Project no. 619347

EAGLE- EnhAnced Government LEarning

Objective ICT-2013.8.2 Technology-enhanced learning;

c) Holistic learning solutions for managing, reaching and engaging learners in the public administrations

Small-scale Collaborative Project (STREP)
FP7-ICT-2013-11

www.fp7-eagle.eu

Deliverable D5.1

OER-LP Design Principles

WP 5 – OER Learning Platform Development
Lead Participant: Fraunhofer FOKUS

Approval Panel	Name / Partner short name	Department / / Function	Date
Author	Arun Prakash Santosh Kumar Rajaguru (Fraunhofer)	Senior Researcher Project Assistant	21/05/2014- 30/10/2014
Author	Violeta Damjanovic-Behrendt Dietmar Glachs Felix Strohmeier (SRFG)	Senior Researcher Senior Researcher Senior Researcher & Software Architect	21/05/2014- 27/10/2014
Author	Jan Pawlowski Julia Stoffregen (HRW)	Professor Project Assistant	21/05/2014- 24/10/2014
Author	Eric Tobias (TUDOR)	R&D Engineer	21/05/2014- 24/10/2014
Reviewer	Violeta Damjanovic-Behrendt (SRFG)	Senior Researcher	27/10/2014

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement No 619347.

Table of Contents

List of Figures

List of Tables

Executive Summary

This deliverable presents design principles, software development methodology, and software standards that will be employed during the development of the EAGLE OER-LP software platform. The target audience for this deliverable is a team of EAGLE developers, thus serving as their go-to reference handbook. EAGLE OER-LP is based on development experiences and findings from the OpenScout project, which created a federated repository for management of Open Educational Resources (OERs). For example, we consider the OpenScout application profile for the repository to be reused in EAGLE.

In addition, Section 2 of this deliverable provides a detailed overview of the metadata specifications and interoperability standards for describing learning objects, learning and educational resources. For example, we review Learning Object Metadata (LOM) standard, Dublin Core (DC), Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), IMS Learning Technology Interoperability (IMS LTI), IMS Question and Test Interoperability (IMS QTI), IMS Accessibility for All. Based on our analysis of a current set of metadata specifications and interoperability standards for learning and educational resources, we propose the usage of an RDF representation of the LOM standard, OAI-PMH for harvesting, but also IMS QTI, IMS LTI and finally, IMS Accessibility for All.

In addition, this deliverable provides a review of Learning Management Systems (LMS), and specifically LMS in public administration. We also review open-data platforms and tools, such as GovData.DE Open Data platform, TAO platform for testing and item generation in computer based assessment. Herein, we also provide overview of EAGLE background technologies such as Semantic Web technologies, Representational State Transfer (REST), Linked Data, and finally, Linked Data platform such as Apache Marmotta.

Section 3 of this deliverable discusses the design principles that should be used for the design of EAGLE OER-LP components and modules. For example, core design principles in EAGLE are based on SOLID principles, such as **S**ingle responsibility, **O**pen closed principle, **L**iskov's substitution principle, **I**nterface segregation principle, and **D**ependency inversion principle. Regarding the software development methodology in EAGLE, we base our software development on established methods used in existing open source projects, such as Apache Marmotta. We follow an agile software development process, with short iteration cycles and small modules, each of them having its own planning, requirements analysis, design, coding, testing, and documentation phase. In addition, all EAGLE sources and related resources must be stored in a common source code repository, such as a Git repository. Section 3.2 of this deliverable describes EAGLE build and deployment process, issue tracking, conventions (naming conventions, coding conventions, etc.), messaging and testability.

Glossary

CKAN	Comprehensive Knowledge Archive Network (CKAN) is used by various national and local government as an open source data management system for the storage and distribution of data. .
Creative Commons license	Creative Commons (CC) licenses provide an easy way to manage the copyright terms that attach automatically to all creative works under copyright. The licenses allow those works to be shared and re-used under terms that are flexible and legally sound. Creative Commons offers a core suite of six copyright licenses. Because there is no single "Creative Commons license," it is important to identify which of the six licenses you are applying to your work, and which of the six licenses has been applied to a work you intend to use. .
e-Enabling	E-enabling is a term that refers to the process of enabling the interaction with the outside world via electronic networks such as the Internet and the World Wide Web. The "E" means "electronic networks" and describes the application of electronic network technology, including Internet and electronic data interchange (EDI) to improve and change business processes. E-enabling is closely identified with E-Commerce, E-Business and the Knowledge Economy (k-Economy) (Karensoft Group).
GMS	eGovernment Metadata Standard
GNU license	The GNU (General Public License) GPL is a free, copyleft license for software and other kinds of works. (Free Software Foundation).
IMS	(Instructional Management Systems) IMS is a consortium in the learning technology sector supporting the Open Foundation and developing open interoperability standards, adoption with technical services, and encourages adoption through programs that highlight effective practices
IMS LIP	IMS (Learner Information Package) LIP
IMS QTI	IMS (Question & Test Interoperability) QTI is a standardized data format for online learning material.
Linked Data	Describes a method of publishing structured data so that data can be interlinked and more useful. It builds upon standard Web technologies such as HTTP and URIs, but rather than using them to serve web pages for human readers, it extends them to share information in a way that can be read automatically by computers. This enables data from different sources to be connected and queried (Bizer, Heath and Berners-Lee).
LOM	Learning Object Metadata
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) is a low-barrier mechanism for repository interoperability
OER-LP	(OER Learning Platform) OER LP

Open Data	Data is open if anyone is free to use, reuse, and redistribute it (Open Knowledge Foundation).
OER	(Open Educational Resources) OERs are teaching, learning or research materials that are in the public domain or released with an intellectual property license that allows for free use, adaptation, and distribution (UNESCO).
OpenScout	OpenScout provides a free platform around open content for business and management education.
Open Source Software (OSS)	When a software program is open source, the program's source code is freely available to the public. Unlike commercial software, open source programs can be modified and distributed by anyone and are often developed as a community rather than by a single organization.
OWL	Web Ontology Language (OWL) is a specification of W3C to write, publish and distribute ontologies in a formal language.
RDF	Resource Description Framework (RDF) was developed by W3C as a standard to describe metadata. It is considered one of the main components of the Semantic Web.
REST	REST stands for Representational State Transfer and it is an architecture for designing networked applications.
SPARQL	SPARQL Protocol And RDF Query Language is a graph-based query language for RDF.
TAO	TAO is the French acronym for Testing Assisté par Ordinateur (Computer Based Testing). It is an open architecture for computer-assisted test development and delivery and was developed by the EMACS research unit of the University of Luxembourg and the SSI department of the Centre de Recherche Public Henri Tudor.
W3C	World Wide Web Consortium

1 Introduction

As described in the *Description of Work* (*DoW*) – *Part B* [1], the EAGLE project would develop an open source Learning Platform (LP) that would be adapted for individual local governments. EAGLE LP will be a novel, extensible platform that would aggregate, refine and use OERs (Open Educational Resource) based on existing tools and frameworks such as Ariadne, OpenScout and the *state of the art* in Linked Data [1]. In this deliverable, we describe the design principles and the software development methodologies that would be employed for the realization of the **EAGLE OER-LP**.

1.1 Scope of the Deliverable

Work Package 5 (WP5) deals with the design and development of the core EAGLE OER-LP. The main objectives of this WP involve the design of the OER Data registry (for the harvesting and management of OERs), enrichment of the OER information with Linked Data and provision of navigation, user & community services that furnish a rich learning environment for the users. The platform is planned to be developed in two iterations. Thus, the activities of the WP are divided into 6 tasks, with Task 5.1 committed to the design and architecture of the OER-LP.

The application of design principles and a development methodology is important to the creation of good software. They both provide a framework for the design, development and maintenance of the software and for the planning, control and execution of its development process. A bad design and/or development methodology inevitably results in poor software that affects its usability, and has a direct effect on its success.

The EAGLE OER-LP requires specification of software design principles and a development methodology for its creation and realization. Thus, the scope of this deliverable is to describe the design principles, the development methodology and the software standards that would be employed to realize EAGLE OER-LP. In addition, this deliverable is targeted towards the individual developers and would be their go-to reference handbook.

1.2 Structure of the Deliverable

Deliverable D5.1 is divided into four chapters. In this chapter, we described the scope and structure of deliverable. **Chapter 2** deals with the background and related work. In chapter 2, a detailed overview of the metadata specifications and interoperability standards are provided. Additionally, a review of Learning Management Systems (LMS) and open-data platforms and tools are also provided. **Chapter 3** deals with the description of the actual design principles, guidelines and software development methodology that would be employed for the creation of the EAGLE OER-LP. Finally **Chapter 4** concludes the deliverable and gives an outlook on the development of the EAGLE OER-LP.

2 Background & Related Work

As described in DoW – Part B [1], the initial OER content for the platform would be harvested from existing learning portals. Furthermore, the various components of the platform would be built from other open-source tools and frameworks. Thus, it is necessary that we review existing tools, technologies and standards as they have a direct impact on the design principles, development methodology and design guidelines that we propose for the realization of the EAGLE OER-LP. Thus, in this chapter we document the outcome of our literature survey. In Section 2.1, we document our review of the current metadata specification and interoperability standards. In Section 2.2 we present our conclusion with respect to the metadata standard that should be considered for the EAGLE OER-LP. In Sections 2.3 and 2.4 we present *state of the art* of learning platforms, open data platforms and tools.

2.1 Metadata Specifications and Interoperability Standards

The goal of this section is to provide input regarding interoperability of the portal. The main outcome is an application profile (metadata) for the repository. It is based on findings and development experiences from the OpenScout project [2], which created a federated repository for management and related Open Educational Resources (OERs). The main standards described in this deliverable have been previously successfully applied in the OpenScout project [3]. In addition, we base theoretical considerations of the EAGLE OER-LP on the main outcome of the OpenScout project (see D1.1 [4] and D1.2.1 [5] which are project public reports under the Creative Commons License).

In the following, we briefly describe the main approaches for describing Learning Objects (also Learning / Educational Resources) and their related standards.

2.1.1 Learning Object Metadata (LOM)

Learning Object Metadata (IEEE 1484.12.1 – 2002 Standard) [6] refer to systematic, unified descriptions of resources, intended for learning, informational, or other purposes. The LOM standard has become the most widely used solution for classifying and describing digital resources intended specifically for learning and education.

The LOM standard, usually encoded in XML, includes 76 data elements, covering wide-ranging characteristics attributable to LOs, including their size, level and type of interactivity, and the educational context to which they are best suited. LOM defines all of its data elements in interrelationships that are both hierarchical and iterative. At the top of the hierarchy of LOM elements are nine broad category elements: General, Lifecycle, Meta-metadata, Technical, Educational, Rights, Relation, Annotation and Classification. The category elements each contains sub-elements, which, in turn, often contain further sub-elements. Many of the category elements, sub-elements, and subordinate elements can be repeated. This results in complex hierarchical and iterative structures, allowing for a total of over 16,000 possible, concatenated element repetitions.

Given its relative size and complexity, as well as the fact that it is the first technical e-learning standard to be widely adopted, the implementation of the LOM presents an excellent opportunity for study and research. By looking at how it has been implemented in

projects and in specific metadata records, it is possible to learn valuable lessons about e-learning standards implementation, and how to develop and refine further standards to meet developer and educator needs.

One possible representation of LOM is in Resource Description Framework (RDF) [7], although several challenges regarding its representation can be found. Hence, an adaptation is still necessary to be done; a guideline has been presented in [8] and can serve as guidance for the project.

2.1.2 Dublin Core (DC)

A related metadata specification to LOM is Dublin Core Metadata Element Set [9], known as Dublin Core (DC), which provides a simple, loosely-defined set of elements with some overlap with LOM, and which is useful for sharing metadata across a wide range of disparate services. It is a conceptual schema that can be used to describe a metadata model such as LOM.

The DC metadata standard, defined by ISO in *ISO Standard 15836* [10], and NISO Standard Z39.85-2007 [11], is a simple yet effective element set for describing a wide range of networked resources. The DC standard includes two levels: Simple and Qualified. Simple DC comprises fifteen elements; Qualified DC includes three additional elements (Audience, Provenance and Rights Holder), as well as a group of element refinements (also called qualifiers) that refine the semantics of the elements in ways that may be useful in resource discovery.

The semantics of DC have been established by an international, cross-disciplinary group of professionals from librarianship, computer science, text encoding, the museum community, and other related fields of scholarship and practice. Implementations of DC typically make use of XML [12] and RDF [13], which allows multiple objects to be described without specifying the detail required.

2.1.3 Open Archive Initiative Protocol for Metadata Harvesting (OAI-PMH)

OAI-PMH (Open Archives Initiative Protocol for Metadata Harvesting) [14] is a protocol developed by the Open Archives Initiative. It is used to harvest (or collect) the metadata descriptions of the records in an archive so that services can be built using metadata from many archives. An implementation of OAI-PMH must support metadata representation in DC, but may also support additional representations.

OAI-PMH is based on a client–server architecture, in which a harvester requests information on updated records from the repositories. Requests for data can be based on a *date stamp* range, and can be restricted to named sets defined by the provider.

Within the database layer, OAI-PMH is used for harvesting content and domain metadata. Data describing the usage (usage metadata) is collected using the RSS (Rich Site Summary) protocol. While OAI-PMH is suited to collect changing metadata, RSS is used only when new metadata instances (like in log files) are added.

Both RSS and OAI-PMH build on the same common technologies although their intents are rather different. Both use XML documents that are transported over HTTP, and both can

support multiple vocabularies, although RSS is predominantly used for syndicating content (usually via references to that content), while OAI-PMH is primarily focused on the job of harvesting metadata. RSS defines a simple encapsulation methodology that can be used by several classes of applications - typically (though not limited to) RSS readers, while OAI-PMH defines both a schema and an application-level protocol. RSS is particularly suited to lightweight data transfers to the user desktop or handheld, while OAI-PMH was developed to manage system-to-system processes (typically institutional repository-to-repository synchronizations).

2.1.4 Further Standards

There is variety of standards, which might be relevant for further considerations in the EAGLE project. A careful analysis has been performed by Pawlowski & Kozlov [15].

- CEN MLO-AD (CEN/ISSS, 2008[1]): The main goal of this specification is to provide metadata for learning opportunities, in particular advertising information for content providers. This might be relevant once further content provider get involved as part of sustainability actions.
- IMS LTI (Learning Tools Interoperability[2]) is a specification to connect learning systems with different applications. This might be relevant when external tools are integrated in EAGLE.
- IMS QTI (Question and Test Interoperability, Version 2.1[3]): The specification aims at the interoperability of assessments between different learning (management) systems. Even though this specification is currently not widely used, the project should consider the use in the design and implementation of assessment items.
- IMS Accessibility for All[4] provides accessibility guidelines in the learning context. This should be considered when designing learning services and resources. A related base standard has been developed in the international standardization group ISO/IEC (ISO/IEC 24751 series), which influenced the IMS standard (in a previous version). These standards provide a framework and concrete guidance and should thus be considered carefully. Specific tools for validation against the specification are available (IMS Validator[5]).

