system for task create content, user input and check the conditions, login system and visualization.

7.2 Future Work

This section names some use cases which are not considerate in the first phase. Instead, they should be taken into account for phase 2, regardless of the decision. If a service or installation is selected the features of the software have to be checked in detail. The ideas for future work can be separated into can be two parts: the software application and modifications of Open Badges. It should not be forgotten that the findings of the section 2.2, how systems which use Open Badges should be constructed.

A login for the Recipients can improve the quality of the input as well as the user can observe the badges they have been earned and the levels of badges they can reach. A link to a Backpack services or a social media site can improve the acceptance and utilisation. Other systems also provide a link to the Assertion so that Consumer can see the specific achievement on a single site. However, this is also a task for a Displayer. The login for the OER-representative can be used to upload the list of participants of OER further training so that the input from the lecturer can be avoided and no one has to upload the data or files. Considering these list of additional features, the final feature will be a link in the business card of the Campus Management Systems. For the Consumer of the website, a faceted search for the metadata of OER can be interested. However, first things first.

The implementation and publishing of a new Extension data class for Open Badges can use for several purposes. After the decision which metadata standard can be used for this application, a new Extension should be defined, published and used to append into every Open Badge which is awarded for the publishing of OER metadata. The next big thing for Open Badges is blockchain [3]. BadgeChain [4] is a nutshell to shift Open Badges forward.

[3] https://www.w3.org/2016/04/blockchain-workshop/interest/lemoie.html, accessed 20.10.2017
[4] https://medium.com/badge-chain, accessed 20.10.2017

Bibliography

Abramovich, Samuel, Christian Schunn, and Ross Mitsuo Higashi [2013]. "Are badges useful in education?: It depends upon the type of badge and expertise of learner". *Educational Technology Research and Development* 61.2 [2013], pages 217–232 (cited on page 6).

Adrian, Albisser, Burton-Monney Stephanie, Johner Patrick, Sarah Rossini-Freuler, Scherer-Hug August, Schiller Hansueli, and Nelly Buchser [2017]. *Applikationsprofil LOM -CH Version 1.3*. Mar 2017. http://www.educa.ch/sites/default/files/uploads/2017/02/lom-chv1.3_de.pdf (cited on page 35).

Alliance, Badge [2016a]. *About Open Badges*. Jun 2016. https://openbadges.org/about/ (cited on page 7).

Alliance, Badge [2016b]. *Developers Guide*. Dec 2016. https://openbadges.org/developers/ (cited on page 8).

Anderson, Ashton, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec [2013]. "Steering user behavior with badges". In: *Proceedings of the 22nd international conference on World Wide Web*. ACM. 2013, pages 95–106 (cited on page 6).

Arens, Robert [2008]. "Aspekte des barrierefreien Betriebs von TYPO3 mit dem Schwerpunkt der Pflege von Inhalten" [2008] (cited on page 76).

Association, TYPO3 [2009]. *Developing TYPO3 Extensions with Extbase and Fluid*. May 2009. https://docs.typo3.org/typo3cms/ExtbaseFluidBook/0-Introduction/Index.html (cited on page 56).

Association, TYPO3 [2017a]. *Form - Form Framework latest (9-dev) documentation*. May 2017. https://docs.typo3.org/typo3cms/extensions/form/Index.html (cited on page 62).

Association, TYPO3 [2017b]. *TYPO3 CMS - TYPO3 - The Enterprise Open Source CMS*. Jul 2017. https://typo3.org/typo3-cms/ (cited on page 52).

Atkins, Daniel Ewell, John Seely Brown, and Allen L Hammond [2007]. *A review of the open educational resources (OER) movement: Achievements, challenges, and new opportunities*. Creative common, 2007 (cited on page 32).

Bakhtiari, Shahram, Reihaneh Safavi-Naini, Josef Pieprzyk, et al. [1995]. "Cryptographic hash functions: A survey". *Centre for Computer Security Research, Department of Computer Science, University of Wollongong, Australie* [1995] (cited on page 28).

Ballesté, Alex [2013]. *Mozilla Open Badges 1.0. Show your skills to the world.* Apr 2013. https://aballeste.blogspot.co.at/2013/04/mozilla-open-badges-10-show-your-skills.html (cited on page 12).

