FORGE
Forging Online Education through FIRE

Project acronym: FORGE
Project full title: "Forging Online Education through FIRE"
Grant agreement no: 610889

D4.1.2 FORGE architecture: Introducing FIRE to the learning community (Update)

Deliverable Editor: **Christos Tranoris (UoP)**

Other Contributors: **Kostas Bakoulias (UoP), Kostas Lampropoulos(UoP), Alexander Mikroyannidis (OU), Daan Pareit (iMinds), Ingrid Moerman (iMinds), Pieter Becue (iMinds), Johann M. Marquez Barja (TCD), Valina Katsaounou (GRNET)**

Reviewers: **Olivier Fourmaux (UPMC)**
Guillaume Jourjon (NICTA)

Document Identifier:	FORGE-2015-P-D412	Due Date:	31/07/2015
Version:	1.0	Submission date:	31/07/2015
State:	Final	Distribution level:	Public

This document is part of a research project funded by FP7 Framework Programme. Grant agreement number 610889.

Change Log

Version	Date	Amended by	Changes
0.1	01/06/2015	Christos Tranoris	First ToC and responsibilities
0.2	06/07/2015	Christos Tranoris	First draft available
0.3	21/07/2015	Christos Tranoris	Accepting reviewer 1 recommendations
0.4	22/07/2015	Christos Tranoris	Accepting reviewer 2 recommendations
0.5	27/07/2015	Christos Tranoris	Final editing
1.0	31/07/2015	Aneta Tumilowicz	Final QA

Table of Contents

List of Figures

1. Executive Summary

This deliverable contains some architectural updates to D4.1.1 (FORGE architecture: Introducing FIRE to the learning community) containing the latest refinements, including some experience from the implementation and execution of the prototype lab courses, plus some experience from the opening of the platform. We revisit again the FORGE architecture and all the related entities of widgets and FIRE Adapters.

We present how we adopted new technologies in order to embrace effectively the educational domain. Interoperability with existing learning environments is addressed by adopting the well supported LTI API and SCORM. Learner's behaviour which is addressed by Learning Analytics, is now enabled by the Experience API. In the document we present how we use these technologies in our assets, namely FORGEBox and widgets.

Finally, we have included generic widget architecture, which is the result of our experience through FORGE while we implemented several widgets. We expect this architecture to guide widget developers that target FIRE facilities for education. Still, we tried to abstract the requirements so that it can be applied to any facility that wants to offer remote resources for learning.

2. Introduction

The overall architectural approach of FORGE was presented in Deliverable D 4.1.1, which described the FORGEBox approach but also documented the initial user roles and requirements. Nevertheless, after the implementation of our prototype elements, that are widgets, FIRE Adapters and FIRE Courses, a refinement and extension of our initial proposal is necessary. This deliverable reflects several refinements and ideas made to FORGE concepts during development and deployment of courses. All these architectural refinements are towards accomplishing our initial FORGE challenges that for clarity are also mentioned here:

I. To make the reservation of resources in (different) facilities easy for both teachers and learners;
II. To allow easy fast experimentation control, from various devices and means, during the learning process;
III. To know the identity of the user who is currently performing an experiment that is initiated from within a client web browser;
IV. To access resources that can only be reached over IPv6 or over a VPN;
V. Avoid breaking the logical flow of an educational experiment when the user behaved unexpectedly;
VI. To allow multiple users sharing the same experiment;
VII. Handle a large number of simultaneous users.

Equally important, though, is to use the existing offered eLearning technology and try to seamlessly integrate it with our FORGE efforts. Thus, all further developments made in our core entities, considered open and well known eLearning technologies. We investigated solutions of exploiting those eLearning technologies in two areas: interoperability and means to study user behaviour during learning on top of FIRE. These technologies are the Learning Tools Interoperability (LTI) standard, SCORM and the Experience API xAPI or most known as TinCanAPI.

LTI adoption will provide better integration between FORGE technology and existing Learning Management Systems (LMSs) and Virtual Learning Environments (VLEs). LTI will provide a seamless experience of learners while interacting both with the LMS/VLE content and the remote FIRE resource. That is, LTI will make much easier for organizations to adopt and use the FORGE technologies and integrate them with their own already deployed learning systems. xAPI on the other hand will allow Instructors to study learners' behaviour during interacting with a facility. Next paragraphs present how these technologies are exploited within FORGE. Our core proposal is the reference architecture of widgets and FIRE Adapters while interacting with remote facilities of FIRE through a VLE. We provide also a reference architecture that can contain all these technologies, which is prototyped as implementations in FORGEBox and FORGEStore.

3. FORGEBox reference architecture update

3.1 Overview

Our approach of a middleware solution called FORGEBox was presented extensively in D4.1.1 (FORGE architecture: Introducing FIRE to the learning community), which also presented a detailed description of supported user roles and services. After the first course deployments, several issues have been identified in the widgets and FIRE adapters implementation. Moreover, the initial demand for interoperability between existing eLearning technologies and FORGE offerings had to be also satisfied. Therefore we have refactored our initial architectural approach to reflect these new ideas, as next figure shows.

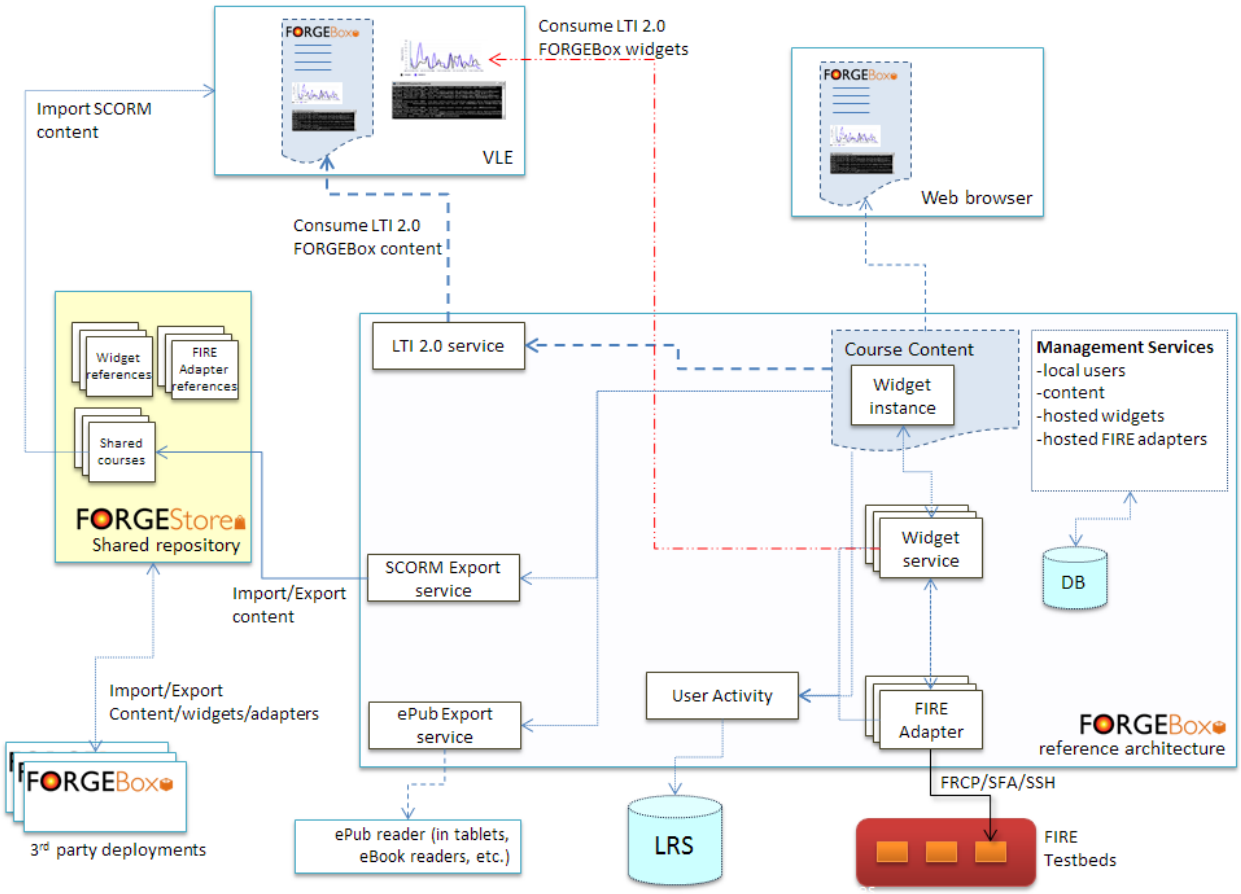

Figure 1: Refactored our FORGE architectural approach

FORGEBox is our reference architecture for the FORGE middleware technology, capable not only to mediate between eLearning tools and FIRE facilities, but also to facilitate several use cases. At a minimum, a main usage of a FORGEBox is to host widgets and FIRE Adapters, acting like a gateway, thus making it capable to offer courses on top of FIRE.

