



Mario Wüster, BSc.

# Open Badges for Learning Environments

## Master's Thesis

to achieve the university degree of  
Diplom-Ingenieur  
Master's degree programme: Computer Science

submitted to:  
Graz University of Technology

Supervisor:  
Assoc.Prof. Dipl.-Ing. Dr.techn. Martin Ebner

Institute for Information Systems and Computer Media  
Head: Prof. Dipl.-Ing. Dr.techn. Frank Kappe

Graz, September 2015

## 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 marked all material which has been quoted either literally or by content from the used sources.

Graz, \_\_\_\_\_  
Date

\_\_\_\_\_  
Signature

## Eidesstattliche Erklärung<sup>1</sup>

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.

Graz, am \_\_\_\_\_  
Datum

\_\_\_\_\_  
Unterschrift

---

<sup>1</sup>Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

# Acknowledgments

I would like to express my sincerest gratitude to my supervisor, Dr. Martin Ebner, who has supported me throughout my thesis even on his richly deserved vacation. 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 Alexei and Gerald for their patience and helping hands regarding iMooX. No one demanded that in the way they did.

Then, I would like to thank my parents, 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.

Finally, my dear Hanna. I have to thank her for more than just peruse this thesis. She has been unbelievable patient, caring and motivating throughout my whole studies. She is all I ever wished for.

Mario Wüster  
Graz, September 2015

# Abstract

Increasingly, learning takes place online. Though there are quite good opportunities like MOOCs for example, acquired knowledge is not approved appropriately. The most common approach is to generate personalized certificates of participation for successfully passing a course. However, those are mostly not digitally signed, which makes them easier to counterfeit and therefore reduces their value against verifiable documents.

Apart from that, such certificates typically represent summative feedback which is just the result of a learning process. Quite frequently, people enroll to online courses to have access to certain course material, but they do not have any intention to take exams or to pass the course. Although they acquire some knowledge or skills, traditional certifications do not consider them.

Mozilla's Open Badges, which are digital artifacts with embedded meta-data, could help to solve these problems. An Open Badge contains, beside its visual component, data to verify its receipt. An issuer can digitally sign such badges, which can be verified by everyone and anytime. In addition, badges of different granularity, called Micro- and Meta-Badges, cannot just certify successful course completion, but also help to steer the learning process of learners through formative feedback.

To use those advantages for the learning platform iMooX, a web application was developed which enables iMooX to issue Open Badges to its users. Thereby, Micro- as well as Meta-Badges were considered and issued within the iMooX course COER15. The applications suitability was confirmed by voluntary feedback of badge earners as well as the lecturers.

# Kurzfassung

Lernen findet immer häufiger online statt. Obwohl es dafür sehr gute Möglichkeiten, zum Beispiel MOOCs, gibt, ist dabei erworbenes Wissen nicht geeignet nachweisbar. Die gängigste Methode ist, Lernenden die erfolgreiche Absolvierung eines Kurses mit einer personalisierten Teilnahmebestätigung zu zertifizieren. Dabei handelt es sich meistens jedoch nicht um digitale Zertifikate, welche daher leichter fälschbar sind und somit keinen Wert gegenüber verifizierbaren Dokumenten haben.

Abgesehen davon entsprechen solche Teilnahmebestätigungen einer summativen Evaluation des erworbenen Wissens, sprich nur dem Resultat eines Lernprozesses. Häufig werden Online-Kurse zwar besucht um Wissen zu erwerben, die Intention Prüfungen abzulegen oder den Kurs zur Gänze zu absolvieren bleibt jedoch aus. In diesen Fällen wird, mit traditionellen formalen Zertifizierungsmethoden, erworbenes Wissen nicht berücksichtigt.

Mozilla's Open Badges, digitale Artefakte mit integrierten Meta-Daten, könnten dabei helfen genau diese Probleme zu lösen. Ein Open Badge beinhaltet, neben einer visuellen Komponente, Daten, mit deren Hilfe man den Erhalt solch eines Badges verifizieren kann. Sie können vom Aussteller digital signiert und von jedem und jederzeit überprüft werden. Außerdem können unterschiedlich mächtige Badges, sogenannte Micro-Badges und Meta-Badges, dabei helfen, sowohl den Lernprozess Lernender zu steuern als auch deren Leistungen zu bestätigen. Damit wäre es möglich nicht nur die erfolgreiche Absolvierung eines Kurses, sondern auch erbrachte Teilleistungen verifiziert anzuerkennen.

Um diese Vorteile für die Lernplattform iMooX zu nutzen, wurde im Zuge dieser Arbeit eine Web-Applikation entwickelt, welche iMooX erlaubt dessen Benutzerinnen und Benutzern Open Badges zu verleihen. Dabei wurden sowohl Micro- als auch Meta-Badges implementiert und dessen Einsatz im Zuge des COER<sub>15</sub> Kurses evaluiert. Freiwilliges Feedback von Benutzerinnen und Benutzern, die Badges erhalten haben, als auch des Kursleiters, zeigte, dass die Applikation durchaus zur weiteren Verwendung für iMooX Kurse geeignet ist.

# Contents

<b>Abstract</b>	<b>iv</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Open Badges</b>	<b>4</b>
2.1 Theoretical Foundations . . . . .	4
2.1.1 Hashing . . . . .	4
2.1.2 Base64 Encoding . . . . .	5
2.1.3 Public Key Cryptography . . . . .	6
2.1.4 Digital Signature . . . . .	7
2.1.5 JSON Web Signature . . . . .	8
2.2 Physical Badge vs. Digital Badge vs. Open Badge . . . . .	8
2.3 Open Badges Infrastructure . . . . .	11
2.3.1 Meta-Data Specification . . . . .	12
2.3.2 Image-Baking . . . . .	21
2.3.3 Backpack . . . . .	23
2.3.4 Displayer . . . . .	24
2.3.5 Issuer . . . . .	24
<b>3 Learning Environments</b>	<b>27</b>
3.1 Learning . . . . .	27
3.1.1 Learning Theories . . . . .	28
3.1.2 Motivation . . . . .	30
3.2 Learning Technologies . . . . .	31
3.2.1 LCMS and CrMS . . . . .	32
3.2.2 PLE . . . . .	33
3.2.3 MOOC . . . . .	35
3.3 Educational E-Assessment . . . . .	38
3.3.1 Assessable Knowledge . . . . .	39
3.3.2 Assessment Tasks . . . . .	39
<b>4 Integrating Open Badges in Learning Environments</b>	<b>42</b>
4.1 Source Code Modification . . . . .	42
4.2 Plug-In . . . . .	43

## Contents

4.3	Add-In/Add-On . . . . .	44
4.4	Web application . . . . .	45
4.5	Software as a Service . . . . .	45
<b>5</b>	<b>Implementation of a Badge Issuing System for iMooX</b>	<b>47</b>
5.1	iMooX . . . . .	47
5.2	Overview . . . . .	49
5.2.1	iMooX Scripts . . . . .	50
5.3	Components . . . . .	52
5.3.1	Database . . . . .	53
5.3.2	Hibernate . . . . .	56
5.3.3	Model DAO . . . . .	58
5.3.4	RSA PK . . . . .	58
5.3.5	Web.xml . . . . .	59
5.3.6	Servlets . . . . .	59
5.3.7	JSP . . . . .	60
5.4	Use-cases . . . . .	65
5.4.1	Back-End . . . . .	65
5.4.2	Front-End . . . . .	69
5.4.3	The Awarding Process . . . . .	71
<b>6</b>	<b>Evaluation</b>	<b>74</b>
6.1	COER15 . . . . .	74
6.2	Statistics . . . . .	76
6.3	Lessons Learned . . . . .	77
<b>7</b>	<b>Conclusions</b>	<b>79</b>
7.1	Discussion . . . . .	79
7.2	Future Work . . . . .	81
	<b>Bibliography</b>	<b>85</b>

# List of Figures

2.1	Generalized hashing process within the Open Badges specification. . . . .	5
2.2	Basic steps within the digitally signing and verification process. . . . .	7
2.3	Structure of a Java Web Signature. . . . .	8
2.4	Famous examples of physical badges . . . . .	9
2.5	Some digital badges of the FourSquare platform. . . . .	10
2.6	The Open Badges Infrastructure . . . . .	12
2.7	Structure of a Badge Assertion JSON object . . . . .	13
2.8	Responsibilities of an issuer . . . . .	24
3.1	Example TUG PLE page with some installed widgets . . . . .	34
5.1	Self-assessment quiz sample. . . . .	48
5.2	Example of a traditional certificate of participation. . . . .	49
5.3	Basic components of badgeit . . . . .	53
5.4	Used database schema. . . . .	54
5.5	Possible badge state transitions. . . . .	55
5.6	Relation between Servlets and JSPs . . . . .	61
5.7	The homepage of <i>badgeit</i> . . . . .	63
5.8	Web page to define a new badge. . . . .	63
5.9	Badge details web page . . . . .	64
5.10	Badge edit web page . . . . .	64
5.11	Example showing the two possible criteria defining input forms. . . . .	66
5.12	Popping up dialog inquiring the earners email address. . . . .	67
5.13	The two implemented statistic visualizations. . . . .	68
5.14	Overview of the badge awarding process triggered by the user. . . . .	70
5.15	The personal badge collection page of a user (front-end). . . . .	72
6.1	All achievable badges of the COER15 iMooX course. . . . .	75
6.2	Quiz attempts per date . . . . .	76



# List of Tables

2.1	Base64 alphabet . . . . .	6
4.1	Promising open-source Plug-Ins . . . . .	44
6.1	Overview of earned and generated COER15 course badges. . . . .	77

# List of Listings

2.1	URL response of a revoked badge. . . . .	14
2.2	Example revocation list containing two revoked badges. . . . .	15
2.3	Example issuer data encoded as json object. . . . .	15
2.4	Example alignment object. . . . .	16
2.5	Example Identity Object with a plain text email address. . . . .	16
2.6	Example Identity Object with a hashed recipient. . . . .	17
2.7	Example Verification Object for a hosted badge assertion. . . . .	17
2.8	Example Verification Object for a signed badge assertion. . . . .	17
2.9	Example Badge Class object. . . . .	18
2.10	Example decoded signed badge assertion (identity hash has been shortened) . . . . .	19
2.11	Example Extension (taken from official specification). . . . .	20
2.12	Example Extension object (taken from official specification). . . . .	20
2.13	Example Validation that links to a validation schema for badge assertions (taken from official specification). . . . .	20
2.14	Structure of the badge image's SVG child element. . . . .	22
2.15	Sample call to the REST baking API of the Mozilla backpack. . . . .	22
5.1	Example response of the course list script. . . . .	51
5.2	Sample iMooX-script response for a certain user. . . . .	52
5.3	Hibernate database connection properties. The string "hn" is the abbreviation for "hibernate" and was just used within that listing to avoid line breaks. . . . .	57
5.4	Hibernate entity mapping with an own entity configuration file. . . . .	57
5.5	Hibernate attribute annotations for Assertion.java. . . . .	57
5.6	Hibernate entity mapping with annotations in corresponding Java classes. . . . .	58
5.7	Typical DAO class method returning an instance of a certain table. . . . .	58
5.8	Generate RSA private key with openssl . . . . .	58
5.9	Example response to an <i>/public-key.pem</i> servlet call . . . . .	60
5.10	Sample response of the GetUserBadgesServlet containing one custom object including two badges. . . . .	70

# 1 Introduction

Formally earned grades and degrees are mostly the only product that communicates learning success to the general public. As those are final results, the *path* towards achieving them is mostly ignored. That has not been important for a long time, because typically, curricula exactly state what students have to know to get a grade. However today's learning is not the same as just one decade ago [15, 53, 62]. It can no longer be seen as an isolated and rigid concept. Someone is not automatically sophisticated just by possessing a university degree. To be successful today, one needs to have individual learn paths, skills, specializations and experiences.

Through social media and technologies, learner can connect, collaborate, discuss, be creative and make experiences which are not possible in traditional formal contexts. So learning is no longer simple consumption, instead participating and producing content get into focus that others can comment on, overhaul or acknowledge. Therefore, also learning environments are no longer considered to be single online places rather than distributed, open and connected environments.

More and more educational initiatives like OpenCourseWare<sup>1</sup>, Peer-2-Peer University<sup>2</sup>, edX<sup>3</sup>, Coursera<sup>4</sup> and others offer free education that is also mostly clear of common obstacles formal education has, with tuition fees, physiological presence and time pressure naming just a few of them.

However, it is hard to motivate students to actively participate in non-formal and informal educational systems if they do not get any recognition for their commitment [30]. Traditionally, complimentary learning environments certify successful course attendance with some printable sheet of paper, but, to be honest, that has no value. These can easily be counterfeit which makes them not trustworthy or validatable and also, due to missing standards, there is no way to manage earned certificates of different platforms online and centralized. Therefore, as non-formal and informal learning has greatly increased there is need for appropriate recognition outside traditional educational systems.

Steering user behavior of online communities by game elements has become a popular mechanism [2, 4, 30]. As active participation rates are commonly used to

---

<sup>1</sup><http://ocw.mit.edu/index.htm>, August 22, 2015.

<sup>2</sup><https://www.p2pu.org/en/> (P2PU), August 22, 2015.

<sup>3</sup><https://www.edx.org/>, August 22, 2015.

<sup>4</sup><https://www.coursera.org/>, August 22, 2015.

## 1 Introduction

measure success of social networks [30], the endeavor has always been to amplify user commitment. As that has seemed to work well, also educators were interested to make use of gamification [36], which is, according to Deterding et al. [18], defined as “using game design elements in non-game contexts”. Educational gamification is thereby not meant to be turning learning environments into games, but rather to apply game elements in order to motivate students to complete tasks or to arouse the excitement and fun in exploring complex topics. Commonly used game elements are achievement badges, time constraints, clear goals, challenges, skill-adaptive levels, points and leaderboards [18, 28].

In the educational context, badges are basically digital artifacts that are used to acknowledge activities, quality, commitment, effort, skills or achievements [2, 22, 26]. However, until 2011, there was no common standard which made it hard to share earned badges across social networks and to manage them centralized. In addition, it was hard to verify the receipt because badges were just small images that could be easily forged.

In 2011, Mozilla<sup>5</sup> developed an open technical standard which enables everyone to issue, earn and display digital badges [41]. Embedded meta-data links back to the issuer, the badge description and the moment the badge was awarded. With also including the earner identity, those badges, Mozilla called them *Open Badges*, are trustworthy verifiable.

Therefore, they can assist to capture individual learning paths from any environment, be awarded from multiple sources and for theoretically limitless individual skills or achievements of any granularity [22, 23, 26]. Learner can manage earned badges centralized and share chosen ones on places that matter. Thereby, which badges are shared and published is completely up to the earner. So one can stack together badges that relate to a certain job description or the earner is especially proud of. Collections of badges can serve as virtual résumés that capture ones competencies and qualities.

As provision of proper formative as well as summative feedback is essential for the success of learning processes and their outcomes [22, 29], it could be beneficial to use Open Badges as such feedback instrument. Several related studies [16, 20, 27, 28, 51, 52] made confirming experiments which made it highly interesting to investigate how to integrate badging in learning environments.

Therefore, this thesis explains the concept of Mozilla’s *Open Badges* in detail and presents general approaches to integrate badging into online educational learning environments. Based on that theoretical background, a specific learning environ-

---

<sup>5</sup>In cooperation with the P2PU and funds of the McArthur Foundation ([www.macfound.org](http://www.macfound.org), August 22, 2015).

## 1 Introduction

ment was equipped with a completely new developed badging system. Thereby components, concepts and implementation details get described in detail.

It is not about learning environments in general nor does it represent pedagogical or psychological point-of-views on the usage of Open Badges within such environments. It neither goes into detail about how to design badges nor does it compare or comment about already existing implementations.

First, Mozilla's Open Badges Infrastructure (OBI) is explained in detail as occurring terms and concepts are important for later chapters. Following that, chapter 3 briefly covers concepts of learning, related theories and some relevant learning environments. Chapter 4 then presents general approaches of integrating badging capabilities in such learning environments whereby related work is mentioned at appropriate locations. One specific integration is then presented in chapter 5 followed by a short evaluation. Finally, some found issues are discussed as well as a couple of ideas for future work stated within chapter 7.

# 2 Open Badges

## 2.1 Theoretical Foundations

Before defining and discussing Open Badges, a few technical concepts should be explained. Therefore, the following sections provide rough definitions to get used to certain terms. It is not intended to provide complete definitions as that would definitely go beyond the scope of this thesis. Nevertheless, it is important to understand the principles as they were needed within the Open Badges meta-data specification as well as for the implementation described in chapter 5.

### 2.1.1 Hashing

Hashing is the process of applying mathematical one-way functions<sup>1</sup> (hash function  $H()$ ) to certain input data  $x$  [48]. An arbitrary digital data set of variable length gets thereby mapped to a, usually way smaller, data set of fixed length ( $h_s = H(x)$ ). The output is often called hash value, checksum, hash code or message digest.

Within this thesis, hashing is used to ensure data integrity which is a classical cryptographic use-case. Thereby, hash functions should secure input data from intruders, which means to *"make it difficult for someone to decrypt or change the data without affecting the hash value"* [48].

Therefore, important properties of cryptographic hash functions are:

- $H()$  must produce the same output for identical inputs (*determinism*)
- $H(x)$  must be collision resistant. That means, that two different inputs ( $x \neq x'$ ) are not allowed to map to the same output ( $h_s(x) \neq h_s(x')$ )

Therefore  $x$  should be mapped as uniformly as possible to the range of output data (*uniformity*). That basically means, that each output should be generated with equal possibility. Ideally, one flipped bit produces a completely different hash value

---

<sup>1</sup>One-way functions are easy to compute, but hard to invert given only output values.

## 2 Open Badges

- The output is of fixed length, so to ensure that the computational complexity for brute-force inversion<sup>2</sup> suffices, the size of the output range has to be large

To increase the entropy and therefore the security of a hash function, randomly selected bytes can be pre- or suffixed to the input data. These randomly selected bytes are called *salt*. The Open Badges specification demands, that *if* the badge earner identity gets salted, the used salt has to be suffixed to the identity string.

The general hashing procedure used within Open Badges is illustrated in figure 2.1. There, the dashed box indicates the final earner identity as it is represented within the corresponding JSON object (*Identity Object*, refer to 2.3.1.4) of the embedded meta-data.

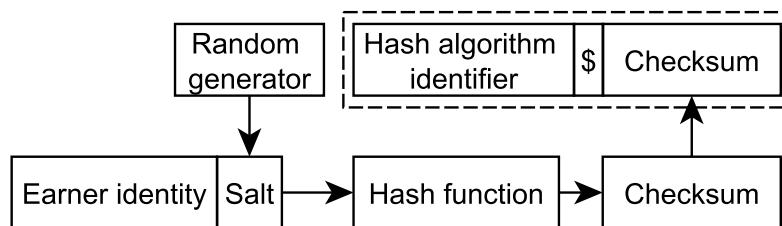


Figure 2.1: Generalized hashing process within the Open Badges specification.

### 2.1.2 Base64 Encoding

Encodes binary data (octets) to ASCII<sup>3</sup> chars [32] to be able to smoothly ship data across networks. That avoids miss-interpretation of binary data between communicating parties. A sequence of bits gets thereby partitioned into three byte blocks ( $3 \times 8 = 24$  bit). If the bit sequence cannot be divided by three (octets), the remaining missing bytes get padded with zero bits. If a byte then solely consists of *padded* zero bits, it is encoded as "=".

Then, these three byte blocks get treated as one concatenated sequence of four sextets, each representing a number between 0 and  $2^6 - 1 = 63$ . Each of those numbers then indexed a character in the Base64 alphabet (see table 2.1). If corresponding characters for index 62 and 63 of table 2.1 get replaced by "-" (minus) and "\_" (underscore), the alphabet forms the "URL and Filename safe" Base 64 alphabet, often referenced as *base64url* alphabet [32].

For example, the Base64 encoded title of this thesis ("Open Badges in Learning Environments") is "T3BlbiBCYWRnZXMgaW4gTG9hcm5pbmVzRW52aXJvbm1lbnRz".

<sup>2</sup>A method which calculates all possible input data from the given hash value. The more steps that procedure needs, the more secure the hash function is.

<sup>3</sup>American Standard Code for Information Interchange <http://tools.ietf.org/html/rfc20>, August 22, 2015.

## 2 Open Badges

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

Table 2.1: Base 64 alphabet, taken from [32] table 1.

### 2.1.3 Public Key Cryptography

For digital signatures and their certitude, it is important to understand *private-* and *public-keys*. Therefore, this section touches on the principle of asymmetric cryptography, also known as Public Key Cryptography (PKC). Asymmetric, because two parties do not share a common secret, as it is the case of symmetric cryptography. To still be able to interchange private messages, special key pairs, each consisting of a private- and a *corresponding* public key, are used. For example, RSA<sup>4</sup> keys, which are also used within the Open Badges specification, are generated by mathematical one-way-functions based on huge prime numbers [49]. The private key obviously has to be retained as a secret whereas the public has to be publicly accessible.

If someone, the sender, wants to encrypt a message for a certain receiver (consider the optional path in figure 2.2a), the public key of the receiver is used to encrypt the message. That could be done by anybody of course. Decrypting that encrypted messages by contrast, is solely possible with the appropriate private key of the receiver. A potential attacker, depending on the actually used algorithm, can typically not derive the private key from the public key because of those one-way-functions. Thereby, it is easy to calculate the public key from the private key, but it is particularly hard to recover the private key from the public key. That is basically the main idea behind asymmetric cryptography [48] and that knowledge definitely suffices within that thesis to understand digital signatures.

---

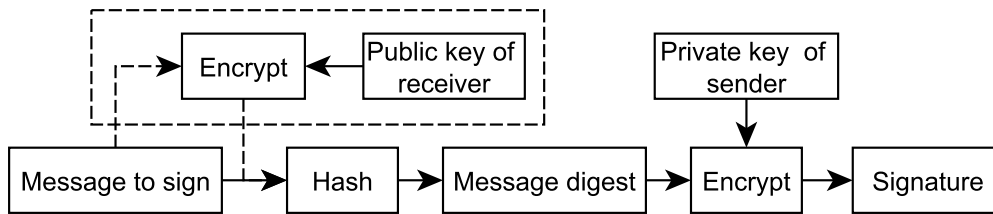
<sup>4</sup>Encryption algorithm by Ron Rivest, Adi Shamir and Leonard Adleman (1977).



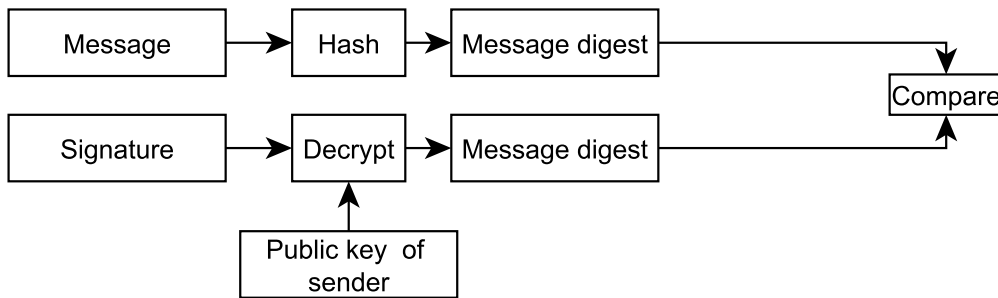
### 2.1.4 Digital Signature

In terms of public key cryptography, a digital signature is a cryptographic procedure that basically encrypts the hash value of a bit stream (also called message or electronic document) with a private key and appends the result to the input of the signing procedure. Therefore, it is message- as well as signer-dependent which ensures *authentication*, *non-repudiation* and *integrity* [48]. Thus, a possible receiver can be sure, that the signed message really originates from the sender and has not been altered during transmission. Figure 2.2a illustrates the basic steps within a signing procedure whereas figure 2.2b presents the verification process.

However, this is one of the most important advantages of Open Badges, because according to their definition, badges can be signed. An earner as well as any other party can than validate the receipt of earned badges, which is not possible for traditionally used, not digitally signed, certificates.



(a) The process of digitally signing a message. The dashed box represents an optional path used if the message to sign should get encrypted *for* a certain receiver.



(b) The process of verifying digitally signed messages.

Figure 2.2: Basic steps within the digitally signing and verification process.