Banos, Vangelis and Yannis Manolopoulos [2015]. "Web Content Management Systems Archivability". In: *East European Conference on Advances in Databases and Information Systems*. Springer. 2015, pages 198–212 (cited on page 77).

Barker, Phil and Lorna M. Campbell [2015]. "LRMI, Learning Resource Metadata on the Web." In: *WWW (Companion Volume)*. Edited by Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi. ACM, 2015, page 687. ISBN 978-1-4503-3473-0. http://dblp.uni-trier.de/db/conf/www/www2015c.html#BarkerC15 (cited on page 36).

Belshe, Mike, Roberto Peon, and Martin Thomson [2015]. *Hypertext Transfer Protocol Version 2 (HTTP/2)*. RFC 7540. May 2015. doi:10.17487/RFC7540. https://rfc-editor.org/rfc/rfc7540.txt (cited on page 26).

Ben-Kiki, Oren, Clark Evans, and Brian Ingerson [2009]. *YAML Ain't Markup Language (YAML) (tm) Version 1.2*. Technical report. YAML.org, Sep 2009. http://www.yaml.org/spec/1.2/spec.html (cited on page 61).

Bergmann, Sebastian and Stefan Priebsch [2013]. *Softwarequalität in PHP-Projekten;[mit Fallstudien von Firmen wie Facebook und Projekten wie TYPO3, Symfony und Zend Framework; Extra: mit kostenlosem E-Book]*. Hanser, 2013 (cited on page 76).

Berners-Lee, Tim [2006]. *Linked Data*. World wide web design issues. Jul 2006. http://www.w3.org/DesignIssues/LinkedData.html (cited on pages 11, 27).

Boutell, T. [1997]. *PNG (Portable Network Graphics) Specification Version 1.0*. RFC 2083 (Informational). Internet Engineering Task Force, Mar 1997. http://www.ietf.org/rfc/rfc2083.txt (cited on page 24).

Boyer, Mr Jérôme and Hafedh Mili [2011]. "IBM websphere ilog jrules". In: *Agile business rule development*. Springer, 2011, pages 215–242 (cited on page 40).

Bray, Tim [2014]. *The JavaScript Object Notation (JSON) Data Interchange Format*. RFC 7159. Mar 2014. doi:10.17487/RFC7159. https://rfc-editor.org/rfc/rfc7159.txt (cited on page 27).

Breitenstrom, Christian, Clemens Micklisch, and Małgorzata Mochól [2013]. *Sicherheitsstudie Content Management Systeme (CMS)*. 2013. https://www.bsi.bund.de/DE/Publikationen/Studien/CMS/Studie_CMS.html (cited on page 77).

Buckingham, James [2014]. "Open digital badges for the uninitiated". *The Electronic Journal for English as a Second Language* 18.1 [2014], pages 1–11 (cited on page 5).

Camilleri, Anthony F, Ulf Daniel Ehlers, Jan Pawlowski, et al. [2014]. *State of the art review of quality issues related to open educational resources (OER)*. Luxembourg: Publications Office of the European Union, 2014 (cited on page 33).

Chung, Lawrence, Brian A Nixon, Eric Yu, and John Mylopoulos [2012]. *Non-functional requirements in software engineering*. Volume 5. Springer Science & Business Media, 2012 (cited on page 50).

Commons, Creative [2011]. *Free to Learn Guide/Different Types of OER Meet Different Needs - Creative Commons*. May 2011. https://wiki.creativecommons.org/wiki/Free_to_Learn_Guide/Different_Types_of_OER_Meet_Different_Needs#cite_ref-1 (cited on page 33).

Consortium, IMS Global Learning [2017a]. *Open Badges Baking Specification*. Mar 2017. https://www.imsglobal.org/sites/default/files/Badges/OBv2p0/baking/index.html (cited on page 24).

Consortium, IMS Global Learning [2017b]. *Open Badges v2.0*. Mar 2017. https://www.imsglobal.org/sites/default/files/Badges/OBv2p0/index.html (cited on pages 12, 25).

Dahbi, Mohammed, Abdoulaye Diakité, Magdallen Juma, Andrea Hope, Sally M. Johnstne, and Vijay Kumar [2002]. *Forum on the Impact of Open Courseware for Higher Education in Developing Countries*. Technical report. Paris: UNESCO, Jul 2002. http://unesdoc.unesco.org/images/0012/001285/128515e.pdf (cited on page 31).