2.1.5 Intermediate Conclusion

We believe that at least the LOM standard must be considered to enable interoperability to other repositories. Furthermore, we would recommend utilizing OAI-PMH for harvesting. More specific standards should be considered by the related work package/task. In particular, attention should be paid to *IMS QTI* by WP3, *IMS Accessibility for All* and *IMS LTI* by later tasks of WP5.

[1] http://www.cen-wslt.din.de/sixcms_upload/media/3378/CWA15903.pdf
[2] http://www.imsglobal.org/lti/index.html
[3] http://www.imsglobal.org/question/
[4] http://www.imsglobal.org/accessibility/index.html
[5] http://validator.imsglobal.org/accessibility/

As LOM needs to be implemented in the very first steps of the EAGLE-OER LP development, we particularly focus on specifications presented in the following section.

2.2 LOM in EAGLE

LOM plays a major role in describing the learning objects in federated infrastructures and repositories such as EAGLE. As stated in the previous section, the chosen schema describes all the relevant attributes of learning objects. The centralized repository should follow the LOM schema and encloses the information extracted from each single content repository. Since not all repositories follow the specifications of the standard LOM, a previous step of mapping each singular schema to the federated one is necessary. This is done during the harvesting phase (see Section 2.4.1), where the harvester component accesses each individual repository and transfers the available metadata to the centralized repository.

2.2.1 Classifications

A classification is used to better describe learning objects and for searching/ browsing purposes (for an extensive review of classifications, see the guideline by IFLA, 2010 [16]). As an example, user could browse through a hierarchical classification from their main field of interest to more specific resources.

Here, only few classifications exist for the domain of public administration. Due to the broad field of professions in public administration, even the classification of the Library of Congress (Classification for Political Science [17]) does not provide structures which would support searching / browsing. However, initial candidates, which include country specific resources, are as follows:

- Statistical classification of economic activities in the European Community [18],
- North American Industry Classification System (NAICS) Canada 2012 [19],
- Catalogue des Formations (Luxembourg)[6] providing topic areas for training purposes,
- e-library topics[7] HRMA Montenegro.

[6] http://www.fonction-publique.public.lu/fr/formations/formation-continue/index.html
[7] http://www.uzk.co.me/index.php?option=com_content&view=category&layout=blog&id=45&Itemid=96&lang=sr

SECTION O — PUBLIC ADMINISTRATION AND DEFENCE; COMPULSORY SOCIAL SECURITY

84			Public administration and defence; compulsory social security	
	84.1		Administration of the State and the economic and social policy of the community	
		84.11	General public administration activities	8411
		84.12	Regulation of the activities of providing health care, education, cultural services and other social services, excluding social security	8412
		84.13	Regulation of and contribution to more efficient operation of businesses	8413
	84.2		Provision of services to the community as a whole	
		84.21	Foreign affairs	8421
		84.22	Defence activities	8422
		84.23	Justice and judicial activities	8423*
		84.24	Public order and safety activities	8423*
		84.25	Fire service activities	8423*
	84.3		Compulsorysocial security activities	
		84.30	Compulsorysocial security activities	8430

FIGURE 1: NACE CLASSIFICATION – PUBLIC ADMINISTRATION

Up to our knowledge, the Eurostat NACE classification presented in Figure 1, could be a promising candidate for EAGLE OER-LP, as it provides classification of activities which might relate to most job profiles and related work activities in the field of public administration. For example, NACE classifies public administration into the following sub-activities [18]:

- executive and legislative administration of central, regional and local bodies,
- administration and supervision of fiscal affairs,
- operation of taxation schemes,
- duty/tax collection on goods and tax violation investigation,
- customs administration,
- budget implementation and management of public funds and public debt,
- raising and receiving of money and control of their disbursement,
- administration of overall (civil) research and development policy and associated funds,
- administration and operation of overall economic and social planning and statistical services at the various levels of government.

However, in the EAGLE project, the final level of granularity still needs to be discussed with end users to provide useful guidance. Regards country specific classifications, we note that in many cases there exist classifications or catalogues designed for specific purposes such as course catalogues. In EAGLE, it will be useful to extend and map the initial classification. Regarding multilingualism in EAGLE, classifications have to be translated and mapped in a way to be useful for all participating countries and their respective languages. Furthermore, different classifications might be used in different countries, so it could be also possible to have different classifications mapped.

2.2.2 Summary

Finally, we recommend using an RDF representation of LOM. We also recommend to allow multiple, country-specific classifications which is the key issue to support searching and

browsing. As a starting point, we have presented only an activity-oriented classification. We expect our approach to be piloted, validated and extended/ refined over the project.

2.3 Review of Learning Platforms

2.3.1 Learning Management Systems (LMS)

Learning Management Systems refer to a range of technical infrastructure and applications used for educational and professional training means [20, 21]. There is no standard definition what LMS are and what functionalities are subsumed [20, 21]. Still there tends to be agreement that LMS are more than content management systems, which provide access to learning documents. They act more like an overall (technical) framework to manage the organization and conduct of courses within an educational or professional context [21], [20].

The main functionalities of LMS include computer based instructions, learning processes and content management. Apart from that LMS enable the registering, tracking and testing of students, interaction between learning groups, including the authoring and exchange of learning content [21] (p.27), [20] (further overview [20] (p.33)).

LMS and its applications can develop and accumulate within a given technical and organizational structure, and are increasingly available on a free and open source basis [21], [20]. One beneficial feature of open LMS can be as follows: the potential adaptation of systems and platforms without great costs thought the systems may be less evaluated and ‘robust’ [21] (p.28).

One important concept to add to the functionality of open LMS is Open Educational Resources (OER). OER is a type of learning objects that can be defined as “any digital object which can be freely accessed and used for educational purposes”, [22] (p.23). Examples of open platforms are Open Scout, WebCt and Moodle [21].

2.3.2 Learning Management Systems (LMS) in Public Administration

One of the main features of LMS in public service is that it provides civil servants with the tools and services to acquire visual and audio learning materials in a variety of multimedia formats. Technical functions of the LMS range from editing, evaluation and sharing contents in groups to avail of and create linked online resources [23, 24]. An overview of the LMS features and technical functions is available presented in [25, 26].

Although there is no conceptual discussion so far about LMS in the context of public administrations (as in [20]), open platforms and LMS like Moodle can be used and adapted for the context of public administrations [23, 24]. However, openness in context of administrations tends to focus mainly on the reuse of contents and deployment of open source platforms. Less often studies refer to the concept of OER, the free, open sharing and collaboration regarding digital resources. Learning resources seem to be less shared (and accessible) beyond administrative boundaries or even outside the sector [27, 28].

The findings in the literature can be refined based on the barrier study in the EAGLE project (see D2.2.). For example, the interest in having access to multimedia formats can be supported. Despite that only few participants knew about online learning platforms and OER,

a comprehensible list of required functions was developed for this work package. Among others, these functionalities are to provide:

- Applications for information sharing in form of updates and notifications,
- Collaboration and chat functions within groups, and among selected users,
- The functionality to have own profiles and learning space,
- Applications for content development, rating, including recommender systems,
- Feedback-, activity- and tracking mechanisms (for performance evaluation),
- Role based access restriction (technical and policy wise),
- Guidance and functions that suffice established security standards,
- The integration of the platform with available training offers, i.e. organizational practices, systems, resources and rules within the administration.

A more detailed overview can be found in the D2.2 (chapter 7).

Several points will be approached in the EAGLE project with the development of unique applications and change management processes. Two points to outline in this deliverable is the demand to learn based on work-place relevant contents as well as the specific requirement of both managers (in work-life-context) and employees (learners in platform-context), to trace how time was spent and what has been learned.

2.4 Review of Open Data Platforms & Tools

The Open Government Data initiative makes the government data available to the public for accomplishing transparent and accountable governance. This allows the third parties to fulfill public requirements through the development of applications, which leverages the potential of the open government data. Several nations have incorporated this strategy by distributing the open data through their Open Governmental Portal.

In this section, we summarize the various open-data tools and technologies available.

2.4.1 GovData.DE Open Data Platform - Data Management Module

The GovData.DE [29] portal launched by the German Federal Ministry of the Interior in February 2013 is considered the one of the successful application of the open-data initiatives. The design, development and deployment of the open data platform have been taken care by Fraunhofer Institute.

This portal provides the study of government data, which includes processes such as the extraction of data from various data providers, the transformation of the data and harvest to the CKAN so that it can be used further. Comprehensive Knowledge Archive Network (CKAN) [30] is considered to be the de-facto standard for culturing open data sets and the backend database of the portal. The harvesting mechanism was implemented by using simple Python or CKAN extension, which is described in Figure 2 [31]. CKAN extensions are tightly coupled and prone to adapt other database vendors. Since data grows exponentially, CKAN can be replaced by big data vendors. The inability to produce modular and scalable scope for extending the mechanism to deal with the exponential growth of data led to the

upgrade the harvester to a Java based OSGI [32, 33] framework. This allows the distributed processing of large data sets using big data vendors.

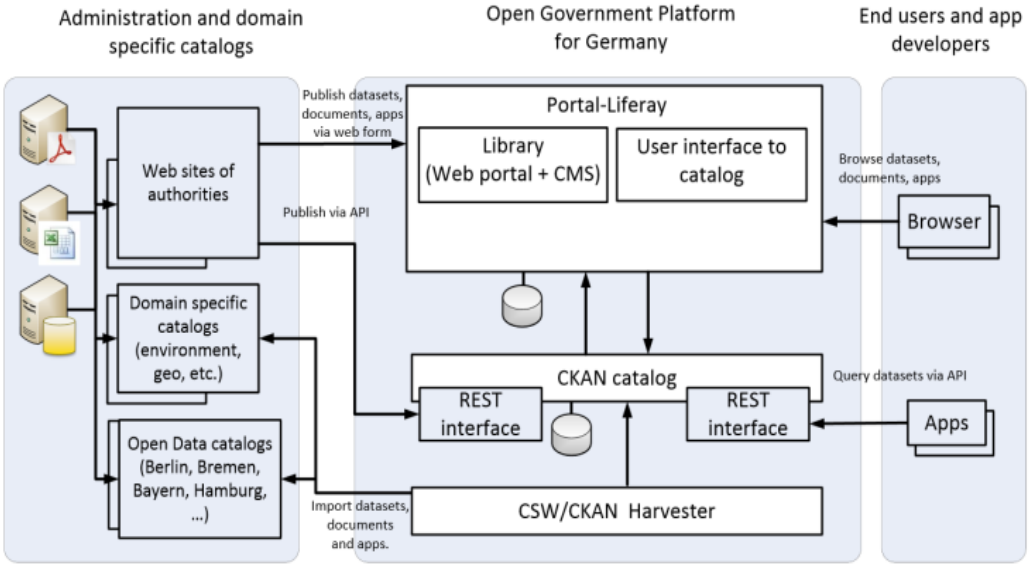

FIGURE 2 : OVERALL ARCHITECTURE OF THE OPEN-DATA PLATFORM ESTABLISHED FOR GOVDATA.DE – CKAN EXTENSION PYTHON BASED

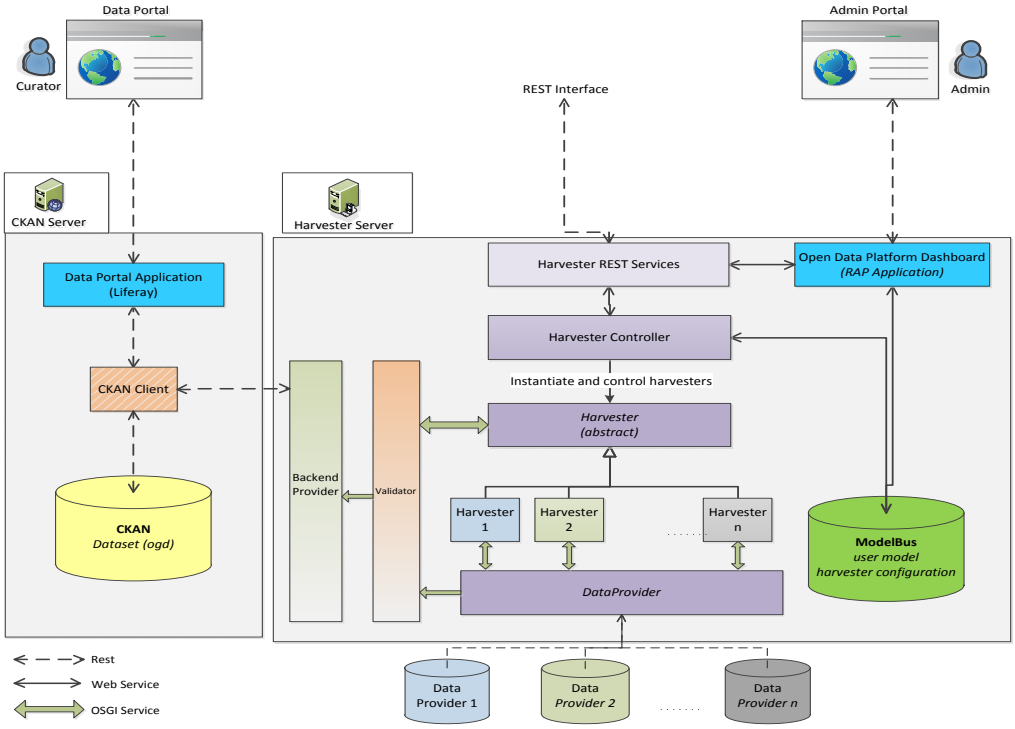

FIGURE 3 : OVERALL ARCHITECTURE OF THE OPEN-DATA PLATFORM ESTABLISHED FOR GOVDATA.DE – MODEL-DRIVEN, JAVA-OSGI BASED

Figure 3 [34] illustrates the architecture of the open data platform developed for GovData.de, which is robust in nature and implements the model-driven OSGI framework in Java (see Section 3.1.7).

The semantics and knowledge of open data sets are well maintained by the Model-driven Engineering (MDE) [35]. This is achieved by validating datasets in terms of models with their corresponding meta-models in Eclipse Modelling Framework (EMF) [36]. Furthermore, the ability of incorporating multiple data providers and transformation of one data provider to another taken care by transformation rules defined at the meta-model level using MDE, which is explained in Figure 4 and Figure 5 [34].

