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 D5.4

Releases of the Go-Lab Portal and the App Composer – Initial

Editor	Adrian Holzer (EPFL)
Date	24th October, 2014
Dissemination Level	Internal
Status	Draft

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
Adrian Holzer, Sten Govaerts, Andrii Vozniuk, Wissam Halimi, Maria Jesus Rodriguez Triana, Aubry Cholleton, Denis Gillet	EPFL
Yiwei Cao, Nils Faltin	IMC
Pablo Orduña, Luis Rodríguez	UDEUSTO
Panagiotis Zervas, Alexandros Trichos	CERTH

Legal Notices

Executive Summary

This deliverable presents the initial release of the Go-Lab portal and the app composer following the specifications presented in D5.2. We first present the features of the Lab Repository, then we discuss the ILS Platform before we introduce the App Composer.

The main requirements in D5.2 focused on the management of ILS, Labs and Apps. For instance finding them on the Lab Repository, potentially modifying them and using them in the ILS Platform and publishing them on the Lab Repository. Further requirements focus on the community, allowing to share apps and ILS, provide social features and ensure usability, privacy and facilitated access.

The features of the Go-Lab Portal and the App Composer delivered in this initial release are the result of the ongoing collaboration between pedagogical requirements expressed by WP1 (see D1.1 for more details) as well as participatory design sessions and evaluation results provided by WP3 (see D3.1 and D3.2 for more details) and work on the lab metadata in collaboration with WP2 (see D2.1 and D2.2 for more details). Together these features provide a first fit to almost all elicited requirements, with the exception of recommendations for labs, apps and ILS in the Lab Repository and the ILS Platform, as well as the App Composer integration with the Go-Lab portal. However, some features only include basic functionality and need to be extended for the final release.

We are currently working on the open features and are maturing some other features. We will continue to build on cross work package inputs to further refine the existing features. The next iteration of the personalisation features will be presented in D5.5 at M32 and the next iteration on the Go-Lab Portal and the App Composer will be released in D5.6 at M36.

Table of Contents

List of Figures

1 Introduction

The specifications for the Go-Lab portal and the App composer were elicited in D5.2. The main product of this deliverable is the developed software. The accompanying text in this deliverable describes the software development related to the initial release of the Go-Lab portal and the App Composer. This deliverable is structured as follows. First we describe the releases of both components of the Go-Lab portal, i.e., the Lab Repository (Section 2) and the ILS platform (Section 3). ILS stands for Inquiry Learning Space as introduced in previous deliverables.

More specifically, Section 2 discusses the Lab-repository (`http://golabz.eu`) by first presenting an overview, then detailing the software architecture and finally relating the Lab Repository features to the requirements elicited in 5.2. Among the noticeable features, The Lab Repository now hosts 48 labs, 28 apps, and 8 ILS. The lab repository also allows to publish ILS and reuse ILSs.

Section 3 presents the ILS Platform (`http://graasp.eu`) starting with an introduction, then discussing the architecture and finally the requirement fit. Among the noticeable features, it allows to manage ILS from the creation to publication via exploitation. The ILS Platform has undergone an important interface and architectural redesign.

Section 4 presents the App Composer (`http://composer.golabz.eu`) by first giving an overview of its two modules, namely the Adaptor and the Translator. Then their architecture and the requirement fits are presented. The Translator allows to translate apps into any language and for each language a target group (based on age) can be selected. The Adaptor allows to fine tune apps.

Finally, Section 5 summarizes the requirement fit and discusses the next steps. For instance, the final release of personalisation features and inquiry learning apps will be delivered in D5.5 at M32 and the final release of the Go-Lab Portal and the App Composer will be delivered in D5.6 at M36.

2 The Lab Repository

The Lab Repository is available at `http://www.golabz.eu/` and its source code is available on a subversion repository.[1]

2.1 Introduction

The Lab Repository is a starting point for teachers that want to create an ILS and it contains resources to build an ILS. The repository enables access to online labs, apps and shared ILS created by other teachers. The first prototype of the Lab Repository was launched before M18, which was linked to Go-Lab project web site in April 2014. This first prototype was already described in an appendix of D5.2. Through continuous development, this deliverable presents a stable Lab Repository at M24.

Figure 1 shows the current landing page of the portal. The Lab Repository has currently integrated all online labs selected by WP2 (see D2.1 and D2.2) and some extra third-party labs to demonstrate a first integration with the Smart Gateway and Smart Devices (see D4.1 and D4.3), e.g., PhET labs [2]. With regard to the apps, we have reviewed all apps (over 150 apps) from the ROLE Widget Store[3] from the previous EU IP project ROLE (Dahrendorf, Dikke, & Faltin, 2012) and migrated the apps relevant for the Go-Lab context. This prototype of the Lab Repository also contains the ILS integration. ILS can be created based on labs or shared ILS and can be shared from the ILS Platform to the Lab Repository.

The remainder of this section gives an overview of the Lab Repository and its integration with the ILS Platform, presents the Lab Repository architecture and discuss its requirement fit with respect to D5.2.

2.1.1 Facts and numbers:

The statistics of the resources in the Go-Lab Portal as of September 30, 2014, is summarised below.[4]

- *Labs:* 48 labs;
- *Apps:* 28 apps, among them over 10 apps are exported from the ROLE Widget Store;
- *ILS:* 8 inquiry spaces including 5 full and 3 mini inquiry spaces;
- *Big ideas:* 8;
- *Lab owners:* ca. 55 lab owners.

2.1.2 Terms

Some terms have evolved since the publication of D5.2 to better fit the target audience, namely teachers and students.

[1] `http://svn.research.im-c.de/golab-labrepository/trunk/go-lab-repository`
[2] `http://phet.colorado.edu/`
[3] `http://www.role-widgetstore.eu/`
[4] Statistics can be found at `http://www.golabz.eu/about-go-lab-portal`

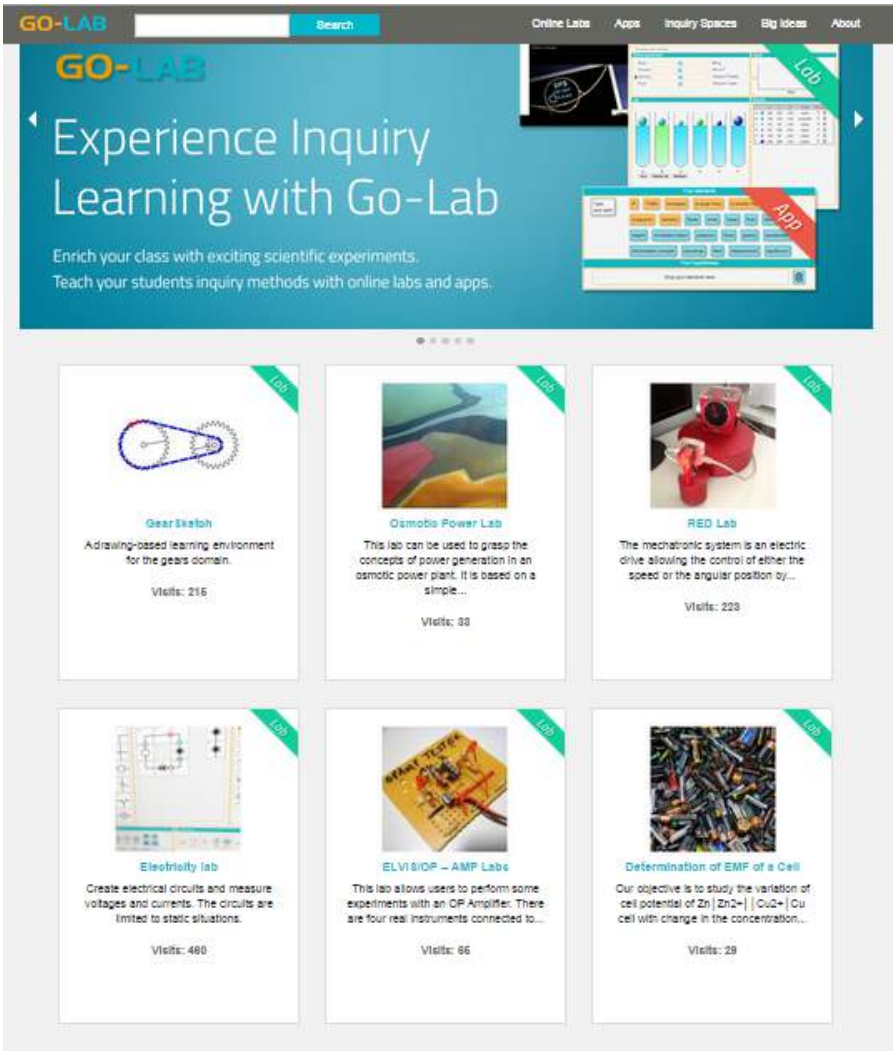

Figure 1. The Lab Repository home page

Golabz is the site name of the Lab Repository. It consists of three resources: *Online labs*, or labs for short, include remote laboratories, virtual experiments, and data sets. *Apps* are Web apps to help conduct inquiry learning activities. *Inquiry learning spaces*, spaces or ILS for short, are structured containers for labs and apps together with other instructional documents. Besides, the concept of *Big ideas* is used to organise online labs. All online labs are associated with certain big scientific ideas in order to be learnt with practical knowledge contexts. These aforementioned terms appear in the Lab Repository menu to structure the resources.

2.2 Architecture

This section details the technical implementation of the architecture. Detailed functionality and implementation technologies are discussed in the next section.

2.2.1 User interface design

The user interface is designed to deliver a valuable platform, in order to reach the final goal of an innovative inquiry learning platform for schools. To that end, we employ Peter Morville's user experience honeycomb (Morville, 2005) to assess the user experience (UX) of the lab repository. Peter Morville models seven UX factors derived from an information architecture model as depicted in Figure 2.

Figure 2. Morville's UX honeycomb model

Among them, the factor *valuable* in the middle could be considered as a top-level goal which all design elements contribute to. We describe our measures to different factors according to Morville's UX model below.

Useful. The user interface gives users ideas and useful information about what Golabz is. An example is users can easily attain the information what Golabz is via reading the header images on the home page. Furthermore, the content on the Lab Repository and its functionality (e.g., previewing apps and creating an ILS based on lab or shared ILS) can be useful for teachers to build an ILS.

Usable. The user interface is easy to use and has a clear structure. The home page is organised as a grid to host all labs and apps, while there is a lab catalogue, an app catalogue, and an ILS catalogue respectively. Furthermore, the detail page of each lab, app, and ILS lists the informative metadata. In the future, we have to really assess the usability of the Lab Repository through user evaluations.

Findable. Obviously, finding resources is essential for the Lab Repository. Users can search on the Lab Repository through different means. A faceted search is integrated for labs and apps, which makes content search intuitive by a couple

of clicks. A search box is embedded the header of every page, which enables keyword-based search. Furthermore, different explorative navigation mechanisms are available. For instance, labs and shared ILS can be explored through the use of the Big Ideas (see D2.2).