Daigle, L., D. van Gulik, R. Iannella, and P. Faltstrom [2002]. *Uniform Resource Names (URN) Namespace Definition Mechanisms*. RFC 3406 (Best Current Practice). Internet Engineering Task Force, Oct 2002. http://www.ietf.org/rfc/rfc3406.txt (cited on page 11).

Deimann, Markus et al. [2007]. *Volitional-supported learning with Open Educational Resources*. 2007 (cited on page 32).

Deimann, Markus, Jan Neumann, and Jöran Muuß-Merholz [2015]. "Whitepaper Open Educational Resources (OER) an Hochschulen in Deutschland–Bestandsaufnahme und Potenziale 2015". *Hamburg: open-educational-resources. de. URL: http://openeducational-resources. de/wp-content/uploads/sites/4/2015/02/Wh OER-Hochschule-2015. pdf* [2015] (cited on pages 32–33).

Diffie, Whitfield and Martin Hellman [1976]. "New directions in cryptography". *IEEE transactions on Information Theory* 22.6 [1976], pages 644–654 (cited on page 28).

Dürst, Martin and Michel Suignard [2005]. *Internationalized Resource Identifiers (IRIs)*. Proposed Standard 3987. The Internet Engineering Task Force (IETF), Jan 2005. http://tools.ietf.org/html/rfc3987 (cited on page 11).

Duval, Erik [2002]. *Learning object metadata (LOM)*. Technical report 1484.12.1 - 2002. IEEE, 2002 (cited on page 35).

Easley, David and Arpita Ghosh [2016]. "Incentives, gamification, and game theory: an economic approach to badge design". *ACM Transactions on Economics and Computation* 4.3 [2016], page 16 (cited on page 6).

Ebner, M., C.F. Freisleben-Teutscher, O. Gröblinger, M. Kopp, K. Rieck, S. Schön, P. Seitz, M. Seissl, S. Ofner, and C. Zwiauer [2016]. *Empfehlungen für die Integration von Open Educational Resources an Hochschulen in Österreich*. 2016. http://www.fnm-austria.at/fileadmin/user_upload/documents/Buecher/2016_fnma-OER-Empfehlungen_final.pdf (cited on pages 1, 37).

Ebner, Martin, Elly Köpf, Jöran Muuß-Merholz, Martin Schön, Sandra Schön, and Nils Weichert [2015]. "Ist-Analyse zu freien Bildungsmaterialien (OER)". *Die Situation von freien Bildungsmaterialien (OER) in Deutschland in den Bildungsbereichen Schule, Hochschule, berufliche Bildung und Weiterbildung. Berlin: Wikimedia Deutschland. URL: http://l3t. eu/oer/images/band10. pdf* [2015] (cited on pages 32–33).

Ebner, Martin and Christine Stöckler-Penz [2011]. "Open Educational Resources als Lifelong-Learning-Strategie am Beispiel der TU Graz". *The lifelong learning university* [2011], pages 53–60 (cited on page 1).

Ebner, Martin, Paolo Budroni, Victoria Buschbeck, Asura Enkhbayar, Andreas Ferus, Christian F Freisleben-Teutscher, Ortrun Gröblinger, Robert Hafner, Michael Kopp, Ina Matt, et al. [2017]. "Konzept OER-Zertifizierung an österreichischen Hochschulen" [2017] (cited on pages 2, 37, 73).

Education, Pearson [2013]. *Open badges are unlocking the emerging jobs economy*. 2013 (cited on page 25).

ElGamal, Taher [1985]. "A public key cryptosystem and a signature scheme based on discrete logarithms". *IEEE transactions on information theory* 31.4 [1985], pages 469–472 (cited on page 29).

Erdogmus, Hakan, Grigori Melnik, and Ron Jeffries [2010]. *Test-Driven Development.* 2010 (cited on page 54).

fnm-austria, editor [2012]. *FNM Jahresbericht - 2012*. 2012. http://www.fnm-austria.at/fileadmin/user_upload/documents/Jahresberichte/Jahresbericht_2012.pdf (cited on page 37).