### 2.1.5 JSON Web Signature

A JSON<sup>5</sup> Web Signature (JWS) “represents content secured with digital signatures or Message Authentication Codes (MACs)<sup>6</sup> using JSON-based data structures”<sup>[12]</sup>. JWSs are strings structured by three parts which are separated by a single dot. These parts are the header, the payload and the signature. The header contains the cryptographic parameters, especially which algorithm was used to secure the content. The payload contains the actual data to sign and the last part, the signature, is a digital signature or MAC over the JWS header *and* payload. Both parts are signed as concatenated string, inclusive their separation dot. To ensure URL- and filename compatibility, corresponding parts are base64url (see section 2.1.2) encoded before they get signed. Figure 2.3 illustrates the construction of a JWS within the context of Open Badges. The recommended algorithm combination by the Open Badges specification is due to compatibility reasons “RSA-SHA256”. Thereby, RSA is used for encryption and SHA256<sup>7</sup> for hashing.

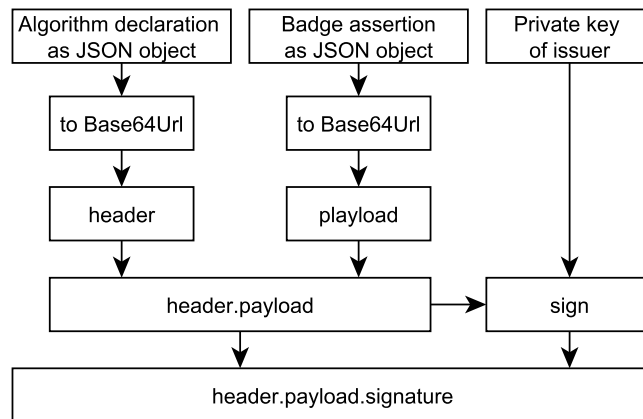


Figure 2.3: JWS as single dot separated concatenation of the Base64 encoded header, payload and signature.

## 2.2 Physical Badge vs. Digital Badge vs. Open Badge

The word badge (pronounced [baj]) has no distinctive definition and its etymology is not clear, but at least Boutell [11] already mentioned a badge in his work about the “English heraldry”. There, a badge was a sign of affiliation to an individual or a

<sup>5</sup>JavaScript Object Notation <http://json.org/>, August 22, 2015.

<sup>6</sup>Similar to digital signatures, except that sender and receiver share a common secret.

<sup>7</sup>Secure Hashing Algorithm <https://tools.ietf.org/html/rfc4634>, August 22, 2015.

## 2 Open Badges

household worn on parts of a person's clothes.

Considering *analog/physical badges*, that definition is still valid, although it is no longer restricted to individuals and households. Probably most famous examples are the American Boy Scouts (figure 2.4a<sup>8</sup>), soldiers (figure 2.4b<sup>9</sup>) as well as club, national or association<sup>10</sup> badges on athlete's jerseys (figure 2.4c<sup>11</sup>).



Figure 2.4: Famous examples of physical badges.

The digital equivalent of such badges is called *digital badge*<sup>12</sup>. Although their meaning is essentially the same as for analogous badges, these *digitally created artifacts* are typically used as incentive- or credentialing instrument within games, online social platforms and meanwhile also in learning environments.

A common application is to steer user behavior through publicly honoring user commitment. The more active or productive a user is, the more acknowledged and esteemed he or she becomes within corresponding communities. That should strengthen the motivation as well as to prize users for their contribution and commitment.

Examples may be StackOverflow<sup>13</sup>, which is a popular Q&A website that issues badges for certain activities. Those can be asking good questions, giving valuable answers or correcting other contributions. Depending on the amount and quality

<sup>8</sup><http://www.somdnews.com/article/20130403/NEWS/130409874/1059/>, July 10, 2015

<sup>9</sup><http://www.edudemic.com/guides/the-teachers-guide-to-badges-in-education/>, July 10, 2015

<sup>10</sup>The Fédération Internationale de Football Association (FIFA) "World Champion Badge" for example, which is demonstrated in figure 2.4c, has been awarded to the Italians Men National Football team in 2006. Note, that in contrast to a personal badge, this one represents an achievement of an association. The important difference is, that even if the whole team would have changed a day after their victory on July, 9th in 2006, the award would still have been valid for the team until the next World Cup in 2010, because the badge was not awarded to individual players, but to the national team itself.

<sup>11</sup><http://www.fifa.com/worldcup/photos/galleries/y=2008/m=9/gallery=the-fifa-world-champions-badge-868165.html>, July 10, 2015

<sup>12</sup>Note, that just taking a picture of an analog badge is not considered to be a digital badge.

<sup>13</sup><http://stackoverflow.com/>, August 22, 2015.

## 2 Open Badges

of performed activities, the user earns bronze, silver or gold badges<sup>14</sup>. The more badges one has earned, the more esteemed that user becomes within the StackOverflow community. The principle is used in various other platforms too, differencing just in design and certain activities to perform.

A more game-like, but more popular application for digital badges was FourSquare<sup>15</sup>, which was a location-based application that awarded badges to users that *checked-in* at certain locations. Depending on how frequent and diverse their check-ins were, different badges were awarded. Figure 2.5 shows a couple of achievable badges and states their criteria to get them.



Figure 2.5: Some digital badges of the Foursquare platform (taken from <http://www.4squarebadges.com/foursquare-badge-list/> on July 10, 2015). These were awarded if the user checks-in: (a) the first time, (b) 10 times (c) 25 times, (d) four days in a row, (e) three times in one week at the same place and (f) 30 times within a single month.

Within the educational context, these digital badges are usually used for credentialing, which means to acknowledge learned skills and honor achievements. Popular representatives are the educational initiative Khan Academy<sup>16</sup> as well as open course platforms from universities like edX, Coursera and CourseWare.

The limitation of digital badges is that they are platform dependent and neither exportable nor do they follow a common standard which makes it impossible to store and manage earned badges in a single place. Although, since Facebook<sup>17</sup>, Twitter<sup>18</sup>, Instagram<sup>19</sup> and these kind of social networks, sharing has become more and more important and meanwhile many platforms provide the possibility to automatically share to such websites, there has been no common standard so far allows a user to collect and share his or her digital badges in one single place. In addition, no digital badge, no matter where it is from, is verifiable as it is just an image that can be easily counterfeit.

<sup>14</sup>A list of badges is given at <http://stackoverflow.com/help/badges>, 22 August, 2015.

<sup>15</sup><https://de.foursquare.com/>, August 22, 2015. Foursquare outsourced the social aspect (check-ins) to a new application named Swarm <https://de.swarmapp.com/>, August 22, 2015.

<sup>16</sup>Khan Academy badges <https://www.khanacademy.org/badges>, August 22, 2015.

<sup>17</sup><https://www.facebook.com/>, August 24, 2015.

<sup>18</sup><https://twitter.com/>, August 24, 2015.

<sup>19</sup><https://instagram.com/>, August 24, 2015.

## 2 Open Badges

In 2011, Mozilla<sup>20</sup>, which is a non-profit organization, started to change this. Financially supported by the MacArthur Foundation, Mozilla invented an open technical standard for digital badges which specifies the meta-data each badge must contain. Meta-data is "*structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource*" [43]. It links back to the issuer, the badge description and the criteria to fulfill to earn this badge. Badges that follow the specification are then called *Open Badges*. Due to the open technical standard, everyone is able to issue, earn and display digital badges which also allows to verify their receipt. But it is not only about verification. Open Badges also allow to manage earned badges from multiple issuers in a single place and sharing becomes as easy as it should. Earners can assign badges to collections and manage their visibility separately.

So the developed standard is a huge step forward to form a sort of digital badge economy which has the potential to also be a serious alternative to common formal credentialing approaches.

However, Mozilla names the round-up of all tools, specifications and components *Open Badges Infrastructure* (OBI), which is described in the following section in detail.

### 2.3 Open Badges Infrastructure

The Open Badge Infrastructure (OBI) supports issuing, managing and displaying of badges by specifying the meta-data structure and providing tools to design, issue, verify, store and share badges. Figure 2.6 illustrates the structure and relation of the three completely independent components, namely the issuers, backpacks and displayers. In this context, *independence* means that each of those components does not care what is happening at what instance, as long as the transferred data aligns with the defined standard. For example, the backpack does not care about who issued the badge it should import and the issuer does not care about which backpack is used to store the awarded badge. Analogous to that, a displayer does not care about who issued the badge or which backpack stored it, it simply takes the data and displays it.

Badges that acknowledge a set of skills instead of particular ones are called *Meta-Badges*. For example, passing courses about the programming languages C, Java and Python could lead to a meta-badge called "*Programming Basics*" as the completion of those three courses represents a set of basic programming language skills. Badges that are part of a meta-badge and represent a specific skill are called *Micro-Badges*.

---

<sup>20</sup><https://www.mozilla.org/de/>, August 22, 2015.

## 2 Open Badges

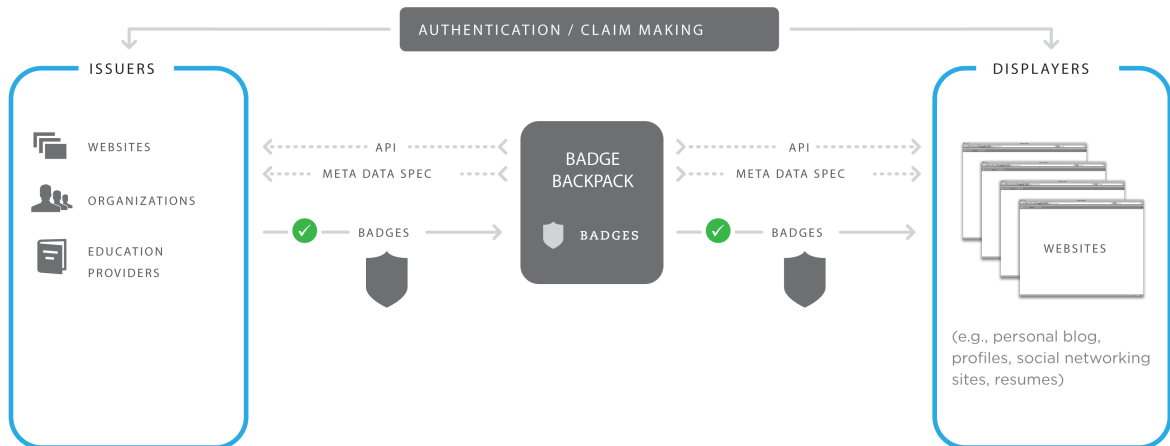


Figure 2.6: The OBI eco system which clearly illustrates its range of duty (image taken from Mozilla Wiki <https://wiki.mozilla.org/Badges/Onboarding-Issuer> on August 12, 2015).

The remaining section introduces each of those components. As all of them rely on the meta-data structure, this part is described first and in more detail.

### 2.3.1 Meta-Data Specification

An Open Badge consists of two main parts, the badge image and the meta-data which is attached to the image file, called *badge assertion* (details in section 2.3.1.7). The image represents the badge visually and should already reveal the badges' purpose. However, it basically could be any image. Attached meta-data basically states *who* issued *whom* *which* badge. Without this meta-data part, the badge would be a simple digital badge.

The badge assertion itself is encoded as JSON object (an example is shown in listing 2.10) and contains embedded as well as links to several other data. Awarding a badge means to generate the corresponding badge assertion. Figure 2.7 illustrates how certain objects relate to each other. Each of those get briefly described in the subsequent sections. The three objects *Badge Assertion* (see 2.3.1.7), *Badge Class* (see 2.3.1.6) and *Issuer Organization* (see 2.3.1.2) form the often referenced *core objects* of an Open Badge.

In early May, 2015, the Badge Alliance<sup>21</sup>, a Mozilla working group maintaining and improving Open Badges, published the current version<sup>22</sup> of the meta-data specification [7], introducing the recommendation of using JSON-LD<sup>23</sup> to get rid

<sup>21</sup><https://www.badgealliance.org/>, August 22, 2015.

<sup>22</sup>Version 1.1 <https://openbadgespec.org/>, August 22, 2015.

<sup>23</sup>W3C recommended lightweight syntax to serialize linked data in JSON <http://www.w3.org/TR/json-ld/>, August 22, 2015.

## 2 Open Badges

of the ambiguity problem by mixing JSON data from multiple sources. The core idea was to give data context. Each *core object* should therefore include the following three<sup>24</sup> attributes.

- **@context:** Is used to reference a certain schema that tells the processing application how to interpret included data. The official one is currently from the badge alliance (<https://openbadgespec.org/v1/context.json>, August 22, 2015) but could theoretically be also any community or private schema
- **@id:** Identifies the object itself by Internationalized Resource Identifiers (IRI<sup>25</sup>). Therefore, in case of hosted badges, it refers to its location on the issuer's web server. In case of signed badges that token is omitted
- **@type:** Declares the type of a certain data object. In this context, that is either "Assertion", "BadgeClass" or "IssuerOrg"

Those additional attributes allow Open Badges to be better indexed by search engines as well as to link individual or community based meta-data declarations [7]. JSON-LD does not only improve data validation but also introduces highly customization to badge meta-data.

As those alterations were made not very long ago, scarcely anybody has implemented them already. However, because the current version is downwards-compatible, the developed web application, described in chapter 5, already implemented them, if only on a basic level.

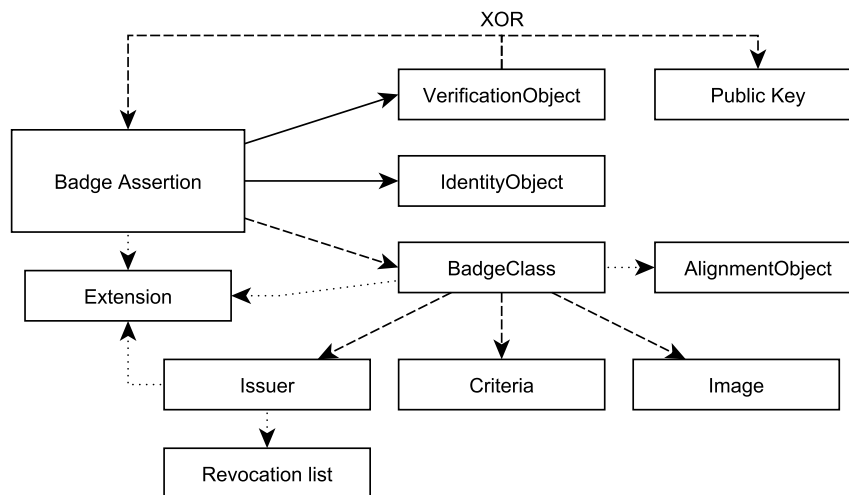


Figure 2.7: Relation of certain data objects. Solid lines represent embedding, dashed lines indicate URLs and the dotted lines represent optional relations.

<sup>24</sup>See <http://www.w3.org/TR/json-ld/> for a complete list, August 22, 2015

<sup>25</sup><https://www.ietf.org/rfc/rfc3987.txt>, August 22, 2015.

### 2.3.1.1 Hosted vs. Signed Badge

The specification defines two types of badges, namely *hosted* and *signed*. A badge is said to be hosted if the corresponding assertion is available as plain text JSON string on a publicly accessible URL of the issuer's (see sections 2.3.1.2 and 2.3.5) web server. In addition, that badge must not be digitally signed. One can verify such badges by simply checking if the corresponding assertions exist on the issuer's web server or not. Therefore, the verification URL contains the link to the assertion file or at least to a Servlet that returns corresponding data.

In contrast, a badge is said to be signed if its assertion is represented as JWS. Neither its plain text assertion nor its JWS representation have to be stored as file on the issuer's web server, as it is the public key that verifies such badges.

The advantage of signed badges is the assurance of data integrity, authority and non-repudiation (see section 2.1.5 for details).

### 2.3.1.2 Issuer Organization

This JSON object represents the issuer and is usually presented to the user in backpacks or on displayer websites. Its purpose is to instantly indicate where the badge is from. Therefore, the JSON object has to contain at least

- the name of the organization, which can be any string
- the URL to the issuing platform, which is usually the issuer's website

Optionally, to appear more informative to the viewer, the following data is suggested to be declared too

- a short description presenting the issuer
- an image which might be the company logo or an emblem of the institution
- an email address as contact point

In addition, the issuer can revoke badges. Displaying web sites can access that information and ignore revoked badges. Hosted badges are verified by their location on the issuer's website. If a hosted badge was revoked, that URL returns just the revoking information as JSON object instead of its badge assertion (see listing 2.1).

```
1 {  
2   "revoked": true  
3 }
```

Listing 2.1: URL response of a revoked badge.



## 2 Open Badges

For signed badges, each issuer may maintain a revocation list. That list is represented as JSON object (see listing 2.2) where the keys are the unique identifiers of the awarded badge instance and corresponding values state the reason for revocation.

```
1 {
2   "1028237_108482" : "mistakenly issued",
3   "029999a_421234" : "wrong email address"
4 }
```

Listing 2.2: Example revocation list containing two revoked badges.

Issuer information, except the revocation list, is static for each platform and the corresponding file has to be placed on a stable and publicly reachable server address. The following listing shows an example of such a JSON object.

```
1 {
2   "@context"      : "https://w3id.org/openbadges/v1",
3   "@id"           : "http://example.org/example.json",
4   "@type"         : "IssuerOrg",
5   "name"          : "Some Organization Name",
6   "url"           : "http://example.org/",
7   "description"   : "This is an example organization",
8   "image"         : "http://example.org/images/logo.png",
9   "email"         : "contact@example.org",
10  "revocationList": "http://example.org/revokedBadges.json"
11 }
```

Listing 2.3: Example issuer data encoded as json object.

### 2.3.1.3 Alignment Object

States which educational standards the corresponding badge aligns to. Educational standards define what students should know and which skills they should have at certain educational levels. Several initiatives<sup>26</sup> provide such definitions for different subjects. An example may be the *Common Core State Standards Initiative*<sup>27</sup> which focuses on primary and secondary levels in the USA.

Each alignment object contains the following information

- the name of the alignment object
- the URL linking to the standards definition
- (optionally) a short description of the standard

---

<sup>26</sup>An overview of standards used in the USA might give the University Library of Illinois <http://www.library.illinois.edu/sshel/education/educstandards.html>, August 22, 2015.

<sup>27</sup><http://www.corestandards.org/>, August 22, 2015.

## 2 Open Badges

These objects may be useful for badges used to acknowledge or certify students based on certain achieved knowledge or competency levels. Listing 2.4 shows an example, taken from the Open Badges specification examples.

```
1 {
2   "name" : "CCSS.ELA-Literacy.RST.11-12.3",
3   "url"  : "http://www.corestandards.org/ELA-Literacy/RST/11-12/3",
4   "description": "Follow precisely a complex multistep procedure when
   carrying out experiments, taking measurements, or performing
   technical tasks; analyze the specific results based on
   explanations in the text."
5 }
```

Listing 2.4: Example alignment object.

### 2.3.1.4 Identity Object

Represents the identity information of the recipient - the badge earner. Open Badges get asserted solely to email addresses. That this could lead to problems, if only in rare cases, will be discussed within the concluding chapter (section 7.1). Nevertheless, there is currently no better alternative out there so the email address of the recipient is the only connection to the badge earner. Hence, this object has to contain following data:

- the identity type
- the actual identity of the recipient, either as plain text (see listing 2.5) or its hashed version (see listing 2.6, but note that the hash value was shortened out of lucidity reasons.)
- an indicator if the email address was hashed or not. If yes, then
  - the resulting string is constructed out of the used algorithm (e.g. SHA256) followed by a dollar sign (\$) and the hash string (refer to figure 2.1 for an illustration of that process)
  - possibly added salt

```
1 "recipient" : {
2   "type"      : "email",
3   "identity"  : "example@ex.org",
4   "hashed"    : false
5 }
```

Listing 2.5: Example Identity Object with a plain text email address.

## 2 Open Badges

```
1 "recipient" : {
2   "type"      : "email",
3   "identity"  : "sha256$f61f427de298bbc...cf3bf06cd0dcffb",
4   "hashed"    : true,
5   "salt"     : "0b338c582330744c9e5a59fa13e1d4830d6c"
6 }
```

Listing 2.6: Example Identity Object with a hashed recipient.

### 2.3.1.5 Verification Object

Depending on the badge type (refer to section 2.3.1.1 for details), either the URL of the badge related assertion (see listing 2.7) or the URL of the public key (see listing 2.8) is returned. Therefore, a Verification Object contains two attributes:

- the type of the badge assertion (either "hosted" or "signed")
- and the validation URL

```
1 "verify" : {
2   "type" : "hosted",
3   "url"  : "http://example.org/a-certain-badge-assertion.json"
4 },
```

Listing 2.7: Example Verification Object for a hosted badge assertion.

```
1 "verify" : {
2   "type" : "signed",
3   "url"  : "http://example.org/public-key.pem"
4 },
```

Listing 2.8: Example Verification Object for a signed badge assertion.

### 2.3.1.6 Badge Class

The so-called *badge class* actually describes the achievement and contains all publicly viewable data, which in particular are:

- the name of the badge which has according to the specification no restrictions. However, it certainly makes sense to keep the name quite short and to choose one that obviously states what that badge represents. Ideally, the name and the image form a clear statement not just for the earner but also for external viewers
- a short description mentioning more details than the image and the badge name can represent

## 2 Open Badges

- an URL to the image associated with the badge, which should be square, not exceed 256 kb in size but always be greater than 90x90 pixels
- an URL to corresponding criteria which could basically be of any file format. Recommended ones are JSON or HTML files, where links to additional sites or files is also allowed
- an URL to the issuing institution (see 2.3.1.2 for details)

Optionally, the badge owner can add alignment objects (see 2.3.1.3 to details) and tags to the badge. Tags can be used to label the badge, which could enable users to search for specifically labeled badges on appropriate websites, like the *Open Badges Dictionary*<sup>28</sup> or within the issuing site itself. A badge class JSON object may look like listing (2.9).

```
1 {
2   "@context" : "https://w3id.org/openbadges/v1",
3   "@id"      : "http://example.org/badges/xy.json",
4   "@type"    : "BadgeClass",
5   "name"     : "Deeply Committed Student Badge",
6   "description" : "This badge is dedicated to ...",
7   "image"    : "http://example.org/images/xy.png",
8   "criteria" : "http://example.org/criteria/xy_criteria.json",
9   "issuer"   : "http://example.org/someIssuer.json",
10  "tags"     : ["Commitment", "Bonus", "2015"]
11 }
```

Listing 2.9: Example Badge Class object.

### 2.3.1.7 Badge Assertion

An assertion is a mapping of a specific badge (2.3.1.6) to a certain earner (2.3.1.4). Figure 2.7 illustrates how these data objects, among others, relate to each other to form the badge award. The resulting JSON object (an example is given in listing 2.10) then represents a fully<sup>29</sup> valid Open Badge.

In addition to the already defined data objects (see sections 2.3.1.2 - 2.3.1.6), an assertion also has to contain:

- a unique identifier which should be locally unique on the hosting server to be able to identify assertions for revoking and validation
- the date and time the assertion got generated, either as ISO 8601<sup>30</sup> date time or as 10-digit UNIX time stamp

<sup>28</sup>Currently a prototype <http://directory.openbadges.org/examples/browser/#/search>, August 22, 2015.

<sup>29</sup>Because the image is already part of the badge assertion.

<sup>30</sup><http://www.ietf.org/rfc/rfc3339.txt>, August 22, 2015.

## 2 Open Badges

All used URLs have to be publicly accessible, otherwise backpacks will not accept the Open Badge. The final structure of an assertion can be checked by Mozilla's *OpenBadges Metadata Validator*<sup>31</sup>

```
1 {
2   "@context" : "https://w3id.org/openbadges/v1",
3   "@type"    : "Assertion",
4   "uid"      : "61f1d38000_1434964589",
5   "recipient" : {
6     "type"    : "email",
7     "identity" : "sha256$60534...2a8bde23588c832101cad",
8     "hashed"  : true,
9     "salt"    : "2299b13d42e8c7b78c64b587adb22"
10  },
11  "badge"    : "http://someurl/exampleBadgeClass.json",
12  "verify"   : {
13    "type"    : "signed",
14    "url"     : "http://someurl/public-key.pem"
15  },
16  "issuedOn" : 1434964589
17 }
```

Listing 2.10: Example decoded signed badge assertion (identity hash has been shortened)

However, this object can additionally contain:

- the already baked image (must be the URL to a PNG file)
- a URL to the evidence to earn the badge (can also be a HTML file that links to further sites)
- the information when the badge expires. After that expiration date, the assertion should be treated as invalid

### 2.3.1.8 Extensions

Version 1.1 also introduced the concept of *Extensions*, which enables issuers to add additional meta-data beyond those of the Open Badges specification. For example, one could add the course location or its lecturer to a certain badge. Although it was already allowed to add custom parameters to each of the described entities in the initial release, there was no protection against naming conflicts or misinterpretation of their purpose or data format. Extensions avoid that problem by building schema-validated JSON-LD objects embedded in any of the Open Badge core objects. Therefore, each Extension contains an attribute *@context* which links to the schema that declares the purpose, semantics and structure of corresponding parameters.