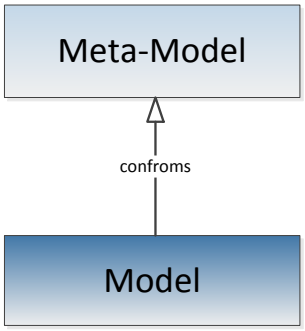

FIGURE 4: RELATIONSHIP BETWEEN A META-MODEL AND MODEL

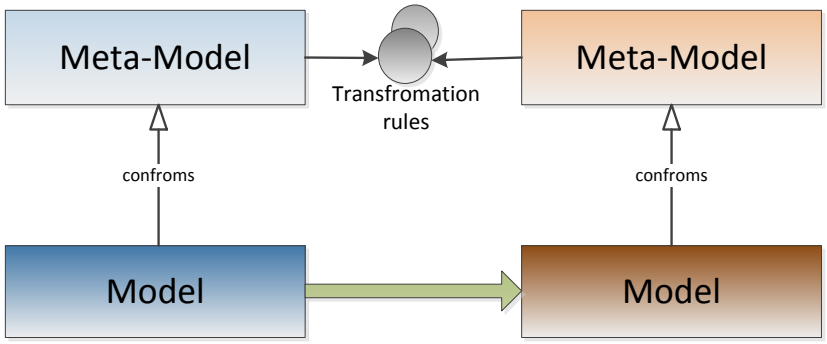

FIGURE 5: TRANSFORMATION OF SOURCE MODEL TO TARGET MODEL USING MDE

Thus, the usage of OSGi along with MDE lead this platform to a robust, flexible and configurable open data portal by upgrading the harvester logic through migrating it from CKAN or Python extension to a modularized system.

2.4.2 TAO

TAO is an Open Source project that provides a solution for Computer-based Assessment. It was developed in a joint effort by the University of Luxembourg and CRP Henri Tudor and is now housed by OAT, a spin-off company that monetizes the result of the previous research. In the context of EAGLE, TAO will be used to provide the functionalities of test and item generation, test and item repository, and delivery.

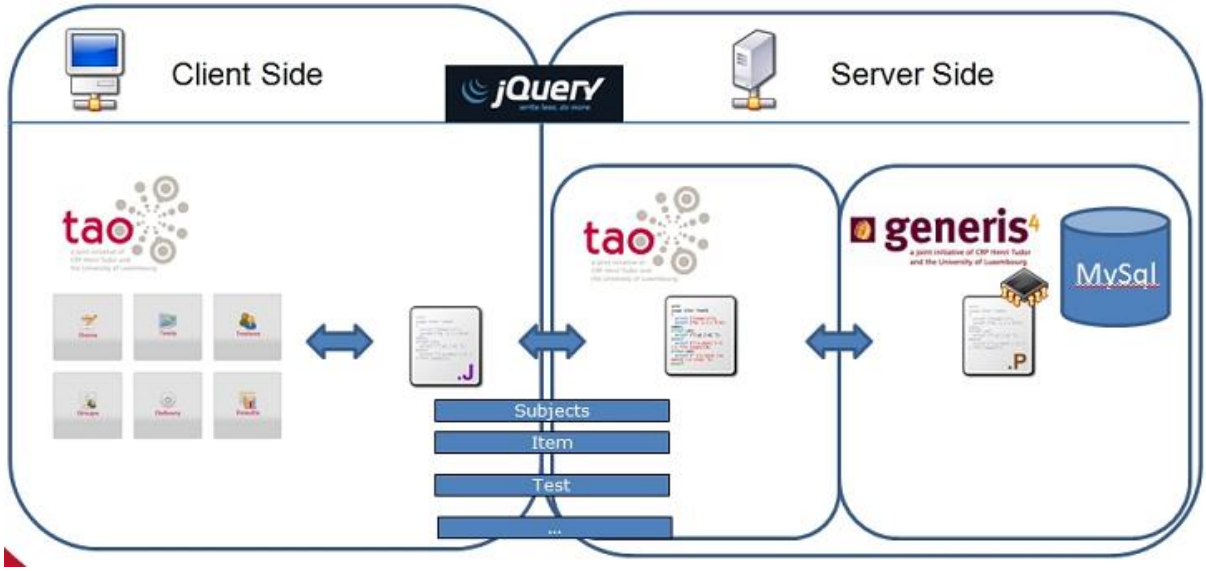

FIGURE 6: TAO ARCHITECTURE OVERVIEW

TAO implements existing industry standards such as QTI and is able to provide tests to any LTI compatible platform, such as Moodle, or any web browser. As TAO aims to satisfy the needs of novices as well as experts, its facilities will allow for users to compose their own test items, which in turn can be arranged as tests by experts or stakeholders.

2.4.3 Semantic Web Technologies Stack

In 2001, Tim Berners-Lee described the so-called 'Semantic Web Layer Cake' (see Figure 7) as a technology stack for his vision of a Web of Data ([37, 38]). The distinct layers represent classes of different abstraction, which together allow the development of the Semantic Web.

Anything that might be used or referred to in the Web of Data is called a resource. The foundation of this stack is the Unique Resource Identifier (URI), which ensures the uniqueness of informational and non-informational resources, which represent either content or informational resources as documents (like JPG files, Videos) or things in the real world (e.g., a person, locations), often referred to as non-informational resources.

The Resource Description Framework (RDF[8]), the RDF Vocabulary Definition Language (RDFS[9]) and the Web Ontology Language (OWL[10]) define a model for describing resources, documents and relations in between. RDF encodes data in the form of subject, predicate and object triples. Any resource is expressed as a set of triples consisting of a subject (the resource itself), the predicate (the meaning of the data) and the object (the data itself), which might be a literal or an URI pointing to another resource.

[8] http://www.w3.org/RDF/
[9] http://www.w3.org/TR/rdf-schema/
[10] http://www.w3.org/TR/owl-features/

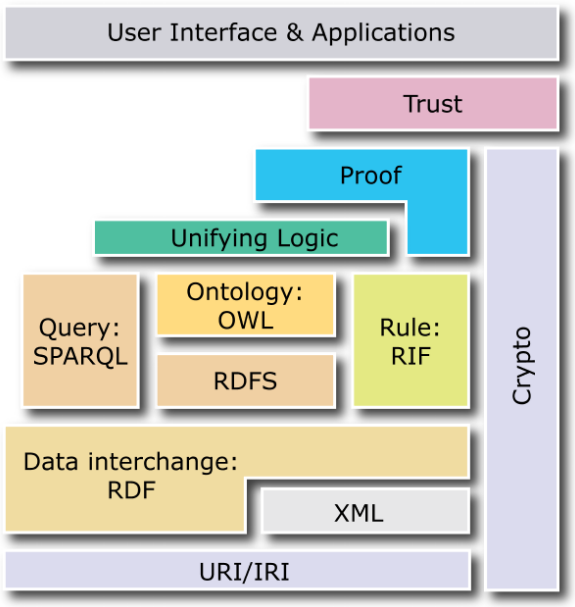

FIGURE 7: SEMANTIC WEB LAYER CAKE (W3C)

When creating links between resources, both subject and object of a triple are URI's that each identifies a resource. The predicate specifies how the subject and object are related, and is also represented by a URI. To create triples which express common meaning it is required to create and use vocabularies to interlink with, to use commonly accepted predicates, specified with RDFS, for the data in order to allow re-use of the data. This allows querying the data by using the SPARQL Query Language for RDF (SPARQL[11]).

However, it is not sufficient to use meaningful predicates when creating triples. To foster the creation of the Web of Data, Tim Berners-Lee outlined best practice guidelines for creation and usage of data on the web [39]:

1. Use URI's as names for things.
2. Use HTTP URI's so that it is possible to lookup those things on the web.
3. When a resource is requested, provide useful information using the standards (HTTP, RDF)
4. Include links to other resources so that people can discover more things.

By considering these principles, it is possible to open formerly closed data repositories to the web and to provide the data in a well-defined, well-structured way. Thus, the formerly closed data becomes readable by humans and interpretable by machines for further use. This vision led to the Linking Open Data (LOD[12]) project where the publication as well as the consumption of Linked Data is explained but also an overview of the available datasets is provided. According to the above-mentioned principles, it is obvious that the LD servers provide its resources by means of REST services.

[11] http://www.w3.org/TR/sparql11-overview/
[12] http://linkeddata.org/

2.4.4 Representational State Transfer (REST)

Representational State Transfer (REST) is an architectural style for network-based software architectures originally introduced by Roy T. Fielding in 2000 [40]. It specifies how resources may be addressed on the web, without the need for the server to hold a state of its resources per client.

Given that self-explaining URIs identifies resources, they can be easily accessed if they provide accompanying RESTful web services. In addition to data access, RESTful web services allows the full range of CRUD operations[13], which are usually mapped to the existing using HTTP methods:

- GET for retrieving resources,
- PUT for updating resources,
- POST for creating resources, and
- DELETE for deleting resources.

The data itself can be represented in different formats, which are usually JSON or XML (or both). The generic data formats enable communication between heterogeneous service implementation (JAVA, .NET, Perl, Python, etc.), which will enable EAGLE to use a wide range of already existing services and tools.

2.4.5 Linked Data

The vision of Linked Data [39] is to provide a uniform access infrastructure for data on a global scale. The Linked Data approach aims at making data available on the Web not only for consumption by humans, but also for machines. This is accomplished by using standardized W3C formats (e.g. RDF, OWL) and access mechanisms, enabling developers of applications to re-use data easily and in a unified manner. One part of this goal is to connect different sources that provide similar data, forming a global graph that can be traversed by clients in order to discover new information. This global data graph is also denoted as the Web of Data (http://www.w3.org/standards/semanticweb/data) and described as triples according to the RDF specification (http://www.w3.org/RDF/), which is a W3C standard model for data interchange on the Web.

To allow the seamless integration of datasets in the Web of Data regarding to data provenance, trust and licensing, there are standards and recommendations like the provenance ontology (PROV-O) [41] or the Creative Commons (CC) licenses schema [42] to express and share license information in a standardized way. Linked Data technologies, open ontologies and the mentioned provenance and license model build a suitable backbone for upcoming data markets and data hubs as envisaged by EAGLE.

In the past eight years, several open and commercial solutions and toolsets were developed to support the entire Linked Data value chain [43] along the life cycle [44], covering single or combined stages from creation, interlinking, enrichment, quality analysis, repair, exploration, extraction and querying. The FP7 integrated project LOD2 provided a stack

[13] CRUD operations are create, retrieve, update and delete.

(http://stack.lod2.eu/blog/), comprising a number of tools for managing the life-cycle of Linked Data. To highlight just a few, we point to tools for building structured (linked) data sets manually. The most famous is Protégé (http://protege.stanford.edu/), a complex system for ontology experts. More lightweight approaches are e.g. Poolparty (http://www.poolparty.biz) or skosjs (http://www.w3.org/2001/sw/wiki/SKOSjs), both restricted to the creation and maintenance of SKOS vocabularies. To create Linked Data conforming data sets out of existing proprietary data – as is the case in some of the dataset used by EAGLE - in a (semi-) automatic way, tools like Open Refine with RDF extension (http://refine.deri.ie) and The Data Tank (http://thedatatank.com) are helpful. The publication of data is supported by existing tools for the publication of RDF as Linked Data such as Pubby (http://www4.wiwiss.fu-berlin.de/pubby), Paget (http://code.google.com/p/paget), or tools that allow generating RDF from relational data such as D2RQ (http://www4.wiwiss.fu-berlin.de/bizer/d2rq), Triplify (http://triplify.org/Overview), OpenLink Virtuoso (http://virtuoso.openlinksw.com) and BigData (http://www.bigdata.com/). A young member of the Linked Data platform family is the Open Source community project Apache Marmotta (http://marmotta.apache.org) that will be used as the Linked Data component in the EAGLE project.

In parallel to development of the standards and toolsets a tremendous amount of datasets has been published openly and since the initiation of the Linking Open Data (LOD) community project[14] in 2007, the amount of data published according to the Linked Data principles is steadily growing. The primary sources and directories open datasets are CKAN and datahub.io: CKAN (http://ckan.org) is the world's leading open-source data portal platform that makes data accessible by providing tools to streamline publishing, sharing, finding and using data. CKAN is aimed at data publishers (national and regional governments, companies and organizations) wanting to make their data open and available. Within EAGLE, CKAN will be used for providing an OER registry. datahub.io (http://datahub.io) is the free, powerful data management platform from the Open Knowledge Foundation (https://okfn.org). Additionally many datasets are registered in directories like the European Union Open Data Portal (https://open-data.europa.eu), and the open data directories of the national governments (e.g. http://www.data.gov/ in the U.S., http://data.gov.uk/ in the U.K.).

2.4.6 Linked Data Platform – Apache Marmotta

In 2009, Tim Berners-Lee started discussing the extension of the Linked Data principles that would make the Web of Data writeable: In his article he concluded that "the world of Linked Data can be extended to a world of read-write linked data easily" (http://www.w3.org/DesignIssues/ReadWriteLinkedData.html). Subsequently Tom Heath and Christian Bizer talked about evolving the web into a global data space using read/write Linked Data [45]. This idea finally resulted in the foundation of the Linked Data Platform (LDP) Working Group (http://www.w3.org/2012/ldp/) of the World Wide Web Consortium (W3C LDP WG) in 2012, whose objective was to produce a W3C Recommendation for HTTP-based (RESTful) application integration patterns using read/write Linked Data. In 2014, the version 1.0 of the functional specification of a Linked Data Platform was released.

[14] http://esw.w3.org/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

Already in 2010, Salzburg Research (SRFG) started to work on similar read/write concepts for a Linked Data server with additional extensions to handle media resources according to the Linked Data principles: In the Competence Centre for Excellent Technologies "Salzburg NewMediaLab – The Next Generation" (SNML-TNG, http://www.newmedialab.at/) SRFG lead a consortium of leading representatives of the Austrian media industry (Red Bull Media House, Austrian Broadcasting Corporation – ORF, derStandard.at, Salzburger Nachrichten). In this research centre five use cases were implemented: The Open Source "Linked Media Framework" (LMF, http://code.google.com/p/lmf) served as a read/write Linked Data server with strong support of the search mechanisms requirements in enterprise information integration scenarios.

When the W3C Linked Data Platform Working Group started their work it soon became obvious, that the core of the Linked Media Framework fulfilled all the requirements of the upcoming specification of a Linked Data Platform. This gave birth to the incubation of Apache Marmotta (in December 2012) by setting aside LMF's Read-Write Linked Data server code and some related libraries. In November 2013, finally Apache Marmotta graduated as a Top Level Project. The main contributors initially came from the project group at SRFG, but meanwhile got strong support by an international team of contributors. Apache Marmotta consequently is listed as a reference implementation of the LPD Working Group's specification (http://www.w3.org/wiki/LDP_Implementations).

