# Initial Architecture for Fast Small-Scale Deployment

**Date:** October 31, 2013  
**Responsible Partner:** RWTH  
**Authors and Contributors:** Michael Derntl, Ralf Klamma, István Koren, Milos Kravcik, Petru Nicolaescu, Dominik Renzel (RWTH); Kiarii Ngua, Jukka Purma (AALTO); David Zaki (HsKA); Tamsin Treasure-Jones (LEEDS); Graham Attwell, Owen Gray (PONT); Tobias Ley, Vladimir Tomberg (TLU); Christina Henry, Chris Whitehead (TRIBAL); Dieter Theiler, Christoph Trattner (TUG); Ronald Maier, Markus Manhart, Maria Schett, Stefan Thalmann (UIBK)  
**Deliverable Number:** D6.1  
**Work Package:** 6
## Revision History

<table>
<thead>
<tr>
<th>Vers.</th>
<th>Date</th>
<th>Partner(s)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>06 Aug 2013</td>
<td>RWTH</td>
<td>Initial outline based on month 6 preview document; distributed to WP6 partners</td>
</tr>
<tr>
<td>29</td>
<td>10 Oct 2013</td>
<td>RWTH</td>
<td>First complete draft; sent to PONT, UIBK and coordinator for internal review</td>
</tr>
<tr>
<td>40</td>
<td>27 Oct 2013</td>
<td>RWTH</td>
<td>Revised draft addressing comments by internal reviewers PONT, UIBK and the coordinator</td>
</tr>
<tr>
<td>47</td>
<td>31 Oct 2013</td>
<td>RWTH</td>
<td>Final version; submitted to coordinator</td>
</tr>
</tbody>
</table>
Executive Summary

The main goal of WP6 – “Architecture and Integration” – in Learning Layers is to bridge the divide between fast and flexible deployment of Layers tools and technology in the application partner environments by providing a unified, configurable, lightweight and distributed infrastructure. To this end WP6 work during the first project year was focused on the deployment of a small-scale architecture for fast deployment, which is also the subject of this deliverable. As outlined in Section 2, the Layers architecture vision is built on three cornerstones. First, *research orientation* to be able to align the engineering processes with the challenges of scaling in informal learning scenarios. Second, *open source development* with the aim of involving and integrating as many user and developer groups as possible—also externally. Third, *continuous integration* on the co-design and infrastructure levels to ensure a cohesive and connected user experience based on an integrated Layers architecture. Realizing these cornerstones, the core part of the deliverable comprises Sections 3 to 5, reporting respective work streams and successes during Year 1 as summarized in the following.

A survey of existing architectural models and infrastructure-level technologies was conducted during the first six months of the project. With the main first-year objective being fast deployment it was clear that we would not be able to adopt technologies that are too complicated to setup, deploy, manage, or integrate. The attention was therefore put on technologies that were brought to the project by the technology partners, and to further develop those technologies in tight integration with the co-design work pursued by the four Layers design teams. The *House of Quality (HoQ)* approach was adopted to offer a consulting tool for assessing architectural options and comparing available tools for informal learning support. The end-user requirements were elicited through co-design activities and captured in *Requirements Bazaar*, a social requirements engineering toolkit that was initially developed by RWTH in the ROLE project and was awarded the best demo paper award at the 2013 IEEE International Conference on Requirements Engineering.

A major driver for developing and integrating Layers technology during the first year was the *Layers Developer Task Force (LDTF)*, an association of developers from the technical consortium partners who met regularly and acted largely autonomously. To facilitate this distributed development workforce we followed the vision of open (source) development by setting up the *Open Developer Library (ODevL)*, a development infrastructure that is open to external developers and enables eliciting and managing user requirements, hosting the source code, tracking issues, and managing a continuous integration process. The 1st Layers Developer Camp organized by LDTF was a full success in involving external developers by bringing together 26 developers from the technical partners in the Learning Layers project and the external developer community from the Aachen region. As a further measure for involving external developers, two student teams from the “High-tech Entrepreneurship and New Media” lab course at RWTH were recruited to support development in the CAPTUS design team and in integrating TUG’s Social Semantic Server with RWTH’s Semantic Video Annotation technology.

The Architecture Board was established with the responsibility of taking project-wide decisions on the overall Layers architecture. It signed off the year-one small-scale Layers architecture for fast deployment, which lays the foundation for the continuous integration vision on the co-design and infrastructure levels. The architecture is rooted in the design team and development activities during Year 1 and the use cases stemming from these co-design activities. It integrates components contributed by WP6 technical partners on three layers: the cloud infrastructure layer based on i5Cloud and m-learning platform; the application services layer featuring social-semantic services, community support, semantic video annotation, mobile Web, and formative evaluation services; and the service access layer, which will serve as a means for increasing flexibility during deployment in the application clusters in the coming project years.

http://Learning-Layers.eu
# Table of Contents

1. **Introduction** .................................................................................. 1

2. **Layers Architecture Vision** ............................................................. 2

3. **Survey of Technologies for Scaling in Layers** ................................. 4
   3.1 Methodology ................................................................................. 4
   3.2 Technology Collection .................................................................. 5
   3.3 Requirements Engineering .......................................................... 7
   3.4 Technology Assessment: House of Quality .................................... 10

4. **Open Development and Integration Process and Infrastructure** ...... 13
   4.1 Layers Architecture Board .......................................................... 14
   4.2 Layers Developer Taskforce ......................................................... 15
   4.3 Layers Developer Camp 2013 ..................................................... 16
   4.4 Open Developer Library .............................................................. 16
      4.4.1 Requirements Bazaar ............................................................. 18
      4.4.2 Issue Tracking ..................................................................... 21
      4.4.3 Source Code Management .................................................. 22
      4.4.4 Continuous Integration ......................................................... 24

5. **Initial Layers Architecture** .............................................................. 25
   5.1 Architecture Components .......................................................... 26
      5.1.1 i5Cloud ............................................................................... 27
      5.1.2 Tribal M-Learning Platform ................................................. 29
      5.1.3 LAS/LAS2peer ................................................................. 30
      5.1.4 Social Semantic Server ....................................................... 34
      5.1.5 ROLE SDK ......................................................................... 34
      5.1.6 SeViAnno ............................................................................. 37
      5.1.7 MobSOS .............................................................................. 38
      5.1.8 Intradoc247 (Candidate) ..................................................... 40
   5.2 Overview of Layers Web Services ................................................ 41
   5.3 Data Security and Privacy ............................................................ 43
   5.4 Integration with Co-Design Process .............................................. 44
      5.4.1 Feature Extraction ............................................................... 44
      5.4.2 CAPTUS Design Team ......................................................... 45
      5.4.3 Bits and Pieces Design Team ................................................ 46
      5.4.4 Sharing Turbine Design Team ............................................... 46
5.4.5 PANDORA Design Team
5.4.6 Implications for the Integrated Architecture
5.5 Propelling External Developer Engagement
5.5.1 Summer School Workshops
5.5.2 High-Tech Entrepreneurship Lab at RWTH
5.5.3 Student Co-Supervision

6 Conclusion and Outlook
6.1 Conclusion
6.2 Outlook on Year 2

7 References

Appendix A: Technology Survey
Appendix B: Functional Requirements
Appendix C: Non-Functional Requirements
Appendix D: SVN Repository
Appendix E1: CAPTUS HoQ
Appendix E2: Bits & Pieces HoQ
Appendix E3: Sharing Turbine HoQ
Appendix E4: PANDORA HoQ
List of Abbreviations

AAN .................. Artifact Actor Network
AGENTUR ........... Agentur für Nachhaltiges Bauen GmbH (consortium partner)
AnViAnno .......... Android Video Annotation
APD ................... Application Partner Days
API ..................... Application Programming Interface
CoP .......... Community of Practice
DoW ............... Description of Work
DUI ..................... Distributed User Interface
HENM ............... High-Tech Entrepreneurship and New Media
HoQ .......... House of Quality
HsKA ............. Hochschule Karlsruhe Technik und Wirtschaft (consortium partner)
HTTP ............ Hypertext Transfer Protocol
IS .................... Information System
ITB .............. Institut Technik und Bildung, Universität Bremen (consortium partner)
IWC ................ Inter-Widget Communication
LAS ................ Lightweight Application Server
LDTF .......... Layers Developer Task Force
MobSOS .......... Mobile Service Oracle for Success
NFC .......... Near Field Communication
NZNB ............ Norddeutsches Zentrum für Nachhaltiges Bauen
ODevL .......... Open Developer Library
ODL ................. Open Design Library
OSS ............. Open Source Software
P2P ............ Peer-to-Peer
PLE ............... Personal Learning Environment
PONT ............ Pontydysgu Ltd. (consortium partner)
QFD .................. Quality Function Deployment
RDF ........... Resource Description Framework
RE .................. Requirements Engineering
REST ............ Representational State Transfer
ROLE .......... Responsive Open Learning Environments (http://www.role-project.eu)
RWTH..................RWTH Aachen University (consortium partner)
SDK..................Software Development Kit
SeViAnno.........Semantic Video Annotation
SME ..................Small and medium enterprises
SOA..................Service Oriented Architecture
SRE ..................Social Requirements Engineering
SSN ..................Social Semantic Network
SSS ..................Social Semantic Server
SVN .................Apache Subversion
TLU ..................Tallinn University (consortium partner)
TUG ..................Technische Universität Graz (consortium partner)
UML ..................Unified Modeling Language
UWE .................University of the West of England, Bristol (consortium partner)
WP1..................Work Package 1: Workplace Learning
WP2..................Work Package 2: Networked Scaffolding – Interacting with People
WP3..................Work Package 3: Creating and Maturing Instructional Material
WP4..................Work Package 4: Digitally-Enhanced Artefacts, Mobiles and Multimedia
WP5..................Work Package 5: Social Semantic Network
WP6..................Work Package 6: Architecture and Integration
WP7..................Work Package 7: Implementation in Regional SME Clusters
XMPP.................Extensible Messaging and Presence Protocol
1 Introduction

This deliverable provides a summary of the work done in the first year of the Learning Layers project in the WP6 “Architecture and Integration”, which coordinates and synchronizes all technology development tasks across the project as well as provides a base architecture to support this. The main goal of WP6 is to provide a unified, configurable, lightweight and distributed infrastructure for fast and flexible deployment of Layers tools and services, as required by WPs 2-4. WP6 also maintains a distributed, open software engineering process with development procedures and tools for requirements engineering, source code hosting, documentation, and issue tracking.

The aim of D6.1 is to present an initial architecture for small-scale deployment, reporting on the outcomes of the technology survey and the chosen infrastructure (T6.1), the specification of and instruments for the Layers software engineering process (T6.2), a description of the deployed default version of the Layers architecture (T6.3), as well as initial technology-lessons learned from the application in test beds (T6.5).

Our offerings are coming from the Web engineering/analytics area and via systematic Requirements Engineering (RE) in informal learning communities we can understand how practices in informal learning can be reshaped by such tools. These successful transformations or shaping of practices is what we want to scale in the end. For instance, the SeViAnno scenario as described in Section 5.1.6 being picked up by the CAPTUS design team (see D4.1, Section 4) illustrates how a piece of Web engineering can be used in an informal learning scenario. The idea of Communities of Practice (CoP) [1] is central for all three areas that are considered: research, development and integration. The spread of Social Software caused a cultural and technological shift that had a large impact on competence development, including both organization of knowledge work and operation of professional communities (see Table 1).

<table>
<thead>
<tr>
<th>Cultural and Technological Shift by Social Software</th>
<th>Impact on Knowledge Work</th>
<th>Impact on Professional Communities</th>
</tr>
</thead>
<tbody>
<tr>
<td>personal website and content management</td>
<td>Microcontent</td>
<td>Social learning</td>
</tr>
<tr>
<td>blogging and wikis</td>
<td>Providing commentary</td>
<td>Identifying competences</td>
</tr>
<tr>
<td>User generated content</td>
<td>Personal knowledge</td>
<td>Emergent Collaboration</td>
</tr>
<tr>
<td>Participation</td>
<td>publishing</td>
<td>Trust &amp; Social capital</td>
</tr>
<tr>
<td></td>
<td>Establishing personal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>networks</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Testing Ideas</td>
<td></td>
</tr>
<tr>
<td>directories (taxonomy) and slickness</td>
<td>Ranking</td>
<td></td>
</tr>
<tr>
<td>Tagging (&quot;folksonomy&quot;) and syndication</td>
<td>Sense-making</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Remixing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aggregation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Embedding</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Emergent Metadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Collective Intelligence</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wisdom of the Crowd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Collaborative Filtering</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Visualizing Knowledge Networks</td>
</tr>
</tbody>
</table>
These technological and cultural developments provide also new opportunities how to scale the support for informal learning at the workplace. WP6 will integrate achievements of other related work packages – including networked scaffolding (WP2), maturing instructional materials (WP3), mobile and contextual tools (WP4), as well as a social semantic network (WP5).

In the following we first outline the Layers architecture vision. Then we survey technologies for scaling in the project, introducing our methodology, technology collection, requirements engineering, and technology assessment using the House of Quality instrument. The next part explains our open development and integration process and infrastructure, including the Layers Architecture Board, the Layers Developer Taskforce (LDTF), and the Open Developer Library (ODevL). The core of this deliverable is the description of the Layers initial architecture, presenting its components, the integration with the co-design process, and further co-design activities. Finally we conclude this document and give an outlook for the next year.

2 Layers Architecture Vision

The vision of a scalable, flexible, fast and deployable Learning Layers architecture is built on three cornerstones.

1. **Research Orientation:** Like in the EU FP7 IP ROLE (Responsive Open Learning Environments; [http://role-project.eu](http://role-project.eu)) we research a large-scale, Web-based engineering process [3] [4] [5] based on a deep understanding of fundamental laws of the Web and the elicitation and processing of real user needs. It has been a fundamental insight from the ROLE project that in informal learning the requirements of learners and learning communities will be much more heterogeneous than in any other formal learning situation. Furthermore, informal learning faces serious challenges in scaling up learning technologies. The research opportunity here is the adaptation of Web-based engineering processes for a huge number of informal learning communities. An example would be the adaptation of open-source-software development processes within communities that have been relying on collections of simple Web 2.0 tools or commercial off-the-shelf software packages. The management of such large-scale engineering processes demands tool support since the sheer mass of incoming feedback data cannot be processed manually anymore. Together with the other work packages, in particular with WP5, we develop new Web analytics methods, e.g. for identifying experts in informal learning processes based on contribution structures in the Web 2.0 (learning analytics), or overlapping community detection algorithms to identify boundary objects and boundary spanners for scaling-up informal learning across different communities (community analytics; see also [6]). Figure 1 gives an overview about the research approach for the development of the Layers architecture (top-most rectangle). The outer rectangles (Web Engineering, Web Analytics and Requirements Engineering) denote the research areas. In the inner part the left-hand pieces of the pie are more related to Web Engineering while the right-hand pieces are related to Web Analytics. We deeply integrate Web Analytics with Web Engineering, which is visualized by the arrows between the pieces of the pie chart. Since the Web and the mobile world converge, also mobile application developments becomes more and more influenced.
by Web technologies. The two major use cases peer production and scaffolding will be realized in a community oriented way. Last but not least, we connect Web Engineering and Web Analytics by the Requirements Engineering set up as a Web-based process itself with means of the Requirements Bazaar which is explained later in this deliverable.

![Figure 1. Research Orientation of the Layers Architecture](image)

2. **Open Source Development**: We propose an Open Developer Library from the beginning of the project to facilitate open source processes through the whole project to integrate as many as possible user and developer groups. The Open Developer Library integrates with our requirements engineering approach, facilitating the communication between users in many learning communities and open source developers for the deployment of open innovation processes. To overcome cold start problems we introduce a tool we already used in the ROLE project: a developer task force with regular developer camps. The developer camps and the developer task force are facilitating a community of practice of Layers developers and attract at the same time also open source developers which are not directly related to the Layers project. This has led to a wide recognition of ROLE in the open source community and has been also successful in Learning Layers in the first year.

3. **Continuous Integration**: From the very beginning, all Learning Layers components and offerings are to be integrated in the overall Layers architecture. Integration has several dimensions within the project. First, there is the continuous integration with the design teams, which is expressed in the systematic requirements engineering process. Second, there is the backbone or server-side integration. The Layers architecture vision can be expressed best as a box, which can be easily deployed in a new context – a company, a network, a cluster. Based on a cloud computing platform, Layers components can be integrated either on a very coarse level as instances
running on a cloud platform or on a very fine-grained level as Layers services which are deployed as RESTful Web services. Additionally, components can offer programming language interfaces to support server side integration. Last but not least, there is the client-side integration. Integration efforts are made mainly in the design teams by building mockups and first prototypes. From this the project will provide common libraries for development like an extension of the ROLE SDK, which was developed for supporting widget-based Personal Learning Environments (PLE) [7]. By its abilities for supporting real-time communication in the browser [8] and the recent multi-device extension [9] it is a perfect candidate for the facilitation of client-side integration.

