BREAKING EDUCATIONAL BARRIERS WITH CONTEXTUALISED PERVASIVE AND GAMEFUL LEARNING

Grant Agreement No. 687676

Innovation Action

ICT-20-2015

D4.4 Prototype of the Authoring and Procedural System Framework Component

Due date	[Month 28]
Actual date	[Month 30]
Deliverable author(s)	António Coelho, João Faria, João Jacob, Pedro Cardoso
Partner(s)	INESC TEC
Version	V1
Status	Final
Dissemination level	PU

Project Coordinator

Coventry University

Sylvester Arnab

Priory Street, Coventry CV1 5FB, UK

E-mail: s.arnab@coventry.ac.uk

Project website: http://www.Beaconing.eu

This project has received funding under the Horizon 2020 Framework Programme European Union – Grant Agreement No. 687676

Version control				
Version	**Date**	**Author**	**Institution**	**Change and where applicable reason for change**
V1	23.04.2018	António Coelho João Faria Pedro Cardoso	INESC TEC	Draft version
V2	28.04.2018	António Coelho João Faria	INESC TEC	Integration of the reviewers' feedback.
V3	29.04.2018	Pedro Cardoso	INESC TEC	General revision.
V4	11.05.2018	João Faria Leonel Morgado	INESC TEC	Updated LMS section
V5	16.05.2018	João Faria	INESC TEC	Updated PCG section

Quality control				
QA Version	**Date**	**QA Responsible**	**Institution**	**Change and where applicable reason for change**
V1	26.04.2018	Luca Morganti	Imaginary	Review of the document
V1	27.04.2018	Michael Loizou	COVUNI	Review of the document
V5	03.06.2018	Ioana Stefan	ATS	Final review of document. Minor changes requested
V5	12.06.2018	Michael Loizou	COVUNI	Internal review of new version, acceptance of changes
V5	12.06.2018	Luca Morganti	Imaginary	Internal review of new version, acceptance of changes

Release approval				
Version	**Date**	**Name**	**Institution**	**Role**
V5	05.06.2018	Ioana Stefan	ATS	WP lead
V5	12.06.2018	Michael Loizou	COVUNI	Internal reviewer
V5	12.06.2018	Luca Morganti	Imaginary	Internal reviewer
V5	15.06.2018	Jannicke Baalsrud Hauge	BIBA	QM

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

TABLE OF CONTENTS

LIST OF FIGURES

List of Abbreviations

GLP	Gamified Lesson Plan
PCG	Procedural Content Generation
JSON	JavaScript Object Notation
GUI	Graphical User Interface
AT	Authoring Tool
GI	Game Instance
LMS	Learning Management System
LBG	Location Based Game

SUMMARY

The Deliverable 4.4 document describes the Prototype of the Authoring and Procedural System Framework Components which are the technical components of the BEACONING platform that implements the functionality to create, edit and gamify a Lesson Plan.

The authoring tool is the main component as it allows the creation and editing of the Gamified Lesson Plans (GLP), integrating the pedagogical content with the game plots. So, the architecture integrates most of the BEACONING components and the graphical user interface is intuitive in order for a teacher to understand how the game plot will drive the educational activities.

From a gamified game plot, the teacher is able to deploy the games for each of her/his students. With the procedural content generation component, it is possible that each game is adapted to each student's needs, both in what concerns learning performance and accessibility profile. The teacher is also able to integrate BEACONING with the current Learning Management System (LMS) in use on her/his institution.

STRUCTURE OF THE DOCUMENT

The deliverable is structured as follows:

- **Section 1** provides an overview of the authoring tool component, focusing on its architecture and Graphical User Interface (GUI);
- **Section 2** describes how to author the structure of a GLP**;**
- **Section 3** provides an overview of the Procedural Content Generation component, describing its three main functionalities;
- **Section 4** describes the integration with an LMS.

1 AUTHORING TOOL

The authoring tool is the component that allows the Learning Designer to create new gamified lesson plans (see 4.8 Gamified Lesson Plans) in order to structure learning activities related to a specific topic. This tool can also be used by any teacher to adapt a specific GLP.

In the next sections, we present the Authoring Tool Architecture and its GUI.

1.1 THE AUTHORING TOOL ARCHITECTURE

The Authoring Tool (AT) runs on a server, with a vast extent of endpoints to allow a broad manipulation of elements inside the GLPs.

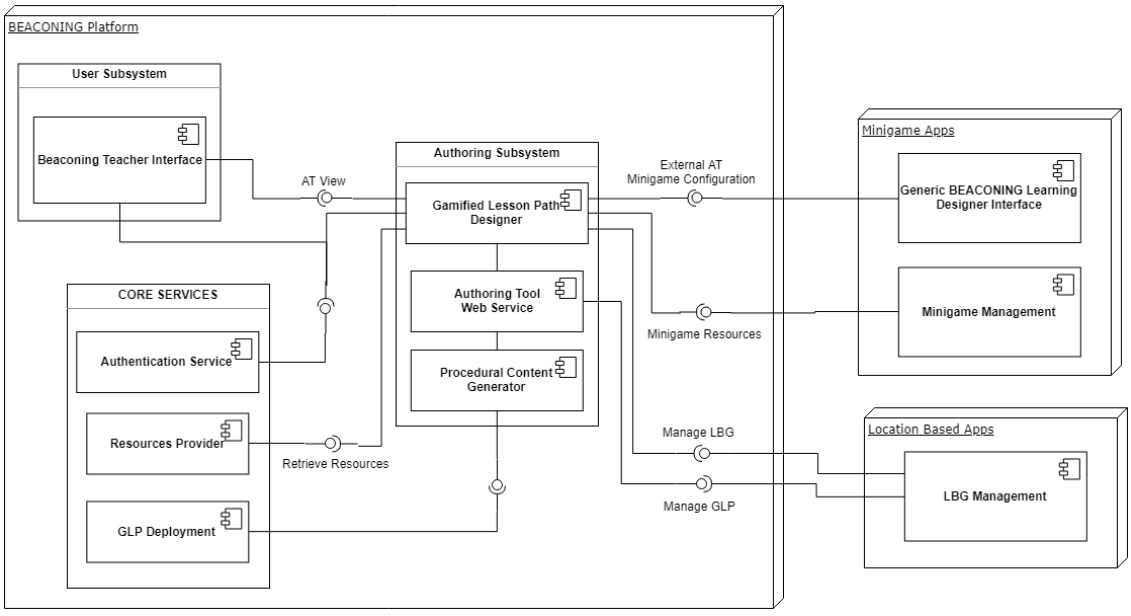


Figure 1 - Authoring tool architecture

Figure 1 describes the component architecture of the Authoring tool (AT).

The main services provided by the AT are:

- Service to open the configuration of one GLP displayed in the Teacher GUI. This service can be used without the Teacher UI as well.
- Service to update one GLP consumed by the Location Based Apps.
- Service to deploy one GLP capable of adaptation of the contents based on the students' profiles. This service is consumed by the core-services.

The main services consumed by the AT are:

- Authentication provided by the core-services. This service uses OAuth2.0.
- Management of the GLPs, from creating to updating them, provided by the core-services.
- Retrieval of minigames and location-based apps available in the BEACONING ecosystem, along with all the information required to use them.
- Retrieval and storage of minigame configurations provided by the minigame apps.
- External authoring of minigames provided by minigame apps. This external edition means there is no configuration on AT side.
- Management of location-based games provided by the location-based apps.

1.2 THE AUTHORING GUI

The graphical user interface was designed with the purpose of being intuitive for a Learning Designer or Teacher, focusing on a hierarchical view of the Lesson Plan. The integration with the game plots (see D3.5 - Game Design Document) is achieved by a graph-based description that depicts the unfolding of the underlying narrative. This allows the user to easily understand how the Lesson Plan will progress in the plot of the game.

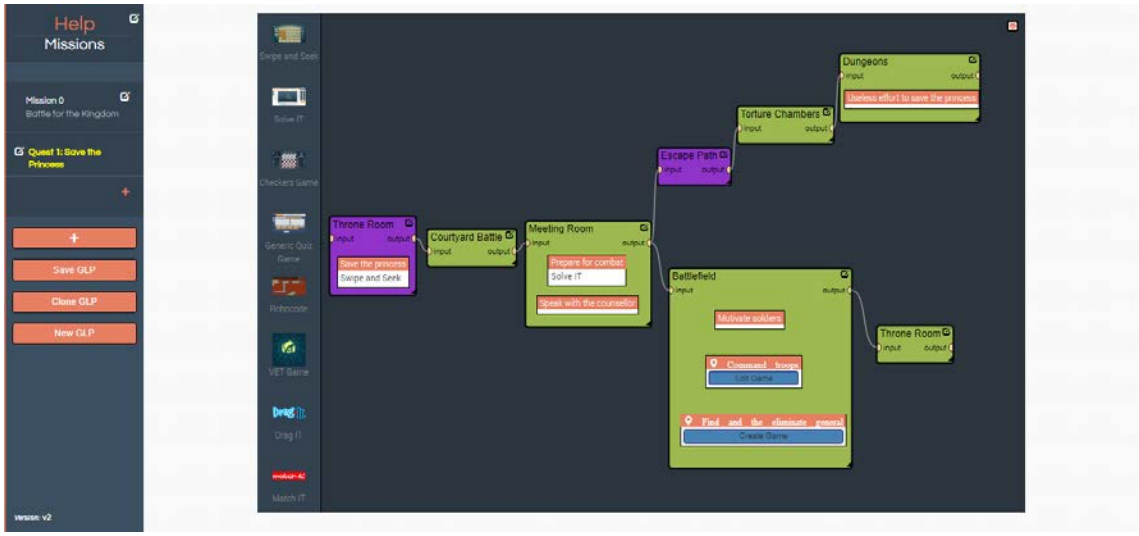
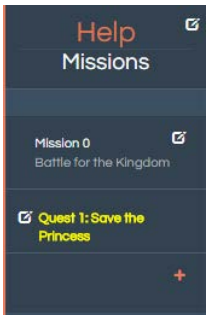
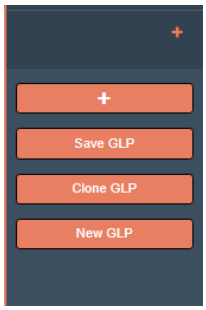