<sup>31</sup><http://validator.openbadges.org/>, August 22, 2015.

## 2 Open Badges

Attribute *type* is an array of data type declarations of data contained in the Extension. Following that declaration, the actual data gets added to the JSON object.

Listing 2.11 shows an example schema which defines one property with the name "exampleProperty" of type "text". What that "text" is describes the schema defined by the given URL.

Listing 2.12 shows the already *embedded* Extension object, declaring the "exampleProperty" of listing 2.11 to be "I'm a property, short and sweet".

```
1 {
2   "@context": {
3     "obi": "https://w3id.org/openbadges#",
4     "exampleProperty": "http://schema.org/text"
5   }
6 }
```

Listing 2.11: Example Extension (taken from official specification).

```
1 {
2   "extension:ExampleExtension": {
3     "@context": "https://openbadgespec.org/extensions/exampleExtension
4     /context.json",
5     "type": ["Extension", "extensions:ExampleExtension"],
6     "exampleProperty": "I'm a property, short and sweet."
7   }
8 }
```

Listing 2.12: Example Extension object (taken from official specification).

### 2.3.1.9 Validation

Also since version 1.1, Validation objects can be embedded in assertions, badge classes and issuer instances. Their purpose is to test if generated JSON objects are valid against a certain schema. Currently, just type validation is supported, which may look like listing 2.13.

```
1 {
2   "validation": [{
3     "type": "TypeValidation",
4     "validatesType": "Assertion",
5     "validationSchema": "https://openbadgespec.org/v1/schema/
6     assertion.json"
7   }]
8 }
```

Listing 2.13: Example Validation that links to a validation schema for badge assertions (taken from official specification).

### 2.3.1.10 Endorsement

The concept of badge endorsement [6] plays an important role in how to priorities badges. External organizations (stakeholders) can publicly indicate which badges align to their needs and which they find to have most value for them. Earners can then search and seek for particular job or organization endorsed badges. Issuers get feedback from organizations if their defined badge has value to them or not. Viewers get supported by determining the value of a badge more easily. So in short, everybody wins.

Technically, issuers provide third-parties the possibility to endorse their badges. Thereby, an *Extension Object* is added to the Badge Class of interest. If an issuer then awards an instance of such an endorsed badge, the endorsement information is contained. So basically, depending on the issuer's implementation, controlling who is actually able to endorse a badge and where this information is displayed is completely up to the issuer.

A possible implementation would be to receive endorsement candidacies through a publicly accessible button or via email and, depending on the applicant, the issuer decides if the concerning badge should be endorsed or not.

How to realize that concept is, at least to that date, still in its infancy. Much more research has to be done to answer questions like what happens to already awarded badge instances of the newly endorsed badge. If the corresponding data is simply added to the Badge Class definition, then only new awards will benefit of the endorsement. Another issue to resolve will be how to assess endorsers. Letting everybody endorse a badge will make it petty whilst only considering high-class institutions like the MIT<sup>32</sup> or NASA<sup>33</sup> may lead to never get endorsed.

### 2.3.2 Image-Baking

Open Badges are images with embedded meta-data. The process of writing that meta-data to the image file is called *baking*. Within the OBI, supported file formats are Scaled Vector Graphics (SVG) and Portable Network Graphics (PNG). The reason has not been stated, but it might be their relatively simple handling, their stability and scalability as well as their broad application.

---

<sup>32</sup>Massachusetts Institute of Technology <http://web.mit.edu/>, August 22, 2015.

<sup>33</sup>National Aeronautics and Space Administration <http://www.nasa.gov/>, August 22, 2015.

## 2 Open Badges

### 2.3.2.1 SVG baking

In the case of SVG files, baking is quite simple because it suffices to embed an additional markup element with the tag-name *openbadges:assertion* within the SVG data declaration. The badge type, hosted or signed, influences which data gets written to the additional markup element. In case of hosted badges, the `verify` attribute (see listing 2.14) carries the URL to the assertion on the issuer's web server and the Character Data (CDATA) block embeds the plain text assertion string. In case of signed badges, the `verify` attribute carries the JWS string and the CDATA block is considered to be blank.

```
1 <openbadges:assertion verify="https://example.org/assertion.json">
2   <![CDATA[
3     // assertion plain text goes here ...
4   ]]>
5 </openbadges:assertion>
```

Listing 2.14: Structure of the badge image's SVG child element.

### 2.3.2.2 PNG baking

The PNG file format is a bit more complicated, but simple enough that nearly every programming language has appropriate libraries one can use to embed meta-data. Utilizing one of those, one has to insert an *iTXt Chunk* with the keyword *openbadges*. The text can either be the uncompressed plain text assertion or the corresponding JWS string.

### 2.3.2.3 OBI Baking Service REST API

Mozilla provides a *baking API* as part of their backpack. One can simply call a URL passing the link to the assertion to bake as parameter. If the given assertion is valid, the API returns the baked image file. A sample call is given in listing 2.15.

```
1 http://backpack.openbadges.org/baker?assertion=http://x.com/assertion.json
```

Listing 2.15: Sample call to the REST baking API of the Mozilla backpack.



### 2.3.2.4 Open Badges-Bakery Tool

This is a *Node.js*<sup>34</sup> module which has to be installed and can then be used<sup>35</sup> via the command line or directly from the application code to bake a badge.

### 2.3.2.5 Issuer-API

Depending on the actual implementation of the front-end, the issuer or the earner can make use of the OBI provided Issuer-API to push badges directly to corresponding backpacks. A typical use-case is that the issuer front-end side offers the earner the option to push the earned badge directly to its preregistered Mozilla backpack. If that is the case and the user confirms that action, the implementation connects to the backpack and pushes just the assertion string. As all needed information exists within the assertion (refer to section 2.3.1.7 for details) the backpack does not need to have the image separately. If the user wants to have a baked badge, he or she can then simply export it from the backpack.

## 2.3.3 Backpack

Backpacks are used to store, collect and manage badges by the earner. Such a backpack, especially the one from Mozilla, offers the opportunity to order earned badges within so-called *collections*. Each collection has its own URL which can be shared across the Internet. This makes it particularly easy to share some badges only with friends and others with colleagues.

To that date, there are no real alternatives to the Mozilla Backpack out there. One of the most promising seems to be *OpenBadgePassport*<sup>36</sup>, but currently they fail to import signed badges. Because the principle of backpacks is pretty simple, it can be expected that there will be some more in near future. As backpacks are intended to be single access points for all of ones Open Badges, each earner will probably choose *one* favorite. So it will be primarily due to design, managing and sharing features which backpack will become the most chosen one.

---

<sup>34</sup>Node.js platform <https://nodejs.org/>, August 22, 2015.

<sup>35</sup><https://github.com/mozilla/openbadges-backpack/wiki/Badge-Baking#open-badges-bakery>, August 22, 2015.

<sup>36</sup><https://openbadgepassport.com>, August 14, 2015.

### 2.3.4 Displayer

A displayer in the context of the OBI is any website that uses the *Displayer API*. A displayer's purpose is to load badges from a shared source, like a backpack, and present their contained badge meta-data. Which badges can be displayed is controlled by sharing options the earner can set in the backpack settings. Displayers are scarce to date, but more and more websites try to implement features to let users show and share their achievements. Examples are Mahara<sup>37</sup>, LinkedIn<sup>38</sup>, Credly<sup>39</sup> and OpenBadgeFactory<sup>40</sup>.

### 2.3.5 Issuer

An issuer is any organization or individual that issues Open Badges. Typically, such an issuer defines badges, handles their availability and finally awards them to the earner. Figure 2.8 shows an overview of the responsibilities an issuer has. Technically, issuing simply means to generate an appropriate badge assertion (see

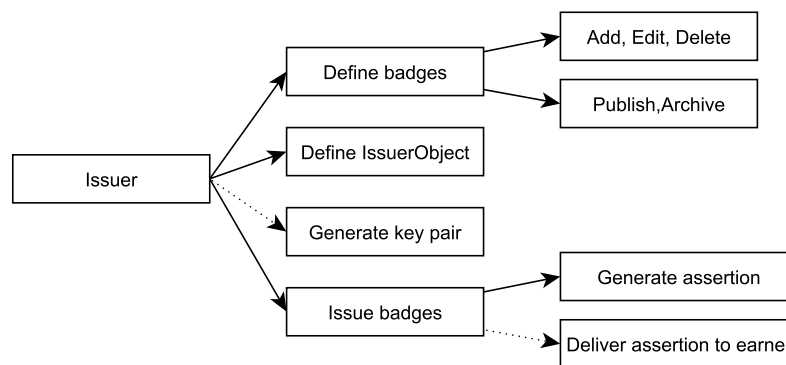


Figure 2.8: Overview of an issuer's responsibilities. Dotted lines represent optional relations. E.g. if the issuer does not generate signed badges, there is no need to generate a key pair. Red bordered tasks are usually done just once.

2.3.1.7) where the issuing process is completed as soon as the assertion has been generated. Delivering the assertion to the user is basically not part of the issuing process, although in practice, it is common to address both if talking about issuing. How the issuer handles and awards badges is up to the issuer, as long as generated assertions, and thereby linked data, are conform to the meta-data specification,

<sup>37</sup><https://mahara.org/>, August 14, 2015.

<sup>38</sup><https://www.linkedin.com/>, August 14, 2015.

<sup>39</sup><https://credly.com/>, August 14, 2015.

<sup>40</sup><https://openbadgefactory.com/>, August 14, 2015.

## 2 Open Badges

which is described in section 2.3.1 in detail. The reason is that only the issuer knows what is best for its community - the earner. So the front-end design and how badges are integrated into the system is purely up to the issuer.

Assuming that an issuing organization wants to award badges on its own, the issuer has to fulfill certain technical requirements if no third party platform should be used.

First, the issuer has to have web hosting facilities to be able to make badges and their meta-data publicly accessible. Therefore, the web server has to be capable of serving requests to the Internet and must be able to respond with JSON data.

Second, the issuer has to have access to the email addresses of its earner because, as already mentioned in section 2.3.1.4, an earner is represented by a corresponding email address.

The third requirement depends on the used badge type. In case of hosted badges, the issuer has to maintain issued assertion files at unique and publicly accessible paths. Once a badge has been issued, the thereby linked paths must not be modified afterwards to ensure that others can verify badges even after years. In case of signed badges, the issuer must provide a public key that corresponds to the private key which was used to sign them. The private key must not change after being used at least once, because otherwise no one can verify badges that have already been signed before.

If an organization decides to outsource that capabilities and to use a third party platform to issue Open Badges, above mentioned requirements are not needed. Nevertheless, the issuing organization still needs to have the email addresses of their earner, define badges inclusive corresponding criteria and, of course, some sort of triggering approach.

### 2.3.5.1 Delivering the assertion to the earner

The OBI provides three possibilities to deliver an assertion to an earner, which also means to get the issued badge out of the issuer's ambit.

The first one is the most unusual one, which is to simply provide the assertion string. This is valid, but is not very user friendly. In particular, an issuer that delivers raw JSON strings to the earner forces the earner to use a backpack which allows to import assertions or to bake the badge on ones own. Most of the backpacks out there restrict the import to image files<sup>41</sup>, so it is not recommended to choose this option.

The second option is to bake the assertion into the image file and to offer the earner

---

<sup>41</sup>However, an earner could circumvent that by importing the assertion string to the Mozilla Backpack and then importing badges from the Mozilla Backpack in other backpacks (e.g.: [openbadgespassport.org](http://openbadgespassport.org), August 22, 2015), but it is definitely not good usability to force the earner to do so.

## 2 Open Badges

the file as download. Currently, that is probably the most sensible option, because the earner can choose what happens with his earned badge and which backpack he wants to use and does not have to care about baking the image on his own. This is also the option which was chosen in the implementation described in chapter 5. The third option is to utilize the so-called *Issuer-API* provided by the OBI. Using that, the issuer is able to directly push the generated assertion to the registered Mozilla Backpack of the earner. Currently, just Mozilla's Backpack is supported which restricts the freedom of choice for the earner. On the other hand, the earner typically just has to click a button to import the recently earned badge into the corresponding backpack.

## 3 Learning Environments

This chapter clarifies some commonly known but differently interpreted terms and concepts about learning, technologies that support it and ways to assess its progress as well as outcomes.

### 3.1 Learning

The term *learning* is defined differently across various research areas. Within this thesis, the definition of the online-encyclopedia and dictionary Merriam-Webster.com [39] was used which states that learning is "*the activity or process of gaining knowledge or skill by studying, practicing, being taught, or experiencing something*".

Learning does not solely happen in formal settings, like schools, universities and work places. It happens all day long and on various locations, mostly even unintentional. The recognition of these learning forms has earned high attention and priority over the last years. The Europe 2020 strategy explicitly calls for "*the promotion of the recognition of non-formal and informal learning*".

To clarify their definition [38][59], *formal* learning typically happens in educational institutions, is organized, structured, supported, intentional and certified. *Non-formal* learning is similar to formal learning, except it does not happen in educational institutions and typically does not lead to certification. Nevertheless, it is organized, structured and goal oriented. *Informal* learning is what happens in real life situations through made experiences, no matter in which context. It may be goal oriented, but usually, it is unintentional, unstructured, unorganized and not certified.

Psychologists investigated different learning processes and tried to formulate these by learning theories.

### 3.1.1 Learning Theories

Regarding E-Learning, the three most important learning theories, namely behaviorism, constructivism and cognitivism, are briefly described for better understanding the learning environments presented at a later stage of this thesis. Connectivism, which is a still not scientifically accepted distinct learning theory is also presented as kind of additional, but not equivalent, learning theory as it is important for example for cMOOCs.

#### 3.1.1.1 Behaviorism

This model is primarily based on the work of Iwan Pawlow<sup>1</sup> (1849-1936) who observed the conditioning of a dog's salivary production. After a couple of test runs, where the dog got meat powder in his mouth, Pawlow noticed that the dogs salivary production started already as soon as the feeding tester entered the room. Pawlow investigated that behavior and named this phenomenon *conditional reflex*. He postulates, that it is possible to combine unconditional reflexes (like the salivary production) with conditional ones (tester entered the room). This combination is known as *respondent or classical conditioning*. Edward L. Thorndike (1874-1949), John B. Watson (1878-1958) and Burrhus Frederic Skinner (1904-1990) focused on improving learning results by amplifying behavior consequences, which is called *operant conditioning* [33, 53].

Behaviorism therefore describes conditioning by rewarding on success and punishment on failure. Rewarding means to introduce positive or to remove negative stimuli (positive amplification) whereas punishment adds negative or removes positive stimuli (negative amplification). Controlling these actions influences the probability that a certain behavior occurs.

In education, behaviorism describes teaching factual knowledge. Usually, the learner gets instructed by a tutor or teacher in detail and the current learning progress is tested by exam questions where the answer is either correct (=success) or not (=failure). So the focus is on the output of the learning process, not on the learning process itself [60].

#### 3.1.1.2 Cognitivism

Humans do not just learn from reinforcements. They also learn from observing others and how they solve tasks. Such learning is commonly known as *observational*

---

<sup>1</sup>Nobel price owner in medicine 1904.

## 3 Learning Environments

*learning*. The corresponding learning theory is called *cognitivism*. This model combines cognition with emotions, also called *learning by discretion*. According to Sauter and Kulmann [53] these observations are processed in complex cognitive processes which lead to understanding and models of own behavior. This kind of learning happens everywhere and many times without even having the intention to learn. So in contrast to the behaviorism model, it is not about answering right or wrong, it is about understanding how to solve given tasks.

Therefore, according to Vogt and Hechenleitner [60], in education, cognitivism is learning by understanding. The focus is on *how to do something*, instead of pure factual knowledge. *Doing something* thereby means to choose appropriate tools and methods to solve a certain task, whereby actually solving the task then follows the "trial and error" principle. The real objective is to find *any* path instead of *the one* towards the target.

### 3.1.1.3 Constructivism

In contrast to the before mentioned models, constructivism does not tutor the learner excessively, but instead, provides the learner with an environment to test and to tinker with something ("learning by doing"). The brain permanently interprets perceptions from all its sensory organs. This makes the brain *simulate* the environment, without knowing how it really looks like [53]. To understand something therefore means to interpret it in a way that it makes sense, regarding one's experiences. So the word constructivism is based on the active construction of subjectively meaningful interpretations.

In education, constructivism is also known as *social learning*, as it focuses the communication within a community. Multiple learner discuss and share information on topics without explicit tutoring by a teacher. Therefore, knowledge is acquired by common projects and discussions with other learners instead of excessive instruction by teachers. The teacher provides the learning environment, acts as moderator and only intervenes if the learning processes of the group tend to run in an unfavorable or false direction. However, the learning process gets controlled by the learner [35, 60].

### 3.1.1.4 Connectivism

Siemens [55] introduced this learning theory due to the ongoing usage of the Internet as learning medium. He highlighted that informal learning has become more and more important and that the way people think and act has changed because

### 3 Learning Environments

of the increasing possibilities due to technology development. As a consequence, individual learning merges with organized learning. Therefore, behaviorism, cognitivism as well as constructivism do not meet the needs of modern learners. Knowledge increases exponentially and individuals are not able to make every experience on their own. Due to the huge amount of available knowledge, *what* one learns depends on the corresponding environment. Knowledge is renewed as well as lost quite frequently. This theory therefore focuses on where to find needed information, instead of knowing everything. As a consequence, one's environment fitted network of persons, databases and other information sources is more important than learning factual knowledge or understanding how something exactly works. Within this concept, it is essential to be able to distinguish between relevant and irrelevant information which requires interconnections between multiple and different areas. That is fortunately supported by social media platforms and networks. So the learning process consists of building and maintaining connections. Although connectivism is in wide use and is multiply referenced within current literature, it has not yet been officially accepted as distinct learning theory.

#### 3.1.2 Motivation

As motivation is a critical factor in one's learning process as well as for using Open Badges, the term is shortly explained.

Ryan and Deci [50] define a motivated person as someone who is "*moved to do something*". The Self-Determination Theory (SDT) [17] states two basic types of motivation, namely the *intrinsic* and *extrinsic* motivation. *Intrinsically* motivated people act out of conviction and inherent satisfaction, not for external rewards or pressures. The Cognitive Evaluation Theory (CET) [17] as sub theory of the SDT argues, that social events (e.g. rewards or feedback) that contribute to conceive competence can enhance intrinsic motivation. In addition, CET states, that it is not enough to just perceive competence, "*people must also experience their behavior to be self-determined*" [50].

*Extrinsic* motivation in contrast is doing something in order to attain some certain outcome, which, according to SDT, varies eminently in the degree of autonomous. As this may be hard to understand Ryan and Deci [50] described that with an unblemished example:

*"a student who does his homework only because he fears parental sanctions for not doing it is extrinsically motivated because he is doing the work in order to attain the separable outcome of avoiding sanctions. Similarly, a student who does the work because she personally believes it is valuable for her chosen career is also extrinsically motivated because she too is doing it for its instrumental value rather than because she finds it interesting.*



### 3 Learning Environments

*Both examples involve instrumentalities, yet the latter case entails personal endorsement and a feeling of choice, whereas the former involves mere compliance with an external control. Both represent intentional behavior, but the two types of extrinsic motivation vary in their relative autonomy."*

Open Badges may serve as such separable outcome that enhances the motivation of learners.

## 3.2 Learning Technologies

According to [62], a *learning space* is every *place*, whether physical or virtual, where someone is actively supported in a learning process. Therefore, a traditional class room is as much a learning space as any library or online course. A *learning environment* is such a learning space equipped with "a wide set of features that affect learning"[62]. Such features have changed over time, primarily due to the giant leap in computer assisted technologies and their interconnection. Fifty years ago, a learning environment was basically either a school, university or a library, where lecturers stood in front of physically present students to teach certain contents. Actively learning was therefore restricted to such buildings, apart from reading books at home. Hence, features were book-rentals, overhead-projectors and of course, the classic blackboard in a class room.

That changed once personal computers became affordable to the general public. It rapidly was the most important feature enabling basic text processing, programming applications and simulating complex algorithms [9, 54]. Therefore, computers were mainly used to teach students certain practical skills. In addition, electronic resources could be stored on portable storage devices and brought home to learn and improve skills also after school. Such computer assisted learning is generally called *Computer Based Training* (CBT).

By increasing interconnection of computers, resources could be provided and shared via the Internet. Students therefore had access to (online) learning resources any time and without spatial and temporal constraints. CBT done online has then been called *Web-Based Training* (WBT). Both can be subsumed to *Computer Assisted Learning or Teaching* (CAL/CAT)[9], depending on the perspective. From a learner's point of view or within the educational context, concepts and environments are usually called *learning*, whereas coaches, trainer and companies call them *training*.

However, through progressing pervasive computing, which means that each person has more and more Internet connected devices, it is possible to access learning resources from anywhere and anytime.

It is common to use the term *E-Learning* (the *E* stands for "electronic") to describe

### 3 Learning Environments

the concept of CAL/CAT. Such technologies are highly dynamic and include various roles, like administrators, teachers, students, IT-staff and others which interact within one system [62].

From the technical point of view, a today's online learning environment is basically a kind of *Learning Management System* (LMS), which is complex software executed on a web server providing user management, courses including their administration and presentation, group policies, communication features (chats, forum) as well as learning supporting features [9, 40, 47]. Its main purpose is to manage users and content to enable and support their learning process.

*Communication* in online learning environments is either *synchronous* or *asynchronous*. Synchronous communication requires all participating parties to be connected at the same time, although one, a few or all attendees are located on different places. Common technologies are instant messaging, *Voice over IP* applications, forums, live lessons (trainings) and virtual class rooms. Asynchronous communication does not require all participants to be online at the same time, which is the more important one in real life scenarios. Email for example is a common technology to use for asynchronous communication, although it could be seen as synchronous communication too if both, sender and receiver, are online at the same time. Usually, that is even the case, but the receiver does not directly respond to the sender so it is basically an asynchronous communication though. A more clearly understandable example would be a classic forum. One asks a question opening a new thread and others can answer, anytime and from anywhere.

However, this thesis focuses on utilizing Open Badges, which are online distributed digital artifacts. Hence, the set of considered learning environments is reduced to online platforms in the educational context which represent potential target environments for Open Badges. As there exist a lot of different terms and concepts, the following sections should present and define the most important ones. Because there is no explicit focus on them, exact definitions and classifications of used technologies and concepts are omitted.

#### 3.2.1 LCMS and CrMS

A derivation of basic LMSs are *Learning Content Management Systems* (LCMS). Those extend LMSs by adding *advanced* authoring tools. The focus of LCMSs is on the creation, maintenance and reuse of *Learning Objects* (LO). Those are any piece of information (image, text, animation, video) with attached meta-data that describes the LO. In addition to most of the functionality of a common LMS, a LCMS provides content versioning, management of complex content objects and cataloging through

## 3 Learning Environments

meta-data tagging. Such systems have the advantage of constructing dynamic course structures out of multiple smaller content objects [9].

*Course Management Systems (CrMS)*<sup>2</sup> are LCMs specialized on courses and corresponding features. The focus of these systems is to provide live instruction-led training which basically means to enable the instructor to post messages, provide additional comments to contents, assign students to groups and to evaluate student work during the online session[9].

However, literature as well as online communities do typically not distinguish between those variants and call all of them just LMS. Popular open source representatives are Moodle<sup>3</sup>, ILIAS<sup>4</sup>, JoomlaLMS<sup>5</sup> and OpenElms<sup>6</sup>. Blackboard Learn<sup>7</sup>, Canvas<sup>8</sup>, Docebo LMS<sup>9</sup>, Oracle Learning Management<sup>10</sup> and SAP Enterprise Learning<sup>11</sup> are proprietary whereby the two last mentioned ones are mainly used commercially instead of in the pure educational context.