A step further is to offer some minimum functionality of creating course content by easily integrating it with interactive elements like widgets. Many existing solutions can offer content, from a simple Drupal installation to a full LMS like Moodle. The Web first approach followed by FORGE, allows publishing to virtually any platform: Modern web browsers, any LMS, ePub3, Apple iBooks.

Still a FORGEBox requirement is to integrate the content with widgets and be interoperable with existing technologies. Thus to support the above, two known eLearning technologies are recommended: SCORM and LTI2.0. Next sections will provide details how these technologies are used.

Another consideration for FORGEBox was the adoption of Learning Analytics especially while interacting with resources from FIRE facilities. Learning Analytics aims to collect and analyse user activities to make learning more effective and efficient. FORGE considers Learning Analytics services in order to provide means to track user activities and analyse this tracked data. This provides the foundation for guidance mechanisms for students, as well as intelligent decision support for teachers and lab owners. Next sections will also provide more details regarding the adoption of Learning Analytics in FORGEBox.

3.2 The role of a shared repository: FORGEStore

D4.1.1 (FORGE architecture: Introducing FIRE to the learning community) presented the idea of FORGEBox as a middleware that can be deployed in multiple organizations. On the other hand, FORGEStore instantiates the idea of a common shared repository of lab courses, widgets and FIREAdapters.

With FORGEStore, users will be able to:

- Search for lab courses and download them as a SCORM package;
- Upload lab courses as SCORM packages. FORGEBox users will be able to do this automatically;
- Search for available widgets that they can refer their web endpoints in their courses;
- Search for FIRE Adapters and download their binaries.

Developers of remote lab artefacts, widgets and FIRE Adapter facilities will be able to register and upload or register their services.

A first FORGEStore instantiation is deployed under www.forgestore.eu. Details of its architecture and implementation will be provided in D3.2 (Educational Widget Store).

3.3 Teacher Companion Lab courses Widgets/FIRE Adapters

A requirement within FORGE is to support the instructors when deploying a lab course on top of a FIRE facility. A Teacher's companion lab course is a kind of a special Widget/FIRE adapter combination to assist Teachers on easily managing remote lab FIRE facilities for a course. Usually these companions will be built by experts of FIRE facilities. They will be applied over a specific lab course in order to automate some reservation and provisioning processes of the underlying target infrastructure needed by the learners to perform the course. For example the TCP congestion control lab course has a companion for setting easily the setup on iMinds Virtual Wall, regardless the number of users. A Teacher's companion lab service can provide the following features:

Simple predefined testbed setup

Given a simple user interface, teachers will be able to just set the requirements of the lab (i.e. number of students, time schedule, etc.) and the setup will be provisioned to the FIRE testbed.

Hide some complexity from FIRE tools

Since these companions will be built by FIRE experts, they can hide all the complexity of setting up specific experiments and exposing only some necessary information just to quickly set up all the needed resources. Usually a FIRE testbed can be set up by tools like jFed, MySlice, or through RSpec specifications. These can be hidden and pre-defined and a FIRE adapter will wrap all the functionality. The only things to expose will be reservation schedule and how many resources needed to be reserved for the course.

Simple reservation scenarios

Resource reservation mechanisms are crucial for a FIRE remote facility because of the limited physical resources. Besides that, teachers want to be able to reserve a FIRE remote facility for a certain period during their class. FIRE tools like MySlice offer aggregated views of resources across testbeds so potentially this functionality could be reused by these companions.

Repeatability of deployment

Having these companions, teachers are ensured that the same setup can be safely repeated for many lab courses in future.

Portability of deployment

Having FIRE as the basis of making lab courses, the APIs to manage and control resources are the same across different testbeds. Thus a certain setup described by a companion lab recipe could be repeated in a similar FIRE testbed.

Enable scalability

Similar as above, the common APIs of FIRE allow us to smoothly scale the resources if there is an increased demand. Given the federated nature of FIRE, if the resources of a testbed are not enough for a lab course, they could be complemented with resources from another testbed offering similar technology.

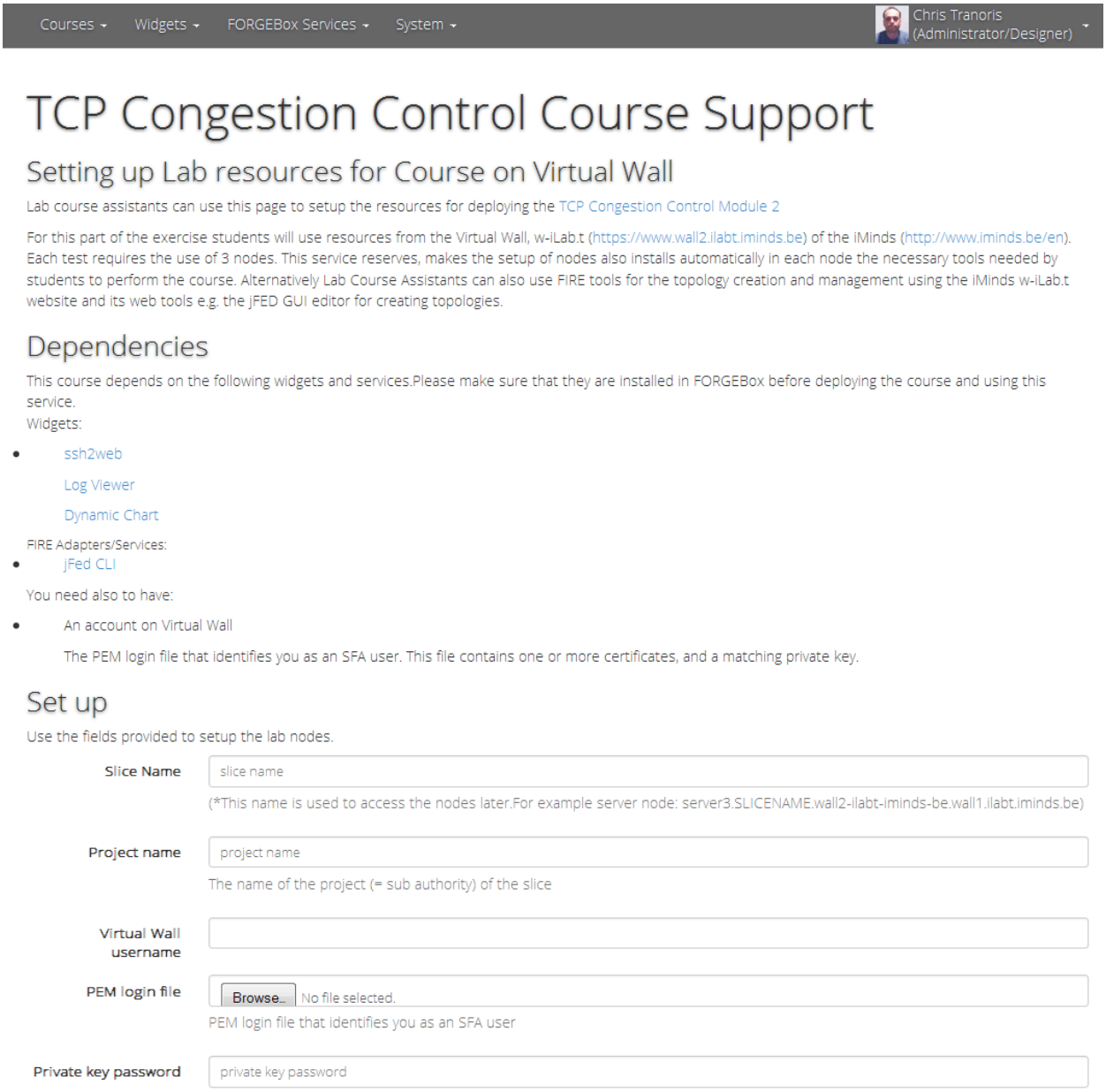

Figure 2: The TCP Congestion Control Course Setup Companion

Prototypes of such companions for FIRE labs are the TCP Congestion Control Course Setup (as figure 2 shows), which is used to recreate the experimentation environment needed to execute the course for congestion control on the Virtual Wall testbed. Its prototype implementation is presented in D2.1 (Adaptations made to the FIRE APIs to support education in prototype lab course). Also iMinds developed a w-iLab.t testbed lab course companion service, allowing a teacher to reserve and provision multiple non-interfering wireless instances on the w-iLab.t testbed at a chosen time slot without technical knowledge of the underlying tools and mechanism that is presented in more detail in D4.2.1 (Methodology and process for creating FIRE courseware).