Figure 2 - GUI of the Authoring Tool

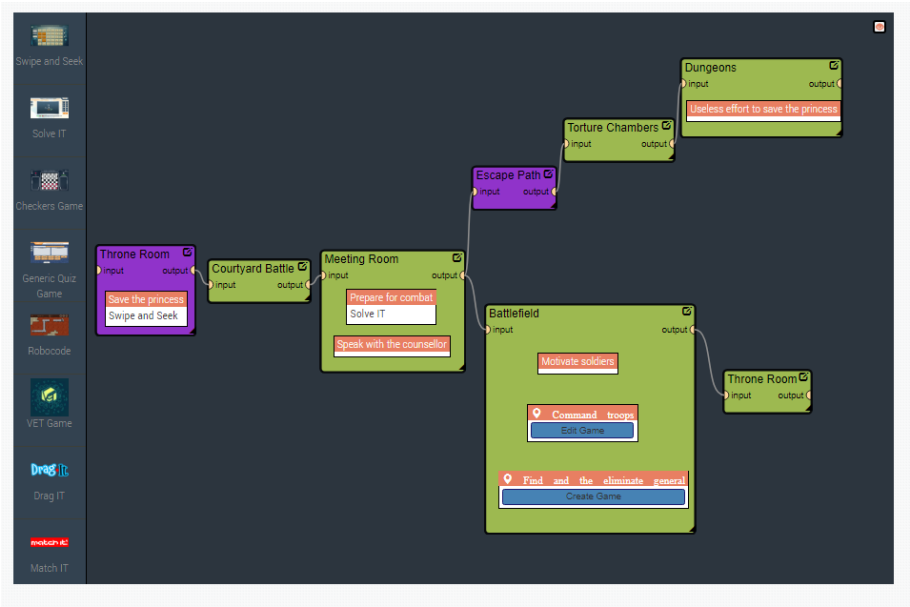
Figure 2 describes the main page of the Authoring Tool. This main page is divided into distinct sections:



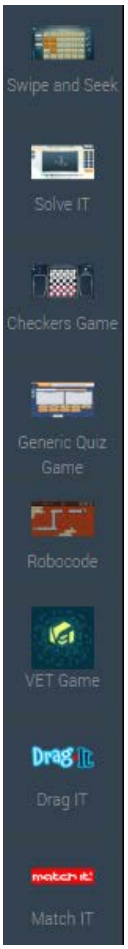
1. The Lesson Plan container. This container is used to define the distinct missions that compose the Lesson Plan and the quests contained in each mission.



2. Options regarding the GLP, such as adding a Quest/Mission, saving the GLP, cloning the GLP, and creating a new GLP.

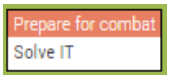


3. Graphical view of the Game Plot associated to the selected Quest.



4. List of the existing minigames in the BEACONING ecosystem.

5. Scene, which holds the external values of the game plot.



6. Node, where minigames can be added.

The meta information regarding the GLP and the Lesson Plan containers can be edited by using a dropdown menu that can be opened by pressing the edit icon (Figure 3).

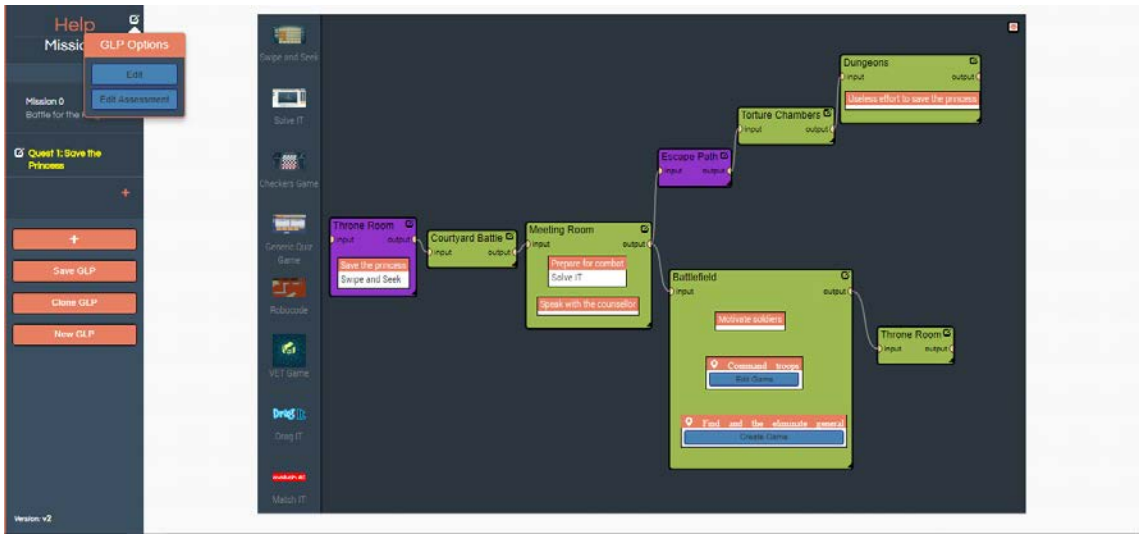


Figure 3 - Editing the GLP¨

There are two options for each of the Levels of the Lesson Plan:

1. **Edit** - This option is used to edit the meta information. An example of the information that can be edited (Game name, Description, Domain/ Area/ Subject, Topic) is shown in Figure 4.

Figure 4 - GLP editing dialogue

2. **Edit Analytics** - This option allows the edition of the analytics (e.g. Number of attempts available for each game, Time limit, Thresholds) for a particular level of the GLP.

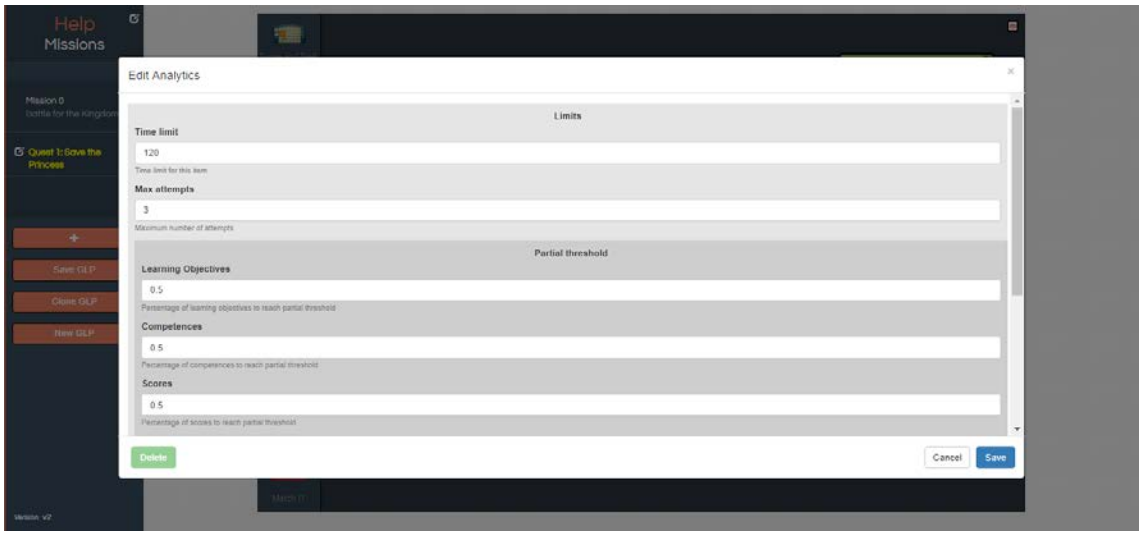


Figure 5 - Editing the GLP analytics

As seen in Figure 5, the Edit Analytics section of the Lesson Plan only defines the thresholds, which are used to allow a student to move forward in the GLP and to reward a bonus.

The activities (minigames) inside Nodes can also be edited. The dialogue that is shown to edit an activity is dynamically generated from a JSON scheme provided by the minigame developer. In case a configuration already exists, the fields are automatically filled and, if required, new elements are added to arrays. The scheme follows the Schema Draft V4 standard. Figure 6 shows an example of a dynamically generated configuration (including Game language, Game description, Topic).