### 3.2.2 PLE

Traditional LMS systems do not focus on the learner. Their focus is to provide and manage content inclusive their presentation. To shift the focus from content provision towards the learner brought up a new concept, called Personal Learning Environment (PLE) [5]. Learner support their individual learning process by creating, collecting and managing contents, tools and resources on their own. Therefore, learner have much open space to unfold themselves and to manage individual learning. Creating, collecting and managing is done with a centralized system which allows to integrate external resources, content, applications, contacts and other social media.

As an example, a learner could add a new website to the personal web space within the system. On that website, the learner could integrate (= embed) a Google search

---

<sup>2</sup>These are often falsely abbreviated as *CMS*, but this abbreviation is used for *Content Management System*, which is a very different concept.

<sup>3</sup><https://moodle.org/>, July 23,2015.

<sup>4</sup><http://www.ilias.de/>, July 23,2015.

<sup>5</sup><https://www.joomlalms.com/>, July 23,2015.

<sup>6</sup><http://openelms.org/>, July 23,2015.

<sup>7</sup><http://de.blackboard.com/>, July 23,2015.

<sup>8</sup><http://www.canvaslms.com/>, July 23,2015.

<sup>9</sup><https://www.docebo.com/>, July 23,2015.

<sup>10</sup><http://www.oracle.com/us/products/applications/ebusiness/human-capital-management/053815.html>, July 23,2015.

<sup>11</sup><http://www.sap.com/austria/training-education/learning-software-svc/learn/solutions/enterprise/index.html>, July 23,2015.

### 3 Learning Environments

bar, a Facebook-, Twitter or LinkedIn account, Wikipedia and perhaps various other widgets or applets for converting units or translating languages. Additionally, learning resources from the learner's university LMS, exam results and maybe even course related data could be embedded onto that personal web page. Which resources or applications are used is completely up to the user and highly configurable. So PLEs provide a system to integrate and present certain collected resources.

To present an example, the Technical University of Graz (TUG) implemented such a (mash-up) system which is based on widgets and capable of embedding university related material (through the used LMS) as well as external resources [57]. Figure 3.1 shows an example user page with some installed widgets.

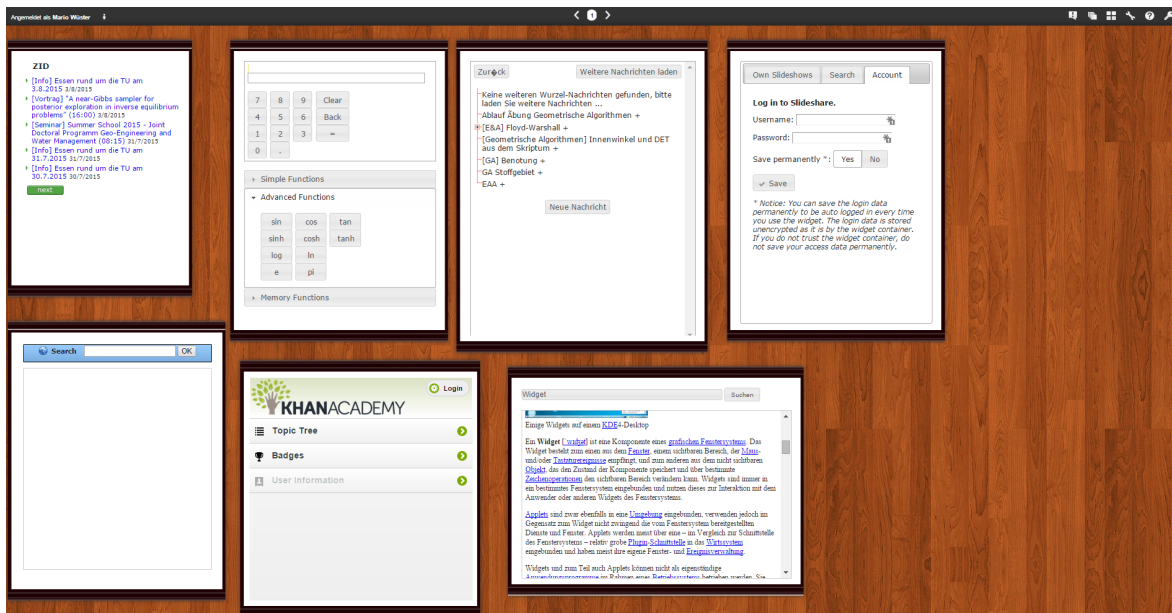


Figure 3.1: Example TUG PLE page with some installed widgets (<http://my.tugraz.at/>, screenshot taken on August 3,2015).

#### 3.2.2.1 Open Badges in PLEs

PLEs typically have one problem. Potential users do not understand what PLEs are capable of and therefore see no overvalue for them. Hence, they do not use it. Promoting an "Explorer Badge" could lead to motivate a user to explore the system in detail which then could help to apprehend the potential of PLEs.

Santos et al. [51] conducted an experiment, not exactly for a PLE but with the same goal, where young students who were also new to their campus system earned a badge for exploring it guided by a challenge-based tutorial. 90% of polled students

appreciated the usage of badges and stated that they felt more motivated towards using the system.

Another application could be to acknowledge active usage. Assuming a student uses the PLE like it is intended, for example at least four times a week with about three quarters of all possible features. To recognize that commitment, the system provider could award the user a "*Commitment Badge*" as acknowledgment and motivation to go on using it that way.

Backpack and displayer functionalities could be implemented to allow the user to manage earned badges from multiple sources and to share them with colleagues or friends university intern or publicly.

A PLE wide web page could be added to show badges from all its users. That would enable a user to explore also badges others have earned. Recommender systems could be implemented to suggest badges to earners, based on badges linked friends and colleagues have earned.

#### 3.2.3 MOOC

Another relatively new concept is the one of Massive Open Online Courses (MOOCs). Those are *exclusively online* courses, mostly on a university level and publicly accessible. Therefore, each course typically has a massive<sup>12</sup> amount of enrollments.

MOOCs have conquered the market continuously over the last few years. The New York Times proclaimed 2012 as the *Year of the MOOC* [46]. In 2015, the French startup *OpenClassrooms*<sup>13</sup> partnered with *IESA Multimedia*<sup>14</sup> and offered the first state-recognized Bachelor degree program that completely relies on MOOCs [19]. MOOCs are so successful because everyone with an Internet connection can participate, mostly even complimentary. Because of missing tuition fees and the fact, that resources like videos and other well-prepared course materials are provided online, there are much less obstacles to learn than if one has to subscribe to a university to get the same.

Over the last years, two main types, the *xMOOCs* and *cMOOCs*, have been established. However, there also exist a couple of specialized MOOCs like for example the *language MOOC* (lMOOC) [37] or *fitness MOOC* (fMOOC) [13]. As those specializations surely justify their usage within particular areas, they do not really differ

---

<sup>12</sup>This quantity varies highly depending on the used platform and popularity of the course topic. But in general, such courses have thousands, if not hundreds of thousands of enrollments.

<sup>13</sup><https://openclassrooms.com/>, July 24, 2015.

<sup>14</sup>A french multimedia school located in Paris which basically offers bachelor programs for multimedia-development and strategies (<http://www.iesamultimedia.fr/>, July 24, 2015).

### 3 Learning Environments

from the two basic types. Therefore, such specializations were omitted within this thesis.

The first and most used MOOC concept is the *xMOOC*. The prefix *x* stands for *extension* and has its origin in the naming convention of the Harvard University and their online course collection. It seems to be a kind of standard in America that programs which are not part of the core offering, but relate in some way to it are indicated by such pre- or suffixed *x*. Examples may be TEDx, edX and MITx which all represent kind of *extension* to the core offer of the respective program.

However, this variant is kind of the basis type which provides learning resources as multimedia content on a single website. Typically, a collection of short<sup>15</sup> videos, each explains a certain topic, form a unit and the learner has to take an exam after each unit in order to proof that the content has been understood. Such exams are usually multiple-choice questions and the examinee has to achieve a certain percentage of correct answers or points. Also kind of homework, textual elaborations or calculations, are common to claim, but due to the huge amount of attendees, those usually do not get assessed by lecturers, but from other learners, then called peer-review. The focus of xMOOCs is on teaching topics and assessing the learner. However, most xMOOC providers additionally offer features like forums or chats to let learners connect and discuss.

A *cMOOC* follows the principle of connectivism<sup>16</sup> (refer to section 3.1.1.4) by combining xMOOCs with an emphasis of building social networks of learners. In contrast to xMOOCs, where the lecturer provides detailed information about certain topics in a predefined schedule and on a single website, cMOOCs rely on the active contribution of the learners throughout multiple sources [37]. Lecturer provide basic topics and a few introductory videos or slides. In addition, they may define a rough time schedule for focusing on new topics. Sometimes, just single questions are asked or statements made without caring to elaborate them. Dealing with it, that means to discuss about and find answers to open questions, is the duty of the learner. They should interconnect with each other, write blog posts and tweets, make reports or slides and a few brave even provide self-made explanation videos. All of that learner-provided content is linked to the course and therefore accessible to all course participants. They then usually comment, patch, correct and discuss on created content which creates a kind of self-learning social network. Lecturers act as moderators and only intervene if something goes into wrong direction.

---

<sup>15</sup>One of the most popular xMOOC platform Coursera uses videos of 15 minutes length at maximum.

<sup>16</sup>Which is also the reason why this type of MOOC is called cMOOC.

### 3.2.3.1 Open Badges in MOOCs

MOOCs appertain perfectly for the integration and usage of Open Badges because they provide a wide set of possible badging applications and use cases respectively.

Within xMOOCs, micro-badges can be used as incentives to complete courses or to contribute to discussion activities in forums, chats or other social networking features. Additionally, micro- as well as meta-badges can be used for certain achievements like passing five courses within one year or getting more than 90% assessment score of a peer-review. They can also act as credentialing instrument for passed courses.

Cross, Whitelock, and Galley [16] for example, conducted experiments on such university level xMOOCs. Badges were awarded for active course participation ("*One-, Three- and Six-Week Badge*"), contributions ("*Resource Gatherer Badge*" or "*OER Developer Badge*") as well as the meta-badge "*Completed Course Badge*". Students' feedback from Twitter (special hashtag) as well as pre- and post-course surveys were analyzed which revealed that just one quarter remained skeptical or concerned about badging. The rest endorsed, although with great variety of reasons.

cMOOC platforms could integrate badges in a similar manner, but with a strengthened focus on social activities like writing 20 blog posts or creating an explanation-video for other learner.

Leaderboards, automatic E-Portfolio generation, badge sharing capabilities and, for example, badge importing from other MOOC platforms might be conceivable applications too, also for xMOOCs.

To name an example integration, Santos et al. [52] extended their social media platform based learning analytics tools with Open Badges. It was not really a cMOOC as defined in that thesis, but approximately relates to it. Student created content via Twitter and WordPress blogs were tracked, aggregated and monitored. Badges have been awarded depending on their activity, quality and produced results. But, they did not only award badges for positive activities, they also used negative ones for users who, for example, showed no contribution. However, they also concluded that using achievement badges has potential to motivate, if designed correctly.

### 3.3 Educational E-Assessment

Within E-Learning, especially in MOOCs, the probably most applied way a learner earns a badge is to achieve a certain goal, which is mostly to complete a course. If that requires to prove acquired knowledge, skills, attitudes or competency, the learner has to be assessed by some capable validator [58]. If the assessment is done by support of information- and communication technologies, it is called *E-Assessment*.

According to [10, 34, 58], assessments are distinguished by

- the *moment* a learner gets assessed. Thereby, five categories seem to be commonly established. These are *advisory*, *diagnostic*, *formative*, *summative* and *quality assuring*. The first two are performed before the learning process even started. The last one is applied after the course and serves as feedback to improve it next time. The two remaining types, namely *formative* and *summative*, directly correspond to the learning process and are therefore relevant for presented learning technologies as well as the application of badges. Formative assessments get applied *during* the learning process to reflect the current learn progress [58]. The aim is to steer and possibly adapt learning and to repeat content<sup>17</sup>. Such assessments should also motivate to deepen learned tenors and to self-study certain topics. Common examples for motivation enhancements are simulating tools where learner can build or program certain contents (e.g. circuits or algorithms) and instantly check if it works. Typically, there is some visual feedback with hints for potential errors. Experiencing success typically acts as an excellent motivator [34, 50]. Summative assessments get applied right after the learning process has been finished. It evaluates the final result of the learning process which is the most similar to traditional "end-of-term" exams at universities or schools.
- the used *infrastructure* to perform the assessment. Commonly used setups are *Stand Alone*, *Closed Network* and *Internet* assessments. *Stand Alone* means that assessments are performed on a network detached computer. Therefore, the examinee has to be present and results are stored locally. *Closed Network* assessments are done on computers that are connected to a local or restricted network. Therefore, examinees can perform the assessment in separate rooms. *Internet* assessments can be performed on any Internet connected device.
- the *assessing instance* itself, which are mostly *Expert Reviews*, *Peer-Reviews* and *Computer Automated Scoring Systems*. In *Expert-Reviews*, people with already ascertained competency within the subject assess the examinee. In *Peer-Reviews*, others within a corresponding community undertake that assessment. *Com-*

---

<sup>17</sup>As a side note, according to Bjork [10], elements of formative tests were usually more recognized than components which were not tested.



*puter Automated Scoring Systems* assess due to algorithmic procedures which implies that just tasks with a finite set of solutions can be applied. Common tasks are Multiple-Choice-Tests (MC-Tests), Drag and Drop Tasks (D&D) and Segmentation Tasks.

### 3.3.1 Assessable Knowledge

With respect to E-Assessments, there are primarily two types of assessable knowledge, namely *declarative* and *procedural* [61]. The first one represents factual knowledge that is building relations between certain objects. For example, declarative knowledge would be to know that a car has an engine. *Procedural* knowledge in contrast is to know how [53] the parts of an engine work together to move the car if the driver accelerates. It is declarative knowledge extended with a chain of logical rules, so-called procedures [61].

### 3.3.2 Assessment Tasks

The possibilities to estimate the knowledge, whether declarative or procedural, of a learner depends on the used learning technology as well as intended learning theory. It definitely makes a difference using a MOOC following the connectivism model (cMOOC) or a basic CrMS with the behaviorism model. The way to assess learners has to fit to the underlying technology and learning theory.

Nevertheless, an essential distinction between assessment tasks is to categorize them based on the presence of answers. *Closed* tasks are those where possible answers to statements are given and the learner has to select the correct ones. Therefore, especially recognition skills are tested. In contrast, in *open* tasks the learner has to answer the question completely on his own which mainly demands for reproduction skills [61].

#### 3.3.2.1 Closed Tasks

Because of the enormous potential to automatically evaluate such tasks, they find broad attention in LMSs and MOOCs. Which method should be used depends on the discussed course content, as it obviously makes a difference if a course taught basic mathematic concepts or a programming language. The following list represents the most used tasks within educational learning environments.

- **True/False:** The learner has to decide if a given statement is correct or not.

### 3 Learning Environments

- **Selective:** A question or a statement is given together with potential answers. Those answers do not have to be correct at all. The learner has to decide if none, one, multiple or all of the given answers are correct or not. Those tests are the most used ones within MOOCs and LMSs because of their simple design, re-usability and potential to form easy as well as complex questions.
- **Image Selection:** Typically, an image is given and the learner has to identify something within that image. Also, especially common in tasks for children, a set of images is given and the learner has to select a certain one.
- **Assigning:** Different objects are given and the learner has to relate them to each other. Another example would be a commonly known fill-in-the-blank text, but with a provided set of potential answers.
- **Sorting:** A set of objects is given and the learner has to sort them in a certain order.

#### 3.3.2.2 Open Tasks

These tasks try to test procedural knowledge, although that is usually not possible with written exams [61]. The learner has to *describe* the procedures which then results in a number of declarations. Nevertheless, it allows to test procedural knowledge in a sufficient manner for the purpose of online learning environments.

Problems arise by automatically evaluating these tests which is the reason for their scarce usage in MOOCs. As a typical LMS course does not handle that many learners, such open tasks are not the problem at all because a couple of tutors or peers are able to review them. Because of the, mostly, huge amount of participating students in MOOCs, that is not possible any more. Therefore, MOOC platforms try to use Peer-Reviews, which enables learner of a community to rate the submissions of each other.

However, typically [61] used open tasks are:

- **Short text:** Statements describe something that the user has to identify or name in a few words. The problem regarding automatic evaluation is that there could be more than just a few correct answers. The evaluation system should therefore be capable of synonyms and small deviations of the expected answers.
- **Long text:** These are the most commonly known tasks from the analogous world. The learner has to describe or define a relation or procedure in own words. There is some research about automatic essay grading ongoing [8], but those systems are currently not really supported by MOOCs or LMS and still contain errors that force someone to check results which hampers the advantage of automation. But surely, in some years that will probably be standard.

### 3 Learning Environments

- **Subset:** The learner has to provide a subset of correct answers to pass such a task. An example would be to let the learner name five advantages of a certain technology, although ten were discussed in the course. Such tasks are also common in LMS systems, but rare in MOOCs.
- **Cloze:** As in the case of the closed task's fill-in-the-blank text, but without providing the set of correct answers. The learner has thereby to understand the provided parts of a statement and to find appropriate words to complete it. Such tasks are common for language or mathematical courses where learners have to fill in correctly set quotation marks, colons, words, digits of certain numerical sequences or results of equations.

In addition to those written tasks, LMS systems can also provide the option to take oral exams like (live-) interviews via video chat, observations by completing certain tasks (e.g. through shared screens) or (group-) debates on certain topics. Standard MOOCs (x- and cMOOCs) mostly follow the concept of asynchronous communication. Therefore, such oral exams are neither used nor demanded. That may not be the case of specialized MOOCs like the *language MOOC* where personal contact is part of the concept.

## 4 Integrating Open Badges in Learning Environments

University internal CrMSs typically impute formal course achievements of authorized students to their learn process by awarding credits (like credit points or ECTS). That suffices to motivate students to complete courses by taking final exams. External institutions that provide non-formal or informal learning can currently not provide credits students can impute at their universities. So to motivate those as well as the general public to enroll to courses and furthermore to also complete it, other incentives have to be offered.

As Open Badges have the potential to do that, it may be a good idea to integrate them in already existing learning environments. That integration depends primarily on the used system and the effort one wants to put in. Motivating learners is not achieved by just implementing a backpack or appearing as displayer for already earned badges from other platforms. Therefore, a platform offering education should always at least act as issuer. Backpack and displayer capabilities are surely beneficial, but not as important as acting as issuer. Therefore, possible ways to implement issuer capabilities are described in the following sections. In addition, selected related work is mentioned if their contribution has been considered to be suitable.

### 4.1 Source Code Modification

If open source LMSs or institution internal developed systems are used, a versed programmer could directly modify the source code. Contingent additional components like a database or a private key may be added and referenced properly.

This approach has high potential on introducing errors in the base system and to make the code not maintainable any more. In addition, updates of the base system could harm the badging implementation if not coded properly. Furthermore, there is no community that maintains and improves the code or implements additional features for the badge support. Every time the Open Badges specification changes, if minor or major changes do not matter, the base system code has to be modified.

So in short, that approach is not recommended unless a clear concept and versed programmers do exist.

### 4.2 Plug-In

A *Plug-In* is a software module which is usually developed from third parties and *extends* the functionality of already existing software. Typically, it is dynamically loaded at application start and does not influence core features which also means, that Plug-Ins can be removed without disturbing the main program. Therefore, the application has to provide an interface which Plug-Ins can use. That also means, it is not possible to create Plug-Ins for any software product, especially if the base system is proprietary.

Plug-ins are probably the most used badging integration method out there at the moment. The reason is, that nearly every learning environment provides APIs that others can use. Additionally, most of the systems currently out there explicitly support the development of Plug-Ins, especially open-source systems that make use of the huge community that is behind such systems. Meanwhile, there exist at least one Plug-in for each popular open source system. An overview of promising open-source Plug-Ins is given in table 4.1.

As example, Albrecht [1] utilized the open-source Plug-In *WPBadger* for their WordPress based *Saxon Open Online course*<sup>1</sup> which was jointly held by the University of Technology of Chemnitz and Dresden as well as the University of Siegen. She concluded that the application of Open Badges with the mentioned Plug-In may be useful for simple concepts and a low number of students, but not for bigger courses and more complex badges as the implementation effort may become too large. Improved Plug-Ins with support of automatic awarding have to be developed to make badging more usable for large courses.

That conclusion states something important by considering plug-ins, if commercial or not does not matter. Externally developed plug-ins can hardly fulfill the needs of all LMS providers without code adaptation. If one wants to have a highly customizable solution or one that perfectly satisfies the requirements an institution has regarding their badge system, then an own solution has to be programmed or at least, an existing adapted. Just installing an existing one will most likely lead to suboptimal results. That does not mean, that third-party solutions are bad, but each issuer has different needs and demands on its system which other plug-in developers cannot know or fulfill in advance.

Therefore, among other reasons, Haaranen et al. [27] for example, developed their

---

<sup>1</sup><https://soopal.wordpress.com/>, August 8, 2015.

## 4 Integrating Open Badges in Learning Environments

own badging system as Plug-In for their university's A+ LMS. Also Domínguez et al. [20] examined the effect of gamification on E-Learning by coding an own Plug-In for their used university learning environment *Blackboard Learn*<sup>2</sup>.

Although both had still minor recommendations for improvements, there badging system worked well.

Base System	Plug-In	URL
WordPress	WPBadger BadgeOS	<a href="https://github.com/davelester/WPBadger">https://github.com/davelester/WPBadger</a> <a href="http://badgeos.org/">http://badgeos.org/</a>
Drupal	Open Badge-It	<a href="http://drupal.org/sandbox/kayelle/1788572">http://drupal.org/sandbox/kayelle/1788572</a>
Rails	Rails engine for badge issuing	<a href="https://github.com/PRX/badges_engine">https://github.com/PRX/badges_engine</a>
PHP	Badge-it-gadget-lite	<a href="https://github.com/Achievery/badge-it-gadget-lite">https://github.com/Achievery/badge-it-gadget-lite</a>
Java	Merit	<a href="https://github.com/wookoouk/merit">https://github.com/wookoouk/merit</a>
Moodle	Moodle Open Badger	<a href="https://github.com/totara/openbadges">https://github.com/totara/openbadges</a>
Joomla	JomBadger	<a href="http://www.bolli.fr/en/openbadges/jombadger">http://www.bolli.fr/en/openbadges/jombadger</a>
Canvas	Canvas Badges	<a href="https://github.com/whitmer/canvabadges">https://github.com/whitmer/canvabadges</a>

Table 4.1: Promising open-source Plug-Ins for popular base systems (URLs accessed on August 8, 2015).

### 4.3 Add-In/Add-On

In contrast to Plug-Ins, an *Add-On* is *installed* upon an existing application or Plug-In. It makes use of application libraries and files but can be uninstalled without harming the main application though. Examples would be a web browsers extensions or themes, which are technically simple Add-Ons.

As Add-Ons are similar to Plug-Ins, they are of course an option to integrate Open Badges in learning environments. The disadvantage they have is that updates of the base system (application or Plug-In) could make the Add-On unusable. Also if underlying Plug-Ins get removed, the corresponding Add-On is unavailable too. So Add-Ons *rely* on the base system they are built upon.

A meanwhile popular Add-On regarding Open Badges is the so-called "*Open Badges*

---

<sup>2</sup>Blackboard Learn (former Blackboard Learning Management System) is a commercial learn environment developed by Blackboard Inc. <http://www.blackboard.com/about-us/who-we-are.aspx>, August 8, 2015.

## 4 Integrating Open Badges in Learning Environments

*Issuer Add-on*<sup>3</sup> that is built upon the *BadgeOS* Plug-In for the base system WordPress.

*Add-Ins* are Add-Ons but their installation influences core libraries or files such that uninstalling would harm the main application. Therefore, to get rid of an Add-In, the main program would have to be completely reinstalled.

Using that for Open Badges is technically of course a legit option, but it has nearly the same disadvantages like modifying the source code of the base system would have. It is therefore clearly recommended not to implement Add-Ins to enable badging.

### 4.4 Web application

Another option is to implement a web application that somehow communicates with the base system. The developer thereby has full control of the features, data and amount of autonomous performance, just like in the case of Plug-Ins. But, the web application does not have to be deployed on the same web server as the LMS or MOOC and can be designed to be completely independent of course or user data. However, regarding award automation it is necessary to have some sort of communication protocol.