Desirable. We promote the integration of images on the Lab Repository, e.g., screenshots and logo photos of labs. Apps also have a preview field (the app preview) to enable teachers to try out the lab on the lab repository to give an idea of their purpose after integration in an ILS. Big ideas also have their representing icons to attract users. Furthermore, when teachers share an ILS on the Lab Repository, the Go-Lab consortium can award social badges to it to make it more desirable for teachers to share their ILS (Note: currently, only the 'Go-Lab approved' badge is used).

Credible. User communities are integrated via social commenting. Teacher communities can review and rate the labs, apps, and ILS to improve the credibility of the lab repository. Furthermore, a social badge system can enable users to identify high quality resources (see above).

Accessible. The platform is responsive to different screen sizes and is accessible via mobile devices. The images and metadata display fits mobile devices. Note that not all lab and app previews can be displayed on mobile devices due to size or technological limitations (e.g., Java applets).

User testing is proposed as an efficient means to improve user interface (Nielsen, 2012) and different elements of the honeycomb can be targetted in separate evaluations. We have prepared surveys for different user groups. During the Go-Lab Summer School in Marathon in July 2014, we conducted a desirability testing study (Benedek & Miner, 2002) with 25 teachers. Each teacher was able to select as many words as they like from a predefined list to express their positive and negative impressions. Figure 3 shows the results of this study.

The most popular words to describe the Go-Lab Portal are *Attractive, Accessible, Creative, Easy-to-use, Innovative, Organised*, and *Useful*. This survey has been embedded in the Lab Repository, so any user can participate.[5]

2.2.2 Implementation of the Portal Interoperability

This section discusses the implementation of the portal interoperability specified in D5.2.

Lab interoperability

As specified in D5.2, lab interoperability is realised via compliance to Smart Devices and the Smart Gateway (see D4.1 and D4.3). Enhanced lab interoperability contributes to a federation of online labs. To further federate labs, we categorise them using the *Big ideas* (see Deliverable D2.2). Big ideas of science can cluster different online labs even in different subject domains. Figure 4 shows a screenshot of four of the eight Big Ideas on the Lab Repository.[6] Fig-

[5] The survey is available at `http://www.golabz.eu/about-go-lab-portal`.

[6] `http://www.golabz.eu/big-ideas`

Figure 3. Survey results of Go-Lab Summer School 2014 - teachers' impression on Go-Lab Lab Repository

ure 5 shows how these Big Ideas are connected to the labs.[7] Each lab can be related to more than one Big Idea.

Figure 4. A screenshot of a big ideas catalogue in Golabz

[7] http://www.golabz.eu/lab/electricity-lab

Figure 5. An online labs associated to several Big Ideas

Metadata interoperability

Since the development of the Lab Repository started before WP2 had finished the lab and ILS metadata, we started out with a limited set of metadata. Since the initial D5.2 specification, Work Package 2 and the technical cluster have refined the metadata scheme and conducted surveys with school teachers (see D2.2). Accordingly, the survey results are reflected in the Lab Repository. The metadata for labs, apps, and ILS are improved in order to describe those learning resources properly with consideration of teachers' opinions on metadata. These results and WP2 outcomes are gradually implemented in the Lab Repository.

For a better overview to input lab metadata, it is organised in a structured form with five tabs: General, Educational, Technical, Screenshots, and Big Ideas as depicted in Figure 6. Currently such lab metadata annotation is done by Go-Lab partners and is not publicly available for external lab owners.

In this release, we have not decided yet on the exact metadata exchange format we will adopt. One possibility is to extend the GOLC metadata schema (Richter, Grube, & Zutin, 2012). The final implementation of the lab interoperability will be described in D5.6.

Interface interoperability

For the interface interoperability, a RESTful service is developed in the Lab Repository for other platforms to access and retrieve metadata. The CRUD operations can be specified to resources. Thus, metadata can be created, read, update, and deleted remotely if this functionality is needed. JSON and XML are supported for requests and responses. These services can be used by other Go-Lab and third party platforms and services, e.g., the Learning Analytics service (see Deliverable D4.2). For example, when this URL `http://www.golabz.eu/rest/node?parameters[type]=lab` is called, the REST server returns all labs with their basic information such as id and title in XML.

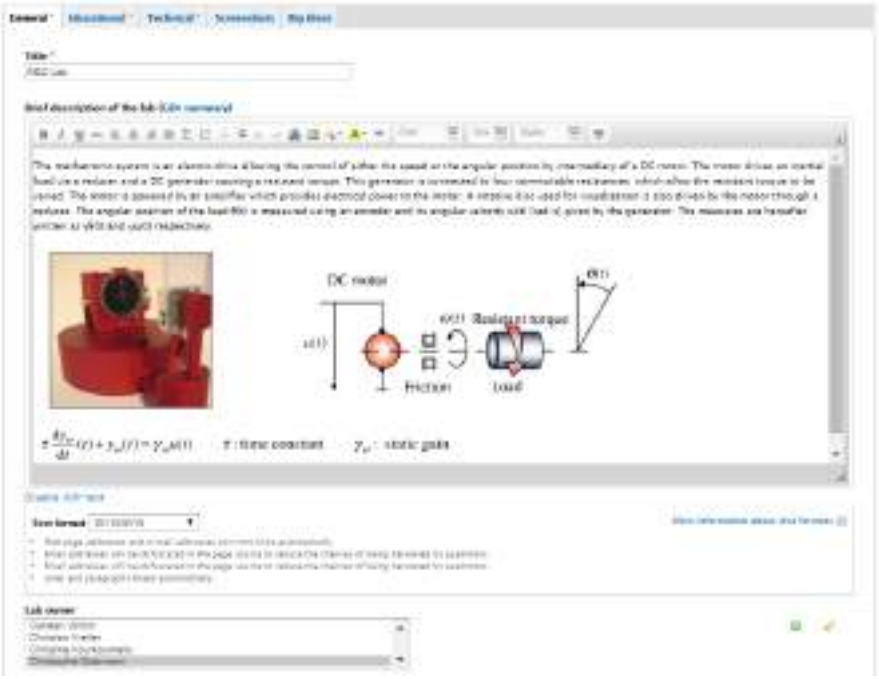

Figure 6. Structured lab metadata input form

2.2.3 The Drupal content management system

The Lab Repository is realised using Drupal 7,[8] which is a content management system supported by PHP, the Apache server, and MySQL database. Each resource in the Lab Repository is handled as a node belonging to a content type, i.e., the lab, app, ILS, and Big Idea content types. Each content type is specified with its metadata attribute fields. Some content types may share common metadata fields with different labels. For example, all labs, apps, and ILSs share the field creator to store the information about lab owners and app or ILS creators. In the back-end database, all this information is stored in the same table, while it has the different labels in the front-end, according to the content type it belongs to.

For the layout in Drupal 7, views are used to a great extent. Each content type has two basic views: default and teaser. They are used to present the detail page and the catalogue of labs, apps, or ILS. In addition, the homepage is also another view of all nodes in grid.

2.2.4 Technical server set-up and back-up

Developer, staging and production servers are established for collaborative development and a smooth launch of Golabz. The staging server (`http://staging.golabz.eu`) is employed to show the Go-Lab consortium members the new features of the Lab Repository. The new features are migrated to the productive server after approval.

[8] `https://www.drupal.org/`

2.2.5 Responsive design

To support ubiquitous access to the Lab Repository, it is developed using responsive design to adapt the UI to the hardware specifications of different devices (e.g., the UI adapts to the screen size of a tablet). The Lab Repository uses the Foundation 5[9] framework, which supports responsive design and development quickly. One of Foundation features, Grid, offers the default 12-column Foundation grid. The 12 columns can be applied to display content in small, medium, and large screen by giving an appropriate CSS class, e.g., class="small-2 large-4" in HTML. Thus, the responsiveness of the UI is fulfilled via using this predefined Foundation styling classes.

2.3 Requirement fit

Here we show how the Lab Repository fits the requirements elicited in D5.2.

2.3.1 Publishing Labs

The Lab Repository supports lab owners to publish their online labs with a selective metadata set. Currently, the labs are added by Go-Lab partners through Drupal's node creation forms. We do not have a public form where external lab owners can add their labs. However, in the near future we plan to integrate the Google Form created by WP2 as a first step towards publishing labs by an external lab owner. Figure 7 shows the *Electricity* virtual lab.

Figure 7. A virtual lab published in the Go-Lab Portal

[9] http://foundation.zurb.com/

The lab logo and the Big Ideas icons illustrate the lab. The brief metadata sets are listed on the right side of the lab logo, while the comprehensive metadata sets are placed on the lower part of the page, with screenshots at the end. When each screenshot picture is clicked, a pop-up window shows the zoom-in image with a navigator bar on the bottom as depicted in Figure 8. Thus, besides text metadata, the Go-Lab Portal attempts to employ more images to build an illustrative platform for a better visual impression.

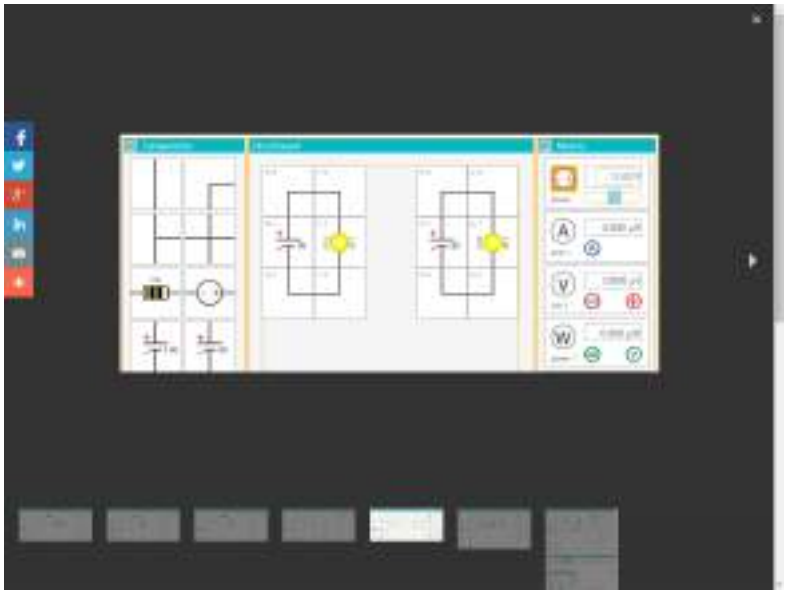

Figure 8. The metadata data *lab screenshots*

The subject domain is expanded with a tree structure to display the relationships among subject domains. Figure 9 illustrates the subject domain of the lab Osmotic Power Lab.[10]

Figure 9. An example of a lab's subject domains

[10] http://www.golabz.eu/lab/osmotic-power-lab

2.3.2 Creating ILS

From the Lab Repository, it is possible to either create a copy of an existing ILS previously published by a teacher, or create a new ILS embedding a specific lab. On each inquiry space page on the Lab Repository, the *Create a copy of this space* button instantiates a copy of the ILS on the ILS Platform and redirects the user to the newly created ILS on the ILS Platform (Graasp), which users can then modify, exploit and possibly republish (see next section for more details about the ILS Platform). On each lab page, the *Create Inquiry Space* button connects to the ILS Platform where users will find a new empty ILS with the selected lab in the Investigation phase (see the orange button under the lab logo in Figure 7).