Friesen, Norm and Christine Wihak [2013]. “From OER to PLAR: Credentialing for open education”. *PLA Inside Out: An International Journal on Theory, Research and Practice in Prior Learning Assessment* 2.1 [2013] (cited on page 6).

Garrison, D Randy and Heather Kanuka [2004]. “Blended learning: Uncovering its transformative potential in higher education”. *The internet and higher education* 7.2 [2004], pages 95–105 (cited on page 38).

Gibson, David, Nathaniel Ostashewski, Kim Flintoff, Sheryl Grant, and Erin Knight [2015]. “Digital badges in education”. *Education and Information Technologies* 20.2 [2015], pages 403–410 (cited on pages 1, 3, 5).

Goligoski, Emily [2012]. “Motivating the learner: Mozilla s open badges program”. *Access to Knowledge: A Course Journal* 4.1 [2012] (cited on page 1).

Grant, Sheryl [2014]. *What counts as learning: Open digital badges for new opportunities*. BookBaby, 2014 (cited on page 9).

Grothaus, Thomas [2016]. *form4: Wie sicher ist TYPO3?* May 2016. http://www.form4.de/artikel/wie-sicher-ist-typo3/ (cited on page 77).

Group, W3C XML Working [2008]. *The XML 1.0 Standard (5th Edition)*. W3C Recommendation. 2008. http://www.network-theory.co.uk/w3c/xml/ (cited on page 27).

Halavais, Alexander MC [2012]. “A genealogy of badges: Inherited meaning and monstrous moral hybrids”. *Information, Communication & Society* 15.3 [2012], pages 354–373 (cited on page 3).

Hellman, Martin E [2002]. “An overview of public key cryptography”. *IEEE Communications Magazine* 40.5 [2002], pages 42–49 (cited on page 28).

Hickey, Daniel T, J Willis, and J Quick [2015]. “Where badges work better”. *EDUCAUSE Learning Initiative ELI* [2015] (cited on page 7).

Huotari, Kai and Juho Hamari [2012]. “Defining gamification: a service marketing perspective”. In: *Proceeding of the 16th International Academic MindTrek Conference*. ACM. 2012, pages 17–22 (cited on page 6).

Johnstone, Sally M [2005]. “Open educational resources serve the world”. *Educause Quarterly* 28.3 [2005], page 15 (cited on page 32).

Jones, Michael, John Bradley, and Nat Sakimura [2015]. *JSON Web Signature (JWS)*. RFC 7515. May 2015. doi:10.17487/RFC7515. https://rfc-editor.org/rfc/rfc7515.txt (cited on page 27).

Josefsson, S. [2006]. *The Base16, Base32, and Base64 Data Encodings*. RFC 4648 (Proposed Standard). Internet Engineering Task Force, Oct 2006. http://www.ietf.org/rfc/rfc4648.txt (cited on page 27).

Jurišic, Aleksandar and A Menezes [1997]. “Elliptic curves and cryptography”. *Dr. Dobb’s Journal* [1997], pages 26–36 (cited on page 29).

Kopp, Michael and Martin Ebner [2017]. “Certification of MOOCs–Advantages, Challenges and Practical Experiences”. *REVISTA ESPANOLA DE PEDAGOGIA* 75.266 [2017], pages 83–100 (cited on page 6).

Kreutzer, Till et al. [2013]. *Open Educational Resources (OER), Open-Content und Urheberrecht*. : 2013 (cited on page 32).

López, Juan Miguel, Afra Pascual, Llucia Masip, Toni Granollers, and Xavier Cardet [2011]. “Influence of web content management systems in web content accessibility”. In: *IFIP Conference on Human-Computer Interaction*. Springer. 2011, pages 548–551 (cited on page 51).

Ludwig, Luise, Kristin Narr, Sabine Frank, and Daniel Staemmler, editors [2013]. *Lernen in der digitalen Gesellschaft – offen, vernetzt, integrativ*. 2013. http://dl.collaboratory.de/reports/Ini7_Lernen.pdf (cited on page 33).

Mauthe, Andreas and Peter Thomas [2004]. *Professional content management systems: handling digital media assets*. John Wiley & Sons, 2004 (cited on page 51).

McGonigal, J [2011]. "We don't need no stinkin badges: How to reinvent reality without gamification.[Video]". *Retrieved from GDC Vault: http://www. gdcvault. com/play/1014576/We-Don-t-Need-No* [2011] (cited on page 4).