Web applications can be used just like Plug-Ins, except that usually, the base system has to be modified to serve and accept data to and from the web application if existing API functions do not fit. Therefore, if the base system can be modified and the badging system should be implemented as generic as possible, a web application might be the best choice.

Example related work might be Santos et al. [52] who implemented badging as a separated web application that interacts with existing components of their learning analytics tools.

### 4.5 Software as a Service

Another popular and the probably simplest way to issue Open Badges is to use a third party website that offers such services. Most of respective platforms out there offer free, pro and enterprise licenses which mainly differ in the amount of features one can use. The current leader regarding hosted solutions seemed to be *Credly*, followed by the *OpenBadgeFactory*. Depending on the payed license, one cannot just

---

<sup>3</sup><https://wordpress.org/Plug-Ins/badgeos-open-badges-issuer-add-on/>, August 8, 2015.

## 4 Integrating Open Badges in Learning Environments

issue badges but also track them and generate enhanced statistics to earners and their badges. At that moment, the PRO license of Credly costs \$495 per year, where tracking and statistics has not been part of the quotation.

However, such services are principally not free of charge and anyone who wants to issue Open Badges has to decide if it pays off to pay for that service. In many cases, it probably makes sense because the service one gets with such sites is great and to implement it on one's own would probably not be cheaper at all. In addition, one could start badging right away and if problems occur, a professional team fixes them relatively fast.

The disadvantage such services have is that one has to deliver user, teacher and course data. All of such platforms state that they treat data with highest caution and do not sell them or whatever, but because it is not one's own code, one can never be sure what really happens with those data. Furthermore, although platforms like Credly and OpenBadgeFactory offer a huge amount of features, some institutions actually have specific needs or beliefs regarding the issuing process. Some need a specific badge type, some want to add customized Extension objects to the badge. Such demands can probably not be satisfied by those third party websites.

Another aspect to consider is what happens if the use of the service gets canceled sometime due to increasing costs or other reasons. Then, one has to configure and import the data to another system, which could make troubles because of incompatible data formats between the old and the new system.



# 5 Implementation of a Badge Issuing System for iMooX

## 5.1 iMooX

In March 2014, Austria's first MOOC-Platform, called iMooX, went online<sup>1</sup>. It has been a joint project initiated and developed by the University of Graz (KFU) and Graz University of Technology (TUG) and got sponsored by the "Zukunftsfond Steiermark"<sup>2</sup> [24]. Since October 2014, iMooX has also been under the aegis of the UNESCO for being an important contribution to open access and the promotion of information- and communication technologies in education. The main idea was to offer open<sup>3</sup> courses not only for students of the mentioned universities, but for the general public. With iMooX, all ages have the complimentary possibility to learn which supports *lifelong-learning*. On this year's symposium about "How MOOCs change university doctrine" Ebner [21] (translated by the author) presented, that 54% of all course participants were older than 35 years. That trend was also observed within the evaluation of the implemented web application (see chapter 6 for details), where even 75% of those who completed the course under evaluation were older than 35 years<sup>4</sup>. This may indicate, that iMooX is on a good way to achieve intended goals.

Compared to other MOOC platforms, the extraordinary precept is that each course is offered as Open Educational Resource (OER). Such declared resources are, depending on the actually used creative common license, explicitly allowed to be distributed or even modified.

However, iMooX implements the xMOOC concept. As it is usual for that concept, users can check themselves if presented content has been understood. This is done by so-called *self-assessment quizzes*. These are basically multiple-choice tests with about five to ten questions where no, one or multiple correct answers exist per question (an example is given in figure 5.1a). For each quiz, the user has five

---

<sup>1</sup>[www.imoox.at](http://www.imoox.at), August 10, 2015.

<sup>2</sup><http://www.zukunftsfonds.steiermark.at/cms/ziel/79305483/DE/>, August 22, 2015.

<sup>3</sup>Open means that registration and course participation is for free.

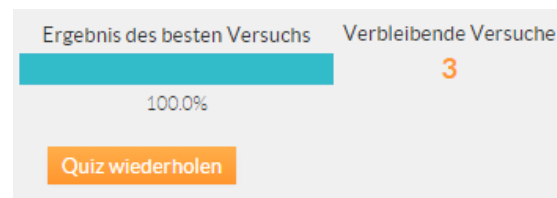
<sup>4</sup> 44 years on average.

## 5 Implementation of a Badge Issuing System for iMooX

trials. On each trial, the questions remain stable, although the ordering of their answers varies randomly. If the user commits his quiz answers, feedback is given immediately (an example is given in figure 5.1b). Such quizzes are not intended to be a university level exam but rather to help the user to check if topics have been understood. Therefore, the user can display a hint for each wrongly ticked off answer which should help the user to *understand* why it was wrong. Nevertheless, to introduce some seriousness, the user has to have at least 75% fully correct answers to "pass" a quiz. Only if all, usually one per course unit, self-assessment quizzes have been passed, the user can generate a personal certificate of participation for that course (see figure 5.2 as an example) which can then be downloaded as PDF file.



(a) Shows one of six questions from the COER15 course with all answers correct.



(b) Shows the feedback someone gets by committing quiz answers. The beam shows the percentage of correct answers and the number next to it shows the amount of remaining trials.

Figure 5.1: Self-assessment quiz sample.

To utilize the promising advantages of Open Badges as formative as well as summative feedback and recognition instrument, iMooX got equipped with a badge issuing system. On the one hand, this should enable iMooX to issue course-level badges, further called *Meta-Badges*, instead of the currently used PDF file. On the other hand, allowing iMooX to issue quiz-level badges, further called *Mirco-Badges* could help to strengthen the users volition to pass all quizzes of a course, even if that might not be the user's intention. If the user does not complete the course, then, at least, partial performances can be recognized.

The iMooX system is based on the *WBT-Master*<sup>5</sup> software, which is a learning infrastructure to support the CORONET<sup>6</sup>-Train methodology [3], and was developed at Graz University of Technology (TUG) [31].

<sup>5</sup><http://coronet.iicm.edu/wbtmaster/welcome.html>, August 10, 2015.

<sup>6</sup>CORONET: Corporate Software Engineering Knowledge Networks for Improved Training of the Workforce

## 5 Implementation of a Badge Issuing System for iMooX

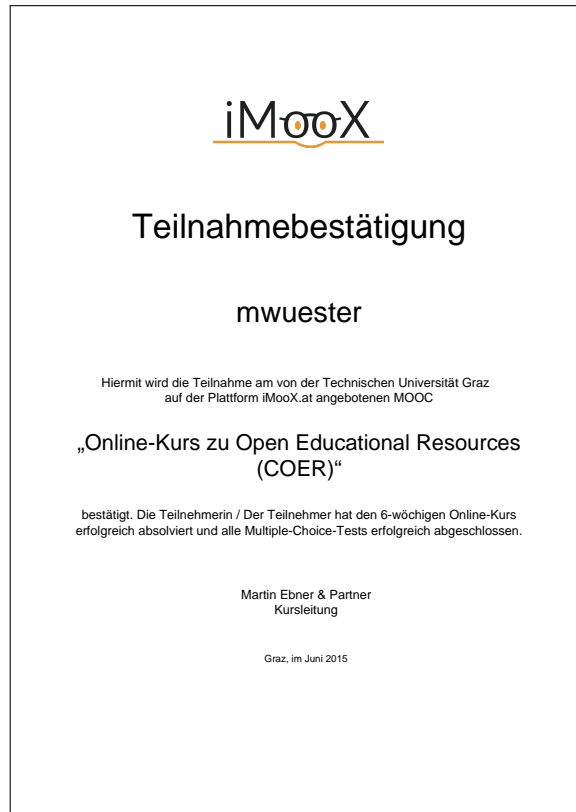


Figure 5.2: Example of a traditional certificate of participation.

### 5.2 Overview

To keep interacting components on TUG servers and to ease maintenance, the implemented web application was deployed on the same server as iMooX, which has been an Apache Tomcat<sup>7</sup> web server<sup>8</sup>. Additionally, generated URLs used in badges are consistent with those of the issuer without the need of redirection. However, it could be deployed on any other web server too.

To accomplish given requirements on the system, which are to create, maintain and automatically issue Micro- as well as Meta-Badges, two types of badges were defined:

1. *Quiz Mastery Badge* (QMB) which is awarded if the user passes a certain amount of self-assessment quizzes. That badge type represents a Mirco-Badge and should acknowledge partial performances as well as motivate the learner to also take the next quizzes, if it is not already the user's intention to do so.

<sup>7</sup>Project website <http://tomcat.apache.org/>, August 10, 2015.

<sup>8</sup>According to Netcraft [42] and their analyzed data from June 2015, the Apache Tomcat web server has still the lead with 49.53% market share over all active websites.

## 5 Implementation of a Badge Issuing System for iMooX

2. *Certificate of Participation Badge* (COPB) which is used to certify passed courses replacing the previously used PDF file corroborations. That badge type represents a Meta-Badge.

The implemented issuing system was named *badgeit*, composed out of the action *to badge* and *it* as the Open Badge to issue. It was realized as JAVA web application using Eclipse<sup>9</sup> Luna (version 4.4) as Software Development Environment (SDK) and the Java Development Kit (JDK) 1.6<sup>10</sup> as Java build and Runtime Environment (JRE). The resulting web application was deployed on the same web server as iMooX, a Tomcat 7.0 web server.

For storing badge and assertion data, a MySQL<sup>11</sup> 5.1 database was used. However, as the application was programmed database-agnostic utilizing Hibernate<sup>12</sup> also various other database systems could have been used.

### 5.2.1 iMooX Scripts

To be able to realize automatic awarding, the issuing system has to have access to certain user and course data. Because data might change quite frequently, they have to be provided dynamically. *badgeit* therefore calls two distinct iMooX-scripts that return data in a predefined and appropriate manner. One returns a list of courses, the other one user performances. The location of those scripts within the iMooX (wbtmaster) directory can be configured in the application's configuration file (see section 5.3.5).

#### 5.2.1.1 Course List

Calling this script returns a list of course objects formatted in JSON. Each object is thereby described by its system intern identification string (*courseID*), the corresponding full name (*courseName*) and the amount of self-assessment quizzes that are defined for that course (*numberOfQuizzes*). The first two attributes are used to store and display courses. The last attribute, the number of quizzes, is primarily used for generating the check-boxes as criteria for QMBs (see section 5.4.1.1 and figure 5.11a). In addition, if iMooX requests badges of a certain user (see Servlet *GetUserBadgesServlet* in section 5.3.6), the number of possible course quizzes is appended to the response and displayed to the user. Listing 5.1 shows an example response of the course list script call including three iMooX courses.

---

<sup>9</sup>Eclipse project website <https://eclipse.org/>, August 10, 2015.

<sup>10</sup>To be compatible with the used web server.

<sup>11</sup>Open source database, website <https://www.mysql.de/>, August 10, 2015.

<sup>12</sup>Hibernate: A Java persistence ORM framework <http://hibernate.org/>, August 22, 2015.

## 5 Implementation of a Badge Issuing System for iMooX

```
1 {
2   "courses" : [{
3     "courseID" : "socialmedia2015",
4     "courseName" : "Soziale Medien & Schule: fuer wen, wieso, wozu?",
5     "numberOfQuizzes" : "8"
6   },{
7     "courseID" : "phy2",
8     "courseName" : "Aha-Erlebnisse aus der Experimentalphysik Teil
9       2",
10    "numberOfQuizzes" : "7"
11  },{
12    "courseID" : "gadi",
13    "courseName" : "Gesellschaftliche Aspekte der
14      Informationstechnologie",
15    "numberOfQuizzes" : "10"
16  }]
17 }
```

Listing 5.1: Example response of the course list script.

### 5.2.1.2 User Data

That is a user specific script which replies a JSON object containing the email address of the user (*-email*) and all courses this user has ever attended (*courses*). Each of those courses contains a list of all passed self-assessment quizzes (*quizzes*) and a flag (*-cop*) indicating if the user has managed the whole course or not, independent of the amount of passed quizzes. Only if this flag is set to true, the user will get the corresponding COPB. That flag is needed because the requirements to completely pass a course may vary. For example, one course has a mandatory questionnaire attached, the other one has not. So to know if a user has fulfilled all requirements of a course, it does not last to just know the ratio of passed quizzes. The decision, when or if a course has been completed should always be in duty of iMooX, as that maintains high customization in course design. The badge issuing system just has to know when that is the case for a user and does not have to care about checking potential complex criteria. Listing 5.2 demonstrates an example for a fictive user, who attended one course and passed two quizzes. Future works might also add the time stamp when those quizzes had been passed to allow issuing further Micro-Badges like an *"Early Bird Badge"* for example that rewards quiz attempts right after the corresponding unit instead of making all unit quizzes at the end of the course.

## 5 Implementation of a Badge Issuing System for iMooX

```
1 {
2   "user_data": {
3     "-email": "john@doe.com",
4     "courses": [{
5       "-id": "someCourseId",
6       "-cop": "false",
7       "quizzes": {
8         "quiz": [
9           {"-id": "1"},
10          {"-id": "2"}
11        ]
12      }
13    }]
14  }
15 }
```

Listing 5.2: Sample iMooX-script response for a certain user.

### 5.3 Components

The web application primarily consists of following components:

- **Servlets** which receive HTTP requests and return either an image, a JSON string or an HTML file
- **Java Servlet Pages (JSP)**. Servlets typically collect or construct data and redirect them to corresponding JSPs which then generate the appropriate HTML content
- **Database** to store badges and their awards
- **Models** and their Data Access Objects (**DAO**). Models hold data in memory whereas their DAOs are used to access (load from and save to) corresponding tables of the database
- The **Hibernate configuration file** which controls the access to the used database instance.
- **Signature Key** to sign badges

Figure 5.3 illustrates the structure and adumbrates the interaction between those components.

The remaining section describes each of those components. Thereby, underlying technologies will *not* be described in detail as that would go beyond the scope of this thesis. Therefore, it is assumed that the reader has at least basic knowledge of JSPs, HTML, Servlets as well as SQL or, in general, relational database models (RDBMs).

## 5 Implementation of a Badge Issuing System for iMooX

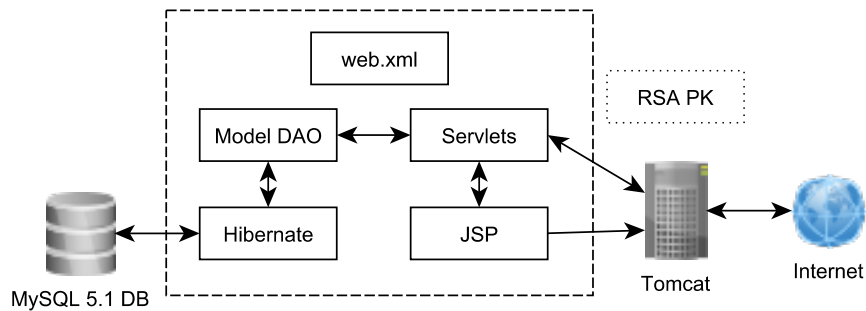


Figure 5.3: Basic components of badgeit. The dashed border outlines the application core.

### 5.3.1 Database

This section covers the main idea of the used database schema and describes its table's purpose. Figure 5.4 illustrates the twelve concerning tables of the normalized database schema.

#### 5.3.1.1 Tables *earners*, *awards* and *assertions*

These tables are only used in combination. If an iMooX user earns a badge the first time, its email address is stored in table *earners*. In addition, each generated badge assertion (see section 2.3.1.7) is stored in table *assertions*. Table *awards* combines both with the dedicated awarded badge class (see section 2.3.1.6) and the time stamp that happened.

That combination basically serves as badge award tracker. Entries reveal *who* earned *which* badge and *when*. That information is essential in three cases.

1. To check if a certain badge is permissible to be awarded or not. A badge owner can set a badge to be unique, which means that any user can earn that badge just once. So the application needs the information if a certain earner has already earned that badge or not
2. The amount of possible badge awards can be limited. For example, the earner limit of a certain badge can be set to three. Then, just three instances of that badge can be awarded. Therefore, the application needs to know how many instances of a certain badge have already been issued
3. For general statistics like the amount of different earners or the day most of the badges were issued

## 5 Implementation of a Badge Issuing System for iMooX

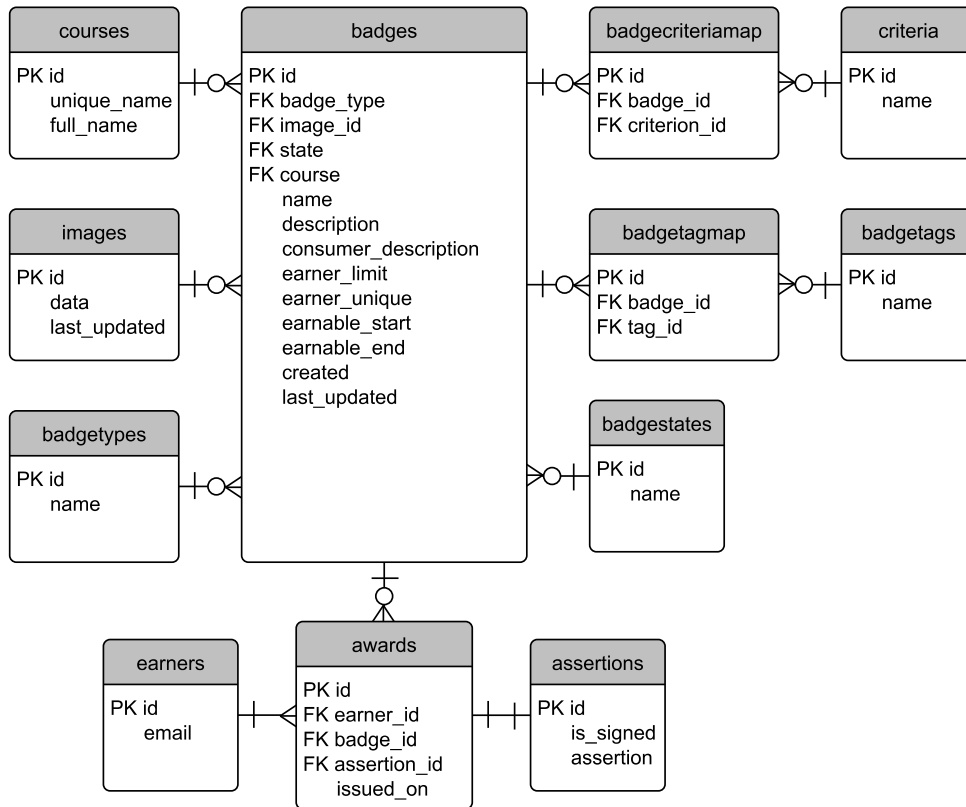


Figure 5.4: Used database layout, illustrated as Entity Relationship Model (ERM)

### 5.3.1.2 Table *badgestates*

Each badge can be of three states, namely *draft*, *published* or *archived*. Table *badgestates* encapsulates their names which makes it particularly easy to rename existing or add other states. *draft* is primarily used if a badge should be added to the system without making it public. *published* badges are those that can be earned. *archived* is typically used if a badge, that was issued at least once, should not be achievable anymore. Especially if a badge got issued once, it cannot be deleted, so to make it unachievable again, one has to archive it. Figure 5.5 illustrates possible state transitions. If a new badge should be added, the badge owner can decide whether to mark it as *draft* or to directly as *published*.

### 5.3.1.3 Table *badgetypes*

The web application currently implements two types of badges but can be extended in future very easily. One type represents a Micro-Badge and the other is used



## 5 Implementation of a Badge Issuing System for iMooX

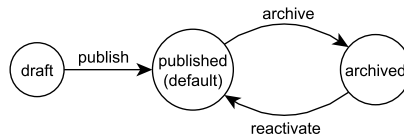


Figure 5.5: Possible badge state transitions.

as Meta-Badge. Corresponding names are *Quiz-Mastery-Badge* and *Certificate of Participation Badge*. Each badge has to be of one of those types.

### 5.3.1.4 Table courses

This table is needed because badges have to be awarded automatically. Therefore, each of them has to be assigned to a specific course. Course data is retrieved directly from iMooX, but to be able to handle the automation process, at least the unique course name has to be stored also within *badgeit*. The full-name is basically not needed, but it improves usability.

### 5.3.1.5 Table images

Each badge is linked to one image. That image can be updated or also used for other badges. Therefore, it has to be stored in a separate table. Although that feature is currently not implemented as each new badge has to upload an own image, but future works could change that relatively easy by providing a list of already existing images, if the user wants to use an existing one. That feature was not implemented within that work, because typically, at least on iMooX, each badge contains the course name as well as the year the course took place. Therefore, it hardly makes sense to reuse an image from an already existing badge. It definitely would make sense, if one decides to just use the course name within the image or to use completely individual badge images.

### 5.3.1.6 Table criteria, bidgetags, badgecriteriamap and bidgetagmap

The badge owner can add criteria to each badge. Depending on the badge type, those criteria are either quizzes (for QMB) or pure textual data (for COPB). To avoid redundant consideration of equal criteria, unique ones get stored within table *criteria* and mapped to a certain badge via the relation table *badgecriteriamap*. If a criterion has no reference within that relational table anymore, then it is removed

## 5 Implementation of a Badge Issuing System for iMooX

from the criteria table. If a criterion should be updated, it does not override the existing entry as that would effect also any other badge that links to that criterion via the relation table. Therefore, a new criterion gets inserted and linked to the badge which criterion was edited.

Tags of badges are handled similar to *criteria*. They are used to label a badge with certain keywords that describe its purpose or meaning. Depending on the implemented features of an issuer, backpack or displayer, badges can be searched by tags. Future works could let an iMooX user search for special badges, based on their tags.

### 5.3.1.7 Table badges

This table is the core of the database, linking to or linked from all other tables. In addition to the already described relations, *badges* contains optional parameters to customize the badge and it's availability.

The *description* is for internal use only and should inform the badge owner or his colleagues what this certain badge represents. It is not public and can be left empty as well. In contrast, the *consumer\_description* is mandatory, public and basically for the badge earners as well as other viewers to understand what this badge stands for.

The *earner\_limit* can be used to issue some kind of bonus badge. For example, someone could issue a badge to the student who is the fastest in a certain context like for example, in finishing all quizzes or passing a quiz with 100% correct answers. Therefore, it would suffice to simply set the earner limit to one and just the fastest will earn it. Currently, that is no option for iMooX, but it is basically possible to do something like that without introducing a completely new badge type.

There are also two dates present in that table, namely *earnable\_start* and *earnable\_end*. Those two dates can be used to set a timespan in which a certain badge is achievable. That information is just for the badge owner, but follow-up works could display them also to the users who would then know when a certain badge is achievable and when it is not. Both dates can be left empty which makes the *published* badge always achievable.

### 5.3.2 Hibernate

*badgeit* uses Hibernate to connect and access the used database. Hibernate enables Java applications to be programmed database agnostic, which basically means that the source code is independent of the actually used database technology. That is achieved by, roughly said, introducing an additional layer between the database and the application. To configure that additional layer, Hibernate uses a configuration

## 5 Implementation of a Badge Issuing System for iMooX

file named *hibernate.cfg.xml* which contains all needed information to connect to and access the data of the database. Needed properties to connect to a MySQL database are shown in listing 5.3.

```
1 <property name="hn.connection.driver_class">com.mysql.jdbc.Driver</property>
2 <property name="hn.dialect">org.hibernate.dialect.MySQLInnoDBDialect</property>
3 <property name="hn.connection.url">jdbc:mysql://domain:protocol/dbname</property>
4 <property name="hn.connection.username">username</property>
5 <property name="hn.connection.password">password</property>
```

Listing 5.3: Hibernate database connection properties. The string "hn" is the abbreviation for "hibernate" and was just used within that listing to avoid line breaks.

The mapping of Java classes to their corresponding database entity can be done in two ways. First, each concerning database entity gets an own configuration file, named *classname.hbm.xml*. Such files contain the location of the corresponding Java class as well as the database table attributes as listing 5.4 demonstrates for the entity "Assertion".

```
1 <hibernate-mapping>
2   <class name="at.tug.mwuester.badgeit.model.Assertion" table="assertions">
3     <id name="id" column="id" type="long"><generator class="native"/></id>
4     <property name="signed" column="is_signed" type="boolean"></property>
5     <property name="assertion" column="assertion" type="string"></property>
6   </class>
7 </hibernate-mapping>
```

Listing 5.4: Hibernate entity mapping with an own entity configuration file.