The goal of Apache Marmotta is to provide an open implementation of a Linked Data Platform that can be used, extended and deployed easily by organizations that want to publish Linked Data or build custom applications on Linked Data. As a Top Level Project the software has to meet the strict policies and guidelines of the Apache Software Foundation regarding the publication of releases and the use of Apache infrastructure for the code base.

The Apache Marmotta Platform (http://marmotta.apache.org) is implemented as a light-weight Service-Oriented Architecture (SOA), and as illustrated in Figure 8, consists of a collection of modules (the modules are described in detail on the Apache Marmotta web-site). Each module implements several layers:

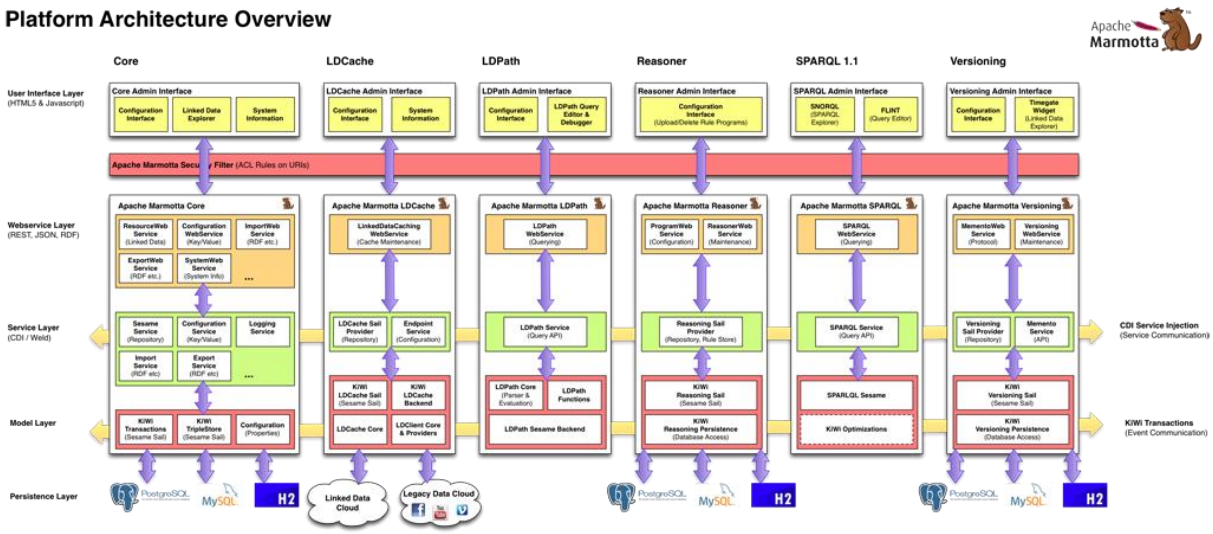

FIGURE 8: APACHE MARMOTTA PLATFORM ARCHITECTURE

- The **user interface layer** is always implemented in HTML5 and JavaScript and accesses the server via REST web-service calls; since Marmotta is a server application, the user interface mostly consists of admin and development interfaces and is not intended for end users.
- The **web-service layer** offers REST web-services to access most of the server functionality; the REST web-services typically consume and produce JSON and/or different RDF formats.
- The **service layer** offers CDI services inside a Java environment that can be called directly from Java.
- The **model layer** offers persistence and data access functionality.
- The **persistence layer** is outside the Apache Marmotta Platform. Marmotta can use a number of free and commercial database systems for persistence (including Open Source systems like PostgreSQL, MySQL and H2).

In EAGLE, Apache Marmotta provides in a modular way the basic functionalities of the Linked Data Platform, such as read/write access to Linked Data, RDF triple store, querying (SPARQL, LDP and LDPath queries), and transparent Linked Data caching. This allows semantically interlinking and accessing internal data sources as well as Linked Data sources on the Web of Data (e.g. geonames.org, DBpedia.org and other valuable datasets). The use of the Linked Data approach in EAGLE supports lightweight integration of datasets that are already available (e.g. geonames.org) or will be made available as Linked Data sources in the future (see e.g. the European Union Open Data Portal (https://open-data.europa.eu/) for available datasets). By using a Linked Data Server, EAGLE will also be able to publish datasets as Linked Data and thus give other applications the possibility to access and link to these datasets.

3 Design Principles and Development Methodology

In the previous chapter, we reviewed the existing tools, technologies and standards relevant to the EAGLE learning platform. Based on the review, and our experiences in software development, we further present the design principles, development methodology and design guidelines that developers need to employ in the creation of the EAGLE OER-LP. In **Section 3.1**, we present the design principles that should be utilized in the design of each component/module of the EAGLE OER-LP. In **Section 3.2**, we describe the software development methodology that would be applied to plan, control and execute the creation of the EAGLE learning platform. Finally, in **Section 3.3**, we have documented the design guidelines that must be adhered during the development of the individual modules.

3.1 Design Principles

In this section, we present the design principles that would be applied in the creation of the various software components of the EAGLE OER-LP.

SOLID principle is the mnemonic acronym, which stands for 5 basic principles, which helps to create stable and scalable software architecture. SOLID stands for:

S – Single Responsibility

O – Open Closed Principle

L – Liskov's Substitution Principle

I – Interface Segregation Principle

D – Dependency Inversion Principle

3.1.1 Single Responsibility (SRP)

This design principle signifies that a particular class or module should be responsible for a particular task. For a large-scale application, it is important to maintain separation in code as it leads to easier maintainability and minimal coupling. Figure 9 depicts the SRP design principle as defining classes for a particular functionality. In the example, the responsibility is segregated upon the nature of work. Hence, two separate classes have been created for creation of shape and creation of text. However, excessive use of SRP can lead to premature optimization instead of a stable design. This causes creation of too many classes and results in a complex architecture.

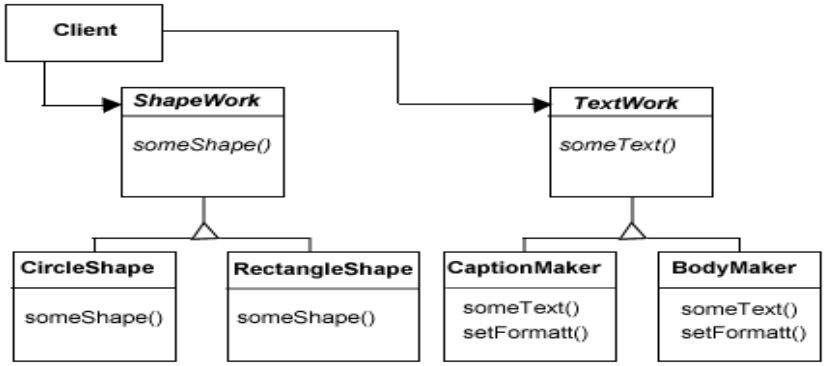

FIGURE 9: SOLID DESIGN PRINCIPLE – SINGLE RESPONSIBILITY (SRP) [46]

3.1.2 Open Closed Principle (OCP)

OCP emphasizes on extension rather than modification of the existing application or class [[47]. Entities or Beans or Models should be open for extension but closed for modification. This can be achieved by using Inheritance and Abstractions having base classes with override-able functions. This helps us to create different derived class which does things differently without changing the base functionality. As illustrated in Figure 9, SRP can be achieved by segregating the different task and assigning a particular class for the particular type of work. Applying OCP, different shapes can be extended from the ShapeWork. This can be achieved by making ShapeWork as abstract class and override the someShape method according to the need. In this regard, the classes are getting extended, which decreases the degree of complexity.

3.1.3 Liskov's Substitution Principle (LSP)

This design principle says that a derived class or type should behave in a fashion similar to their base types. That means if an instance of the base class is substituted by a derived class method, it should not interrupt the flow of the process. Basically using inheritance, functions that use references to base classes must be able to use objects of derived classes without knowing it. This verifies the inheritance is applied correctly in the subsequent base classes.

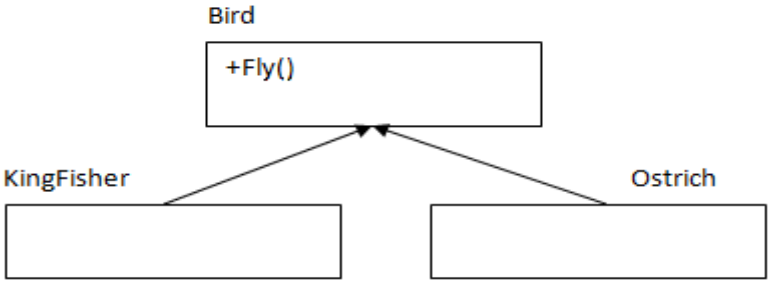

FIGURE 10: SOLID DESIGN PRINCIPLE – LISKOV'S SUBSTITUTION PRINCIPLE (LSP) [48]

If LSP is not correctly implemented, then unit tests for the base classes would never succeed for the subclasses, which in return says whether the inheritance is done correctly or not. For instance Figure 10 shows the pitfalls in of Inheritance while designing the categories

of bird. It's very problematic to inherit Kingfisher and Ostrich from the base class Bird having fly method as its significant commonalities. Hence, this design in the figure proves to be faulty in the behavioral aspects of ostrich. Therefore, to avoid strange behavior as well as chaotic class hierarchies, it is important to structure the inheritance of class with LSP.

3.1.4 Interface Segregation Principle (ISP)

As described in [49], *Clients* should depend upon the specific interfaces related to a particular task instead of a general purpose interface which contains tasks inapplicable to the client [49]. This means we need to define different interfaces for different purpose, which helps in achieving Single responsibility of an object along with flexibility and understandable architecture. This leads to Minimal coupling and reusability and easy unit testing. For instance in Figure 10 Ostrich has to implement fly method, even though it cannot fly. However in Figure 11 fly method is separated from the interface bird by applying ISP. Therefore, the design in which ostrich extends IBird and kingfisher extends IBird and IFlyingBird, satisfies Interface Segregation Principle.

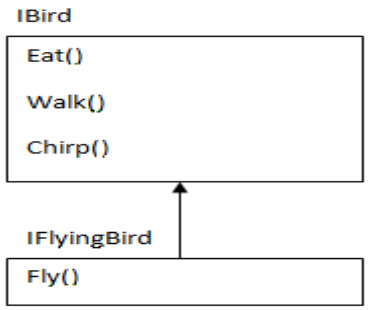

FIGURE 11: SOLID DESIGN PRINCIPLE – INTERFACE SEGREGATION PRINCIPLE (ISP) [48]

3.1.5 Dependency Inversion Principle (DSP)

The prime goal is to decouple the concrete logic from the high level components, which depends upon the abstraction [49]. DSP consists of all the principles discussed above. In simple words, this emphasizes on designing the modules as a pluggable nature not as a single unit. These pluggable components interact or assemble through abstraction which leads to a complete object or module. In Figure 12, according to DIP, the high level module *Car* is not dependent on the engine or wheels but on the abstraction (interfaces) of engine and wheels, by which the car will be resilient to the occurrence of change. Since, the interfaces and their details are isolated to each other; this is much easier to maintain [49]. This leads to an application, which will be cohesive in nature and low in coupling. If DIP is not properly adhered, the cost of maintaining and coupling of software components would be high.

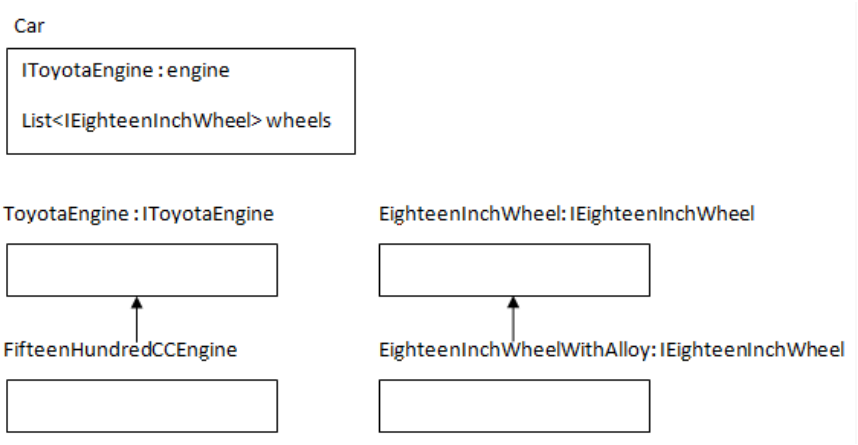

FIGURE 12: SOLID DESIGN PRINCIPLE – DEPENDENCY INVERSION PRINCIPLE (DSP) [48]

3.1.6 Reusability

Reusability means being able to create a new component that uses the feature of existing component without recording those features. This component can be object, class or modules. This can be achieved by inheritance, containment or delegation, aggregation as well as component reuse. Reusability decreases the overhead of designing and developing time and cost of the component

3.1.7 Modularity

Modularity is about segregating a large system and focusing on a particular aspect into some smaller modules, which makes the system easier to understand. This results into modules, which are highly extensible and scalable in nature. In Java platform, the physical design to achieve modularity is done by Java Jar files [50]. Similarly in .NET environment, that can be achieved by dynamic linked libraries (DLL) [51].

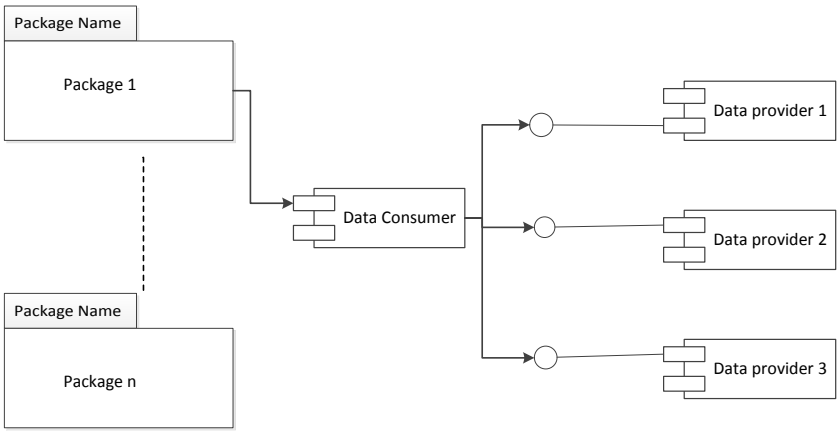

FIGURE 13: USAGE OF MODULARITY

For example in Figure 13 modularity has been described by an application, which contains different packages. The whole system consists of different modules and is communicating within themselves through the services. For instance, the data consumer component of *Package 1* uses the data provider plugins. These provider plugins are connected through some contracts with the component. Therefore, being a modular system it is resilient to

adopt new changes (in this context the data providers) as well as allows modules to hide their implementation details.