Menezes, Alfred J, Paul C Van Oorschot, and Scott A Vanstone [1996]. *Handbook of applied cryptography*. CRC press, 1996 (cited on page 29).

Meyer, Robert and Martin Helmich [2016]. *Praxiswissen Typo3*. O'Reilly Germany, 2016 (cited on page 76).

Mirdha, Aakanksha, Apurva Jain, and Kunal Shah [2014]. "Comparative analysis of open source content management systems". In: *Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference on*. IEEE. 2014, pages 1–4 (cited on page 78).

Montola, Markus, Timo Nummenmaa, Andrés Lucero, Marion Boberg, and Hannu Korhonen [2009]. "Applying game achievement systems to enhance user experience in a photo sharing service". In: *Proceedings of the 13th International MindTrek Conference: Everyday Life in the Ubiquitous Era*. ACM. 2009, pages 94–97 (cited on page 4).

Nirgude, Kirti R and Kaushal K Giri [2009]. "Open source content management software: a comparative analysis" [2009] (cited on page 51).

OECD [2007]. *Giving Knowlegde for Free*. Technical report. 2007. http://www.oecd.org/dataoecd/35/7/38654317.pdf (cited on page 31).

Peña-López, Ismael et al. [2007]. "Giving knowledge for free: The emergence of open educational resources" [2007] (cited on page 33).

Presant, Don [2016]. *6 Predictions for Open Badges in 2016*. Jan 2016. https://littoraly.wordpress.com/2016/01/01/6-predictions-for-open-badges-in-2016/ (cited on page 9).

Press, Cambridge University [2017]. *badge Meaning in the Cambridge English Dictionary*. 2017. http://dictionary.cambridge.org/dictionary/english/badge (cited on page 3).

Press, NISO [2004]. "National Information Standards Organization". *Understanding Metadata* [2004] (cited on page 27).

Ramsden, Andy and Follow Following Unfollow Andy Ramsden [2015]. "Open Badges: Trying to navigate through the melting pot of ideas" [2015] (cited on page 7).

Rau, Jochen, Sebastian Kurfürst, and Martin Helmich [2013]. *Zukunftssichere TYPO3-Extensions mit Extbase und Fluid*. O'Reilly Germany, 2013 (cited on pages 54, 74).

Rensing, Christoph [2013]. "Standards für Lehr-und Lerntechnologien-Metadaten, Inhaltsformate und Beschreibung von Lernprozessen". *Lehrbuch für Lernen und Lehren mit Technologien* [2013] (cited on pages 34, 36).

Riley, Jenn [2017]. *Understanding metadata: What is metadata, and what is it for?* Technical report. National Information Standards Organization, 2017 (cited on page 27).

Rivest, Ronald L, Adi Shamir, and Leonard Adleman [1978]. “A method for obtaining digital signatures and public-key cryptosystems”. *Communications of the ACM* 21.2 [1978], pages 120–126 (cited on page 29).

Rogaway, Phillip and Thomas Shrimpton [2004]. “Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance”. In: *International Workshop on Fast Software Encryption*. Springer. 2004, pages 371–388 (cited on page 28).

Smith, Sue and et altera [2014]. *Verifying Badges for Display*. Jul 2014. https://github.com/mozilla/openbadges-backpack/wiki/Verifying-Badges-for-Display (cited on page 25).

Sporny, Manu, Dave Longley, Gregg Kellogg, and Markus Lanthaler [2017]. *JSON-LD Syntax 1.1*. Technical report. Jun 2017. http://json-ld.org/spec/latest/json-ld-syntax/ (cited on pages 11, 27).

Stöckl, Andreas and Frank Bongers [2005]. *Einstieg inTYPO3: [Installation, Konfiguration und Betrieb ; erste Schritte zum eigenen TYPO3-Projekt ; inkl. Templates, TypoScript und Extensions ; mit TypoScript ; aktuell zur neuesten Version 3.7]*. Galileo computing. Bonn: Galileo Press, 2005. ISBN 3-8984-2604-1 (cited on page 58).

W3C [2008]. *Scalable Vector Graphics*. http://www.w3.org/Graphics/SVG/. seen at April 2008 (cited on page 24).