3.4 LRMI

The Learning Resource Metadata Initiative (LRMI[1]) was established in 2011 and is currently led by Creative Commons and the Association of Educational Publishers, representing both open education and commercial publishers of learning resources. Its purpose is to help educators and learners to find learning resources via major search engines and specialised resource discovery services. The adopted approach has been to extend the schema.org ontology so that educationally significant characteristics and relationships can be expressed.

The most important properties used by LRMI to describe educational characteristics and relationships are the following[2]:

- **Educational alignment**: an alignment to an established educational framework. This property can be used to specify that the resource addresses some part of a shared curriculum or competence framework, or that it can be classified against some other educational scheme such as one describing educational levels;
- **Educational use**: a text description of the educational purpose of the resource, for example ‘assignment’ or ‘group work’;
- **Interactivity type**: the predominant mode of learning supported by the learning resource. Acceptable values are ‘active’, ‘expositive’ or ‘mixed’;
- **Based on url**: a link to a resource that was used in the creation of the resource that is being described. This may be useful if a learning resource is a derivative of another resource;
- **Learning resource type**: a text description of the predominant type or kind characterizing the learning resource. For example, ‘presentation’, or ‘hand-out’;
- **Time required**: the approximate or typical time it takes to work with or through this learning resource for the typical intended target audience;
- **Typical age range**: the typical range of ages of the content’s intended end user.

LRMI is adopted within FORGE as the mean to adopt an abstract vocabulary representing the most common descriptions of learning resources used by existing educational metadata standards (e.g., Learning Object Metadata). Thus FORGEBox architecture promotes the usage of LRMI metadata within each course content. The prototype implementation of FORGEBox, at www.forgebox.eu, offers courses that are annotated automatically with LRMI metadata. This metadata are also included when exporting the course content as SCORM, thus ready to be used by an LMS.

[1] http://www.lrmi.net

[2] Barker, P., & Campbell, L. M. (2014). Learning Resource Metadata Initiative: using schema. org to describe open educational resources. *Proceedings of OpenCourseWare Consortium Global*.

3.5 Scheduling of resources

One of the most important challenges for interactive experimentation on FIRE facilities is the availability and scheduling of resources to enable experimentation. The most important trade-off in this problem is the scarcity of resources versus the peak audience requesting access to these resources.

We also consider the difference between planned in-classroom exercises and ad-hoc remote experimentation from an eBook, LMS or website. In the former case, the expected load on the FIRE facility is known beforehand and availability of resources can be guaranteed and planned. In the latter case, the required FIRE resources are not known beforehand and a mechanism should be in place to handle unavailable resources, we call this graceful degradation.

Guaranteed availability of resources

Every FIRE facility has its own policies concerning resource usage and scheduling. Facilities can feature user quota, best effort or guaranteed resource availability, and different types of resource reservation mechanisms.

In general, the scarcer the FIRE resources offered by a facility are, the more restrictions are in place to use these resources. These can be time restrictions or mandatory reservations. On the other hand, when a FIRE facility offers a large amount of homogeneous resources, they more often operate on a “best effort” or “gentleman’s agreement” policy.

Good examples of this distinction are the iMinds iLab.t facilities. The w-iLab.t wireless testbed offers 60 wireless nodes in a specific topology, together with spectrum sensing equipment and Software Defined Radios. W-iLab.t features a mandatory reservation system where specific resources need to be selected in timeslots of maximum 8 hours. The Virtual Wall on the other hand features 300+ practically identical machines and has currently a 720 hours maximum usage duration as its only restriction.

When considering education on FIRE facilities, especially during organized classroom lab sessions featuring many students, guaranteed availability of resources is an essential property. In these cases a FIRE facility featuring a mandatory reservation system is preferred, as long as there are enough resources available in total with the required properties.

When deploying large-scale experiments on FIRE facilities that only provide best effort live provisioning, a fall-back system must be in place in case there are not enough resources available. In this case, alternative FIRE facilities with similar properties should be used to provision the remaining resources, but in the end resource availability cannot be guaranteed. Therefore, it is important to optimize the use of the resources that are available; so multiple learners can share the same resource if needed.

Thus, it is advised to Lab Course Designers and instructors to plan for resource usage and use any available lab setup recipe that the lab course may provide. However if there is a large demand of resources that a single tested cannot satisfy then resources from multiple testbeds can be used. The FIRE APIs can handle these cases. Though, in other scenarios involving specific resources like IoTs or Smartcity related testbeds, due to current FIRE’s nature a separate negotiation with testbed providers should take place.

Maximizing resource usage

When trying to maximize resource usage, we consider two approaches to this problem: educational and technical. Although both approaches can be applied separately, aligning both approaches brings the highest optimization promise.

Educational approach

When designing the educational content of an interactive lab session, there is almost always the question of dedicated resource usage: the time a student needs exclusive access to the experimentation machine to perform an exercise without interference.

When designing a series of exercises, it is also important to consider if each exercise has a well-defined a priori and a posteriori state, or that the outcome of one experiment should influence another experiment. Such chains of correlated exercises require continuous exclusive access by the learner and should be kept as short as possible, if feasible, to allow the sharing of one set of resources by multiple learners.

Of course, when multiple students share the same resource, technical considerations arise to guarantee the correct state of the resources and non-interfering use by the learners.

Technical approach

When multiple learners request access to the same resources, the need arises to queue these requests so that exclusive and non-interfering access is guaranteed. Depending on how a course is designed, we envision multiple forms of queuing:

- Live execution, experiment level
  Multiple learners have access to the same resource and experimentation requests are queued and executed when the previous learner finishes an exercise. This approach is only feasible if the waiting time for the learner remains short, so it is preferably combined with short bite-sized exercises and a limited amount of simultaneously active users.
- Live execution, course level
  One learner gets exclusive access to the resources for the duration of an entire session of exercises. Once the learner finishes the resources are released, reset and available for the next student. This approach is preferably combined with predefined timeslots per user so learners can plan around the availability of the resources.
- Delayed execution
  Learners perform their experiments through a shared interface, but the execution of the actual experiment is not guaranteed within a certain timeframe. All experiments are queued by a background service and learners can check back later to collect the results. This type of queuing is preferably combined with remote learning and single shot exercises, so multiple experiments by the same learner can be queued without having to wait for the intermediate result.

It is also possible to reduce the load of users on certain resources with a load balancer that redirects incoming experimentation requests based on the availability of resources and their current load. This can be performed on a per experiment basis, where each time a learner performs an experiment the resources with the lowest load are selected. Or the selection can occur

on the course level, where a new learner is redirected to the resources with the lowest load when the course is accessed for the first time. This latter approach is the most straightforward since this is a regular use case of load balancers, while load balancing each request requires a custom load balancing algorithm and it is nontrivial to maintain a session state throughout multiple requests.

3.6 Single Sign-On

Another important feature of the FORGEBox reference architecture is the support for Single Sign-on scenarios. This means that a FORGEBox could be part of a federation or could just support solutions like OAuth 2.0 with other organizations. SSO will enable a seamless experience of users, when they would like to use FORGEBox services with a single account. Widgets also will greatly benefit from adopting SSO, although the LTI 2.0 approach (discussed later) is much more important and offers same capabilities of user identification.

A first prototype implemented at our FORGEBox instance at forgebox.eu, is the implementation of the GRNET federation API. This allows to all Greek students to sign in and use FIRE lab courses with their institutional accounts. Since many organizations managing their LMSs/VLEs are probably already members of a federation, FORGE widgets will be able to identify the end user through LTI.