2.3.3 Publishing ILS

Teachers can share the ILS they have created with the teacher community on the Lab Repository. Such a shared ILS can be reused by other teachers as described above. To share an ILS a teacher has created on the ILS Platform, she can click the Publish inquiry space button in the Share tab, see Figure 10.

Figure 10. The Sharing tab of an ILS page on the ILS Platform with the button to Publish inquiry spaces

Once clicked on this button, the ILS data structure and some metadata available on the ILS Platform (e.g., user info and space info) are sent to the Lab Repository. The teacher is then referred to an online form available on the Lab Repository, see Figure 11. Here the teacher is asked to fill in metadata on the ILS. However most of the metadata is pre-filled because most information can be derived from Graasp and the metadata of the lab used in the ILS. We use the lab metadata because we assume the metadata of the ILS will be highly correlated, e.g., the subject domains of the lab are likely similar or identical to the ILS metadata. The teacher can however edit all metadata in case the generated metadata is not correct. After editing, the form can be saved and the ILS will be available on the Lab Repository.

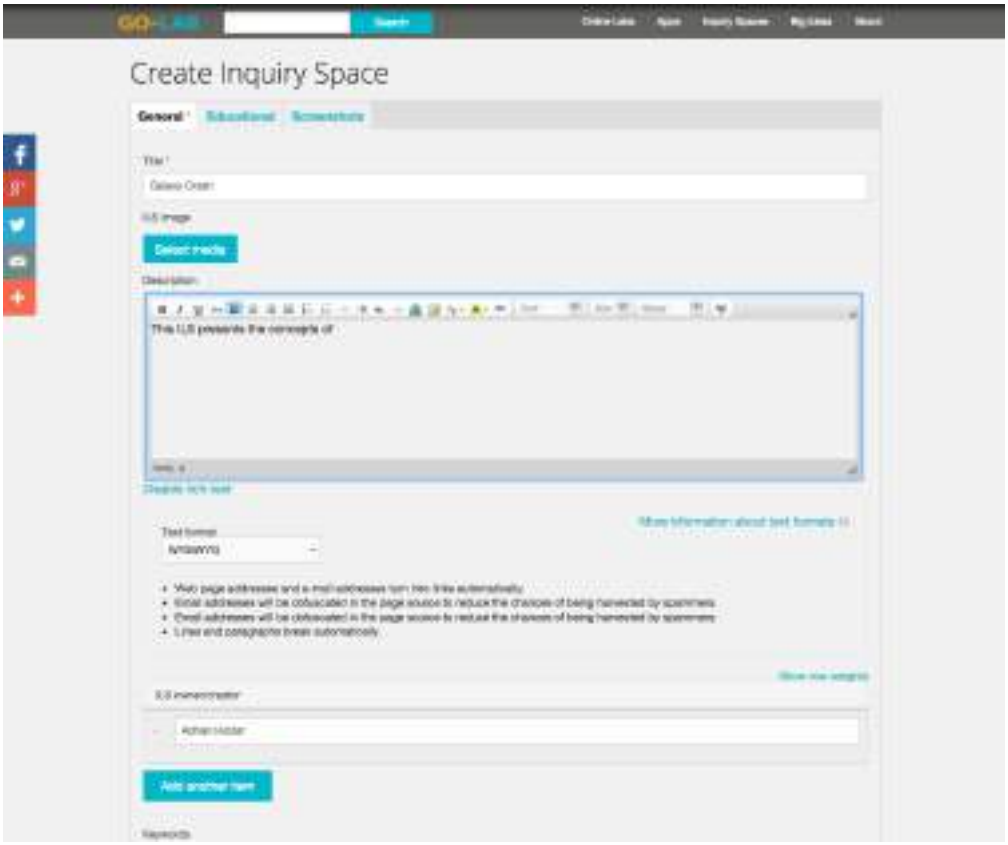

Figure 11. The ILS metadata form on the Lab Repository.

2.3.4 Supporting Apps

The Lab Repository hosts inquiry learning apps as a main resource, besides labs and ILSs. The app catalogue is accessible via the menu item *app* in the top bar in Golabz and at `http://golabz.eu/apps`. Each app is also displayed with their metadata including subject, app type, app creator, category, license, source code, keyword, and description (see Figure 12). Besides a simple app logo on the top, an app preview is implemented for users to test out the app before they make a decision to select it for their inquiry space. The app preview is realised in an iFrame which renders the app's source code using the Application Container of the ILS Platform (see Section 3.2.2). Thus, users are able to play with these apps before embedding them in an ILS.

2.3.5 Supporting Learning Scenarios

Learning scenarios demonstrate the practical use cases of the learning resources in the lab repository. They were called lesson plans in D5.2 and changed to learning scenarios based on feedback of the pedagogical cluster. They are embodied as inquiry spaces. As the third main resources in the Lab Repository, Inquiry Learning Spaces also have a simple catalogue with a list of full inquiry spaces and mini inquiry spaces (see Figure 13). These inquiry spaces are evaluated from a pedagogical perspective by Work Package 1. Figure 13 displays an ILS in the repository together with its metadata fields (e.g., grade level, sub-

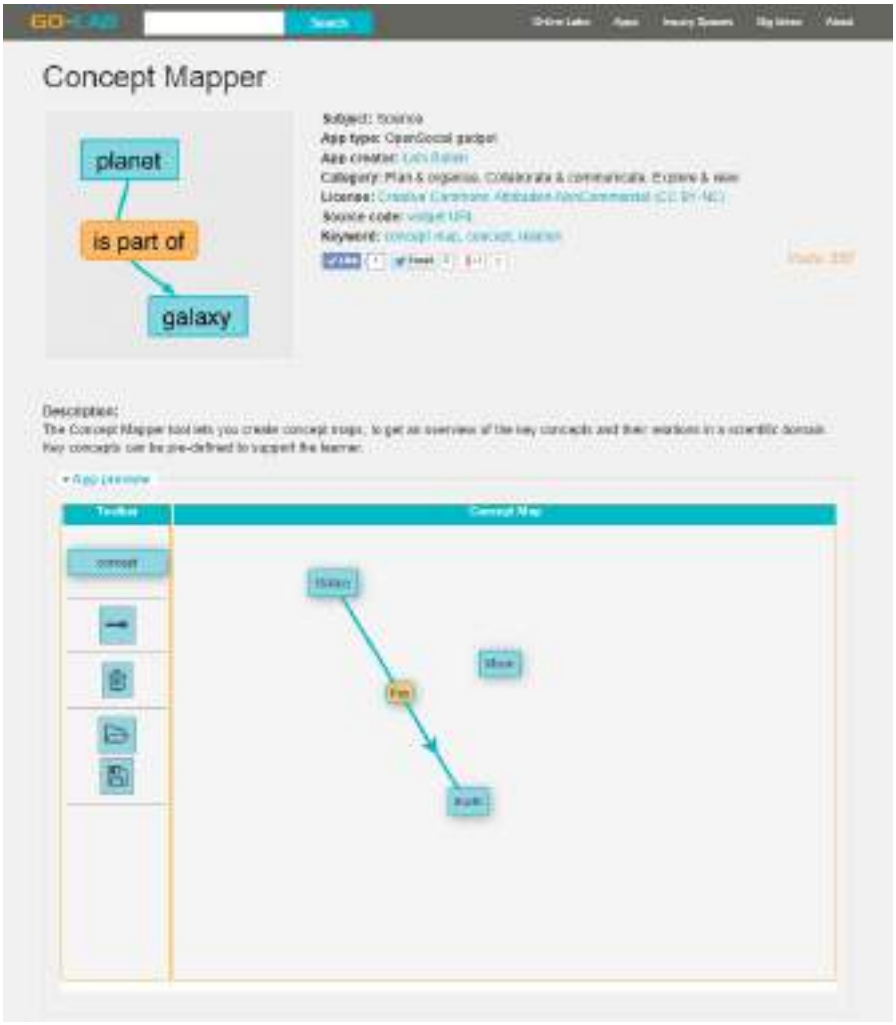

Figure 12. An app page in the Go-Lab Portal

ject, and language) on the right side of the ILS logo image. More comprehensive metadata including description and learning objectives are placed below the logo. Screenshots can illustrate the ILS in Standalone View (i.e., the view that students will typically access). The ILS consists of a set of inquiry learning phases which are display in an accordion view. Thus, by clicking different inquiry learning phases, it opens a sub-frame to show their resources respectively. For example, the teacher renamed the standard *Investigation* phase to *Experiement!* and it contains the *Splash* lab.

2.3.6 Searching labs, apps & ILS.

With a variety of content in the lab repository, it is necessary to develop a search mechanism. We have developed two main search approaches in the lab repository: faceted search and free-text search. Figure 14 shows the lab catalogue after applying the faceted search filter on the right side. It consists of subjects, grade levels, languages, interaction levels, difficulty levels, lab types and keywords. Each term is clickable to apply the filtering mechanism. More terms can be selected or cancelled by clicking the “(-)” in the front. And they can also be

Figure 13. The inquiry space in the portal

used in combination. The example in the figure shows the filter to list all online labs whose subject domain is astronomy, for the grade levels of secondary education in both English and French with a difficulty level of medium. The free-text search input form is placed in the top bar and can be accessed from any page. It returns a list of results which contain the searched terms. We have used the Drupal Integrated Search Server to enable search. The search server makes an index of all nodes in the repository. This search server returns appropriate results with the current repository. With the further increase of the content, we consider to employ Elastic Search[11] to enhance the search results and speed up the search processing.

[11] `http://www.elasticsearch.org/`

Figure 14. The ILS catalogue in the Go-Lab Portal

2.3.7 Social features

In order to involve more users and motivate their interactions with the Go-Lab repository, we have integrated two Web services AddThis[12] and Disqus[13] to implement the features of social rating and social comments. The AddThis Web service is integrated in two ways. First, the sharing sidebar is applied and a side bar with Facebook like and Twitter etc. is displayed on each page. Second, on the page of online labs and apps, the original sharing buttons are used to show how many Facebook likes or Google plus that lab or app has received. For social comments, the Lab Repository employs the Disqus Web service to enable users to leave their comments to labs, apps, and ILS without the need to log in on the Lab Repository. Users can quickly log in with their Google, Facebook, Twitter, or Disqus account, and leave a comment as illustrated in Figure 15. In addition, we have embedded social channel links in the footer.

[12] http://www.addthis.com/
[13] https://www.disqus.com/

Figure 15. The social comment box in the Go-Lab Portal

2.3.8 Tracking user activities

Piwik and Google Analytics are both applied to check the Web traffic of the Go-Lab Lab Repository. [14] Relevant statistics about the number of visits are directly displayed on the Lab Repository. This tracking information will be exploited in the future by the Learning Analytics service (see D4.1) and the recommender engine (see D5.1).

[14] The Piwik server is set up at `http://piwik.golabz.eu/`

3 The ILS Platform

Graasp, the ILS Platform, is accessible at `http://graasp.eu/`.

