Releases of the Learning Analytics, Scaffolding Services, and Add-on Services – initial

Tobias Hecking, Yiwei Cao

▶ **To cite this version:**

Tobias Hecking, Yiwei Cao. Releases of the Learning Analytics, Scaffolding Services, and Add-on Services – initial. [Research Report] Go-Lab Project. 2014. hal-01201982

HAL Id: hal-01201982
https://telearn.hal.science/hal-01201982

Submitted on 18 Sep 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Go-Lab

Global Online Science Labs for Inquiry Learning at School

Collaborative Project in European Union's Seventh Framework Programme
Grant Agreement no. 317601

Deliverable D4.4

Releases of the Learning Analytics, Scaffolding Services, and Add-on Services – initial

Editors	Tobias Hecking (UDE) Yiwei Cao (IMC)
Date	31st October, 2014
Dissemination Level	Public
Status	Final

©2014, Go-Lab consortium

The Go-Lab Consortium

Beneficiary Number	Beneficiary Name	Beneficiary short name	Country
1	University Twente	UT	The Netherlands
2	Ellinogermaniki Agogi Scholi Panagea Savva AE	EA	Greece
3	École Polytechnique Fédérale de Lausanne	EPFL	Switzerland
4	EUN Partnership AISBL	EUN	Belgium
5	IMC AG	IMC	Germany
6	Reseau Menon E.E.I.G.	MENON	Belgium
7	Universidad Nacional de Educación a Distancia	UNED	Spain
8	University of Leicester	ULEIC	United Kingdom
9	University of Cyprus	UCY	Cyprus
10	Universität Duisburg-Essen	UDE	Germany
11	Centre for Research and Technology Hellas	CERTH	Greece
12	Universidad de la Iglesia de Deusto	UDEUSTO	Spain
13	Fachhochschule Kärnten - Gemeinnützige Privatstiftung	CUAS	Austria
14	Tartu Ulikool	UTE	Estonia
15	European Organization for Nuclear Research	CERN	Switzerland
16	European Space Agency	ESA	France
17	University of South Wales	USW	United Kingdom
18	Institute of Accelerating Systems and Applications	IASA	Greece
19	Núcleo Interactivo de Astronomia	NUCLIO	Portugal

Contributors

Name	Institution
Tobias Hecking, Sven Manske, H. Ulrich Hoppe	UDE
Anjo Anjewierden, Lars Bollen, Jakob Sikken	UT
Yiwei Cao, Nils Faltin	IMC
Sten Govaerts, Andrii Vozniuk, Wissam Halimi, Christophe Salzmann, Denis Gillet	EPFL
Danilo Garbi Zutin	CUAS
Pablo Orduna	UDEUSTO
Sofoklis Sotiriou (internal review)	EA
Panagiotis Zervas (internal review)	CERTH

Legal Notices

The information in this document is subject to change without notice. The Members of the Go-Lab Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the Go-Lab Consortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material. The information and views set out in this deliverable are those of the author(s) and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

Executive Summary

This deliverable describes and demonstrates the initial release of learning analytics, scaffolding, and add-on services of Go-Lab. However, the main product of this deliverable are the software prototypes of the learning analytics, scaffolding, and add-on services, many of which are accessible online. Currently, we have realised the first prototypes specified in the initial specification of D4.2. It serves as a documentation of the development progress of the first prototypes. It also reports early evaluation results of the first prototypes.

This deliverable consists of two major parts: (i) the learning analytics and scaffolding services and (ii) the add-on services including a booking system and a tutoring platform (which was called the Bartering Platform in D4.2). The initial release of the learning analytics and scaffolding services comprise of the prototypical implementation of the action logging, back-end services and guidance mechanisms that have been specified in D4.2.

The add-on services prototype of the booking system includes a set of mock-up components, while the first prototype of the tutoring platform is accessible at `http://tutoring.golabz.eu`. The prototypes show that they are able to support the federation of online labs via the booking mechanism and to motivate the user community to interact on the portal actively via skills bartering.

Based on the experiences and prototypes described in this deliverable and our future work, the specifications of the learning analytics, scaffolding, and add-on services will be updated and finalised in D4.6 (M33). Furthermore, the evaluation of the first prototypes will contribute to the development improvement of the final release in D4.8 (M36).

Table of Contents

1 Introduction **8**

2 Learning analytics and scaffolding services **10**
2.1 Log data acquisition and privacy 10
2.1.1 Data format for action logging 10
2.1.2 Client libraries 11
2.1.3 Privacy mechanisms 11
2.2 Feedback mechanisms for guidance 14
2.2.1 Notification format 14
2.2.2 Client libraries 15
2.3 The Analytics Workbench 15
2.3.1 Modes of Execution 15
2.3.2 Deployment and Access 16
2.3.3 Analytics app creation and usage 16
2.3.4 Adaptation to Go-Lab 18
2.3.5 Usage Example 18
2.4 The learning analytics back-end services 20
2.4.1 Overview 20
2.4.2 Required technology 21
2.4.3 Interface specifications 22
2.4.3.1 Action Logging Service 22
2.4.3.2 Notification Broker 23
2.4.3.3 Analytics Service 23
2.4.4 The agent system 23
2.4.5 The shared memory 24
2.4.5.1 Shared memory templates 25
2.4.6 Agent prototypes 26
2.4.6.1 Concept Mapping Agent 26
2.4.6.2 Hypothesis Observer Agent 30
2.4.6.3 Reflection Agent 31
2.4.6.4 Wiki Agent 32
2.5 Demo ILS 33
2.6 Summary and outlook 34

3 Add-on services **36**
3.1 Introduction 36
3.2 Go-Lab Booking System 36
3.2.1 Architecture of Go-Lab booking system 36
3.2.2 Implementation 38
3.2.2.1 Book and use a Smart Gateway lab 38
3.2.2.2 Book and use a Smart Device lab 40
3.2.2.3 User interface of the calendar manager 40
3.2.3 Integration plan and open issues 41
3.3 Go-Lab Tutoring Platform 43
3.3.1 Architecture of the Go-Lab Tutoring Platform 44

3.3.1.1 Architecture refinement 44
3.3.1.2 Requirements review and implementation overview 44
3.3.1.3 Data model 45
3.3.2 Implementation of the Go-Lab Tutoring Platform 45
3.3.2.1 User interface 47
3.3.2.2 REST Service 51
3.3.2.3 Google Hangout 53
3.3.3 Evaluation of the Go-Lab Tutoring Platform 54
3.3.4 Open issues and future plan 57

4 Conclusion **59**

A Appendix A - Initial concept for message internationalisation **61**
References 62

1 Introduction

This deliverable reports on the first prototypes of learning analytics, scaffolding, and add-on services. For a better readability, we structure the deliverable in two parts: (i) learning analytics & scaffolding and (ii) add-on services. This structure is also in line with the initial specification D4.2.

According to the initial specification, we have developed a series of prototypes and collected some first-hand prototype development experiences. Based on these valuable practical experiences, we review the requirements, architecture and use scenarios of the specification in this deliverable D4.4. The feedback to the specification in D4.2 is positive, which will be reported in the two main parts in detail respectively below.

The initial release of the Learning Analytics and Scaffolding services comprises the prototypical implementation of the features described in D4.2. This includes mechanisms for action logging with special mechanisms to meet privacy and data security requirements, feedback mechanisms and analysis techniques. In particular, action logging mechanisms include client libraries that enable developers to easily integrate action logging in apps. Action logging mechanisms are accompanied by a concept for privacy which makes action logging controllable by the teacher. In order to fulfill the requirements for immediate system generated feedback for students the first implementation of a publish-subscribe based notification concept was implemented. This first prototype also includes the learning analytics back-end services specified in D4.2. These services have been designed as an agent system offering interfaces that can be accessed over the web. The integration of existing analytics tools is ongoing. Currently the Analytics Workbench (Göhnert, Harrer, Hecking, & Hoppe, 2013) has been adapted to the needs of the Go-Lab analytics services and functions as a tool for assembling analytics apps in a visual fashion.

As add-on services, we have released the lab booking system and a tutoring platform which was called bartering platform before in D4.2. We still have to formally decide on a final name of the tutoring platform within the Go-Lab project. The booking system implements the Go-Lab booking schemes and realised the three use scenarios respectively (Cf. Section 3.5.2.2 Interface specifications and detailed interaction between components in D4.2): (i) booking of a lab using Smart Device; (ii) booking of a lab using the Smart Gateway; and (iii) administrator of the booking calendar. Three components have been developed, as the mock-up demonstrators. Each of them works, while the integration work is still needed in the near future.

Last but not least, the first prototype of the tutoring platform is released at `http://tutoring.golabz.eu`. Using Google Hangout, it offers Go-Lab users a platform to exchange their online lab expertise, knowledge and skills among themselves using video conferencing.

This release deliverable illustrates that the initial specifications of D4.2 (M18) can be implemented and work. It offers the practical prototype evaluation to improve and finalise the specification in D4.6 (M33). Furthermore, the final

release will be described in D4.8 (M36).

2 Learning analytics and scaffolding services

This section highlights the functionality implemented for the learning analytics and scaffolding services based on the specification in D4.2.

2.1 Log data acquisition and privacy

In D4.2 the acquisition of log data that captures the student's actions in an ILS has been pointed out as a major prerequisite for learning analytics and scaffolding services. Hence, this section describes the implementation of the action logging in the Go-Lab portal.

2.1.1 Data format for action logging

As specified in deliverables D4.2 and D5.1 activity logs are encoded in the ActivityStream format.[1] A detailed description on the special properties of an ActivityStream object in Go-Lab can be found in D4.2 Section 2.4.6. In order to keep action logs semantically consistent and to ensure that the logs can be understood by different processing applications, a predefined vocabulary of verbs has been specified. In general, it is possible to send any valid activity stream to the learning analytics back-end services, however, to enable proper learning analytics functionality logs need to adhere to the logging vocabulary. There is a semantic distinction between content-oriented, process-oriented and storage-oriented actions. Table 1 lists the currently used verbs.

Semantics	Verbs
content-oriented	"add", "remove", "change"
storage-oriented	"open", "create", "update", "delete"
process-oriented	"access", "start", "cancel", "send", "receive"

Table 1. Verb vocabulary and semantics for action logs.

Content-oriented actions refer to the direct manipulation of artifacts. Typical examples are adding a concept to a concept map or editing a wiki page. The storage-oriented verbs are similar words but have different semantics. Storage-oriented actions are always related to persistency of artifacts. In general, whenever an action involves the ILS Artefact Storage Library (see D5.3) storage-oriented action logs will be created. The third group of verbs refer to actions that are part of general processes in an ILS. In contrast to content-oriented logs, which always imply the manipulation of an artefact, process-oriented logs are used in order to indicate that specific functionality of apps, labs, or spaces was used. These logs are typically more coarse grained than the content-oriented logs. Examples for process-oriented action logs are: a student accessed an inquiry learning phase, accessed an app, started an experiment, etc.

[1] `http://actvitystrea.ms`

2.1.2 Client libraries

The source code of the ActionLogger Javascript library for the implementation of action logging in apps can be found at: `https://github.com/go-lab/ils/tree/master/applog`. For a detailed description we refer to D5.3 and for a developer documentation to `https://github.com/go-lab/ils/wiki/ActionLogger`.

2.1.3 Privacy mechanisms

Recently, personal data privacy has received global attention, e.g. due to the revelations in the NSA scandal[2]. This made some users of online services much more concerned about their data privacy. They demand more transparency with regard to what personal information is collected, who collects and processes it and what for. Moreover, the users want to have more control over the data collection and processing policies.

To be able to control data privacy, it is important to understand the many aspects and definitions of data privacy. According to the state of the art analysis of data privacy done in the framework of the SPION project (Acquisti et al., 2011), two types of data privacy can be identified: (1) social privacy and (2) instrumental privacy. Raynes-Goldie (Raynes-Goldie, 2010) defines social privacy of users as "the control of information flow about how and when their personal information is shared with other people". As an example, social privacy can be achieved by introducing a trust-aware data sharing mechanism (Li, Najafian-Razavi, & Gillet, 2011). According to Boyd (Boyd, 2008), instrumental privacy is defined as the control of data access by corporations and government, for instance for data analysis or data mining. Instrumental privacy is the focus of this section.

Data privacy is regulated in different ways by many national governments and often depends on the target audience. For instance, in the case of schools and teachers, student data privacy is of particular importance, since student online activity tracking is subject to stronger legal privacy regulations, due to the age of school pupils. For instance, the European Union provides a data privacy framework, through the EU directives 95/46/EC (Data Protection Directive) and 2006/24/EC (Data Retention Directive).

The approach to managing activity tracking permissions implemented in the ILS Platform is called AngeLA (Vozniuk et al., 2014). AngeLA is motivated by the privacy mechanism and policy embedded into a physical classroom. In a physical classroom the teacher is in control of the privacy of what the students do in the classroom, e.g. she can decide which student behaviour she shares with the parents. If the teacher wants to discuss in private, she can ask all unwanted parties to leave the room. AngeLA mimics in a virtual space this privacy control mechanism that works well in a physical space. In a collaborative online space (e.g. an online learning space or a chat room), user management mechanisms exist to grant and revoke access to the space or resource. AngeLA is essentially a software agent that can be invited into an online space together with other members. When AngeLA is a member of an online space, AngeLA has

[2]The Guardian NSA Files, `http://www.theguardian.com/world/the-nsa-files`

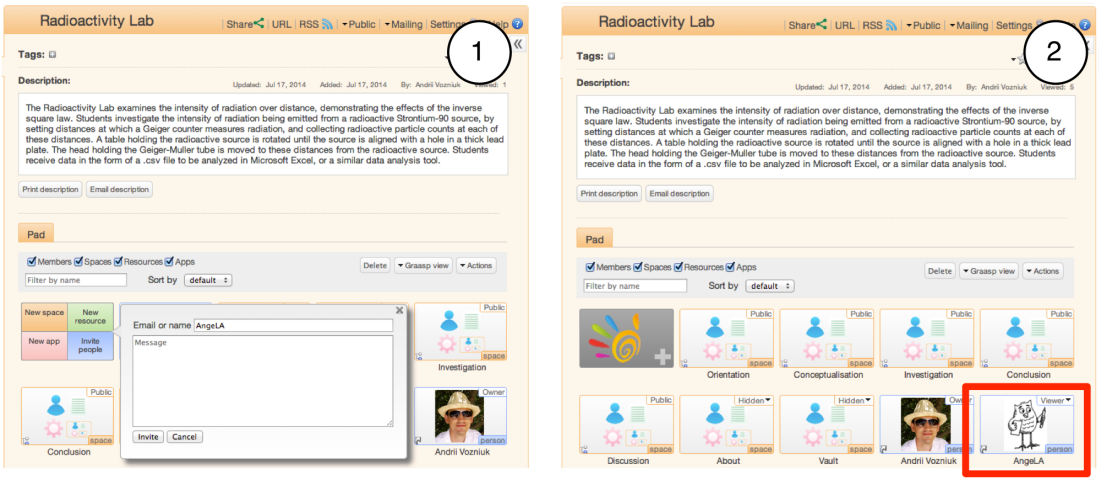

Figure 1. (1) AngeLA is invited to become a member of the "Radioactivity Lab" space in the ILS Platform (Graasp). (2) AngeLA is a member of the space.

access to any activity taking place in this space, just like any other member of this space and like the teacher in the classroom. AngeLA can thus collect all activities of the space members and can store them in the local database and, as in our case, can send this information to a third-party LA service. The owner of the online space can revoke AngeLA's access to the space, after which AngeLA can no longer track any user activity in this space. AngeLA's permissions can be configured per space depending on the activity context in the same way as a person can be invited to be present in some room and in the same time not invited to other ones.

By managing privacy via access control of AngeLA to an online space, the teacher is in full control of when the student activity is tracked and when not. This privacy control happens through already familiar user management functionality of inviting collaborators. Furthermore, by having the tracking agent as a visible space member, the teacher is aware of AngeLA's presence and hence that the tracking is turned on. Having easy privacy control and high visibility can also increase trust in the system due to being open about the tracking policy (Tsai, Egelman, Cranor, & Acquisti, 2007). With AngeLA we implement a soft paternalistic privacy policy, where the system does not force a user to make specific privacy decision, but rather make her aware about the ongoing activity tracking and provide an easy and intuitive way to change it.

We implemented the proposed approach in Graasp and integrated it into the Portal architecture (see Deliverable 5.4). The Learning Analytics Tracking Agent (AngeLA) architecture as presented in Figure 2 consists of the following three components:

Tracking Permissions UI. AngeLA aggregates user activities only from the learning spaces where it is a member. This provides an easy-to-use and familiar manner to manage privacy: (i) to enable the activity tracking in a space

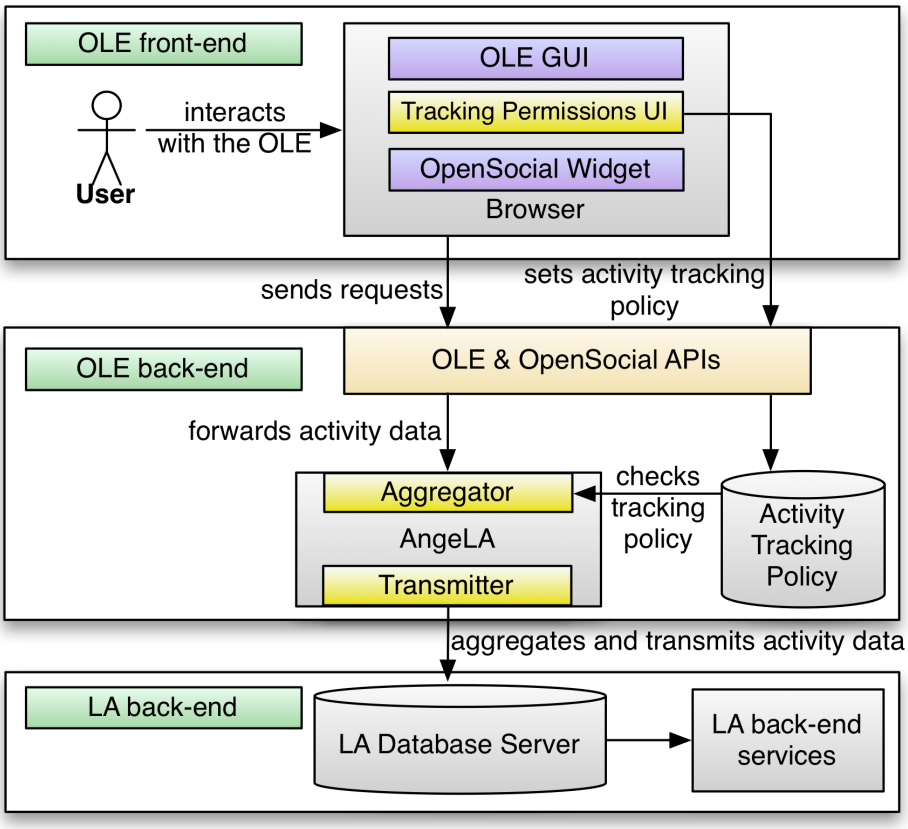

Figure 2. An architecture of the Learning Analytics Tracking Agent (AngeLA).

the teacher just needs to invite AngeLA to this space (see Figure 1 (1)) and (ii) to disable tracking the teacher can just removed AngeLA from the space. When AngeLA is present in the space (as in Figure 1 (2)), all the activities of space members will be collected, stored and sent to the LA back-end. This behaviour is intuitive and familiar for the teacher, since all space members are expected to be aware of the space activities.