3.7 Experience from the opening of the FORGE platform

FORGE has announced the opening of its platform and tools to external contributors in April 2015. Four proposals for the development of online interactive courses have been submitted so far and are currently under the initial stages of design and implementation of their work. These proposals cover a rather wide range of themes and are expected to introduce different types of requirements to the project and its architecture, as presented hereafter.

Proposal 1 “Metrology of the Internet”, by INRIA, France: The proposed course covers the measurements that can be conducted in local area networks, access networks, and transit networks in order to monitor network performance. These measurements will be performed through lab exercises deployed on the large-scale network testbeds provided by FIRE+, while making use of several FORGE tools. The basic feature that is expected to bring possible modifications to the FORGE architecture is that this course is planned to be offered as a real-time Massive Open Online Course (MOOC) in English via the online platform France Université Numérique (https://www.france-universite-numerique-mooc.fr/). Therefore, this will be a real use case where the transformation of FIRE tools to support the learning future of online and massive courses could be studied.

Proposal 2 “Project-based learning for master level students” by the Universitat Politècnica de València, Spain: This course presents a problem-solving approach on the “Networks and security” theme, within the context of a real-time master degree course. The FORGE toolbox will be used as a clear framework for accessing the FIRE experimental facilities that are required to deploy the course material. It is expected that the implementation of this course will open up a new path for FORGE, which is the introduction of the Internet of Things and Smart Cities concepts.

Proposal 3 "FORGE-based Local Area Networks", by Universidade de Brasília, Brasil: The goal of this course is to teach protocols, algorithms, and mechanisms behind current standards of local area networks, both wired and wireless ones. The proposal mainly deals with the deployment of already available FORGE courses; therefore not many new requirements are expected to be brought into FORGE by this proposal. However, this course is planned to be integrated into the university's Moodle platform, so some requirements may come up regarding interoperability.

Proposal 4 "GÉANT Testbeds Service – User Certification Programme", by the GÉANT Association and the Friedrich-Alexander University of Erlangen-Nuremberg: The proposed course aims at presenting the GÉANT Testbeds Service (GTS, http://services.geant.net/GTS/Pages/Home.aspx) and its innovative design principles to users combined with a light-weight certification programme. This course is expected to benefit from the Learning Analytics FORGE architecture and may bring in some new requirements. In addition, this is a non-FIRE testbed, which is expected to join FIRE soon.

More information on the requirements that will be brought into the project by the external lab courses will be provided within the next few months, when the proposals will have progressed with their foreseen work.

4. Interoperability with LMSs and VLEs

Integrating FORGE technology with tools that organizations are using for deploying their courses, such as advanced Virtual Learning Environments (VLEs), will increase FORGE's impact. Therefore within FORGE we need to consider: i) how FORGE content can be easily consumed by VLEs, ii) how widgets/FIRE Adapters can seamlessly exchange user information with the VLE and be integrated with the content provided by a VLE. The technologies considered for interoperability with VLEs are:

- The Learning Tools Interoperability (API) and its latest version 2.0;
- SCORM packaging.

The Learning Tools Interoperability (LTI[3]) is a specification that standardizes the APIs between LMSs and external tools, enabling external tools to function as if they were native tools inside the LMS. LTI has been developed by the IMS Global Learning Consortium, a non-profit member organisation with a mission to enable the growth and impact of learning technology in the education and corporate learning sectors worldwide. The IMS Global Learning Consortium develops open interoperability standards and supports their adoption with technical services and programs that highlight effective practices.

The LTI specifies a standard way of integrating learning applications with platforms like LMSs, portals, or other educational environments. As shown in Figure 3, the primary use case behind the development of the LTI specification is to allow the seamless integration of web-based, externally hosted applications and content with other platforms. For example, an interactive assessment tool or a virtual lab hosted by a LMS can be securely connected to another educational platform in a standard way without having to develop and maintain custom integrations for each platform.

[3] http://www.imsglobal.org/lti/index.html

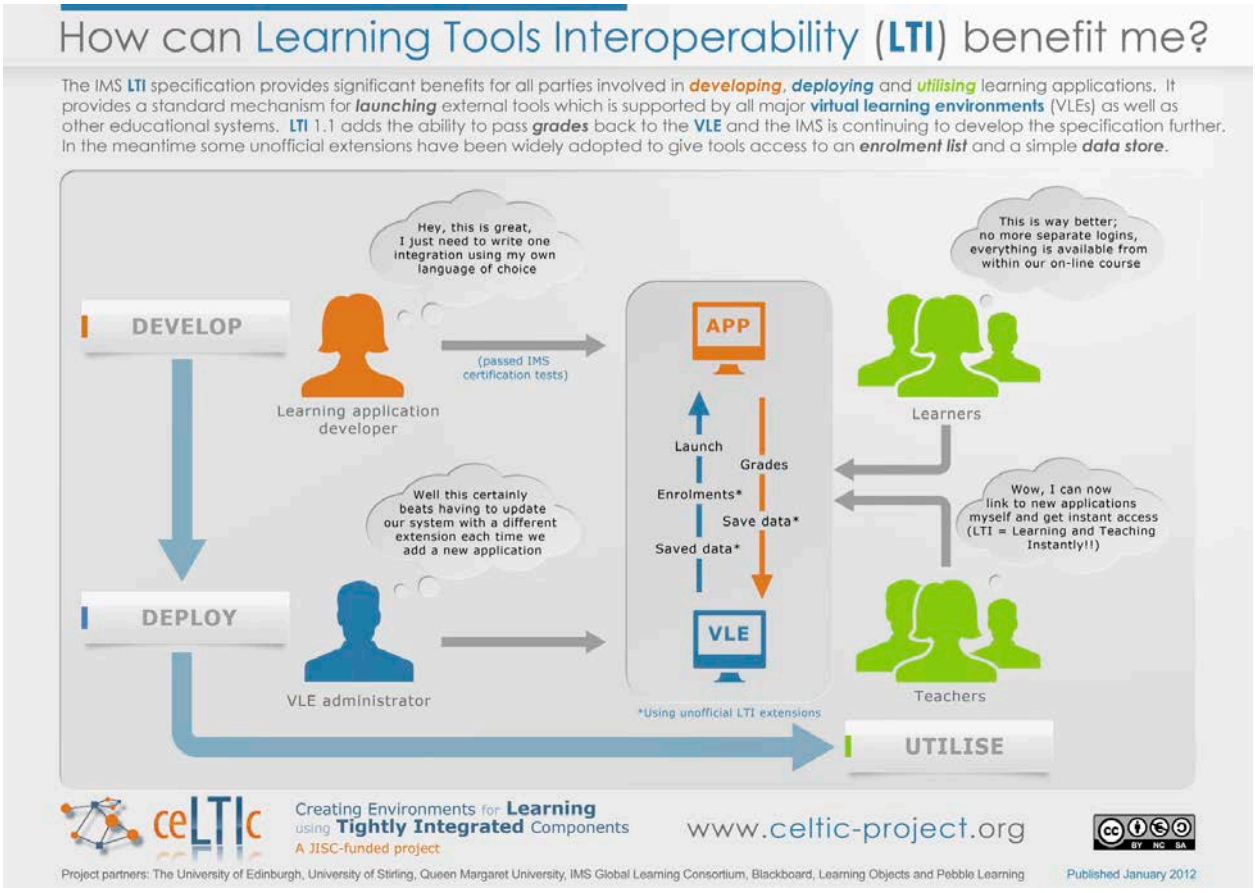

Figure 3: A usage scenario of the IMS LTI specification[4]

LTI's primary usage within FORGE exists on widgets and FORGEBox created content in order to integrate tools like FORGE widgets that enable the interactivity with FIRE remote testbed resources. LTI will allow FORGE widgets be securely connected and integrated to an educational platform in a standard way without having to develop and maintain custom integrations for each platform. In LTI these learning applications are called Tools (delivered by Tool Providers) and the VLEs are called Tool Consumers.

SCORM on the other hand will help FORGE content to be easily consumed by systems able to host SCORM content. SCORM stands for "Sharable Content Object Reference Model" and is all about creating units of online training material that can be shared across systems. SCORM defines how to create "sharable content objects" or "SCOs" that can be reused in different systems and contexts.