3.1.8 Abstraction

Basically, the selection of significant attributes is regarded as Abstraction. This describes regarding the building of an entity with all the necessary aspects included whilst ignoring all the trivial differences. This entity remains as representation of the original real object. For example, in the Figure 14, the classification of human has been done into two categories; i.e.; man and woman. This has been done taking significant commonalities into account and ignoring some trivial differences. Hence, creation of different types of men can be executed by extending Abstract class "*Man*".

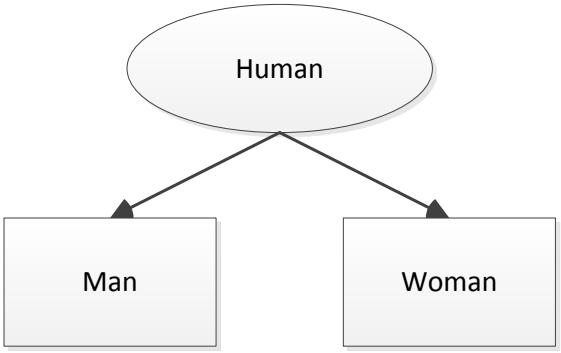

FIGURE 14: USAGE OF ABSTRACTION

3.1.9 Interoperability

Interoperability describes as the ability of systems to work together irrespective of the way they are being developed. For instance the nature of a web service consumer and a provider is a good example for interoperability as the provider and consumer may be developed with different technologies, but are able to operate with each other. In web services, a consumer can consume the service of the provider through the common *Web Service Description Language* (WSDL) [52].

For example Figure 15 illustrates, irrespective of the production platform, an application should be able access the API of different service providers upon some common WSDL. The service gateway, using WSDL and SOAP messages, consumes the API. This gives a scope of linking multiple systems to share information in a dynamic fashion. A large complex application can also be easily manageable provided it incorporates interoperability.

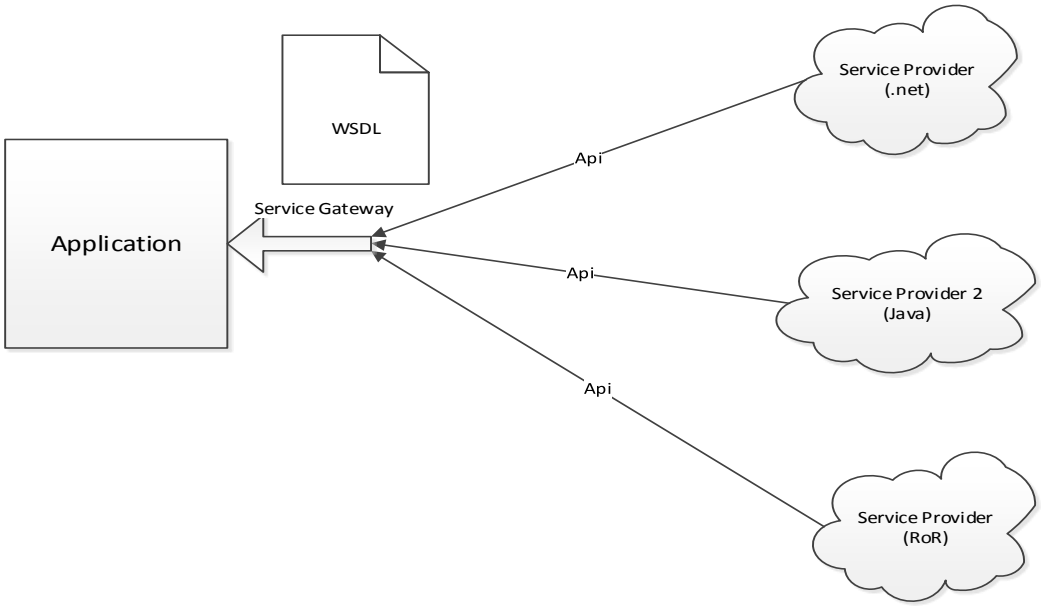

FIGURE 15: USAGE OF INTEROPERABILITY

3.1.10 Accessibility

This means defining proper scope of the component within and between components. This helps us in achieving security related to business functionalities.

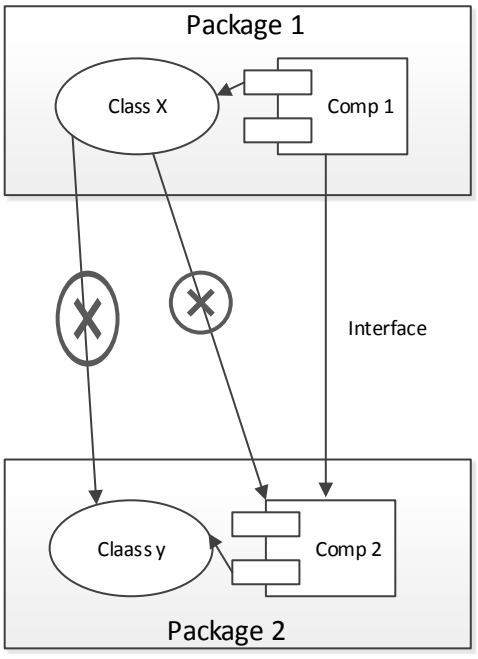

FIGURE 16: USAGE OF ACCESSIBILITY

Like in the Figure 16 the accessibility of the classes has been defined. In this scenario, Class X, is used by component 1, should not be exposed to component 2 or to the class Y beyond its scope, which violates the accessibility rule. This can be solved by the means of interface through which the component comp 1 and comp 2 interact with each other. Likewise, these dependencies must be resolved according to the accessibility rules.

3.1.11 Open Source

Generally, Open source deals with developing products through public shared information and also focuses on the availability of the product publicly. The author of open source programs share the source code to the public users, who would like to alter, use, copy as well as learn it. A user who uses or alter the open source code for their purposes must also share that publicly. Eventually it considered as more secure and stable than proprietary software.

3.2 Development Methodology

The EAGLE software development team is distributed among several project partners, each of them with a different background and experience in different technologies. We base our software development on established methods used in existing open source projects, such as Apache Marmotta[15]. We follow an agile software development process, having short iteration cycles and small modules, each of them having its own planning, requirements analysis, design, coding, testing, and documentation phase.

3.2.1 Source Code Repository, Versioning and Branching

All EAGLE sources and related resources (e.g. i18n) must be stored in a common source code repository. For this purpose, TUDOR provides a “git” source code repository. Milestones and major releases will be labeled (tagged) in the Source Code Management (SCM) system. Each component is versioned independently.

As shown in Figure 17, EAGLE will use a *Gitflow Workflow*[16] with the following conventions:

- Instead of a single master branch, we are using two branches to record the history of the project: the master branch stores the official release history, and the develop branch serves as an integration branch for features. The whole project development works around the distinction between these two branches.
- New features could optionally open feature branches for some topics/issues, which are typically identified with the issue key from our issue tracker (EAGLE-XYZ).
- In addition, hotfix branches (e.g. hotfix-3.0.x) are also used to quickly patch production releases without interrupting the current development cycle.

[15] http://marmotta.apache.org/development.html
[16] http://www.atlassian.com/git/workflows#!workflow-gitflow

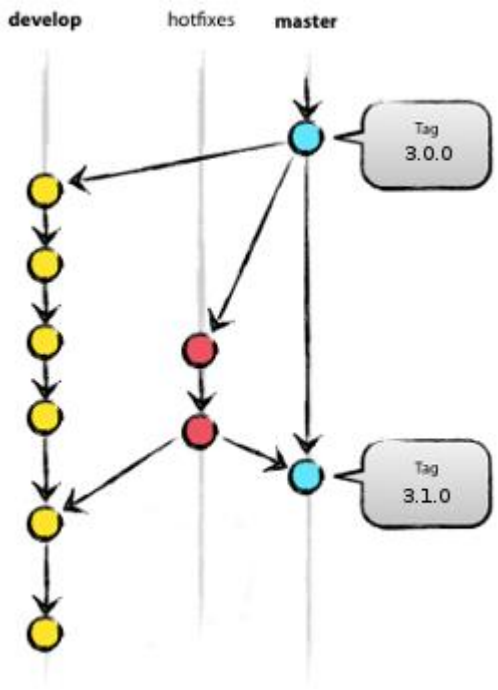

FIGURE 17: GITFLOW WORKFLOW

For the final prototype at the end of the project runtime, EAGLE version number 1.x is targeted. During the project runtime, interim MILESTONEs will be defined and tagged as required. Before the final release, there will be release candidates available for evaluation.

3.2.2 Build Environment

The EAGLE learning platform must be deployable on a single machine but it must be also able to be deployed on several machines interconnected via network.

During development a continuous integration service such as Jenkins is used to check the systems integrity whenever new code is committed. Therefore, the build and deployment process may not depend on any IDE such as Eclipse, Netbeans or IntelliJ. Using the Maven software project management and comprehension tool, and following its directory layout, will ensure compatibility with the majority of IDEs as well as CI servers. Therefore, all components should follow a predefined structure to be automatically built with Maven.

3.2.3 Issue Tracking

For issue tracking the built-in issue-tracking system from GitLab, as provided by TUDOR is used. This will be the exclusive place for bug reports, feature requests. When addressing issues, the commit message will link to the issues solved, i.e. there should be no commit without a corresponding ticket. The following tables describe the issue types, priorities and the workflow and issue can take.

When creating a new issue, select the issue type based as follows:

Issue type	Description
Bug	Bug reports are used for cases where EAGLE fails not function as it should (as defined by some documentation). If you are not certain whether the issue you've found is actually a bug, please ask at the mailing list first for help.
New Feature	Use a feature request when EAGLE does not have some functionality you need.
Improvement	Use an improvement request to suggest improvements to existing features. Typical improvement requests are about updating documentation, increasing stability and performance, simplifying the implementation, or other such changes that make EAGLE better without introducing new features or fixing existing bugs.
Test	Use this type when contributing test cases for existing features. Normally test cases should be contributed as a part of the original feature request or as regression tests associated with bug reports, but sometimes you just want to extend test coverage by introducing new test cases. This issue type is for such cases.
Task	Used only for issues related to project infrastructure.

TABLE 1 ISSUE TYPES

Issue priority should be set according to the following:

Issue priority	Description
Blocker	Legal or other fundamental issue that makes it impossible to release EAGLE code
Critical	Major loss of functionality that affects many EAGLE users
Major	Important issue that should be resolved soon
Minor	Nice to have issues
Trivial	Trivial changes that can be applied whenever someone has extra time

TABLE 2 ISSUE PRIORITIES

EAGLE issues can transition through a number of states while being processed:

State	Description	Next states in workflow
Open	The issue has just been created	*In Pogress*

In Progress	Work has started on the issue	*Resolved, Open*
Resolved	The issue has been resolved from the developers' point of view. Documentation and Testcases have been created and updated as required. Issue is ready for release.	*Reopened, Closed*
Reopened	A resolved issue has been recognized to contain bugs or to be incomplete and thus has been reopened.	*In Progress, Resolved*
Closed	Work on this issue has finished and it is included in the release.	--

TABLE 3 ISSUE WORKFLOW

3.2.4 Conventions

The following conventions are agreed.

- **Naming Conventions:** The Java Naming conventions[17] are used. Directory and file names with whitespaces must be avoided. Path and script names must be case-insensitive.
- **Coding Conventions:** For the development of the distinct components, common coding conventions are applied. A check style configuration file will be provided in the source code repository.
- **Package Structure:** The eagle platform will be built on existing solution. The developed functionality will follow a high-level package structure such as
  - eu.eagle.<component>.api: The definition of external interfaces of this component.
  - eu.eagle.<component>.model: The definition of an arbitrary data model
  - eu.eagle.<component>.impl: The implementation of the specified API
  - eu.eagle.<component>.util: For common purpose classes & functionality
  - eu.eagle.<component>.rest: The web service interface
- **Software Tests:** Tests are located in the same package as the tested class but in the (mavenized) test folder. The name of the test class follows the tested class, but with suffix "Test" of Unit tests, or "IT" for integration tests (which have dependencies on other modules).
- **Source Code Headers:** Each of the committed files of the source code needs a *licensing header*, as well as *author information*. Source code template files will be provided in the Git repository.

[17] http://www.oracle.com/technetwork/java/codeconventions-135099.html

3.2.5 Messaging

Message exchange between the single EAGLE components will be required. With respect to the communication requirements, the EAGLE components should take the following into account:

- Asynchronous message exchanging is required. According to the use cases, both standard asynchronous messaging exchange models are required: direct Point-to-Point component communication and Publish-Subscribe via a central message instance.
- Message routing abilities are also required. According with the use cases and the requirements, the message end point can/may be unknown in the moment when the message is send. In these cases the message end point(s) can be inferred (depending on the message content and meta-data) and routing abilities are required in order that the send message reach the needed end point(s).
- Messaging exchange patterns (e.g. request-reply, call-back, etc.) are also required.

It is presumed that extensive message related handling and operations are required (e.g. enrichment, trimming or message filtering, routing like described by Enterprise Integration Patterns (EIP).

3.2.6 Testability

Each (main) component must be testable according to its contract, i.e. the API specification. The proof of the functionality (tests) can be done in standalone mode or combined with several components involved. For the standalone test mode, all dependencies must be emulated. Tests must be platform independent and must be able to be automatically evaluated prior to deployment (using the Maven Surefire Plugin). Documentation about parameters and eventually pre- and post-conditions are expected.

3.3 Design Guidelines

The design guidelines cater to three major branches of development: architecture, code, and documentation.

3.3.1 Architecture

The architecture will be drafted based on the design principles highlighted above with a focus on all SOLID principles. The implementation of said principles will be realized by the application of well-established patterns [53] while avoiding known anti-patterns. All software stakeholders will be involved in the design architecture design process and peer reviews will be used to keep the quality of the final architecture high. All aspects in the architecture that are formal, thus not including any mock-ups, are to be written using the UML 2.0 (or later) standard.

3.3.2 Code

All application code should relate to the previously drafted architecture. The main guidelines for writing code will be highlighted in a code style document. The main reference for this document will be the well-known Google Java Style[18]. As EAGLE is predominantly based on Java, the style guide will be adapted to produce the EAGLE Code Style document which will reference common conventions. Additional resources such as a shareable style file for common IDE will be elaborated as well.

3.3.3 Testing

To ensure a certain measurable quality of the software, a testing plan will be established which will encompass all EAGLE software components. All tests will be made available to the community as well to entice trust in the software. Furthermore, attempts will be made to automate the testing process to provide historical testing data. To reach this goal, the use of a continuous integration server will be solicited. Code will be kept accessible in a common code repository based on Git. The repository front-end will also provide bug and feature tracking facilities.

3.3.4 Documentation

All design decisions will be documented. This will allow to root any issues that may crop up as well as to clearly establish causality chains. Documentation of the code will be detailed in the EAGLE Code Style document. Each software module of EAGLE must be self-contained regarding documentation and include all interface definitions for outgoing and incoming communications. Interface constraints will also be documented clearly using an approach from Design by Contract. A global EAGLE documentation will be established in the form of a Javadoc as well as a usage guide listing requirements and installation guidelines.