User Activity Aggregator. AngeLA continuously aggregates activity streams of the users across the spaces where it is a member into a single activity stream. In the case of Go-Lab the data is coming from the following sources: (1) ILS Platform, i.e. Graasp, where the ILS authoring is done. (2) ILS Student View, where students interact with the platform. In the latter case the activity data is submitted from within OpenSocial apps via the Opensocial API (see Figure 2).

Activity Data Transmitter All the activity records collected are sent to the LA back-end (Hecking et al., 2014a) for further processing. The Activity Streams format is used to represent the actions during the transmission. As a mean to provide additional privacy in the Go-Lab portal, as proposed by Li et al. (Li, Holzer, Govaerts, & Gillet, 2014), students use nicknames instead of real names to represent their identity in an ILS. In this case only the teacher is able to do the mapping between the nickname and real student name and hence knows a

student's identity.

To enforce privacy it is important to setup a strict and clear default privacy policy (Lai & Hui, n.d.). In the ILS Platform we suggest teachers to decide if they want to have AngeLA as a member of a space upon its creation. In this way a teacher is able to define a clear permission policy for a space from the very beginning, before students start working with the space. After the space is created it is still possible to change the policy at any time by simply inviting or removing AngeLA from the space.

2.2 Feedback mechanisms for guidance

This section outlines the current solution for feedback mechanisms in the Go-Lab portal. This includes a format for notification messages that are created by the learning analytics back-end services and client libraries for apps.

2.2.1 Notification format

In order to enable immediate feedback for students by the system it is necessary to implement mechanisms that allow server-side analytics components to actively send messages to apps in the ILS platform (c.f. D4.2 in Section 2.4.6). Notifications can be feedback messages that may result in prompts on the client side, resource recommendations or re-configuration of tools. To make sure that these notifications can be handled by apps they have to be in an agreed format. The preliminary notification format has been described in D4.2. In this release deliverable the specific construction of a notification will be discussed more specifically. The following grammar specifies a valid notification object.

```
notification: type, importance, target, content
type: "prompt"|"resource"|"configuration"
importance: [1| ... |10]

target: type, id
target.type: "ILS"|"app"|...
target.id: uuid

content: (messageId, messageParams) | url | configuration

messageId: Id of the notification message
messageParams: Message parameters
url: Resource url
configuration: Configuration object for a specific app.
```

Listing 1. Simplified grammar for valid notification objects.

In order to make sure that a client app can be addressed uniquely by the server component the target id must be composed of the id of the current ILS, the id of the current user and the specific id of the current app instance.

2.2.2 Client libraries

The NotificationClient Javascript library supports app developers in subscribing apps as listeners for notifications that are generated by the learning analytics back-end services. When an app that uses the NotificationClient is started, the library needs to be instantiated. This will trigger the subscription process automatically. The required metadata for the subscription are extracted automatically from the ILS. No further effort of the app developer is necessary. Apart from the subscription of apps as recipients for notification messages the NotificationClient can also be used as a filter for messages. For this purpose the NotificationClient offers a *register(premise, handler)* method which accepts two callback functions as input. The first function *premise(message)* is called whenever a notification message is received. The *premise* function must check the message and return a boolean value to indicate if the message should be accepted or rejected. When a message was accepted by the premise function, the second callback function *handler(message)* is called with the message as parameter. The handler function contains the logic of how the message should be handled by the app, e.g. displayed as a prompt.

The source code of the NotificationClient library can be found at: `https://github.com/go-lab/ils/tree/master/appnotify`
Library documentation:
`https://github.com/go-lab/ils/wiki/NotificationClient`

2.3 The Analytics Workbench

The Analytics Workbench is a utility for creating and deploying analytics services for the Go-Lab portal. It has been developed during the EU project SISOB [3]. The original system was intended as a tool for data analysts but in an adapted version for Go-Lab the produced results can be fed back into the Go-Lab portal in order to support different stakeholders, such as teachers, students, lab owners, and scientists in the Go-Lab consortium. The basic idea of this subsystem has been explained in D4.2 and (Manske et al., 2014). In this section we will describe the technical concepts behind the implementation and point out the level of integration into the Go-Lab Portal. The usage of this tool will be demonstrated by an example. The Analytics Workbench follows a pipes and filters metaphor. An analyst can compose complex workflows by linking simple building blocks (filters) to longer execution chains. The user interface is depicted in Figure 3 with a visual representation of an analytics workflow.

2.3.1 Modes of Execution

One of the basic features that differentiate different types of Analytics services is the mode of execution and the perspective.

1. On demand:
    a) Embedded: Graphical representation: Analytics Services as Open

[3] SISOB: An Observatorium for Science in Society based in Social Models. FP7-SCIENCE-IN-SOCIETY-2010-1, Grant agreement no.: 266588

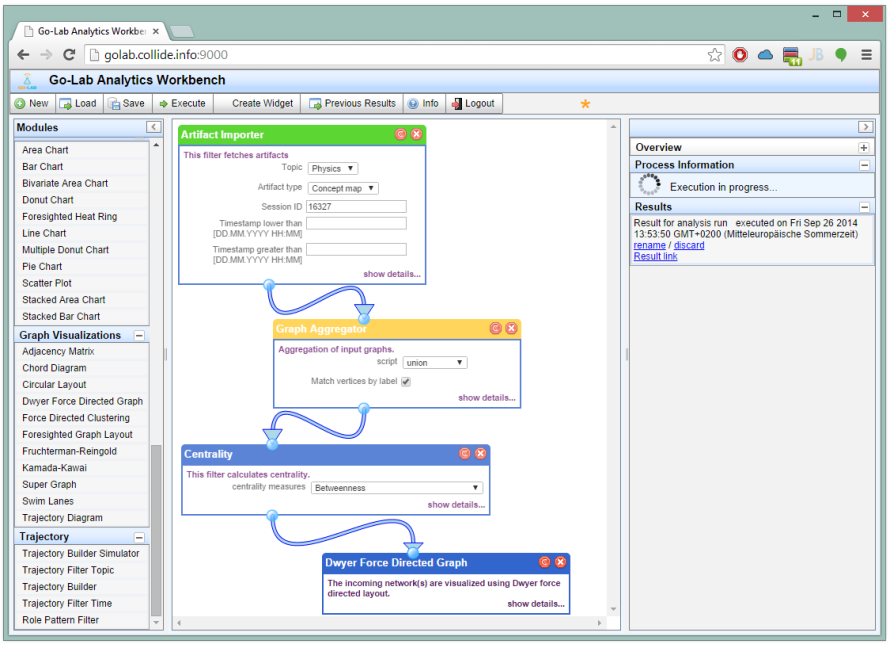

Figure 3. The user interface of the Analytics Workbench

Social apps

b) Standalone: workbench frontend

2. Cron jobs: Regular/timed execution of stored workflows

Services can be executed (1) on demand or (2) as a regularly re-occurring task (cron job). If a service should be executed on demand, the workbench offers two possibilities: an embedded approach where an app is created from a workflow and added to an ILS, or the execution of a workflow and the display of the results directly inside the workbench. Both approaches serve different purposes. While the first one is used to deliver visual representations of analysis results to the teachers, students or lab owners, the latter one can be used by experts for ex-post analysis and data investigation. The second mode can be used by analysts to define analytics workflows as regular and timed tasks.

2.3.2 Deployment and Access

The workbench is deployed together with the learning analytics back-end services. The frontend for the workbench for the authoring of Analytics Apps and Services can be found on `http://golab.collide.info:9000/`. The following account can be used:
Username: golab
Password: golab

2.3.3 Analytics app creation and usage

One main objective of the Analytics Workbench was the accessibility of analytics services for multiple stakeholders. A useful feature of the Analytics Workbench is that visually composed workflows as in Figure 3 can be serialised into a work-

flow description format. The execution engine of the system executes workflows based on those workflow descriptions. Thus, other components such as apps that are able to send workflow descriptions to the execution engine of the Analytics Workbench can trigger the execution of workflows as well. This allows for the (automatic) creation of analytics apps using the workbench user interface as a graphical composer (Manske et al., 2014). Figure 4 shows the three steps starting with the creation of Analytics Apps in the workbench with the goal of embedding such app into an ILS.

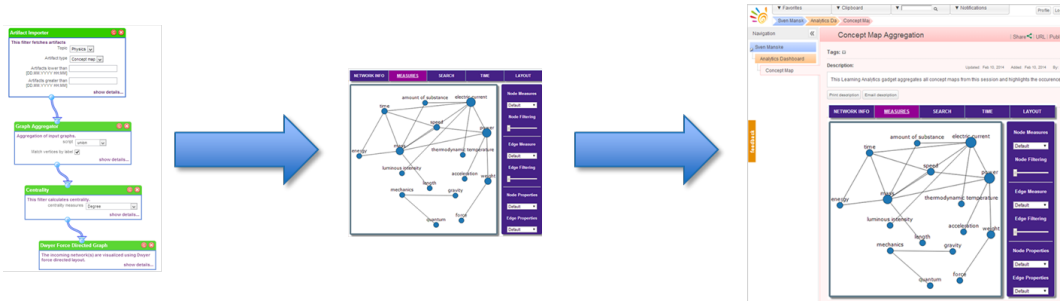

Figure 4. Creating analysis apps from workflows

The first step is the authoring of a workflow. Usually, this task is done by an analyst who knows how to assemble different components into a meaningful workflow. A workflow starts with the specification of a data source and usually ends with a visualisation of the results. In between, methods and algorithms for analysis can be specified. A data source can be – specific to Go-Lab – the database for the action logs or learners' artifacts, e.g. their concept maps created in an ILS (cf. D4.2).

While in execution mode 1 the composition of a workflow is done by an analyst, the visual representation of the results are dedicated to teachers or students. Therefore, the workbench is capable of creating OpenSocial apps, which can be embedded into the ILS Platform (cf. deliverable D5.1). These apps are able to trigger the execution of a workflow with given data and to display its result. The app file will be served statically from within the Analytics Workbench and is accessible via URL which can be used to embed the analytics app in the ILS Platform through the usual mechanisms. Section 2.3.5 illustrates this process by means of a practical example.

The created apps follow the OpenSocial standard. Therefore, stubs for such OpenSocial apps are enriched with workflow-specific parameters and an appropriate visual representation, selected during the authoring. A templating engine enhances an app template with three components: First, the interfaces for calling RESTful web services are added. A REST API for executing workflows through the web exists and accepts serialised workflows. Second, the serialised workflow (JSON format) is stored. Finally, the container context will be added. This context contains adapters for retrieving contextual parameters, such as a session id. In the ILS Platform, this would be filled with an identifier for the current space the app is running in. This enables contextual analytics, e.g.

monitoring of artifacts of a single ILS. Figure 5 shows this templating process.

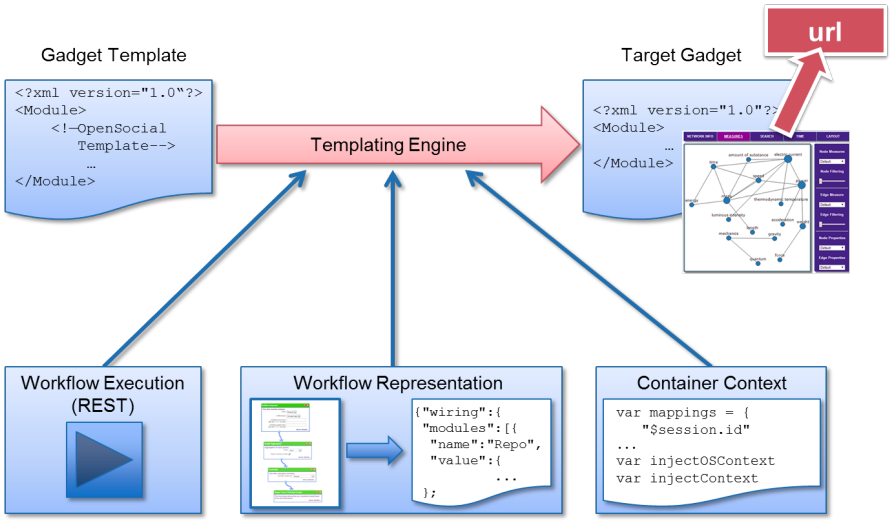

Figure 5. The templating process.

2.3.4 Adaptation to Go-Lab

The Analytics Workbench – in its original implementation – had the intention, to directly execute workflows and to act as a target platform for visualisations. The Go-Lab approach, however, requires adaptation to different execution modes and contexts. Thus, the model of the Analytics App export is a new contribution. Specific to Go-Lab, the target platform for the app export is the ILS Platform (Graasp) and therefore the system uses an OpenSocial template for the app creation (see D5.1).
In contrast to the regular workflow execution on demand triggered by the workbench front-end, the execution triggered by analytics apps need context information from an ILS. Therefore, the concrete workflow serialisation needs to have some degrees of freedom, namely the parametrisation of its context (ILS), which will be injected during runtime.
Figure 6 highlights the integration of the Analytics Workbench into the learning analytics back-end services and the front-end side. While the integration on the front-end is realised through the creation of OpenSocial apps, which can be integrated into the ILS Platform, the back-end integration is on the storage and communication layer. A common data source is used for data access and the shared memory component is used for the agent communication.

2.3.5 Usage Example

In this section, we present an example demonstrating the Analytics Workbench for authoring an analytics app and the usage of the created app inside an ILS. The goal is the aggregation of concept maps created by students into a single graph. The result has two possible recipients namely teachers and students. An app that displays an aggregated version of all concept maps created by the students can be fed into a teacher dashboard. The teacher can use this in order to get an overview of the collective knowledge of his or her students. On the

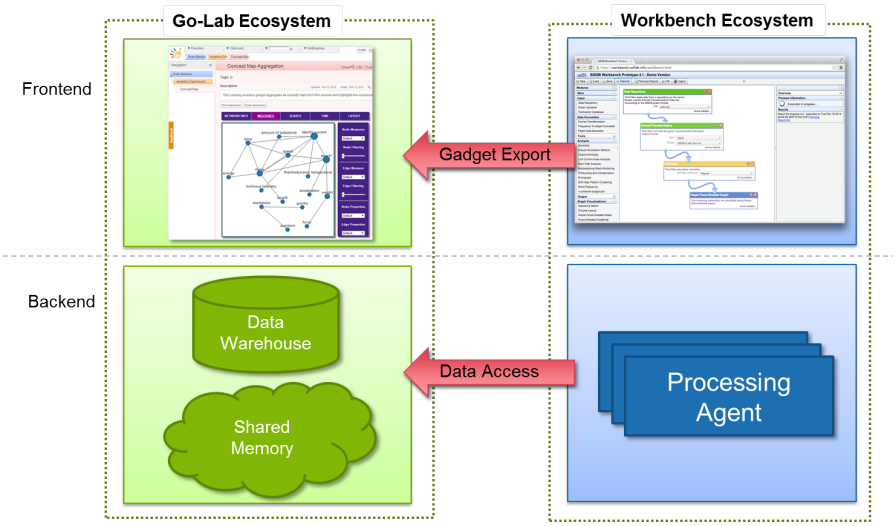

Figure 6. Integration in the learning anlytics infrastructure.

other hand it can also be used as an app for students that supports them in reflecting on their work and to foster discussions. Figure 7 shows the idea of this concept map aggregation to have a shared representation, which highlights frequently used nodes (concepts). A slightly modified workflow can be used to highlight the differences between the aggregated concept map and another individual or reference map.

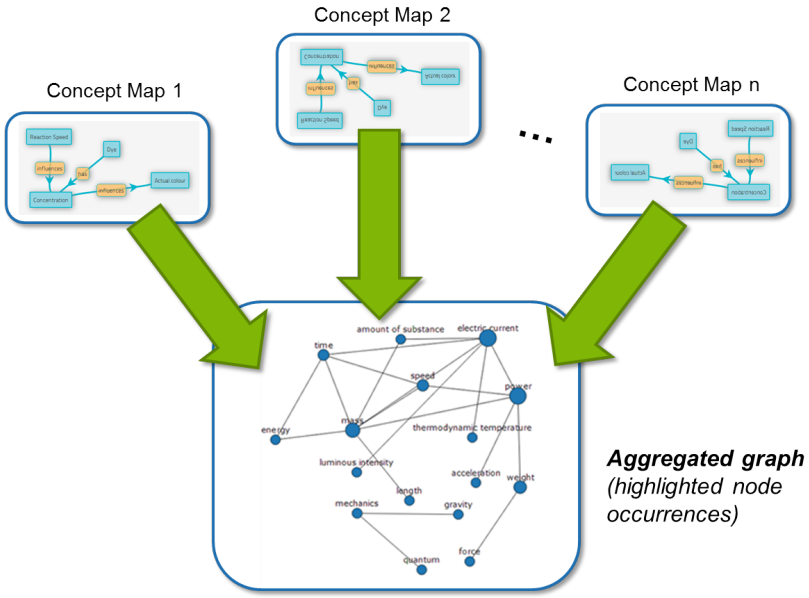

Figure 7. Aggregation of several concept map to a super-graph.

The description of this example will be aligned to the aforementioned three step process.

a. Workflow creation

The front-end of the Analytics Workbench will be used to create the workflow

representing the different analysis steps. For the purpose of aggregating concept maps a workflow must include (1) the import of the concept maps (as graphs) of a certain ILS, (2) the aggregation of the graphs representing the concept maps, (3) the application of network measurements to highlight nodes according to metrics, and (4) the specification of a graph visualisation. Figure 8 shows the graphical definition of the workflow.