3 Survey of Technologies for Scaling in Layers

To obtain an overview of existing technologies for a scalable infrastructure in Layers, task T6.1 was active in the first six months of the project to survey, evaluate and compare existing technology frameworks and models to support a unified Layers infrastructure. As part of the research orientation in the Layers architecture vision, WP6 partners performed a systematic collection and assessment of available technologies with a focus on technologies available and accessible from within the project consortium. In month 6 of the project, we presented a preliminary internal report1, which focused on presenting and applying the selected collection and assessment methodology to see whether it would allow us to make grounded recommendations and decisions regarding the Layers software architecture and technical infrastructure. This section presents the consolidated findings, which represent the fundament for the initial architecture as presented in Section 5 and is structured as follows. In Section 3.1 we outline the methodology applied for collecting and assessing relevant technologies and architectural models. The methodology has three core activity streams, namely technology collection, requirements engineering, and technology assessment, which are described in detail in Sections 3.2, 3.3, and 3.4, respectively.

3.1 Methodology

Layers technical solutions were promised to be fast and easy to deploy, and to run on existing infrastructures and devices as far as possible to keep deployment and maintenance efforts at a minimum. Also, the technology shall align smoothly with the tasks and processes at work, which is assured in Layers through the work in the design teams, four of which have been established so far (see D1.1).

Meeting these challenges requires a sound methodological basis to guide the architecture and integration tasks in Layers over the coming years. The selected survey methodology relies on three activities, including requirements engineering based on the activities in the design teams and involving end-users; collecting and describing technologies that are available to support Layers scenarios and use cases; and employing an assessment

1 https://docs.google.com/file/d/0B2jesE8OfDgRWXhuRHNmdmdfYVE
instrument that shall facilitate decision making by comparing products in terms of how well they support the elicited requirements.

In order to document and compare the different technologies and to obtain a traceable approach to architectural and infrastructural decision making by the Architecture Board (cf. Section 4.1) in Layers, the technology survey task was approached using the methodology illustrated in Figure 2. It relies on three activities:

1. **Technology Collection**: the objective of this activity was to collect architectural models and concrete infrastructure-level technologies that are potentially relevant and related to Layers using a desk research approach. In this deliverable we limit the presentation of the results to short descriptions and point to the Layers wiki for detailed listings. See Section 3.2.

2. **Requirements Engineering**: the objective of this activity was to elicit, consolidate, and prioritize user requirements from different end-user sources and design activities in Layers following an open development approach as postulated in the Layers architecture vision. The requirements are the main ingredient to building an infrastructure that serves the Layers purposes. See details in Section 3.3.

3. **Technology Assessment**: based on the artifacts obtained in the requirements engineering and technology survey activities, House of Quality (HoQ) [10] was adopted as an instrument for obtaining and assessing technical requirements to be met by technology products using the collected technologies and the prioritized user requirements. See details in Section 3.4.

![Figure 2. Technology Survey Activities and Outputs.](image)

### 3.2 Technology Collection

We created a structured collection of existing infrastructure-level technologies and architectural models for the Layers infrastructure. This was achieved using a structured template and the Layers MediaWiki’s semantic capabilities. MediaWiki properties were used to tag and categorize pages. The technologies collected by WP6 partners contain architectural models as high-level technologies and concrete systems and products that implement or are based on these models. Architectural models represent a high-level view on the characteristics, architecture, patterns, standards, etc. which may be used to categorize and describe a set of specific products. As an example, we consider “Software as a Service (SaaS)” in Cloud Computing to be an architectural model, which describes how software and data can be hosted in the cloud. In this particular case, products such as Google Docs,
Amazon Web Services, i5Cloud, etc. are implementations of SaaS, which have in common similar architectural principles.

The technology collection includes both prototypes from the project partners and existing external systems. Aiming at fast deployment, the focus was put on technologies contributed by WP6 partners. The models and technologies collected on the Layers wiki are listed in Appendix A. Below, we provide a short description of the most relevant models and products available.

**Architectural Models:**

- **Federated IS Architecture** is built up of independently deployed computing nodes that are linked together by a network structure and thus operate in a collective way. It can support well different application types (new, third party, legacy) in a distributed environment.

- **Service Oriented Architecture (SOA)** exposes well defined functionalities of a system as services. SOA supports important characteristics of software systems, such as loose coupling, re-usability and a simple/composite implementation.

- **Resource Oriented Architecture** applies the principles of **Representational State Transfer (REST)** [11] and specifies how RESTful Web Services are developed. Design principles are statelessness, addressability, connectedness and the uniform interface.

- **Cloud Computing** enables on-demand, scalable network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services)\(^2\). Reliability, performance and scalability are among the most important characteristics enabled by this model.

**Products:**

- **ROLE Infrastructure**, based on the federation and resource-oriented models, offers support for developing open personal learning environments where users can plan their learning process, learn, collaborate, and reflect on the results of their learning processes (see Section 5.1.5)

- **LAS2peer** is a federated platform for distributed servers, allowing the distribution of community services in a peer-to-peer infrastructure (see Section 5.1.3). Integrating with LAS2peer, **MobSOS** is based on a resource-oriented architecture for providing community service success analytics (see Section 5.1.7)

- **i5Cloud** is based on a cloud model and provides a scalable and reliable solution for multimedia services, which can be used for a large range of devices (see Section 5.1.1)

- **SeViAnno** is a cloud service, which provides semantic video annotation functionalities. It represents a good Layers-related use case for supporting informal learning and an example for integration of different models and products (see Section 5.1.6)

\(^2\) National Institute of Standards and Technology (NIST)
• *Social Semantic Server* is a service framework for social semantic data (see Section 5.1.4 and deliverable D5.1)

• Tribal’s M-Learning Platform is another product based on a cloud computing model for providing mobile and multimedia cloud support (see Section 5.1.2)

### 3.3 Requirements Engineering

In parallel to the technology collection activity described in the previous section, which is mainly driven by the technical partners in Layers, there were several initiatives in the project led by design partners to elicit user requirements with end-user involvement. Software architectures are built based on functional and non-functional requirements. In Layers, the user requirements were elicited as a first step through various activities from end-user partners in the two Layers application clusters, i.e. healthcare and construction. These activities and related artifacts are sketched in Deliverable D1.1 in the Figure labeled “Interaction of Empiric and Design Work”. We reproduce this figure here as Figure 3. The knowledge of workflows in the SMEs represented in the application clusters is fed into the Layers design teams, which have been established after the Design Conference that was held in March 2013 in Helsinki (see D4.1 Section 4.3 for details). The design teams are creating and working on use cases, storyboards, and basic interactions using wireframes, different prototypes, and agile software development methods. Additionally, requirements that have been collected during the Application Partner Days (APDs; see D4.1 Section 4.2) in Leeds and Bremen and during user interviews, respectively, were also used in the Design Conference as starting point for the formation of design ideas. These are also reflected in the gathered architectural requirements.

The artifacts that were created as part of the activities illustrated in Figure 3 were “scanned” for potential requirements. Additionally the Layers DoW was included as a source of high-level requirements. Many non-functional requirements were gathered from there, as the technical work packages contained useful information about the desired infrastructure (e.g. scalable, federated, etc.). The DoW also contained two example scenarios from the construction and healthcare domains. As further input, the context cards produced from the APDs were used, with an example shown in Figure 4.

Figure 5 represents an example of a storyboard, representations of design ideas which resulted after the Helsinki Design Conference and around which the design teams were formed. The storyboards, together with further ideas (e.g. use cases) gathered from the design teams were used by the technical partners in the requirements elicitation phase. The requirements were collected in each Wiki page of the respective design team, numbered and added as requirements source to the dedicated Wiki page for the architectural requirements elicitation\(^3\). Further steps involved the refinement of the first set of requirements. The first formulated requirements from all sources were rephrased in order to obtain a consistent set of functional and non-functional requirements.

---

\(^3\) [http://htk.tlu.ee/layers/MW/index.php/Architectural_Requirements](http://htk.tlu.ee/layers/MW/index.php/Architectural_Requirements); cf. [34], p. 36
The architectural requirements collected and refined from the DoW, context cards, user stories, design ideas were split into functional and non-functional requirements and collected as tables using semantic properties into the Layers Wiki. In the end, there were 42 functional requirements and 26 non-functional requirements available. These are listed in Appendix C.

4 Reproduced from Deliverable D1.1 with permission by UIBK.
The functional requirements obtained were ingested into Requirements Bazaar\(^5\) \cite{12} \cite{13} \cite{14} (cf. Section 4.4.1), a tool developed by RWTH Aachen University in the context of the ROLE project. The Requirements Bazaar is a browser-based social software platform (see screenshot in Figure 6) for Social Requirements Engineering (SRE) addressing the challenge of a feedback cycle between users and developers in a social networking manner. Stakeholders from diverse Communities of Practice (CoPs) are brought together with service providers into an open, traceable process of collaborative requirements elicitation, negotiation, prioritization and realization. A vital communication between all stakeholders of an open source project is essential in this regard \cite{3} \cite{4}. The Bazaar aims at supporting all stakeholders in reaching their particular goals with a common base: CoPs in expressing their particular needs and negotiating realizations in an intuitive, community-aware manner; service providers in prioritizing requirements realizations for maximized impact.

The Requirements Bazaar was used in the requirements engineering step for a collective voting process, in order to achieve a ranking of the elicited functional requirements, as an input for the quality function deployment (see Section 3.4). Partners were asked to express their opinion through casting a vote on the most important requirements. The vote consisted of a like on a certain requirement. The voting options available for each requirement were “Like”, no action and “Dislike”. Through this collective process, all the existing requirements were rated, enabling the prioritization of requirements. The ranking was constructed by sorting the requirements list according to the scores obtained after the voting procedure. The most popular requirement (i.e. Activity Tracker) obtained a total number of 16 positive votes, while the last requirement in the hierarchy was listed with only 2 votes. A portion of the obtained prioritized list can be seen in Figure 6. Moreover, partners were also

\(^5\) \url{http://requirements-bazaar.org}
encouraged to comment on the requirements, in order to allow further refinement of the available requirements descriptions.

The resulting weighted list of requirements was end-user input to the House of Quality approach described in the next section.

Figure 6. Requirements Voting in Requirements Bazaar

3.4 Technology Assessment: House of Quality

As explained in the Layers architecture vision in Section 2, we will continuously integrate components into the overall architecture. While doing this, we need to be able to make controlled architectural decisions, which are informed by the actual needs—both functional and non-functional—of end-users. Therefore we chose to deploy a well-established methodology that will allow us to map technical features offered by new and existing components with end-user requirements in the context of use, which will typically be defined by one or more design teams. The general methodology we chose is called Quality Function Deployment (QFD) [5], and the particular instrument we adopted to map features and requirements is called House of Quality (HoQ) [10].

QFD is a methodology that aims to drive product design by customer requirements. Its instantiation is HoQ, a product development technique that follows the principles of QFD and has been originated in Japan in 1972 long before in the 1980s it was adopted by large U.S. firms such as Ford, Xerox and AT&T for their product development activities. The instrument allows identifying those parameters of a technology that are especially important taking into consideration the user requirements. In the case of the Layers integrated architecture, we use the HoQ to get a weighted list of functional requirements based on the respective demands and needs of the design team scenarios.
HoQ establishes a matrix of requirements coming from both the customer and the engineers designing the product. Using this approach, user requirements can be transformed into a weighted list of engineering characteristics that need to be met by the candidate products. It also supports the assessment of existing technologies in terms of how well they perform when meeting the user requirements.

Figure 7 shows an exemplary scheme of each HoQ. On the left side, customer requirements are entered one-per-row together with a weight calculated in user surveys. On the right existing products are rated by the end users, thus resulting in a market analysis. Engineering characteristics are entered on a column basis together with an improvement direction. At the bottom end of each column, improvement targets and the difficulty of reaching this target is recorded.

The House of Quality

In the next step, all engineering attributes are related with each other in the roof of the HoQ. Hereby positive or negative correlations in two graduations each are entered. That is, a plus is entered if on improvement of attribute A also attribute B is improved, and a minus is entered, if improvement of attribute A degrades attribute B at the same time.

The most important part of the HoQ methodology that also leads to the weighted engineering characteristics as output is setting the customer requirements in relation with the latter attributes. Hereby relations are rated in a numerical system with the higher number being higher related. A system is adopted that assigns strong relationships the value ‘9’, medium relationships a ‘3’ and weak relationships a ‘1’. Stronger relationships also lead to a stronger influence in the weight calculation at the bottom.
Another benefit of using HoQ is that it enables traceability of product attributes as for every weighted engineering characteristic the original user requirement can be traced in the matrix. In further steps, the output of one matrix may also be cascaded as input of a new one thus enabling traceability. For software products, further matrices may be applicable in terms of software modules within a broader infrastructure.

In the end, all the weights of customer requirements are charged against the product attributes according to their relationship factor. The output on the bottom is a list of weights for each product attribute that can then be incorporated in the product design. As described above, the results may be integrated into another HoQ matrix.

In Layers, a preliminary HoQ was built up to month 6 to test-drive the methodology. The user requirements elicited and then weighted through the voting process in the Requirements Bazaar were used as input for the left part of the HoQ. Pure non-functional requirements were instantiated for the technical part on the top. We found that the requirements were partially too general for enabling decision making using the Quality Function Deployment method. It was also observed that the result was not necessarily offering proper support to early prototypes and systems developed in the design teams. This was due to the usage of non-functional requirements as engineering characteristics in the first, preliminary HoQ.
Considering these identified issues, we finally decided to tune the methodology and use the functionalities of our existing architectural components as technical input; the concrete procedure is described in Section 5.4.1. Also, instead of generalizing the user requirements by merging them across design teams, we rather chose to focus on each design team perspective separately to get a clearer overview on what functionality is important there. By employing the same set of technical requirements as input for the columns of the design team HoQs, we manage to obtain a generalized view from an integrated architecture perspective.

To promote and explain the HoQ methodology and foster participation of the design teams, we developed a collaborative Web application based on Google Drive that allows multiple persons to jointly work on a HoQ, as shown in Figure 8. The tool is also available on the Google Chrome Store\(^6\). The methodology and the tool were both presented in the Graz Consortium Meeting as well as several online meetings with specific audiences, including the LDTF, WP6 and design team meetings. Further tutorials and training materials are available in the Layers Wiki\(^7\). The resulting HoQs evaluated in Section 5.4 were all created by LDTF members and together with their design teams.

We plan to seamlessly integrate the HoQ Web application into our portfolio of development tools. More specifically, the output of the Requirements Bazaar could be directly fed into the HoQ to allow a coherent workflow. Besides the overall usage for gathering requirements for the Layers prototype, the integration is also highly interesting from a research perspective: on one hand within the field of Requirements Engineering for studying how requirements gathering is evolving within communities by providing social and intuitive tools; on the other hand the collaborative approach involves many aspects within Web Engineering like social computing and shared editing \([15]\).

## 4 Open Development and Integration

### Process and Infrastructure

This section presents the instruments we have established and deployed during the first year in task T6.2, which runs throughout the project lifespan and aims to support the distributed process and platform for Layers software engineering and integration. Fuelled by the Layers architecture vision, the design of these instruments has been guided by analyzing the particular needs of the Layers software engineering process and the previous successful experiences with similar challenges in the ROLE project. All required instruments, in particular the decision making body (Layers Architecture Board; Section 4.1), the Layers Developer Task Force to propel and manage the distributed development workforce (Section 4.2), and the Open Developer Library as a versatile toolkit to facilitate internal and external developer communities (Section 4.4) are up and running ensuring a smooth progress in the coming project years.

\(^6\) [https://chrome.google.com/webstore/detail/house-of-quality/jembpnanfpbdeklibcogaafjolommfim](https://chrome.google.com/webstore/detail/house-of-quality/jembpnanfpbdeklibcogaafjolommfim)

4.1 Layers Architecture Board

In accordance with the DoW, the Layers Architecture Board was established and coordinated by WP6 as an instrument to make project-wide decisions on the overall Layers technical architecture.

Board Composition. At the Layers kick-off meeting in Barcelona it was decided that the Architecture Board should be composed of the following members:

- Core members: RWTH, TLU, TRIBAL. These partners have person months in task T6.2, which officially coordinates the Architecture Board.
- Other technical members: Aalto, TUG, HsKA, Nortal. These partners have significant person months in WP6 tasks.
- Design teams: Each design team shall be represented; currently these are Bits & Pieces, CAPTUS, PANDORA, and Sharing Turbine.

Decision Making. When making decisions in Architecture Board meetings, each of the above listed members has one vote, represented by a delegate, which shall be named at the beginning of the meeting. Decision making requires presence of the core members, and in total at least half of all the members must be represented. Meetings can be held physically or virtually using videoconferencing software.

Decision Support. To support the board members in decision making, decisions of relevance to the project-wide architecture shall be informed by Houses of Quality (HoQ), if applicable, e.g. by the ones created for the Layers design teams. A guide to HoQ-informed decision making can be found in Section 3.4.

Coordination. The Architecture Board has its space on the Layers Wiki. The minutes and decisions of all meetings are communicated on this space. Similar to the Layers work and administrative mailing lists, a mailing list for board-internal communication was created at layers-architecture-board@googlegroups.com.

Inaugural Meeting. The Architecture Board was formally established during its inaugural meeting at the Graz Consortium Meeting in June 2013. The main results of this meeting were:

- The Architecture Board established itself as a consulting instrument for the strategic implications of architectural choices for the Learning Layers consortium.
- The Architecture Board will create the necessary communication tools for internal and external communication such as an email list, wiki pages and a sub-domain for the developer taskforce in the Layers website.
- The Architecture Board supports the Learning Layers Developer Camp which was held in October 2013 in Aachen, Germany.
- The Architecture Board committed itself to facilitate the preparation of the integration meetings at M18 and M42.