3.3.5 Document formats and tools

The guidelines for the subjects above will rely on some technologies and tools. To ensure a good cooperative environment, the exchange formats are standardized. All documents, with the exception of scientific papers, should be published using the DOCX extension if they are intended to be modified and the PDF format once they are final.

Code will be produced either in Java or PHP using the respective code extension. Documentation of the code is produced using Javadoc or similar and should be outputted using the HTML family.

The code repository will be based on Git with GitLab as a frontend that provides bug and feature tracking.

All UML diagrams should be produced using the UML eXchange Format[19] (UXF) extension which can be read by many tools.

[18] https://google-styleguide.googlecode.com/svn/trunk/javaguide.html
[19] http://en.wikipedia.org/wiki/UXF

4 Conclusion

In this deliverable D5.1, we presented the *design principles*, *guidelines* and the *software development methodology* that we intend to employ in design and realization of the EAGLE Learning Platform (OER-LP). In addition to specifying the software development methodology, we have also reviewed the tools, technologies and standards that we intend to use in and during development of the OER-LP.

This deliverable is written to serve as the go-to guidebook for all developers working on the EAGLE OER-LP. The *design principles*, *guidelines* and the *development methodology* specified in this deliverable are based on our experiences in previous software development projects and tailored to the needs and constraints of the EAGLE project. Though we have documented the methodology that we would employ for the development of the LP, this is by no means final. The methodology is expected to evolve during the lifecycle of the project.

5 References

[1] EAGLE Consortium, “EAGLE - EnhAnced Government LEarning,” Sep. 2013, FP7-ICT-2013-11 /2013.8.2 Technology-enhanced learning; Objective c) Holistic learning solutions for managing, reaching and engaging learners in the public administrations, Grant agreement no: 619347. [Online]. Available: http://-www.eagle-learning.eu/

[2] OpenScout, “Skill based scouting of open user-generated and community-improved content for management education and training,” 2009 - 2012, Grant ECP 2008 EDU 428016. [Online]. Available: http://openscout.net/

[3] K. Niemann, U. Schwertel, M. Kalz, A. Mikroyannidis, M. Fisichella, M. Friedrich, M. Dicerto, K.-H. Ha, P. Holtkamp, R. Kawase, E. Parodi, J. Pawlowski, H. Pirkkalainen, V. Pitsilis, A. Vidalis, M. Wolpers, and V. Zimmermann, “Skill-Based Scouting of Open Management Content,” in *Sustaining TEL: From Innovation to Learning and Practice*, ser. Lecture Notes in Computer Science, M. Wolpers, P. Kirschner, M. Scheffel, S. Lindstaedt, and V. Dimitrova, Eds. Springer Berlin Heidelberg, 2010, vol. 6383, pp. 632–637. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-16020-2_66

[4] R. Kawase, M. Fisichella, F. Deng, M. Friedrich, K. Niemann, D. Boerner, P. Holtkamp, K. Hun-Ha, K. Maxwell, E. Parodi, J. Pawlowski, H. Pirkkalainen, C. Rodrigo, U. Schwertel, P. Vassilis, and A. Vidalis, “D1.1 Analysis Report on Federated Infrastructure and Application Profile,” OpenScout Consortium, Tech. Rep., Apr. 2011. [Online]. Available: http://www.openscout.net/filesrep/category/11-wp-1-deliverables?download=17%3ADeliverables/d1-1-openscout-federated-infrastructure-v2.pdf

[5] R. Kawase, M. Fisichella, K. Niemann, M. Friedrich, A. Vidalis, V. Pitsilis, H. Pirkkalainen, P. Holtkamp, K. Maxwell, E. Øverby, U. Schwertel, A. Mikroyannidis, M. Kalz, and M. Dicerto, “D1.2.1 Initial Federated Infrastructure V1.0,” OpenScout Consortium, Tech. Rep., Nov. 2010. [Online]. Available: http://-www.openscout.net/phocadownload/d1-2-1-initial-federated-infrastructure.pdf

[6] *1484.12.1-2002 - IEEE Standard for Learning Object Metadata*, LOMWG12 - Working Group for Learning Object Metadata Std., 2009, Superseded by: 1484.12.1-2002_Cor_1-2011. [Online]. Available: http://-standards.ieee.org/findstds/standard/1484.12.1-2002.html

[7] M. Nilsson, “From Interoperability to Harmonization in Metadata. Standardization Designing an Evolvable Framework for Metadata Harmonization,” Doctoraat, KTH School of Computer Science and Communication, Stockholm, 2010, ISBN 978-91-7415-800-7. [Online]. Available: http://kmr.nada.kth.se/-papers/SemanticWeb/FromInteropToHarm-MikaelsThesis.pdf

[8] M. Nilsson, M. Palme;r, and J. Brase, “The LOM RDF Binding : Principles and Implementation,” in *Proceedings of the Third Annual ARIADNE conference, Leuven Belgium, 2003*, 2003, qC 20101028.

[9] A. Powell, M. Nilsson, A. Naeve, and P. Johnston, “DCMI Abstract Model,” Tech. Rep., Mar. 2005. [Online]. Available: http://dublincore.org/documents/abstract-model/

[10] *ISO 15836:2009: Information and documentation – The Dublin Core metadata element set*, ISO/TC 46/SC 4 Std., Rev. 2, Jun. 2014, Stage: 90.60. [Online]. Available: http://www.iso.org/iso/iso_catalogue/-catalogue_ics/catalogue_detail_ics.htm?csnumber=52142

[11] *ANSI/NISO Z39.85 - The Dublin Core Metadata Element Set*, Content and Collection Management Topic Committee NISO Standard, Rev. 2, May 2007, ISBN:978-1-880124-73-4. [Online]. Available: http://-www.niso.org/kst/reports/-standards?step=2&gid=&project_key=9b7bffcd2daeca6198b4ee5a848f9beec2f600e5

[12] W3C Recommendation, *Extensible Markup Language (XML)*, W3C Std. 1.0, Rev. 5, Nov. 2008. [Online]. Available: http://www.w3.org/TR/REC-xml/

[13] ——, *RDF 1.1 Concepts and Abstract Syntax*, W3C Std. 1.1, Feb. 2014. [Online]. Available: http://-www.w3.org/TR/rdf11-concepts/

[14] Open Archives Initiative. OAI-PMH - Open Archives Initiative Protocol for Metadata Harvesting. [Online]. Available: http://www.openarchives.org/pmh/

[15] J. M. Pawlowski and D. Kozlov, “Analysis and Validation of Learning Technology Models, Standards and Specifications: The Reference Model Analysis Grid RMAG,” *Int. J. IT Stand. Stand. Res.*, vol. 8, no. 2, pp. 1–19, Jul. 2010. [Online]. Available: http://dx.doi.org/10.4018/jitsr.2010070101

[16] Members of the IFLA Working Group on Guidelines for Subject Access by National Bibliographic Agencies, *Guidelines for Subject Access in National Bibliographies*, ser. IFLA Series on Bibliographic Control, Yvonne Jahns, Deutsche Nationalbibliothek, Ed. Walter de Gruyter, 2012, vol. 45, ISBN 3110281104, 9783110281101. [Online]. Available: http://www.ifla.org/publications/ifla-series-on-bibliographic-control-45

[17] Library of Congress, “Library of Congress Classification Outline: Class J - Political Science.” [Online]. Available: http://www.loc.gov/aba/cataloging/classification/lcco/lcco_j.pdf

[18] Eurostat, “NACE Rev 2: Statistical classification of economic activities in the European Community,” European Commission, Luxembourg, Tech. Rep., 2008, ISBN 978-92-79-04741-1. [Online]. Available: http://epp.eurostat.ec.europa.eu/cache/ITY_OFFPUB/KS-RA-07-015/EN/KS-RA-07-015-EN.PDF

[19] *NAICS 2012 - North American Industry Classification System*, Statistics Canda Std., 2012. [Online]. Available: http://www23.statcan.gc.ca/imdb/-p3VD.pl?Function=getVDStruct&TVD=118464&CVD=118465&CPV=91&CST=01012012&CLV=1&MLV=5

[20] W. R. Watson and S. L. Watson, “EnglishAn Argument for Clarity: What are Learning Management Systems, What are They Not, and What Should They Become?” *EnglishTechTrends*, vol. 51, no. 2, pp. 28–34, 2007. [Online]. Available: http://dx.doi.org/10.1007/s11528-007-0023-y

[21] B. Holmes and J. Gardner, *E-Learning: Concepts and Practice*. SAGE Publications Ltd, 2006, ISBN 978-1-4129-111-5.

[22] H. Pirkkalainen and J. Pawlowski, “Open Educational Resources and Social Software in Global E-learning Settings,” in *Sosiaalinen Verkko-oppiminen*, P. Yli-Luoma, Ed., IMDL, Naantali, 2010, pp. 23–40.

[23] A. Conci and C. Bramati, “E-Learning in Public Administration,” *SLOOP : Sharing Objects in an Open Perspective*, pp. 81–85, 2007, formaper, Agency of Milan Chamber of Commerce.

[24] C. I. Silvestru, R. C. Bere, and L. Nemes, “E-Learning Platform For Public Administration - Case Study,” *Proceedings of the 9th International Scientific Conference "eLearning and Software for Education"*, vol. 1, pp. 486–491, Apr. 2013.

[25] R. C. Bere, C. I. Silvestru, and L. Nemes, “Challenges and Opportunities in Knowledge sharing in e-Learning programs for public administration,” *Proceedings of the 10th International Scientific Conference "eLearning and Software for Education"*, vol. 1, pp. 461–469, Apr. 2014.

[26] I. Hartescu, “A Case Study of Implementing Blended Learning Courses in Public Administration,” *The 8th International Scientific Conference eLearning and Software for Education, Bucharest*, pp. 497–503, Apr. 2012.

[27] L. Stefanick and E. LeSage, “Limitations to developing virtual communities in the public sector: A local government case study,” *Canadian Public Administration*, vol. 48, no. 2, pp. 231–250, 2005. [Online]. Available: http://dx.doi.org/10.1111/j.1754-7121.2005.tb02189.x

[28] OECD, “The Learning Government: Introduction and draft results of the survey of knowledge management practices in ministries/departments/agencies of central government,” 27th Session of the Public Management Committee, Tech. Rep., 2003.

[29] “GovData.DE,” Online, Federal Ministry of the Interior, Feb 2013. [Online]. Available: www.govdata.de

[30] CKAN, “Comprehensive Knowledge Archive Network.” [Online]. Available: www.ckan.org

[31] N. Tcholtchev and A. Prakash, “On the need for Model-Driven Engineering for Data Harvesters based on Experiences from the German GovData.DE Portal,” in *Share-PSI 2.0 - Lisbon Workshop: Encouraging open data usage by commercial developers*, Oct. 2014, submitted. [Online]. Available: http://www.w3.org/2013/-share-psi/workshop/lisbon/

[32] Wikipedia. OSGi. [Online]. Available: http://en.wikipedia.org/wiki/OSGi

[33] Open Service Gateway Initiative. [Online]. Available: www.osgi.org

[34] A. Prakash, S. K. Rajaguru, and N. Tcholtchev, "An OSGI-based Model-driven Data Management Module for Robust Open Data Harvesting," in *Share-PSI 2.0 - Lisbon Workshop: Encouraging open data usage by commercial developers*, Oct. 2014, submitted. [Online]. Available: http://www.w3.org/2013/share-psi/-workshop/lisbon/

[35] J. D. Haan. (2005, November) 15 reasons why you should start using Model Driven Development. Online. [Online]. Available: http://www.theenterprisearchitect.eu/blog/2009/11/25/15-reasons-why-you-should-start-using-model-driven-development/

[36] D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, *EMF: Eclipse Modeling Framework 2.0*, 2nd ed. Addison-Wesley Professional, 2009.

[37] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities," *Scientific American*, pp. 29–37, May 2001.

[38] T. Berners-Lee and J. Hendler, "Publishing on the Semantic Web," *Nature*, pp. 1023–1025, Apr. 2001.

[39] T. Berners-Lee, "Linked data: Design issues," 2006. [Online]. Available: http://www.w3.org/DesignIssues/-LinkedData.html

[40] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, Information and Computer Science, University of California, Irvine, 2000.

[41] T. Lebo, S. Sahoo, D. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, and J. Zhao, "PROV-O: The PROV ontology," 2014, TW3C Recommendation, World Wide Web Consortium.

[42] H. Abelson, B. Adida, M. Linksvayer, and N. Yergler, *The Digital Public Domain: Foundations for an Open Culture*. Open Book Publishers, Mar. 2012, ch. ccREL: The Creative Commons Rights Expression Language, pp. 149–189. [Online]. Available: http://www.openbookpublishers.com/product/93

[43] A. Latif, A. U. Saeed, P. Höfler, A. Stocker, and C. Wagner, "The Linked Data Value Chain: A Lightweight Model for Business Engineers," in *5th International Conference on Semantic Systems, Graz, Austria, September 2-4, 2009. Proceedings*, 2009, pp. 568–575. [Online]. Available: http://www.i-semantics.at/-2009/papers/the_linked_data_value_chain.pdf

[44] S. Auer, L. Bühmann, C. Dirschl, O. Erling, M. Hausenblas, R. Isele, J. Lehmann, M. Martin, P. Mendes, B. van Nuffelen, C. Stadler, S. Tramp, and H. Williams, "EnglishManaging the Life-Cycle of Linked Data with the LOD2 Stack," in *EnglishThe Semantic Web – ISWC 2012*, ser. Lecture Notes in Computer Science, P. Cudré-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. Parreira, J. Hendler, G. Schreiber, A. Bernstein, and E. Blomqvist, Eds. Springer Berlin Heidelberg, 2012, vol. 7650, pp. 1–16. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-35173-0_1

[45] T. Heath and C. Bizer, *Linked Data: Evolving the Web into a Global Data Space*, 1st ed. Morgan & Claypool, 2011. [Online]. Available: http://linkeddatabook.com/

[46] W. B. Sanders. (2009, May) Design Pattern Principles for ActionScript 3.0: Single Responsibility Principle. [Online]. Available: http://www.as3dp.com/2009/05/design-pattern-principles-for-actionscript-30-single-responsibility-principle/

[47] B. Meyer, *Object-Oriented Software Construction*, 1st ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[48] A.-F. Shubho. (2010, Aug.) How I explained OOD to my wife. [Online]. Available: http://-www.codeproject.com/Articles/93369/How-I-explained-OOD-to-my-wife

[49] R. C. Martin, *Agile Software Development: Principles, Patterns, and Practices*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2003.