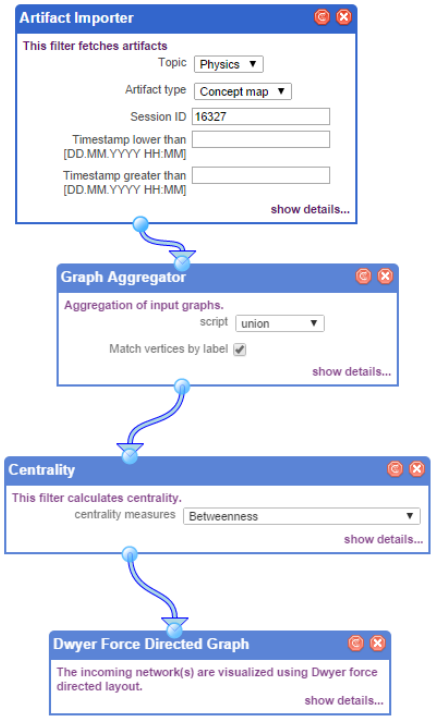

Figure 8. Workflow for concept map aggregation.

b. Created app. After the workflow has been created, the gadget can be created. According to the templating engine and the external formats for workflows, an app has been built and is served on the workbench server with a unique URL, making it accessible for embedding into external systems. The app itself contains the logic for executing a workflow, which is contained inside in its serialized (JSON-based) and parametrised format.

c. Embedding into an ILS. With the URL of the app, it can be used to be integrated into the ILS Platform. The contextualization of such app (if used) defines and restricts the access to the data sources. If the app is embedded into an ILS, it can only read the concept maps that belong to it. Figure 9 shows the "Concept Map Aggregation" app inside the ILS Platform.

2.4 The learning analytics back-end services

2.4.1 Overview

The main functional requirements for the Go-Lab learning analytics infrastructure have been specified in D4.2. The learning analytics infrastructure must offer facilities for action logging, immediate user feedback, and data storage

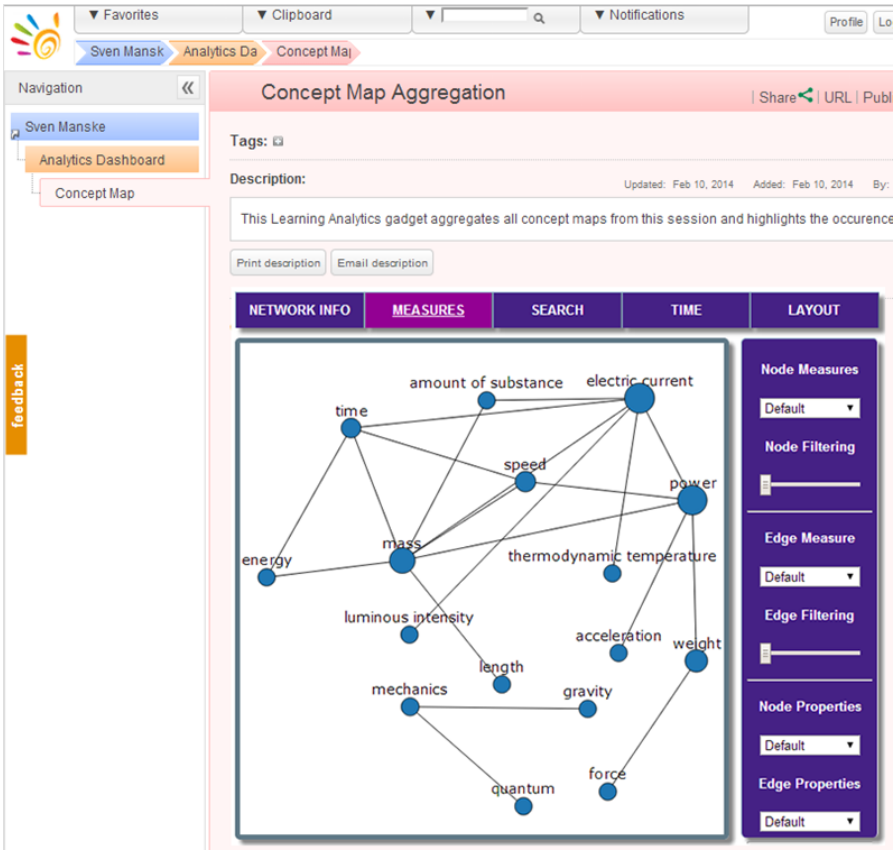

Figure 9. The result of the workflow is displayed inside an app in the Go-Lab portal.

and analysis. The learning analytics back-end services (Hecking et al., 2014b) meet these requirements by providing loosely coupled components for data acquisition, analysis and feedback mechanisms. The communication with other components, e.g. apps in the ILS platform, is ensured through predefined interfaces that are accessible over the web. These interfaces are depicted in Figure 10 and will be described in Section 2.4.4. The communication between the interface components and other back-end components is mainly achieved by establishing an agent system with components communicating over a shared memory. The concrete implementation of this agent system will be described in Section 2.4.3.

2.4.2 Required technology

SQLSpaces

SQLSpaces is an implementation the Tuple Spaces concept (Gelernter, 1985). The SQLSpaces server and client libraries are documented and can be downloaded at `http://projects.collide.info/projects/sqlspaces`. The server is distributed under the LGPL v3 licence. Client libraries are available for various programming languages and are distributed under the AGPL v3 licence. As described in deliverable D4.2, SQLSpaces are used in order to realise the shared memory as an active component for coordination and data exchange of

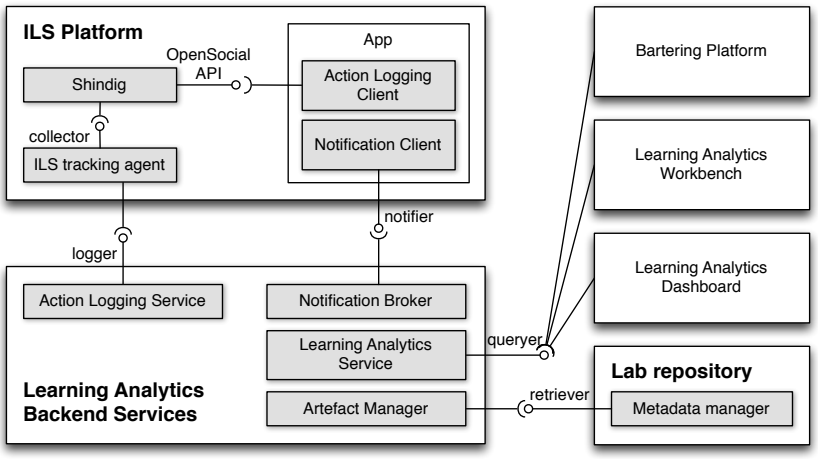

Figure 10. The learning analytics back-end services and interfaces to other components.

the agent system. The SQLSpaces concept has been successfully applied in the former EU project SCY [4].

MongoDB
MongoDB is a NoSQL database that is used as persistent long-term data storage for action logs and user generated artefacts like concept maps. It is available at `http://www.mongodb.org/` under the Creative Commons licence.

Analytics Workbench
The Analytics Workbench (Göhnert et al., 2013) is currently under development. It is planned to transform the project to an Open Source project in the future. For now, the built components as well as the source code are available upon request (see `http://workbench.collide.info` for details). The frontend of the workbench has been adapted to Go-Lab. The sources can be found on `http://projects.collide.info/svn/golab-wb/`.

2.4.3 Interface specifications

2.4.3.1 Action Logging Service

As specified in D4.2 the Action Logging Service is the central interface to which applications can send action logs. Incoming logs are stored persistently for long-term analytic purposes. Furthermore, the log object is written into the SQLSpaces which is a central coordination component for the agent system (see Section 2.4.5. Other components can register as listeners at the SQLSpaces and can directly react on log events. The Action Logging Service accepts action logs encoded in the JSON based activity stream format that are sent to the web service URL: `http://golab.collide.info/activity`. Action logs are accepted as valid as long as they contain an actor and a verb. A validity test application for action logs is available at `http://golab.collide.info/`.

[4] SCY: Science Created by You. FP7-FP7-ICT-2007-1, Grant agreement no.: 212814

2.4.3.2 Notification Broker

The Notification Broker implements a publish subscribe mechanism using WebSockets, in particular socket.io[5]. This enables apps which use the learning analytics back-end services to subscribe as listeners for certain feedback messages. The WebSocket endpoint is available at `http://golab.collide.info:80`. The subscription and the listening process can fully be handled by the NotificationClient library (see Section 2.2.2) so that app developers do not need to establish the socket connection manually. Technically the Notification Broker is connected to the SQLSpaces as shared memory (see Figure 11). In order to send a message to a client, the message creating instance does not use the Notification Broker directly. When an analytics agent sends a notification to a client, it writes it into the shared memory. In Table 2 the specific form of a notification entry will be described in more detail. The notification entry contains the necessary information to identify a client, namely ILS id, user id, and app id. The Notification Broker reacts on such notifications and publishes them on the corresponding socket channel. The strict separation of the message creator and the message sender with the shared memory as mediating instance leads to large flexibility and adaptivity of the implementation of the Notification Broker.

2.4.3.3 Analytics Service

The analytics service provides web service endpoints for applications requesting analytics functionality from the learning analytics back-end. Components that are not part of the learning analytics back-end can use this service in order to enable analytics supported features. As depicted in Figure 10 the analytics service endpoints are intended to be used by the tutoring platform, the Analytics Workbench (Section 2.3) and analytics dashboards. In the current prototype there are endpoints for the Analytics Workbench and learning resource recommendations based on text. The latter is currently used by the Wiki app and will be explained more concretely in Section 2.4.6.4.

2.4.4 The agent system

In D4.2 an agent system has been specified which allows for the easy implementation of analytics and intervention mechanisms as agents. Figure 11 summarises the general architecture. The communication of the analytics agents with the Action Logging Service and the Notification Broker is established by message exchange over a shared memory as a central component. The current implementation of the shared memory is based on the Tuple Spaces concept (Gelernter, 1985), and in particular the SQLSpaces implementation (Weinbrenner, 2012). The message exchange is based on writing and reading tuples to and from the shared memory. Analysis agents can specify templates for specific tuples and become actively notified whenever a tuple is written that matches the template. The concrete handling of the shared memory will be described in more detail in Section 2.4.5. The currently available prototypes of the analytics agents will be described in Section 2.4.6.

[5] `socketio`

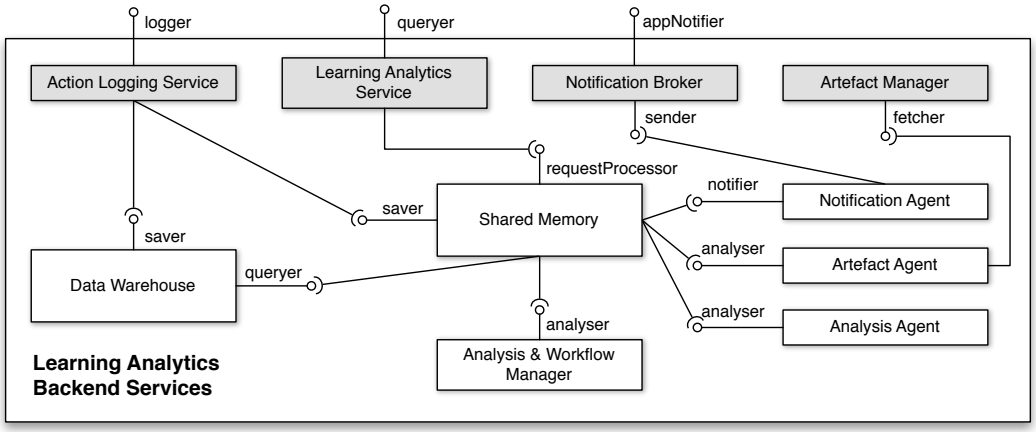

Figure 11. Components of the learning analytics back-end services.

2.4.5 The shared memory

Even if it has been decided to use the SQLSpaces implementation of the tuple spaces concept for the shared memory (see D4.2), the system in general is designed in a way that allows for the use of different shared memory providers. Thus, the communication to the shared memory can be described regardless of the underlying implementation of the shared memory. As already mentioned above, the basic communication via the shared memory is based on the exchange on tuples. Clients connected to the shared memory can write tuples into the shared memory that can be read by other clients. The retrieval of tuples from the shared memory follows the query by example paradigm. Clients retrieve tuples from the shared memory that matches the given tuple template. The SQLSpaces additionally provides a mechanism for active notifications. Clients that register to the shared memory as listeners for certain tuples must specify templates of relevant tuples. Whenever a tuple is written into the shared memory that matches the template the client becomes actively notified. An example is the processing of incoming action logs. The Logging Service, maps the incoming logs to tuples and forwards these to the shared memory interface. The shared memory interface then maps the log to an appropriate tuples format (depending on the shared memory implementation) and then writes it into the shared memory. Agents can register as listeners for action logs by specifying a template how the relevant logs should look like.

All shared memory implementations have to implement the SharedMemoryInterface. This interface contains functions such as connect/disconnect, read-/write, registerEvent as well as template generation functions.

This mechanism is depicted in Figure 12.

The following section summarises the tuple templates that are currently used by the described prototype.

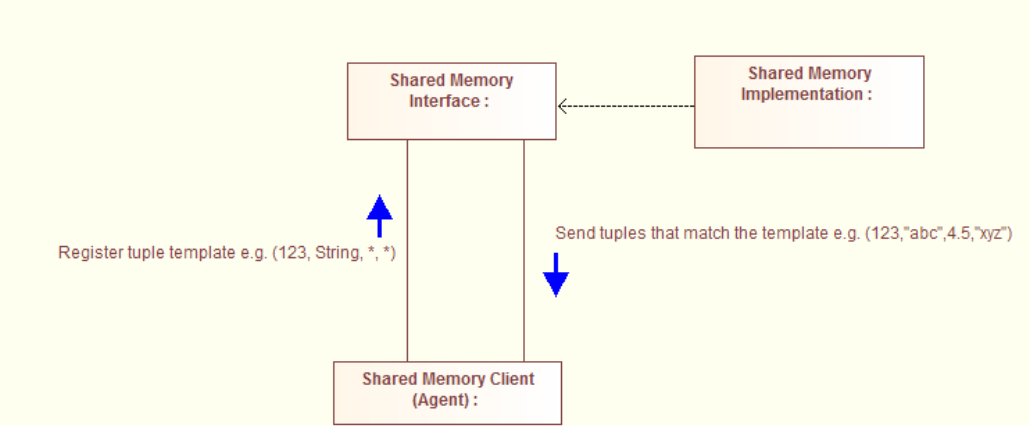

Figure 12. Communication by tuple exchange.

2.4.5.1 Shared memory templates

The following Table 2 describes the currently used templates used for communication via the shared memory.

Type	Structure	Description
Log Tuple	(id, clientType, content)	This type of tuple represents an action log received by the Action Logging service. The **id** parameter is a unique id for the tuple. The field **clientType** is extracted from the action log and specifies the log generating instance, for example, “Hypothesis App”. This enables the analytics agent to listen for action logs generated by a specific type of application by registering via a corresponding template. The third field **content** contains the actual action log in Activity Stream format.
Notification Tuple	(clientId, type, importance, target, content)	Notification tuples are used to trigger the distribution of feedback in the form of notifications to subscribing apps in the ILS platform. The tuple structure defines the content as well as further information about a notification message. The **clientId** field identifies the recipient of the notification. The **type**, **importance**, **target** and **content** corresponds to the fields of the notification messages in the format described in Section 2.2.1.

Table 2. Tuple specifications for component communication in the learning analytics back-end.

2.4.6 Agent prototypes

2.4.6.1 Concept Mapping Agent

Function

The concept map observer agent can be used to monitor the development of concept maps. This agent listens to action logs coming from the concept map app (D5.3). Based on these logs the agent maintains a server side copy of the concept map for each user of a certain ILS. It enables two different kinds of automatic interventions. According to predefined intervention rules the agent can send messages with concept suggestions or initiate a reconfiguration of the concept mapping app by providing a list of possible concepts that can be selected by the student. The agent is configurable by a configuration file named config.js

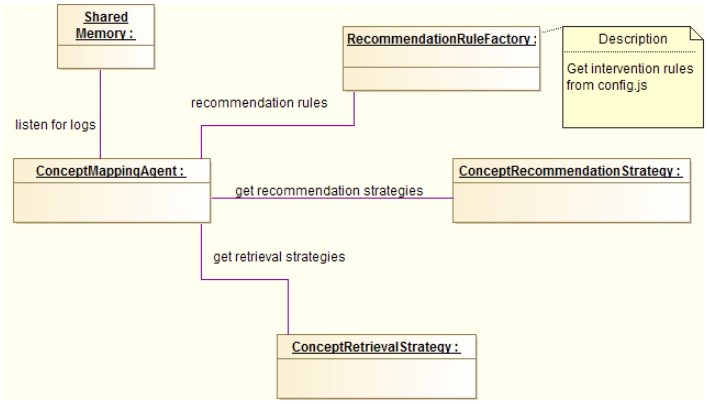

Figure 13. Components related to the Concept Mapping Agent

Integration into the Learning Analytics back-end

The agent is registered at the shared memory as a listener. It reacts on action logs with the client type “concept map” and registers as a log listener at the shared memory by creating a log listening template according to Table 2. The agent is configurable with respect to the intervention rules, the retrieval strategies of possible concept suggestions and recommendation strategies (see Figure 13).

Intervention rules

It is possible to configure different intervention rules that define when to trigger feedback messages. In the current implementation two rules are available which are explained in Table 3. In order to create intervention rules the RecommendationRulesInterface should be used. The interface specifies two functions that can be used to trigger effects:

- *needConceptRecommendation(data)*
- *needConfigurationRecommendation(data)*

These functions are called with data and return a boolean value. The data type is dependent on the actual implementation and can be, for example, number of actions or the current concept map.

Intervention rule	Function
Action count based intervention	A concept recommendation or an app reconfiguration is triggered after n actions on the concept map. This strategy ignores the structure of the concept map and should only be used for development purposes.
Concept count based intervention	Recommendations and reconfigurations are created when n concept nodes are added to the map.