---

8 http://htk.tlu.ee/layers/MW/index.php/Architecture_Board; cf. [34], p. 43
9 http://htk.tlu.ee/layers/MW/index.php/Architecture_Board__Inaugural_Meeting; cf. [34], p. 43
The Architecture Board agreed that the scalability of existing Layers component is sufficient for Year 1 and Year 2.

2nd Meeting. The 2nd Architecture Board meeting was held at the Paphos Consortium Meeting in September 2013. The main decisions were:

- The protocol of the inaugural meeting was signed off in unison.
- The set of components presented by RWTH were accepted in unison as the initial small-scale Layers architecture for Year 1. (These components are described in Section 5.1)
- The next Meeting will be held at the next consortium meeting in February 2014.

4.2 Layers Developer Taskforce

As part of the open development vision in WP6, the Layers Developer Taskforce (LDTF) was established to bundle developer workforce in Layers. To overcome the cold-start problem in the early project stages (e.g., scarcity of available user requirements, incompatible tools by the partners) LDTF has been given considerable autonomy in defining the short- to mid-term development agenda without necessarily involving the whole consortium.

The idea to form such a community of developers was based on previous successful experience with a similar taskforce in the EU-funded FP7 ROLE project. LTDF’s member affiliations are crosscutting all Design Teams and all of the partners that have tasks related to actual development.

Established at the Helsinki Design Conference in March 2013, bi-weekly online meetings have been taking place. Current members comprise Aalto (2), TUG (2), HsKA (2), PONT (1), RWTH (3), and TLU (1). In these meetings current activities are discussed among developers. The group is self-managed which means that there are no designated governing members. However, the task force and its meetings are coordinated by RWTH. The goal is to form an informal community of developers to exchange about recent technologies and help each other in concrete developmental tasks. To foster discussion, a mailing list was created that only addresses LDTF members. Furthermore, a Facebook group enables quick sharing of any kind of information like links to new libraries.

The LDTF had an active role in the requirements elicitation phase presented in Section 3.3, in synchronizing and discussing the developments from the design teams, exchanging of documentation and APIs and in supporting the initial compatibility and integration between the different prototypes developed across the project. At the consortium meeting in Graz, LDTF members organized a demo sprint to present existing tools and ideas of partners. The event was a good occasion to showcase existing systems already mentioned in the DoW (e.g. T4.3 Iterative prototyping 1 – individual interactions and T5.2 Emerging Semantics) to project partners, as well as in understanding the implementation details and capabilities of the systems.

---

10 [http://htk.tlu.ee/layers/MW/index.php/Architecture_Board_2nd_Meeting_(Paphos,_Cyprus)]; cf. [34], p. 45
11 [http://htk.tlu.ee/layers/MW/index.php/Tool_Demos_Graz]; cf. [34], p. 60
LDTF members also participated in generating the Houses of Quality for the individual design teams. Another success was the organization of the Layers Developer Camp 2013, which is detailed below.

4.3 Layers Developer Camp 2013

The 1st Layers Developer Camp 2013 was a one-week event held from September 30th to October 4th, 2013 at RWTH Aachen University. One part included a face-to-face LDTF meeting. The other part involved students interested in Layers technologies and use cases to promote them the project ideas and involve them in the open source development process.

The first day, the LDTF members internally presented recent developments concerning the design team prototypes, presented APIs and exchanged ideas upon possible integration of the tools. As an example, i5Cloud and Social Semantic Server APIs (see D5.1) were presented during the LDTF meeting. Based on the presentation, the possible benefits of using the systems with the design team prototypes were assessed. Different development techniques (RESTful Web Services, development of Widget applications) were also approached during the sessions.

On the second and third day students from RWTH Aachen were invited to participate in the event. Twenty students joined the Developer Camp, amongst others participants of lab courses held by the RWTH and student researchers. The camp was advertised as an event that presents some real-life Layers scenarios to be tackled with recent cloud, Web and mobile technologies. After initial presentations about the Layers project as well as Semantic and Web technologies, students were asked to form groups of 4-5 participants and develop a creative solution that incorporated the concept of “annotation” as a task. Students were encouraged to find innovative solutions for the “annotation” scenario in informal workplace learning and to reuse existing code from the project previously presented during the tutorials session. As a result, four individual prototypes were developed from both the construction and healthcare sectors. Though the code is of limited use due to the short development time, software ideas and parts of code are considered to be useful in future implementation tasks. Furthermore, the event was considered a first step in the open source initiative, i.e. in engaging third-party developers into the project and increasing the awareness around the Layers project.

Concerning development for Design Team prototypes, the Ach So! prototype’s video upload functionality was successfully linked to the backend provided by SeViAnno (for further details see D4.1 Section 5.2.4).

4.4 Open Developer Library

The Open Developer Library (ODevL) is a central concept on the way to an integrated architecture as it ties together all resources related to actual software development. It is the core instrument to achieve the Layers architecture vision of open source development and continuous integration on the technical level. It targets all aspects of software development, namely the planning, the implementation and finally the deployment and maintenance. As can be seen in Figure 9, it takes into consideration the input from the Open Design Library (see D2.1), as the Requirements Bazaar is used for Social Requirements Engineering. That
is, we aim towards a tight integration of ODevL with the Open Design Library (ODL) by means of the Requirements Bazaar. Issues are reflected in the JIRA issue tracker that is itself linked with SourceForge as open source code repository. Jenkins is a continuous integration tool allowing automated tests and nightly builds. The respective parts are explained in detail in the next sections.

The ODevL’s main visible part is a website that provides a one-stop-shop for all these development related activities. It is available at [http://developer.learning-layers.eu](http://developer.learning-layers.eu). Its main goal is to focus the attention of both internal and external developers on all the tools needed for the development process. Next to the tools mentioned above, its content comprises a Downloads section, an API library and further information on how to take part in the development process of the Layers project.

![Open Developer Library Overview](image)

ODevL’s user base consists of three groups:

A. Users who are interested in the technical side of the project like getting familiar with latest technologies for informal learning. Therefore, a download section is provided. In this context, a Google Play Store account was set up to allow us uploading apps so that they are available over the official Android Market, which increases the target audience.

B. App Developers (API consumers) who are interested in developing apps that leverage APIs provided by Layers backend technologies. We provide API documentation and demo apps to help developers in getting started.

C. Infrastructure Developers (API producers) who are interested in extending the Layers backend API by developing new tools.

So far, the download section contains the AnViAnno app for mobile video annotation provided by RWTH. The app is available through Android’s Play Store and can also be
found in Learning Layers’ Google Play listing. Apps offered in app stores are linked directly to the respective Open Developer Library website for support purposes. API documentations currently include detailed instructions for accessing the user management and cloud storage facilities provided by i5Cloud. The Social Semantic Server section features detailed documentation on using its client-side libraries. JIRA and Requirements Bazaar participation was mainly driven by Developer Camp activities (see Section 4.3).

4.4.1 Requirements Bazaar

As described in D1.1, most requirements elicitation efforts employed in Year 1 only scale for small and local populations of end-users involved in the participatory development process. Taking into account the Layers architecture vision (cf. Section 2) and earlier experiences in the ROLE project, we employ Requirements Bazaar as a Web-based tool for scaling up and analyzing a more distributed requirements engineering process involving Web 2.0-style end-user participation in the project.

This section is mainly based on our paper “Requirements Bazaar: Social Requirements Engineering for Community-Driven Innovation”[14], which includes a compact description of the Requirements Bazaar. Requirements Bazaar was originally developed in the EU FP7 IP ROLE (Responsive Open Learning Environments; http://role-project.eu) employed in the Requirements Engineering process of the project. However, instead of the screenshots in the original paper we show screenshots relevant to the requirements engineering process in Layers (cf. Section 3.3). Then, we discuss first feedback collected from the Layers requirements engineering process and serving as pointer to future work.

The Requirements Bazaar was accepted as a demo paper at the 21st IEEE International Conference on Requirements Engineering (RE 2013)[14], where it has won the Best Demo Audience Award after gathering the most positive votes from the conference participants.

The innovation potential of niche communities often remains inaccessible to service providers due to a lack of awareness and effective negotiation between these two groups. Requirements Bazaar, a browser-based social software for Social Requirements Engineering (SRE) [12][13], aims at bringing together communities and service providers into such a negotiation process. Communities should be supported to express and trace their requirements and eventually receive a realization. Service providers should be supported in discovering relevant innovative requirements to maximize impact with a realization. Thus, Requirements Bazaar focuses on four aspects: requirements specification, a workflow for co-creation, workspace integration and personalizable requirements prioritization.

4.4.1.1 Requirements Specification

In Requirements Bazaar, a requirements specification is an aggregate of basic metadata, an arbitrary number of artifacts (annotated images, user stories, etc.) and traceable means for social participation (vote, share, comment, follow, contribute artifact, commit to lead, commit to develop, test prototype/ solution) being part of the Requirements Bazaar workflow (cf. next section). All information on a particular requirement along with its community participants and role-dependent participation options are condensed on a requirement page (cf. Figure 10).
4.4.1.2 Co-Creation Workflow

The 4-phase co-creation workflow of Requirements Bazaar aims to continuously integrate communities into the entire service development process. It allows phase-dependent co-creation operations: reporting new requirements, refining by adding artifacts or contributing to discussion, negotiating by voting or commenting, providing/testing a prototype/solution and acknowledging a solution. In the initial Idea Generation (1) phase a stakeholder reports an unfulfilled need in the form of an open requirement with initial metadata and artifacts. In the Idea Selection (2) phase refinement and negotiation among stakeholders takes place, until one service provider commits to take the lead for realization. The requirement is then assigned, thus transitioning to the Idea Realization (3) phase. Refinement and negotiation continue. The leader invites collaborators and links testable prototypes. Once realized and sufficiently negotiated and tested, the requirement enters the Idea Release (4) phase, where a final solution is acknowledged, possibly leading to new requirements.

4.4.1.3 Workspace Integration

To lower entry barriers, Requirements Bazaar provides different means to integrate with end-user and developer workspaces. Ready-to-use dialogs for reporting application-specific requirements can be easily integrated into arbitrary Web applications by instrumentation with a small Web plug-in. Furthermore, stakeholders can monitor requirements by subscribing to event notifications. For assigned requirements, Requirements Bazaar supports bi-directional integration with issue trackers in terms of synchronization, thus providing
traceability beyond its borders. We demonstrate such integration with an Atlassian JIRA issue tracker (cf. Section 4.4.2).

4.4.1.4 Personalizable Requirements Prioritization

Requirements Bazaar supports requirements prioritization with a modular extensible requirements ranking framework. A linear weighted combination of multiple scoring providers serves as foundation for computing one normalized requirement relevance ranking score. A requirements discovery page (cf. Figure 11) serves requirements ranking lists along with a break-down of influences from individual scoring providers.

Scoring factors are assigned to the categories user rating (e.g. up/down-voting), user behavior (e.g. monitored activity in Requirements Bazaar, external service usage) and community belonging (e.g. relation to stakeholders with similar interests). Category weights are configurable with respect to personal preferences of scoring factor importance. Reasonable initial default weight configurations are given (user rating only, equal weight distribution over all categories).

![Figure 11. Requirements Bazaar – Requirements Discovery Page](image)

4.4.1.5 Initial Usage Feedback & Future Work in Layers

Requirements Bazaar has been continuously used in Layers requirements engineering, until now mainly for the collection of architectural requirements. The results of initial Requirements Bazaar usage served as input for the HoQ approach widely pursued in the project. Furthermore, feedback on system usage was reported as a set of improvement suggestions. In particular the voting process was suggested to become possible and more clearly visible on the Requirements Discovery Page. In the original version this was intentionally avoided to force users into reading the full description of a requirement before allowing them to vote. Furthermore, users suggested to limit the number of active votes per
user as one means to avoid the “rich-get-richer” effect found in voting data. Another means was indirect voting by pairwise comparisons of requirements in a gamified fashion, similar to elements of the Analytic Hierarchy Process [16]. Furthermore, community and project awareness should be supported by adding community or project-centered requirements discovery pages and tagging, thus allowing for filtering for those requirements relevant for the respective context. Finally, the interface of the discovery page involved a set of minor UI glitches to be fixed.

In future, the project intends to employ Requirements Bazaar for collecting requirements on tools developed in the context of the Layers Design Teams. For this purpose, we intend to continue development of Requirements Bazaar towards the improvements derived from feedback in Layers.

The Requirements Bazaar is provided by and maintained at RWTH and is reachable at http://requirements-bazaar.org.

### 4.4.2 Issue Tracking

The Layers ODevL employs Atlassian JIRA\(^\text{12}\) as its main issue tracker. JIRA is currently one of the most used issue trackers world-wide. Although JIRA is a commercial product, Atlassian offers free licensing to OSS (Open Source Software) projects or organizations, providing their software under an OSI (Open Source Initiative) approved license\(^\text{13}\). A prominent example is the Apache Foundation, using JIRA issue tracking for many of their projects under Apache License 2.0.

Since the above conditions are also fulfilled for Layers, the project was granted a free license for JIRA and a list of commercial plug-ins without any additional costs besides public hosting of an installation. At the current stage we maintain a public JIRA installation including an SVN plug-in for the integration with the Layers SourceForge SVN repository and the GreenHopper plug-in for managing Agile Development with the help of Kanban boards. Experience from previous EU projects, e.g. the ROLE project proved JIRA’s usefulness in terms of efficient development work management and quality assurance. Project partners can collect issues and prioritize, discuss and trace solutions and organize collaborative Agile Development work. Thereby, issues cannot only be added and discussed directly via the JIRA Web frontend, but also via the Requirements Bazaar once a developer committed to realizing a prioritized requirement. JIRA’s additional analytics support will serve as tool for monitoring and reporting on development progress. Currently we maintain one overall Layers project in JIRA, which however can be further structured with the concept of components, thus allowing issue tracking for specific software components developed in Layers, e.g. Social Semantic Server, i5Cloud, etc.

The Layers JIRA installation is provided by and maintained at RWTH and is reachable at http://layers.dbis.rwth-aachen.de/jira.

\(^{12}\) [https://www.atlassian.com/software/jira](https://www.atlassian.com/software/jira)

\(^{13}\) [http://opensource.org/licenses](http://opensource.org/licenses)
4.4.3 Source Code Management

Source code for all open source development in the Learning Layers project is hosted in SourceForge, a leading platform for open source software development and distribution. The SourceForge project contains general information about Learning Layers, a Wiki for code documentation, user management for developers and a SVN code repository for version control. As depicted in Figure 9, having the open source code hosted in a well-known site makes Learning Layers prototypes more attractive and trustworthy for open source developers. Moreover, the usage of SourceForge allows official downloads of the latest code versions and usage statistics of project code. Awareness of code commits serves technical integration as it allows developers to recognize areas of active development and those developers that have expertise on each part of Layers infrastructure. Having one common repository for source code also helps individual developers from the project to stay aware about developments in other work packages and can prove very useful for the integration phase of the various prototypes.

The projects currently hosted at Layers SourceForge repository are design teams prototypes (Ach So!, Non-obvious, AnViAnno), systems (Social Semantic Server) and prototypes from the Developer Camp 2013. Including the participants to the Developer Camp 2013, there are currently 24 registered developers in the system. As of end of October 2013, there have been more than 250 code commits in the SourceForge repository.

The structure of the SVN SourceForge repository is presented in Appendix B, Figure 23. The productive versions of the code from different partners are located in the main line (trunk) of the SVN code management system. For other code development purposes, branches are used. Here, parallel versions of the Layers systems or new prototypes which are still under development can be created. As an example, a special branch named “devcamp13” has been created in order to host the code developed for the Layers Developer Camp 2013.

Regarding the main line hosting the productive Layers systems, the overall increase in the amount of code is depicted in Figure 12. The code production shows a sharp increase during the Developer Camp early October, and the total code base has reached more than 220,000 lines of code by the end of Year 1.

![Figure 12. Lines of Code in Learning Layers SVN Repository Trunk](http://Learning-Layers.eu)
4.4.3.1 SocialSemanticServer

Within the Learning Layers project, the Social Semantic Server (SSS) (for detailed information, see D5.1) was made Open Source to provide projects partners as well as interested Open Source developers the possibility to actively contribute to respective code. A short description of the Social Semantic Server can be found in Section 5.1.4. To make Open Source development most convenient for developers, the Apache License 2.0 was chosen. Moreover, a respective license was selected to be able to link the SSS code to other software products as well as to leave it up to third-party developers whether they publish changed SSS code under a different license if needed.

Currently, ongoing developments with regard to the SSS code are being committed to the Learning Layers SourceForge repository 14.

Basically, the repository is separated into client-side JavaScript library projects providing reference implementations for working with the SSS’s REST- and WebSocket API respectively and server-side implementations. Following client-side libraries are available: JSUtilities containing common JavaScript method wrappers and constants and SSSClientInterfaceGlobals containing common methods and constants with regard to SSS communication issues used in SSSClientInterfaceREST and SSSClientInterfaceWebSocket libraries, which in turn provide convenient connectors to the SSS’s service API’s. The server-side Java Maven project consists of several sub-projects responsible for service implementations (ss-serv), core functionalities (ss-core), interfaces to the SSS (ss-adapter), global data types (ss-datatypes) and utility, logging and configuration modules (ss-util, ss-log, ss-conf).

As certain third-party libraries are required to execute SSS code, it is planned to replace respective GPL 2.0-licensed libraries / implementations by implementations following the Apache License 2.0 as well. Moreover, to attract Open Source developers to contribute to or (re-) use SSS code, certain efforts within the Learning Layers Developers Task Force Group have already been made (e.g. the Learning Layers Developer Camp) and are planned to be continued in the second year of the project.