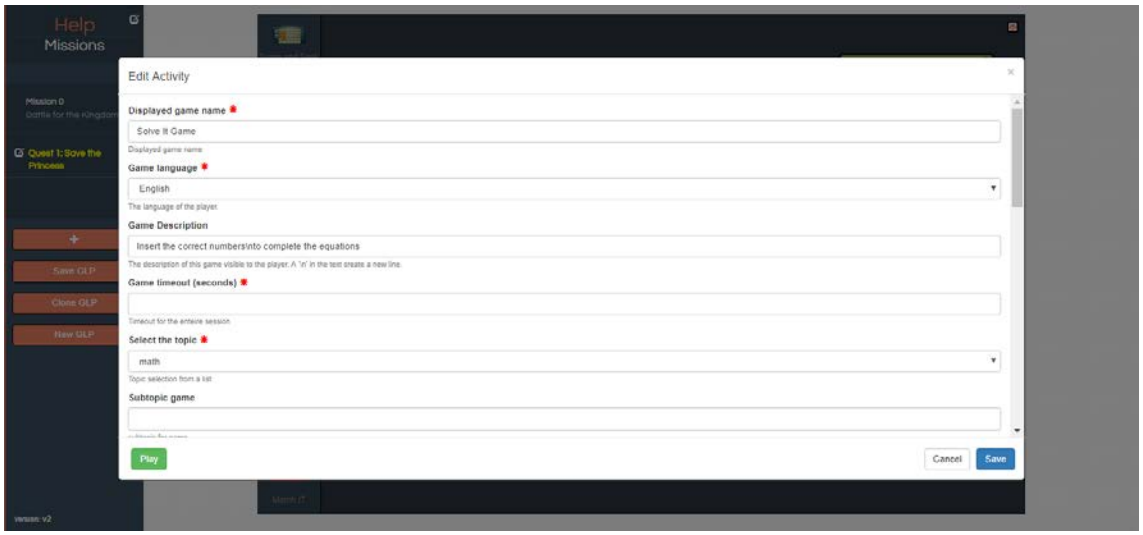


Figure 6 - Editing a minigame

In figure 6, one can see that there is a field missing in the configuration, so trying to save the configuration will result in an error. The AT also has a simple and user-friendly way to show errors in configurations as, by following a mapping of elements inside the configuration, the location of the error can be tracked and displayed to the user.

An Activity has extra options that allow to configure the analytics component, according to the Lesson Plan nodes. Figure 7 shows these new options.

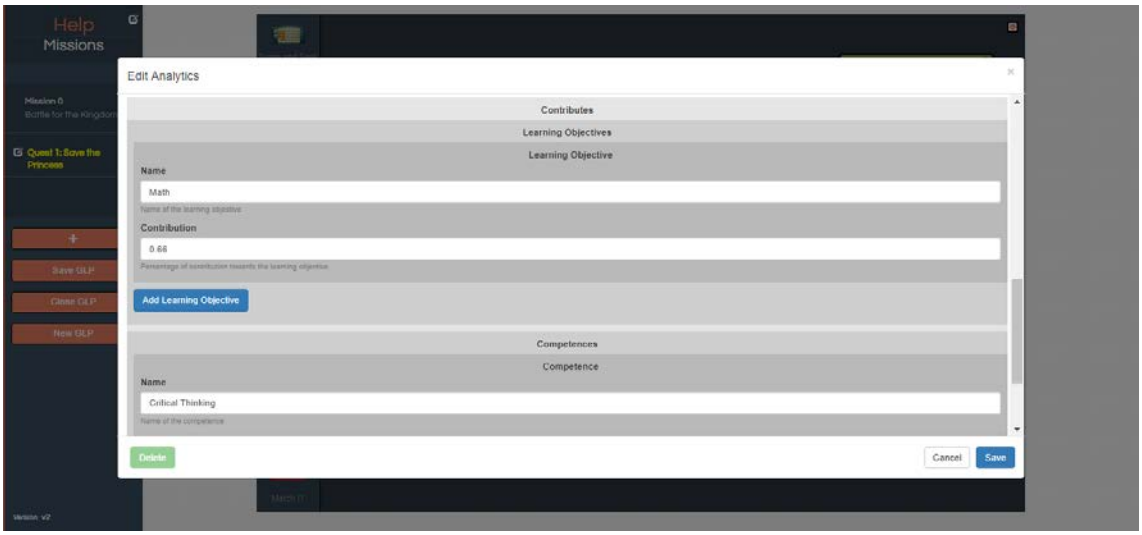


Figure 7 - Editing the analytics options of a minigame

There is a new object “Contributes” inside the Edit Analytics dialogue that allows defining how the Activities contribute to achieving specific learning objectives and competences based on the activity result obtained by the student.

Figure 8 shows the dialogue for creating a new GLP. This form is similar to the one to edit GLP but it has the selection of the game plot for the first Quest.

Figure 8 - Creating a new GLP

2 AUTHORING THE GLP

The GLP is the core structure of the BEACONING system, mapping the Lesson Plans to Game Plots. The authoring tool is responsible for the mapping of these two concepts: learning + gaming. Figure 9 describes the GLP data model.

The authoring tool manages the hierarchical structure of missions and quests inside a GLP and integrates the analytics. This way the learning designer/teacher is able to define the structure of the Lesson Plan in terms of learning objectives and competences (missions) and also the educational activities (quests).

At quest level, a graph instantiates a game plot, the game narrative that will gamify and drive the learning process. The graph contains game scenes, each comprising nodes where minigames can be instantiated (activities) and the configuration of the analytics to be collected.

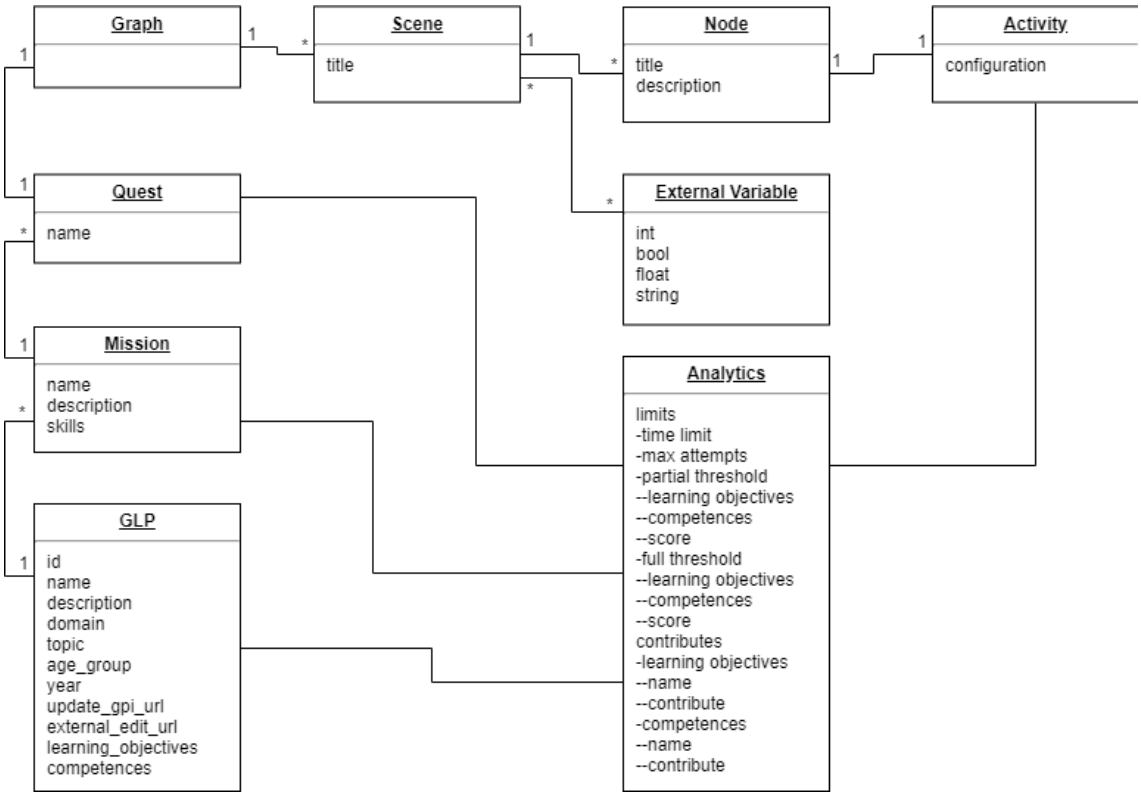


Figure 9 - GLP data model

2.1 GLP

The GLP structure represents a Gamified Lesson Plan that students will undertake. A GLP is represented by its ID and has a series of fields that are defined as:

- **Name**: The name of the GLP.
- **Description**: The description of the GLP.
- **Domain**: The domain the GLP is inserted in.
- **Topic**: The topic the GLP relates to.
- **Age group**: The target age group of the GLP.
- **Year**: The year the GLP was created/ modified.
- **Learning Objectives**: The learning objectives of the GLP.
- **Competences**: The competences that the GLP helps train.

2.2 MISSIONS

A GLP is composed of Missions. Each Mission has:

- **Name:** The name of the Mission.
- **Description**: The description of the Mission.

- **Skills**: The skills acquired by successfully completing this Mission.

2.3 QUESTS

A Mission is composed of Quests. Each Quest represents an educational activity that is associated to a Game Plot.

The Quest only has a **Name** and a **Graph** associated with it, which is the representation of the Game Plot.

2.4 SCENES

Inside the Graph, we have a representation of the Game Plot. This representation consists of an aggregation of Scenes, which are the basic blocks of the plot of the game (rooms, worlds, levels, …).

The Scene can be identified by its title. The order of the scenes can be perceived by the connections shown in the GUI. Inside a Scene there are:

- **External Variables**: values from the game plot that can be modified and are reflected into the game, such as the name of a character or some dialogues. One external variable can be located in several Scenes.
- **Nodes**: the slots that include minigames.

2.5 NODES

A Node is a slot that can be filled with a minigame for students to play. Nodes can have special types that don't allow minigames to be dropped inside when they are related to Location-Based Games (LBG) – see D4.2 Location-based Component. In the event of a Node belonging to a LBG, several types can be shown:

- **Create**: The node is filled with a button to create a new location-based game.
- **Edit**: The node is filled with a button to edit the LBG, which only happens when it has no points of interest.
- **Minigame**: The node is a regular one, but has an icon that represents it as belonging to a LBG and signifies whether it is a GPS point or a Beacon.
- **Checkin**: The node has the phrase "Check In" inside. This node doesn't possess any activity. The player only needs to reach a certain location on the map.
- **Upload**: The node has the word "Upload". It is used for the student to upload one file upon reaching a certain location associated with it.