Table 3. Intervention rules for the Concept Mapping Agent

Concept retrieval strategies

Table 4 explains different concept retrieval strategies. Before a concept recommendation can be initiated possible candidate concepts have to be retrieved. This is also configurable by setting the "conceptRetrievalStrategy" property in the config.js file. Currently three strategies are implemented. For examples and testing purposes there is a fixed list of candidate concepts. A more sophisticated strategy is to retrieve the keywords specified for the particular subject in the lab from the Lab Repository. This however, requires knowledge about the subject domain of the underlying ILS which is not always the case. The third concept retrieval strategy is to use the DBPedia ontology. This ontology is extracted from Wikipedia and is available in various languages. It maps the category tree of Wikipedia which allows for finding related concepts.

Concept retrieval strategy	Function
Fixed list of concepts	The Concept Mapping Agent maintains a fixed list of concepts. Recommendations are then made based on a static list. This strategy is not recommended to be used in production because it does not allow to adapt to different subject domains. However, it is useful for testing and development purposes.
Concept retrieval from the Lab Repository	The Lab Repository contains keywords for different lab and apps. When the subject domain of the ILS that contains the concept mapping app is known to the learning analytics services, the access endpoints to the Lab Repository can be used by the ConceptMappingAgent to query related keywords from the Lab Repository.
Concept retrieval from DBPedia	DBPedia (Auer et al., 2007) is an open ontology derived from Wikipedia in various languages. The DBPedia DAO of the Artefact Retrieval Service offers methods that can be used to map concepts to DBPedia resources and to retrieve related concepts based on the DBPedia category tree.

Table 4. Concept retrieval strategies for the Concept Mapping Agent

Concept recommendation strategies
After the set of candidate concepts has been created by the ConceptRetrievalStrategy one or more concepts have to be selected from this set for the actual recommendation. Currently two recommendation strategies are implemented and are listed in Table 5.

Concept recommendation strategy	Function
Random	Selects concepts randomly from a list of candidate concepts.
Frequency based	Selects the concepts that have to most relations to concepts already existing in the concept map.

Table 5. Concept recommendation strategies for the Concept Mapping Agent

Usage example
In Figure 14 an example of the usage of the Concept Mapping Agent is given. When a student clicks on a concept node, a list of concept suggestions appears which has been generated by the agent.

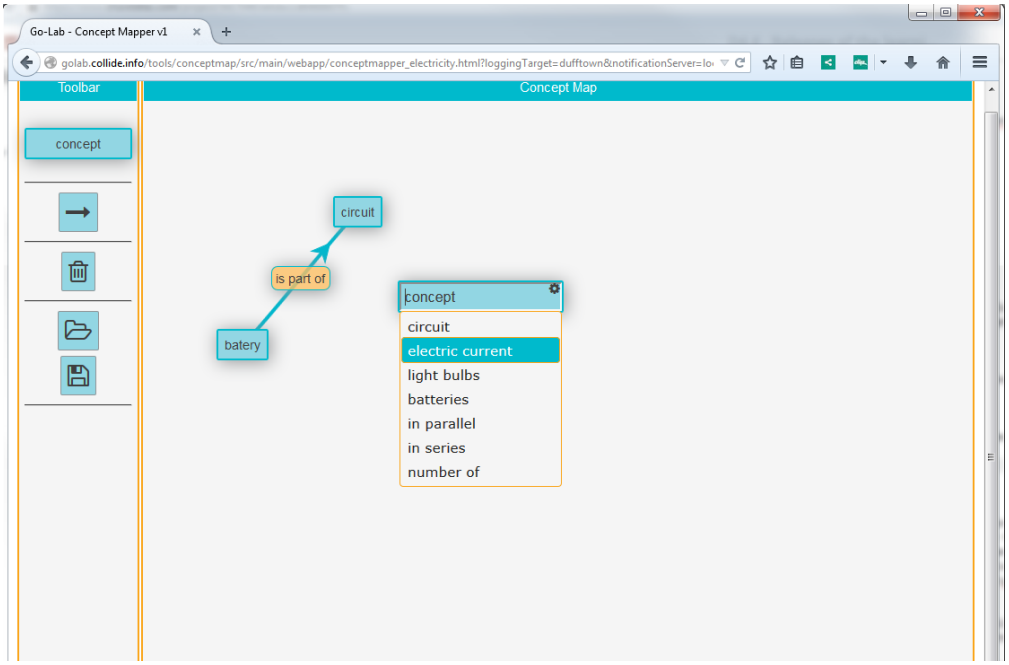

Figure 14. Recommendation of concept lists.

2.4.6.2 Hypothesis Observer Agent

Function

The Hypothesis Observer Agent can be used to investigate the hypotheses created by students in an ILS. When a student uses the Hypothesis Scratchpad (D5.3) all modifications of hypotheses are continuously logged and send to the Activity Logging Service of the learning analytics back-end services. If the Hypothesis Scratchpad is used together with the Concept Mapping app in the same ILS, the hypothesis observer compares the created hypotheses with the created concept map. A hypothesis usually has the form “If X then Y”, for example, “If electric current increases then voltage decreases”. This induces a directed relationship between concept X and concept Y, but not necessarily from concept Y to concept X. The Hypothesis observer checks the direction of the relations in the hypothesis and the concept map. If concepts are related in different directions in a hypothesis and the concept map, a message is created that suggests the student to have a closer look on the relation between the two concepts.

Integration into the Learning Analytics back-end

The agent is registered at the shared memory as a listener. It reacts on action logs with the client type “hypothesis”. The component architecture is depicted in Figure 15. In order to compare hypotheses and concept maps it has a direct relation to the Concept Mapping Agent. The ConceptMappingAgent class manages all currently active concept maps for a user. Based on the unique identifier of a user within an ILS, the HypothesisAgent can retrieve the corresponding concept map for the comparison with the hypothesis of the user.

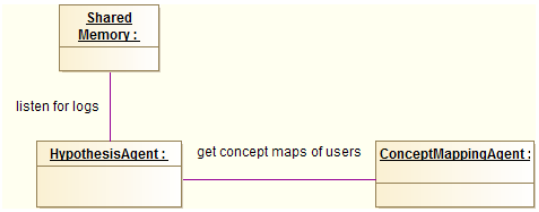

Figure 15. Components related to the Hypothesis Agent

Usage example Figure 16 depicts a typical intervention triggered by the Hypothesis Agent. The concepts mass and density are linked in different directions in the concept map and the hypothesis. This results in a message prompt generated by the Hypothesis Scratchpad app. This intervention is intended to stimulate students to think more intensively about their hypotheses and to avoid early misconceptions of an experiment. How students should deal with this concretely, however, depends on the pedagogical scenario and the instructions of the teacher.

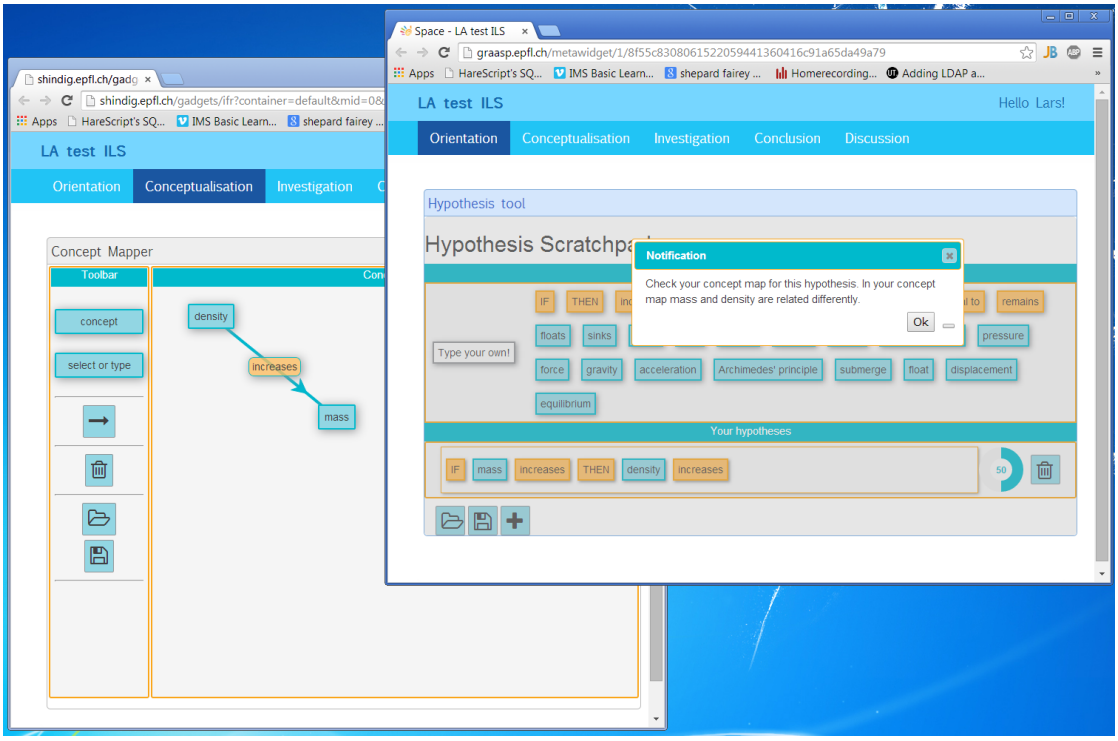

Figure 16. Intervention triggered by the Hypothesis Agent.

2.4.6.3 Reflection Agent

Function
One of the proposed tools is a reflection tool in which students can see how much time they spend in various phases of the Go-Lab inquiry cycle (see D1.1). The reflection tool creates a visualisation based on data provided by a reflection

agent. As proof of concept, a prototype of a reflection agent has been developed that listens to actions performed in the EDT (Experiment Design Tool). The EDT has, similar to an ILS, several phases (design, run, analyse).

Integration into the Learning Analytics back-end
Like the agents listed above, the reflection agent registers itself with the shared memory (SQLSpaces) and listens for actions logged by the EDT.

Usage example Each time an EDT action is found, the time spent by a user for the phase is updated and a notification message is forwarded to the prototype reflection tool which visualises the modified reflection information (see Figure 17).

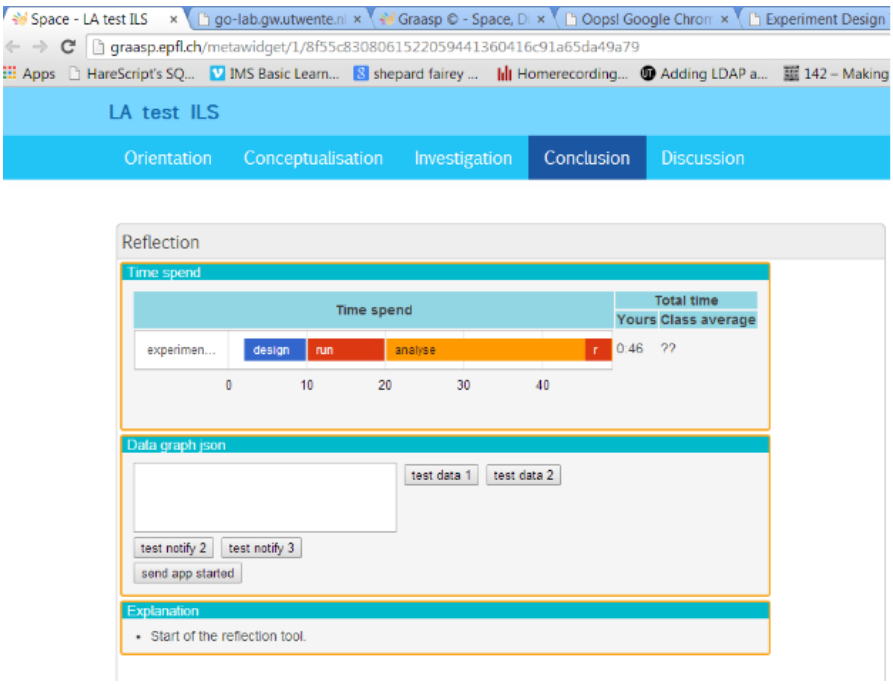

Figure 17. Prototypical version of the reflection tool.

2.4.6.4 Wiki Agent

Function
The Wiki Agent differs from the previous agent in the sense that it does not actively trigger interventions. Instead of monitoring the students activities based on action log data, it performs analysis and scaffolding on demand. The agent works in conjunction with the Wiki app (D5.3). The app offers the functionality of displaying web resources related to a particular wiki page upon user request. The creation of wiki pages is one possible activity in the conclusion phase. The presentation of related web resources can help to create curiosity and show possible further learning activities. The Wiki Agent can retrieve these related resources from the text of a wiki page independently from its language. When a user requests resource recommendations for a particular wiki

page, the app sends the complete textual page content to the Analytics Service interface. In particular, it is the web-service at `http://golab.collide.info/getResourceRecommendations`. The service passes the request to the Wiki agent, which starts the following three step procedure: (1) Determine the language of the text using bigram analysis (Collins, 1996). (2) Extract important concepts from the text and map them to resources in DBpedia using the DBPedia spotlight service (Mendes, Jakob, García-Silva, & Bizer, 2011). Different query endpoints for different languages are available. (3) Request corresponding Wikipedia pages and related web resources for the most often used concepts in the text from DBPedia.

The Analytics Service respond to the Wiki app with a list of links of the web resources and corresponding screenshots of the web pages, which can then be displayed to the user by the wiki app.

Integration into the Learning Analytics back-end The recommendation of web resources based on the textual content of a wiki page involves three components (see Figure 18. The aforementioned Analytics Service (see Section 2.4.3.3) is the endpoint for the Wiki app to retrieve the resources. The WikiAgent class performs the three steps described before. The communication with the DBPedia endpoint is encapsulated in a data access object named DBPediaDAO. It is involved in the retrieval of the actual resources from DBPedia.

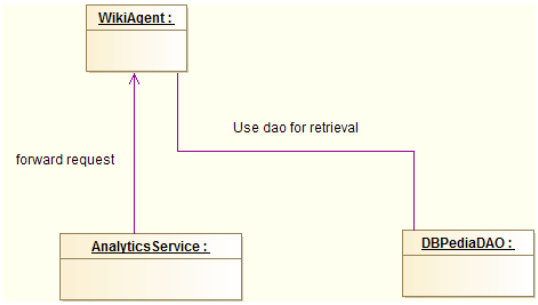

Figure 18. Components related to the Wiki Agent

Usage example One application of the Wiki app is to use it in the conclusion phase. Students can write their main conclusions and share their ideas. Based on the produced text further learning resources such as Wikipedia articles can be recommended in order to give ideas for further learning activities. Figure 19 shows a short text in the Wiki app and the corresponding recommendations generated by the Wiki Agent. Apart from the usage as a support tool for students, the same logic can also be used in order to support teachers in finding appropriate additional learning material.

2.5 Demo ILS

For demonstration and testing purposes a demo inquiry learning space has been set up. This demo ILS is available at `http://graasp.epfl.ch/metawidget/1/4bc1bf1c7963650fdd5020ff131abfd24d74e6c6`. The space is designed for

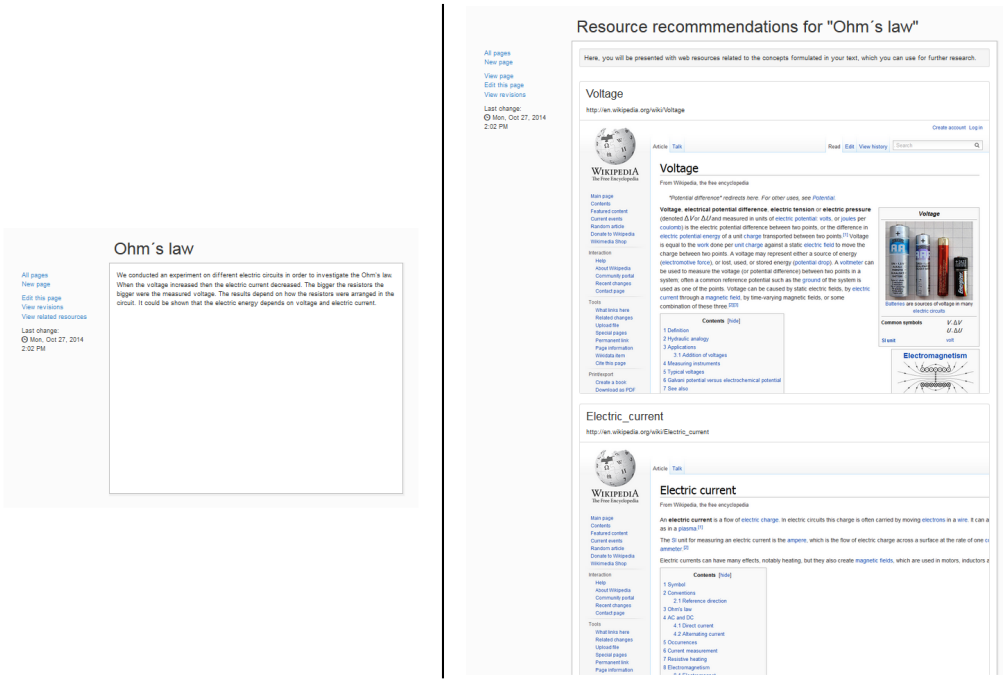

Figure 19. On the right hand, the reading recommendation based on the text on the left side.

online experiments on Ohm's law. This space integrates analytics supported features into a real scenario in order to demonstrate and test them. The demo ILS currently includes:

- Recommendation of concepts during concept map creation (Section 2.4.6.1).
- Checking of hypotheses by comparison with concept maps (Section 2.4.6.2).
- Resource recommendations based on texts created with the wiki app in the conclusion phase (Section 2.4.6.4).
- Concept map aggregation for self-reflection support (Section 2.3.5).

2.6 Summary and outlook

This chapter on the learning analytics and scaffolding services has given descriptions of implementation details of the current prototype. In the current state the logging of student actions is fully integrated into the Go-Lab portal. The learning analytics back-end services are deployed and its added value is demonstrated in the demo ILS described in Section 2.5. In the future, action logging and notifications have to be integrated in most of the apps and online labs. A tutorial for developers on how to integrate logging can be found at `https://github.com/go-lab/ils/wiki/ActionLogger-Tutorial`. Regarding the feedback mechanisms the next prototype will include internationalisation of textual feedback messages. The general concept for that can be found in appendix A. Apart from that, the next steps will be the conceptualisation and realisation of self-reflection dashboards for students and monitoring tools for

teachers. This will be done in coordination with the pedagogical project partners.