The other variant is to declare the database relation attributes within the Java class itself (see listing 5.5 for an example) and to just tell Hibernate where to find that class (listing 5.6). That declaration is done with appropriate annotations. *badgeit* uses this variant because it is more flexible as changes can be coded directly within the class definition and no additional files have to be maintained.

```
1 @Entity
2 @Table(name = "assertions")
3 public class Assertion implements Serializable {
4
5   @Id
6   @GeneratedValue(strategy = GenerationType.IDENTITY)
7   @Column(name = "id", nullable = false, unique = true)
8   private long id;
9
10  @Column(name = "assertion", nullable = false, columnDefinition = "text")
11  private String assertion;
12
13  @Column(name = "is_signed", nullable = false, columnDefinition = "boolean default true")
14  private boolean signed;
```

Listing 5.5: Hibernate attribute annotations for Assertion.java.

## 5 Implementation of a Badge Issuing System for iMooX

```
1 <mapping class="at.tug.mwuester.badgeit.db.model.Assertion"/>
2 // ... list here all other classes
3 <mapping class="at.tug.mwuester.badgeit.db.model.BadgeType"/>
```

Listing 5.6: Hibernate entity mapping with annotations in corresponding Java classes.

### 5.3.3 Model DAO

To create, read, update and delete (CRUD) tuples, so-called DAO classes were used. These encapsulate every data manipulation method to provide a single access point to corresponding data. Each database entity has exactly one such DAO class which provides mentioned functionality. DAO classes are static, instantiate Hibernate sessions and set up appropriate transactions. If errors occur, Hibernate exceptions get thrown which are also handled by these classes. Listing 5.7 demonstrates a typical method within such a DAO class. It takes an identifier and returns the corresponding instance from the database table.

```
1 public static Assertion getAssertionById(long pAssertionId) {
2     try {
3         Session session = HibernateUtil.getSessionFactory().getCurrentSession();
4         session.getTransaction().begin();
5         Assertion assertion = (Assertion) session.get(Assertion.class, pAssertionId);
6         session.getTransaction().commit();
7         return assertion;
8     } catch (HibernateException e) {
9         // some error handling
10    }
11 }
```

Listing 5.7: Typical DAO class method returning an instance of a certain table.

### 5.3.4 RSA PK

To sign assertions (refer to section 2.1.5 for details), a 2048-bit RSA (private) key is used. The location of that key can be configured within the applications *web.xml* file. To generate that key *openssl*<sup>13</sup> was used. An example generation call is demonstrated in listing 5.8.

```
1 openssl genrsa -out private-key.pem 2048
```

Listing 5.8: Generate RSA private key with openssl

<sup>13</sup><https://www.openssl.org/>, August 10, 2015.

For verification requests (see `GetPublicKeyServlet` in section 5.3.6), the corresponding public key has to be provided. Therefore, the application loads the public key bytes directly from the private key, which has the advantage that the public key does not have to be stored in an extra file which reduces the maintenance effort.

### 5.3.5 Web.xml

The application is configured by the *web.xml* file. It was used to declare the path to the private key as well as to the iMooX scripts. This has the advantage, that if paths change, the web application code does not have to be modified, which would be necessary if paths were hard-coded. In addition, path declarations are central and can be easily looked-up if needed.

### 5.3.6 Servlets

Java Servlets are used within Java web applications to service HTTP requests [14]. *badgeit* implements several of such Servlets to process administrator as well as iMooX requests and to generate appropriate dynamic web content. The following list briefly describes each relevant Servlet and its purpose.

- **LandingServlet:** Serves as starting point within *badgeit*, reacts on */start* pattern and does not take any parameters. Its purpose is to collect all existing badges and to delegate them to the corresponding starting JSP (*default.jsp*).
- **CrudServlet:** The most important Servlet regarding badge management. It is responsible for adding, editing and deleting badges. Several actions get caught and handled appropriately. It reacts on the pattern */crud* followed by the parameter *action*, which can be either *add*, *archive*, *reactivate*, *edit* or *delete*.
- **GetBadgeImageServlet:** It takes an image id as parameter (*id*) and loads the corresponding image file from the web server. On success, the image data is returned as Base64 (refer to section 2.1.2) encoded byte array with set response header to *"Content-Disposition:inline"*.
- **GetPublicKeyServlet:** Loads and appends the public key bytes as character text to the HTTP response in PEM<sup>14</sup> format (see listing 5.9 for an example response). It is usually called from websites that want to verify a signed badge as these get this Servlet's URL written to its assertion.

---

<sup>14</sup>DER Base64 encoded with an additional header and footer line.

## 5 Implementation of a Badge Issuing System for iMooX

```
1 -----BEGIN PUBLIC KEY-----
2 MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAs8cZv/6NiMuCT2HYwu0vB
3 S03g3/u23oPMK/ahx9cc3vAjB7jw8NpzFu03eD5N2M+dL4PqS4rki1yIX423iYEN
4 16PZklmp3Sm68u3MjcGMjLCqHUFdo1HjkkTJnSJPduXkvYuze+IKwaT+j6GjmIE6
5 8PhtSmkt3ZGYpWfnYBhkZ0W9r0Sd2FszCWM51IhIzJgct0rTU7q0rQ/Qlpi+Zg1z
6 dxhheb+cE7qgqVLG4pkTXPTtMZ2HdpGEwj5MfKNHVgYRjEMXNMVHmzDY7hF6oe54
7 XzSPrttqz/SUODHjiuknJUR0x+9ktnalyqY0pzENe+xm0NNjdT29xR6+MOE+STUp
8 twIDAQAB
9 -----END PUBLIC KEY-----
```

Listing 5.9: Example response to an `/public-key.pem` servlet call

- **GetBakedBadgeServlet:** Gets called if an existing assertion should be embedded (baked) into the corresponding badge image. Therefore, it takes an assertion identification number as parameter (*id*) and returns the processed image file as attachment to the HTTP-response. iMooX triggers that Servlet call if a user clicks on the badge download button.
- **GetUserBadgesServlet:** Takes the parameter *user*, which is the unique iMooX account name of the user for whom badges should be generated. If the parameter is provided, the iMooX script to gather user performance (see 5.2.1) is called. The script's response is a JSON object with a specific structure (see listing 5.2). After parsing that, deserved but not already awarded badges are generated. The response of that Servlet is then, as always, a collection<sup>15</sup> of badges dedicated to the given user.

### 5.3.7 JSP

Java Servlet Pages (JSP) combine standard HTML and Java and is considered as own (web) programming language [44]. Therefore, JSPs enable the generation of highly dynamic web content.

*badgeit* uses four JSP files, each representing another web page. Each JSP is called by dedicated Servlets. The reason is, that, although JSPs could technically fetch data on their own, within *badgeit*, Servlets gather and process data before delegating them to appropriate JSPs. That approach was clearer and easier to implement because mixing data fetching in both would have made it harder to maintain or to find bugs within the source code. The relation<sup>16</sup> between JSPs and their Servlets is illustrated in figure 5.6.

However, the purpose of each of those four JSP files is briefly described in the following list.

<sup>15</sup>If the user has never earned any badge, this collection is an empty JSON array.

<sup>16</sup>Servlet calls from JSPs are mostly done by corresponding JavaScript functions as most actions get triggered dynamically by the user. However, those JavaScript files were omitted in figure 5.6 out of lucidity reasons.

## 5 Implementation of a Badge Issuing System for iMooX

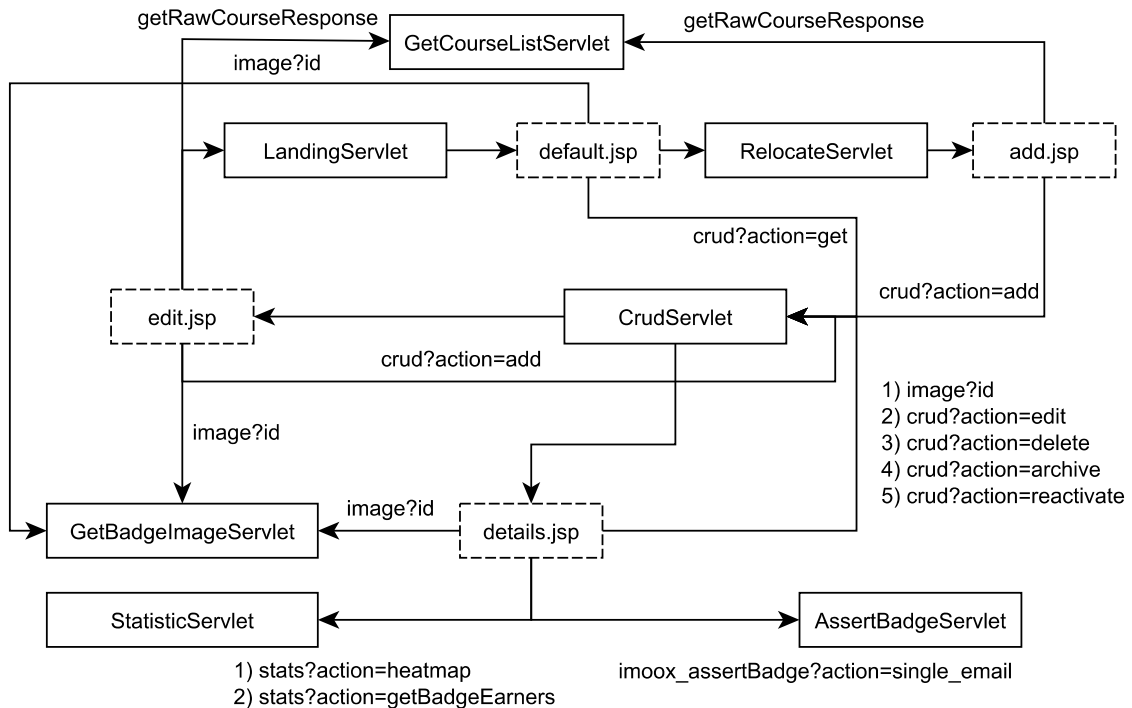


Figure 5.6: Relation between JSPs and Servlets. Parameters were omitted due to lucidity reasons. Arrows just represent calls, not communication in general.

- **default.jsp:** Represents the "homepage" of the application and gets its data from the *LandingServlet*. It presents the viewer (the administrator or badge owner) a collection of all existing badges categorized by their assigned course (see figure 5.7). That layout should make it easy to find certain badges even if there exist a lot<sup>17</sup>. Not achievable badges, whether due to their state (draft or archived) or to their defined time span (start and end date), get shown with reduced opacity to indicate their elusiveness instantly. The following three files are visually pretty similar. The reason was to provide some consistency across the pages making it easy to find ones' bearings.
- **add.jsp:** Is used to define a new badge and gets called from *default.jsp* via the *RelocationServlet* which fetches all available courses from iMooX. Figure 5.8 shows the HTML form and all customization options.
- **edit.jsp:** Is visually the same as the *add.jsp*, but with visible input form elements. Depending on the badge data and its state, form elements are en- or disabled. Figure 5.10 shows the HTML form and all customization options.
- **details.jsp:** Shows badge details and provides the user a couple of actions to perform, which depend on the current *badge state*. For example, if a badge

<sup>17</sup>Future works could additionally implement a filter to search for certain badges.

## 5 Implementation of a Badge Issuing System for iMooX

is published and has never been awarded, the user can still edit everything. In contrast, if the badge has been awarded at least once, the badge criteria, image, name, course and type can not be edited any more. In addition, the badge can only be archived or asserted, but not deleted or reactivated. Figure 5.9 shows an example of the web page representing badge details.



## 5 Implementation of a Badge Issuing System for iMooX

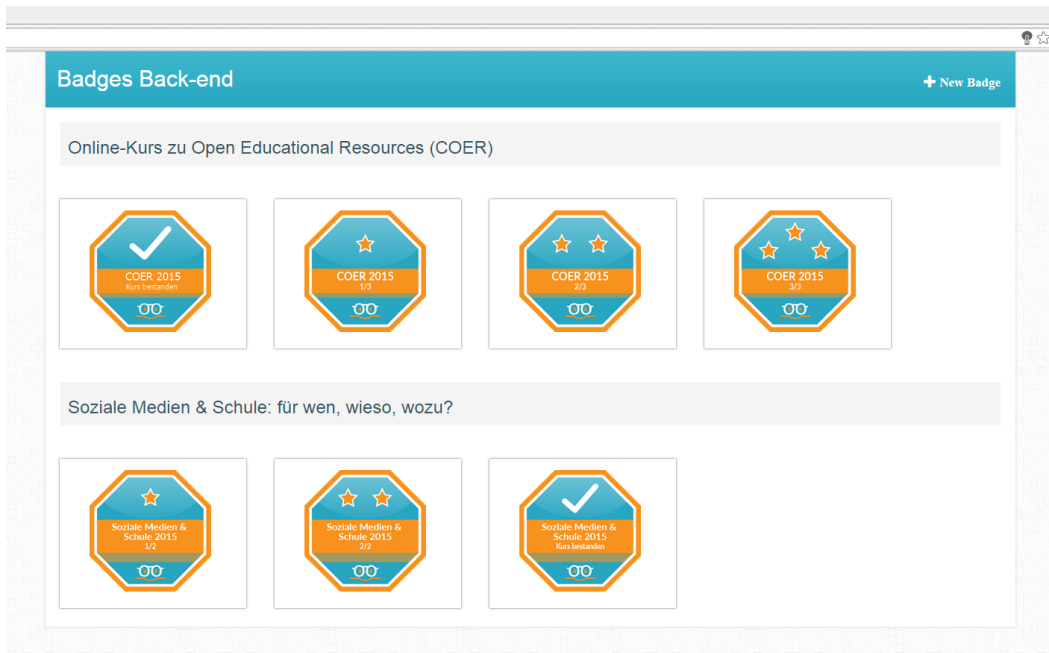


Figure 5.7: The "homepage" of *badgait* (*default.jsp*). Existing badges are presented according to their assigned course.

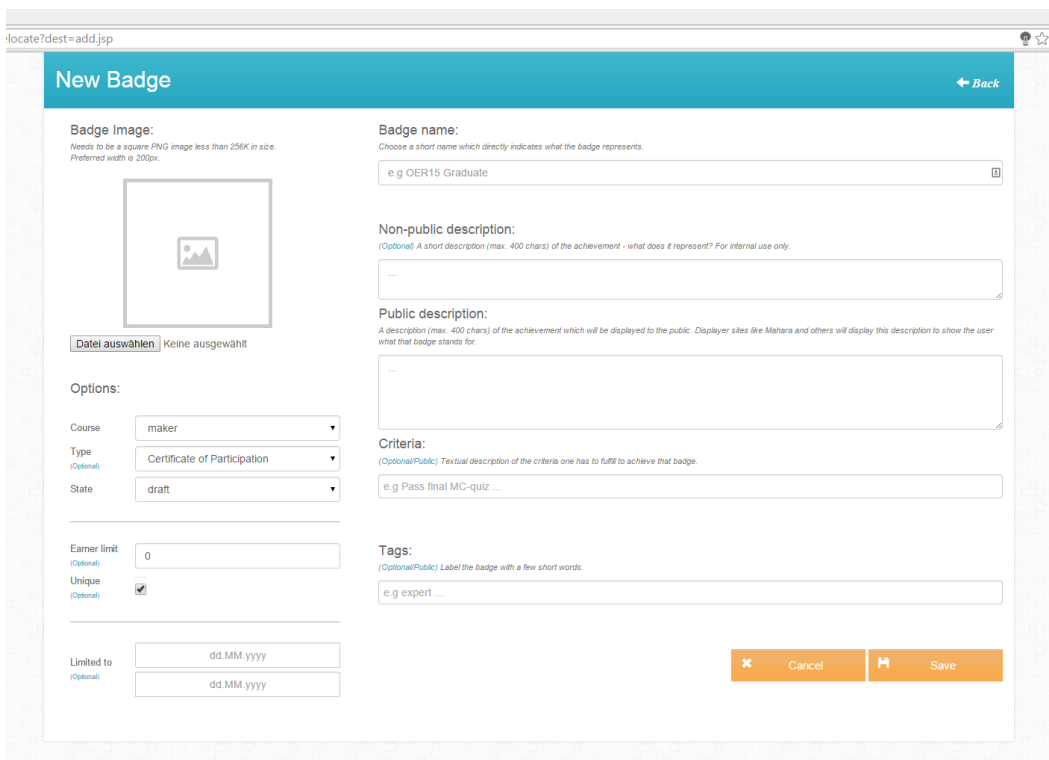


Figure 5.8: Web page to define a new badge (*add.jsp*).

## 5 Implementation of a Badge Issuing System for iMooX

The screenshot shows the 'Soziale Medien AbsolventIn' badge details page. The badge image is a blue octagon with a white checkmark and the text 'Soziale Medien & Schule 2015 Kurs bestanden'. The page is divided into several sections:

- Badge Image:** Shows the badge image and a 'Back' button.
- Options:** Course: socialmedia2015; Type: Certificate of Participation; State: published; Earner limit: Unique (checked); Currently achievable: yes; Achievable from: -; Achievable till: -; Created: 09.06.2015 22:49; Last Updated: 09.06.2015 22:58.
- Non-public Description:** Teilnahmebestätigung für den MOOC "Soziale Medien & Schule: für wen, wieso, wozu?", durchgeführt im Frühjahr 2015 (Start am 20.4.2015 mit einer Dauer von 8 Wochen). Dieser Badge bestätigt, dass eine BenutzerIn alle Quizzes des Kurses erfolgreich absolviert und auch die abschließende Kursevaluation durchgeführt hat.
- Public Description:** Dieser Badge bestätigt, dass der/die EigentümerIn einen umfassenden Überblick über Theorie und Praxis zum Thema "Soziale Medien & Schule" erhalten und mit Hilfe von Self-Assessments gezeigt hat, dass er/sie die Zusammenhänge des Themas auch verstanden hat. Dieser Kurs umfasst neben grundlegenden theoretischen Informationen zum Thema "Soziale Medien & Schule" praxisrelevante Tipps zum Einsatz sozialer Medien im Lehr- und Lernprozess.
- Criteria:** No criteria defined.
- Tags:** expert, iMooX, Soziale Medien.
- Statistics:** This badge has been awarded 5 times. A calendar shows the award dates from June 2015 to May 2016.

Figure 5.9: Representation of badge details for an already awarded badge (*details.jsp*).

The screenshot shows the 'Soziale Medien AbsolventIn' badge edit page. The badge image is the same as in Figure 5.9. The page is divided into several sections:

- Badge Image:** Shows the badge image and a note: 'Needs to be a square PNG image less than 256K in size. Preferred width is 200px.'
- Options:** Course: socialmedia2015; Type: Certificate of Participation; State: published; Earner limit: 0; Unique: checked; Limited to: dd.MM.yyyy.
- Intern description:** A short description (max. 400 chars) of the achievement - what does it represent? Teilnahmebestätigung für den MOOC "Soziale Medien & Schule: für wen, wieso, wozu?", durchgeführt im Frühjahr 2015 (Start am 20.4.2015 mit einer Dauer von 8 Wochen). Dieser Badge bestätigt, dass eine BenutzerIn alle Quizzes des Kurses erfolgreich absolviert und auch die abschließende Kursevaluation durchgeführt hat.
- Consumer description:** A description (max. 4000 chars) of the achievement which will be displayed to the public. Dieser Badge bestätigt, dass der/die EigentümerIn einen umfassenden Überblick über Theorie und Praxis zum Thema "Soziale Medien & Schule" erhalten und mit Hilfe von Self-Assessments gezeigt hat, dass er/sie die Zusammenhänge des Themas auch verstanden hat. Dieser Kurs umfasst neben grundlegenden theoretischen Informationen zum Thema "Soziale Medien & Schule" praxisrelevante Tipps zum Einsatz sozialer Medien im Lehr- und Lernprozess.
- Criteria:** No criteria defined.
- Tags:** expert, iMooX, Soziale Medien.

Figure 5.10: Web page to edit an existing badge which has already been awarded (*edit.jsp*).

## 5.4 Use-cases

*badgeit* provides primarily five meaningful use cases for badge owners (administrators) and one for the iMooX system, which is triggered by the user (badge earner).

### 5.4.1 Back-End

In this context, the *Back-End* describes the use cases and the user interface for the administrator. These are:

1. Define a new badge and publish it
2. Edit an existing badge because of orthographic mistakes or to control the badges availability
3. Delete or archive a badge because it is not needed any more
4. Assert a badge manually
5. Examine awarding statistics or search for certain earners

Each of those is shortly described in the following sections, whereby list items 1-3 were logically pooled as "manage badges".

#### 5.4.1.1 Manage Badges

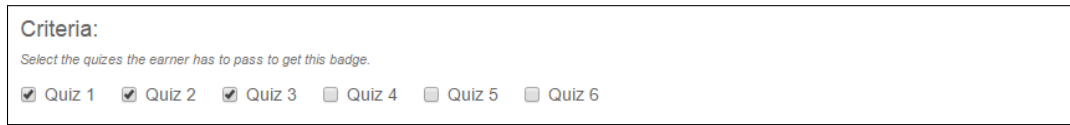
Defining a new badge requires the badge owner to provide some mandatory pieces of information. Most of them were already presented and discussed. Nevertheless, there are some additional specific remarks to mention.

*badgeit* issues two types of badges, which, according to their definition, only differ in the way their criteria are defined. If a course was selected (one is always selected per default) and the badge type was set to QMB, the amount of course specific quizzes is dynamically<sup>18</sup> fetched and the corresponding amount of check-boxes inserted into the web page (an example is shown in figure 5.11a). The badge owner than has to tick those quizzes which a user has to pass to earn that badge. This is needed for automatically checking if a certain user earns a QMB by the amount of already passed quizzes. Otherwise, the badge owner would have to enter all quizzes in textual form, like "Quiz1, Quiz2, ..." which is bad usability. In the case of COPB, criteria is actually a text. Therefore, the input is a text field where each criterion is visualized as tag (see figure 5.11b for an example).

---

<sup>18</sup>Using AJAX calls

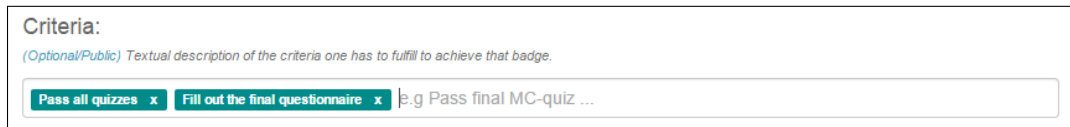
## 5 Implementation of a Badge Issuing System for iMooX



Criteria:  
Select the quizzes the earner has to pass to get this badge.

Quiz 1  Quiz 2  Quiz 3  Quiz 4  Quiz 5  Quiz 6

- (a) The generated criteria check boxes which are shown if a certain course and the badge type *QMB* have been selected.



Criteria:  
(Optional/Public) Textual description of the criteria one has to fulfill to achieve that badge.

Pass all quizzes x Fill out the final questionnaire x e.g Pass final MC-quiz ...

- (b) The criteria text input if the badge type *COPB* is selected.

Figure 5.11: Example showing the two possible criteria defining input forms.

However, after adding a badge, one can display the badge details on clicking on the badge image, which is presented on the "homepage". As long as no one else has earned this badge, one can edit all of its data, including the image, the course, the state and so on. Such badges can also be deleted, as long as no one else has already earned them. If at least one user has earned the badge, just optional attributes and the consumer description can be edited anymore. Everything else is read-only, as modifying those information may change what the badge represents. Those badges cannot be deleted, but archived in cases where the badge is not needed anymore. Deleting and archiving is triggered by pressing corresponding buttons on the *details* page.

### 5.4.1.2 Assert Manually

Beside the automatic awarding process, the badge owner can also trigger the process manually. To do so, the badge owner has to navigate to the details view of the badge that should get awarded. If the badge is not declared as draft and not archived, the button "Assert" is presented. Clicking on this button will trigger a pop-up dialog to appear (figure 5.12 demonstrates that) which asks the caller to enter an email address. This should obviously be the email address of the earner. If the earner has not already earned that badge (assuming that the badge is declared to be unique), the badge will be instantly issued and the badge owner gets a confirmation message if it succeeded or not. If the earner then navigates to his personal badges collection, the newly asserted badge will be displayed.