3.1 Introduction

Graasp is a social media platform designed to support the requirements elicited for collaboration in online learning communities.For learning communities, it is typical to collaborate in groups, collect information from different sources and disseminate it to others. More broadly, Graasp supports many types of collaboration including information management. We present the main concepts of Graasp and its architecture.

3.1.1 General Concepts

The main concepts of Graasp are illustrated in the user interface screenshot in Figure 16.

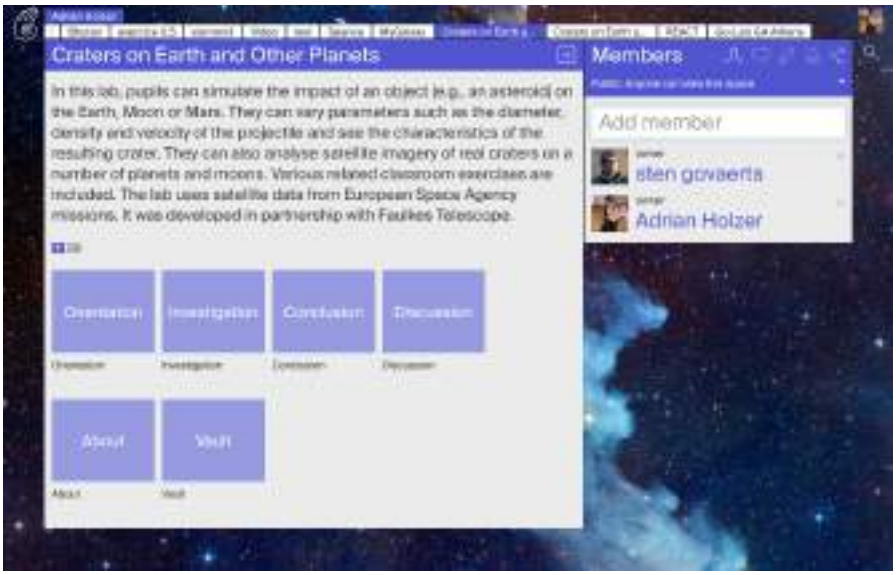

Figure 16. ILS in Graasp

Spaces. A space is the central concept in Graasp. A space encapsulates the context of a collaboration. A space in Graasp can be loosely compared to a folder with associated permissions. A spaces s can include another space s'. We use the family tree metaphor and say that s' is a child of s and that s is the parent of s'. Each space has a *name*, a *description*, a set of *sub items*, and an associate space information. The space information includes a set of *members* with *permissions*, associated *discussion* threads, an *edit* option for settings, information about the *activity* in the space, and social *sharing* information. The space information is gathered in the right panel of the user interface in tabs as illustrated in Figure 17.

Navigation within a space tree is done through a top navigation component that shows the tree structure. For each space, its siblings, its ancestors and their siblings are shown.

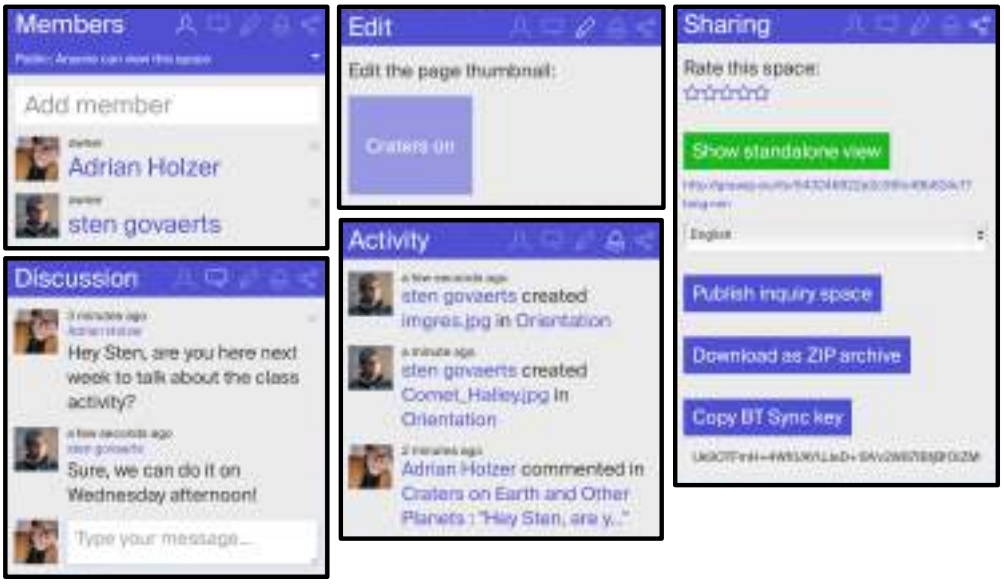

Figure 17. Space information in Graasp

Items. Each space acts as a container for *items*. Items can be of three main types: (1) spaces , (2) resources (coming from the local disk and the cloud), and (3) apps.

Members. Each space has a set of members. By default, when creating a child of a space, members are inherited from with the same permissions. Also, when adding a member to a space, this member gets by default automatically added to all descendant spaces with the same permissions.

Permissions. Each member – an individual or a group – of a space has a defined permission in this space from the set of permissions {`owner`, `editor`, `viewer`}, where owners have more rights than editors, which have more rights than viewers. Each space must have at least one owner. An owner of a space is also an owner of all its descendants:

Visibility. Each space can be shared with others, either by setting its visibility level to *public*, which makes it accessible to anyone on the Web, or by keeping the space *private* allowing only the space members to view and work with the space content. With the search functionality offered by Graasp, the content of public spaces can be found by anyone. In the future, public items can be set to unlisted so they do not show up in the Graasp search and are not indexed by external search engines. Users can still access unlisted public items by their URL. When a new item is created, it inherits the visibility level of the parent item, but afterwards the level can be changed.

Inquiry learning spaces. Graasp supports both simple spaces and inquiry learning spaces (ILS). ILS are the typical spaces created and shared by teachers through Graasp and Golabz. ILS differ from simple spaces in that they come with 7 child spaces when they are created. These 7 spaces are the five inquiry phases as sub spaces as well as an About and a Vault space. Furthermore they can be shared as *Standalone View* through a secret URL and only require a nickname to be accessed. Figure 18 depicts the Standalone View of an ILS. The upper bar contains the title of the ILS – on the left – and the student's nickname – on the right. The tabs represent the inquiry phases (orientation, conceptualization, investigation, conclusion and discussion) and, in the specific case of the *Orientation* phase, this example contains a video and a concept mapper app (more information about the ILS specifications can be found in D5.2).

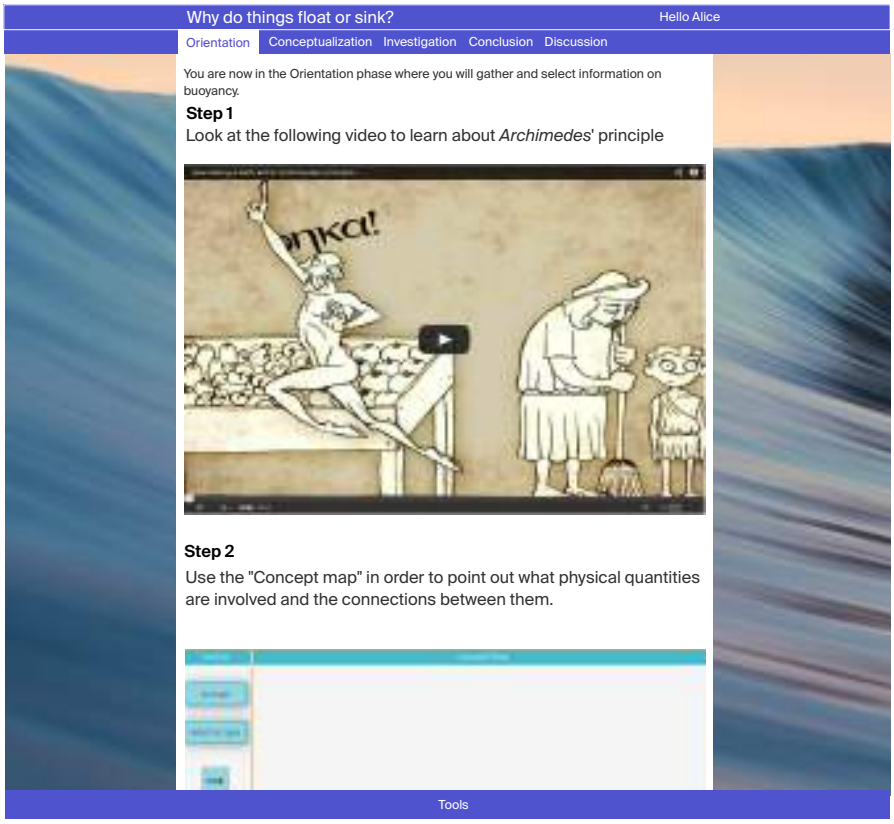

Figure 18. ILS Standalone View

Finally, the toolbar at the bottom of the screen is displayed if the teacher has added some apps to the space. This toolbar is always visible to the students and offers immediate access to all of the tools that were added during the authoring of the ILS. The design of the Standalone View is the result of participatory design with WP1 (see D1.1) and WP3. For instance, the mapping of the inquiry cycle to tabs, the internal structure of a phase (a unique scrollable page) and the position of the toolbar are the results of the findings of D3.1.

3.2 ILS Platform Architecture

The overall ILS Platform architecture, illustrated in Figure 19, consists of three main parts: (1) Graasp, used as a ILS editor, (2) the Application Container rendering apps and (3) the ILS Standalone View implemented as a *metawidget*.

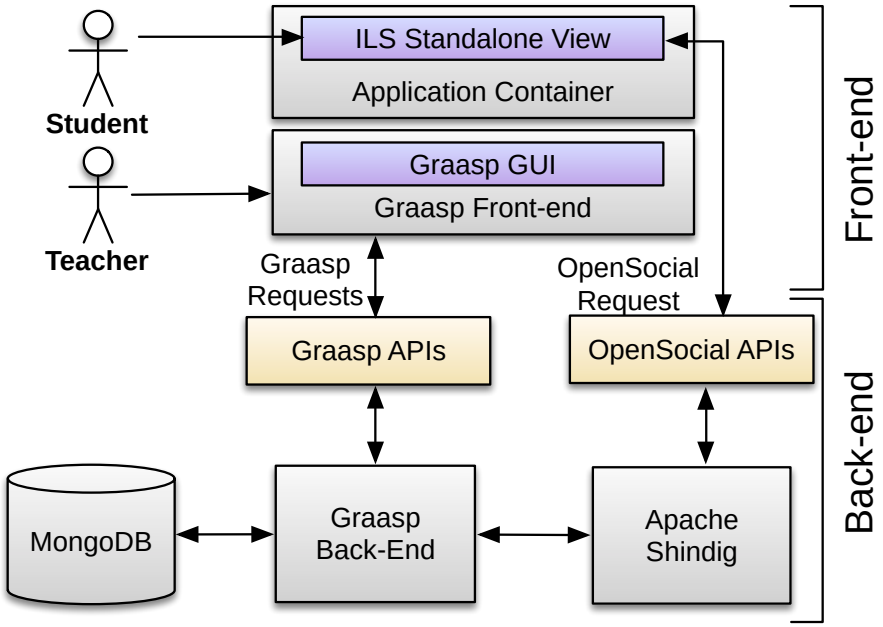

Figure 19. The ILS Platform architecture

3.2.1 Graasp

The Graasp online system is a single page web application (SPA) using JavaScript end-to-end. As such, it is composed of the front-end part, running in a browser and the back-end part installed on a server.