3 Add-on services

3.1 Introduction

As specified in D4.2, the Go-Lab Add-on Services consist of two main platforms, the Booking System and the Bartering Platform. We have renamed the Bartering Platform to the Go-Lab Tutoring Platform preliminarily in order to clarify for teachers the purpose of this platform. A final decision on the name needs to be taken in the near future by the Go-Lab General Assembly. This deliverable presents the first prototypes of the Booking System and the Tutoring Platform.

The Booking System is still in an early state due to the necessary integration with different labs and few labs are ready for such integration at this time. However this deliverable demonstrates the feasibility to implement and integrate the Booking System. The demonstrators are mock-up services of all scenarios, the Smart Gateway mock-up booking service, the Smart Device mock-up booking service, and the user interface including calendar management. The Tutoring Platform prototype is a first fully working version of the specifications laid out in D4.2.

The remainder of this section elaborates on the releases of both platforms, their functionality, development progresses, and future development plans. Furthermore, early evaluation results of the Tutoring Platform are presented. The demonstrators of the booking system are several links presented in each section. The first prototype of the Go-Lab Tutoring Platform is accessible at `http://tutoring.golabz.eu/`

3.2 Go-Lab Booking System

As specified in D4.2 Section 3.5.2, we have proposed the Go-Lab booking schemes to handle lab booking of different types of online labs. Go-Lab booking is a complex task which involves many different components, such as the Go-Lab Smart Gateway (see D4.1 & D4.3). It also needs to deal with the diversity of booking mechanisms of existing online labs.

3.2.1 Architecture of Go-Lab booking system

The proposed architecture in D4.2 has specified the components of the Go-Lab booking system and its interactions with other components. As an add-on service, the Go-Lab booking system offers booking search, a calendar manager, a booking manager, and a notification manager. Teachers use the Go-Lab Portal, search the labs, and book the labs use through the booking system (booking search, calendar manager and booking manager). Lab owners use the Go-Lab Portal to manage the lab usage time via the booking system (calendar manager). The booking system uses the back-end booking service developed in the Smart Gateway and Smart Devices (booking manager).

Recalling the Go-Lab booking schemes in D4.2, we focus on the Go-lab booking and extend it with main components and demonstrators in Table 6.

The implementation progress of each demonstrator is described in the next sec-

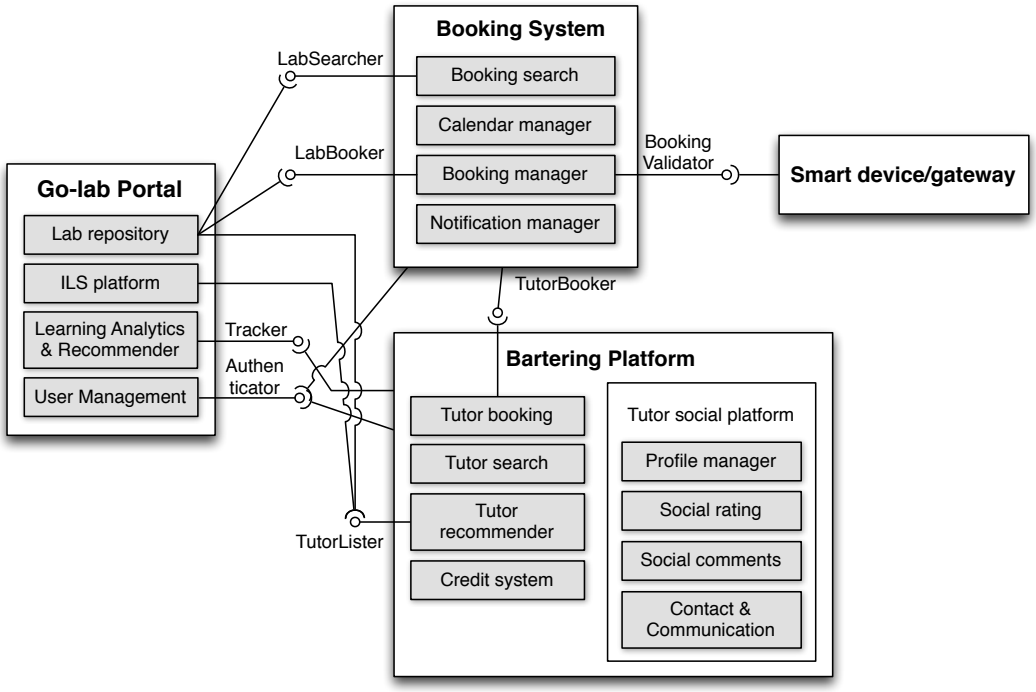

Figure 20. Architecture of the Go-Lab Add-on Services.

Table 6. Go-lab booking schemes.

Booking schemes		Key components	Use cases	Demonstrators	Sections
Go-Lab booking	only calendar integrated	calendar manager	- add lab usage time to go-lab calendar - book the lab usage time	(front-end) Calendar manager UI	3.2.2.3
	transformed	booking manager	- book a Smart Gateway lab - use a Smart Gateway lab	(back-end) Smart Gateway booking service	3.2.2.1
	fully integrated	booking manager & notification manager	- book a Smart Device lab usage time - use a Smart Device lab - booking duration & time left using a Smart Device lab and close session	LabView Smart Device booking	3.2.2.2

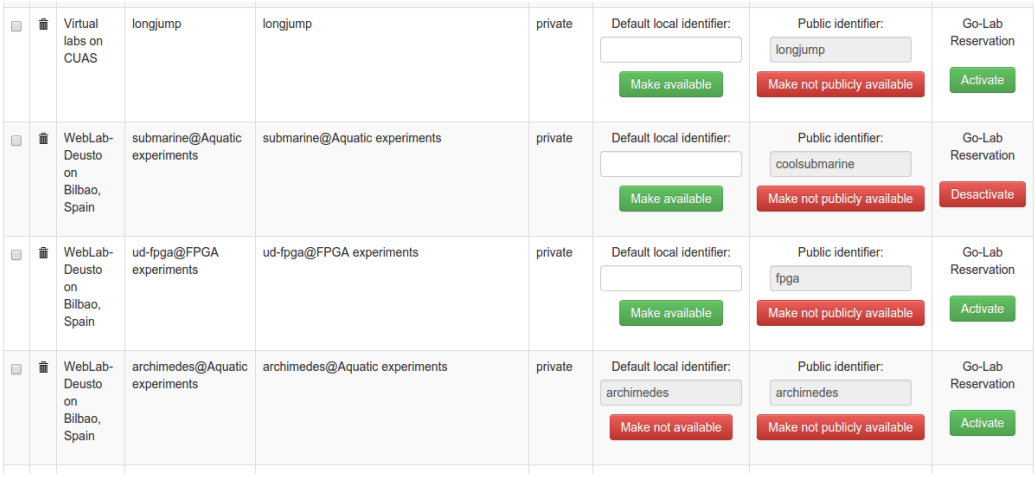

Figure 21. Smart Gateway administrative interface: on the right column it is possible to activate / deactivate the Go-Lab reservation mechanism.

tion. The booking system relies on the Go-Lab Smart Devices and Smart Gateway. More technical details of the Smart Devices and Smart Gateway can be found in the other WP 4 deliverables D4.1 and D4.3.

3.2.2 Implementation

3.2.2.1 Book and use a Smart Gateway lab

The Smart Gateway provides a unified interface for other Go-Lab components to enable external laboratories to be integrated in Go-Lab. To that end, it supports a set of plug-ins for different laboratory management systems. Each plug-in may support one or more laboratories.

The booking mechanism used in the Smart Gateway is based on token provision and reservation. The procedure works in this way. In order to support the booking mechanism, the Smart Gateway lets the Smart Gateway administrator (who registers the plug-ins and laboratories in the Smart Gateway) define whether a laboratory requires the Go-Lab reservation system or not, see Figure 21. If it is defined that one particular laboratory uses it, the only way to access this laboratory through the Smart Gateway is by providing a valid token which has previously been generated by the Go-Lab Booking System. This way, users attempting to use the laboratory without the token will not be able to access, and the Smart Gateway will not contact the final laboratory. If they provide a token, it is checked in the Go-Lab Booking System. And if the Go-Lab Booking System authenticates its validity successfully, the user uses the laboratory. Therefore, it implements the transformed booking scheme in Table 6.

The following screenshots show this mechanism. In Figure 21, the activation / deactivation process is displayed, while in Figure 22, an error message is displayed in case of the token provided is invalid.

As we have pointed out in the Go-Lab booking schemes in Table 6, the trans-

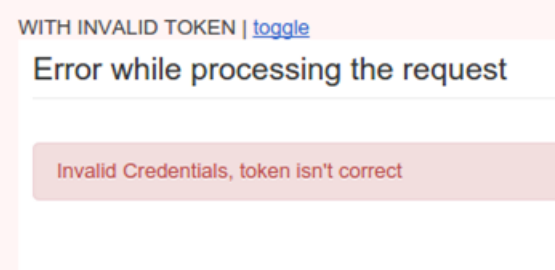

Figure 22. Error accessing a laboratory through the Smart Gateway with an invalid token.

formed booking scheme varies according to the legacy lab.

As the demonstrator is implemented, we set prerequisites for simplicity. One of the prerequisites is that the lab owners can explicitly define that certain periods of time of the week (e.g. Mondays and Wednesdays) are available for Go-Lab users to book. Accordingly the laboratory will only be accessed by Go-Lab in those predefined time periods. During those time periods, the Go-Lab Booking System will guarantee that only those users who had booked the system have a valid token. Since the lab owner will grant for these time slots exclusive access to Go-Lab. In this way, a lab owner could have its own users in the rest of the time using potentially another system than Go-Lab.

This is a coarse-grained mechanism for managing groups of users (e.g. school classes). The Lab owner is still responsible for providing a fine-grained mechanism if the laboratory requires it. For example, it may support many users at the same time, or it may use a queue of users, but Go-Lab guarantees that the number of concurrent students accessing that queue is limited.

The token retrieval mechanism is only a prototype at its early stage. Right now, it can be provided in the URL of the app, by passing a "?token=" query argument, but in the future this mechanism should be replaced with other propagation mechanism in the ILS Platform.

A test prototype is deployed at `https://reacttest.epfl.ch:8093/verify_token?token=skfjs343kjKJ` (where a different token returns "false", while the valid token returns "true") to to demonstrate the scenario. The Smart Gateway deployed at `http://gateway.golabz.eu`, in the demonstrator pointed out above, internally uses this service to validate the provided token. Therefore, using `http://gateway.golabz.eu/os/pub/travoltage-booking/w_default.xml?token=invalid` in the ILS Platform will result in not being able to use the laboratory, while using `http://gateway.golabz.eu/os/pub/travoltage-booking/w_default.xml?token=skfjs343kjKJ` in the ILS Platform will result in loading the laboratory. This approach is described in Figure 9 in D4.2, and it relies also on the generic token validation mechanism described in Figure 8 in D4.2.

As explained in D4.3, the Smart Gateway relies on a software component called gateway4labs. The code repository of the component of gateway4labs where the support for the booking system has been added is `https://github.com/`

gateway4labs/labmanager/.

3.2.2.2 Book and use a Smart Device lab

The procedure to book a Smart Device was specified in D4.2 and D4.1. In this section, we provide a first implementation as a proof-of-concept. To implement this first prototype, we rely on a mock booking service, which is basically an HTTPS GET Web Service that validates a booking token against a set of hard-coded booking tokens authorised by the booking system. This mock service is also used by the Smart Gateway booking prototype.

As decided in D4.2, the Go-Lab booking system will manage the complete booking calendar of a Smart Device. The Smart Device only needs to validate the booking token with the Go-Lab booking system. The underlying scenario is Figure 6 in D4.2 – Booking of a lab using a Smart Device. It illustrates how a client app with a reservation can use a Smart Device (see D4.1 and D4.2 for details). We have implemented the communication between the Smart Device and the Booking System in an example Smart Device developed in LabView[1], available at https://github.com/go-lab/smart-device/blob/master/Desktop/Simple%20examples/SmartDevice_SimpleExample.zip. The Smart Device gets a booking token from the client and then validates this token with the Booking System. This is the only functionality that a Smart Device needs to implement to enable booking. Essentially, our prototype connects securely to the Booking System to validate the authentication token. When the token is valid, the user can act on the Smart Device actuator, if the token is invalid, an error is returned to the client and the actuator value is not updated. Note that in the above example the sensor access ignores the token and allows access.

3.2.2.3 User interface of the calendar manager

This is the front-end of the booking system and accessed by end users in the Go-Lab portal. The first demonstrator implements the scenario depicted in Figure 10 in D4.2 – Administering the booking calendar of a remote lab. It is used by all three Go-Lab booking schemes. It is exclusively applied in the booking scheme only-calendar integrated, as addressed in Table 6.

Lab owners create and manage a calendar for their labs. School teachers can explore the calendar and select their wished lab usage time slots and conduct the booking procedure. Teachers get a notification after a successful booking. Both teachers and lab owners can cancel the booking before the starting time point of the lab usage time slot.

In detail, we create a calendar for each lab which requires booking in the Lab Repository. Lab owners can create and manage time slots / sessions into the calendar. Since a lab can have multiple devices, lab owners can also fill in how many bookings are allowed for each time slot / session, which indicates how many school teachers can book at the same time slot / session. Figure 23 shows the input form for lab owners to create the sessions for booking (see Figure 23 left). After the session is created, all information is listed with the

[1] LabVIEW, http://www.ni.com/labview/

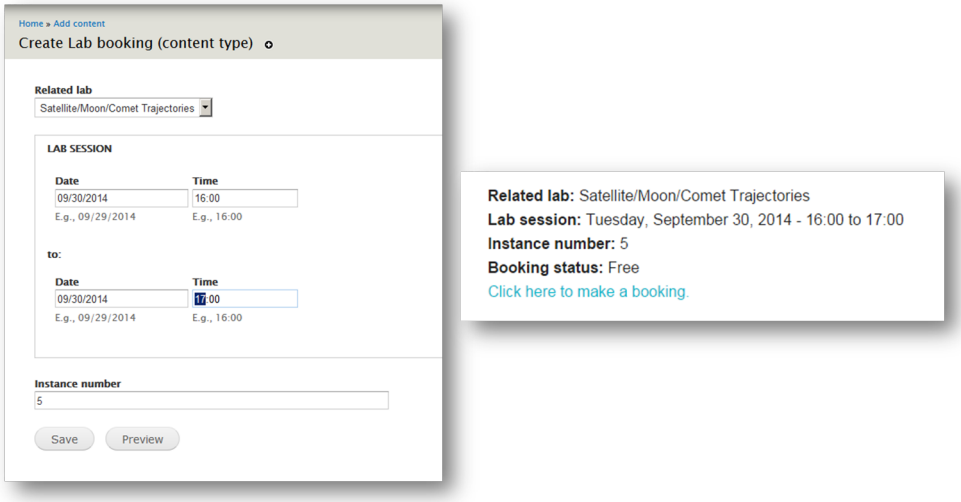

Figure 23. User interface of lab usage time slot creation.

session to be ready for teachers' booking (see Figure 23 right).

Teachers are allowed to book free time slot / session. After the booking, both the teacher and the lab owner get booking notifications. On the calendar page of this lab, the lab owner can see who has booked which lab (cf. the screenshots in Figure 24).

This is just the first demonstrator of the booking user interface, based on the calendar manager. How it interacts with the three Go-Lab booking schemes is discussed in the next section.

3.2.3 Integration plan and open issues

As described before, all three demonstrators work on their own. As a whole, they are still in the phase of test services. That means, there is no data communication among the demonstrators. In the future, we are going to integrate these three demonstrators to consolidate the booking system as a whole. Above all, more labs will be integrated with the booking system via the Smart Gateway and Smart Device specification of which prototypes were developed in this deliverable to illustrate their feasibility.

Furthermore, for the only-calendar integrated booking scheme (see Table 6), Go-Lab is going to forward the booking information to lab owners. All bookings are booked by a standard "Go-Lab user" for simplicity. We still need to realise the data communication among Go-Lab and the lab owners.

For both the transformed and the fully-integrated booking schemes, there is no "Go-Lab user" needed. The Smart Gateway or Smart Devices get the information about which user has booked which lab at what time via the integrated booking system user interface in the Go-Lab Portal. In the next step, a token is generated with the user, time, and lab information to complete the booking process. A mechanism to retrieve the token from the booking system to the client lab apps also still needs to be devised. While the lab is used, the token

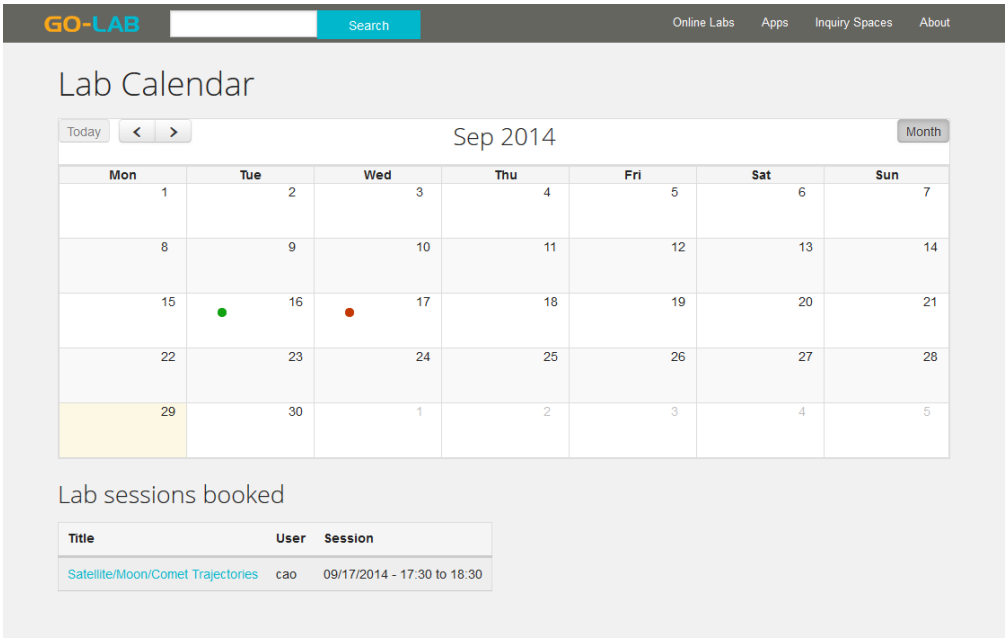

Figure 24. The calendar user interface with booking information at the bottom.

provided is validated via the Smart Gateway or the Smart Device.