4.4.3.2 AnViAnno

AnViAnno is an Android prototype for semantic video annotation, developed by the ACIS Group, RWTH Aachen University. It is the Android mobile version of the SeViAnno prototype (presented in Section 5.1.6). The Android app can record videos, annotate them locally, upload and transcode videos using the i5Cloud (see Section 5.1.1). It can capture videos on-site and replay the annotated videos on the mobile device. For this purpose, AnViAnno uses the local storage for the videos which are not uploaded to a server and an on-memory SQL Lite database for the annotations. The app was released under the Apache 2.0 license and represents a starting point for the Ach So! prototype, developed for the CAPTUS design team and described in the D4.1 deliverable.

The AnViAnno code was released for Android 2.2 (Android API level 8). The AnViAnno project contains source code, a README file, Android metadata and layout definitions, logos and images and third party dependencies.

4.4.3.3 AchSo

AchSo contains source code for the Android App ‘Ach so!’ developed by Aalto for the CAPTUS design team (for details see D4.1). ‘Ach so!’ is a mobile video recorder app for construction cluster, where users record short situational clips under genres “Problem”, “Problem solved”, “Tricks of trade” and “Don’t do this”. These clips can be associated to a specific place (filming location) or to a specific item by reading a QR-tag or a barcode. This allows other users to quickly view if there are any warnings or instructions associated to that place or item.

The clips recorded in ‘Ach so!’ can be annotated with semantic annotations. ‘Ach So!’ video clips and MPEG-7\textsuperscript{15}-compatible semantic annotations are stored in RWTH’s i5Cloud system.

The codebase contains project file for building the app in Android Developer Tools, including source code, Android metadata, layout definitions, image files and open source JAR-libraries necessary for project. ‘Ach so!’ is compatible with Android SDK versions 11 (Ice-Cream Sandwich) and upwards. The code is published under Apache 2.0 open source license.

4.4.3.4 Non-obvious

Non-obvious is a prototype for reflection tool developed by Aalto for design team CAPTUS (for details see D4.1). The purpose of Non-obvious is to allow written reflection with minimal effort wasted on describing the obvious aspects of the experience. Non-obvious is a simple browser-based interface for asking reflective notions from user about a given theme and it is planned for use in CAPTUS sustainable construction exhibition next to exhibition artifacts. One of the design features of Non-obvious is that the reflection environment adjusts its colors to harmonize with the background image, supporting reflection on a photo or an image.

The source code for Non-obvious has a Web app client built with HTML5 and jQuery and required stylesheets, instructions and a set of background images. The client is not connected to server backend, but the reflections are persistent and stored to browser’s DataStorage. The implementation of storages is modular and new storage interfaces to Layers infrastructure can be easily built.

Non-obvious is published under Apache 2.0 license.

4.4.4 Continuous Integration

Realizing the vision of continuous integration in WP6, the Layers ODevL employs Jenkins\textsuperscript{16} as its main tool. Jenkins is an award-winning extendable open source continuous integration server which is widely used by both companies and organizations.

In the context of ODevL Jenkins is mainly integrated with the SVN repository maintained at SourceForge (cf. Section 4.4.3) for several purposes. Main purpose is the automated and traceable production of nightly builds directly from the contents of the Layers SourceForge SVN repository. The current configuration of the Layers Jenkins installation includes one

\textsuperscript{15} http://mpeg.chiariglione.org/standards/mpeg-7

\textsuperscript{16} http://jenkins-ci.org
job, which takes an Apache Maven master build file from the Layers SourceForge SVN as input for automated builds of a hierarchy of sub-modules, including the execution of automated unit tests, if available. However, we foresee the configuration of additional jobs in later project stages, e.g. automated deploys and availability tests of Layers infrastructure elements or the breakdown of the current master-job into several distinct jobs for individual and clearly separable software components, e.g. the Social Semantic Server, the I5 cloud and a set of applications developed within the project context. With a rich ecosystem of available Jenkins extensions and the possibility to execute arbitrary native command line scripts, we are even prepared for integrating other classes of jobs besides Apache Maven-based builds, e.g. Mobile app builds.

Jenkins provides a public interface for downloading build artifacts, tracing changes in the source code repository and accessing build and test reports including respective visualizations. Furthermore, Jenkins provides rich means of notification, e.g. in case of failed jobs. Currently, the Layers installation is configured to push any job failure notifications to a Layers developer mailing list. With clearer definitions of responsibility becoming clearer in future project stages, notifications can be directed to relevant developers. Jenkins will furthermore serve the purpose of tagging official releases of Layers components in SVN. As such, Jenkins serves as a quality assurance and tracing tool.

The Layers Jenkins installation is reachable at http://layers.dbis.rwth-aachen.de/jenkins.

5 Initial Layers Architecture

In the first year, the main objective of task T6.3 was to deploy an initial architecture for fast, small-scale deployment. The components of this architecture are described in Section 5.1. The candidate components were surveyed in the technology survey (Section 3). Requirements, which were elicited with the design teams (Sections 3.3 and 5.4), along with the features extracted from these components (Section 5.4.1), form the cornerstones for the House of Quality (HoQ) approach outlined in Section 3.4. For each design team a HoQ was built to expose the matching between requirements and features to make sure that the selected components are sufficient to satisfy the end-user requirements. The Architecture Board, based on these available bits of information, signed off the Year 1 architecture presented in this section. As expected—and evident from Section 5.2—the components offer many interfaces for integration which are at present not fully integrated for the small-scale architecture, although integration successes can already be reported (e.g. following the 1st Developer Camp; Section 4.3). The bulk of the integration work lies ahead of WP6 in its endeavor to provide a flexible architecture for large-scale deployment in the application clusters.

The Layers project is about scaling technology for informal learning. To this end, D7.1 outlines several initiatives and strategies for employing the application clusters as instruments for scaling. One important dimension of scaling—that needs to be considered as complementary to these strategies—is the development and deployment of technologies that scale up to different uses in informal learning, to increasing numbers of users and increasing volumes of data. Many of the existing solutions for informal learning do not scale because the scaling technology is missing in the backend. WP6 strives to overcome these gaps through the strong emphasis on sound Web Analytics and Web Engineering (cf. Figure 1)
methods in Layers architecture work, particularly as the mobile world and the Web are converging to a Mobile Web.

5.1 Architecture Components

Figure 13 provides an abstract overview of our initial architecture for fast small-scale deployment and its components.

Principally, the architecture is divided in three layers: a Cloud Service Layer, an Application Service Layer and a Service Access Layer. The Cloud Service Layer involves services for managing and controlling up-scaling, as well as additional support services shared among applications and especially suited for the execution in a cloud environment, such as shared data storage, media transcoding. The Application Service Layer involves actual application service logic as well as frameworks to develop new services and collect service usage and survey data. The Service Access Layer will consist of a Layers Adapter as described in the DoW to providing transparent access to the Layers service ecosystem for Layers applications. The Layers Adapter symbol has a dashed outline to indicate that it part of future activities in WP6.

The architecture includes two Cloud computing environments, spanning both Cloud and Application Service Layers: i5Cloud (RWTH; cf. Section 5.1.1) and m-learning (Tribal; cf. Section 5.1.2). Both environments provide cloud services for managing automatic and elastic upscaling either with new instances on local machines or in external commercial cloud
provider platforms. Furthermore, both environments comprise application services and platforms powering Layers client applications.

Both i5Cloud and m-learning expose a variety of different service Application Programming Interfaces (APIs) to be bundled in a Layers Adapter for convenient, unified and transparent use on the Service Access Layer in later project stages.

Besides the services on the Cloud Service Layer, each i5Cloud instance comprises three different systems: the Lightweight Application Server (LAS) and its recent extension LAS2peer enabling a distributed, federated LAS Peer-to-Peer infrastructure (cf. Section 5.1.3), the Social Semantic Server (cf. Section 5.1.4), and the ROLE SDK providing services for the design of Responsive Open Learning Environments (ROLES) (cf. Section 5.1.5). A set of Semantic Video Annotation (SeViAnno) Services is described in detail in Section 5.1.6. Furthermore, i5Cloud is prepared to incorporate MobSOS Usage Monitoring and accompanying LAS services as integrated analytics tools (cf. Section 5.1.7) supporting formative evaluation in Layers. In the following sections, we provide all infrastructure components and their particular purpose in more detail.

In terms of scaling, the architecture as outlined in Figure 13 includes components that scale on different layers:

- The cloud solutions (i5Cloud, Tribal m-learning platform) scale on the Cloud Service Layer;
- RESTful Web Services offered by all of the components support scaling on the Application Service Layer; and
- HTML5/JavaScript/CSS based front-end solutions that exploit this architecture will scale on the Service Access Layer.

### 5.1.1 i5Cloud

i5Cloud is a cloud solution that has first been developed by the ACIS group at RWTH in the context of the Excellence Initiative of German National Science Foundation (DFG) within the research cluster Ultra-High-Speed Mobile Information and Communication (UMIC)\(^{17}\). Its purpose is to provide a hybrid cloud for mobile and multimedia services [17]. i5Cloud provides basic infrastructure services for cloud computing which include user management, a virtual machine environment as well as cloud storage. For Layers, it was migrated to the industry-leading OpenStack platform that itself is deployed on a Sun Fire server within RWTH’s network. It represents a complete stack cloud solution, providing the infrastructure and platform for different multimedia related services. It uses a hybrid cloud computing strategy, i.e. it takes advantage of in-house hardware infrastructure mostly available in companies or institutions. In case of a cloud burst, where suddenly due to high user demand more resources are needed, i5Cloud can automatically allocate resources on external cloud providers such as Amazon EC2 [18]. By using the hybrid approach, the i5Cloud has appropriate security characteristics for sensitive business data in the competitive SME environment.

---

\(^{17}\) http://www.umic.rwth-aachen.de
Above the infrastructure layer, i5Cloud hosts various services for mobile cloud computing. The most notable service is the cloud video transcoder, a service that allows users to upload any videos and let them be transcoded to formats easily consumable by most mobile operating systems like Android and iOS.

As presented in Figure 14, the i5Cloud architecture is composed of three layers: infrastructure, platform and multimedia service layer.

The Infrastructure Layer contains storage, compute, and basic software infrastructure. The Platform Layer represents the middle layer of the i5Cloud. The resource manager elastically scales up and down active computing nodes (virtual instances) according to the request demand. In addition to computation and storage services, the i5Cloud platform also provides services for data streaming from the cloud to clients and vice versa. Media streaming is achieved by using standard software such as FFmpeg\(^{18}\) and Wowza2 servers\(^{19}\), and text-based data (or metadata) is streamed using OpenFire\(^{20}\) XMPP (Extensible Messaging and Presence Protocol\(^{21}\)) servers.

Finally, the Service Layer delivers mobile services to users and communities. Since the i5Cloud is an in-house developed software by RWTH, it is fully available for adaptation in Layers.

\(^{18}\) http://www.ffmpeg.org/
\(^{19}\) http://www.wowza.com/
\(^{20}\) http://www.igniterealtime.org/projects/openfire/
\(^{21}\) http://xmpp.org/
5.1.2 Tribal M-Learning Platform

The m-learning Platform developed by TRIBAL is a cloud (both public and private) deployed content repository allowing mobile users to browse, select and download a range of learning materials e.g. structured learning (courses), e-books, PDFs, videos (download and/or stream) and podcasts.

It has been (and continues to be) developed using open source technologies such as PHP, MySQL, etc. with the current live version deployed to Amazon servers. Architecturally, it has been designed to operate under a multi-tenant infrastructure meaning multiple accounts being serviced under one deployment.

The m-learning platform primary function is to deliver “learning content” to mobile users:
Formal - The creation of structured learning courses using HTML5 technologies. All interactions with the content by the mobile user is tracked and stored locally on the device in preparation for later transmission to the central content repository.
Informal - This is content which is distributed by the online repository but unlike the “formal” content does not offer any structure or tracking. Such content can include eBooks, videos and podcasts.

The architecture for the M-Learning platform is presented in Figure 15. The diagram also depicts the workflows for accessing and downloading content via the REST interfaces of the platform. The mobile content is referred to as packages. A package generally represents content that will be available on the device; this content can be anything, e.g. a course. Content developers that are managed by other users, which have the content manager role, develop courses. The content manager may upload several different package files and attach them to a package.

Features (which relate to LAYERS requirements) of the m-learning platform include:

- Create and publish news feeds to mobile users.
- Formal and informal learning content support
- Server streaming of uploaded content for mobile devices
- All learning content once downloaded to the mobile device can operate in an offline mode.
- Extensive tracking capabilities for formal learning content.
- Creation / definition of user communities allowing the account admin to target specific content to specific communities (i.e. mobile user grouping).
- Learning Analytics capability to report on mobile user's experience and progress.
- Multiple metadata association (tagging) for packages, package files and users.
- Data tracking for storing information sent by mobile applications (e.g. user, object id, timestamp, source, etc.)
- Video streaming and transcoding

The m-learning platform has an extensive (RESTful) API. This allows third party development companies to fully integrate their own propriety systems into it. For a full
Among the characteristics relevant for the Layers project, M-Learning has a multi-tenant architecture meaning each partner (Medical center such as GP practice, Construction learning Centre) can operate and administer their own account with ease. It features a cross platform learning content using HTML technologies and a centrally deployed to a cloud (n-tier architecture) environment, maintained by Tribal.

As part of the Layers project, further development of the m-learning platform consists from an online content authoring tool for the creation of structured learning content, a learning portfolio in line with the Layers requirements and networking to allow learners/users to network using their mobile device.

5.1.3 LAS/LAS2peer

The Lightweight Application Server (LAS) [19] is a community service middleware developed at RWTH. Its main features are (1) the support for managing a large number of users and their affiliations with multiple Communities of Practice [1]; (2) the extensible support for developing and hosting of a variety of Web services supporting the particular online practices of participating communities; and (3) a community-aware fine-grained access control management for both service functionality and resources shared among users, communities and services in mash-up end-user applications.

The main focus of LAS is rather on server-side aspects than on front-end generation facilities. Thus, LAS is much more lightweight and easier to use and configure than application servers like Tomcat or other J2EE application servers. The main difference to such frameworks is that LAS tries to offer service programmers as much flexibility as
Figure 16 provides an overview of the LAS architecture. At its core, the LAS Java API allows arbitrary extensions by the three basic element types of services, components and connectors. Services define the actual functionality of LAS which is publicly exposed to LAS-powered client applications or to other services, thus enabling the development of higher-level services from lower-level ones. Currently, there exist over 50 different LAS services for various purposes and communities. Among the top-used services are a set of MPEG-7 services powering SeViAnno (cf. Section 5.1.6) and set of services powering the Requirements Bazaar (cf. Section 4.4.1). Furthermore, LAS includes a set of core services realizing user and community management, security management and LAS element management. Another set of MobSOS services supports the controlled access to and management of monitored service usage and online survey data. These services are mainly dedicated to the analysis and evaluation of service success (cf. Section 5.1.7 for details), thus being a potential tool supporting formative evaluation in Y2 of the project. LAS components encapsulate functionality for common background tasks recurring in multiple services, e.g. the access to databases. As such, their functionality is not publicly exposed to clients. Connectors (and respective connector clients) realize the actual communication with LAS and its services using particular protocols and service access paradigms. Connector implementations exist for various protocols and paradigms, among them a proprietary LAS protocol based on HTTP, but also standards such as REST, Jabber RPC and SOAP. Connector client implementations exist for a variety of programming and runtime environments, e.g. for Java, Objective C, JavaScript, thus enabling the development of LAS-powered applications for both mobile and Web applications. LAS already fulfills the requirements for a “fast and flexible deployment” of services in a “service-based [...] technology platform that empowers key stakeholders to build additional services”. However, the purely client-server-based approach of LAS rather reflects
“monolithic in-house installations that lack flexibility for inter-SME networking in response to fast changing environments” and is as such not suited for “fast and flexible deployment in a networked SME setting with heterogeneous infrastructural requirements and conditions” (cf. DoW, Sections 1.1.1 & 1.1.2, pp. 8-9). Furthermore, LAS is not available as open source software yet.

Project requirements thus emphasized the need for “a service-based open source technology platform” that “allows for delivering novel ways of cloud-based” information services in a workplace learning setting. Furthermore, this setting “calls for a lightweight architecture that can be deployed flexibly and in a scalable way while supporting custom privacy and security standards”. Especially the interaction of users and communities within and across SME calls for “high standards of quality of service, privacy and security”.

Since federation is one of the means to cope with scalability challenges in the Layers project, LAS was recently extended towards LAS2peer, an open source framework for creating federated networks of distributed LAS servers allowing the distribution of community services in a peer-to-peer infrastructure. Previous experiences with LAS were taken into account for making service development and deployment even more flexible in LAS2peer. Most of the successful LAS concepts were adopted, however streamlined where possible and adapted to the new P2P paradigm, where necessary. Figure 17 shows the distributed, federated architecture of LAS2peer. The system consists of nodes which are loosely connected via message exchange and shared data storage provided by the underlying P2P protocol. Different kinds of agents (end-users, communities, services, monitoring agents, etc.) register to particular nodes to participate in the overall P2P network communication and are assigned a numeric unique identifier. In principal, any node can be configured as an own LAS instance, thereby hosting its own set of services freely configurable by its maintainer.