The main objectives of the SCORM standard are the easy portability of learning content from one LMS to another, as well as the reusability of learning objects. The metadata model of the LOM standard integrated in the SCORM supports the retrieval of learning objects. SCORM denominates the smallest unit that can be administered by an LMS as a SCO, which represents the lowest level of content granularity that can be tracked by an LMS. A SCO is independent of any learning

[4] http://www.imsglobal.org/images/celticposter.pdf

context in order to be reusable for different learning purposes. SCOs can form learning or exercise units on a superordinate level[5].

SCORM is based on the following three components[6]:

- **Content packaging**: It refers to the packaging of all resources needed to deliver a course into a single zip file. The format for this file is described by the SCORM aggregate model, which is based upon the IMS Content Packaging Specification. The zip file contains not only the course files, but also an XML file, referred to as the manifest file, describing the course contents and content sequencing;
- **Runtime communications**: These are conducted using two elements:
  - Runtime commands for communicating student information to and from the LMS;
  - Student metadata for storing information on individual students.
- **Course metadata**: These are metadata packaged with a course when it is archived in a SCORM repository. These metadata enable a course author or student to search a learning repository and to identify the learning content they want to use or view. For example, the course title, description, keywords, etc. are all considered course metadata.

The following sections discuss briefly how these technologies are used in FORGE.

4.1 FORGEBox as an LTI Tool Provider

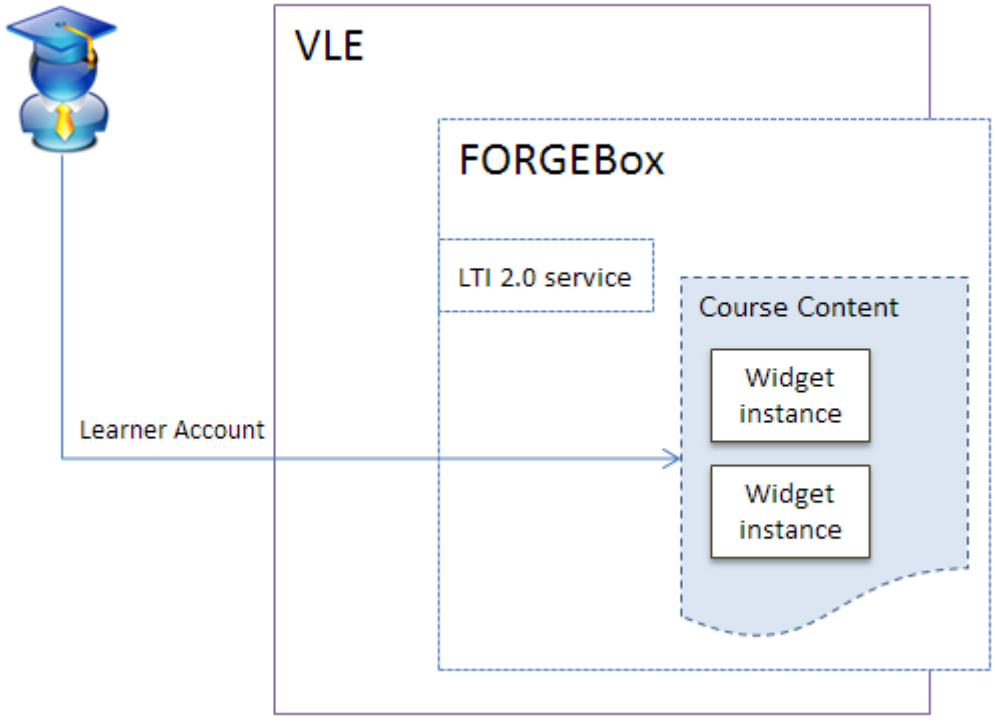

Figure 4: A VLE user access to a FORGEBox course through LTI

[5] Dodds, P. (2001). *Advanced* Distributed Learning Sharable Content Object Reference Model Version 1.2, The SCORM Content Aggregation Model

[6] Jones, E. R. (2002). Implications of SCORM™ and emerging e-learning standards on engineering education. In *Proceedings of the 2002 ASEE Gulf-Southwest Annual Conference* (pp. 20-22).

The first usage of LTI is to support interoperability scenarios in order to allow LMSs/VLEs to consume course content directly from a FORGEBox instantiation. This is useful in cases like the www.forgebox.eu deployment, where FIRE courses are hosted in a FORGEBox. In this case the whole FORGEBox will be offered as an LTI Tool Provider, thus the VLE will be able to consume the whole course together with its interactive widgets. For such cases, the VLE administrator will need to register the VLE to FORGEBox as a tool consumer and need to select courses that will be offered through the VLE. Any interactive content provided by widgets will function the same as it works when providing the course directly from FORGEBox.

An advantage of this approach is the direct simple consumption of a whole course with interactive elements and no further configuration. A disadvantage is that widgets cannot be directly manipulated by the VLE, but offered as it is through the FORGEBox provided course.

Figure 5, displays how a FORGEBox course is displayed in Moodle. Moodle via LTI is a Tool Consumer and FORGEBox is a Tool Provider. Through LTI user information is exchanged and thus Moodle users are able to access a FORGEBox course.