The lab owners can manage the booking calendar of their labs in the integrated booking system in the Lab Repository. They could specify the time slots and how many physical remote lab instances can be used concurrently. For legacy labs (labs connected to Go-Lab via the Smart Gateway) the mechanism to verify the booking will not differ from the one used by Smart Devices. However, legacy labs are likely to be managed by an existing legacy booking system and the lab owner might want to limit the time a lab is available to Go-Lab users.

In this case this information about lab availability during certain times shall be exchanged between the Smart Gateway and the Go-Lab booking system at real-time. This availability exchange avoids the case that a teacher books a lab when it is actually not available. A scheme to define these time slots will be created. For instance, a lab owner might decide that on Mondays, only Go-Lab will use the laboratory. This way, the Go-Lab booking system will only display Mondays. If a Go-Lab teacher books it for a session with students on Monday at 10:00, no other teacher will be able to use it at that same time. Therefore, the lab owner knows that the Go-Lab booking system guarantees exclusivity among Go-Lab users on the time slots provided by the lab owner through this scheme which is under development.

In summary, the integration will be processed in this way. After the lab owners make their decisions to use the Go-Lab booking system, the Smart Gateway administrative interface will configure the lab access rights. Meanwhile, the lab owners are able to manage their lab usage time in the calendar in the Go-Lab Repository by themselves. This calendar management task can be totally taken

over by Go-Lab administrators, since the lab owners see one “Go-Lab user” for simplicity. After the booking process, teachers’ booking record is noted in the repository. This information can be transformed into a valid token passed to the Smart Gateway. The validity of the token will be checked when the lab usage is called in the Go-Lab ILS Platform (Graasp).

3.3 Go-Lab Tutoring Platform

The initial release of the Go-Lab Tutoring Platform is `http://tutoring.golabz.eu` for testing and users’ feedback.

The Go-Lab Tutoring Platform is a social tutoring platform where teachers can request help sessions with peers or experts through different communication channels. Sharing practices and user interactions are considered important factors for learning by many educators (Dewey, 1997; Wenger, 1998; Garrison & Arbaugh, 2007). Delta initiative’s[2] research shows that social platforms (Web 2.0) for teachers to share their practices have played a rather more important role. Social support has become a complementary means to the conventional learning management systems in recent years.

The use scenario of the prototype is as follows (details are available in D4.2). Teachers and tutors will have user profiles to describe their expertise and their skills. Tutors’ reputation can evolve based on social ranking and commenting by teachers. Additionally, the social aspect is also reflected in a bartering process for help sessions through a credit system to reward tutors. This bartering process allows tutors to barter their knowledge and time against credits, e.g. social credits such as positive comments and badges. Teachers looking for help can become tutors after they get expertise from peer assistance. Thus, the ranking of the tutors may influence teachers’ opinion on selecting a tutor’s help. For example, tutors can lose their tutor position if they get a poor rating from users.

Thus, an average teacher’s skill growing process makes this use scenario reach to the real-world practices of teachers’ vocational training. Figure 25 shows that many teachers invest time in attending courses & workshops (71% of teachers), and conferences & seminars (44% of teachers) for professional development (OECD, 2014).

Surely, virtual meetings (virtual courses, workshops, conferences and seminars) cannot replace all of the face-to-face offline activities. But they can replace some meetings to a great extent with two benefits. First, teachers are able to save time and expenses in organising trips and in attending these events. They are able to save this resources for other means of professional development. Second, online platforms also make workshops and other events possibly available anywhere at anytime (7/24).

The Go-Lab Tutoring Platform shows the potential to take over some teachers’ professional training events online. For example, a tutor offers a help session to some teachers together on how to use an online lab or how to create a better

[2] `www.deltainitiative.com`

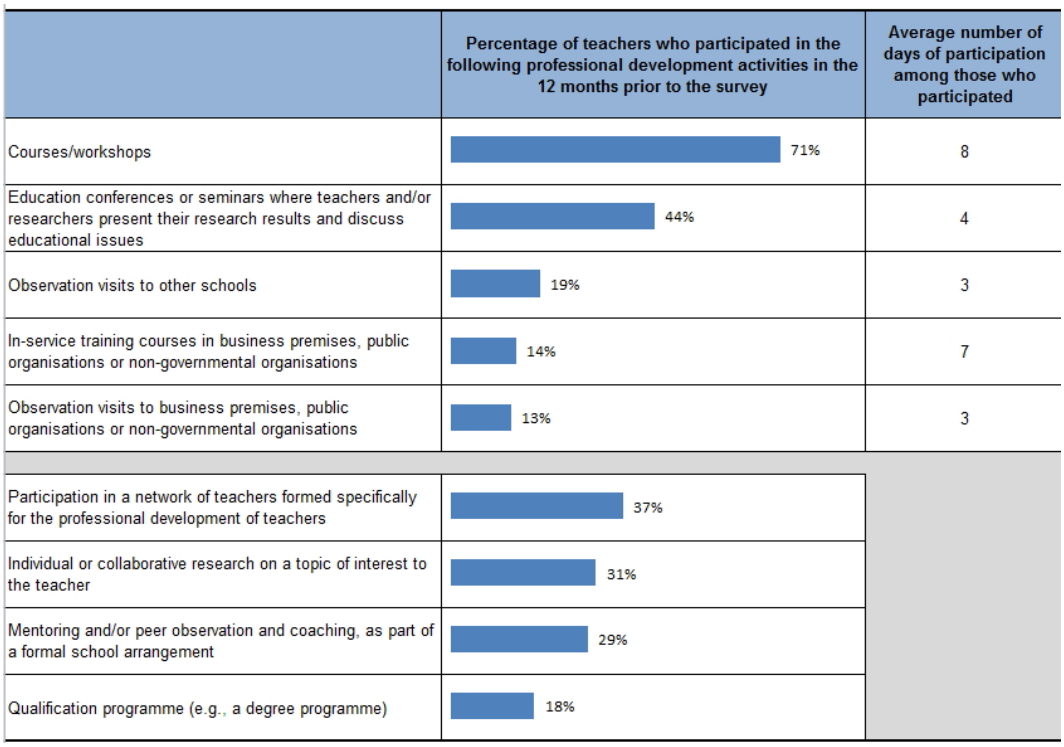

Figure 25. International teachers' professional development types, 2013 results (Figure taken from OECD, 2014).

inquiry learning space.

3.3.1 Architecture of the Go-Lab Tutoring Platform

3.3.1.1 Architecture refinement

The refined architecture diagram of Go-Lab Tutoring Platform is depicted in Figure 26. The differences to the architecture diagram in D4.2 are:

- It is extracted from the add-on service architecture in D4.2 Figure 5.
- The term tutoring has replaced the term bartering for better user understanding.

The tutoring platform focuses on its social aspects with the social platform component to group all social functionality. Tutor booking uses the same approaches as implemented in the Go-Lab Booking System.

3.3.1.2 Requirements review and implementation overview

In this section, we review the requirements specified in D4.2 and list their features, implementation technologies, and current evaluation (feedback to the evaluation) for an overview. Details on the functionality will be presented in Section 3.3.2.1 with main technology details in Section 3.3.2.

Table 7 lists the overview of requirements, features, implementation, and evaluation. The implementation field also list the user interface screenshots which are described in Section 3.3.2.1 with the same letters (tu-Y). Some simple user in-

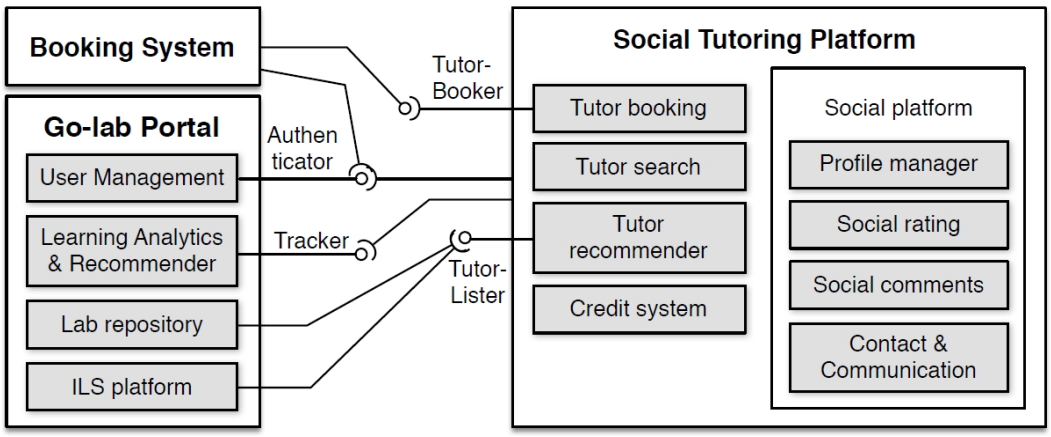

Figure 26. Architecture of the Go-Lab Tutoring Platform.

terfaces are left out for the brevity. The requirement of exchange credits among Go-Lab users and tutors will be implemented in the next phase, which is not listed in the table. The evaluation is preliminary based mainly on developer's opinions, which is further discussed in the next sections.

3.3.1.3 Data model

Based on the D4.2 specification, we have designed a data model that is depicted in an entity relationship diagram in Figure 27. The main entities include user, user profile, offer, and comment. Users create offers and comments. Users also manage user profiles which display comments.

Besides these aforementioned main entities in the ER diagram, offers consist of the entity session; session creates a calendar. Offers refer to a help offered by tutors and taken by those users who need help. Offers have titles and could be related to certain Go-Lab online labs which are external to the Go-Lab Tutoring Platform, since this data resides on the Go-Lab Repository. The entity session refers to the help session with certain time slots, e.g. 13:00-15:30. It has the attribute of participants number. This specifies how many users can use this session at the same time. The entity Google account is also external which is marked in grey in the ER diagram, as this data is managed by Google.

3.3.2 Implementation of the Go-Lab Tutoring Platform

The Go-Lab Tutoring Platform is implemented using Drupal 7. The data storage in Drupal 7 reflects the data model in Figure 27. Furthermore, the tutoring platform relies on Google Services for calendar management (i.e. Google Calendar[3]) and video conferencing (i.e. Google Hangout[4]).

[3]Google Calendar, `http://calendar.google.com`
[4]`http://www.google.com/hangouts/`

Table 7. An overview of Go-Lab Tutoring Platform: Requirements and features.

Requirements	Features	Implementation	Evaluation
Single sign-on	user login (tu-1)	Drupal user account connecting to Google account	The user account with Go-Lab Portal has not been implemented
	user registration (tu-1)		
Managing a user profile	creation of profile	A specific content type user profile	Additional input of Google mail address
	update of profile		
	display of profile (tu-2)		
Commenting and rating	review writing	use of the five-star module	Any user can comment and rate.
	five-star rating		
	average rates listing		
Contacting, bartering, and communicating for tutor session	contact mailing	Google Hangout	email notification is good.
	booking notification messages		
	Google Hangout integration (tu-3)		
booking tutor session	session creation (tu-4)	calendar manager services and UI	Calendar validation rules could be added
	session cancellation		
	booking using a calendar (tu-5)		
recommending tutors	see “list tutors”	use the rating information	Tutor list according to ratings
	a tutor list on the homepage (tu-6)		
searching tutors	search within the tutoring platform(tu-6)	Drupal search API	Tutor names and help offers are searched.
Listing tutors	a tutor list in golabz (tu-7)	REST service & view	lab experiences are not offer-related, but user profile related.

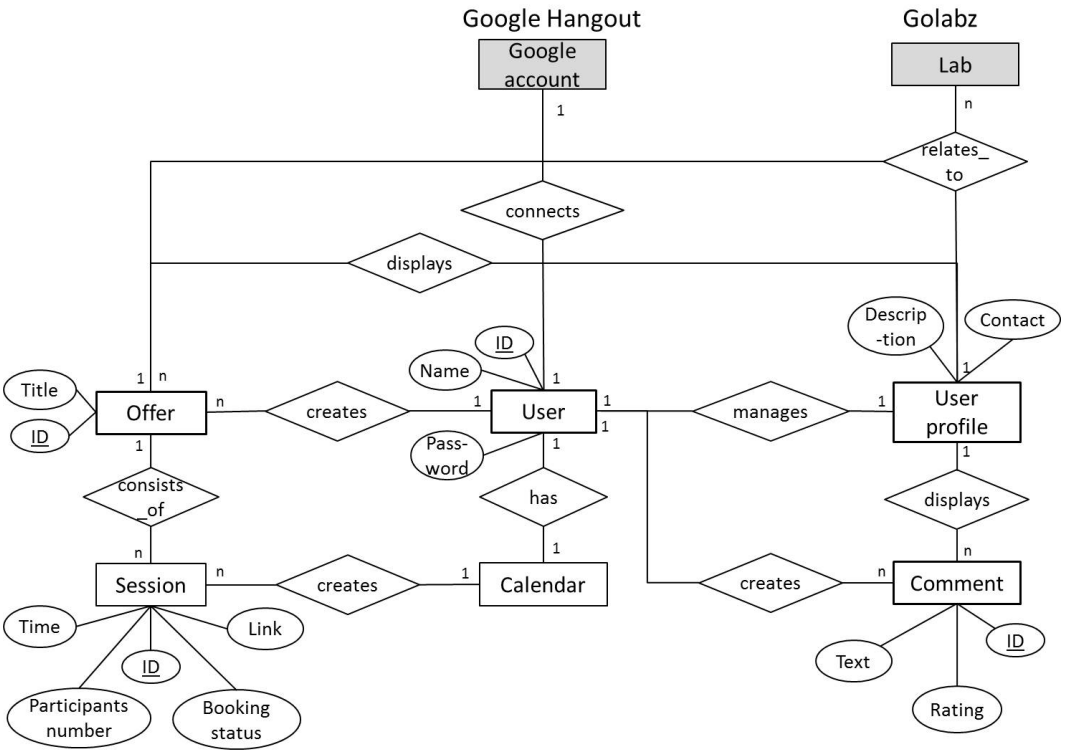

Figure 27. Entity relation diagram of the Go-Lab Tutoring Platform.

3.3.2.1 User interface

In this section, implementation of the previously introduced features are explained together with screenshots. The labels of "(tu-??)" in the screenshots figure caption matches the labels in Table 7. Although some screenshots may cover more than one feature listed above, we try to label them with the key feature for the sake of simplicity. We also try to order the sequence of the screenshots below along with the work flow and data communication in the Go-Lab Tutoring Platform.

The Go-Lab Tutoring Platform is accessible at `http://tutoring.golabz.eu` with the home page after login (see Figure 28). It consists of user login information on right upper corner. The main menu shows the notification if there are some new related actions having happened since users' last login.

Those new related actions can be:

- As a tutor: my help session has been booked by other users.
- As a tutor: my help session which was booked has been cancelled by the user who booked previously.
- As a user: my booked help session has been cancelled by the tutor.

The search box on the header image can be used to search for tutors. It is implemented with the view module of Drupal 7. Tutors' profile and tutors' offer are filtered as the search results. After the search, each tutor with their name,

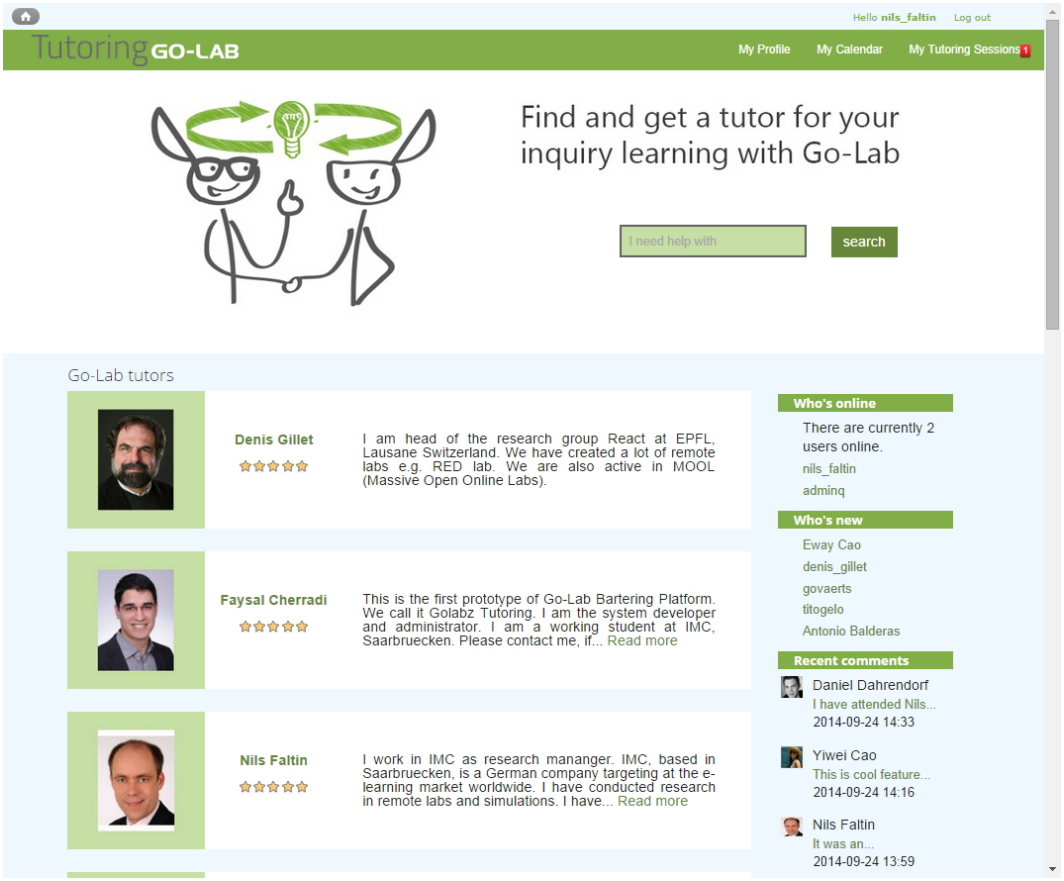

Figure 28. Home page of the Go-Lab Tutoring Platform (tu-6).