One problem currently persists. If the user has registered to iMooX with a different email address (named A for the moment), then the new badge, even if it was asserted to the correct email address (email B), will not be visible to the user. The reason is that the personal badge collection of iMooX always shows badges assigned

## 5 Implementation of a Badge Issuing System for iMooX

to the email address used to register the iMooX account (A). Although the manually issued badge considered the correct one (B) the badge is not accessible to the user. The current workaround is to email the earner a link where the new badge can be downloaded but, to do that, the administrator has to look up the assertion identification number in the database, which is pretty unfavorable. Therefore, if that manual issuing should be further considered, then that issue should be addressed first in future works.

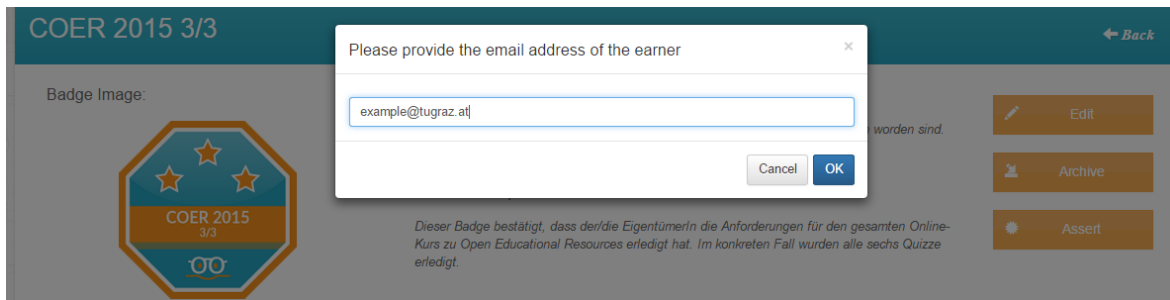


Figure 5.12: Popping up dialog inquiring the earners email address.

### 5.4.1.3 Examining Statistics

Badge owners can examine some basic statistics on each badge details page. This includes looking up who earned a particular badge and when. The moment a badge was issued is presented in two ways. First, as so-called *heatmap* (figure 5.13a) where the amount of awards *per day* is color encoded and displayed on a calendar. That view should give an instant overview on which days a certain badge was awarded. If badges become more popular on the iMooX platform, then the color encoding will indicate peaks.

Second, the exact time stamp in addition to the corresponding earner is displayed in an own dialog (see figure 5.13b) which the administrator has to open by clicking on the "eye" icon next to the amount of already asserted instances of that particular badge.



### 5.4.2 Front-End

In contrast to the *Back-End*, the *Front-End* describes the user interface for, and processes triggered by, standard iMooX users (also called earners, learners or students). The most probable and therefore considered to be the "standard" use case is that an iMooX user logs in, does something on the platform and somehow, if accidentally or on purpose, navigates to the personal badge collection web page (see figure 5.15). Each time that happens, the user triggers its own badge awarding process. That is done by setting up a request to the *GetUserBadgesServlet* (see section 5.3.6). The system automatically adds the current and iMooX unique user name to the request which is needed within the Servlet to retrieve further user data. That is achieved by calling a special iMooX script (see section 5.2.1.2).

**Remark:** This data could also be gathered within iMooX and afterwards passed to the *GetuserBadgesServlet* request to protect the unique user name from being transferred. However, that approach could result in forgery. As JavaScript code is visible to the user, one who wants to deceive, assume user *Eve* for example, could therefore see that the data is gathered on iMooX and only then transferred to *badgeit*. Thus, *Eve* could request user data on her own (because *Eve* knows her email and user name and in addition sees the link to the script), manipulate it, for example to add quizzes to courses which *Eve* actually never tried, and pass that manipulated data as HTTP request parameter to the badge issuing system. *badgeit* trusts iMooX data completely and would therefore issue badges also to such modified data. So in other words, it would be relatively easy to betray and to get more badges than deserved.

To prevent that, iMooX just passes the user name and *badgeit* calls the script. So the link to the script is not visible to the user and *badgeit* calls the correct data itself. That minimizes the probability that someone can modify the data in between.

However, after parsing and processing the obtained data, the Servlet determines which badges the given earner deserves. If there are any deserved, that have not already been issued, the Servlet triggers the assertion process for each of them separately. Figure 5.14 illustrates that process.

The Servlet finally returns a JSON formatted list of objects. Each of those objects contain the following information:

1. The course identifier (unique within iMooX)
2. The course name
3. The amount of badges assigned to that course
4. A list of badge preview objects. Each of those contains
  - The assertion number which is used for possible later baking requests

## 5 Implementation of a Badge Issuing System for iMooX

- The name of the badge
- The consumer description of the badge
- The badge image as Base64 encoded byte stream

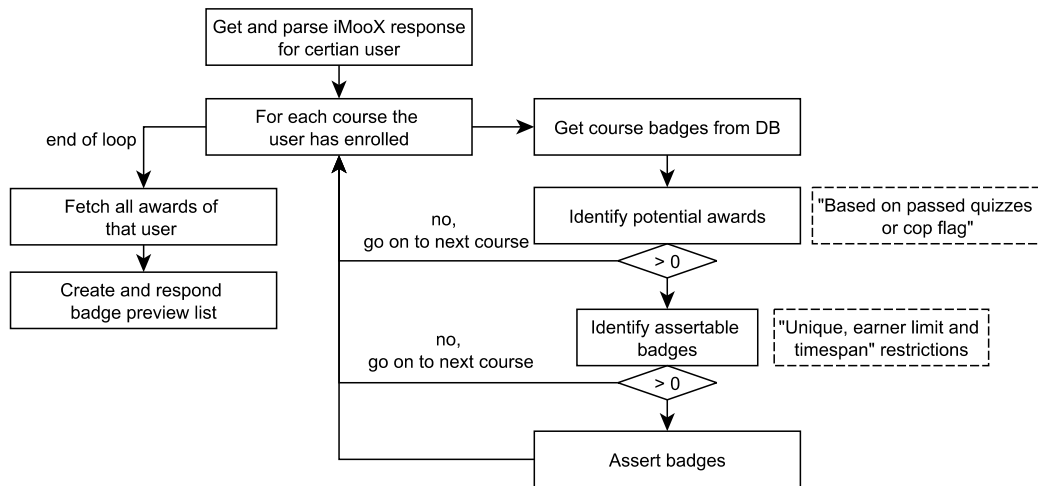


Figure 5.14: Overview of the badge awarding process triggered by the user.

```

1  [{
2  "courseId" : "someCourseId1",
3  "courseName" : "Some Course One",
4  "cntCourseBadges" : 4,
5  "badges" : [{
6    "id" : 37,
7    "name" : "SCO 1/3",
8    "description" : "Some description for badge 37",
9    "base64Image" : "data:image/png;base64,iVBOR..."
10   }, {
11    "id" : 41,
12    "name" : "SCO 2/3",
13    "description" : "Some description for badge 41",
14    "base64Image" : "data:image/png;base64,iVBOR..."
15   }
16 ]
17 }]
  
```

Listing 5.10: Sample response of the GetUserBadgesServlet containing one custom object including two badges.

Items 1-3 are used for usability reasons (see figure 5.15a). Item 4, the “badge preview item” can be seen as subset of a real badge which is used to present the user a kind of preview of the earned badge. It was introduced to reduce the amount of data to transfer as the badge image can also be scaled to arbitrary, but square, dimensions. However, it contains all relevant information at that time. The user is



able to identify the badge name, the description and image. If the user hovers over one of the presented badge previews, a tool-tip is displayed which contains the badge description (see figure 5.15b). A fictive example of such a Servlet response is given in listing 5.10. There, the requesting user earned two badges of one course. Overall, there are obviously four badges assigned to that course, whereby the user just earned two of them. The given "id" is the assertion identifier which is used to download the *real* badge if the user clicks on the download button right beneath the badge image file.

Since badges are mostly unknown to most of the users, a short introduction video<sup>19</sup> has been prepared to tell users what a badge is, which types iMooX implemented and what earners can do with it. That video has also been placed on the personal badge collection side, at the right top corner (see figure 5.15).

### 5.4.3 The Awarding Process

Awarding a badge means to generate a badge assertion, which basically contains the information *who* got *which badge* and *when*. This section describes the way the web application processes the information from triggering the assertion process to saving the final assertion string.

To recap, there are two possible ways an earner can get a badge awarded.

1. An iMooX administrator navigates to a certain badge and triggers the awarding process manually
2. The user navigates to the personal badge collection web page which automatically triggers the awarding process

Anyway, the email address of the earner and the badge to award have to be given<sup>20</sup>.

First, a unique identifier (UID) is generated by utilizing the `java.rmi.server.UID` class which "generates an identifier that is unique over time with respect to the host it is generated on" [45]. In the case of hosted badges, these UIDs are used as file names for the badge assertion file so the potential special character ":" (colon) has to be removed from that UID to surely avoid not allowed file names. As this modification (slightly) increases the probability that two identical UIDs get generated, the current time stamp in millisecond resolution gets concatenated to that UID, separated by an underscore.

---

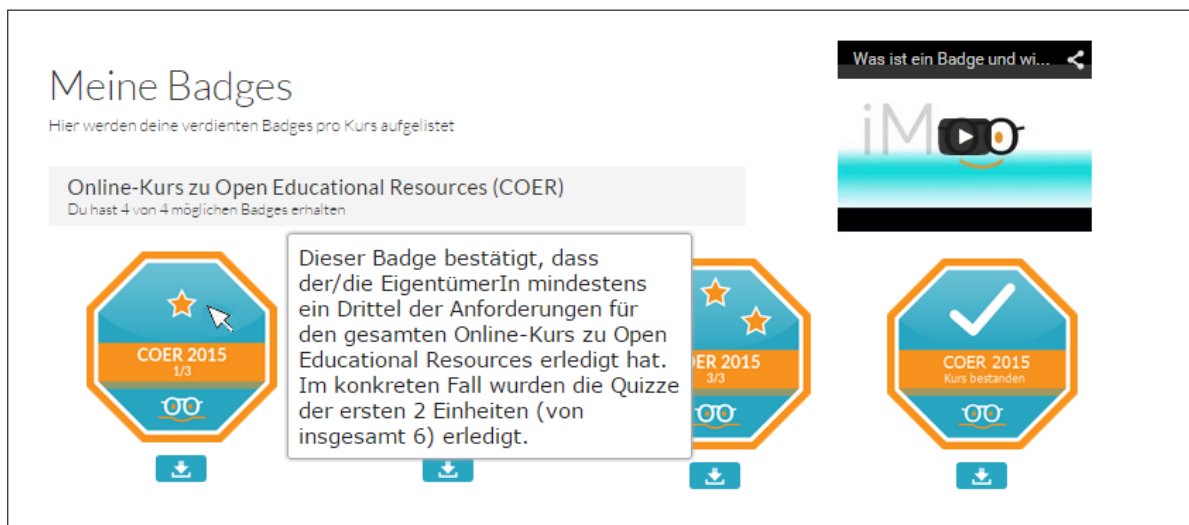
<sup>19</sup>Via Youtube <https://www.youtube.com/watch?v=v1jNSjgBXCg>, August 10, 2015.

<sup>20</sup>In the case of automatic awarding, there will be list of deserved, but still not issued badges, after some data processing. In that case, each of those get considered separately regarding this awarding process. So the starting basis of an awarding process is always having an email and a certain badge.

## 5 Implementation of a Badge Issuing System for iMooX



- (a) Collection of badges pooled by their assigned course. In addition, the amount of earned as well as the amount of possible achievable badges is displayed under the full course name. Beneath each badge image, there is the download button.



- (b) Presented tooltip if the user hovers the badge image. The text in that tooltip is the corresponding badge description.

Figure 5.15: The personal badge collection page of a user (front-end).

## 5 Implementation of a Badge Issuing System for iMooX

Second, the *Identity Object* (refer to 2.3.1.4) gets created, filled and added to the newly instantiated *Badge Assertion* (refer to 2.3.1.7) instance. Per default, the email address of the earner gets salted and hashed before it is appended to the badge assertion to increase data privacy. The 20-byte salt gets generated randomly for each earner individually and added to the users email address as input for the *SHA-256*<sup>21</sup> hashing algorithm. The resulting (hashed-)character sequence is concatenated to the algorithm identifier "sha256\$" as claimed by the Open Badges specification (refer to 2.1.1 and 2.3.1).

Third, the URL of the badge description (the *Badge Class* object - refer to 2.3.1.6) and the current time stamp is added to the assertion instance.

Then, if the assertion should be signed, the *Verification Object* (refer 2.3.1.5) gets set up with the URL of the *GetPublicKeyServlet* (see section 5.3.6) instead of the URL of the badge assertion file as it is the case of a hosted assertion. Additionally, if the badge should be hosted, then the JSON encoded assertion has to be written to an appropriate file on the web server.

The last part of the awarding process is to add the created badge assertion JSON string to the database. If the assertion is written to the database as plain text or as its JWS (refer to 2.1.5) representation purely depends on the badge type.

---

<sup>21</sup>RFC 6234 <http://tools.ietf.org/html/rfc6234>, August 10, 2015.

## 6 Evaluation

This chapter presents a short evaluation of the first iMooX course that actively used the developed badging system. It was used as test run to

- make first experiences with a completely self-implemented badge issuing system
- identify usability issues
- reveal possible specification violations

before making a big deal out of it. Therefore, there was no advertising campaign on websites or social networks beforehand. Only within the introductory video<sup>1</sup> of the course, the lecturer mentioned that in addition to the traditional certificate of participation a badge will be awarded for completing the course. But, nothing was said about micro-badges which are awarded for mastering quizzes. Because the introductory video was only visible to users who had already enrolled to the course, it is legit to assume, that concerning users were primarily interested in the topic and not in earning badges.

Due to missing stipulated earner feedback and the fact that just this course used badges, the resulting metrics may not reflect the real impact badging had on iMooX. However, its main purpose was to test the developed web application and not to study the effect Open Badges have.

### 6.1 COER15

The name of the course was "*Course for Open Educational Resources 2015*" (COER15) and was held by Assoc. Prof. Martin Ebner<sup>2</sup>. The content was presented in German and based on the course COER13<sup>3</sup>. To enroll, one had to have an account on iMooX. For those who did not, registration was for free of course. After the user had logged in, the course had to be selected followed by pressing the button "Zum Kurs anmelden" (en: "Join course", translated by the author) .

COER15 was mainly designed for people who were generally interested in *Open*

---

<sup>1</sup>At 4:13min [https://youtu.be/ub4jg\\_5D-Eo](https://youtu.be/ub4jg_5D-Eo), July 12, 2015.

<sup>2</sup>Personal website [www.martinebner.at](http://www.martinebner.at), July 12,2015

<sup>3</sup>Course website <http://www.coer13.de/news.html>, July 12, 2015.

## 6 Evaluation

*Educational Resources* (OER) including those who want to work with it. Topics discussed were

- What is OER ?
- Searching, finding and creating OER content
- Usage scenarios
- Funding of OER

The course was split into six units which built on one another and started on May 11, 2015. Each week a new unit got activated. Within each unit, at least one short video and a couple of external resources were published. In that course, *each* unit implemented a self-assessment quiz. Users who passed all (6) quizzes and additionally filled out the final questionnaire<sup>4</sup> not only activated their personal certificate of participation (an example is shown in figure 5.2) but also received the corresponding COPB (see figure 6.1d).

Additionally, each user got a certain micro-badge for passing

- Quiz 1 and 2 (led to badge COER15 1/3, see figure 6.1a)
- Quiz 1-4 (led to badge COER15 2/3, see figure 6.1b)
- Quiz 1-6 (led to badge COER15 3/3, see figure 6.1c)

The naming convention of the QMBs (1/3, 2/3 and 3/3) indicates the progress within the course. One third thereby means that the user has passed one third of all quizzes, and therefore units in the case of COER15, of that course.



Figure 6.1: All achievable badges of the COER15 iMooX course.

<sup>4</sup>This survey served as feedback for the course and the iMooX platform itself. Based on that data, iMooX as well as the course responsible lecturers try to improve their offer.

## 6 Evaluation

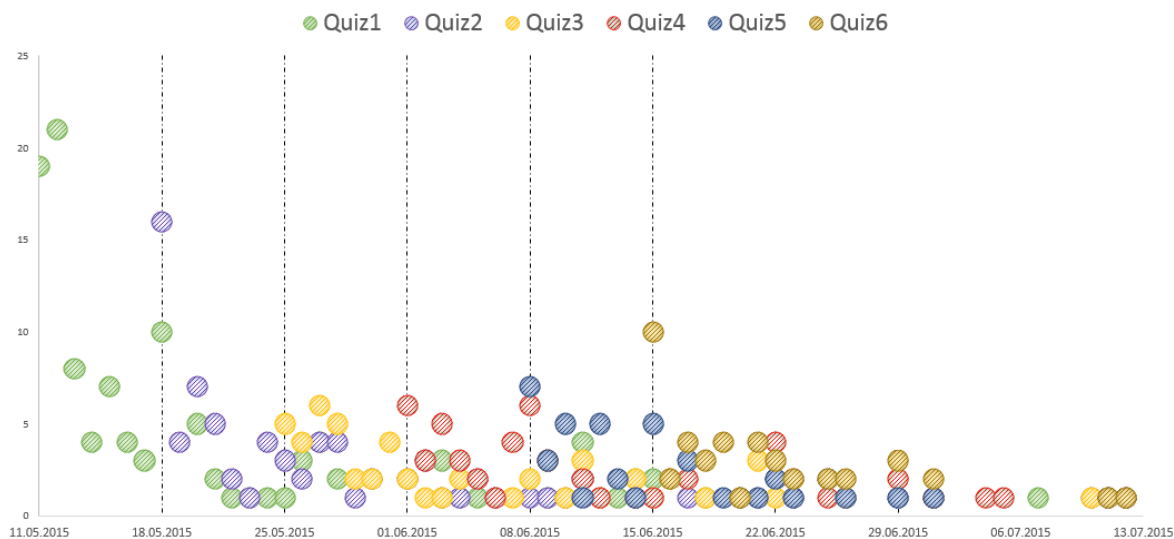


Figure 6.2: Course wide quiz attempts over dates. Vertical lines indicate the start date of the corresponding course unit.

### 6.2 Statistics

After describing what the course COER15 was about and what the user had to do to achieve the course badges, let's look at some numbers. The course had 435 distinct enrollments. That was 12.24% above the average (=387.6) which made it the sixth<sup>5</sup> largest course on iMooX. Three accounts were removed because those were dummy accounts for the lecturer, an iMooX developer and the author of this thesis. It probably makes sense to just consider those, who actively participated in the course, which results in 28,7% (124) who at least tried one quiz.

First, it may be interesting if users self-assessed them after each unit or if they made all quizzes right after each other at the end of the course. Figure 6.2 illustrates the course wide amount of quiz attempts per date. It shows, that most of the users made their quizzes according to the current discussed unit. For example, about 54% of all users that attended the first quiz did that before the second unit was actually activated. Similar ratios were observed for the remaining quizzes. Because it was not mandatory to take the quizzes from the first to the last in exactly that order, one cannot simply take the difference of quiz attempts to identify dropout rates. For such statistics, one would need to merge the resulting lists of quiz attendees, but for this evaluation, such data does not matter. However, 62.9% (78) of the active users theoretically earned the first (6.1a), 41.1% (51) the second (6.1b) and 34.7% (43) the third (6.1c) QMB. 52.6% (41), 68.6% (35) and 76.7% (33) of those who had theoretically earned the first, second and third QMB respectively, also generated

<sup>5</sup>Out of 20 courses that time. The leading course "online learning" had 1081 enrollments.

## 6 Evaluation

them<sup>6</sup>. This could be a consequence of not announcing and promoting these micro-badges or may imply that those who never generated their first badge also did not attend the subsequent quizzes and probably did not even return to the iMooX page at all. Those who recognized their first earned badges probably generated their following on purpose.

Nevertheless, 29% (36) of the active users completed all quizzes *and* additionally filled out the questionnaire. Those 36 users therefore also deserved the COPB (6.1d), which was generated by 77.8% (28) of them. Slightly more women (55.6%) earned all badges and the average age of the course graduates was about 44. An overview of the course badges and their awarding statistics is shown in table 6.1.

Summarized, 432 enrolled users theoretically earned 208 badges. 137 of them were actually generated<sup>7</sup>.





				
Theoretically earned	78	51	43	36
Actually generated	41	35	33	28

Table 6.1: Overview of earned and generated COER15 course badges.

### 6.3 Lessons Learned

Fortunately, there were no problems regarding observance of the Open Badges specification or the awarding process itself. So the developed application worked well from the functional point of view. Nevertheless, thanks to a few badge earners which gave voluntary feedback, a couple of usability issues could be revealed and fixed.

1. There was a request concerning the filename of the downloaded badge image file. That was hard-coded set to *badge.png* and the earner wanted to have that file named like the badge to have instant clarity on the local storage device.

---

<sup>6</sup>Recap: A user had to navigate to the personal badge page to trigger the automatic badge generation process. The menu item (in the navigation bar) was located directly next to the personal course as well as profile page. Therefore, it is unlikely that it got overlooked.

<sup>7</sup>Till the day this evaluation was written (July 12, 2015). If the badge owner did not limit the badge availability or archived the badge, then all of those users who did not yet have generated their badges are still able to do so.

## 6 Evaluation

That constructive feedback was instantly considered and the corresponding code was adapted appropriately.

2. There was a complaint about not mentioning that the email address of the earners Mozilla's Backpack has to be equal to the one on iMooX in time. This issue is an administrative one and has nothing to do with the web application. However, that should no longer happen in future because that course was the test run for future ones and there was, as already mentioned in the introductory text of this chapter, no advertising and detailed information about badges. So in future, as badges will become an integral component of iMooX such information will be announced in time.
3. The last constructive feedback concerned the provision of the criteria. Till that feedback, criteria were represented as a list in JSON format. Each badge had a link to such a JSON file enumerating certain criteria. The request was to make that more user readable. Therefore, corresponding code was adapted to generate a HTML file instead of that JSON file. That enabled styling the list of criteria which made it much more readable (prettier) than a raw JSON string.



# 7 Conclusions

In this thesis, the integration of Open Badges in iMooX has been presented. Therefore, Mozilla's Open Badges project, their developed infrastructure and especially the specified badge meta-data standard have been explained within chapter 2 in detail. Chapter 3 has then given an overview of relevant terms regarding educational online learning environments and presented the most used concepts. General integration approaches for existing learning environments have been discussed in chapter 4. The developed web application, presented and explained in chapter 5 in detail, was tested based on the iMooX course COER15 where the corresponding evaluation has been presented in chapter 6.

Although voluntary feedback given by course participants led to minor design changes, like renaming the badge file or redesigning the criteria presentation, the application performed well from a functional point of view. Also the administrator, who has been responsible for defining and managing badges, assessed the back-end as easy and efficient to use. Therefore, it certainly will get used for future courses as well.

Nevertheless, this work revealed a few issues that should be discussed followed by suggested future work regarding the developed web application.

## 7.1 Discussion

One of the most discussed drawbacks of using gamification in educational contexts is that it could lead to the "*Overjustification effect*" [25]. Thereby, rewards can serve to decrease motivation of intrinsically motivated learner. It is therefore important to make badge earning optional for users. Hakulinen, Auvinen, and Korhonen [28] also noticed that by student feedback in their final questionnaire regarding the usage of badges.

However, enabling that feature could simply be achieved by adding a configuration option in a user's account settings that enables or disables the presentation of badge related contents.

It is also likely, that learners prioritize collecting badges higher than their learning outcome. As it is in the case of iMooX where self-assessment quizzes are relatively

## 7 Conclusions

easy to pass, users could exploit that by just doing those quizzes without even actively participating in the course just to get corresponding badges. There is no real solution to that problem, except to make self-assessment quizzes harder to pass. But that would be against its purpose. Those quizzes are intended to offer an opportunity to check oneself if the discussed content has been understood, not to pressure the learner. However, that is a pedagogical issue and should be discussed by corresponding experts.

A more technical issue is that the only information a badge has about its owner is its email address. Although that is commonly used on the Internet to register to accounts and suffices for most purposes, what if an earner would like to change the email address, for example<sup>1</sup> because of a name change due to marriage? If the issuer does not provide the possibility to delete the old and award a new badge, the earner has simply bad luck as just modifying the embedded meta-data and bake it again into the image file does not work. Although for hosted badges, that may seem possible, but in that case the issuer holds the verification data on its web server. So if someone wants to validate the receipt of a certain badge, both data, the one embedded in the image file and the corresponding data on the issuer's web server have to be equal. For signed badges it should be obvious that it is impossible.