Location-based nodes have connections between them, representing the sequence of the LBG.

2.6 ACTIVITIES

Activities are minigames retrieved from the core services, that can be configured according to the Teacher's needs.

2.7 ASSESSMENT

The assessment of the student is done through the activities. The Lesson Plan aggregations (Missions and Quests) have no effect on the final results. The Teacher can change the assessment through the Contributions present in the Activities. The Contributions can be of two types (learning objectives and competences) and are calculated as a percentage (%) of the score of the Student in each Activity.

3 PROCEDURAL CONTENT GENERATION (PCG)

Procedural Content Generation (PCG) allows the automation of content generation on the BEACONING system in three modes:

1. Game generation according to the students' profile (section 3.1);
2. Graph representation of the game plot as an intuitive representation for the learning Designer (section 3.2);
3. GUI customization of the configuration menus for each minigame (section 3.3).

3.1 STUDENT PCG

The GLP is a generic description of educational activities that a teacher can deploy in her/his class. So, the GLP is not accessible to a student, but only to a teacher. The teacher is responsible to deploy a GLP in a class, a Game Instance (GI) for each student is created providing either the same or a customized experience to the students.

The BEACONING approach is that this GLP can be adapted to the student's profile in order to maximize her/his learning activities. This is done by:

1. Analysing the student's performance, through the analytics. The PCG component is able to adjust the difficulty level for each student;
2. Using the accessibility profile of each student, the PCG component will adapt the game to her/his needs.

In order to deploy a game, the PCG component generates a game instance based on the GLP. Figure 10 describes the game instance data model.

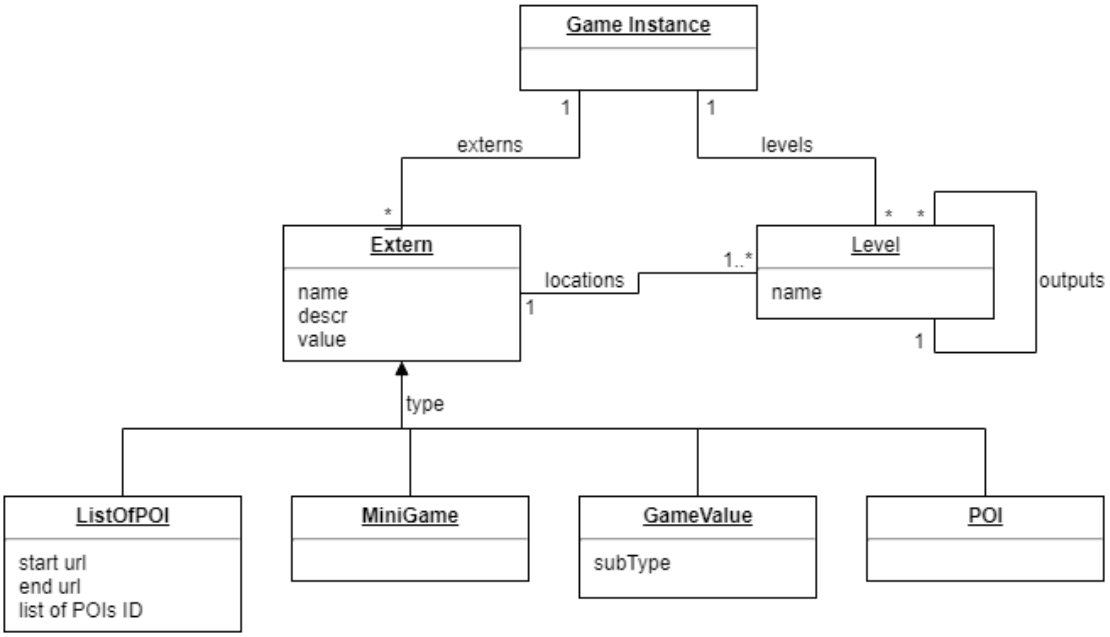


Figure 10 - Game instance data model

In the Game Instance Data Model, we can see that the Instance has two elements:

- **Levels** - This represents the game levels inside the GI. They are presented to the Teacher as Scenes since they represent parts of the game world where challenges occur.

- **Externs** - The externs represent all of the external variables that can be found in the GI. These variables are the values of the game that can be modified without using the Game Plot Editor tool. It is possible, for example, to change the game characters and the dialogues.

Inside each of the Levels, we have two fields:

- **Name** - Represents the name of the level.
- **Outputs** - Represents the other levels that can be reached from the current one. It is a list of names of levels.

Inside each of the Externs there are:

- **Name** - The name of the extern variable.
- **Descr** - The description of the extern variable.
- **Value** - The value of the extern variable. This is a string and its value depends on the type of the extern.
- **Type** - The type of the extern is what defined what kind of handling the AT-GLP will allow. There are 4 types:
    a. **MiniGame** - This is the most useful type of Extern Variable, since it is what allows a game to receive lesson material for the student to be assessed. The minigame's value is an URL to play the minigame.
    b. **GameValue** - This is the most common type of Extern Variable. It allows modification of values inside the Game. These values have subTypes, which determine what can be accepted. The subTypes are:
        - **bool** - A boolean value that is either true or false.
        - **int** - An integer value.
        - **float** - A floating point value.
        - **string** - A string of text. This is the most useful of all the game values since it allows translation of the Game Plot into different languages.
    c. **POI** - A Point of Interest, that represents a location in the physical world. It can be either a beacon or a GPS coordinate. Internally, in the authoring tool, it has a value that represents a URL to play that Location-Based Game (LBG). Therefore, it is similar to a MiniGame, the difference is how the user creates the URL to play.
    d. **ListOfPOI** - This type represents a full LBG. Inside the authoring tool, this type of extern adds more nodes, but those nodes are not additional externs. The list of POI is the only type where the value is not a string. In this particular case, we have an object inside its value, that has the following fields:
        - **startUrl** - The URL used to start the LBG.
        - **endUrl** - The URL that is called once the game is finished.
        - **list** - A list with the playUrl of each of the POIs inside the LBG. Although we are passing the list of the play URLs, it is not the game engine that detects if a POI has been reached or not. The URLs in the list are only used in the event of the student closing the app in the middle of the LBG and then needs to resume the game where he left off.

3.2 SCENE GRAPH PCG

When adding a Quest to the GLP the user associates a Game Plot. This game plot (represented in a JSON file) is converted into the GUI of the authoring tool as a scene graph.

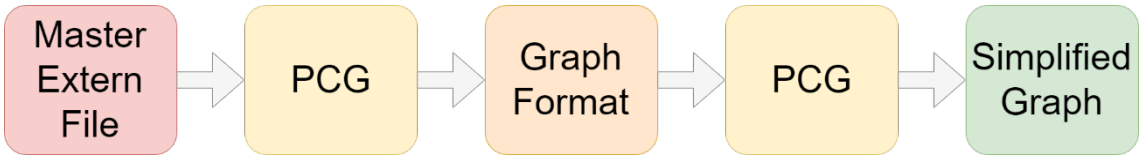


Figure 11 - PCG of the Scene Graph

As seen in Figure 11, the PCG first converts the master extern file into a Graph, so that the Learning Designer can see a representation of the Game Plot. This representation can be explored for a full understanding of all the paths a student can take and all the steps (s)he goes through.

The PCG receives a JSON file containing a list of Levels, which are mapped into Scenes and a list of external variables, which can be mapped to several types of content. First, the PCG converts all of the Levels into Scenes, using the name as a unique identifier. After creating the Scenes, the PCG adds the connections between them. Next, the list of external variables is read. Each variable can be a minigame slot, a game, a location-based game or a single location-based location. The game values are assigned to the respective Scenes they belong to. In the case of Game Values, they can be located in several Scenes. These values can have several subtypes, for example, a dialogue, a character's name, the number of doors a player has to open to find something, etc. In the end, the PCG converted the Master Extern File into a Graph that represents the Game Plot.

Although this is a human-readable representation, it is very hard to understand. Game Plots can be very extensive and intricate, and it is really difficult to figure out how the game plot works. For this, the PCG runs an algorithm that simplifies the graph and turns it into a user-friendly representation of the Game Plot that keeps all of the relevant information and discards all of the unnecessary scenes and connections that offer no customization for the Teacher.