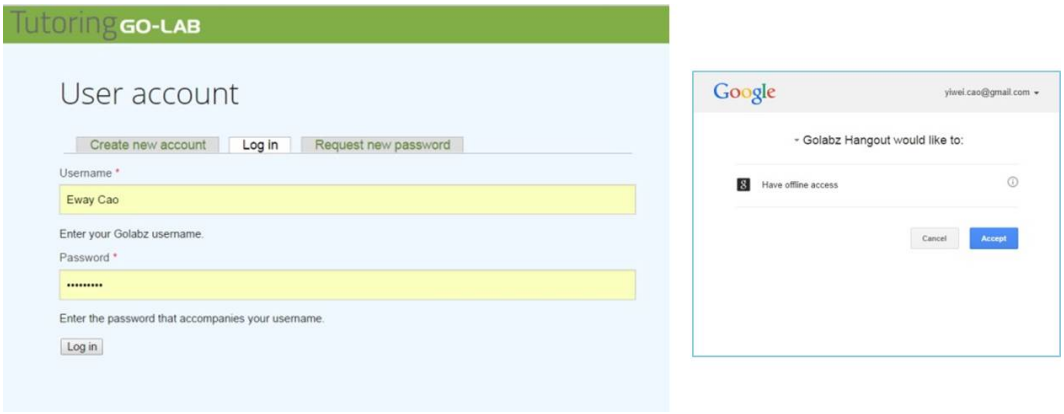

Figure 29. User login of the Go-Lab Tutoring Platform (tu-1).

ranking, and profile is listed, in the same way as depicted in Figure 28 which is before the filtering.

When users come to the Go-Lab Tutoring Platform, they are required to register and login. Without login, they can only view tutor information but they are not allowed to create or book any help session.

The login feature is presented in the screenshot in Figure 29 (left). After the user's successful login, the user gets a confirmation site with one's Google user account to access Google Hangout (see Figure 29, right).

If the user registers into the tutoring platform for the first time, the user needs to create a user profile (mainly a short self-description and the email address) first. Either a tutor or any normal user who wants to get help needs to create her own profile in order to record any interactions with the platform. Tutors are able to create an offer with a series of help sessions. Thus, the user profile page will also display one's offers. User profiles are also viewable by all users. However, the content and actions vary accordingly.

Figure 30 shows the view of a user profile page by clicking "my profile" in the menu. It consists of the header part, the tutoring offers part, and the review part. If the users are not tutors and don't have any tutoring offers, the tutoring offers part is not displayed. Table 8 lists the difference between user's viewing one's own profile page and others' profile pages.

After a tutor login into the platform, the tutor would be interested in creating a tutor offer and creating a tutor session with certain time slots. Figure 31 shows the interface of creating a session which is related to a certain offer. More useful, the tutor is able to enter how many participants can attend this session at the same time. The number range is from 1 to 13 which is the maximum number supported by Google Hangout.

The newly created session is also displayed in the calendar. The color green and red shows the booking status of the sessions. If a session supports multiple

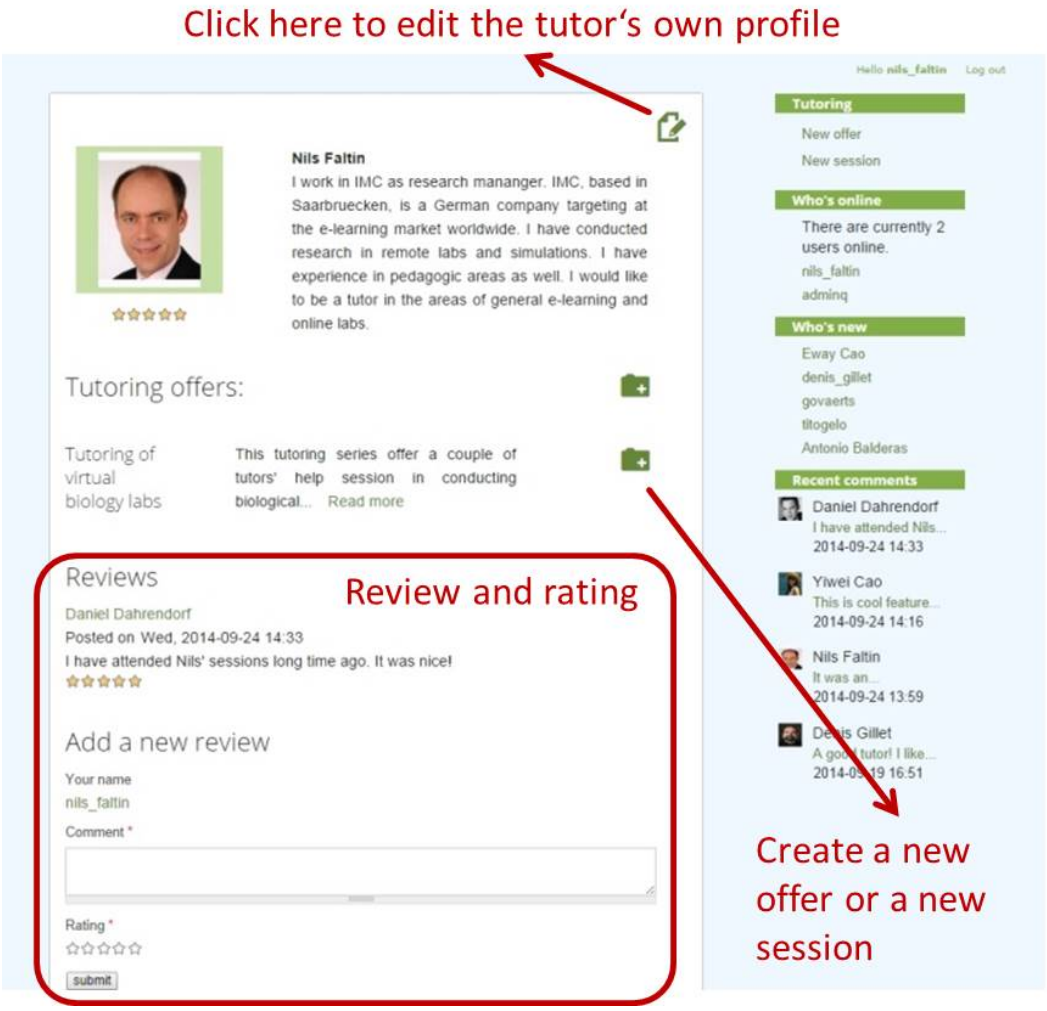

Figure 30. Tutor profile management and display in the tutoring platform (tu-2).

Table 8. Views of own user profile and of others.

Elements		**Own view**	**Others' view**
Header	view name, description, photo &ranking	yes	yes
	view and use the contact button	no	yes
Tutoring offers	view offers	yes	yes
	view the "folder +" button to create a new offer	yes	no
	view the "folder +" button to create a new session	yes	no
	view the "calendar" button to book a session	no	yes
Review & ranking	view reviews & rankings	yes	yes
	add a new review	yes	yes
	rank	yes	yes

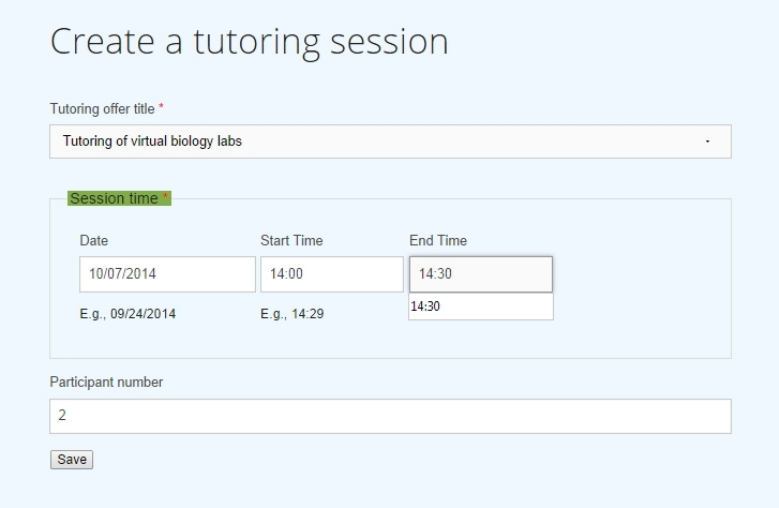

Figure 31. A tutor creates a session filling into the calendar in the tutoring platform (tu-4).

booking, it is displayed in the calendar in green so far the number of the booked participants has not exceeded the maximal participant number given by tutors beforehand (see Figure 32).

Registered users are allowed to click the sessions in green in this calendar for booking a wished help session. They are also allowed to cancel the sessions they have booked until the session starts. If the user has to cancel a help session, the session tutor will get a notification about the cancellation activity. Further rules can be set to validate the cancellation process.

The booked session is listed both on the tutor's and the user's page My Tutoring Sessions as depicted in Figure 34. It displays the information about tutor name, session title and time, users who booked the session, and a session link in Google Hangout. Both the tutor and the users get the same Google Hangout link, so that they are able to conduct a video help session together.

3.3.2.2 REST Service

The interface of TutorLister is implemented with RESTful services. The REST API is used as the web service for data exchange between the Go-Lab Portal and the Go-Lab Tutoring Platform as an add-on service. Figure 35 depicts how REST services are used to list tutors for each lab in the Go-Lab Portal.

Recalling the relationship between offer in the Tutoring Platform and lab in the Lab Repository (Figure 27), the Tutoring Platform sends a request to get the list of all labs in the portal as the first step (1 in Figure 35, same for below). Therefore, tutors can assign their help offers with certain labs (zero to unlimited labs). An offer could have nothing to do with any Go-Lab online labs, could be related to a specific labs, or could even pertain to more than one lab. The REST service of the portal returns a list of labs in XML. Other formats including JSON, JSONP, PHP etc. are also supported by the REST service. The Tutoring Platform imports labs from the results and create nodes for labs as the second

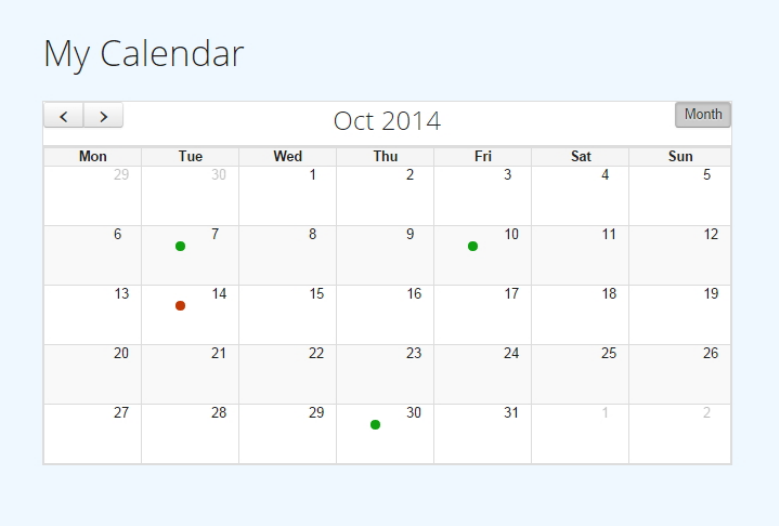

Figure 32. Tutor's calendar for other users' booking (tu-5).

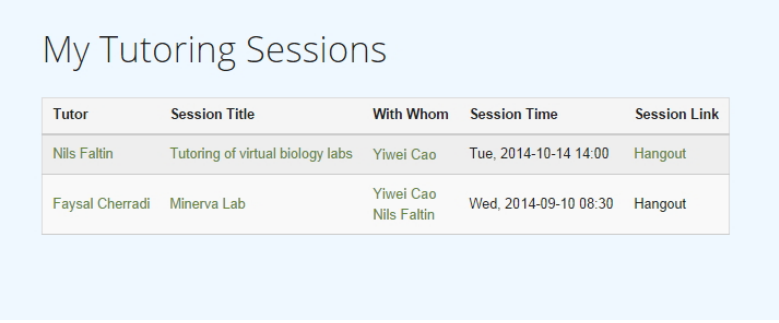

Figure 33. Google Hangout links are created for each help session with tutor and participants information (tu-3).

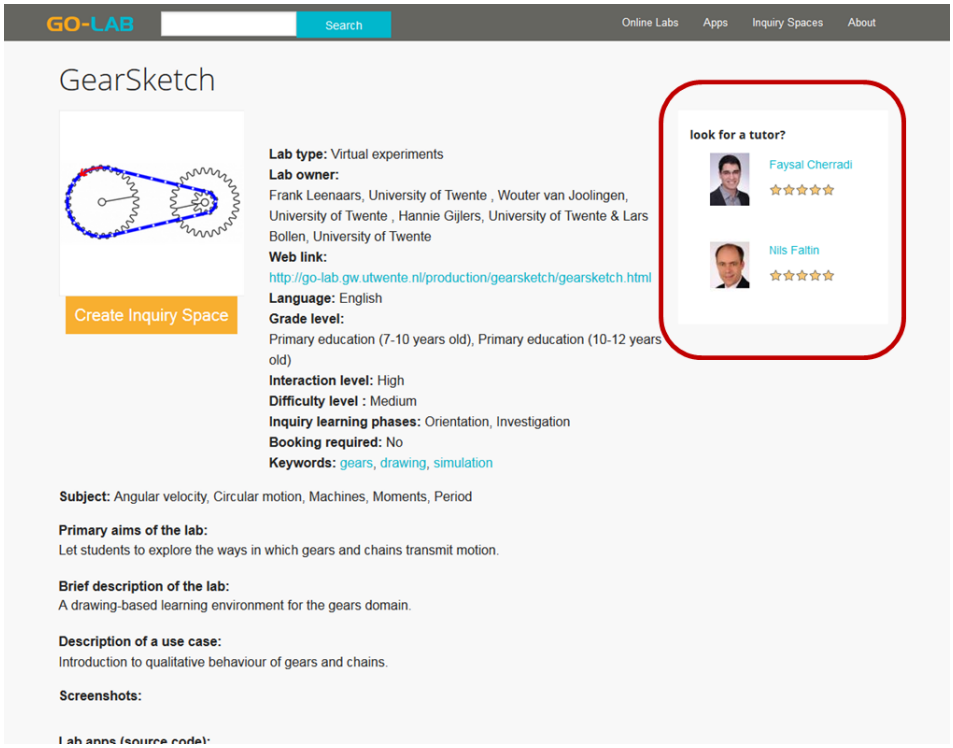

Figure 34. List tutors in the Go-Lab Portal (tu-7).

step (2). Since the lab information is available, tutors are able to connect their offers with certain labs. Vice versa, the portal requests a tutor list to recommend tutors to teachers who visit the web page of certain labs. To prepare handling this request, lab ID is requested and retrieved via RESTful service. Drupal supports universal unique identifiers which are used to identify the lab ID in both platforms (3 and 4). With the retrieved lab ID, the Go-Lab Portal sends a tutor request (5). A view of tutors related to this lab ID is created and sent back to the portal (6).

This connection between the Go-Lab Portal and the Tutoring Platform is configurable, in order to meet the non-functional requirements – portability. In the configuration user interface of the portal, the url of the Go-Lab Tutoring Platform is required. Only the system administrator has this access right for system security.

The module `rest_api_client` implements the configuration user interface and the data communication between the portal and the tutoring platform.

3.3.2.3 Google Hangout

To not reinvent the wheel and its easy integration for the communication between tutors and users, the Go-Lab Tutoring Platform employs Google Hangout for the implementation of the component “Contact & communication”. On the one hand, it needs many resources to develop a stable communication tool. On the other hand, there are many existing communication tools and technologies which can be integrated, such as Google Hangout.

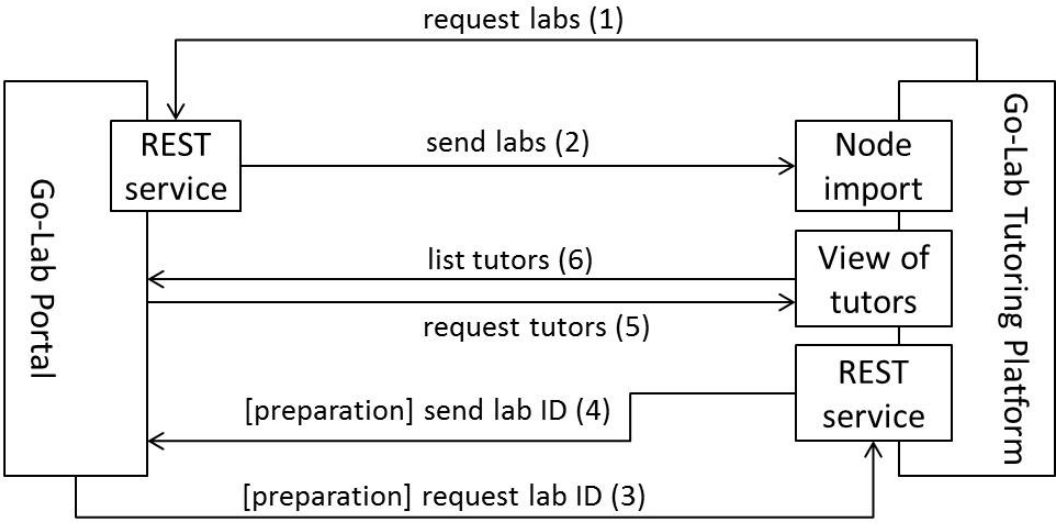

Figure 35. Go-Lab REST services.

Google Hangout is an instant video messaging tool to support real-time communication. It also offers a chat tool, screensharing, and other instant messaging effects. Using the Google API, it is easy and reliable to employ Google Hangout. The openness of Google technologies support better system integration than other communication tools including Skype.

In order to use Google Hangout in the Go-Lab Tutoring Platform, user login has to be connected to the Google account of a user. The tutoring platform checks whether the user's Google session is active or not automatically. Users themselves need to allow this connection. If it is not wished, Google Hangout-based video chatting between tutors and users are not supported. If users allow the Go-Lab Tutoring Platform to access their Google user accounts, the Go-Lab Tutoring Platform uses the Google Calendar Service to create a hangout between users and tutors. Hangouts are dealt with as a video call event in Google Calendar. Accordingly, users need to configure their calendar with a check for confirmation (see Figure 36). While using the tutoring platform, the expiration of Google session is also checked, besides user sessions in the tutoring platform. Users are required to re-login after the hour-long user sessions are expired.

The connection to Google API is also configurable by the system administrator in the portal users interface. The module bartering_login_google has realised the aforementioned functionality.

3.3.3 Evaluation of the Go-Lab Tutoring Platform

Following the proposed "Participatory design survey for the bartering platform" in D4.2 (Page 58), the first survey results are collected during a three-hour workshop at JTEL Summer School 2014 April in Malta.