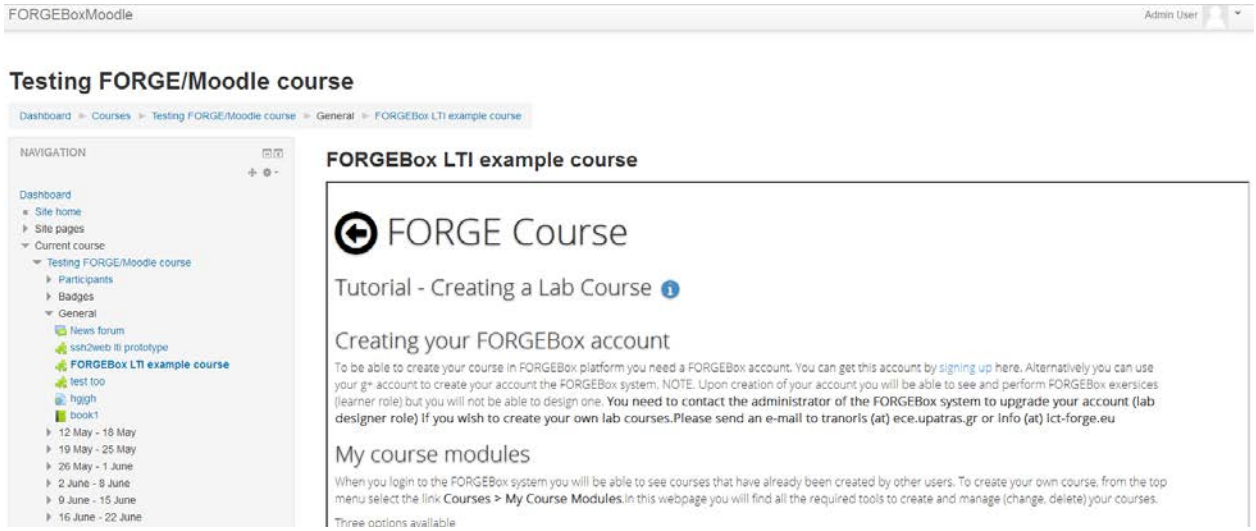

Figure 5: A FORGEBox course consumed by Moodle via LTI 2.0

4.2 Widgets as LTI Tool Providers

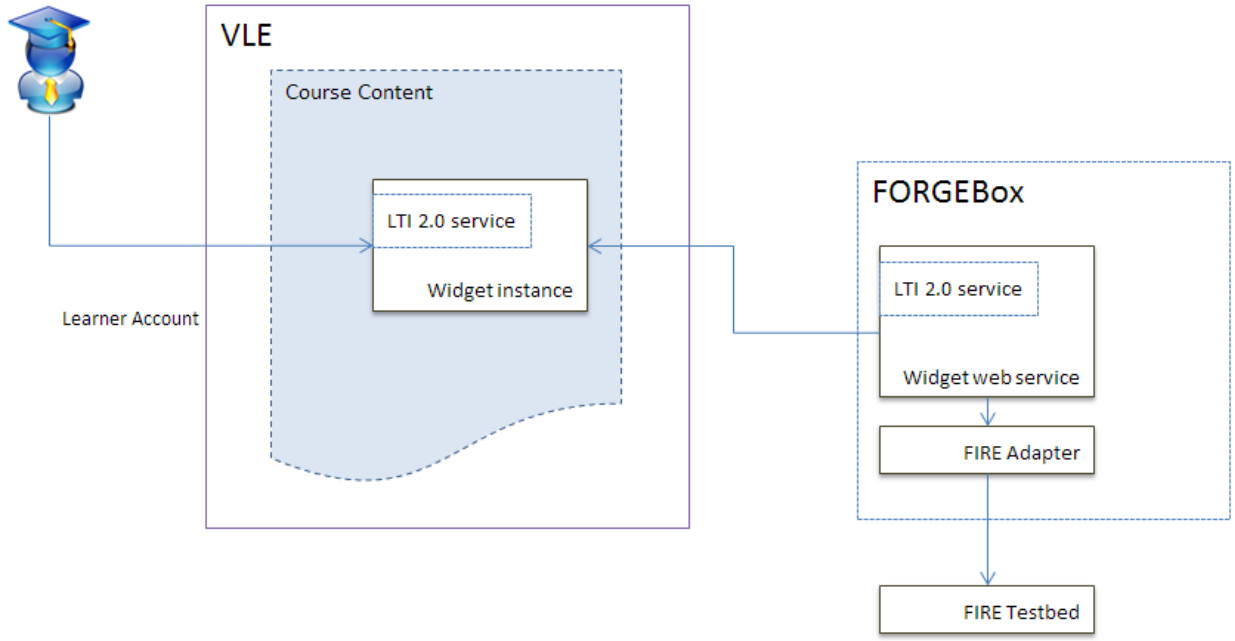

Figure 6: Widgets themselves support the LTI standard

Another way to support interoperability between FORGE technologies and LMSs/VLEs is that widgets themselves support the LTI standard. Although this requires some extra implementation effort by the widget developer, LTI will allow widgets to be securely connected to the LMSs/VLE in a standard way without having to develop custom integrations. A widget therefore will have access to all the features and user information available from the LMS/VLE such as user account, and could also be used to provide feedback about user activity directly back to LMS/VLE.

LMS/VLE Administrators are responsible for registering a widget to their LMS /VLE. When a teacher is creating a course within the LMS /VLE, he needs to integrate and configure the widget with the course content.

An advantage of the widgets supporting LTI is that LTI provides flexibility to lab course designers. A disadvantage is that user experience is sometimes broken by LMSs/VLEs since they put in different web pages each widget and therefore you cannot have many LTI widgets in the same web page. The next example image shows where a widget is directly consumed by Moodle. A problem of Moodle is that it displays LTI tools in separate pages, therefore you cannot have multiple widgets on a single page.

Figure 7: The ssh2web widget is consumed in Moodle via LTI 2.0

4.3 SCORM usage

SCORM usage makes FORGE generated content much more easier to share. The difference here is that the FORGE course content with the widgets will be imported within the LMS/VLE and used by learners from the LMS/VLE directly as they normally use other LMS/VLE content. The problem here is that widgets cannot exchange user information, like LTI, and the administrator of the LMS/VLE need to configure extra information when integrating it with the LMS/VLE. SCORM does not offer the seamless connection of web-based, externally hosted applications and content like LTI does. Nevertheless, an advantage is that SCORM is a simple way to exchange FORGE course content with many learning environments. The disadvantage is that the course is just copied to the new LMS/VLE and it is not easy to pass user information.

5. Pedagogical evaluation framework - Learning Analytics on top of FIRE facilities

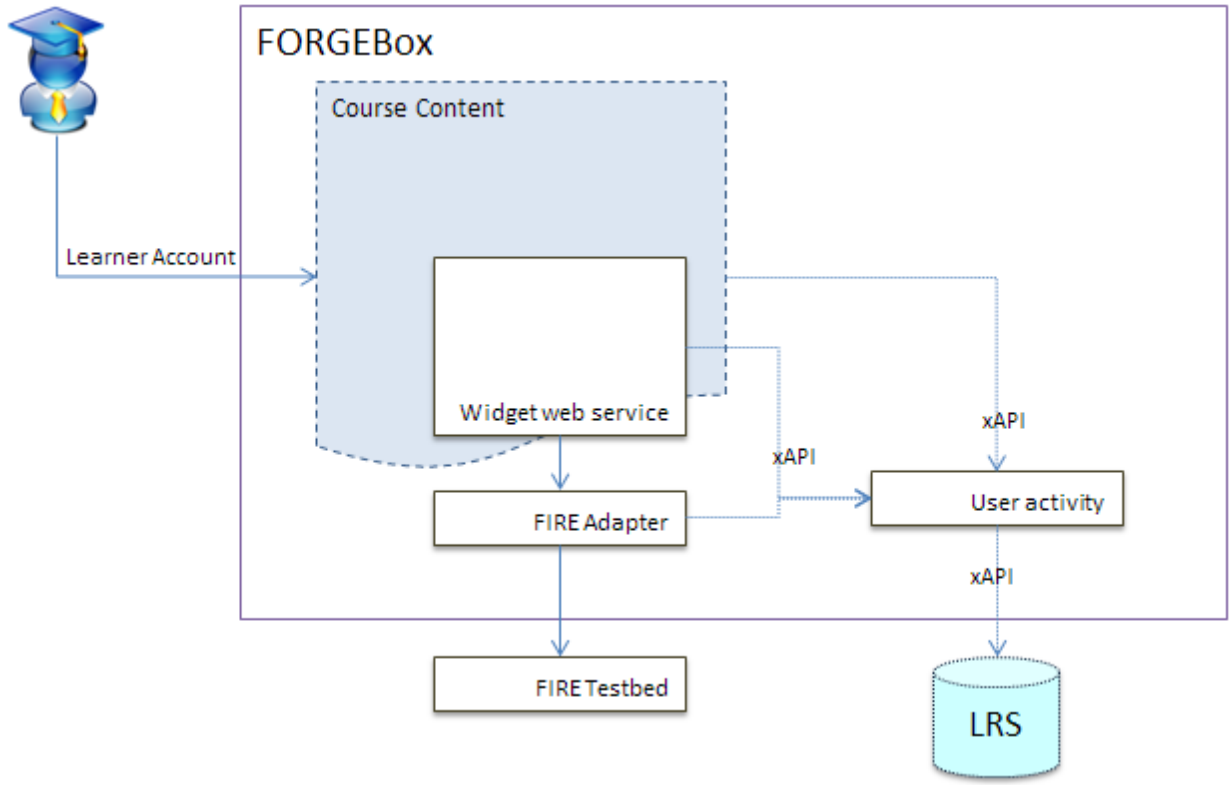

Figure 8: Enabling Learning Analytics in FORGE entities

Learning Analytics is the "measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs[7]". The field of Learning Analytics is essentially a "bricolage field, incorporating methods and techniques from a broad range of feeder fields: social network analysis (SNA), machine learning, statistics, intelligent tutors, learning sciences, and others[8]. Learning Analytics applies techniques from information science, sociology, psychology, statistics, machine learning, and data mining to analyse data collected during education administration and services, teaching, and learning. Learning Analytics creates applications that directly influence educational practice[9].

With Learning Analytics in FORGE it is possible to obtain valuable information about how learners interact with the FORGE courseware, in addition to their own judgments provided via questionnaires. Learning Analytics in FORGE is based on tracking learner activities, which consist

---

[7] 1st International Conference on Learning Analytics and Knowledge – LAK 2011 https://tekri.athabascau.ca/analytics/

[8] Siemens, G. (2014). Supporting and Promoting Learning Analytics Research. Inaugural Issue of the Journal of Learning Analytics. Journal of Learning Analytics 1(1), 1–2.

[9] Buckingham Shum, S. (2012). Learning Analytics. UNESCO Policy Brief. Retrieved from http://iite.unesco.org/pics/publications/en/files/3214711.pdf

of interactions between a subject (learner), an object (FORGE learning activity) and is bounded with a verb (action performed).

To support monitoring and reporting user activity, the FORGE architecture includes components that embrace the usage of Learning Analytics. The idea is to know when and which user experienced some content or interacted with a resource. Within FORGEBox we promote the usage of Experience API (xAPI, also known as the TinCanAPI): a specification for learning technology that makes it possible to collect data. There are many open source implementations around xAPI for many well-known programming languages. xAPI is the FORGE choice to build its Learning Analytics.