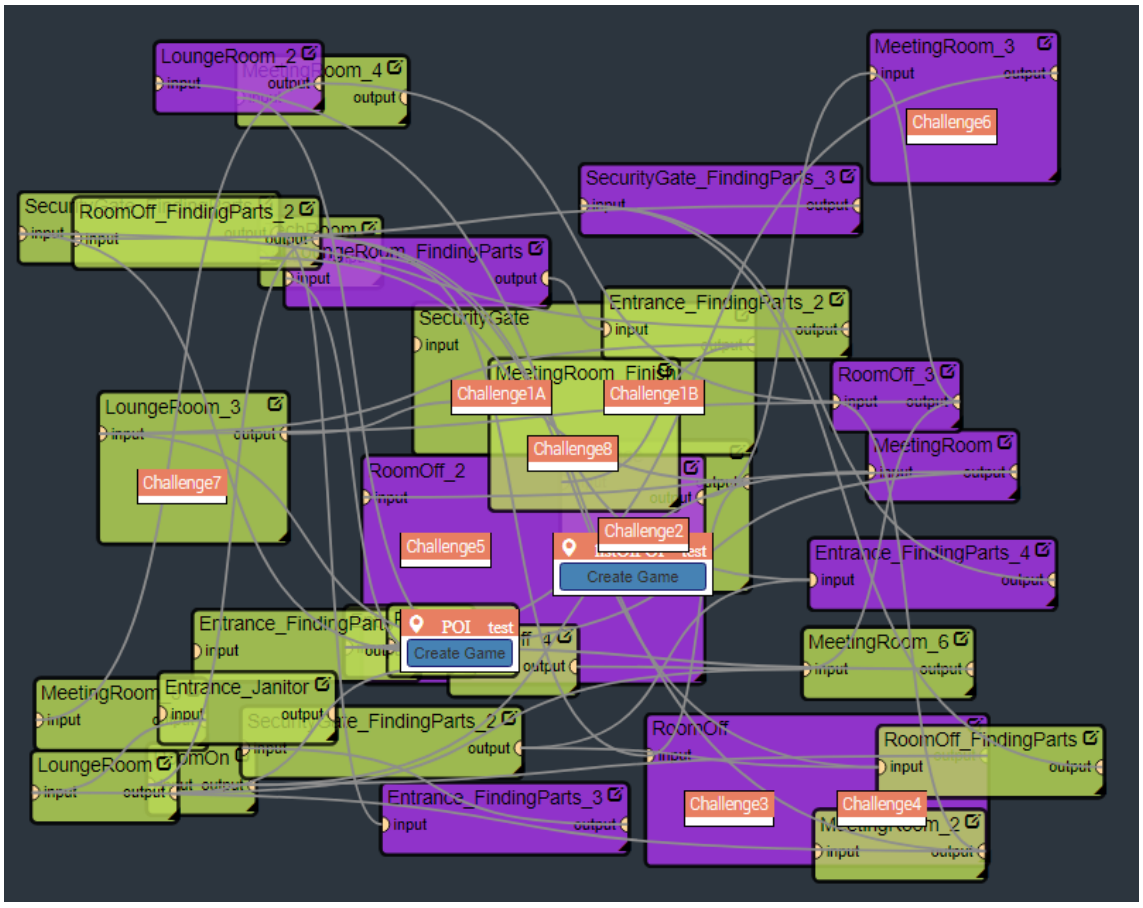


Figure 12 - Representation of all the components of a game plot

Figure 12 shows the graphical representation of a game plot. We have 32 Modules in the game plot and 11 Nodes, two of which are location-based games.

As it can be seen, there are too many modules and connections, preventing the diagram from providing an intuitive presentation of the game plot.

The PCG component selects the main modules that compose the scenes that drive the plot of the game. The algorithm begins with the starting Scenes, adding them to the simplified representation. Then, it goes through the connections and removes every intermediate Scene that doesn't have any nodes or extern variable. The end Scenes are also preserved, as well as the possible connections between the Nodes that remain in the Scene.

The final Graph, after being simplified by the PCG is shown in Figure 13.

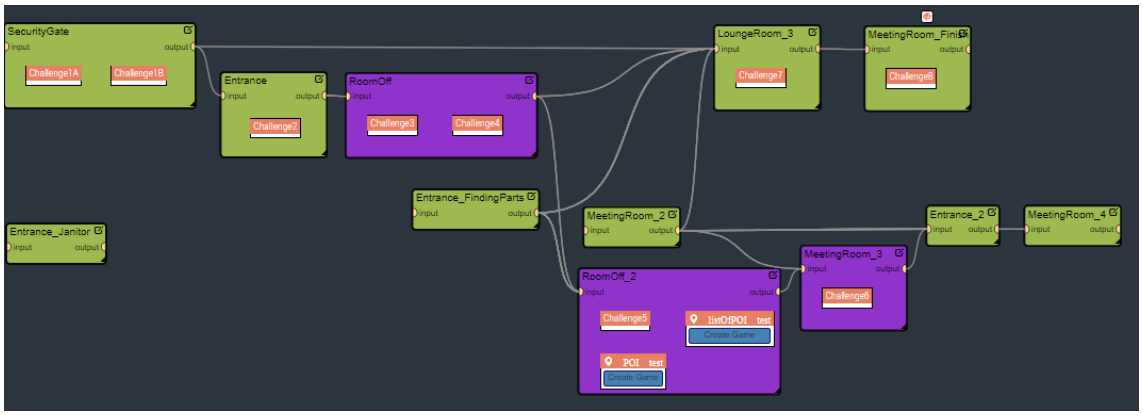


Figure 13 - Game Plot simplification by the PCG component

3.3 GUI CUSTOMIZATION

The PCG component is also able to provide a customized dialogue for each minigame, derived from the JSON file representing its scheme and provided by the developer. It creates a dialogue that follows that scheme, adding every element present in it and all options to customize it. In case there are default values for any of the fields, an additional file with the resources need to be provided.

The authoring tool also loads already existing configurations into the form, adding extra elements to arrays, if needed.

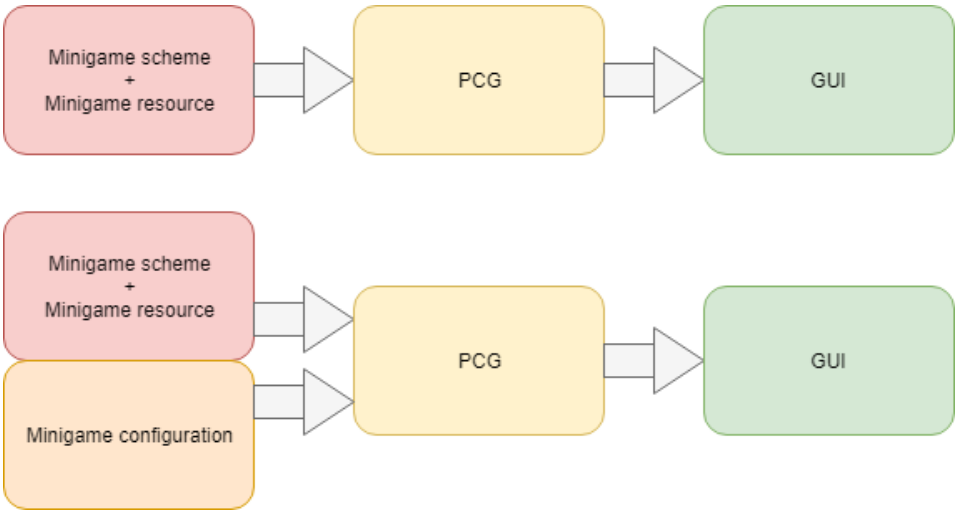


Figure 14 - Custom dialogue creation for minigames

Figure 14 represents the flow of the PCG component to create the dialogue of the minigame based on the JSON scheme.

Inside the scheme, several inputs can be added, according to Schema Draft V4. These inputs are:

- **string** - A string value. It can be either a single line or a text area.
- **integer** - An Integer number.
- **float** - A floating point number.
- **percentage** - A percentage number. Introduced to ease the assessment edition.
- **boolean** - A boolean value.
- **array** - A list of objects, that can be either inputs or objects.

- **object** - An aggregation of inputs that belong to a particular element. Typically, arrays are lists of objects with properties inside.
- **select** - A select input, with a dropdown to choose the answer among given options.

Figure 15 shows an example of a dynamically generated dialogue.

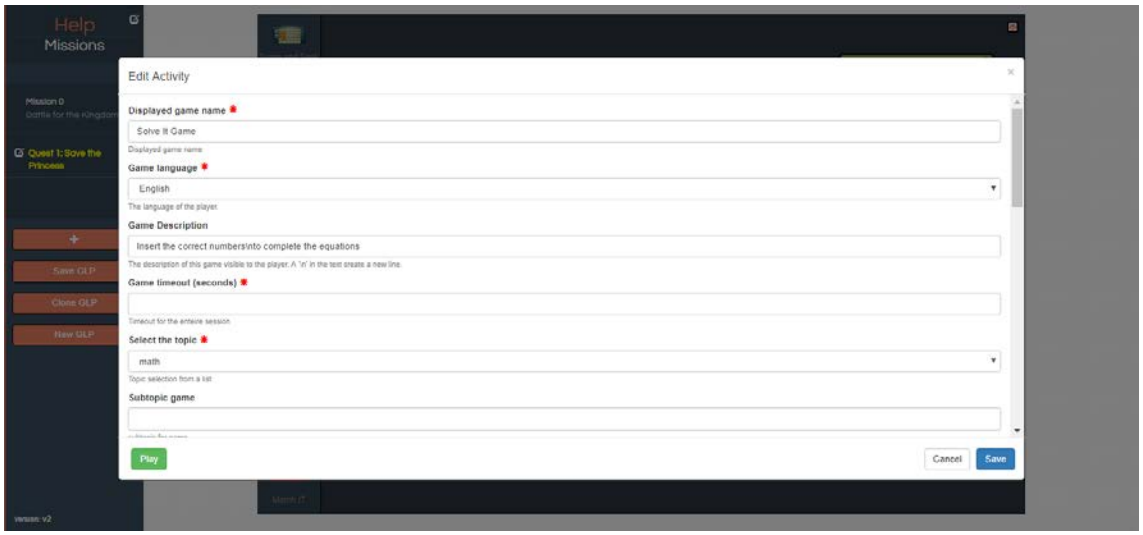


Figure 15 - Dynamically generated dialogue

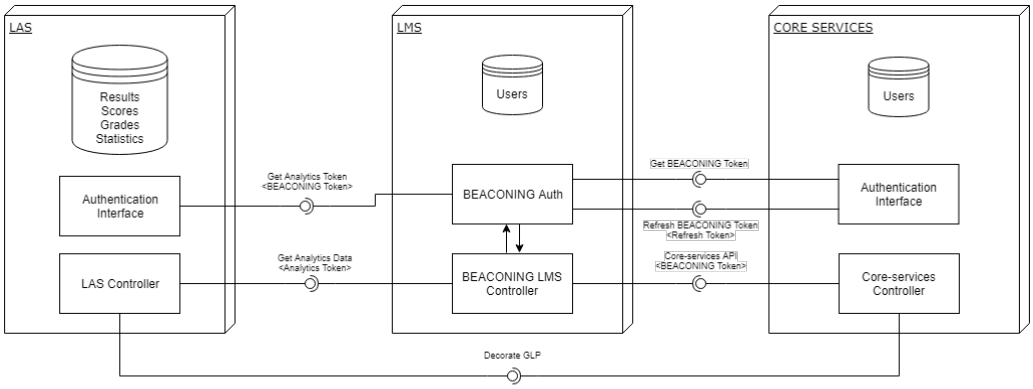
As it can be seen, it can read mandatory fields and flag them.