Then, what if, for whatever reason, a user does not use his email address for a long time and the corresponding provider assigns it to another user<sup>2</sup>? In the worst case, the badges of user A are also valid for user B, if they have the same email address. That is particularly a problem for badge issuers where the email address of a user is not unique. Suppose user A registers to an issuer site, participates actively and earns badges. User A then, for whatever reason, loses the email address of that account. Another user B gets exactly that email address assigned and registers to the same issuer which would be legit. The problem then is, that all badges user A has earned are now also valid for user B, if the badging system does only operate on email addresses and not on additional user information like the user name for example. That sounds trivial, but because of data privacy typically just email addresses are used within badging systems.

A crucial aspect of badging systems is the handling of the signature key. If badges were signed with a certain private key and then why ever that key changes, all signed badges which have already been awarded would be invalid. That definitely must not happen to issuers as they would need to award each badge again, or must be able to recover the previously used key. Depending on the implementation, that may not even be possible.

---

<sup>1</sup>Other examples may be students who graduated or employees changed their company.

<sup>2</sup>GMX ([www.gmx.at](http://www.gmx.at), August 22, 2015.) for example does that after 12 months inactivity.

The upshot was, Open Badges provide a nice way to recognize, acknowledge or motivate learners and should definitely be considered in future research. If done properly, it has the potential to become a serious alternative to existing credentialing systems. It is also the authors believe, that badge *concept and design* decides whether users accept and appreciate their usage or not. If an issuer awards too many badges or they are too easy to achieve, they get petty. If they are too hard to get or the criteria are not transparent and reasonable, users will not even try to reach them. So designing and defining badges should be done seriously and with concept, not licentious.

Independent of the chosen integration method, becoming an issuer is relatively easy. If using software-as-a-service or taking an already existing open-source solution is considered, than issuing is even possible within hours.

### 7.2 Future Work

In addition to the already suggested future works throughout that thesis, some improvements regarding the front-end should be considered.

Gamification suggests to show the user what can be achieved [25, 56]. Therefore, available badges should be presented in a way the user can distinguish which badge has already been achieved and which not. In addition, the learner should see a progress and what it needs to complete a badge *before* it is awarded.

As already mentioned in the discussion above, there should be an option to enable or disable badge support. If a user does not want to earn badges, then this option should be configurable by the user.

Then, it may be useful to give the administrator the right and option to revoke already awarded badges, as it is currently not possible to do so. A use case may be that a user overlooked the fact that it is mandatory to have the same email address for iMooX as for the users backpack. Then, the administrator could revoke the badges and allow the user to change the iMooX email address. Navigating to the users badge page would then trigger the awarding process to the new email address. However, that should be an exception anyway.

Finding ways to track issued badges might be an interesting future work, because then, iMooX would be able to monitor where their badges get shared, stored and applied. It might reveal the value of certain badges and allow iMooX to adapt or review design or concept questions. In addition, that may help to promote courses or at least the platform itself.

## 7 Conclusions

Indicating which badges got newly generated might be sensible for the user to stay on top of earned badges. That could be realized by simply adding a CSS image overlay to the presented badge images. Of course, the back-end has to add an additional indicator for newly generated badges within the response to the assertion generation process triggered by the user.

As a last suggestion, various APIs should be utilized to let the user share earned badges as easy as possible. Recommended sharing targets may be social media platforms like Twitter, Google+, LinkedIn and Facebook, different backpacks like the one from Mozilla or OpenBadgesPassport and E-Portfolio platforms like Mahara or WordPress. Ideally, pushing earned badges to the platforms mentioned should be also made automatically on newly earned badges, if the user configures that appropriately.

# Nomenclature

AJAX .....	Asynchronous JavaScript And XML
API .....	Application Programming Interface
ASCII .....	American Standard Code for Information Interchange
CAL .....	Computer Assisted Learning
CAT .....	Computer Assisted Teaching
CBT .....	Computer Based Training
CDATA .....	Character Data
CMS .....	Content Management System
COER .....	Course for Open Educational Resources
COPB .....	Certificate of Participation Badge
CrMS .....	Course Management System
CRUD .....	Create Read Update and Delete
CSS .....	Cascading Style Sheet
D&D .....	Drag and Drop
DAO .....	Data Access Object
DER .....	Distinguished Encoding Rules
ECTS .....	European Credit Transfer System
ERM .....	Entity Relationship Model
HTML .....	Hyper Text Markup Language
HTTP .....	Hypter Text Transfer Protocol
IRI .....	Internationalized Resource Identifier
ISO .....	International Organization for Standardization
JDK .....	Java Development Kit
JRE .....	Java Runtime Environment
JSON .....	JavaScript Object Notation
JSP .....	Java Servlet Pages
JWS .....	Java Web Signature
LCMS .....	Learning Content Management System
LMS .....	Learning Management System

## 7 Conclusions

LO.....	Learning Object
MAC.....	Message Authentication Code
MC-Tests.....	Multiple Choice Tests
MOOC.....	Massive Open Online Course
OBI.....	Open Badges Infrastructure
OER.....	Open Educational Resources
P2PU.....	Peer 2 Peer University
PDF.....	Portable Document Format
PEM.....	Privacy-enhanced Electronic Mail
PKC.....	Public Key Cryptography
PLE.....	Personal Learning Environment
PNG.....	Portable Network Graphics
Q&A.....	Question and Answer
QMB.....	Quiz Mastery Badge
RDBM.....	Relational Database Model
RSA.....	Rivest-Shamir-Adleman
SDK.....	Software Development Kit
SDT.....	Self-Determination Theory
SHA.....	Secure Hashing Algorithm
SQL.....	Structured Query Language
SVG.....	Scaled Vector Graphics
UID.....	Unique Identifier
URL.....	Uniform Resource Locator
WBT.....	Web Based Training

# Bibliography

- [1] Katharina Albrecht. "Konzeption, prototypische Umsetzung und Evaluation der Mozilla Open Badges für Massive Open Online Courses (MOOCs) am Beispiel des Saxon Open Online Course (SOOC)". Master Thesis. Technische Universität Chemnitz, 2013 (cit. on p. 43).
- [2] Ashton Anderson et al. "Steering User Behavior with Badges". In: *Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2013, pp. 95–105. ISBN: 9781450320351 (cit. on pp. 1, 2).
- [3] Niniek Angkasaputra et al. "The collaborative learning methodology CORONET-Train: Implementation and guidance". In: *Advances in Learning Software Organizations*. Vol. 2640. 2002, pp. 13–24. ISBN: 3-540-20591-8 (cit. on p. 48).
- [4] Judd Antin and Elizabeth F. Churchill. "Badges in social media: A social psychological perspective". In: *CHI 2011*. 2011, pp. 1–4. ISBN: 9781450302685 (cit. on p. 1).
- [5] Graham Attwell. "The Personal Learning Environments - the future of eLearning?" In: *eLearning Papers 2*. January (2007), pp. 1–8. URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.97.3011%5C&rep=rep1%5C&type=pdf> (cit. on p. 33).
- [6] BadgeAlliance. *Badge Endorsement: Getting Started*. 2014. URL: [https://docs.google.com/document/d/1VVf19d72KmGMh1ywrLe7HCKE0qGSI0WjvwfGN%5C\\_8Q2M4/edit](https://docs.google.com/document/d/1VVf19d72KmGMh1ywrLe7HCKE0qGSI0WjvwfGN%5C_8Q2M4/edit) (visited on 07/28/2015) (cit. on p. 21).
- [7] BadgeAlliance. *Open Badges Specification Version 1.1*. 2015. URL: <https://openbadgespec.org/history/1.1.html> (visited on 08/15/2015) (cit. on pp. 12, 13).
- [8] Stephen P. Balfour. "Assessing writing in MOOCs: Automated essay scoring and Calibrated Peer Review". In: *Research & Practice in Assessment* (2013), pp. 40–48. URL: <http://www.rpajournal.com/dev/wp-content/uploads/2013/05/SF4.pdf> (cit. on p. 40).

## Bibliography

- [9] Peter Berking and Shane Gallagher. *Choosing a learning management system*. Tech. rep. 2013. URL: <http://www.adlnet.gov/wp-content/uploads/2014/12/Choosing-an-LMS-1.pdf> (cit. on pp. 31–33).
- [10] Robert Bjork. *Applying Cognitive Psychology to Enhance Educational Practice*. URL: <http://bjorklab.psych.ucla.edu/research.html> (visited on 07/26/2015) (cit. on p. 38).
- [11] Charles Boutell. *The Handbook to English Heraldry*. Ed. by Fox-Davies A.C. London: Cassell, Petter, and Galpin, 1867. URL: <http://www.gutenberg.org/ebooks/23186> (cit. on p. 8).
- [12] John Bradley, Nat Sakimura, and Michael B. Jones. *JSON Web Signature (JWS)*. 2015. URL: <https://tools.ietf.org/html/rfc7515> (visited on 07/26/2015) (cit. on p. 8).
- [13] Ilona Buchem, Kreutel Jörn, and Agathe Merceron. *fMOOC - Beuth Hochschule für Technik Berlin*. 2015. URL: <https://projekt.beuth-hochschule.de/fmooc/> (visited on 08/03/2015) (cit. on p. 35).
- [14] Shing Wai Chan and Rajiv Mordani. *Java Servlet Specification Version 3.1*. Tech. rep. Oracle Corporation, 2013. URL: <http://download.oracle.com/otndocs/jcp/servlet-3.0-fr-oth-JSpec/> (cit. on p. 59).
- [15] John Cook and Matt Smith. “Beyond formal learning: Informal community eLearning”. In: *Computers & Education* 43.1-2 (Aug. 2004), pp. 35–47. DOI: [10.1016/j.compedu.2003.12.003](https://doi.org/10.1016/j.compedu.2003.12.003) (cit. on p. 1).
- [16] Simon Cross, Denise Whitelock, and Rebecca Galley. “The use, role and reception of open badges as a method for formative and summative reward in two Massive Open Online Courses”. In: *International Journal of e-Assessment* 4 (July 2014). URL: <http://oro.open.ac.uk/40593/> (cit. on pp. 2, 37).
- [17] Edward L. Deci and Richard M. Ryan. *Intrinsic Motivation and Self-Determination in Human Behavior*. 1985. ISBN: 0306420228 (cit. on p. 30).
- [18] Sebastian Deterding et al. “From game design elements to gamefulness: Defining “Gamification””. In: *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11*. Tampere, Finland, 2011, pp. 9–11. ISBN: 9781450308168. DOI: [10.1145/2181037.2181040](https://doi.org/10.1145/2181037.2181040) (cit. on p. 2).
- [19] Romain Dillet. *OpenClassrooms Launches First MOOC-Based Bachelor Degree Recognized By French State*. 2015. URL: <http://techcrunch.com/2015/07/07/openclassrooms-launches-first-mooc-based-bachelor-degree->



## Bibliography

- [recognized-by-french-state/%5C#.ease1z:0sU0](#) (visited on 07/17/2015) (cit. on p. 35).
- [20] Adrián Domínguez et al. “Gamifying learning experiences: Practical implications and outcomes”. In: *Computers and Education* 63 (2013), pp. 380–392. ISSN: 03601315. DOI: [10.1016/j.compedu.2012.12.020](#) (cit. on pp. 2, 44).
- [21] Martin Ebner. *iMooX - eine Plattform für Online-Kurse der Uni Graz und der TU Graz*. 2015. URL: <http://de.slideshare.net/mebner/symposium-48534797?ref=http://elearningblog.tugraz.at/archives/7984> (visited on 07/19/2015) (cit. on p. 47).
- [22] Richard Elliott, John Clayton, and J. Iwata. “Exploring the use of micro-credentialing and digital badges in learning environments to encourage motivation to learn and achieve”. In: *In B. Hegarty, J. McDonald, & S.-K. Loke (Eds.), Rhetoric and Reality: Critical perspectives on educational technology*. 2014, pp. 703–707. URL: <http://ascilite2014.otago.ac.nz/files/concisepapers/276-Elliott.pdf> (cit. on p. 2).
- [23] Jonathan Finkelstein, Erin Knight, and Susan Manning. *The Potential and Value of Using Digital Badges for Adult Learners*. Draft for Public Comment. 2013 (cit. on p. 2).
- [24] Helge Fischer et al. “Revenue vs. costs of MOOC platforms. Discussion of business models for xMOOC providers, based on empirical findings and experiences during implementation of the project iMooX”. In: *7th International Conference of Education, Research and Innovation*. November. Seville, Spain, 2014, pp. 2991–3000. ISBN: 978-84-617-2484-0 (cit. on p. 47).
- [25] Ian Glover. “Play as you learn: gamification as a technique for motivating learners”. In: *World Conference on Educational Multimedia Hypermedia and Telecommunications*. Chesapeake, VA: AACE, 2013, pp. 1999–2008. ISBN: 9781939797032 (cit. on pp. 79, 81).
- [26] Emily Goligoski. “Motivating the learner: Mozilla’s open badges program”. In: *Access to Knowledge: A Course Journal* 4.1 (2012), pp. 1–8. URL: <http://ojs.stanford.edu/ojs/index.php/a2k/article/view/381> (cit. on p. 2).
- [27] Lassi Haaranen et al. “How (not) to introduce badges to online exercises”. In: *Proceedings of the 45th ACM technical symposium on Computer science education - SIGCSE '14* (2014), pp. 33–38. DOI: [10.1145/2538862.2538921](#) (cit. on pp. 2, 43).
- [28] Lasse Hakulinen, Tapio Auvinen, and Ari Korhonen. “Empirical Study on the Effect of Achievement Badges in TRAKLA2 Online Learning Environment”.

## Bibliography

- In: *2013 Learning and Teaching in Computing and Engineering* (Mar. 2013), pp. 47–54. DOI: [10.1109/LaTiCE.2013.34](https://doi.org/10.1109/LaTiCE.2013.34) (cit. on pp. 2, 79).
- [29] CB Hodges. “Designing to motivate: Motivational techniques to incorporate in e-learning experiences”. In: *The Journal of Interactive Online Learning* 2.3 (2004), pp. 1–7. ISSN: 15414914. URL: <http://www.ncolr.org/jiol/issues/pdf/2.3.1.pdf> (cit. on p. 2).
- [30] Bernhard Hoisl, Wolfgang Aigner, and Silvia Miksch. *Online Communities and Social Computing*. Ed. by Douglas Schuler. Vol. 4564. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 362–371. ISBN: 978-3-540-73256-3. DOI: [10.1007/978-3-540-73257-0](https://doi.org/10.1007/978-3-540-73257-0) (cit. on pp. 1, 2).
- [31] IICM. *WBT-Master*. URL: <http://coronet.iicm.tugraz.at/wbtmaster/welcome.html> (visited on 07/19/2015) (cit. on p. 48).
- [32] Simon Josefsson. *The Base16, Base32, and Base64 Data Encodings*. 2006. URL: <http://tools.ietf.org/html/rfc4648> (visited on 07/26/2015) (cit. on pp. 5, 6).
- [33] Andrea Kiesel and Iring Koch. *Lernen - Grundlagen der Lernpsychologie*. 1st ed. 2012, pp. 73–81. ISBN: 978-3-531-17607-9 (cit. on p. 28).
- [34] Marc Krueger and Markus Schmees. *E-Assessments in der Hochschullehre: Einführung, Positionen & Einsatzbeispiele Psychologie Und Gesellschaft*. 1st ed. Lang, Peter Frankfurt, 2013. ISBN: 978-3631641514 (cit. on p. 38).
- [35] Elke Lackner and Michael Kopp. “Lernen und Lehren im virtuellen Raum - Herausforderungen, Chancen, Möglichkeiten”. In: *Rummler, Klaus: Lernraeume gestalten - Bildungskontexte vielfältig denken*. 67 (Jan. 2014), pp. 174–186 (cit. on p. 29).
- [36] Joey J. Lee and Jessica Hammer. *Gamification in Education: What, How, Why Bother?* Tech. rep. 2011. URL: [https://www.academia.edu/570970/Gamification%5C\\_in%5C\\_Education%5C\\_What%5C\\_How%5C\\_Why%5C\\_Bother](https://www.academia.edu/570970/Gamification%5C_in%5C_Education%5C_What%5C_How%5C_Why%5C_Bother) (cit. on p. 2).
- [37] Elena Martin-Monje and Elena Barcena. “What Constitutes an Effective Language MOOC?” In: *Language MOOCs. DE GRUYTER OPEN*, 2015, pp. 16–30. ISBN: 978-3-11-042250-4 (cit. on pp. 35, 36).
- [38] Stephen McLaughlin et al. *e-Skills and ICT Professionalism - Fostering the ICT Profession in Europe*. Tech. rep. IVI, CEPIS, 2012. URL: [http://www.cepis.org/media/EU%5C\\_ICT%5C\\_Professionalism%5C\\_Project%5C\\_%20FINAL%5C\\_REPORT.pdf](http://www.cepis.org/media/EU%5C_ICT%5C_Professionalism%5C_Project%5C_%20FINAL%5C_REPORT.pdf) (cit. on p. 27).

## Bibliography

- [39] Merriam-Webster.com. "Learning". URL: <http://www.merriam-webster.com/dictionary/learning> (visited on 08/11/2015) (cit. on p. 27).
- [40] Joi L. Moore, Camille Dickson-Deane, and Krista Galyen. "e-Learning, online learning, and distance learning environments: Are they the same?" In: *The Internet and Higher Education* 14.2 (Mar. 2011), pp. 129–135. DOI: 10.1016/j.iheduc.2010.10.001 (cit. on p. 32).
- [41] Mozilla. "Open Badges for Lifelong Learning". 2012. URL: [https://wiki.mozilla.org/images/5/59/OpenBadges-Working-Paper%5C\\_012312.pdf](https://wiki.mozilla.org/images/5/59/OpenBadges-Working-Paper%5C_012312.pdf) (cit. on p. 2).
- [42] Netcraft. *June 2015 Web Server Survey — Netcraft*. 2015. URL: <http://news.netcraft.com/archives/2015/06/25/june-2015-web-server-survey.html> (visited on 07/19/2015) (cit. on p. 49).
- [43] NISO. *Understanding Metadata*. NISO Press, 2004, p. 20. ISBN: 1-8800124-62-9. URL: <http://www.niso.org/publications/press/UnderstandingMetadata.pdf> (cit. on p. 11).
- [44] Oracle. *JavaServer Pages Technology*. URL: <https://jcp.org/aboutJava/communityprocess/final/jsr245/index.html> (visited on 07/20/2015) (cit. on p. 60).
- [45] Oracle. *UID (Java Platform SE 6)*. URL: <http://docs.oracle.com/javase/6/docs/api/java/rmi/server/UID.html> (visited on 07/21/2015) (cit. on p. 71).
- [46] Laura Pappano. *Massive Open Online Courses Are Multiplying at a Rapid Pace - The New York Times*. New York, USA, Nov. 2012. URL: <http://www.nytimes.com/2012/11/04/education/edlife/massive-open-online-courses-are-multiplying-at-a-rapid-pace.html> (cit. on p. 35).
- [47] Morten Flate Paulsen. "Online Education Systems: Discussion and definition of terms". In: *NKI Distance Education* (2002), pp. 1–8. URL: <http://www.porto.ucp.pt/open/curso/modulos/doc/Definition%20of%20Terms.pdf> (cit. on p. 32).
- [48] Umesh Hodeghatta Rao and Umesha Nayak. "Cryptography". In: *The InfoSec Handbook*. Berkeley, CA: Apress, 2014. Chap. 8, pp. 163–181. ISBN: 978-1-4302-6382-1. DOI: 10.1007/978-1-4302-6383-8 (cit. on pp. 4, 6, 7).
- [49] R.L Rivest, A. Shamir, and Adleman L. *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*. Tech. rep. MIT, 1977. URL: <http://people.csail.mit.edu/rivest/Rsapaper.pdf> (cit. on p. 6).
- [50] Richard M. Ryan and Edward L. Deci. "Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions." In: *Contemporary Educational Psychol-*

## Bibliography

- ogy 25.1 (Jan. 2000), pp. 54–67. ISSN: 0361-476X. DOI: [10.1006/ceps.1999.1020](https://doi.org/10.1006/ceps.1999.1020) (cit. on pp. 30, 38).
- [51] Carlos Santos et al. “Students’ perspectives on badges in educational social media platforms: the case of SAPO campus tutorial badges”. In: *Proceedings - 2013 IEEE 13th International Conference on Advanced Learning Technologies, ICALT 2013*. Beijing: IEEE, 2013, pp. 351–353. DOI: [10.1109/ICALT.2013.108](https://doi.org/10.1109/ICALT.2013.108) (cit. on pp. 2, 34).
- [52] JoseLuis Santos et al. “Evaluating the use of open badges in an open learning environment”. In: *Scaling up Learning for Sustained Impact*. Ed. by Davinia Hernández-Leo et al. Vol. 8095. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 314–327. ISBN: 978-3-642-40813-7. DOI: [10.1007/978-3-642-40814-4](https://doi.org/10.1007/978-3-642-40814-4) (cit. on pp. 2, 37, 45).
- [53] Werner Sauter and Annette Kulmann. *Innovative Lernsysteme Kompetenzentwicklung mit Blended Learning und Social Software*. 2008. ISBN: 9783540778301. DOI: [10.1007/978-3-540-77831-8](https://doi.org/10.1007/978-3-540-77831-8) (cit. on pp. 1, 28, 29, 39).
- [54] Sandra Schaffert and Wolf Hilzensauer. “On the way towards Personal Learning Environments : Seven crucial aspects”. In: *eLearning Papers* 9.July (2008), pp. 1–11. URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.167.4083> (cit. on p. 31).
- [55] George Siemens. *Connectivism: A Learning Theory for the Digital Age*. 2004. URL: <http://www.elearnspace.org/Articles/connectivism.htm> (visited on 07/14/2015) (cit. on p. 29).
- [56] Jorge Simeoes et al. “A gamification framework to improve participation in social learning environments”. In: November (2013), pp. 1–11. ISSN: 1887-1542. URL: [https://www.academia.edu/4793907/A%5C\\_Gamification%5C\\_Framework%5C\\_to%5C\\_Improve%5C\\_Participation%5C\\_in%5C\\_Social%5C\\_Learning%5C\\_Environments](https://www.academia.edu/4793907/A%5C_Gamification%5C_Framework%5C_to%5C_Improve%5C_Participation%5C_in%5C_Social%5C_Learning%5C_Environments) (cit. on p. 81).
- [57] Behnam Taraghi et al. “Personal Learning Environment – a Conceptual Study”. In: *ICL 2009 Proceedings*. Vol. 1. 10. Villach, 2009, pp. 997–1006 (cit. on p. 34).
- [58] The European Centre for the Development of Vocational Training (Cedefop). *European guidelines for validating non-formal and informal learning*. Luxembourg: European Centre for the Development of Vocational Training, 2009. ISBN: 978-92-896-0602-8 (cit. on p. 38).
- [59] The European Centre for the Development of Vocational Training (Cedefop). *Recognition and validation of non-formal and informal learning for VET teachers and trainers in the EU Member States*. Luxembourg: Cedefop Panorama series, 2007. ISBN: 978-92-896-0496-3 (cit. on p. 27).

## Bibliography

- [60] Katrin Vogt and Andrea Hechenleitner. *Theorien des Lernens. Folgerungen für das Lehren*. Tech. rep. München, 2007. URL: <https://www.isb.bayern.de/gymnasium/materialien/t/theorien-des-lernens/> (cit. on pp. 28, 29).
- [61] Michael Vogt and Stefan Schneider. *E-Klausuren an Hochschulen : Didaktik – Technik – Systeme – Recht – Praxis*. German. Tech. rep. Gießen: Justus-Liebig-Universität Gießen Koordinationsstelle Multimedia, 2009. URL: <http://geb.uni-giessen.de/geb/volltexte/2009/6890/> (cit. on pp. 39, 40).
- [62] Tom Warger and Gregory Dobbin. “Learning environments: where space, technology, and culture converge”. In: *EDUCAUSE Learning Initiative* October (2009), pp. 1–14. URL: <http://net.educause.edu/ir/library/pdf/eli3021.pdf> (cit. on pp. 1, 31, 32).