While learners experience a course, xAPI generates learning records. A Learning Record Store (LRS) is a data store system that serves as a repository for xAPI learning records. Therefore the presence of an LRS is mandatory in the FORGEBox reference architecture. Usually it is very easy for developers to use an LRS, since it is a simple RESTful endpoint where users that want to track their learners have an account.

5.1 Prototype implementation of Learning Analytics in FORGEBox and FORGE widgets

For the prototype FORGEBox implementation running at www.forgebox.eu/fb we selected as an Learning Record Store (LRS) the Learning Locker solution. Learning Locker (http://learninglocker.net/) is a well-known open source LRS with very good support. Thus FORGEBox users that want to track learners need to have an account on the LRS. To properly use it, they need to configure the proper LRS API endpoint for their course as well as some authorization keys.

Widgets should also be configured properly, since they are web services running separately from the course content. For courses hosted in FORGEBox we have defined a very simple protocol. Widgets are embedded through the iframe tag thus FORGEBox passes four variables:

xendpoint: the endpoint of the xAPI LRS, for example:

http://www.forgebox.eu/lrs/learninglocker/public/data/xAPI/

xapiauth: the authorization keys in Base64 format as provided by the LRS

actoremail: the email of the user being tracked

actorname: the name of the user being tracked

For most widgets that are web based a simple javascript xAPI implementation should suffice. Since FORGEBox will be also an LTI 2.0 Tool Provider the user identity and his behaviour will be also tracked from the LMS/VLE .

6. Widgets reference architecture

This section defines a generic reference architecture for developing widgets that are coupled with FIRE adapters to enable interaction with remote lab resources while integrating modern technologies from the education domain, like the LTI and Experience API. The proposed architecture intents to be a blueprint and a guide for widget developers that want to achieve the best result of supporting education on top of FIRE resources.

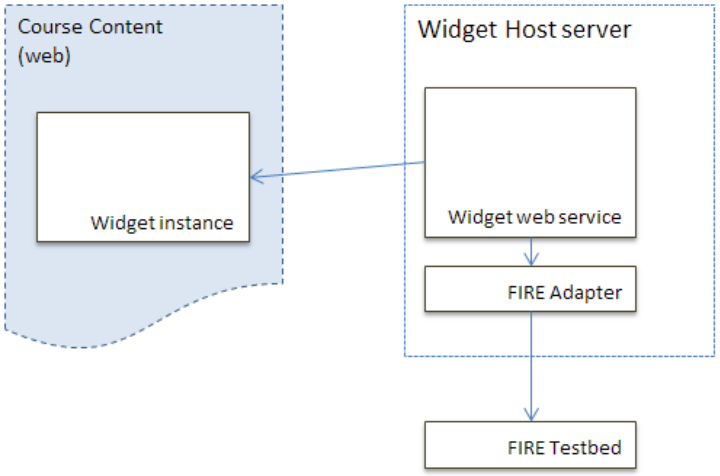

Figure 9: Widgets consumable web applications that are hosted in a web server interacting with remote resources

Figure 9 refers to widgets as FORGE uses within the courses: consumable web applications that are hosted in a web server interacting with remote resources. In FORGE case, they are also a bind with FIRE adapters, the services that handle communication with remote FIRE resources. Next, when we refer to widgets, we refer to this combination of web content and backend support for remote interactivity through FIRE adapters.

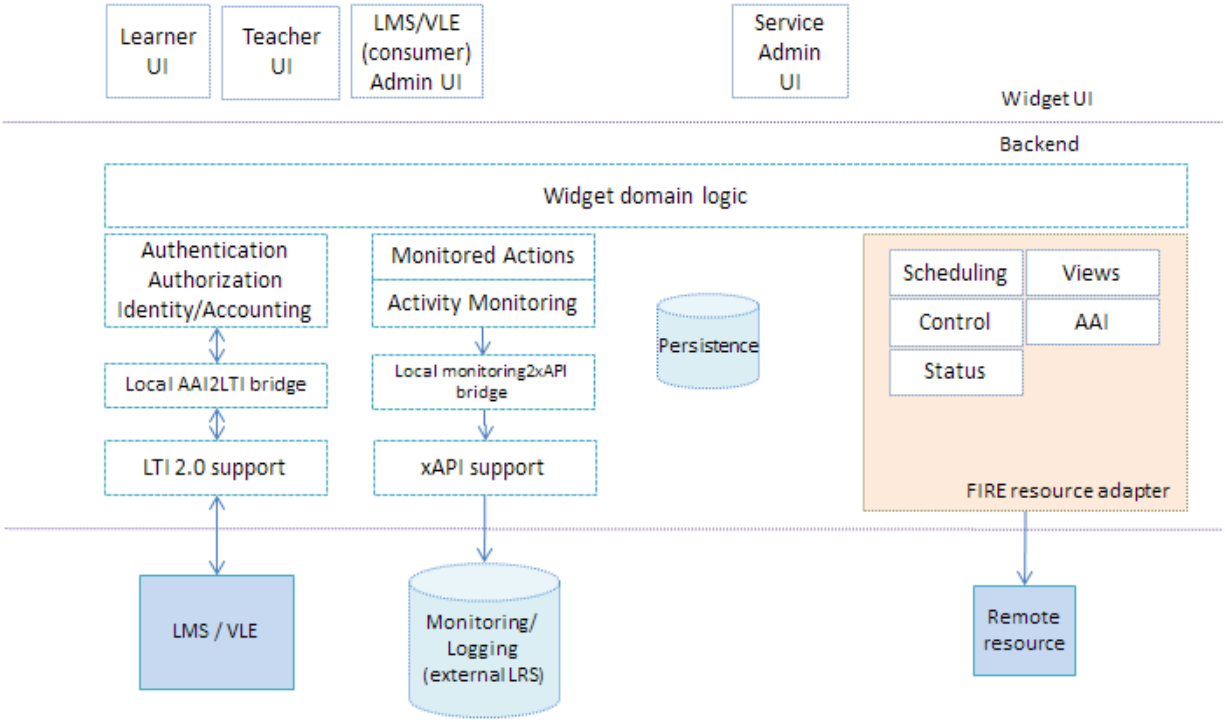

Figure 10: Reference architecture for a widget

Figure 10 displays our proposed reference architecture for a widget, with architectural components that a developer would need to implement in order to achieve the best desirable result of bridging learning with FIRE remote resource interactivity. Since widgets are web services hosted somewhere on the Internet ready to be consumed by other web content, the architecture defines both the widget UI as well as the backend domain logic and core architectural components. Next we discuss supported usage roles and each architectural component.

6.1 User roles of widgets

In this section we define a set of user roles that are able to use a widget.

6.1.1 Widget Service administrator

Service Administrator is the user responsible for the whole widget web service. Service Administrator can login to the host machine and administer the service that provides the widget to consumers. Service Administrator can also manage for example users, registrations etc. The use cases are specific to the capabilities that the widget service will offer. E.g. the administrator of the ssh2web widget can allow specific domains that can use the service.

6.1.2 LMS/VLE administrator

This user is the one responsible to integrate the widget to the target learning system LMS/VLE or even in an eBook. He needs to pay attention to the widget documentation, how it is delivered (i.e. as a URL), its API, its LTI compatibility, etc. For example, an administrator responsible for a Moodle installation could visit FORGEStore and read the documentation of the widget. Then he

could register the widget into the Moodle environment by using the LTI registration URL of the widget service.

6.1.3 Teacher/Instructor

This user will define the behaviour and settings for a specific course. He can also use the interface to reserve resources or setup the testbed.

6.1.4 Learner

This user will interact with the widget and the remote resource during the learning process.

6.2 Proposed User Interfaces for widgets

The widget UI is the main component that a user uses to interact with the widget. To behave correctly, the Widget service must know the context that it works under, in order to properly display the equivalent UI according to the user role. Thus, if possible, the widget should be aware of:

- The consumer service into which it is hosted and operating (i.e. is it an LMS/VLE, the VLE url, an eBook, etc.);
- The kind of consumer (i.e. its capabilities, browser, tablet etc.);
- The identity of the current user and his role;
- The current course (content or page reference).

All this information can be passed either through a widget API (e.g. passing URL parameters) or via more modern ways like the LTI API. According to the user role there should be different UIs. Thus some first requirements for a widget service should be:

- An API to call the widget and pass user identity and context;
  - For this, LTI usage is a good candidate
- Specific endpoints (urls) that will service each user according to his role
  - For example service administrators visit http://www.mywidgeturl.org:8080/admin
  - For example a VLE admin visits http://www.mywidgeturl.org:8080/lti/register

It is not necessary for widgets to implement all these user interfaces. For example the FORGE widgets of Teacher Companion Lab courses don't need to provide a Learner UI since they can be used only by Teachers.