This PCG algorithm is also capable of injecting errors into the form, so that when the user tries to commit some invalid information, the external server that validates the JSON generated from the dialog returns an object that tracks where the errors are and the algorithm is able to present the errors along the dialog, for easier identification by the Teacher.

4 INTEGRATION WITH LMS

BEACONING is a flexible system that integrates with other Learning Management System (LMS) in order to easily adapt to current pedagogical environments. Figure 16 describes the LMS integration with the BEACONING system.

4.1 LMS Components



LMS Database USER

ID	Name	BEACONING Token	Refresh Token	Analytics Token
1	João	jkasjdskKDHAKJBDbjad	RfsjaheHDjnFJad	JdJSfsknfkIfaD
2	Jacob	fsljkknKLakjfskKHkjJJnn	adjkIHIuGadsdsD	kadjGHDjksda
3	Pedro	iopUkmMbvFyfJGUIHhs	asdsjiojHDIAsadf	kfFSinFinwPdf
4	Tony	qweçljssIGFTFfBNMerty	DAdjkdkanuUIHjk	78gJ7gdhjIdsU

Core Services Database User

ID	LMS ID	Token	Refresh Token
253	1	jkasjdskKDHAKJBDbjad	RfsjaheHDjnFJad
257	2	fsljkknKLakjfskKHkjJJnn	adjkIHIuGadsdsD
324	3	iopUkmMbvFyfJGUIHhs	asdsjiojHDIAsadf
521	4	qweçljssIGFTFfBNMerty	DAdjkdkanuUIHjk

Core Services API Clients

ID	Client ID	Client Secret
1	Authoring Tool	gjgdhNkhsjahfsuqoiwu
2	VCloud	HJKJioiu98gbijkhbGJ6
3	Moodle	HkTDLFÇIYOILghjfkui
4	Geomotion	jkhjgFYKjTYCGHVBd

Figure 16 - LMS Component Diagram

4.1.1 LMS Modules

- **BEACONING Auth**: A module that creates a custom authentication into the LMS. This allows the LMS to link a BEACONING user to an LMS user, keeping student data safe and private on the LMS side, while being able to use the full extent of functionalities present in BEACONING ecosystem.
- **BEACONING LMS Controller**: A web controller that handles the communication between the LMS and the BEACONING ecosystem. All the communication should happen through core services, for security.

4.1.2 Core services modules

- **Authentication Interface:** Provides an Authentication Interface that handles the login of LMS.
- **Core-services Controller:** Provides an API for access to BEACONING services. This module allows the registry of LMS students without saving or exposing sensitive information.

4.1.3 Learning Analytics Modules

- **Authentication Interface:** Provides an Authentication Interface that is able to convert a BEACONING Token into an LAS Token to request data.
- **LAS Controller:** This module provides endpoints to retrieve information of the user, like grades and statistics or even analytics if any exist.

4.1.4 LMS Database USER

It provides the link between an LMS user and a BEACONING user. This should store a Token forcore-services calls, a Refresh Token, in case the Token expires and an Analytics Token to make requests to the LAS.

4.1.5 CORE SERVICES DATABASE USER

In the case of LMS integration, the user data is private on the LMS and no user information is stored in the BEACONING database. The BEACONING core services provide the LMS user with an ID and its Token and also a Refresh Token. Since Core Services use OAuth2.0, this table already exists. The LMS user ID is not mandatory, since all the core-services only require the Token.

4.1.6 CORE SERVICES API CLIENTS

An already existing table in the core-services that has client ID and client secret for each component uses it. This client keys are used to retrieve the user Token and REFRESH TOKEN.

4.2 LMS Use Cases

Inside the LMS, there are two users: the teachers and the students.
The core integration of BEACONING with the LMS is to use the LMS students to play BEACONING games and use the grading they get for assessment.
To achieve this, a few conditions need to be met:

- The Student needs to be registered in the BEACONING. The registration of a student in the BEACONING ecosystem must be done by a teacher.
- The LMS needs a Token to access the Analytics of the student, stored in the BEACONING ecosystem. To get it, the LMS needs to use the BEACONING Token of the current user.
- The Student needs to have games available. A teacher needs to deploy games for the student to play.

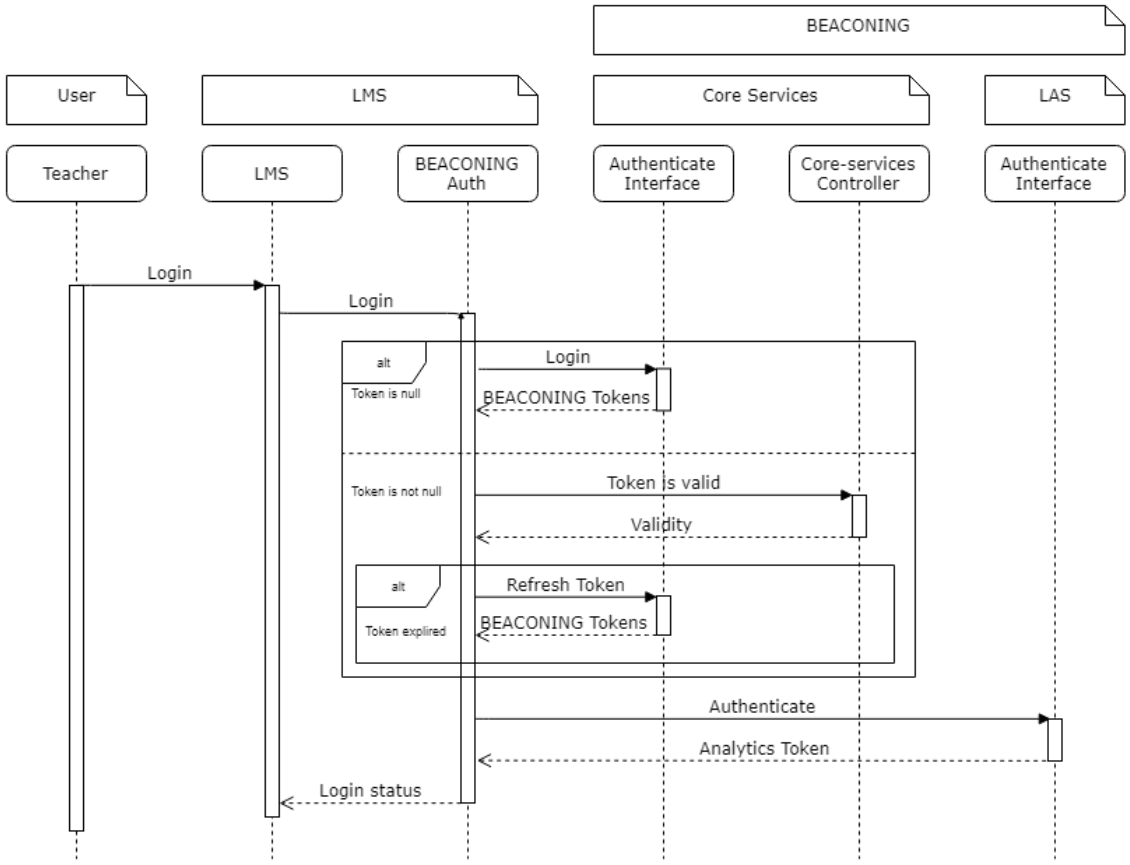
As noted in the points above, the teacher takes a primary role in this simple approach to integrate the LMS. Although he could do all these actions from the BEACONING's Teacher UI, these should also be available in the LMS.
In the next sub-sections, more detail about each user role and use cases will be detailed.

4.2.1 Teacher

The Teacher has the main roles to:

- Register students into the BEACONING ecosystem. This registration only sends an LMS ID. There is no data of the student exposed in the BEACONING, so in the case of a vulnerability in the BEACONING services, no data of the student is exposed.
- Create new GLPs. This creation is meant to improve the Teacher experience by not forcing him to use the Teacher UI.
- Edit GLP. A simple call to the BEACONING AT to edit a GLP.
- Deploy GLP. This deployment creates a new game that students can play.
- See Students' results. This call requires an Analytics Token, therefore the LMS must be able to retrieve it.