The front-end is developed with the AngularJS[1] framework. Thanks to the AngularJS templates, only the required data updates are exchanged between client and server instead of complete HTML pages. This saves bandwidth and decreases latency leading to a more responsive web page, which is particularly important in places with limited Internet connection. In terms of the responsiveness of the user interface (UI), the web application is based on the Bootstrap framework[2], which allows to optimize and adapt the UI for tablets and mobile phones. To ensure ease-of-use, the Graasp UI reuses familiar concepts from existing desktop and web applications. For instance, Graasp allows uploading both individual files and folders with a simple drag and drop action. Graasp is able to aggregate resources both coming from the local computer (e.g., documents, pictures or videos) and coming from the web (e.g., YouTube videos, SlideShare presentations). If the web page content can be recognized, Graasp will create a resource with the embedded content. This resource can then be viewed from within Graasp.

[1] https://angularjs.org/
[2] http://getbootstrap.com/

On the back-end, a Node.js[3] server is handling the business logic and the structured data is persisted in a MongoDB[4] database. Uploaded binary content (e.g., PDFs, MS Office documents or video files) is stored on the server file system in a folder hierarchy corresponding to the space hierarchy in Graasp. As soon as a resource (a file or a web content) is uploaded, the text information is extracted and indexed to facilitate full-text search. This process relies on the MongoDB text indexes.[5] The data exchange between the front-end and the back-end is done by using JSON format via a RESTful API or WebSockets for real-time update propagation (e.g., change of the space name).

3.2.2 Application Container

Graasp implements an OpenSocial [6] container which supports OpenSocial apps (sometimes referred to as widgets or gadgets). Apps are then rendered in Graasp by the corresponding implementation of Apache Shindig Server.[7] Apps are developed according the Gadgets XML Reference[8] and can exchange data with the Graasp database via the OpenSocial API (OSAPI). The supported APIs are: *People*, *Applications*, *Appdata*, *ActivityStreams*, *Spaces* and *Documents*.[9] This enables integration between Graasp and apps, since an app developer can retrieve and save Graasp data related to for example: spaces, people and applications. Information flows between Graasp and Shindig occur as follows:

1. The Graasp Front-end recognizes an app and sends a request to the Application Container (OpenSocial Container) to render the app
2. The Application Container contacts the Apache Shindig Server via the OpenSocial APIs and provides it with the code to render the app
3. Apache Shindig runs the app in the Application Container in the Graasp Front-end and the app can access Graasp data via the OpenSocial APIs.

3.2.3 ILS Standalone View

The implementation results of the Standalone View are illustrated in the *ILS metawidget*, which is an open OpenSocial application that communicates with Shindig for the visualisation of the ILS. The ILS metawidget is made up by the container of the ILS (an XML file), the component in charge of the visualisation (a JavaScript file), and the graphical elements (CSS files and images). The Standalone View has been developed following a responsive design approach. Accordingly we decided to use HTML5 that encompasses HTML, JavaScript, CSS3 and HTML5-friendly media. We use the Twitter Bootstrap framework to offer a consistent user experience across all devices and platforms, as it is illustrated in Figure 20.

[3] `http://nodejs.org/`
[4] `http://www.mongodb.org/`
[5] `http://docs.mongodb.org/manual/core/index-text/`
[6] `http://opensocial.org/`
[7] `https://shindig.apache.org/`
[8] `https://developers.google.com/gadgets/docs/xml_reference`
[9] Documentation on the use of the APIs can be found here: `http://opensocial-resources.googlecode.com/svn-history/r1393/spec/2.0/Social-Gadget.xml`

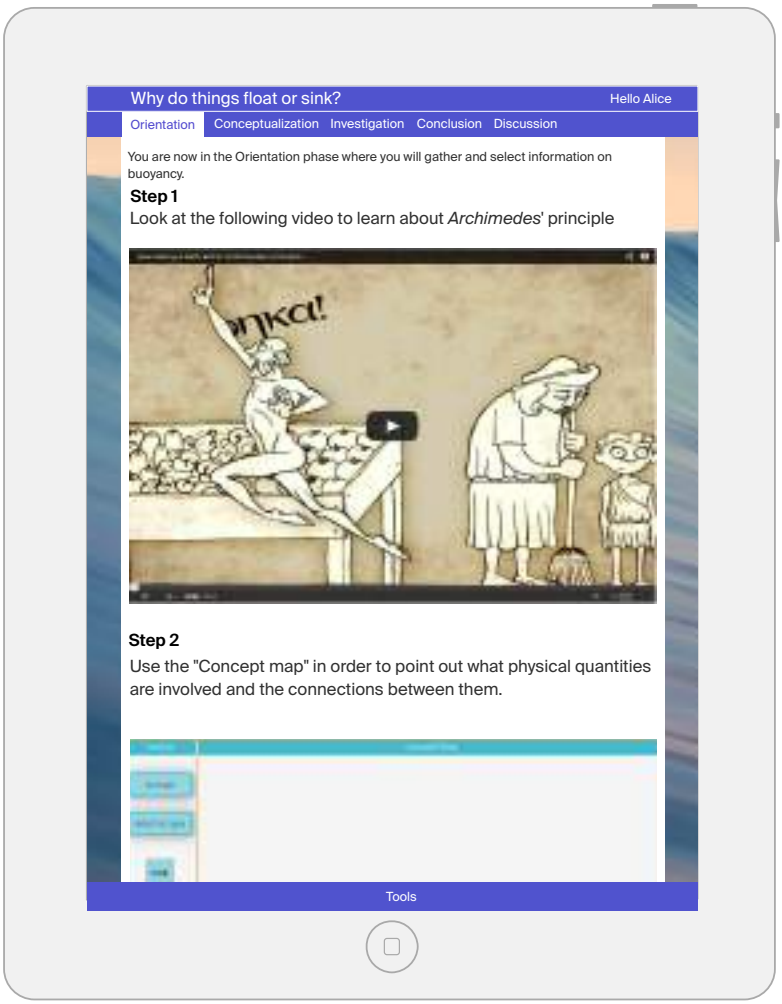

Figure 20. The ILS Standalone View on a tablet

3.3 Requirement fit

Here we show how the ILS platform fits the requirements specified in D5.2

3.3.1 Creating ILS

Lab owners create ILS to demonstrate a lab and teachers create ILS for students. ILS can be created from scratch in the ILS platform. To that end the user presses the + button in the space and then pressed New Inquiry Space. This creates a new ILS with the five predefined spaces of the inquiry cycle (orientation, conceptualisation, investigation, conclusion) as well as the Vault and the About spaces. As described in D5.2, the five spaces of the inquiry cycle are used to gather resources and apps that are displayed in the Standalone View. The Vault and the About spaces are not displayed to students. The Vault space is used as data repository for apps. For instance, the DropFile app stores the files dropped in it in the Vault. Teachers can then typically go to the Vault to see student reports. The About space is the place where teachers can store tutorials, FAQs, and tips on how to best make use of their ILS.

Teachers can then add items in each phase and add descriptions to each items. The ILS platform provides a tile view and an expanded view. The tile view gives an overview of the items in a space by showing only a thumbnail for each item. In this view items can easily be re-ordered by drag-and-dropping them in the desired place. The expanded view shows each item below the other with its description. In this view teachers see the items as they are shown in the Standalone View. To see the Standalone View, the teacher can go to the share tab and press the Show Standalone View button in the Sharing tab as shown in Figure 17.

3.3.2 Modifying ILS

Teachers adapt existing ILS. When an ILS is created from the Lab Repository, it can be modified in the ILS Platform. For instance, teachers can organise the inquiry cycle by reordering phases, removing or renaming some and adding new ones. Furthermore, teachers can add or remove items in each phase and change descriptions. Finally, the language of the Standalone View can be modified independently of the ILS language.

3.3.3 Publishing ILS

Teachers and lab owners publish their ILS to enable reuse. Once an ILS has been created or modified in the ILS platform, it can be shared with other teachers either by inviting them as members into the space, or by publishing it on the Lab Repository. Inviting teachers as members of the ILS means that they can potentially collaborate in the elaboration of the ILS. It also implies that student data will be available to all invited teachers. This type of sharing is most suited for a team of teachers from a same school working together. Publishing an ILS on the Lab Repository implies that any other teacher who visits the Lab Repository can create their own copy of the ILS and edit and use it independently. In order to publish the ILS on Golabz, users must click the Publish on Golabz button in the share tab. The data from the ILS is then transferred to the Lab Repository and the teacher is asked to complete some metadata before publishing the ILS publicly on the Lab Repository, see Section 2.3.3.

3.3.4 Using ILS

Teachers run activities using ILS. Students use ILS provided by teachers to conduct experiments. As shown in Figure 17, the teacher can access the Standalone View by pressing the *Show Standalone View* button in the Sharing tab and can share it with students using its secret URL. The student then simply types a nickname to access it as shown in Figure 21. As students are not registered through Graasp, only a nickname is required for authentication rather than a full username / password set. After they enter a valid nickname, the main ILS is displayed in the Standalone View. The overall ILS layout consists of four main elements: a top information bar, a main navigation bar, the ILS body content and a bottom tool bar, as illustrated in Figure 18.

Figure 21. The ILS Standalone View welcome screen

3.3.5 Supporting guidance apps

Here we conjointly address the *supporting apps* and *scaffolding* requirements presented in D5.2 which stated: *Students are supported in their inquiry learning through apps (e.g., a hypothesis app or online lab interfacing apps). Teachers monitor student progress through learning analytics apps. Students receive assistance from scaffolding apps (e.g., prompts and feedback) based on learning analytics and teacher configurations.* As mentioned above OpenSocial apps can be added to any phase in the ILS (see D5.3 for more information about the developed guidance apps). When AngeLA permits it (see Section 3.3.9), apps in an ILS can access the recorded Activity Streams (either directly or through the OpenSocial API and the Learning Analytics backend, see D4.2 for further details), and they can change their behaviour based on them. This can be used by app developers to provide students with real-time prompts and feedback.

3.3.6 Supporting learning scenarios.

Teachers create lesson plans for ILS. Students use lesson plans provided by the teacher when studying in ILS. As mentioned above, ILS are the embodiment of learning scenarios. The ILS platform supports their creation, modification and exploitation.