Figure 17. The Distributed and Federated LAS2peer Architecture
One particular specialty in the LAS2peer network is the transparent access to the entirety of all services hosted at individual nodes. All services available in the entire network can be accessed from any arbitrary node without explicit knowledge on which node a particular service resides. Transferring this concept to a scenario of multiple networked SMEs, each SME can flexibly develop, deploy and maintain its own services without the need to interfere with other SMEs and at the same time access all services provided by other SMEs participating in the network.

Security and privacy were fundamental concepts already taken into account during the LAS2peer design process, explicitly referring to the requirements stated in the Layers DoW. For the purposes of access control to shared data storage as well as secure end-to-end encrypted communication, each agent owns a public/private key pair for asymmetric encryption. Access to any message and any data envelope by any agent requires explicit checks, which are automatically performed by LAS2peer. Developers have programmatic means to control access to messages and data envelopes by means of cryptographic key management.

The LAS connector concept was adopted in LAS2peer to provide access points for client applications outside the P2P network structure, thus enabling the transparent access to all services hosted by the LAS2peer network via non-P2P APIs. The access to LAS2peer services from browser applications is already working with the proprietary HTTP-based LAS API.

Additionally, we recently completed an implementation of a concept for the collection and central provision of distributed node monitoring data (including service usage data) in LAS2peer networks (cf. Section 5.1.7), which is dedicated to the analysis and evaluation of service success and thus serves as potential tool for formative evaluation (see D1.1) in the project. Also here, explicit means for protecting the privacy of both users and developers were already taken during the design phase and are now implemented.

Altogether, LAS2peer combines the standard accessibility of Web application servers, the robustness, fault-tolerance and scalability of peer-to-peer networks, and a maximum of security and access control, thus contributing to the scalability, privacy and security requirements essential to project success.

LAS2peer code and basic documentation is licensed under an MIT license and available as open source project hosted on GitHub: https://github.com/rwth-acis/las2peer. Further development of LAS2peer is supported by and documented in the Layers JIRA (cf. Section 4.4.2).

Regarding the future research perspective, we plan to investigate applicability, benefits and shortcomings of LAS2peer community service delivery and analytics in a setting of distributed individuals and organizations (e.g. SME, vocational schools). Regarding development, we will make LAS2peer services accessible via RESTful APIs, in particular with respect to uniform bundling of RESTful APIs of Layers services with the planned Layers Adapter. Furthermore, we need to port and adapt some of the existing LAS services relevant to Layers to LAS2peer. Other future work includes the extension and improvement of online documentation on how to use and program with LAS2peer in order to lower the entry barriers to service development in open source developer communities.
5.1.4 Social Semantic Server

The Social Semantic Server (SSS; see D5.1, Section 5.2 for more details) is a Java-based Open Source service framework for social semantic data. Based on Artifact Actor Networks (AANs), services of the SSS provide the possibility to create, maintain, exploit and emerge relationships between users and digital resources. This in turn allows the user of an application using the REST-based services of the SSS to, e.g. retrieve recommended metadata for digital resources, use digital artifacts other users were interested in or have being working on or share his own knowledge with other users. Hence, the SSS allows for scaffolding (informal) learning as well as supporting meaning making for / between individuals. By the integration of the SSS into the i5Cloud infrastructure, the SSS is being made ready to scale with growing amounts of users and service requests.

With regard to the Learning Layers project, the SSS represents the back-end framework establishing the Social Semantic Network (SSN) interrelating users with digital / physical artifacts so that links between respective entities can be exploited to first, infer new knowledge from within a given context and second, emerge the structure of the underlying network. In turn, SSS services working on the spanned network provide several ways to work with given knowledge structures. Hence, the SSS enables design team application prototypes to implement certain functionalities relying on social relationships of given entities of several types (e.g. users, text documents, pictures, videos, links, etc.) and semantic descriptions of users and digital / physical artifacts.

Basically, the SSS is used for all kinds of metadata annotations, structuring of knowledge in access-restricted hierarchical collections or learning episodes, sharing and discussing of knowledge between users, searching for knowledge artifacts and users with the help of (recommended) metadata and content-based keywords or inferring and providing new insights upon given entity relationships / traces of entity usage and automatically created user and artifact models.

The underlying service infrastructure is made to handle multiple parallel requests to the SSS services taking into account performance and memory issues as well. The architecture of the SSS fosters extendibility, separation of functional responsibilities into loose coupled modules as well as the separation of services from core functions, client access and database layers. Additionally, an Apache WebDAV file repository and an Apache Solr index are integrated via respective services. The actual database backend gets represented by a Virtuoso Triple Store instance responsible for storing and querying RDF-based data structures.

5.1.5 ROLE SDK

Responsive Open Learning Environments (ROLE)22 was an EU funded FP7 IP project centered on the idea of self-regulated learning. The project consortium comprised 16 research groups from six EU countries and China. ROLE goals were about supporting teachers in developing open personal learning environments, where students can gather

22 http://www.role-project.eu/
useful resources and tools, plan their learning process, learn, collaborate, reflect on the results of their learning processes, etc.

ROLE makes use of the concept of *Personal Learning Environments (PLE)* [20]. The ROLE Software Development Kit (SDK) developed a semantic model, making use of the ROLE ontology. The semantic model relies on the concept of spaces [21]. Users can create spaces, where they can manage learning content and collaborate with other users. A space can host a bundle of Web widgets, contains a list of participants and enables them to communicate and collaborate. Moreover, for supporting complex applications, the widgets can exchange information across browsers by using the inter-widget communication mechanisms developed within the ROLE SDK.

Figure 18. The ROLE Infrastructure

ROLE Infrastructure (cf. Figure 18) is focused on responsiveness and life-long learning. The infrastructure is oriented in providing support for widgets and collaboration. It is composed of the following components:

- **Widget Spaces** - manage widgets, resources and users
- **OpenApp** - JavaScript library for local inter-widget communication and storage at the space level
- **ROLE IWC** (ROLE Inter-widget Communication library) - a library for local and remote inter-widget communication
- **Authorization and authentication framework**
- **Privacy control schemes**

The ROLE architecture provides the following types of services:

- **Widget services** - widgets and bundles of widgets
- **User services** - user related functionality

https://github.com/ROLE/ROLE/wiki
- Space services - functionality related to the space concept

The ROLE architecture is based on a three-tier model:

- Data Tier provides storage capabilities, hosting of semantic resources and a RESTful service interface
- Logic Tier (User, Space, Resource, Real-time services and Widget Engine)
  - the resource service hosts the semantic model server and provides functionality to other services
  - the space service is responsible with the space context
  - the user service manages the users within a space, applies security restrictions, etc.
  - the real-time service provides collaboration features [22] [8] and a widget engine for widget rendering
- Presentation Tier

ROLE resulted in a collection of open source software, hosted on various open source platforms, such as SourceForge and Google Code. ROLE SDK is currently available as an open source project on SourceForge.

The ROLE SDK provides a good support for individual learners through the use of Mash-up Personal Learning Environments and can efficiently enable near real-time collaboration between learners which are part of the same space. Its architecture can ensure security by the use of standards such as OAuth and OpenID and provides a good framework for personalized learning, as it unifies recommender systems, semantic resources, widget-based technologies, services, etc. However, the ROLE project is not focused towards scaffolding learning between organizations and it requires more support for operations involving digital artifacts. In addition, ROLE SDK lacks dedicated support for mobile platforms. This drawback was recently improved with the extension of the ROLE SDK with the DireWolf framework [9].

DireWolf – Distributing and Migrating User Interfaces for Widget-based Web Applications

The DireWolf framework [9] was developed by RWTH as an extension to the ROLE SDK. The framework enables a transition from “single user on a single device” computing model to rich Web applications with Distributed User Interfaces (DUIs) over a federation of heterogeneous commodity devices supporting modern Web browsers such as laptops, smart phones and tablet computers (see Figure 19).

Using the ROLE SDK, the interface distribution is based on widget technology, realized from a technical perspective using cross-platform inter-widget communication (based on the XMPP standard) and seamless session mobility. Inter-widget communication connects the widgets and enables real-time collaborative features, as well as runtime migration in our framework. Figure 19 presents the concept of widget-based user interface distribution (left side) and the interface of the DireWolf framework. Users can have near real-time

25 http://sourceforge.net/
information of the available online devices, together with the widgets residing on each device.

![Figure 19. DireWolf Concept (left) and User Interface (right)](image)

The widget distribution can be done by simple drag and drop between the available devices. For a single user, DireWolf provides more flexible control over different parts of an application by enabling the simultaneous use of smart phones, tablets and computers, which will be a valuable feature in application environments with heterogeneous devices.

5.1.6 SeViAnno

SeViAnno is a semantic video annotation tool developed RWTH. The tool allows several types of semantic annotations, i.e. Place, Object, Agent, Concept, and Event. Moreover, SeViAnno is a part of the i5Cloud (cf. Section 5.1.1). A mobile version of the application, AnViAnno was also developed to support collaborative video annotation. The AnViAnno application is detailed in the Layers D4 deliverable.

SeViAnno was developed in the context of the German Science Foundation Cluster of Excellence Ultra High Speed Mobile Information and Communication (UMIC; [http://www.umic.rwth-aachen.de](http://www.umic.rwth-aachen.de)) and evaluated in the context of a cultural heritage setting. The features relevant to the Layers use-cases are cloud multimedia upload and video transcoding and streaming, the use of shared media repositories and the annotation services. SeViAnno is a good example of an existing application which was used into an informal learning scenario within the Layers project. The scenario was adopted in the CAPTUS design team, where a new mobile prototype starting from the Android semantic annotation tool (AnViAnno) is under development. By using and further developing the semantic video annotation application, the project benefits from the already engineered work. Furthermore, based on existing research, the transformation to an application for informal learning can be easier made. In the basic SeViAnno use-case, a video can be uploaded using the i5Cloud services. The video is transcoded using cloud services and stored in a shared repository. Streaming services are also available for the videos. Several types of semantic annotations can be added, keeping into account video time points/ time interval information. The
semantic annotations and the simple video metadata are added using services running in the Lightweight Application Server (LAS), described in Section 5.1.3. The metadata is stored using the MPEG-7 standard, which specifies a multimedia content description interface. SeViAnno offers a good informal learning environment through its multimedia annotation capabilities. After a video has been annotated, users can navigate through the video using the semantic annotations available. Agents, places (available using Google Maps API), objects, events can be linked to specific parts of a video. The LAS MPEG-7 annotation services provide also the methods for searching for videos or annotations, etc. Furthermore, the SeViAnno authentication makes use of the user management capabilities offered by the LAS environment.

Currently, there are two client-side implementations available for SeViAnno: a Web-based application implemented using Adobe Flex and a widget-based prototype, which uses the ROLE SDK (see Section 5.1.5). SeViAnno uses a mixture of the Layers infrastructure technologies presented in the current deliverable. Its realization combines cloud technologies with a service layer offered by the LAS environment. Furthermore, it also benefits from customizable and collaborative features offered by the ROLE widget technologies, considering the widget-based prototype. Regarding the future, SeViAnno is a good candidate for an informal learning tool which can benefit from the integrated and scalable Layers infrastructure (e.g. from an integration with the semantic/annotation capabilities provided by the Social Semantic Server).

5.1.7 MobSOS

MobSOS (Mobile Service Oracle for Success) [23] [24] is a comprehensive approach for community service success analytics intended for inclusion in the Layers architecture in Y2. As such it is part of the community analytics work strand of our research orientation. Besides its possibilities for learning analytics in communities [25], MobSOS is expected to serve as data source for usage and survey data and tool suite supporting formative/summative evaluation for the different application areas starting from Year 2.

Theoretical foundations of MobSOS mainly originate from the Information Systems domain. Following the school of Design Science Research in Information Systems [26], the MobSOS approach considers community services as artifacts relevant for communities [1] to successfully carry out their particular practices. As a prerequisite to relevance validation, communities have to be supported in developing situational awareness [27] on service success in their specific context. Such awareness in turn requires modeling and measurement of service success as a complex construct that is strongly varying among communities and thus depends on a variety of common and community-specific success factors.

As a scaffold for the development of a community-specific success model, the MobSOS approach adapts the seminal DeLone & McLean IS Success Model [28] as a success model template, categorizing success factors in the six inter-related dimensions System Quality, Information Quality, Use, User Satisfaction, Individual Impact, and Community Impact (as adaptation of the original dimension Organizational Impact [28]).

---

26 http://mpeg.chiariglione.org/standards/mpeg-7
27 https://developers.google.com/maps/
28 Inspired by Greek mythology figure Mopsos, founder of the Oracle of Apollon in Klaros.
Concrete instantiations of this template add relevant success factors and proxy measures to the respective dimensions in the template from an extensible and customizable catalogue.

Powering a community service success model furthermore requires the collection of data from which success measures can be derived. The MobSOS approach pursues a combined approach of automated service usage monitoring and the additional collection of survey data for those community success aspects that cannot be measured directly. As such it collects and provides both qualitative and quantitative data, not only usable for assessing community service success. Additionally, the MobSOS approach includes a set of services and tools allowing to operationalize the collected data. Figure 20 shows the abstract architecture of the MobSOS approach, embedded in a Community Information System platform.

In general, we differentiate between three classes of components in the MobSOS approach: monitoring modules, services as part of a MobSOS service suite and a set of tools.

For the collection of usage data, the MobSOS approach applies a variety of monitoring modules collecting usage data in MobSOS databases. In LAS (cf. Section 5.1.3), we record service usage and context information [23] [24] from standard protocol connectors, including REST. However, MobSOS monitoring in LAS includes no means of privacy control during the recording of usage data until now.

In LAS2peer we additionally record arbitrary messages between different classes of agents in a P2P network of nodes, including service usage. With special regard to the lack of privacy control in LAS, monitoring in LAS2peer foresees multiple means for privacy protection besides the native LAS2peer end-to-end encryption (cf. Section 5.1.3). First, no agent-relevant information besides its identifier is stored, thus disabling the inference of a real person behind the agent identifier. Furthermore, the monitoring framework includes a simple
mechanism to switch monitoring on and off for each node. This node-level switch is controlled by the node maintainer, which can be a single person or an organization such as an SME. Additionally, developers can control which of their services are monitored with an additional service-level switch, thus allowing more fine-grained control.

Concrete technical implementations of the abstract MobSOS architecture are currently in place in LAS, LAS2peer, and the ROLE Sandbox, but are in principal possible for other components of the Layers Architecture. The overall MobSOS approach was introduced to the project consortium during the Graz Consortium Meeting. Usage data of all LAS services is continuously recorded, including data for SeViAnno services (cf. Section 5.1.6). In the ROLE Sandbox, we continuously collect HTTP protocol logs, which provide information on the access of the ROLE SDK services via its RESTful API [25]. In conclusion, the MobSOS approach already now provides rich usage data from various components of the Layers architecture and together with its support services seems suitable for inclusion in the Layers project’s overall formative evaluation process (see D1.1)

5.1.8 Intradoc247 (Candidate)

Intradoc247 by PinBellCom Ltd. is a Practice Management and Information System, i.e. a GP intranet. It provides a Web interface and is module-based so that GPs can choose their own service plan. The server resides in the surgery on its own hardware which ensures data security. Limited access is available from home via login and password.


Functionality:

- Document Server: Full document control (versioning), virus scanning
- Manages GQC and NHS guidelines and quality control audits
- Complaints logs: to file lessons learned and tasks.
- Calendar, e.g. holiday management (staff can book leaves), room bookings
- Noticeboard for easy two-way communication (like a simple forum)
- Roles based access

They integrate with some HR rota tools and room booking tools as well as with Osmosis, a note-taking tool that is able to generate reports on learning efforts.

Competing solutions for GP practice intranet software in the UK include: GPintranet ([http://www.gpintranet.net](http://www.gpintranet.net)) which serves several hundreds of GPs, and My Surgery Website ([http://www.mysurgerywebsite.co.uk](http://www.mysurgerywebsite.co.uk)), UK’s number one GP surgeries intranet provider with more than 3000 GPs

**Role of Intradoc247 for Layers.** According to the healthcare cluster partner LEEDS, 20% of the General Practitioners (GP) practices in the region are using Intradoc247. There is a possibility that Intradoc247 will be used by all 500+ practices in the region, if it becomes the system chosen by the Commissioning Support Unit (CSU; for detailed explanations of the entities in the healthcare cluster please consult D7.1). For the practices who use it, it is key software that they use on a daily basis to share information and communicate with others.
within the practice. If it is adopted as the intranet system to be used across the CSU region then it could also be used for sharing and communication across the 500+ practices. The aim is that a clinician working at a PC in a GP practice will probably have two pieces of software open at all times – the patient record system and the intranet software.

Users have repeatedly reported to Layers partner LEEDS that they do not want to have to come in and out of multiple systems in order to use the Layers tools and therefore if Layers tools could be linked into (plugged into) one of the key software systems that the practices already use (and in which they already have contact information and shared documents) then this will make it much easier to get the Layers tools adopted and tested/used on a large scale. Communicating with the patient record system is definitely not possible due to security and data confidentiality issues. However, connecting to (communicating with) the intranet system may be possible. Additionally the intranet software will already connect the 500+ GP practices and so working with it should help make using the Layers tools to facilitate inter-organizational learning easier.