The PD workshop aims to evaluate the first prototype and assess whether the user requirement assumptions are correct. The first survey focused on users' skills and knowledge about physical and online labs, their need for experts'

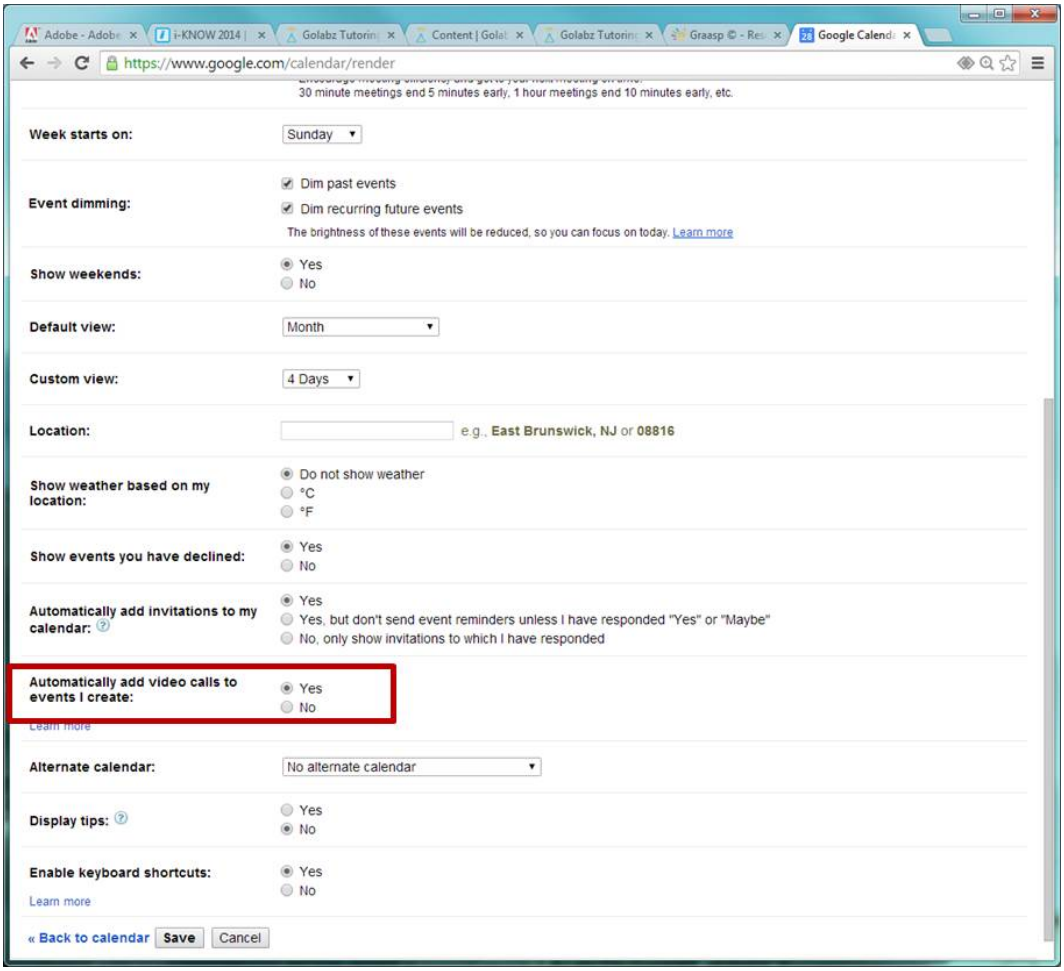

Figure 36. Prerequisite set-up of using Google Hangout.

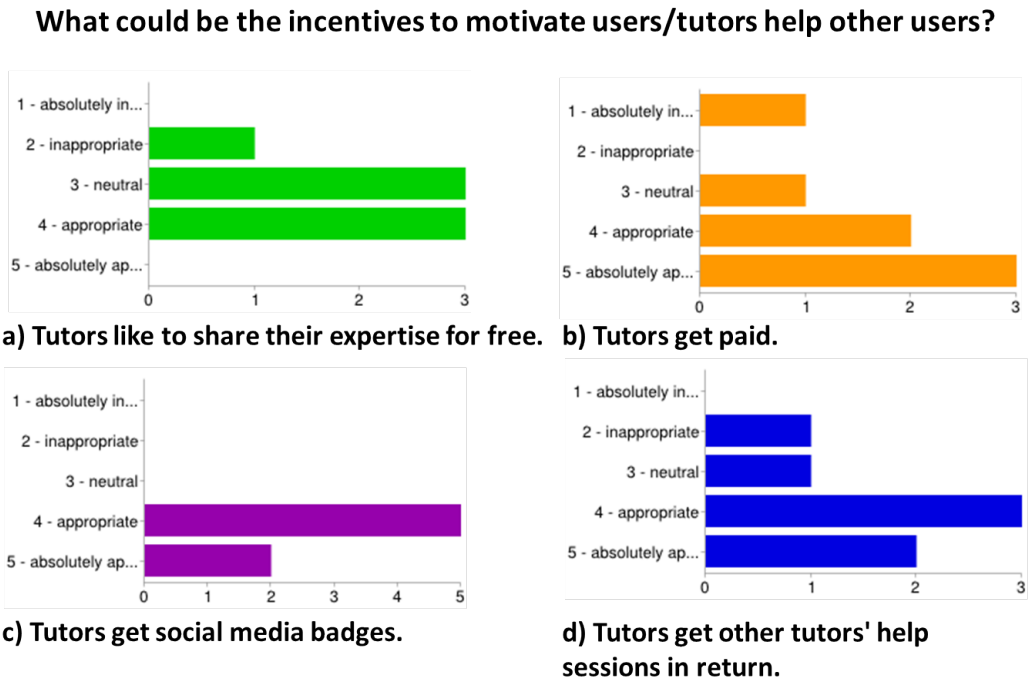

Figure 37. User survey on incentives in the Go-Lab Tutoring Platform.

help, the appropriate incentives for tutors, and their experience with the Go-Lab Tutoring Platform prototype.

Seven survey participants were introduced to the platform and played the role of either a tutor or a teacher. Each created a profile. After that, test tutors created help offers. Test teachers explored the tutors' profiles and searched for help sessions. Afterwards, test teachers booked a help session. If a help session was booked successfully, a Google Hangout link was created for this help session. Then, tutors were helping teachers via Google Hangout. After the hands-on experience, seven participants completed the online survey.

The survey results show: (i) users need help and support because they meet with problems and look for expertise during online lab use; (ii) incentives such as being paid or getting social badges motivate tutors mostly. Figure 37 presents the results related to which incentives users prefer in exchange for tutoring (using Likert scales from '1 – absolutely inappropriate' to '5 – absolutely appropriate'). The participants are indecisive whether they want to provide help for free (Figure 37a) with an average rating of 3.2. On the other hand, they prefer to tutor in return for social media badges (37c, average rating 4.3) or want get paid (37b, average rating 3.9). Bartering for getting help form other tutors (37d, average rating 3.9) is also perceived as positive. Overall, the proposed incentives are deemed appropriate by users.

Furthermore, participants were asked which type of help they would prefer. The two most valued types of help was face-to-face and online meetings. Still positively perceived were a helpdesk and online discussion fora, while the partici-

pants were indecisive on social media and online search.

3.3.4 Open issues and future plan

Based on the PD workshop survey as well as the developer's viewpoints, we came to the first improvement of the tutoring platform. According to testers' feedback, we have realised support of multiple users. Thus, a tutor is able to give a help session to multiple users.

The open issues include:

- If we employ Google Hangout as a long-term option for video chatting, we may re-consider the user registration and login mechanisms. Using Google login will simplify the current login process.
- The design can still be improved especially in the aspect of branding and recognition between the Go-Lab portal and the tutoring platform.
- How to keep the user community active would depend on the business model behind the tutoring platform. The credit system still needs to be implemented. On the one hand, we deploy a free platform for school teachers and students. On the other hand, we want to establish a credit system based on the business model for the sustainability.

The current credit system is only based on the social rating among tutors and teachers. Tutors may get incentives via a better ranking in the tutoring platform. This is also reflected in the list on the home page. The better ranked tutors are on the top of the home page. More research is still needed to switch the credit system to a payment system in the future. However, we can already benefit from the first prototype of the tutoring platform to survey it with the teachers at teachers' participatory design workshop and to explore the feasibility of a monetary credit system in the future.

Above all, we have a better future road map of the tutoring platform, since the first prototype is shaped. As the next development stage, the platform will reach out to the other Go-Lab work packages. Figure 38 shows the tutoring platform's connections and contributions.

We will roll out the first prototype to the core group teachers (CGT) at the participatory design workshops by Work Package 3. Cooperation will be strengthened with Work Package 2 to develop detailed use scenarios with school teachers. The research results together with WP3 and WP2 will be implemented and applied in teacher community building and teacher training within Work Package 6. This teacher training realisation in the real world is the basic idea of the business model development in Work Package 9.

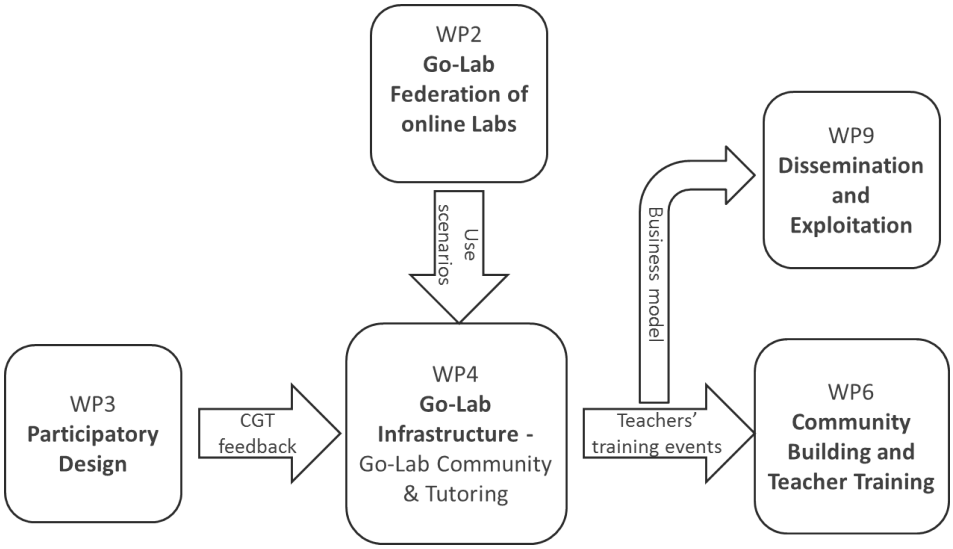

Figure 38. Development plan of the tutoring platform.

4 Conclusion

In this deliverable we documented our software releases for the learning analytics, scaffolding, and add-on services in Go-Lab.

The first prototype of the learning analytics infrastructure is the consequent realisation of the initial specification (D4.2). It follows a definition of four main services and interfaces, connected to the Go-Lab portal, namely for logging of user events, a back channel for notifications or reconfiguration of tools, analytics services through the Analytics Workbench, and the retrieval for artifacts. The connection of the different data sources enables rich possibilities for learning analytics, both in the long and short term. Intelligent software agents can analyse the behaviour of learners. The results can be fed back through the notification channel to generate processable actions for the learner. This deliverable also presented use cases and examples, how the learning analytics infrastructure can enrich the learning experience in Go-Lab.

In the future, the guidance mechanisms will be extended, namely through adding more agents and improved algorithms. The analysis of sequential patterns and process discovery can be used in many ways, for example to support reflection when presenting the learning process to the learner, or to support the collaboration inside the classroom through group formation based on action sequences. The aggregation of different methods can be used to create a learning dashboard, as proposed in D1.3.

The first prototypes of add-on services have demonstrated to provide value-adding, and plugable services to the Go-Lab Portal. Both the booking system and the tutoring platform have an interface to the Go-Lab Portal, which can be switched on or off in a flexible way.

The current development progress of the booking system is still at the phase of mock-up services. However, it works in each mock-up service. They will be integrated together as the next step. The final version of the booking system will contribute to the federation of remote labs. Those labs that don't have a calendar-based booking service could be integrated into the Go-Lab Portal to employ the booking service. In other words, the Go-Lab Booking System could allure more remote labs to join into the Go-Lab Portal.

The tutoring platform offers teachers peer assistance, when teachers have questions or difficulties using the Go-lab portal, creating inquiry spaces, or conducting virtual experiments. At the same time, the tutoring platform also attempts to motivate teacher communities to interact with the Go-Lab Portal actively. It aims to establish a social tutoring platform for active user communities. Currently, the prototype is able to help us establish a feasible business model for the realisation of the credit system, ranging from social rating to payment mechanisms. We will ask more teachers to test the tutoring platform to evaluate their feedback. Teachers' practical feedback will be valuable for development improvement of the tutoring platform.

In summary, all prototypes have demonstrated the feasibility of the specification.

The described systems will used in the upcoming pilot phase B (see D7.2). The data gathered through the learning analytics back-end can support evaluation studies and the tutoring platform will foster an active teacher community. On the other hand the evaluation, test results and further development will contribute to refined specifications and improved prototypes. The refined specifications will be documented in D4.6 (M33) and the final system releases will delivered in D4.8 (M36).

A Appendix A - Initial concept for message internationalisation

Since the Go-Lab portal is intended to be used in various countries the internationalisation of feedback messages is an important issue. There are two possible strategies to achieve that a message that has been created by a server-side analytics component is always displayed in the language of the student. The first strategy would be to do the translation in the message creating component on the server. This however, would require a mechanism that allows client-side apps to send language information to the server in addition to the action logs. Since, an alternative approach has been implemented in the Go-Lab learning analytics infrastructure. As it can be seen in Section 2.3.1 messages can carry a message id and message parameters as its content. In a dedicated codebook which is accessible on the learning analytics server clients can translate messages by themselves. This is reasonable because client-side apps are responsible for displaying messages. They have permanent access to context variables like the user and the language. Consequently the translation of feedback message should also be handled by the apps themselves. Further information can be found at `https://github.com/go-lab/ils/wiki/Message-internationalisation`. A test application for the translation of feedback messages is available at `http://golab.collide.info/internationalisation`.

References

Acquisti, A., Balsa, E., Berendt, B., Clarke, D., Wolf, R. D., Diaz, C., ... Vanderhoven, E. (2011). SPION Deliverable 2.1 State of the Art (COSIC internal report).

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. Springer.

Boyd, D. (2008). Taken out of context: American teen sociality in networked publics (Unpublished doctoral dissertation). University of California-Berkeley.

Collins, M. J. (1996). A new statistical parser based on bigram lexical dependencies. In Proceedings of the 34th annual meeting on association for computational linguistics (pp. 184–191).

Dewey, J. (1997). Experience and education. Simon & Schuster. Retrieved from http://books.google.de/books?id=UWbuAAAAMAAJ

Garrison, D. R., & Arbaugh, J. (2007). Researching the community of inquiry framework: Review, issues, and future directions. The Internet and Higher Education, 10(3), 157-172. Retrieved from http://www.sciencedirect.com/science/article/pii/S1096751607000358 doi: http://dx.doi.org/10.1016/j.iheduc.2007.04.001

Gelernter, D. (1985). Generative communication in linda. ACM Transactions on Programming Languages and Systems (TOPLAS), 7(1), 80–112.

Göhnert, T., Harrer, A., Hecking, T., & Hoppe, H. U. (2013). A workbench to construct and re-use network analysis workflows - concept, implementation, and example case. Niagara Falls, Canada.

Hecking, T., Manske, S., Bollen, L., Govaerts, S., Vozniuk, A., & Hoppe, H. U. (2014a). A flexible and extendable learning analytics infrastructure. In (pp. to appear, accepted 2014/05/12). Springer.

Hecking, T., Manske, S., Bollen, L., Govaerts, S., Vozniuk, A., & Hoppe, H. U. (2014b). A flexible and extendable learning analytics infrastructure. In Advances in web-based learning - ICWL 2014 - 13th international conference, tallinn, estonia, august 14-17, 2014. proceedings (pp. 123–132). Retrieved from http://dx.doi.org/10.1007/978-3-319-09635-3_13 doi: 10.1007/978-3-319-09635-3_13

Lai, Y.-L., & Hui, K.-L. (n.d.). Internet opt-in and opt-out: Investigating the roles of frames, defaults and privacy concerns. In ACM SIGMIS CPR 2006.

Li, N., Holzer, A. C., Govaerts, S., & Gillet, D. (2014). Enforcing Privacy for Teenagers in Online Inquiry Learning Spaces. In Understanding Teen UX workshop at CHI 2014.

Li, N., Najafian-Razavi, M., & Gillet, D. (2011). Trust-aware Privacy Control for Social Media. In Work-in-Progress of CHI 2011.

Manske, S., Hecking, T., Bollen, L., Gohnert, T., Ramos, A., & Hoppe, H. (2014, July). A flexible framework for the authoring of reusable and portable learning analytics gadgets. In Advanced learning technologies (icalt), 2014 ieee 14th international conference on (p. 254-258). doi: 10.1109/ICALT.2014.80

Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011). Dbpedia

spotlight: Shedding light on the web of documents. In Proceedings of the 7th international conference on semantic systems (pp. 1–8). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/2063518.2063519 doi: 10.1145/2063518.2063519

OECD. (2014). TALIS 2013 results: An international perspective on teaching and learning. OECD Publishing. Retrieved from http://www.oecd-ilibrary.org/education/talis-2013-results_9789264196261-en doi: 10.1787/9789264196261-en

Raynes-Goldie, K. (2010). Aliases, creeping, and wall cleaning: Understanding privacy in the age of facebook. First Monday, 15(1).

Tsai, J., Egelman, S., Cranor, L., & Acquisti, A. (2007). The effect of online privacy information on purchasing behavior: An experimental study. In WEIS 2007.

Vozniuk, A., Govaerts, S., Bollen, L., Manske, S., Hecking, T., & Gillet, D. (2014). AngeLA: Putting the Teacher in Control of Student Privacy in the Online Classroom. In ITHET 2014.

Weinbrenner, S. (2012). Sqlspaces: a platform for flexible language heterogeneous multi-agent systems. München: Hut. (Zugl.: Duisburg, Essen, Univ., Diss., 2012)

Wenger, E. (1998). Communities of practice - learning, meaning, and identity. Cambridge University Press. Retrieved from http://www.ewenger.com/theory/index.htm