The Mozilla Foundation Peer 2 Peer University, The MacArthur Foundation [2011]. *Open Badges for Lifelong Learning*. Sep 2011. https://wiki.mozilla.org/images/b/b1/OpenBadges-Working-Paper_092011.pdf (cited on pages 1, 5, 7–8).

Thompson, Matt [2013]. *Introducing Open Badges 1.0 – The Mozilla Blog*. Mar 2013. https://blog.mozilla.org/blog/2013/03/14/open_badges/ (cited on page 12).

W3C [2014]. *RDF 1.1 Concepts and Abstract Syntax*. http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/. 2014. http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/ (cited on page 27).

Weller, Martin [2010]. “Big and little OER” [2010] (cited on pages 33–34).

Wiley, D and E Edwards [2002]. “Online self-organizing social systems”. *Quarterly review of distance education* 3.1 [2002], page 33 (cited on page 33).

Wiley, David [2014]. “The Access Compromise and the 5th R”. *Iterating toward openness* [2014] (cited on page 32).

Wu, Margaret, Dan Whiteley, and Margaret Sass [2015]. “From girl scout to grown up: Emerging applications of digital badges in higher education”. *The online journal of distance education and e-learning* 3.2 [2015], pages 48–52 (cited on page 6).

Wüster, Mario and Martin Ebner [2016]. “How to integrate and automatically issue Open Badges in MOOC platforms”. *Proceedings of the European Stakeholder Summit on experiences and best practices in and around MOOCs (EMOOCS 2016)* [2016], page 279 (cited on page 6).

Xu, Yongwen [2011]. “Literature review on web application gamification and analytics”. *Honolulu, HI* [2011], pages 11–05 (cited on page 4).

Young, Jeffrey R [2012]. “’Badges’ Earned Online Pose Challenge to Traditional College Diplomas”. *The Education Digest* 78.2 [2012], page 48 (cited on page 1).

Youseck Yang Yonggoon Kim, Yangwon Lim and Hankyu Lim [2016]. “A Comparison of Open-Source CMS and Analysis of Security Vulnerability”. *INTERNATIONAL JOURNAL OF COMPUTERS* 10 [2016], pages 82–86 (cited on page 51).

Ziedorn, Frauke, Elena Derr, and Janna Neumann [2013]. “Metadaten für Open Educational Resources (OER)”. *Eine Handreichung für die öffentliche Hand, erstellt von der Technischen Informationsbibliothek (TIB)* [2013] (cited on page 34).

Abbreviations

API Application Programming Interface

ADL Advanced Distributed Learning Initiative

BRMS Business Rule Management System

CC-BY Creative Commons

CMS Content Management System

CRUD Create Read Update and Delete

CS2N Carnegie Mellon University's Computer Science Student Network

DDD Domain Driven Design

DML Digital Media and Learning

ECC Elliptic Curve Cryptography

EXT Extension

HTML Hyper Text Markup Language

HTTP Hypter Text Transfer Protocol

IDE Integrated Development Environment

IRI Internationalized Resource Identifier

JSON JavaScript Object Notation

JSON-LD JavaScript Object Notation-Linked Data

JWS Java Web Signature

JOSE JavaScript Object Signing and Encryption

LL1 Lecturer Level 1

LL2 Lecturer Level 2

LOM Learning Object Metadata

LP1 Lecturer Part 1

LP2 Lecturer Part 2

LMRI Learning Resource Metadata Initiative

MAC Message Authentication Code

MOOC Massive Open Online Course

MVC Model View Controller

MTTF Mean Time to Failure

MTTR Mean Time to Recovery

OER Open Educational Resources

OB Open Badges

OBE Open Badges Ecosystem

OOP Object Oriented Programming

PNG Portable Network Graphics

RAS Rivest-Shamir-Adleman

REST Representational State Transfer

RDF Resource Description Framework

SCORM Sharable Content Object Reference Model

SVG Scaled Vector Graphics

TDD Test Driven Development

UC Use Case

UID Unique Identifier

UL1 University Level 1

UL1 University Level 2

UL3 University Level 3

UP1 University Part 1

UP2 University Part 2

UP3 University Part 3

UUID Universally Unique Identifier

URI Unique Resource Identifier

URL Unique Resource Locator

WWW World Wide Web

XML Extended Markup Language