[50] "Jar format," 2014.10.15. [Online]. Available: http://en.wikipedia.org/wiki/JAR_(file_format)

[51] “Dynamic-link library,” 2014.10.15. [Online]. Available: http://en.wikipedia.org/wiki/Dynamic-link_library

[52] Web service description language. [Online]. Available: http://en.wikipedia.org/wiki/-Web_Services_Description_Language

[53] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-oriented Software*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1995.

Annex 1 – OpenScout Application Profile

In this Annex, we show the full OpenScout Application profile which would be used as the basis for the creation of the EAGLE Application Profile.

Nr	Name	Explanation	Size	Order	Value space	Datatype	Obligation	Example
1	General	This category groups the general information that describes this learning object as a whole.	1	unspecified	-	-	Mandatory	-
1.1	Identifier	A globally unique label that identifies this learning object.	smallest permitted maximum: 10 items	unspecified	-	-	Mandatory	-
1.1.1	Catalog	The name or designator of the identification or cataloging scheme for this entry. A namespace scheme.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Mandatory	"ISBN", "ARIADNE", "URI" , “EAGLE”
1.1.2	Entry	The value of the identifier within the identification or cataloging scheme that designates or identifies this learning object. A namespace specific string.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Mandatory	"2-7342-0318", "LEAO875", "http://www.ieee.org/documents/1234"
1.2	Title	Name given to this learning object.	1	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Mandatory	("en", "Working life and learning ")

1.3	Language	The primary human language or languages used within this learning object to communicate to the intended user. NOTE 1:--An indexation or cataloging tool may provide a useful default. NOTE 2:--If the learning object had no lingual content (as in the case of a picture of the Mona Lisa, for example), then the appropriate value for this data element would be "none". NOTE 3:--This data element concerns the language of the learning object. Data element 3.4:Meta-Metadata.Language concerns the language of the metadata instance.	smallest permitted maximum: 10 items	unordered	LanguageID = Langcode ("-"Subcode)* with Langcode a language code as defined by the code set ISO 639:1988 and Subcode (which can occur an arbitrary number of times) a country code from the code set ISO 3166-1:1997. NOTE 4:-`This value space is also defined by RFC1766:1995 and is harmonized with that of the xml:lang attribute. NOTE 5:--ISO 639:1988 also includes "ancient" languages, like Greek and Latin. The language code should be given in lower case and the country code (if any) in upper case. However, the values are case insensitive.	CharacterString (smallest permitted maximum: 100 char)	Recommended	"en", "en-GB", "de", "fr-CA", "it" "grc" (ancient greek, until 1453) "en-US-philadelphia" "eng-GB-cockney" "map-PG-buin" (Austronesian –Papua New Guinea – buin) "gem-US-pennsylvania"

					"none" shall also be an acceptable value.			
1.4	Description	A textual description of the content of this learning object. NOTE:--This description need not be in language and terms appropriate for the users of the learning object being described. The description should be in language and terms appropriate for those that decide whether or not the learning object being described is appropriate and relevant for the users.	smallest permitted maximum: 10 items	unordered	-	LangString (smallest permitted maximum: 2000 char)	Recommended	("en", "This unit explores the concepts of national and organisational culture and the factors that influence both.")
1.5	Keyword	A keyword or phrase describing the topic of this learning object. This data element should not be used for characteristics that can be described by other data elements.	smallest permitted maximum: 10 items	unordered	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en", "business administration")

1.6	Coverage	The time, culture, geography, region to which this learning object applies. The extent or scope of the content of the learning object. Coverage will typically include spatial location (a place name or geographic coordinates), temporal period (a period label, date, or date range) or jurisdiction (such as a named administrative entity). Recommended best practice is to select a value from a controlled vocabulary (for example, the Thesaurus of Geographic Names [TGN]) and that, where appropriate, named places or time periods be used in preference to numeric identifiers such as sets of coordinates or date ranges. NOTE 1:--This is the definition from the Dublin Core Metadata Element Set, version 1.14 extended by “politics” and “religion” for the OpenScout project.	smallest permitted maximum: 10 items	unordered	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en", "20th century France") NOTE 2:--A learning object could be about tax law in 20th century France: in that case, its subject can be described with 1.5:General.Keyword=("en","tax law") and its 1.6:General.Coverage can be ("en","20th century France").

1.7	Structure	Underlying organizational structure of this learning object.	1	unspecified	atomic: an object that is indivisible (in this context). collection: a set of objects with no specified relationship between them. networked: a set of objects with relationships that are unspecified. hierarchical: a set of objects whose relationships can be represented by a tree structure. linear: a set of objects that are fully ordered. Example: A set of objects that are connected by "previous" and "next" relationships.	Vocabulary (State)	Optional	NOTE:--A learning object with Structure="atomic" will typically have 1.8:General.AggregationLevel=1. A learning object with Structure="collection", "linear", "hierarchical" or "networked" will typically have 1.8:General.AggregationLevel=2, 3 or 4.

1.8	Aggregation Level	The functional granularity of this learning object.	1	unspecified	1: the smallest level of aggregation, e.g., raw media data or fragments. 2: a collection of level 1 learning objects, e.g., a lesson. 3: a collection of level 2 learning objects, e.g., a course. 4: the largest level of granularity, e.g., a set of courses that lead to a certificate. NOTE 1:--Level 4 objects can contain level 3 objects, or can recursively contain other level 4 objects.	Vocabulary (Enumerated)	Optional	If the learning object is a digital picture of the economic cycle 1.7:General.Structure=Atomic and 1.8:General.AggregationLevel=1. If the learning object is a lesson with a picture of the economic cycle, 1.7:General.Structure=Collection or Networked (since there are two descriptions of the same type of Structure) and 1.8:General.AggregationLevel=2. If the learning object is a course about the economic cycle, 1.7:General.Structure=Linear if the documents are intended to be viewed linearly and 1.8:General.AggregationLevel=3. If the learning object is a collection of lessons on the economic cycle from different sources, 1.7:General.Structure=Collection and 1.8:General:AggregationLevel=3. Lastly if the learning object is a set of courses with a full explanation of the economic cycle, NOTE 2:--A learning object with AggregationLevel=1 will typically have 1.7:General.Structure="atomic". A learning object with AggregationLevel=2, 3 or 4 will typically have 1.7:General.Structure= "collection", 1.7:General.Structure=Linear or Hierarchical and

								1.8:General.AggregationLevel=4. "linear", "hierarchical" or "networked".
2	Life Cycle	This category describes the history and current state of this learning object and those entities that have affected this learning object during its evolution.	1	unspecified	-	-	Optional	-
2.1	Version	The edition of this learning object.	1	unspecified	-	LangString (smallest permitted maximum: 50 char)	Optional	("en", "1.2.alpha"), ("nl", "voorlopige versie")
2.2	Status	The completion status or condition of this learning object.	1	unspecified	draft final revised unavailable NOTE:--When the status is "unavailable" it means that the learning object itself is not available.	Vocabulary (State)	Optional	-

2.3	Contribute	Those entities (i.e., people, organizations) that have contributed to the state of this learning object during its life cycle (e.g., creation, edits, publication). NOTE 1:--This data element is different from 3.3:Meta-Metadata.Contribute. NOTE 2:--Contributions should be considered in a very broad sense here, as all actions that affect the state of the learning object.	smallest permitted maximum: 30 items	ordered	-	-	Optional	-
2.3.1	Role	Kind of contribution. NOTE 1:--Minimally, the Author(s) of the learning object should be described.	1	unspecified	author publisher unknown initiator terminator validator editor graphical designer technical implementer content provider technical validator educational validator script writer instructional designer subject matter expert NOTE 2:--"terminator"	Vocabulary (State)	Conditional - if 2.3.2 present or - if 2.3.3 present	-

					is the entity that made the learning object unavailable.			
2.3.2	Entity	The identification of and information about entities (i.e., people, organizations) contributing to this learning object. The entities shall be ordered as most relevant first.	smallest permitted maximum: 40 items	ordered	vCard, as defined by IMC vCard 3.0 (RFC 2425, RFC 2426).	CharacterString (smallest permitted maximum: 1000 char)	Optional	"BEGIN:VCARD\nFN:Joe Friday\nTEL:+1-919-555-7878\nTITLE:Area Administrator\, Assistant\n EMAIL\;TYPE=INTERN\nET:jfriday@host.c om\nEND:VCARD\n"
2.3.3	Date	The date of the contribution.	1	unspecified	-	DateTime	Optional	"2001-08-23"
3	Meta-Metadata	This category describes this metadata record itself (rather than the learning object that this record describes). This category describes how the metadata instance can be identified, who created this metadata instance, how, when, and with what references. NOTE:--This is not the information that describes the learning object itself.	1	unspecified	-	-	Mandatory	-

3.1	Identifier	A globally unique label that identifies this metadata record.	smallest permitted maximum: 10 items	unspecified	-	-	Mandatory	-
3.1.1	Catalog	The name or designator of the identification or cataloging scheme for this entry. A namespace scheme.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Mandatory	"EAGLE", "URI"
3.1.2	Entry	The value of the identifier within the identification or cataloging scheme that designates or identifies this metadata record. A namespace specific string.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Mandatory	"KUL532", "http://www.ieee.org/descriptions/1234"
3.2	Contribute	Those entities (i.e., people or organizations) that have affected the state of this metadata instance during its life cycle (e.g., creation, validation). NOTE:--This data element is concerned with contributions to the metadata. Data element 2.3:Lifecycle.Contribute is concerned with contributions to the learning object.	smallest permitted maximum: 10 items	ordered	-	-	Mandatory	-
3.2.1	Role	Kind of contribution. Exactly one instance of this data element with value "creator" should exist.	1	unspecified	creator validator	Vocabulary (State)	Mandatory	-

3.2.2	Entity	The identification of and information about entities (i.e., people, organizations) contributing to this metadata instance. The entities shall be ordered as most relevant first.	smallest permitted maximum: 10 items	ordered	vCard, as defined by IMC vCard 3.0 (RFC 2425, RFC 2426).	CharacterString (smallest permitted maximum: 1000 char)	Mandatory	"BEGIN:VCARD\nFN:Joe Friday\nTEL:+1-919-555-7878\nTITLE:Area Administrator\, Assistant\n EMAIL\;TYPE=INTERN\nET:jfriday@host.c om\nEND:VCARD\n"
3.2.3	Date	The date of the contribution.	1	unspecified	-	DateTime	Mandatory	"2001-08-23"
3.3	Metadata Schema	The name and version of the authoritative specification used to create this metadata instance. NOTE:--This data element may be user selectable or system generated. If multiple values are provided, then the metadata instance shall conform to multiple metadata schemas.	smallest permitted maximum: 10 items	unordered	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 30 char)	Optional	"LOMv1.0"

3.4	Language	Language of this metadata instance. This is the default language for all LangString values in this metadata instance. If a value for this data element is not present in a metadata instance, then there is no default language for LangString values. NOTE 1:--This data element concerns the language of the metadata instance. Data element 1.3:General.Language concerns the language of the learning object.	1	unspecified	see 1.3:General.Language For this data element, "none" shall not be an acceptable value. NOTE 2:--"none" is unacceptable, because the metadata instance is in one or more human languages. "none" is acceptable for 1.3:General.Language, as the learning object itself may be in no particular human language. For example, a picture of the Mona Lisa has "none" for 1.3:General.Language. If its description (i.e., metadata instance) is in Swedish, then 3.4:Meta-Metadata.Language has value "sv".	CharacterString (smallest permitted maximum: 100 char)	Optional	"en"
4	Technical	This category describes the technical requirements and characteristics of this learning object.	1	unspecified	-	-	Mandatory	-

4.1	Format	Technical datatype(s) of (all the components of) this learning object. This data element shall be used to identify the software needed to access the learning object.	smallest permitted maximum: 40 items	unordered	MIME types based on IANA registration (see RFC2048:1996) or "non-digital"	CharacterString (smallest permitted maximum: 500 char)	Recommended	"video/mpeg", "application/x-toolbook", "text/html"
4.2	Size	The size of the digital learning object in bytes (octets). The size is represented as a decimal value (radix 10). Consequently, only the digits "0" through "9" should be used. The unit is bytes, not Mbytes, GB, etc. This data element shall refer to the actual size of this learning object. If the learning object is compressed, then this data element shall refer to the uncompressed size.	1	unspecified	ISO/IEC 646:1991, but only the digits "0".. "9"	CharacterString (smallest permitted maximum: 30 char)	Optional	"4200"
4.3	Location	A string that is used to access this learning object. It may be a location (e.g., Universal Resource Locator), or a method that resolves to a location (e.g., Universal Resource Identifier). The first element of this list shall be the preferable location. NOTE:--This is where the learning object described by this metadata instance is physically located.	smallest permitted maximum: 10 items	ordered	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Mandatory	"http://host/id"

4.4	Requirement	The technical capabilities necessary for using this learning object. If there are multiple requirements, then all are required, i.e., the logical connector is AND.	smallest permitted maximum: 40 items	unordered	-	-	Optional	-
4.4.1	OrComposite	Grouping of multiple requirements. The composite requirement is satisfied when one of the component requirements is satisfied, i.e., the logical connector is OR.	smallest permitted maximum: 40 items	unordered	-	-	Optional	-
4.4.1.1	Type	The technology required to use this learning object, e.g., hardware, software, network, etc.	1	unspecified	operating system browser	Vocabulary (State)	Optional	-
4.4.1.2	Name	Name of the required technology to use this learning object. NOTE 1:--The value for this data element may be derived from 4.1:Technical.Format automatically, e.g., "video/mpeg" implies "multi-os". NOTE 2:--This vocabulary includes most values in common use at the time that this Standard was approved.	1	unspecified	if Type="operating system", then: pc-dos ms-windows macos unix multi-os none if Type="browser" then: any netscape communicator ms-internet explorer opera amaya	Vocabulary (State)	Optional	-
4.4.1.	Minimum Version	Lowest possible version of the required technology to use this	1	unspecifie	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest	Optional	"4.2"

3		learning object.		d		permitted maximum: 30 char)		
4.4.1.4	Maximum Version	Highest possible version of the required technology to use this learning object.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 30 char)	Optional	"6.2"
4.5	Installation Remarks	Description of how to install this learning object.	1	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en", "Unzip the zip file and launch index.html in your web browser.")
4.6	Other Platform Requirements	Information about other software and hardware requirements. NOTE:--This element is intended for descriptions of requirements that cannot be expressed by data element 4.4:Technical.Requirement.	1	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en","sound card"), ("en","runtime X")
4.7	Duration	Time a continuous learning object takes when played at intended speed. NOTE:--This data element is especially useful for sounds, movies or animations.	1	unspecified	-	Duration	Optional	"PT1H30M", "PT1M45S"

5	Educational	This category describes the key educational or pedagogic characteristics of this learning object. NOTE:--This is the pedagogical information essential to those involved in achieving a quality learning experience. The audience for this metadata includes teachers, managers, authors, and learners.	smallest permitted maximum: 100 items	unspecified	-	-	Optional	-

5.1	Interactivity Type	Predominant mode of learning supported by this learning object. "Active" learning (e.g., learning by doing) is supported by content that directly induces productive action by the learner. An active learning object prompts the learner for semantically meaningful input or for some other kind of productive action or decision, not necessarily performed within the learning object's framework. Active documents include simulations, questionnaires, and exercises. "Expositive" learning (e.g., passive learning) occurs when the learner's job mainly consists of absorbing the content exposed to him (generally through text, images or sound). An expositive learning object displays information but does not prompt the learner for any semantically meaningful input. Expositive documents include essays, video clips, all kinds of graphical material, and hypertext documents. When a learning object blends the active and expositive interactivity types, then its interactivity type is "mixed". NOTE:--Activating links to navigate in hypertext documents is not considered to be a productive action.	1	unspecified	active expositive mixed	Vocabulary (State)	Optional	active documents (with learner's action): · simulation (manipulates, controls or enters data or parameters); · questionnaire (chooses or writes answers); · exercise (finds solution); · problem statement (writes solution). expositive documents (with learner's action): · hypertext document (reads, navigates); · video (views, rewinds, starts, stops); · graphical material (views); · audio material (listens, rewinds, starts, stops). mixed document: · hypermedia document with embedded simulation applet.