There is potentially some overlap between functionality proposed in Learning Layers tools and functionality that will be provided by the intranet software. This makes working with the Intradoc247 developers important since we do not want to risk duplicating functionality in two separate systems. Duplication of functionality would probably mean that the practice staff will just use the key software (the intranet software) to undertake these actions and will not then login to a separate Layers tool to perform the same (or similar) action again.

Currently, Layers partners are liaising with the Intradoc247 partners to identify possible interfaces between this intranet system and the Layers architecture. Therefore Intranet247 has candidate status at the present time and is therefore not explicitly present in the architecture figures.

### 5.2 Overview of Layers Web Services

The current Layers architecture, which is schematically shown in Figure 13, offers several Web services through its components. Components expose service functionality via sets of RESTful Application Programming Interfaces (APIs) to external software programs, e.g. tools developed for the application clusters, based on the work in the design teams. In the Layers context we thus assessed the set of services provided by Layers components. All service components and their particular interface APIs were integrated in a UML component model. To illustrate the evolution of the integration of architecture components based on co-design and development work during Year 1, we include two diagrams in this section:

Figure 21 shows the initial state of the architecture components at the beginning of the project. i5Cloud was hosted on outdated hardware and the migration to new hardware and OpenStack required adaptations to the services offered, i.e. there were no services available in the beginning of Layers. The ROLE SDK was almost completed, since the ROLE project was in its final three months when Layers was launched. The Social Semantic Server did not offer an API yet, and—apart from RWTH’s components hosted as LAS services—all other components were not integrated at all.
In contrast, Figure 22 reveals the integration progress made during Year 1\textsuperscript{29}. All components on the Application Service Layer (Figure 13) are now deployed within the virtual machine environment of i5Cloud, including the Social Semantic Server, LAS, and the ROLE SDK. The i5Cloud services are up and running on the new infrastructure. In the second project year the integration with the cloud based m-learning infrastructure by TRIBAL will be propelled to be able to offer content and services under one umbrella, which is one of the objectives of task T6.4 in WP6.

Further integration successes can be expected soon in Year 2 based on integration activities that have started already and are indicated as interface use in Figure 22. As detailed in Section 5.5.2, the Social Semantic Server and the SeViAnno backend technologies are currently being integrated, and the Social Semantic Server is adapted towards making use of the ROLE SDK.

Last but not least, all service components offer interfaces mostly in the form of RESTful Web Services. This is a great starting point to move towards flexible architecture with tighter integration during the second project year. The current status suggests that a strong integrating role will be played by the “Layers Adapter” component, which is hinted to by a dotted outline shape in Figure 13, and mentioned as the key to the flexible architecture in later stages of the project. It could be used to tie up the loose interfaces in Figure 22, and to provide a transparent proxy to the Layers infrastructure at the application partners’ sites.

\textsuperscript{29} The diagram was created at a level of aggregation that reduces the modeled components to those that are represented as sub-sections of Section 5.1.
5.3 Data Security and Privacy

Assuring the security and privacy of personal data is one of the most important non-functional requirements for the Layers architecture, as stressed in the DoW, voiced during the requirements engineering during Year 1 (see Appendix C), as well as in D7.1 under the heading “issues, challenges, barriers and opportunities” in both application clusters.

While not the absolute priority for the first-year small-scale architecture, several included components do consider high standards and appropriate measures for security and privacy. These are mentioned in several places in this deliverable. Here we provide a cross-cutting summary of the most important characteristics of the architecture in regard to these concerns.

The i5Cloud (cf. Section 5.1.1) as one of the main infrastructure-level components offers a virtual machine (VM) environment that ensures security of the data contained within VMs. Further, it employs a hybrid cloud approach that allows allocation of computational resources to external cloud providers (e.g. the Amazon Elastic Compute Cloud) with appropriate security characteristics for sensitive business data in the competitive SME environment.

LAS (cf. Section 5.1.3) offers services for security management. During the design process leading to the LAS2peer evolution security and privacy were prime concerns. Peer-to-peer networks are robust, fault-tolerant and scalable, and offer a maximum of security and access...
control. Also, the monitoring in LAS2peer (see 5.1.7) foresees multiple means for privacy protection besides the native LAS2peer end-to-end encryption.

The ROLE SDK (cf. Section 5.1.5) also provides privacy control schemes and its architecture can ensure security by the use of standards such as OAuth and OpenID.

Security is also built into application-level components like the candidate Intradoc247 (cf. 5.1.8). The server is hosted on hardware in the GP practice, and access to confidential data is limited from outside.

The issues of data security and privacy will be considered in collaboration with the other technical partners and WP7 as a key success factor during upcoming WP6 activities.

5.4 Integration with Co-Design Process

As prototype development in the Learning Layers project happens on the level of individual design teams (see D1.1 Section 4 for an overview of the approach and of all design teams formed), and currently very much focuses on their respective scenarios, their user requirements for an integrated architecture have to be considered separately. However, as indicated in the Layers architecture vision in Section 2, it is of great importance to combine and unify those demands to build a common backend infrastructure offering cohesive and connected services (cf. task T6.5 in the DoW). Bringing together different demands for various application areas in construction and healthcare is a major challenge.

We tackled this by employing the House of Quality methodology as described in Section 3.4 for each Design Team separately. As for creating the matrix within the HoQ both understanding the underlying use cases as well as technical knowledge is essential, the LDTF members played a big role in the HoQ creation process of their particular Design Team. Additionally, requirements concerning the architectural development were discussed in the bi-weekly online meetings held by the LDTF.

To enable a unification of the diverse scenario user requirements, we established a common vector of technical requirements that is set out by the architecture components introduced by project partners. These functional requirements build the top of the House of Quality, thus allowing us to list the user requirements and relate them to the very same technical attributes.

The House of Quality approach is seen here as an instrument for scaling in the SME clusters, since it is not meant as a one-time effort to compare technologies in Layers. It is a consulting instrument for SME clusters that will support resolving architecture and infrastructure related issues in the clusters long after the Layers project has ended. The House of Quality App that WP6 developed in Year 1 is currently being integrated with the Requirements Bazaar, which will make the scaling of these instruments beyond local cluster communities easier.

In the following, we visit each Design Team’s HoQ separately and show how the approaches combine to leverage a common integrated architecture.

5.4.1 Feature Extraction

The technical features describe the underlying foundation for building the app prototypes upon. As shown in Section 5.1, various technologies offered by different partners can be
used to build a common first-year integrated architecture. To represent these architectural components in the House of Qualities of each Design Team, we employed Feature Extraction, which basically means to characterize certain functionalities by some precise key words. By that, they can be matched to certain other aspects of other components. E.g., the object storage of the i5Cloud is referenced in the HoQs as “REST-based Cloud Storage”. By the generalization, it can also be referenced to the cloud storage capabilities of the Tribal M-Learning Platform.

Table 1 provides an overview on the gathered overall features and links them to the technologies they are tackled in. As can be seen, user authentication is a functionality that is offered by all parties. Currently it remains a challenge to provide a single sign-on solution where users are able to authenticate them once and use their credentials throughout all systems.

Table 1. Matrix of Features Tackled by Architectural Components

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>i5Cloud</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>&lt;</td>
</tr>
<tr>
<td>Tribal M-Learning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAS/LAS2peer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Social Semantic Server</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROLE SDK</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SeViAnno</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.4.2 CAPTUS Design Team

The user requirements of the CAPTUS Design Team are based on a scenario for a permanent exhibition at Norddeutsches Zentrum für Nachhaltiges Bauen (NZNB) in Verden, Germany. The requirements were gathered on a CAPTUS meeting in early July, 2013 at ITB Bremen and consider all stakeholders namely companies displaying exhibits, the NZNB who organizes the event, private house owners and craftsmen who visit the exhibition as well as experts like architects who are able to answer the questions of visitors concerning specific issues. From a technical perspective, peer production is an important feature, as both companies and experts should be able to produce materials to be displayed at the exhibition. During and after the exhibition, visitors are encouraged to file their questions or hands-on experiences along the exhibits.
The formal results of the requirements elicitation have been considered in the CAPTUS HoQ in Appendix E1. As can be seen, the most important technical aspect is a “User Roles / Permission System” which reflects the various stakeholders involved. E.g. content producers need to be able to set very specifically which target audiences should be able to see certain images or videos due to right issues. The second highest ranked requirement is “Metadata Annotation”, which is the underlying fundamental idea of the exhibition scenario.

CAPTUS plays a big role in WP6 activities as a number of activities is directly related to this Design Team. Student involvements are mainly focused in a lab team at RWTH in the winter term 2013/2014. In this context, NZNB (via consortium partner AGENTUR) acts as a software customer while the students of the respective team have to develop the software according to the customer’s requirement. The details of the lab course are presented in Section 5.5.2.

The technical backend of the CAPTUS is the i5Cloud as shown in Section 5.1.1. It provides convenient methods for cloud storage and multimedia services that can be leveraged for the exhibition scenario. This scenario includes several research challenges like how to interact between the physical objects, personal mobile devices owned by the visitors and public terminals available in the exhibition hall. Attempts to overcome such session mobility issues have been tackled by RWTH through the use of the XMPP protocol [29]. XMPP also has a strong role in the ROLE SDK (see Section 5.1.5).

The frontend development for CAPTUS is currently focused within the “Ach so!” prototype (see D4.1 Section 5.2.4). It is based on the SeViAnno and AnViAnno Video annotation tools provided by RWTH (see Section 5.1.6).

5.4.3 Bits and Pieces Design Team

The Bits and Pieces Design Team is mainly led by WP5 activities; more information on the use cases can be found in D5.1. The Bits and Pieces HoQ is depicted in Appendix E2. The most important technical feature based on the scenario’s user requirements is user management. This was already highlighted for the CAPTUS design team, similarly for the demand to have semantic annotation capabilities.

Most notably, the Bits and Pieces user requirements list visualization possibilities as a user requirement, which is not met by any of the technical characteristics. This is due to visualization capabilities being a frontend technology that does not have any correspondent functionalities in the backend architecture. Notifications are of high importance, while multimedia services like image and video processing are not required at the current stage.

Semantic Metadata is of high importance for this prototype. Therefore, this Design Team mainly focuses its backend calls on the Social Semantic Server which is being integrated into the integrated architecture within the i5Cloud.

5.4.4 Sharing Turbine Design Team

Sharing Turbine as Design Team is about the “White Folder” concept that helps apprentices in their daily workflow by collecting (digital) material within a virtual folder that can be accessed at all times; details are available in D3.1. The HoQ of the Sharing Turbine Design Team is attached in Appendix E3. As can be seen in the bottom line, the most important technical feature (16.7%) that is needed to develop the Sharing Turbine prototype is “REST-
based Cloud Storage”. This is due to the “White Folder” approach of the underlying scenario, where apprentices collect learning material in a virtual folder. Another important outcome of the HoQ is that currently, the user requirement “Can operate online and offline” is not matched by any of the technical features on the top. In Sharing Turbine, online and offline operation is important for phases of disconnection, e.g. when radiation is not permitted in certain parts of a hospital and mobile devices operate in airplane mode having all wireless capabilities switched off.

Technically, offline operations have consequences both for the frontend as well as the backend part, as on the mobile device a cache has to be built up for prefetching results and delayed write-back to the server. In the backend, content may be packaged into a single file that is downloaded immediately and consumed later. The packaging concept of the Tribal M-Learning Platform provides adequate means to allow for this. However, the lack of this functionality is an indication that this technical feature is still missing in the upper part of the HoQ and will be added in the next iteration of the process.

Similar to CAPTUS, a user management system is a high demand with 15.4%. Overall, this feature is highly needed and needs to be tackled by a profound and thoughtful realization.

### 5.4.5 PANDORA Design Team

The PANDORA case is explained in detail in D2.1 and D3.1. The input for the PANDORA HoQ is based on co-design workshops in the healthcare cluster. The final HoQ is attached in Appendix E4. It is obvious that both functional requirements concerning multimedia services are not related to any of the user requirements. This is a clear indication, that the PANDORA scenario does not include dealing with images or multimedia files of any kind.

Metadata including comments and ratings as well as semantic annotation both play a big role as they have high importance indicators. The most important requirement is “Notifications (synchronous and asynchronous)”, as general practitioners need to be informed of changes and new comments soon.

### 5.4.6 Implications for the Integrated Architecture

The HoQs produced for the design teams can be used to assess architectural designs and a composition of technologies for the Layers Architecture. Such assessments can be used to provide recommendations to the Layers Architecture Board when it comes to making potentially high-impact decisions on the Layers infrastructure. This allows us to continuously assess and accommodate new technologies and requirements emerging during the runtime of Layers, facilitating the overall goal of fast and flexible deployment of scalable technologies for informal learning at the workplace. The overall result when comparing the individual HoQs for the four design teams is that both user management and cloud-based file storage are very important requirements to be met by the integrated architecture. Other important factors are multimedia services as well as semantic annotations.

Each design team scenario focuses on a specific subset of demanded features. However, we can also see that the technologies required are indeed already tackled by the existing technologies brought into the Layers project by the technological partners. Each of these technologies delivers specialized technologies in its respective fields. The individual architectural building blocks are already integrated within the i5Cloud, and their API may
be accessed separately by client applications. It remains a challenge, to build a common API infrastructure where all these separate interfaces are integrated seamlessly. Especially user management is currently handled by every component itself; thereby it is a highly requested functionality across all design teams.

The HoQs helped us to identify those commonalities and therefore their outputs provide a good perspective on future tasks and challenges.

5.5 Propelling External Developer Engagement

An important aspect of the development work in Layers is the involvement of external communities as part of the open development vision in WP6 as well as in the stakeholder engagement activities managed by WP7. In the first year, several opportunities were seized to propel this engagement, some of which have been reported in previous sections (e.g. the Developer Camp in Section 4.3). In this section we report how students at PhD and master were or are being involved in Layers co-design work (Section 5.5.1) and development work (Section 5.5.2), respectively.

5.5.1 Summer School Workshops

RWTH organized two workshops at two different summer schools for PhD students. The first one took place at the Joint Summer School on Technology Enhanced Learning in Limassol (Cyprus). The second workshop was part of the GALA Summer School on Serious Games in Graz (Austria).

In both cases the main objective was to actively engage PhD students in design and prototyping activities for real scenarios from the Learning Layers project. The main topic was semantic video annotation as a scaffold for informal learning at the workplace. The students were asked to explore different existing user interfaces for semantic video annotation. They were encouraged to explore the features of the classical Web-based SeViAnno application, of a Widget-based SeViAnno version capable of user interface distribution, and of AnViAnno, a mobile version running on Android. After a round of reflection on these different user interface metaphors the participants were asked to design apps and create ideas on how to exploit semantic video annotation for informal learning experiences at the workplace, focusing mainly on gamification aspects at the second workshop. As a starting point, they were provided with material gathered from application partners in the Learning Layers project by involving the end-user clusters from the construction domain.

The annotations could be used to improve the communication between the workers and the masters and to simplify the capturing and assessment of the work activities. A feedback cycle was also mentioned, where videos can be enriched with new annotations, as a result of the collaboration between workers and masters and the identification of problems and knowledge gaps. To raise motivation of users in this process, gamification mechanisms can be integrated. Workshop participants have suggested various rewards (e.g. social badges, leader boards) as well as rating of videos. All these suggestions will be consider in further development of the semantic video annotation application.
5.5.2 High-Tech Entrepreneurship Lab at RWTH

RWTH offers a lab course each semester on “High-Tech Entrepreneurship and New Media” (HENM)\(^30\). The course covers special chapters for the development of complex information products, requiring work in teams of 5-6 people on real-world projects. These projects are offered and represented by companies that act as the customers for the student teams. The lab course design has been used for more than 10 years at RWTH, and it has also been subjected to research from an empirical community-of-practice perspective [30] [31]. For the 2013/14 winter term we adopted three major Layers design and development endeavors to be represented and deeply integrated into the HENM lab:

1. **Developer Camp**: As part of a development process in Layers that involves the external (open source) developer community, all HENM students were invited to participate as developers in the 1\(^{st}\) Layers Developer Camp (see Section 4.3). This participation was part of the preparation for their student projects as listed below.

2. **CAPTUS Design Team**: Together with AGENTUR, RWTH proposed a student project for developing digital support for the permanent exhibition scenario in Verden which is part of the CAPTUS design team (see Deliverable D1.1). The exhibition scenario includes among other features the annotation of physical objects, scanning QR (Quick Response) codes\(^31\), Near Field Communication (NFC) tags with mobile devices, as well as support for creating and maintaining a social network of visitors, architects and exhibitors before, during and after the exhibition visits. A representative of AGENTUR will act as the real-world customer for the student team.

3. **SeViAnno and Social Semantic Server integration**: This student project aims at integrating the semantic video annotation features of SeViAnno (cf. Section 5.1.5) with the Social Semantic Server (SSS) developed by TUG (see Section 5.1.2 as well as D5.1 for details) using the RESTful API offered by the SSS. The real-world customer for this project is PONT, while support for technical issues will be sought from TUG.

The student projects are currently in progress. There will be three formal project reviews hosted and moderated by RWTH for each project, as well as one final public presentation of project results at RWTH.