4.2.1.1 Authentication

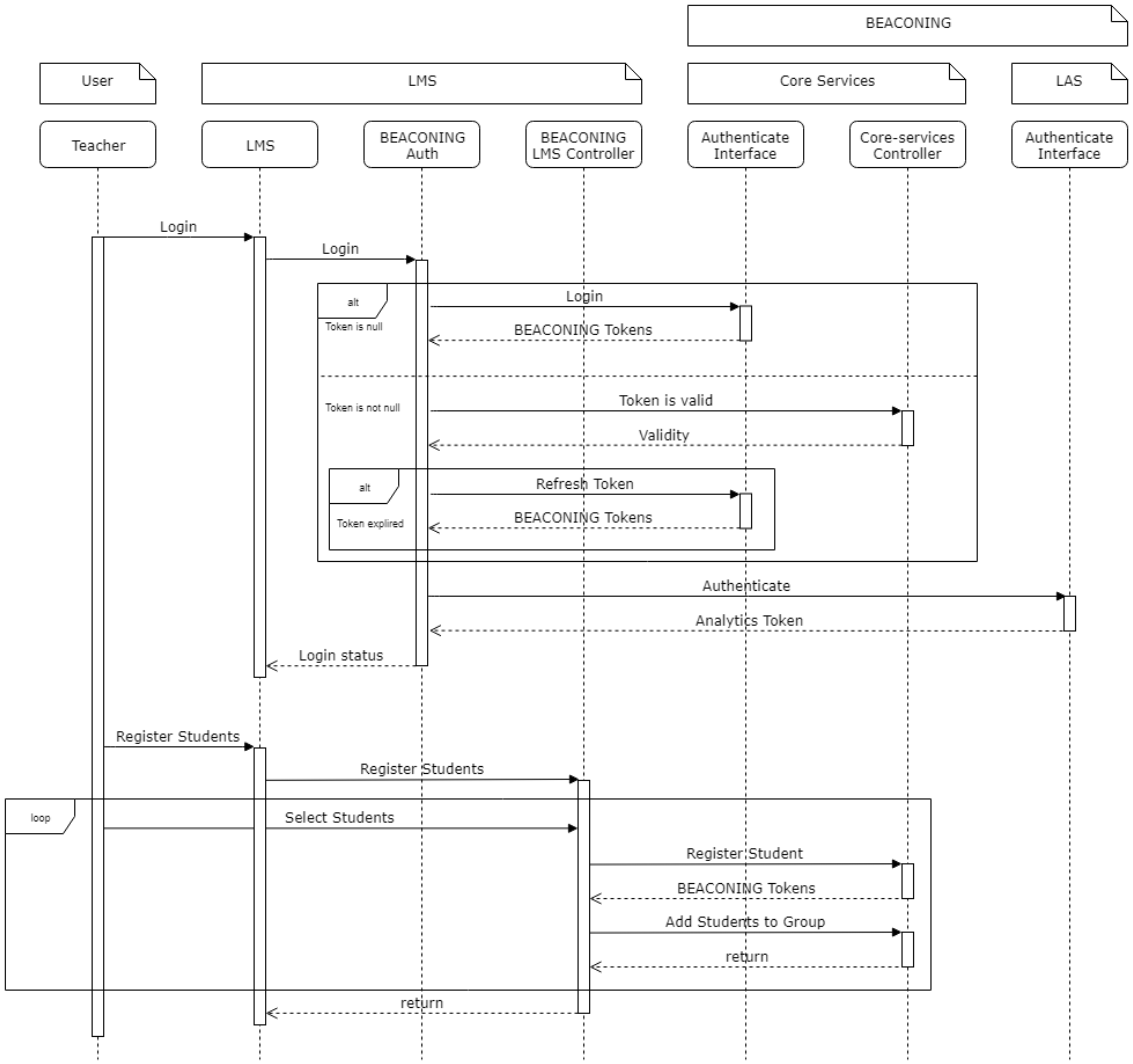


The teacher introduces its LMS credentials for a regular Login.

After the regular LMS Auth, the LMS plugin checks if the teacher has a BEACONING Token. If so, the plugin checks if the token is still valid and in case the validity check fails, it requests a new Token. If there isn't any token, the Teacher is requested to login into BEACONING platform, retrieving this way the Tokens and associating them with the current user.

After obtaining the BEACONING Token, an endpoint on LAS is used to retrieve the Analytics Token, used to make calls to the LAS API.

4.2.1.2 Register Students



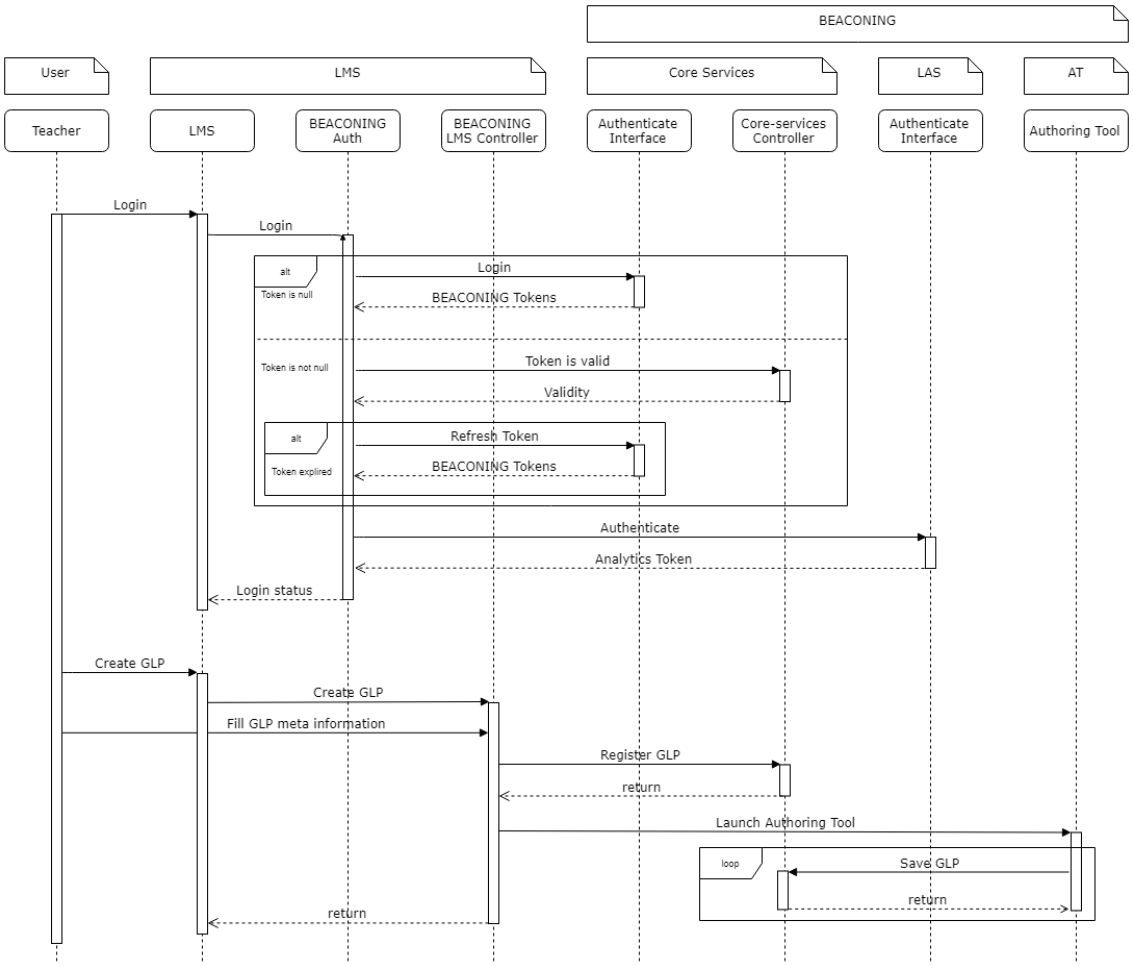
At any time, the Teacher can register students into the BEACONING platform to be able to deploy Gamified Lesson Paths to them.

To do this, the teacher selects the option in the LMS to register students and then selects the students he/she intends to register.

LMS will then register them in the core-services for later usage.

After adding the Students to the BEACONING ecosystem, the LMS adds them to a group with all the students of the current class.

4.2.1.3 Create GLP

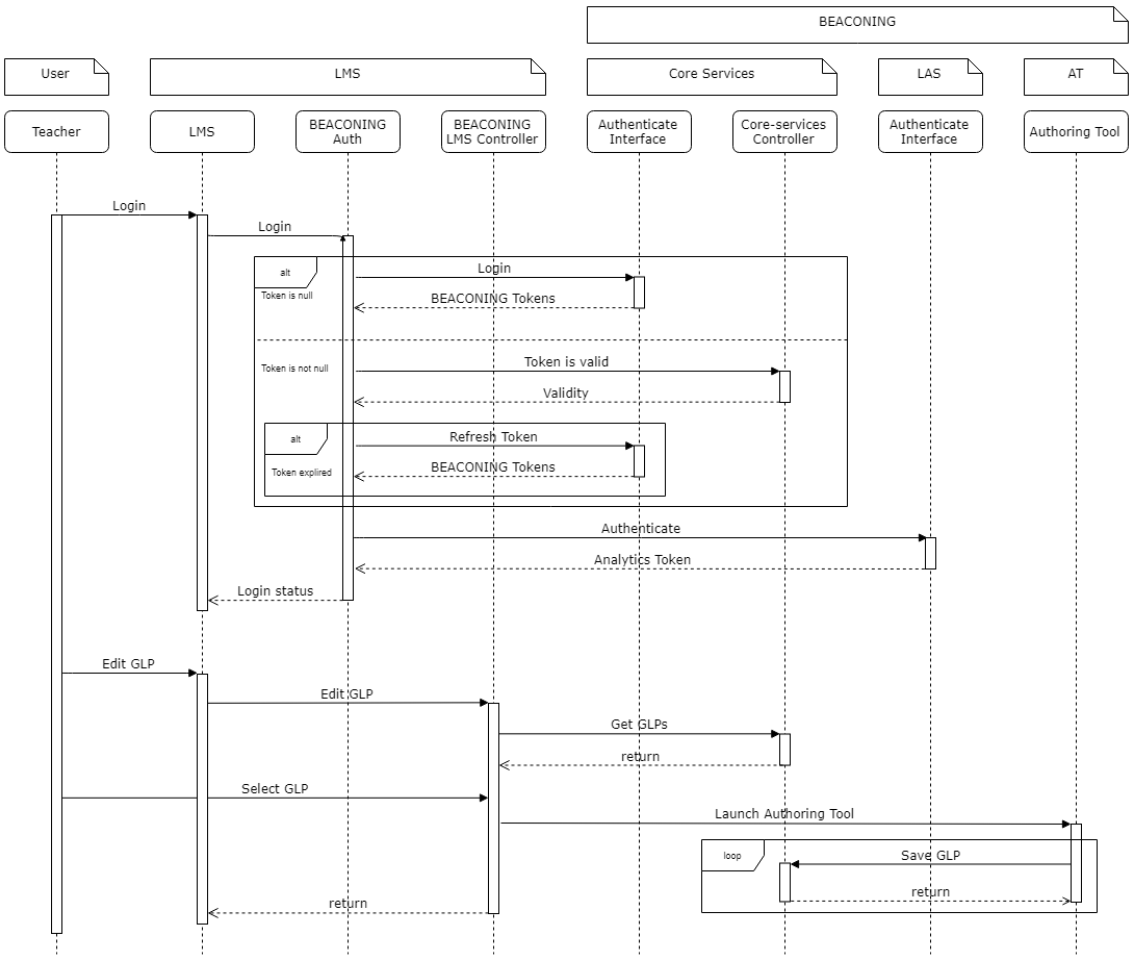


The first Teacher functionality is to be able to create a new GLP right at the LMS.