3.3.7 User management.

To access all of the portal's functionality, users log in only once. Several login options are provided depending on the acceptable privacy level. Teachers log into the ILS platform with their email addresses and a password. Students typically access the Standalone View of an ILS by only providing a nickname. Thus the ILS Standalone View provides privacy-by-design since the system does not record a formal identification for students. With nicknames a student can find

her work where she left it if she logs in again at a later time with the same nickname. Furthermore, we have implemented a prototype of authorization between the Lab Repository and the ILS Platform, where the ILS Platform is an identity provider that enables users to login with the ILS Platform account into the repository. Currently, this mechanism using OAuth2 is not in production yet, but it will enable teachers in the near future to effortlessly share ILS on Golabz with their Graasp account. Additionally, this OAuth2 authorization mechanism will also be used to log in users in the App Composer and Smart Gateway in the future.

3.3.8 Social features.

Teachers and lab owners tag, comment and rate labs and ILS, and share them on social networks. Members of an ILS can leave comments in the Discussion tab of the space. Furthermore, they can share the Standalone View of an ILS on Facebook and Google+ in the Sharing tab. Each space also has a rating system that allows to assign a rating to any space in the ILS platform.

3.3.9 Tracking user activities.

The activities of portal users are tracked and used for learning analytics, recommendation and scaffolding apps. The ILS platform records user action through Activity Streams. The activities are displayed in the Activity tab in each space. These activities can also be visualized using dashboard apps. We have presented some of them in D5.3 and are currently working on other apps to be used in dashboards (see D4.4). Whereas actions of space members are always recorded, actions performed on the Standalone View are only recorded if the AngeLA tracking agent is member of the spaces. More details about AngeLA are presented in D4.2. We plan to provide recommendation of resources, and apps to teachers at the ILS level.

3.3.10 Non-functional requirements analysis

The non-functional requirement include ubiquitous access, usability & data privacy, as well as, scalability & availability.

Ubiquitous access. The ILS platform is accessible online with any modern browser. So schools do not need to spend resources on installing and administering software. To provide lab federation and to support a common ILS platform, interoperability of the labs is essential (see D4.1). When targeting school students, special attention to usability and data privacy (e.g., anonymizing the tracked user activities) is needed.

Usability & data privacy. As mentioned, in terms of data privacy, we provide partial anonymity-by-design for Students by only requiring a nickname when they access an ILS Standalone View. Furthermore, privacy is further guaranteed by the fact that student traces are only recorded when the AngeLA tracking agent is added as member to an ILS (see D4.3).

In terms of usability, the redesign of the ILS platform follows a user-centered design approach (Constantine & Lockwood, 1999) and based on the participatory design activity outcomes documented in D3.1 and D3.2. For instance, following the structure and the visibility principles, related components in Graasp were grouped and clearly separated from others and unnecessary components are hidden to avoid distracting users. Following the simplicity and the reuse principles, adding and moving resources can now be done done simply through drag and drop of documents and folders from the desktop into Graasp. Furthermore, to find content a novel navigation component was designed and the search mechanism was improved.

Additionally, we evaluated individual features to decide which ones were problematic. We conducted a study with 38 master students who had used Graasp for an entire semester about the features that they thought were easy or hard to use, and the ones they thought were useful or not. The results are summarised in Figure 22. The top row shows features with a difficulty score mentioned at least ten times as difficult to use. The middle row shows features mentioned between one and ten times as difficult to use. The bottom row shows features that where never mentioned as difficult to use. The left column shows features with a usefulness score, i.e., the number of useful mentions - the number of not useful mentions, below -5. The central column shows features with a usefulness score between -5 and 5 and the right column shows items with a usefulness score above 5.

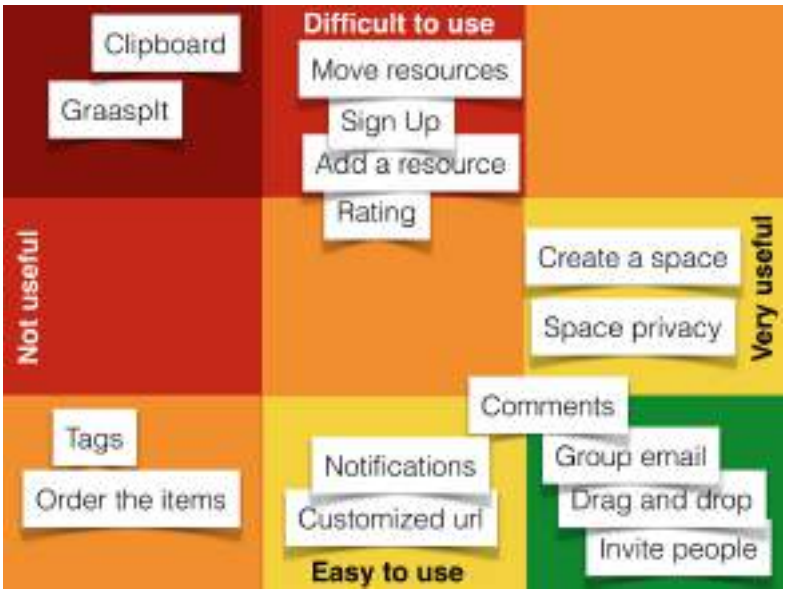

Figure 22. Graasp usefulness and usability study

Typically, one of the most important usability issues was the Clipboard concept on which the GraaspIt aggregation tool is also based. In the redesign we removed the Clipboard concept and have introduced a *Move To* dialogue to move resources, which was also found to be difficult. Currently, cloud resources can be added just by providing an URL and we are working on a mechanism to make

GraaspIt more user-friendly. Then we have redesigned the sign up and login interfaces to mimic typical mainstream social media platforms such as Facebook with which users are familiar. As users enjoy drag and dropping resources, we have extended the feature to allow to drag and drop resources anywhere in the space. As tags were not considered useful we have dropped this feature. We will further refine the usability of the ILS platform based on future feedback from Work Package 3. For more information, we refer to D3.2.

Scalability & availability We use Load Impact[10] to test the scalability of the service. We monitor the availability using pingdom[11] so we get notified every time one of our services goes down. Our uptime for graasp.epfl.ch for the last year is 99.6% Most of the downtime was due to maintenance. We have had probably 2-3 major downtimes (few hours) during the last year.

[10] http://loadimpact.com/
[11] https://pingdom.com

4 The App Composer

The App Composer is available at `http://composer.golabz.eu/`.

4.1 Introduction

The main goal of the App Composer Platform is to provide teachers with the means to take an app that someone else has developed, and to customize it to fit their needs and those of their students. Teachers do not necessarily have technical knowledge or willingness to spend a significant amount of time to customize these apps, so the App Composer must be easy to use and not require programming skills.

Hereafter, we will describe the architecture of the App Composer and its two modules, the translator module and the adaptor module. In D5.2, a builder module was also described, but its usefulness was not demonstrated and was thus not further pursued.

4.1.1 The translator

Often apps are not available in the student's mother tongue. If they are, the number of supported languages is generally low. This makes it particularly hard to share apps and ILS across different countries, especially because the students to who those apps and laboratories are oriented, are often young and do not master foreign languages sufficiently.

To solve this issue, the *Translator* lets teachers take an existing app or laboratory, and publish the user interface translations easily to their students or to the public at large. Thus, for instance, a French teacher with knowledge of English could easily translate an app or laboratory originally in English to French for his or her young French students.

To support different levels of language proficiency among students (e.g., students in elementary school typically have a smaller vocabulary than last year high school students), the *Translator* also supports specific translations for specific groups within the same language.

4.1.2 The adaptor

The purpose of the *Adaptor* is more general. The *Adaptor* will let teachers configure an existing app or laboratory (which must have previously been prepared for customization by its developer) for their students.

That is, starting with an existing app or laboratory, which its developer has created as a *template*, teachers would be able to choose which subsets of the content to show, to specify how to do so or to customize the experience in any other way. This way, teachers can very easily create and publish a version of that app or laboratory which fits exactly the content they wish to teach and the specific circumstances of their students.

4.2 Architecture

The App Composer is relies on the Python-based *Flask* microframework. This fact influences other design choices. The App Composer is split into different components. Some of these components, such as *user management* or *storage* are generic, and to be expected in most web applications. Others, however, are very specific to the App Composer. That is the case of the Composers themselves; the *Translator* and the *Adaptor*.

The App Composer platform integrates with the ILS platform, and most users will access the App Composer through the ILS platform. In the following sections the most important architectural details will be described.

4.2.1 General Overview

Figure 23 shows a high-level view of the App Composer and how it fits within the Go-Lab ecosystem.

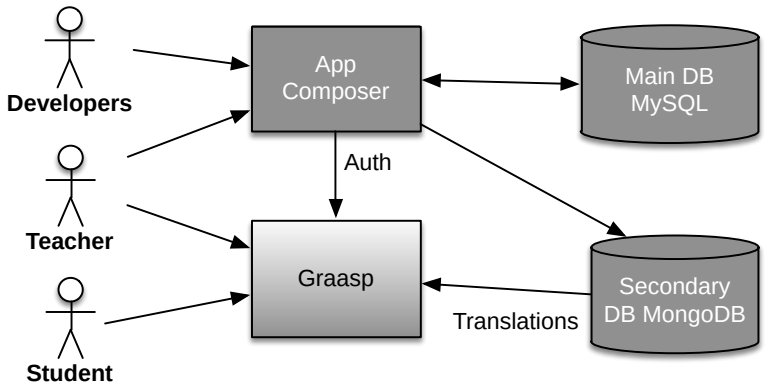

Figure 23. General Overview: Integration

The most common use-case will be through Graasp, though the Platform can also be accessed directly. It is also noteworthy that deployments consist of two different databases. The *MySQL* one is internal to the App Composer, and is meant to store all the information about translations and adaptations. The *MongoDB* one is used for the communication of translations between the App Composer and Graasp (or other external systems), and is less critical. This scheme will be described in further detail in later sections.

4.2.2 Translator's ILS platform Integration

One of the goals of the App Composer to integrate seamlessly with the ILS platform (and potentially with other external systems). Through the ILS platform, hundreds or even thousands of students should be able to access published apps and their translations.

If the translations were held only by the App Composer platform, then essentially for every user who accessed a translated App through the ILS platform, a request to the App Composer would be made. This would result in a very high load on the App Composer (which is not really meant for massive content delivery) and it would harm both scalability and availability. To solve this issue,

the ILS platform and the App Composer are decoupled through a secondary MongoDB-based database. That database can be hosted anywhere, and it mirrors the App translations. Thus, whenever the ILS platform needs to access a translation, it can retrieve it directly from that MongoDB. This relationship is depicted in Figure 23.

The way the translation mirroring system works is depicted in more detail in Figure 24. That figure shows that synchronization happens at two different stages. First, whenever a change in a translation occurs, the App Composer platform tries to synchronize it straightaway. Last, periodically, the whole repository of translations is synchronized. This way, even if the MongoDB database were to fail, be reset or be replaced, or if any other problem would occur, it would not be long before it would be automatically filled with up-to-date translations again.