5.5.3 Student Co-Supervision

Aside involving student communities in the Layers project through the Developer Camp initiative and the practical course work, various master theses are also currently being related to the project, in the context of the shared infrastructure (WP6 activities) and cooperation with other project members. Such a master thesis, targeting to research scaffolding and informal learning by means of a micro-learning study is currently running, as cooperation between RWTH Aachen and UWE (Prof. John Cook being one of the thesis supervisors). The work intends to use the initial infrastructure of the project in order to devise means to bridge self-regulated and community-regulated learning using scrapped content from the Web. The thesis explores also the relation between microlearning and scaffolding informal


learning in Communities of Practice and to evaluate the development with current Layers technologies. The work promises good results for the second year’s research in the Layers project. The research questions proposed in the work are the following:

- Can community-regulated learning techniques scaffold learners to accomplish their learning goals faster and/or more conveniently than self-regulated learning, for a microlearning setting?
- What is the relation between communities of practice and microlearning?
- Can a mixture be used to support informal learning at workplaces?
- Can visualization of tagging/ notes be used in engaging new members based on shared interests?
- Can a widget based Web Dashboard and the principle of learning spaces support individuals and communities in microlearning tasks?

6 Conclusion and Outlook

6.1 Conclusion

This deliverable has summarized the achievements from the first project year in WP6. The main objective was an initial architecture for small-scale deployment. A technology survey has been conducted in the first 6 months and a software engineering process and infrastructure has been specified and set up, together with its mechanisms and supporting tools. Here, the requirements engineering and corresponding technology assessment has played a crucial role in this project phase. These activities led to the development and deployment of an initial version of the Layers architecture.

Our work has built upon three main pillars of the Layers architecture vision: research orientation, open (source) development and continuous integration. We aimed to address these pillars in detail in the respective Sections 3 (survey of technologies), 4 (open development and integration process) and 5 (initial architecture). The process of integrating Web engineering achievements into an informal learning scenario has been illustrated by the SeViAnno scenario, when a semantic Web annotation tool was picked up and further developed by the CAPTUS design team.

The Model of Scaling Support of Informal Learning [32] (see also D5.1 for details), which represents one of the main theoretical outcomes of the first year of the Learning Layers project has been also reflected in the first version of the Layers architecture. Scaffolding of learning is supported by adaptive learning technologies that are enabled by the ROLE SDK. Peer production is facilitated by social networks and communication technologies, which are currently represented by the Requirements Bazaar. Shared meaning emerges from the usage of semantic technologies, like the Social Semantic Server and SeViAnno.
6.2 Outlook on Year 2

After creative divergence in Year 1, integration will be the theme in the next phase. Year 2 will be an important one for WP6 from the perspective of allocation of resources, which implies expectations of an essential progress in the architecture development as well as in the technical integration of various services and tools for our two clusters. These plans are outlined by two milestones. The first one is due in Month 18 (April 2014) and refers to the Integration Conference and its results, including integrated designs and integration plans. The second milestone is due in Month 24 (October 2014), when a flexible architecture and social server layer should allow a configurable integration of various tools. For the end of the second year, the deliverable D6.2 is scheduled, which will present a customizable architecture for flexible small-scale deployment. It will report on the architecture evolution during Year 2, considering technical improvements of the Layers networking infrastructure towards increased customizability and flexibility in deployment and use (T6.3), integrated content storage considering access services and a specification of the App Store (T6.4), as well as an updated report on technical integration of tools and services (T6.5). Cooperation with other work packages is also planned in order to understand and address their technology requirements, including ubiquitous help seeking interactions and collaborative Q&A tool in WP2, tools for creating and maturing instructional material in WP3, tools for artefact and mobile layer in WP4, the updated Social Semantic Network in WP5, and development of API integration (i5Cloud and mobile capability) with WP7. Other relevant requirements in the development of technological solutions contain consideration of sustainable business models, security and privacy issues, as well as the important role of boundary objects (these are all explained in detail in D7.1).

A continuous integration on the technical level is fostered by the Open Developer Library (ODevL), which was set up to enable scaling of the co-design outcomes and the development process, as well as integration of business model ideas and the technical infrastructure. Our technological innovations will be based on the cloud services and we need to consider the concepts of learning analytics [6], which has become a major research area recently (cf. [33]). In the world of informal learning at the workplace, knowledge gained from formal learning analytics is only applicable on a commodity level. Since professional communities need learning support beyond this level, we need a deep understanding of interactions between learners and other entities in community-regulated learning processes (Table 2)—that is, a conceptual extension of self-regulated learning processes.

As mentioned in Section 3.4, in the next phase we plan to integrate the House of Quality Web application with the Requirements Bazaar in order to allow a coherent workflow that will enable to study how gathering of requirements is evolving within communities.

Concerning the project’s technical goal of creating a lightweight architecture that can be deployed flexibly and in a scalable way while supporting custom privacy and security standards, we planned to both evaluate and develop further the LAS2peer framework (cf. Section 5.1.3) towards Layers objectives in Year 2.
### Table 2. Learning Analytics vs. Community Learning Analytics [6]

<table>
<thead>
<tr>
<th></th>
<th>Formal Learning</th>
<th>Learning Analytics</th>
<th>Community Regulated Learning</th>
<th>Community Learning Analytics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment</td>
<td>LMS</td>
<td>EDM/VA</td>
<td>CIS/ROLE</td>
<td>DM/VA/SNA/Role Mining</td>
</tr>
<tr>
<td>Tools</td>
<td>Fixed</td>
<td>LMS Specific</td>
<td>Eco-System</td>
<td>Tool Recommender</td>
</tr>
<tr>
<td>Activities</td>
<td>Fixed</td>
<td>Content Recommender</td>
<td>Dynamic</td>
<td>Content Recommender / Expert Recommender</td>
</tr>
<tr>
<td>Goals</td>
<td>Fixed</td>
<td>Progress</td>
<td>Dynamic</td>
<td>Progress / Goal Mining / Refinement</td>
</tr>
<tr>
<td>Communities</td>
<td>Fixed</td>
<td>Not applicable</td>
<td>Dynamic</td>
<td>(Overlapping) Community Detection</td>
</tr>
<tr>
<td>Use Cases</td>
<td>Courses</td>
<td>Learning Paths</td>
<td>Peer Production / Scaffolding</td>
<td>Semantic Networks of Learners / Annotations</td>
</tr>
</tbody>
</table>

With special regard to the extension of the Layers formative evaluation process (cf. D1.1) towards the inclusion of Layers service and application usage data, cooperation is planned for all work packages involving formative evaluation tasks, starting from Year 2. These are in particular WP2-4 for the formative evaluation of their prototypes, WP5 for the evaluation of the Social Semantic Server Services, and WP7 for the formative evaluation of learning services and tools in the application partner domains of construction and healthcare. Among other instruments, evaluation with the MobSOS approach and its supporting tools and services (cf. Section 5.1.7) is planned for Year 2.

## 7 References


[21] F. Mödritscher, Reference implementation of mash-up personal learning environments to foster the broad community uptake, ROLE Project Consortium, Ed.


Appendix A: Technology Survey

The following lists the architectural models and products described in the wiki\(^{32}\) as part of the technology survey. A complete snapshot of the technology survey list exported from the wiki is available in [34], pp. 1–35.

Architectural Models:
- Federated IS Architecture
- Service Oriented Architecture, Enterprise Service Bus, Resource Oriented Architecture
- Cloud Computing
- Grid Computing

Products:
- ROLE Infrastructure
- LAS2peer
- Apache Wave
- i5Cloud
- Mobile GWT
- SeViAnno
- MobSOS
- Social Semantic Server
- M-Learning Platform
- OpenStack Compute
- XMPP
- HTML5
- WebRTC
- WebSocket
- OSGi Framework

\(^{32}\) [http://htk.tlu.ee/layers/MW/index.php/T6.1_Phase_I#Collected_Input](http://htk.tlu.ee/layers/MW/index.php/T6.1_Phase_I#Collected_Input); cf. [34], pp. 1–35
Appendix B: Functional Requirements

This appendix contains the functional requirements which were available after the elicitation phase. The table can also be found on the Layers Wiki and publicly in [34], p. 36.

Header significance:

- **FR number** have to be added to the requirements, according to prioritization
- **has significance** can have one of the following values:
  - M = Must have
  - S = Should have
  - N = Nice to have
- **has source** contains the main parts(DoW A,B, Context Cards, Design Ideas) from which the requirement originates (for querying purposes)
  - R_DoW_A,B represent parts A and B from the Layers DoW
  - R_CTX_C,H represent the context cards for the construction and healthcare sector respectively
  - R_DI represent the design ideas developed after the Layers Design Conference. The corresponding design teams are reflected with their names: BitsPieces, Pandora, SharingTurbine, Captus
- **has trace** contains the part and requirement number from which the requirement originates

<table>
<thead>
<tr>
<th>Has FR number</th>
<th>Has name</th>
<th>Has significance</th>
<th>Has description</th>
<th>Has source</th>
<th>Has trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR</td>
<td>Learning Analytics</td>
<td>S</td>
<td>Validation of learning activities and content. Provide framework for reflection of workplace learning activities.</td>
<td>R_DI Pandora</td>
<td>R DoW A 10.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW A</td>
<td>R DI Captus</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus 8</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Unique Identifier</td>
<td>M</td>
<td>Unique identifiers for resources and users across organizational borders.</td>
<td>R DoW B</td>
<td>R DI Captus 16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R CTX C 2.2</td>
</tr>
<tr>
<td>FR</td>
<td>Activity Stream</td>
<td>S</td>
<td>Timeline of user actions.</td>
<td>R DoW B</td>
<td>R DoW B 3.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R CTX C 8</td>
</tr>
<tr>
<td>FR</td>
<td>Synchronous Communication</td>
<td>M</td>
<td>Support instant messaging between users.</td>
<td>R DoW B</td>
<td>R CTX H 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Asynchronous Communication</td>
<td>M</td>
<td>Support e-mail-like communication between users. Notifications on activities can be sent outside the system.</td>
<td>R_DI Pandora</td>
<td>R DI Pandora 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R CTX H 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Has FR number</th>
<th>Has name</th>
<th>Has significance</th>
<th>Has description</th>
<th>Has source</th>
<th>Has trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR</td>
<td>Awareness of Changes</td>
<td>M</td>
<td>Support awareness of changes and decisions</td>
<td>R CTX H</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Coopetition</td>
<td>M</td>
<td>Strict enforcement of dynamic access rules for content. Provide support for cross-organizational cooperation while detecting and protecting competitive knowledge.</td>
<td>R DoW A</td>
<td>R DoW B</td>
</tr>
<tr>
<td>FR</td>
<td>Social Network</td>
<td>M</td>
<td>Social network reflecting relationships between stakeholders. Persistent interactions with people across enterprises in the cluster.</td>
<td>R DoW B</td>
<td>R CTX C</td>
</tr>
<tr>
<td>FR</td>
<td>Community Support</td>
<td>S</td>
<td>Share materials and folders with groups of users.</td>
<td>R DI SharingTurbine</td>
<td>R CTX C</td>
</tr>
<tr>
<td>FR</td>
<td>Content Sharing</td>
<td>M</td>
<td>Saved content (files, documents) can be shared between people and organizations.</td>
<td>R CTX C</td>
<td>R DoW A</td>
</tr>
<tr>
<td>FR</td>
<td>Metadata Management</td>
<td>N</td>
<td>Resources contain searchable metadata. Automatic creation and suggestion of metadata if applicable.</td>
<td>R DI BitsPieces</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Anonymization</td>
<td>S</td>
<td>Removing personal details like data about a certain patient.</td>
<td>R DoW B</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Recommender System</td>
<td>S</td>
<td>Recommend resources, tags, annotations, collections, more suitable material, users and communities.</td>
<td>R DI Pandora 15.5R DI Captus 18R DoW B 4.2R CTX C 1.1R CTX H 1</td>
<td></td>
</tr>
<tr>
<td>Has FR number</td>
<td>Has name</td>
<td>Has significance</td>
<td>Has description</td>
<td>Has source</td>
<td>Has trace</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------</td>
<td>------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>FR</td>
<td>Realtime Collaboration</td>
<td>M</td>
<td>Realtime collaboration between professionals on documents and learning resources. Simultaneous access of documents. Several commentators must be able to work simultaneously on a document.</td>
<td>R CTX C R DI Pandora</td>
<td>R DI Pandora 4 R CTX C 3 R CTX H</td>
</tr>
<tr>
<td>FR</td>
<td>Rating Resources</td>
<td>N</td>
<td>Rating is supported to state the difficulty of problems or to rate solutions.</td>
<td>R DI SharingTurbine</td>
<td>R DI SharingTurbine 10</td>
</tr>
<tr>
<td>FR</td>
<td>Offline Support</td>
<td>M</td>
<td>Provide functionality related to occasionally connected resources like caching, prefetching. Example: Information is captured offline and synchronized later with the online infrastructure.</td>
<td>R DI BitsPieces R DI SharingTurbine</td>
<td>R DI BitsPieces 9 R DI SharingTurbine 1.1</td>
</tr>
<tr>
<td>FR</td>
<td>Authorization</td>
<td>M</td>
<td>Provide fine-grained authorization control.</td>
<td>R DoW A</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Q&amp;A Tool</td>
<td>M</td>
<td>Post questions to Q&amp;A tool. Attach multimedia, comments and answers.</td>
<td>R DoW B R CTX H R DI SharingTurbine R DoW B 4.5</td>
<td>R DoW B 3.2 R DI SharingTurbine 8 R DoW B 4.5</td>
</tr>
<tr>
<td>FR</td>
<td>Physical Artifacts</td>
<td>M</td>
<td>Physical tools are reflected in the system and linked from the real world. Tagging of physical artifacts. Capture interactions with physical artifacts.</td>
<td>R DoW A R DoW B R CTX C R DI Captus</td>
<td>R DoW A 12.1.1 R DI Captus 16 R DoW B 2.3 R CTX C 2.1</td>
</tr>
<tr>
<td>FR</td>
<td>Expertise Levels</td>
<td>M</td>
<td>People have expertises assigned to them.</td>
<td>R DI Pandora R DI Captus</td>
<td>R DI Pandora 11 R DI Captus 17</td>
</tr>
<tr>
<td>FR</td>
<td>Online Content Store</td>
<td>M</td>
<td>Online store of tools and content between SMEs. &quot;App store&quot; for learning content.</td>
<td>R DoW A</td>
<td>R DoW A 6 R DoW B 4.8</td>
</tr>
<tr>
<td>Has FR number</td>
<td>Has name</td>
<td>Has significance</td>
<td>Has description</td>
<td>Has source</td>
<td>Has trace</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------</td>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>FR</td>
<td>Multimedia Editing</td>
<td>N</td>
<td>Multimedia files that are stored in the system can be edited and annotated.</td>
<td>R DoW A</td>
<td>R DoW A 6</td>
</tr>
<tr>
<td>FR</td>
<td>Rights Management</td>
<td>M</td>
<td>Manage access rights of documents and document parts like read-only. E.g. guidelines have 'official' immutable content and mutable 'inofficial' content.</td>
<td>R DI Pandora</td>
<td>R DI Pandora 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI BitsPieces</td>
<td>R DI BitsPieces 10</td>
</tr>
<tr>
<td>FR</td>
<td>Copyright Protection</td>
<td>N</td>
<td>License information is attached to resources.</td>
<td>R DI Captus</td>
<td>R DI Captus 6</td>
</tr>
<tr>
<td>FR</td>
<td>Searchability</td>
<td>M</td>
<td>Information may be searched through. Search by tags, keywords, locations, content, name. Automatic suggestions.</td>
<td>R DoW B</td>
<td>R DoW B 3.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td>R DI Captus 9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI SharingTurbine</td>
<td>R DI SharingTurbine 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B 3.7</td>
<td>R DoW B 3.7</td>
</tr>
<tr>
<td>FR</td>
<td>Information Filtering</td>
<td>S</td>
<td>Filter information to right receivers, consider different levels of expertise, find similar cases.</td>
<td>R CTX H</td>
<td>R CTX H 4.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td>R CTX H 5.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI BitsPieces</td>
<td>R CTX H 7.2</td>
</tr>
<tr>
<td>FR</td>
<td>Feedback Cycle</td>
<td>S</td>
<td>Feedback cycle between different users.</td>
<td>R CTX H</td>
<td>R CTX H 6.2</td>
</tr>
<tr>
<td>FR</td>
<td>External Tools Support</td>
<td>N</td>
<td>External tools like Flickr, Google Maps are integrated. Integrate existing prototypes developed by partners. Notifications on activities can be sent outside the system.</td>
<td>R DoW A</td>
<td>R DoW A 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW A 11.2.3</td>
<td>R DI Pandora 5</td>
</tr>
<tr>
<td>FR</td>
<td>Trust and Assignability</td>
<td>S</td>
<td>Enable trust in users and content. Created content is assignable to the authoring users. Support quality assurance processes according to guidelines.</td>
<td>R DoW A</td>
<td>R DoW A 11.2.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Pandora</td>
<td>R DI Pandora 11.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td>R DI Captus 2</td>
</tr>
<tr>
<td>Has FR number</td>
<td>Has name</td>
<td>Has significance</td>
<td>Has description</td>
<td>Has source</td>
<td>Has trace</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------------</td>
<td>------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>--------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>FR</td>
<td>Discussions and Comments</td>
<td>S</td>
<td>Enable discussions through commenting. Guidelines can be commented and worked on.</td>
<td>R DI Pandora 12</td>
<td>R DI Pandora 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI SharingTurbine 15.3</td>
<td>R DI SharingTurbine 15.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R CTX C 9</td>
</tr>
<tr>
<td>FR</td>
<td>User Roles</td>
<td>M</td>
<td>Different user roles (patient, doctor, foreman, ...) are reflected in the system.</td>
<td>R DoW A</td>
<td>R DI Captus 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Content-specific Workflows</td>
<td>N</td>
<td>Different type of content requires different update workflows. E.g. guidelines have their own maturing process.</td>
<td>R DI Pandora 14</td>
<td>R DI Pandora 14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td>R CTX H 8.1</td>
</tr>
<tr>
<td>FR</td>
<td>Organization-level Deployment</td>
<td>M</td>
<td>Different enterprises/practices are reflected in the system. The organizational level should be reflected in the infrastructure.</td>
<td>R DoW A</td>
<td>R DoW B 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B</td>
<td>R CTX H 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Data Export</td>
<td>M</td>
<td>Cross-tool data exchange. Own contributions can be exported as a proof of learning.</td>
<td>R DoW B</td>
<td>R DI Pandora 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td>R DoW B 3.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H 1</td>
<td>R CTX H 9.1</td>
</tr>
<tr>
<td>FR</td>
<td>Learning Episodes</td>
<td>M</td>
<td>Creation and editing of learning episodes and tutorials. Recording personal efforts. Annotated with context metadata.</td>
<td>R DoW B</td>
<td>R DI BitsPieces 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R DI BitsPieces 11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td>R CTX H 9.1</td>
</tr>
<tr>
<td>FR</td>
<td>ePortfolio/Whitefolder</td>
<td>N</td>
<td>Collect contributions as proof of learning. Provide framework for capturing individual and collaborative interactions. Data collections can be exported.</td>
<td>R DI Pandora 10.2</td>
<td>R DI Pandora 10.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW A</td>
<td>R DI Pandora 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R CTX C 1.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td>R CTX C 7</td>
</tr>
<tr>
<td>FR</td>
<td>Activity Tracker</td>
<td>N</td>
<td>Being able to follow and receive notifications about certain activities and resources.</td>
<td>R DoW B</td>
<td>R DI Pandora 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R DoW B 4.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td>R CTX C 12</td>
</tr>
<tr>
<td>FR</td>
<td>Summary Generation</td>
<td>N</td>
<td>Automatic generation of learning episode summaries.</td>
<td>R DoW B</td>
<td>R DoW B 3.7</td>
</tr>
<tr>
<td>Has FR number</td>
<td>Has name</td>
<td>Has significance</td>
<td>Has description</td>
<td>Has source</td>
<td>Has trace</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R DI Captus 20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td>R DoW B 3.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B 4.1</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>Activity History and Versioning</td>
<td>S</td>
<td>Activities like changing a document are saved in a history. Content is versioned.</td>
<td>R DoW B</td>
<td>R DI SharingTurbine</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI SharingTurbine</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B 3.5</td>
</tr>
<tr>
<td>FR</td>
<td>Categorizing Activities</td>
<td>M</td>
<td>Activities are tagged with a category so that users can subscribe to certain actions.</td>
<td>R DoW B</td>
<td></td>
</tr>
<tr>
<td>FR</td>
<td>File Storage</td>
<td>M</td>
<td>(Multimedia) files like video, audio, text, photos, bookmarks, scanned documents can be stored in the system. They can be referenced from the various layers. Automatic capturing of metadata about where, when, who and duration.</td>
<td>R DoW A</td>
<td>R DoW A 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW A 14.1</td>
<td>R DI Pandora 15.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C</td>
<td>R DI Captus 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td>R DI Captus 14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI Captus</td>
<td>R DI BitsPieces 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DI SharingTurbine</td>
<td>R DI SharingTurbine</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B 3.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX C 10</td>
</tr>
</tbody>
</table>
Appendix C: Non-Functional Requirements