5.2	Learning Resource Type	Specific kind of learning object. The most dominant kind shall be first. NOTE:--The vocabulary terms are defined as in the OED:1989 and as used by educational communities of practice.	smallest permitted maximum: 10 items	ordered	exercise simulation questionnaire diagram figure graph index slide table narrative text exam experiment problem statement self assessment lecture Extension of the original LOM draft standard value space: case study	Vocabulary (State)	Optional	-

					working paper presentation research paper checklist course module full course video case study modeling tools games serious games virtual worlds role-playing games MUD			

5.3	Interactivity Level	The degree of interactivity characterizing this learning object. Interactivity in this context refers to the degree to which the learner can influence the aspect or behavior of the learning object. NOTE 1:--Inherently, this scale is meaningful within the context of a community of practice.	1	unspecified	very low low medium high very high	Vocabulary (Enumerated)	Optional	NOTE 2:--Learning objects with 5.1:Educational.InteractivityType="active" may have a high interactivity level (e.g., a simulation environment endowed with many controls) or a low interactivity level (e.g., a written set of instructions that solicit an activity). Learning objects with 5.1:Educational.InteractivityType="expositive" may have a low interactivity level (e.g., a piece of linear, narrative text produced with a standard word processor) or a medium to high interactivity level (e.g., a sophisticated hyperdocument, with many internal links and views).

5.4	Semantic Density	The degree of conciseness of a learning object. The semantic density of a learning object may be estimated in terms of its size, span, or --in the case of self-timed resources such as audio or video--duration. The semantic density of a learning object is independent of its difficulty. It is best illustrated with examples of expositive material, although it can be used with active resources as well. NOTE 1:--Inherently, this scale is meaningful within the context of a community of practice.	1	unspecified	very low low medium high very high	Vocabulary (Enumerated)	Optional	*Active* documents: user interface of a simulation · low semantic density: a screen filled up with explanatory text, a picture of a business process, and a single button labeled "Click here to continue" · high semantic density: screen with short text, same picture, and three buttons labeled "Change compression ratio", "Change octane index", "Change ignition point advance" *Expositive* documents: · medium difficulty text document o medium semantic density or high semantic density.

							Optional	· easy video document o low semantic density: The full recorded footage of a conversation between two experts on the differences between Asian and African elephants; 30 minutes duration. o high semantic density: An expertly edited abstract of the same conversation; 5 minutes duration · difficult mathematical notation o medium semantic density: The text representation of the theorem: For any given set j, it is always possible to define another set y, which is a superset of j. o very high semantic density: The symbolic representation (formula) of the theorem ("j $y: y É j)

5.5	Intended End User Role	Principal user(s) for which this learning object was designed, most dominant first. NOTE 1:--A learner works with a learning object in order to learn something. An author creates or publishes a learning object. A manager manages the delivery of this learning object, e.g., a university or college. The document for a manager is typically a curriculum. NOTE 2:--In order to describe the intended end user role through the skills the user is intended to master, or the tasks he or she is intended to be able to accomplish, the category 9:Classification can be used.	smallest permitted maximum: 10 items		teacher author learner manager	Vocabulary (State)	Optional	An authoring tool that produces pedagogical material is a typical example of a learning object whose intended end user is an author
5.6	Context	The principal environment within which the learning and use of this learning object is intended to take place. NOTE:--Suggested good practice is to use one of the values of the value space and to use an additional instance of this data element for further refinement, as in ("LOMv1.0","higher education") and ("http://www.ond.vlaanderen.be/ onderwijsinvlaanderen/Default.htm"	smallest permitted maximum: 10 items	unordered	school higher education training other Extension of the original LOM draft	Vocabulary (State)	Optional	-

		, "kandidatuursonderwijs")			standard value space: Conference Vocational training SME Vocational training large company			
5.7	Typical Age Range	Age of the typical intended user. This data element shall refer to developmental age, if that would be different from chronological age. NOTE 1:--The age of the learner is important for finding learning objects, especially for school age learners and their teachers. When applicable, the string should be formatted as minimum age-maximum age or minimum age-. (NOTE:--This is a compromise between adding three component elements (minimum age, maximum age, and description) and having just a free text field.) NOTE 2:--Alternative schemes for what this data element tries to cover (such as various reading age	smallest permitted maximum: 5 items	unordered	-	LangString (smallest permitted maximum: 1000 char)	Optional	"7-9", "0-5", "15", "18-", ("en","suitable for children over 7"), ("en","adults only")

		or reading level schemes, IQ's or developmental age measures) should be represented through the 9:Classification category.						
5.8	Difficulty	How hard it is to work with or through this learning object for the typical intended target audience. NOTE:--The " typical target audience" can be characterized by data elements 5.6:Educational.Context and 5.7:Educational.TypicalAgeRange.	1	unspecified	very easy easy medium difficult very difficult	Vocabulary (Enumerated)	Optional	-

5.9	Typical Learning Time	Approximate or typical time it takes to work with or through this learning object for the typical intended target audience. NOTE:--The " typical target audience" can be characterized by data elements 5.6:Educational.Context and 5.7:Educational.TypicalAgeRange.	1	unspecified	-	Duration	Optional	"PT1H30M", "PT1M45S"
5.10	Description	Comments on how this learning object is to be used.	smallest permitted maximum: 10 items	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en", "Teacher guidelines that come with a textbook.")
5.11	Language	The human language used by the typical intended user of this learning object.	smallest permitted maximum: 10 items	unordered	See 1.3:General. Language	CharacterString (smallest permitted maximum: 100 char)	Optional	"en", "en-GB", "de", "fr-CA", "it" NOTE:--As an example, for a learning object in French, intended for English-speaking students, the value of 1.3:General.Language will be French, and the value of 5.11:Educational.Language will be English.

5.12	Region	Intended region for the usage of the learning resource	Smallest permitted maximum: 10 items	unordered	country names given in ISO 3166-2	Vocabulary (State)	Optional Extension of the LOM draft standard	“France”, “United Kingdom”
5.13	Learning Theory	The learning theory the learning resource belongs to.	1	unspecified	behaviourist cognitivist constructivist	Vocabulary (State)	Optional Extension of the LOM draft standard	

5.14	Focus	Abstraction level of the learning resource	1	unspecified	abstract concrete	Vocabulary (State)	Optional Extension of the LOM draft standard	
5.15	Teacher Learner Role	Relation between teacher and learner	1	unspecified	strict control uncontrolled	Vocabulary (State)	Optional Extension of the LOM draft standard	

5.16	Learning Strategy	Strategies expected from the learners	1	Unspecified	individualism collectivism	Vocabulary (State)	Optional Extension of the LOM draft standard	
6	Rights	This category describes the intellectual property rights and conditions of use for this learning object. NOTE:--The intent is to reuse results of ongoing work in the Intellectual Property Rights and e-commerce communities. This category currently provides the absolute minimum level of detail only.	1	unspecified	-	-	Mandatory	-
6.1	Cost	Whether use of this learning object requires payment.	1	unspecified	yes no	Vocabulary (State)	Optional	-
6.2	Copyright and Other Restrictions	Whether copyright or other restrictions apply to the use of this learning object.	1	unspecified	yes no	Vocabulary (State)	Mandatory	-

6.3	Description	Comments on the conditions of use of this learning object.	1	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Mandatory	("en", "Use of this learning object is only permitted after a donation has been made to Amnesty International.")
7	Relation	This category defines the relationship between this learning object and other learning objects, if any. To define multiple relationships, there may be multiple instances of this category. If there is more than one target learning object, then each target shall have a new relationship instance.	smallest permitted maximum: 100 items	unordered	-	-	Optional	-
7.1	Kind	Nature of the relationship between this learning object and the target learning object, identified by 7.2:Relation.Resource.	1	unspecified	Based on Dublin Core: ispartof: is part of haspart: has part isversionof: is version of hasversion: has version isformatof: is format of hasformat: has format references: references isreferencedby: is referenced by isbasedon: is based on isbasisfor:	Vocabulary (State)	Optional	-

					is basis for requires: requires isrequiredby: is required by			
7.2	Resource	The target learning object that this relationship references.	1	unspecified	-	-	Optional	-
7.2.1	Identifier	A globally unique label that identifies the target learning object.	smallest permitted maximum: 10 items	unspecified	-	-	Optional	-
7.2.1.1	Catalog	The name or designator of the identification or cataloging scheme for this entry. A namespace scheme.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Optional	"ISBN", "ARIADNE", "URI"
7.2.1.2	Entry	The value of the identifier within the identification or cataloging scheme that designates or identifies the target learning object. A namespace specific string.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 1000 char)	Optional	"2-7342-0318", "LEAO875", "http://www.ieee.org/"

7.2.2	Description	Description of the target learning object.	smallest permitted maximum: 10 items	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en","A digital text document about the use of Gantt charts.")
8	Annotation	This category provides comments on the educational use of this learning object, and information on when and by whom the comments were created. This category enables educators to share their assessments of learning objects, suggestions for use, etc. Extension of the LOM draft standard: This category also provides comments on the cultural and context specific adaption needs identified for the learning resource.	smallest permitted maximum: 30 items	unordered	-	-	Optional	-
8.1	Entity	Entity (i.e., people, organization) that created this annotation.	1	unspecified	vCard, as defined by IMC vCard 3.0 (RFC 2425, RFC 2426).	CharacterString (smallest permitted maximum: 1000 char)	Optional	"BEGIN:VCARD\nFN:Joe Friday\nTEL:+1-919-555-7878\nTITLE:Area Administrator\, Assistant\n EMAIL\;TYPE=INTERN\nET:jfriday@host.c om\nEND:VCARD\n"
8.2	Date	Date that this annotation was created.	1	unspecified	-	DateTime	Optional	"2001-08-23"

8.3	Description	The content of this annotation.	1	unspecified	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en", "I have used this video clip with my students. They really enjoyed it. Make sure they have a broadband connection or the experience becomes too cumbersome to be educationally interesting.")
9	Classification	This category describes where this learning object falls within a particular classification system. To define multiple classifications, there may be multiple instances of this category.	smallest permitted maximum: 40 items	unordered	-	-	Optional	-
9.1	Purpose	The purpose of classifying this learning object.	1	unspecified	discipline idea prerequisite educational objective accessibility restrictions educational level skill level security level	Vocabulary (State)	Conditional - if 9.2 present or - if, 9.3 present or - if 9.4 present	-

					competency Extension of the original LOM draft standard value space: industry sector			
9.2	Taxon Path	A taxonomic path in a specific classification system. Each succeeding level is a refinement in the definition of the preceding level. There may be different paths, in the same or different classifications, which describe the same characteristic.	smallest permitted maximum: 15 items	unordered	-	-	Optional	-
9.2.1	Source	The name of the classification system. This data element may use any recognized "official" taxonomy or any user-defined taxonomy. NOTE:--An indexation, cataloging or query tool may provide the top-level entries of a well-established classification, such as the Library of Congress Classification (LOC), Universal Decimal Classification (UDC), Dewey Decimal Classification (DDC), etc.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	LangString (smallest permitted maximum: 1000 char)	Optional	("en","ACM"), ("en","MESH"), ("en","ARIADNE")

9.2.2	Taxon	A particular term within a taxonomy. A taxon is a node that has a defined label or term. A taxon may also have an alphanumeric designation or identifier for standardized reference. Either or both the label and the entry may be used to designate a particular taxon. An ordered list of taxons creates a taxonomic path, i.e., "taxonomic stairway": this is a path from a more general to more specific entry in a classification.	smallest permitted maximum: 15 items	ordered	-	-	Optional	{["1",("en","economic")], [“1.2”, (“en”, “banking and finance”)], ["1.2.1",("en","accounting")], ["1.2.1.3",("en","managerial accounting")]}
9.2.2.1	Id	The identifier of the taxon, such as a number or letter combination provided by the source of the taxonomy.	1	unspecified	Repertoire of ISO/IEC 10646-1:2000	CharacterString (smallest permitted maximum: 100 char)	Conditional - if 9.1 present or - if 9.2 present or - if, 9.3 present or - if 9.4 present	"320", "4.3.2", "BF180"
9.2.2.2	Entry	The textual label of the taxon.	1	unspecified	-	LangString (smallest permitted maximum: 500 char)	Optional	("en", "banking and finance")

9.2.2.3	Min EQF	Minimum proficiency level of the respective competence expected to be achieved by the consumer of the LO.	1	unspecified	Proficiency levels defined in the European Qualifications Framework (EQF, http://ec.europa.eu/education/lifelong-learning-policy/doc44_en.html)	Vocabulary (Enumerated)	Optional Extension of the LOM draft standard	
9.2.2.4	Max EQF	Maximum proficiency level of the respective competence expected to be achieved by the consumer of the LO.	1	unspecified	Proficiency levels defined in the European Qualifications Framework (EQF, http://ec.europa.eu/education/lifelong-learning-policy/doc44_en.html)	Vocabulary (Enumerated)	Optional Extension of the LOM draft standard	
9.3	Description	Minimum proficiency level of the respective competence expected to be achieved by the consumer of the LO.	1	unspecified	-	LangString (smallest permitted maximum: 2000 char)	Optional	("en"," Finance is the science of funds management.")
9.4	Keyword	Keywords and phrases descriptive of the learning object relative to the stated 9.1:Classification.Purpose of this specific classification, such as accessibility, security level, etc., most relevant first.	smallest permitted maximum: 40 items	ordered	-	LangString (smallest permitted maximum: 1000 char)	Optional	("en", "customer service")