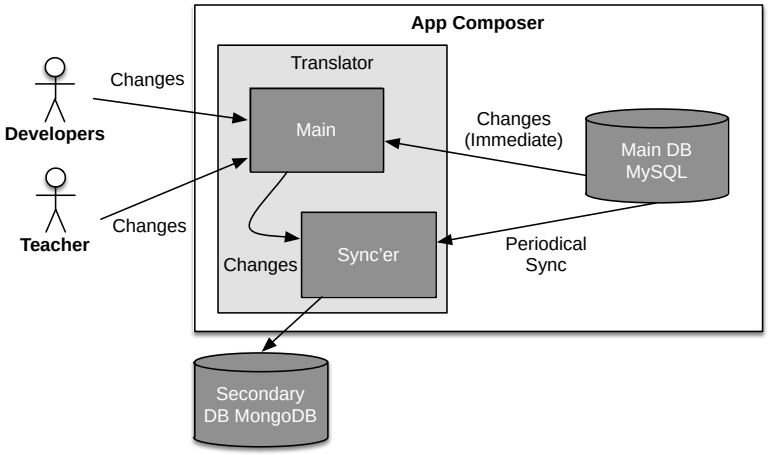

Figure 24. Translation mirroring scheme

4.3 Requirement fit

Here we show how the App Composer fits the requirements specified in D5.2.

4.3.1 Different languages

To support Go-Lab in European schools, internationalisation of apps is required (see deliverable D5.1). After choosing the app to translate, users are shown the language selection screen, which is depicted in Figure 25. Here they can choose the *source language* (from which language to translate) and the *target language* (to which language to translate).

4.3.2 Languages for different target groups

Students targeted by Go-Lab range from 10 to 18 years, which means that we have to deal with different language proficiency levels. Apps can be translated to and from these target groups. As Figure 25 shows, there is a default *ALL* group. However, users can also choose a more specific one, such as *English for Adolescents (14-18)*, as displayed in Figure 26. That language and target group combination can then be translated normally, as if it was a different language. This is displayed in Figure 27.

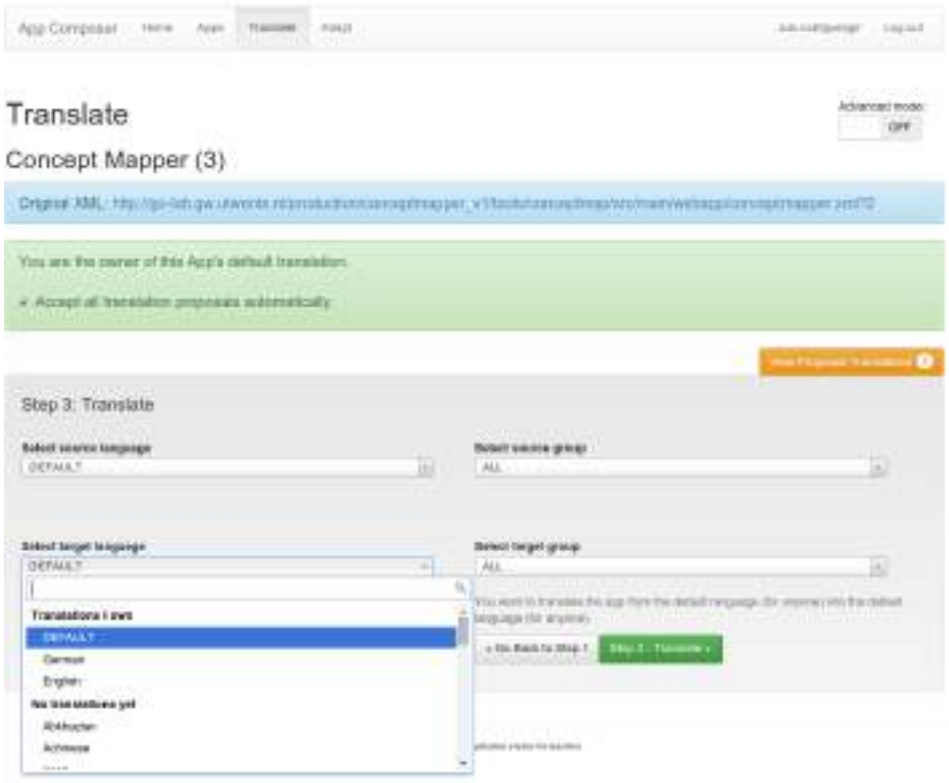

Figure 25. Choosing the language to translate

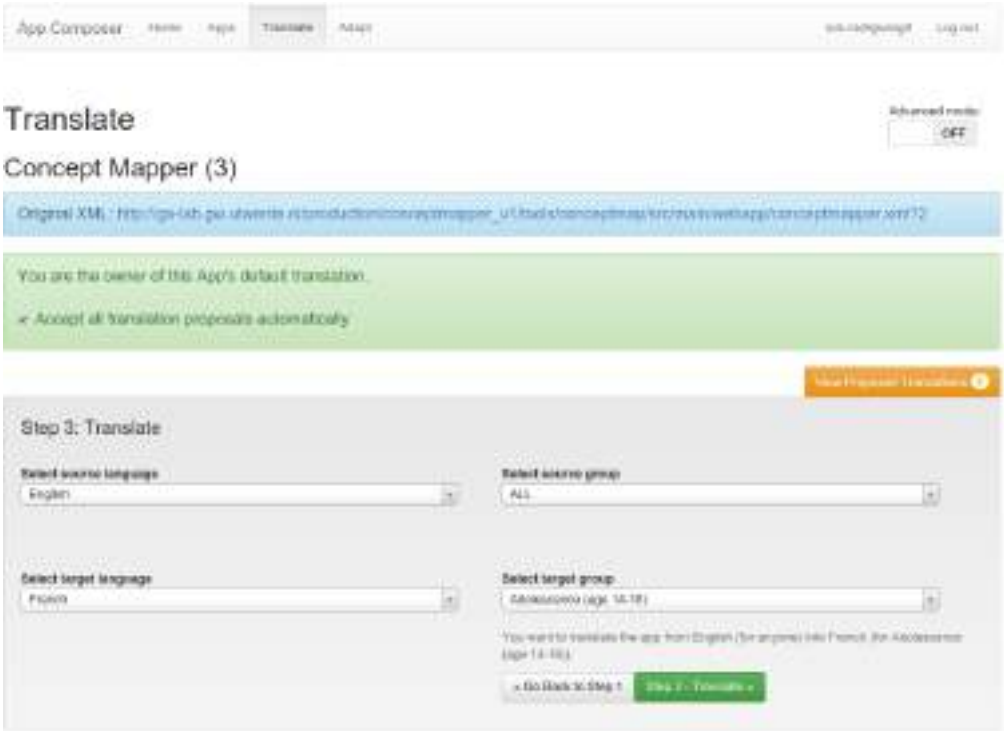

Figure 26. Choosing a specific target group

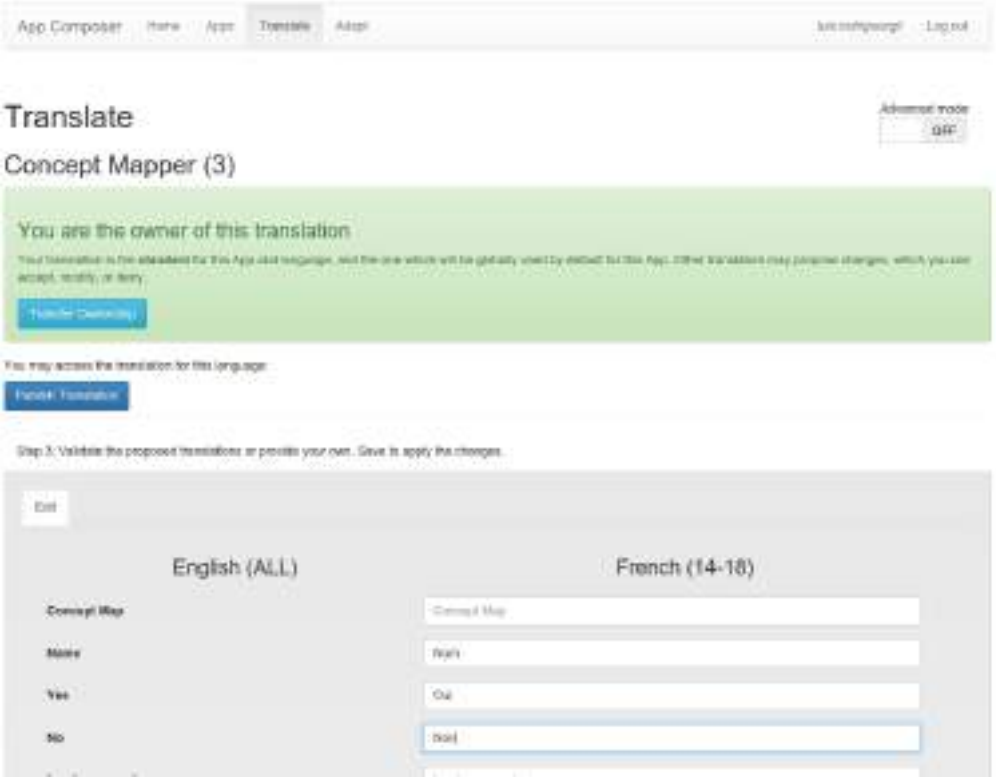

Figure 27. Translator's Edit Language screen

4.3.3 Adaptable app listing

A selection of apps will be available for teachers as a basis for adaptation. Because an app needs to be prepared by the App Developers themselves before they can be adapted, the list of adaptable apps is initially limited. However, it is expected to increase greatly in the future. The adaptable app selection screen can be seen in Figure 28.

Figure 28. Adaptor's App Selection screen

4.3.4 Adapting apps

Users can configure a selection of existing apps with data or models specific to their learning. Figure 29 shows the Adaptor's edit view. The different options and controls that appear depend directly on the app being adapted. The developers of that particular App must have previously prepared their app to be adaptable. In this case, the app being adapted is the Concept Mapper tool, created by the University of Twente.

Figure 29. Adapting a Concept Mapper instance

4.3.5 Publishing apps

Teachers can share the apps they have created or translated with the Go-Lab community by publishing them on the lab repository.

4.3.6 Portal integration

The app composer communicates with the portal to allow teachers to search for existing apps to translate and save apps to their ILS or lab repository. Figure 30 displays the App Selection screen. The list of apps to translate is provided directly by the Go-Lab Portal (through an API). To make it easier for users to find the app they want, a search feature is provided which can be used to filter the list in real time.

4.3.7 Draft support

Often users will require several sessions working with previously saved versions before finishing their applications. Figure 31 shows the application list of a user. The application list contains those application instances which the current user has begin translating or adapting. There can be several versions of the same app, and users can resume or edit a previous translation or adaptation at any time.