This appendix contains the functional requirements which were available after the elicitation phase. The table can also be found in Layers Wiki\(^\text{34}\) and publicly in [34], p. 40.

Header significance:

- **FR number** have to be added to the requirements, according to prioritization
- **has significance** can have one of the following values:
  - M = Must have
  - S = Should have
  - N = Nice to have
- **has source** contains the main parts(DoW A,B, Context Cards, Design Ideas) from which the requirement originates (for querying purposes)
  - R_DoW_A,B represent parts A and B from the Layers DoW
  - R_CTX_C,H represent the context cards for the construction and healthcare sector respectively
  - R_DI represent the design ideas developed after the Layers Design Conference. The corresponding design teams are reflected with their names: BitsPieces, Pandora, SharingTurbine, Captus
- **has trace** contains the part and requirement number from which the requirement originates

<table>
<thead>
<tr>
<th>NFR number</th>
<th>Name</th>
<th>Significance</th>
<th>Description</th>
<th>Source</th>
<th>Trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR</td>
<td>Availability</td>
<td>S</td>
<td>The underlying technological infrastructure should have high availability (90% - 99%) with minimal downtime.</td>
<td>R_DoW A, R_CTX_C</td>
<td>R_DoW A 26, R_CTX_C N3</td>
</tr>
<tr>
<td>NFR</td>
<td>Backup/Data Preservation</td>
<td>N</td>
<td>Data can be backed up.</td>
<td>R_DoW A</td>
<td>R_DoW A 23</td>
</tr>
<tr>
<td>NFR</td>
<td>Cross-Organizational Privacy</td>
<td>M</td>
<td>Protect competitive knowledge of SMEs.</td>
<td>R_DoW A, R_DoW B</td>
<td>R_DoW A 11.2.2, R_DoW A 21, R_DoW B 1</td>
</tr>
<tr>
<td>NFR</td>
<td>Deployment Configuration</td>
<td>S</td>
<td>System deployments offer configuration options to allow organisations to tailor the Layers infrastructure to their requirements</td>
<td>R_DoW A</td>
<td>R_DoW A 14.7</td>
</tr>
<tr>
<td>NFR</td>
<td>Device Adaptation</td>
<td>S</td>
<td>System takes into consideration the constraints of different devices.</td>
<td>R_DI Pandora, R_DI Captus</td>
<td>R_DI Pandora 13, R_DI Captus</td>
</tr>
</tbody>
</table>

\(^{34}\) [http://htk.tlu.ee/layers/MW/index.php/Architectural_Non-Functional_Requirements](http://htk.tlu.ee/layers/MW/index.php/Architectural_Non-Functional_Requirements); cf. [34], p. 40
<table>
<thead>
<tr>
<th>Has NFR number</th>
<th>Has name</th>
<th>Has significance</th>
<th>Has description</th>
<th>Has source</th>
<th>Has trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR</td>
<td>Distributed Infrastructure</td>
<td>S</td>
<td>Light-weight, distributed and federated infrastructure. Deployable in networked enterprise context.</td>
<td>R DoW A</td>
<td>R DoW A 7</td>
</tr>
<tr>
<td>NFR</td>
<td>Easy Learnability</td>
<td>N</td>
<td>Tools are low-barrier and easy to use so that end-users do not require training.</td>
<td>R DoW B</td>
<td>R DoW B 7</td>
</tr>
<tr>
<td>NFR</td>
<td>Efficiency</td>
<td>S</td>
<td>Efficiency of documentation progress and capturing. Provide support for tool deployment with minimum costs for SMEs.</td>
<td>R DoW A</td>
<td>R DoW A 11.1</td>
</tr>
<tr>
<td>NFR</td>
<td>Extensibility</td>
<td>S</td>
<td>Architecture permits the insertion of new tools/services/components.</td>
<td>R DoW A</td>
<td>R DoW A 19.1</td>
</tr>
<tr>
<td>NFR</td>
<td>External Interoperability</td>
<td>N</td>
<td>Components adhere to well defined contracts and hide their implementation details. Integration between existing prototypes and newly developed Layers infrastructure.</td>
<td>R DoW A</td>
<td>R DoW A 12.2</td>
</tr>
<tr>
<td>NFR</td>
<td>Fault-tolerance</td>
<td>S</td>
<td>Robustness in different environments.</td>
<td>R CTX C</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Flexibility</td>
<td>N</td>
<td>Adapt to changes in system (hardware and software architecture) and deployment.</td>
<td>R DoW A</td>
<td>R DoW A 21.2</td>
</tr>
<tr>
<td>NFRa</td>
<td>Individual Privacy</td>
<td>S</td>
<td>Protect personal information</td>
<td>R DoW A</td>
<td>R DoW A 3</td>
</tr>
<tr>
<td>NFR</td>
<td>Internal Interoperability</td>
<td>N</td>
<td>Various internal tools and modules interoperate through the stack of architectural layers. The infrastructure supports communication between layers.</td>
<td>R DoW A</td>
<td>R DoW A 12.1</td>
</tr>
</tbody>
</table>

http://Learning-Layers.eu
<table>
<thead>
<tr>
<th>NFR</th>
<th>Has name</th>
<th>Has significance</th>
<th>Has description</th>
<th>Has source</th>
<th>Has trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR</td>
<td>Modularity</td>
<td>M</td>
<td>Multiple tools reside on different layers. Separation of concerns.</td>
<td>R DoW A</td>
<td>R DoW A 20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW B</td>
<td>R DoW B 8</td>
</tr>
<tr>
<td>NFR</td>
<td>Open APIs</td>
<td>S</td>
<td>Provide open APIs between the architectural components that serve as a base for the overall extensibility.</td>
<td>R DoW A</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Real-time</td>
<td>M</td>
<td>Enable real-time communication.</td>
<td>R CTX C</td>
<td>R CTX C N1</td>
</tr>
<tr>
<td>NFR</td>
<td>Reliability</td>
<td>S</td>
<td>System performs reproducible actions as it is expected by the end-users.</td>
<td>R DoW A</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Responsiveness</td>
<td>S</td>
<td>Good response times</td>
<td>R DoW A</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Security</td>
<td>S</td>
<td>SMEs information is highly secured. Reassuring the user about security when they submit personal data and how it will be used.</td>
<td>R DoW A 22.1</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Stability</td>
<td>S</td>
<td>Architecture is stable over time, does not need major changes.</td>
<td>R DoW A 17.1</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Standard Compliance</td>
<td>M</td>
<td>Compliance to specific standards or formats. Processes comply to changing policies in guidelines.</td>
<td>R DoW B 3.1</td>
<td>R DI Pandora</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R CTX H</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>System Scalability</td>
<td>S</td>
<td>Infrastructure supporting dataflow coming from a growing number of tools.</td>
<td>R DoW A 16.2</td>
<td>R DoW B 10</td>
</tr>
<tr>
<td>NFR</td>
<td>Testability</td>
<td>N</td>
<td>Ensuring that the source code is extensively tested and results published for public consumption.</td>
<td>R DoW A 25</td>
<td></td>
</tr>
<tr>
<td>NFR</td>
<td>Ubiquitous Access</td>
<td>S</td>
<td>Ubiquitous access to resources. Inter sharing from different work locations</td>
<td>R DoW A 4.1</td>
<td>R CTX C 4.1</td>
</tr>
<tr>
<td>NFR</td>
<td>User Scalability</td>
<td>M</td>
<td>Scaling of the infrastructure to server a growing number of users (10-1000),</td>
<td>R DoW A 8</td>
<td>R DoW A 16.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R DoW A 8</td>
<td></td>
</tr>
</tbody>
</table>

**Has NFR number**

- NFR: Non-functional requirement
- NFRa: Non-functional requirement with a different source
- NFRb: Non-functional requirement with a different source

http://Learning-Layers.eu
Figure 23. Learning Layers SourceForge SVN repository structure
Appendix E2: Bits & Pieces HoQ

![Diagram of a triangle with rows and columns, detailed below.]

<table>
<thead>
<tr>
<th>Row</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.7%</td>
<td>1</td>
<td>User can view his/her resources</td>
<td>O</td>
<td>O</td>
<td>▲</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3.7%</td>
<td>1</td>
<td>User can view his/her shared resources</td>
<td>O</td>
<td>O</td>
<td>▲</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3.7%</td>
<td>1</td>
<td>User can filter resources by time, name, location, network, etc.</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3.7%</td>
<td>1</td>
<td>User can rate resources</td>
<td>O</td>
<td>O</td>
<td>▲</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3.7%</td>
<td>1</td>
<td>User can tag resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3.7%</td>
<td>1</td>
<td>User can comment resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3.7%</td>
<td>1</td>
<td>User can send messages</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3.7%</td>
<td>1</td>
<td>User can share resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>3.7%</td>
<td>1</td>
<td>User can sort resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3.7%</td>
<td>1</td>
<td>User can group (classify) resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>3.7%</td>
<td>1</td>
<td>User can set reminders to resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>3.7%</td>
<td>1</td>
<td>User can set reminders to semi-learning objects</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>3.7%</td>
<td>1</td>
<td>User can get recommendations on the level of semi-learning objects</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>3.7%</td>
<td>1</td>
<td>User can share individual resources</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>3.7%</td>
<td>1</td>
<td>User can share semi-learning objects</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>3.7%</td>
<td>1</td>
<td>Users can work on semi-learning objects collaboratively in real time</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>3.7%</td>
<td>1</td>
<td>User can track (his/her) activities in activity stream</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>3.7%</td>
<td>1</td>
<td>System tracks user interactions</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>3.7%</td>
<td>1</td>
<td>System provides recommendations</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>3.7%</td>
<td>1</td>
<td>System supports different views of visualization of semi-learning objects</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>3.7%</td>
<td>1</td>
<td>System provides different access levels (private, shared, public)</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>3.7%</td>
<td>1</td>
<td>System allows publishing of semi-learning object to external services (ontologies)</td>
<td>▲</td>
<td>▲</td>
<td>▲</td>
<td>▲</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>3.7%</td>
<td>1</td>
<td>User can assign semi-learning object to others as a backup</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>3.7%</td>
<td>1</td>
<td>User can request (view and semi-learning objects in personal or collaborative virtual language)</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>3.7%</td>
<td>1</td>
<td>User can share semi-learning objects</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>3.7%</td>
<td>1</td>
<td>Users can start and finish working sessions</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Target or Limit Value

- Low
- Medium
- High
- Very High

Difficulty

- Low
- Medium
- High
- Very High

Max Relationship Value in Column

| 0 | 3 | 8 | 9 | 3 | 8 | 1 | 9 | 9 |

Weight / Importance

| 125.9 | 33.3 | 125.9 | 200 | 25.9 | 218.5 | 0.7 | 100 | 217.4 |

Relative Weight

| 12.1% | 3.2% | 12.1% | 19.2% | 2.5% | 21% | 0.4% | 0% | 15.6% |

http://Learning-Layers.eu
## Appendix E3: Sharing Turbine HoQ

<table>
<thead>
<tr>
<th>Row</th>
<th>Feature/Functionality</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Appr. access learning materials</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Appr. record test progress</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Appr. review previously recorded materials</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Appr. store media (photo, video, sound, text)</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>Appr. share tasks</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>Appr. have a personal learning certificate</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>Appr. access personal learning certificate</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>Appr. receive feedback on submitted tasks</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>Appr. can complete tasks individually</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>Appr. can complete tasks as a group</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>Appr. can share experiences of tasks</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>Appr. can request help</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>Appr. have a personal profile</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>Appr. can exchange through course titles and content</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>Appr. can exchange through chat, live</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>Appr. can access a large knowledge database</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>Appr. have login security for their account</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>Markers can use learning materials</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>Works on both mobile and tablets</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>Can operate online and offline</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>Markers can access view apprentice records</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>Markers can create tasks and share without use of coding</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>Markers can share experiences through a social network</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>Markers can receive help messages from apprentices</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>Markers have more privileges than apprentice accounts</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>Markers have a communication system (live chat)</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>Markers can train a whole success rate progressions rates etc. as apprentices</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>▲</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Target vs Limit Value

- Easy
- Medium
- Hard

### Difficulty

- Very Low
- Low
- Moderate
- High

### Weighted Importance

<table>
<thead>
<tr>
<th>Technique</th>
<th>Weight</th>
<th>Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPSIS</td>
<td>280.9</td>
<td>100</td>
</tr>
<tr>
<td>AHP</td>
<td>157.5</td>
<td>51.6</td>
</tr>
<tr>
<td>ELECTRE</td>
<td>172.7</td>
<td>55.9</td>
</tr>
</tbody>
</table>

### Relative Weight

<table>
<thead>
<tr>
<th>Technique</th>
<th>Relative Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPSIS</td>
<td>16.7%</td>
</tr>
<tr>
<td>AHP</td>
<td>9.9%</td>
</tr>
<tr>
<td>ELECTRE</td>
<td>9.4%</td>
</tr>
</tbody>
</table>
## Appendix E4: PANDORA HoQ

<table>
<thead>
<tr>
<th>Row #</th>
<th>Max Relationship Value in Row</th>
<th>Row Weight</th>
<th>Weight Importance</th>
<th>Comment Count</th>
<th>Importance</th>
<th>Document Count</th>
<th>In Endnote/In Summary</th>
<th>In Short</th>
<th>In Main Text</th>
<th>Total Comment Count</th>
<th>Total Document Count</th>
<th>Target or Limit Value</th>
<th>Difficulty</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>11</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>12</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>13</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>14</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>15</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>16</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>17</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>18</td>
<td>9</td>
<td>5.5%</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

**Max Relationship Values in Column**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>3</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

**Weight / Importance**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>100.7</td>
<td>133.3</td>
<td>86.7</td>
<td>59.7</td>
<td>50</td>
<td>66.7</td>
<td>103.3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Relative Weight**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.3%</td>
<td>19.5%</td>
<td>15.9%</td>
<td>7.3%</td>
<td>4.5%</td>
<td>5.3%</td>
<td>7.8%</td>
<td>21.4%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>