Next sections present these proposed UIs and some minimum example functionality.

6.2.1 Service Admin UI

The administrators of the web service use this UI. Through this interface the widget administrator can manage for example:

- The service, its specific entities and configures it;
- All users and User policies;
- LTI registrations;
- All registered courses that consume service;
- Backend connections etc.

The following figure is an example of a Service Admin UI, where the ssh2web administrator can manage all users and all registered systems.

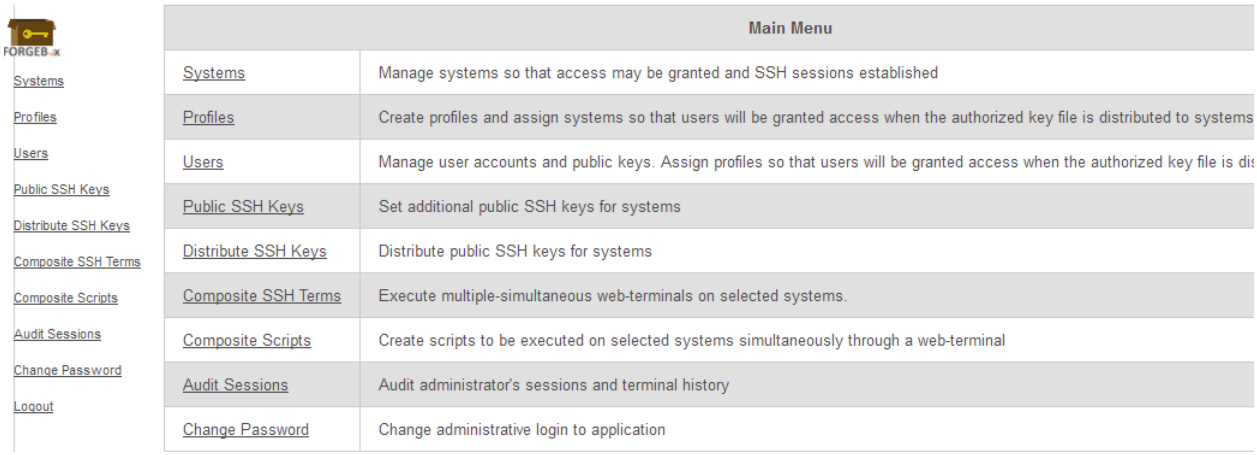

Figure 11: Service Admin UI, of the ssh2web administrator

6.2.2 LMS/VLE Consumer Admin UI

Through this user interface the administrator of a VLE can interact with the widget. For example the LMS/VLE administrator through this UI can:

- Request to register (i.e. for LTI) the service for usage;
  - For example the administrator of Moodle can register the Widget via LTI in order all courses of Moodle to use the widget;
- Configure the service for his LMS/VLE users;
- Configure a globally xAPI end point for Learning Analytics usage in the LMS/VLE;
- Monitor his users and assign policies or change roles (for LTI this could be done automatically for example).

Next figure displays an example of a Moodle Administrator using the ssh2web widget through LTI to administer his users. The widget has automatically recognized the user name (Admin User) and role (Instructor), which are provided by Moodle via the LTI2.0 API.

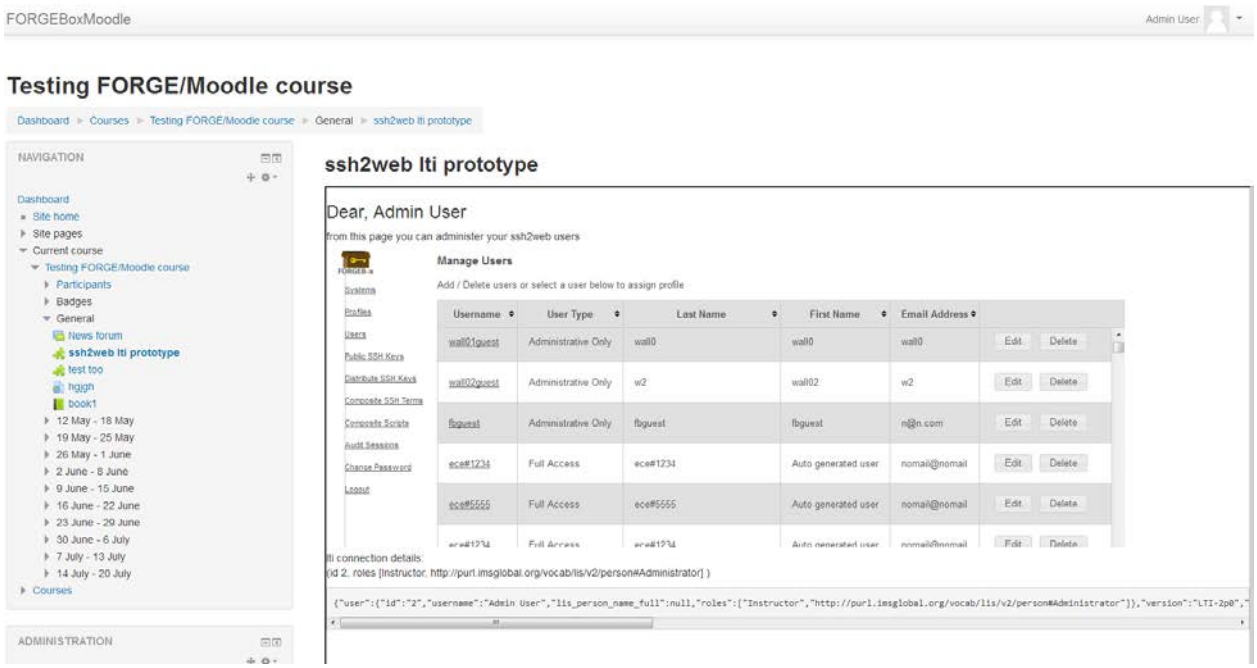

Figure 12: Moodle Administrator using the ssh2web widget through LTI

6.2.3 Teacher/Instructor UI

Through this user interface the Teacher/Instructor of a course that uses this widget could do the following:

- Defines behaviour and settings for a specific course;
- Schedules resources and setup the testbed;
- Might assign users to exact resources (or leave it to assign them by the service).

Teacher/Instructor UI examples are the FORGE widgets of Teacher Companion Lab courses as presented in section 2.3.

6.2.4 Learner UI

Through this user interface the learner interact with the remote FIRE resource. Many examples have been presented so far in different deliverables.

6.3 Backend widget services

The backend of a widget contains all the support services that implement all the widget's domain logic. The core services that are envisaged by FORGE widget reference architecture are presented in next sections.

6.3.1 Authentication Authorization Identity

This service, is almost mandatory, as it should handle the users accessing the widget service, while affecting the widget behaviour according to user role (i.e. provide the equivalent user interface). What is interesting to provide is an LTI2.0 implementation. This will allow, as discussed before, the better integration of the widget with existing VLEs. Thus we recommend widget developers implementing a bridge service between the AAI widget service and an LTI 2.0 support library.

6.3.2 User activity monitoring

This service is responsible for monitoring user activity while interacting with a facility, especially if the widget needs to audit users for their behaviour. What would be useful to implement is to provide to teachers the user behaviour while interacting with the facility. So it is highly recommended for widget developers to consider the integration of Experience API and the ability to report user behaviour to an external Learning Record Store.

6.3.3 FIRE Resource adapter

FIRE adapters are tightly integrated with a widget, since the widget will provide the UI interacting with the FIRE facility. FIRE adapters will use provided APIs and tools from FIRE, especially those developed by the Fed4FIRE project. The ideas as well as how FIRE Adapters should work was presented in D2.1 (Adaptations made to the FIRE APIs to support education in prototype lab courses).

7. Conclusions

The work presented in this deliverable, provides the approach and describes the main architectural elements that FORGE implemented and offers as a technology. The work was affected by the widgets and FIRE Adapters implementation so far but also by the prototype FORGEBox and FORGEStore technologies. Valuable feedback came also from interaction by WP2 and WP3 which implemented these services, as well as information and requirements from the Lab course design and deployment by WP5.