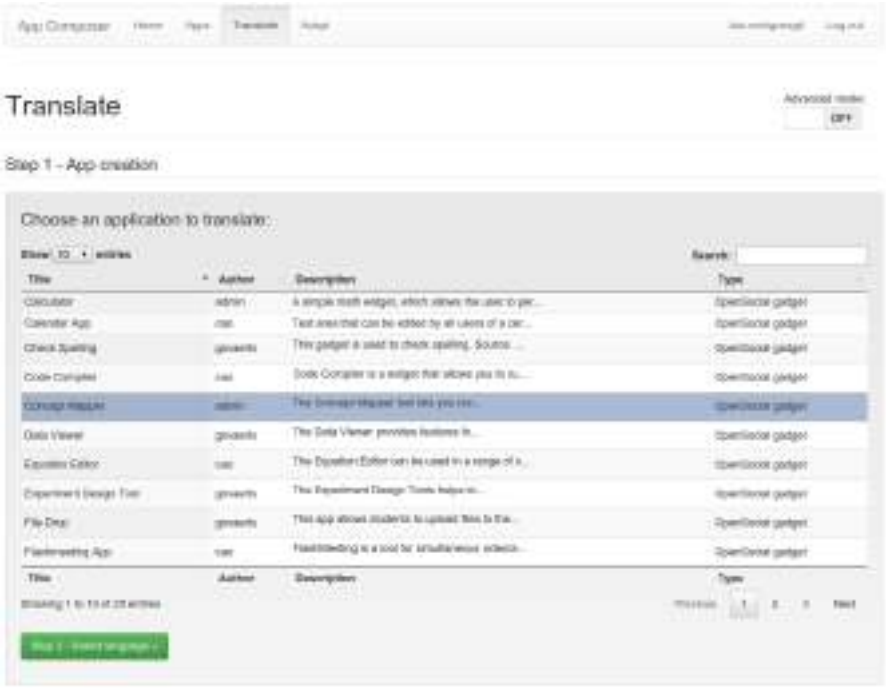

Figure 30. Translator's App Selection screen

Figure 31. User's applications list

4.3.8 Sharing apps

The app can be developed by a group of teachers using basic version control. For a given application, identified by its URL, and for a given language, there is an *owner* of that application and language, who acts as an editor. Other teachers can provide their own translation and automatically propose the *owner* to merge their suggestions. The *owner* can choose to accept proposals automatically, or to review them manually and modify them freely before merging them. The proposal merging user interface is depicted in Figure 32. Through this UI, *owners* of the languages can review the proposals they receive and see each proposed change. They can immediately observe which messages are changed (those whose background is in red) and which are not (those whose background is in green). They can also deny the whole proposal, or deny specific changes (by clicking on the dash or cross, on the right of a message), or even edit the text of each message.

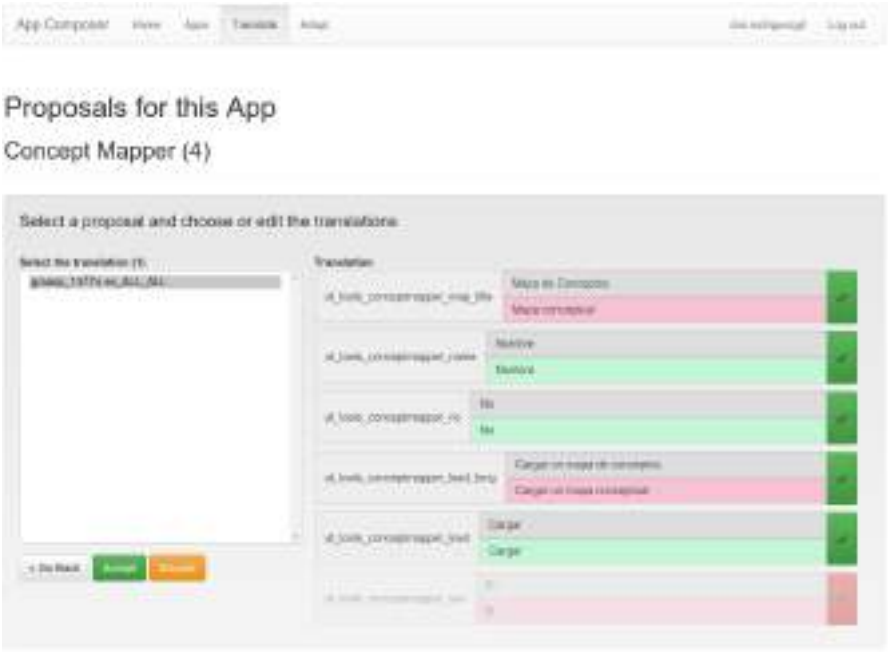

Figure 32. Proposal Merging UI

4.3.9 Authentication

To enable saving apps and version control, a user identity is needed. The portal can provide this, so that users do not have to sign up for an account. Though the App Composer supports standard login/password authentication most users will actually authenticate through the ILS Platform. Currently, this *single sign-on* scheme is loosely based on OAuth, but relies on a different, token-based, ILS platform-specific system. In the future, the ILS Platform will be an OAuth2 provider and the App Composer will properly integrate with the OAuth2 authentication mechanism.

5 Conclusion and outlook

In this deliverable we have presented the implementation work completed for the initial release of the Go-Lab portal and the app composer. The requirements for this initial release were specified in D5.2. We have described the features of the Go-Lab portal and the app composer based on these requirements. However, the main result of this deliverable is the software itself, which is available at `http://golabz.eu`, `http://graasp.eu` and `http://composer.golabz.eu`.

Figure 33 summarises the requirement fit of this initial release and the future refinements that are planned for the final release these components in D5.6 (M36).

More specifically, we plan to release basic recommendations (as requested by the reviewers) for labs, apps and ILS on the Lab Repository and the ILS platform. A first prototype of people recommendations has been integrated in the Lab Repository based on the Bartering Platform profiles (see D4.4). Furthermore we plan to continue our work with WP1 to refine ILS from a pedagogical perspective and investigate how different learning scenarios can be supported by the Go-Lab portal as well as our participatory design approach to creating new guidance apps to populate the Lab Repository. Our collaboration with WP2 will result in new labs added to the Lab Repository and refinement of the implemented metadata schemes on the Lab Repository. We will also continue to take into account the findings of WP3 on the evaluation of the different platforms described in this deliverable. This will allow to improve the adoption of Go-Lab, which is a central aspect of the project. During next year we will also further integrate the WP4 results in terms of learning analytics to provide more fine-tuned guidance apps, furthermore their tasks on the Smart Device and the Smart Gateway will allow to further make the Lab Repository accessible to lab providers. Some features of the Lab Repository will be further extended such as the publishing of ILS and labs, as well as advanced user management and authorization.

The final version of the Portal and App Composer will be documented in Deliverable D5.6 (M36) and afterwards in M48 the sustainable version of the Portal and App Composer will be released (see D5.7).

References

Benedek, J., & Miner, T. (2002). Measuring desirability: New methods for evaluating desirability in a usability lab setting. *Proceedings of Usability Professionals Association*, 8–12.

Constantine, L. L., & Lockwood, L. A. (1999). *Software for use: a practical guide to the models and methods of usage-centered design*. Pearson Education.

Dahrendorf, D., Dikke, D., & Faltin, N. (2012). Sharing personal learning environments for widget based systems using a widget marketplace. In *Proceedings of the ple conference 2012 in aveiro, portugal, july 11-13, 2012. aveiro, portugal.*

Morville, P. (2005). *Ambient findability: What we find changes who we become*.

O'Reilly Media, Inc.

Nielsen, J. (2012). *Usability 101: Introduction to usability.* http://www.nngroup.com/articles/usability-101-introduction-to-usability/, last accessed: 20 Oct.2014.

Richter, T., Grube, P., & Zutin, D. (2012). A standardized metadata set for annotation of virtual and remote laboratories. In *Proceedings of the 2012 ieee international symposium on multimedia* (pp. 451–456). Washington, DC, USA: IEEE Computer Society. Retrieved from `http://dx.doi.org/10.1109/ISM.2012.92` doi: 10.1109/ISM.2012.92

Requirements (from D5.2)	How the requirements are met	Next Step
Go-Lab Portal		
Publishing labs	Labs can be added on the Lab Repository by Go-Lab partners	Ongoing work with WP2 to select and add new labs to the Lab Repository
Creating ILS	ILS copies and new ones using a specific lab can be created on the Lab Repository and new ones can be created on the ILS Platform	Based on WP3 feedback this implementation can be refined
Modifying ILS	ILS can be modified in the ILS Platform. E.g., phases can be reordered, renamed, items can be added and removed	Based on WP3 feedback this implementation can be refined
Publishing ILS	ILS can be published on the portal from the ILS portal	Based on WP3 feedback this implementation can be refined
Using ILS	Teachers can share the Standalone View of an ILS through a secret URL in the ILS Platform	Based on WP3 feedback this implementation can be refined
Supporting apps	The ILS Platform supports Open Social apps. The portal stores a catalog of apps	Ongoing work in WP5 to build new apps
Supporting learning scenario	ILS embody learning scenarios. They are a central component of the Lab Repository and the ILS Platform	Ongoing work on the pedagogical evaluation of the ILS on the Lab Repository with WP1
Searching Labs & activities	The portal provides several search mechanisms for apps, labs and ILS	Ongoing work on metadata needed by teachers with WP2
User management	A single sign-on prototype has been implemented. Students do not need a log in	Single sign-on will be deployed in the coming months
Social features	The Lab Repository and the ILS Platform provide sharing, rating, and discussion options.	Based on WP3 feedback this implementation can be refined
Tracking user activities	The Lab Repository tracks visits using Piwik and ILS Platform records Activity Streams	Further dashboard apps are being developed to visualise activity
Recommendations	There are no recommendations yet in the Go-Lab Portal	A recommender system for the ILS Platform and the Lab Repository is being developed
Ubiquitous access	Go-Lab portal components are accessible on any modern browser and are responsive to mobile devices	Mobile device responsiveness will be improved for the ILS platform
Usability & privacy	New user centered design, the Standalone View does only require nicknames and tracking is only enabled in ILSs when the AngeLA tracking agent is present	Ongoing work on participatory design and evaluation with WP3
Scalability & availability	The ILS Platform uses availability monitoring. Last year the uptime of the ILS Platform was 99.6%.	Depending on the need, we might investigate more scalable architectures and deployments
App Composer		
Different Languages	The translator allows to translate apps in any language	Ongoing work on handling non-standard or invalid apps, and on improving user experience in collaboration with WP3
Different Target Groups	The translator allows to target different age groups for each translation	Depending on the needs, the groups may be redefined
Adaptable app listing	The Adaptor lets users choose from a list of known adaptable apps	The list is currently fixed. Eventually, the list may be retrieved from the Lab Repository
Adapting apps	The Adaptor lets users configure adaptable apps to their needs	Based on WP3 feedback this implementation can be refined
Publishing apps	Both the Translator and the Adaptor let users publish apps, either on the ILS Plarform	Based on WP3 feedback this implementation can be refined
Portal integration	The portal integration is not yet completed	Ongoing work
Draft support	Users can save and load their progress before publishing, while adapting or translating an app	Based on WP3 feedback this implementation can be refined
Sharing apps	Translations can be shared, and published apps as well	Based on WP3 feedback this implementation can be refined
Authentication	A preliminary authentication mechanism is implemented using the ILS Platform	Integration with the ILS Platform with OAuth2 is underway

Figure 33. A summary of the requirement fit and future work