For this to happen, the Teacher must fill a form with all the GLP metadata.

After the LMS successfully registers the new GLP in the core-services, the Authoring Tool opens in a new window with the newly created GLP.

4.2.1.4 Edit GLP

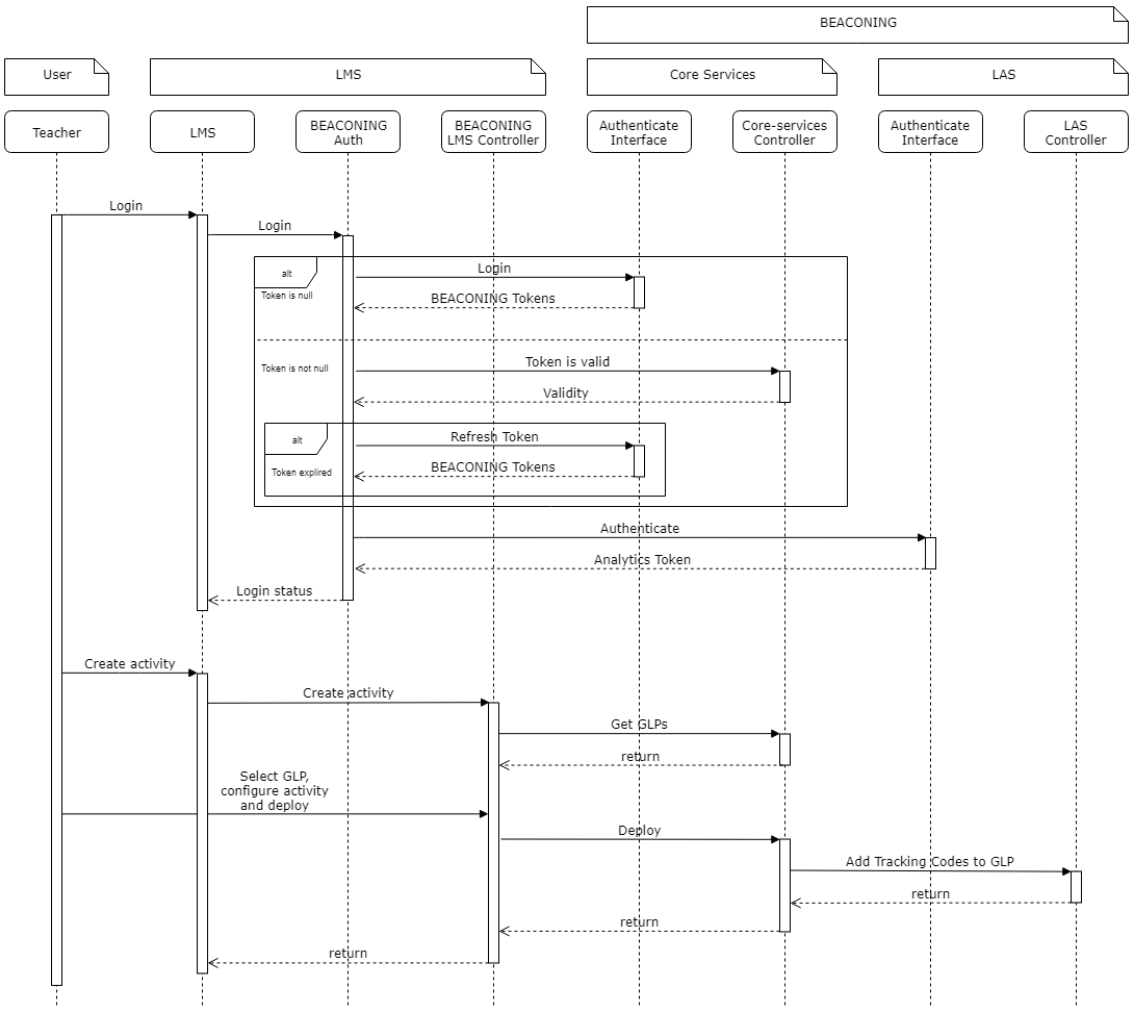


The teacher is also able to Edit any GLP he/she has access to.

LMS starts by requesting core-services for the GLPs the Teacher can access and then displays them.

Once the Teacher chooses the GLP, the LMS launches the Authoring Tool in a new window.

4.2.1.5 Deploy GLP



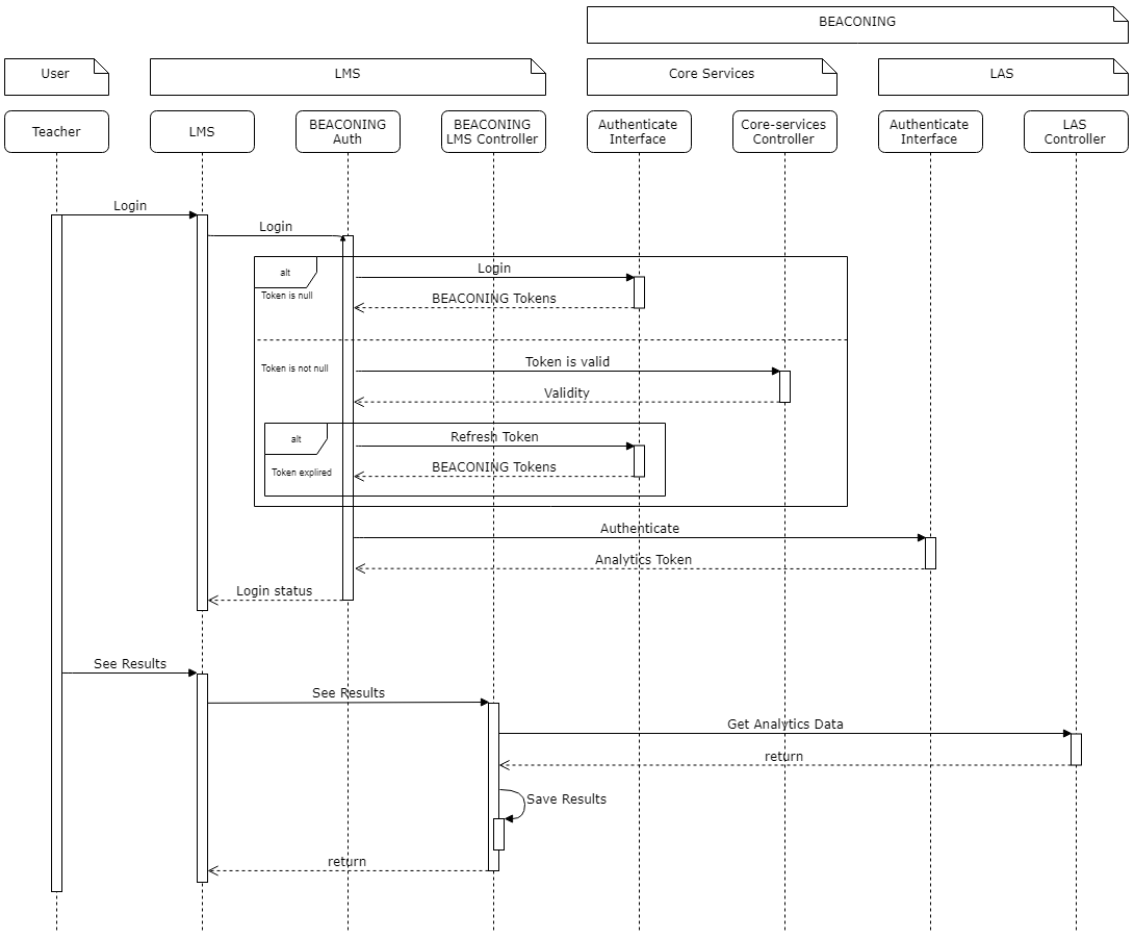
The most important functionality is to allow the Teacher to deploy a GLP to a group of LMS students.

For this to happen, all the students must be registered in the core-services and all of them need to belong to a single group. The Teacher must perform this prior to creating the activity for the students.

After this restriction is ensured, the Teacher can select the GLP and deploy it for the Students to play.

During deployment, the core-services send the GLP to the LAS to create tracking codes to register the student actions.

4.2.1.6 See Grades



The second most important feature is for the Teacher to see the results and current progression of the Students, at any time.

For the LMS to get the game analytics, it uses an ID returned by the core-services when deploying the GLP.

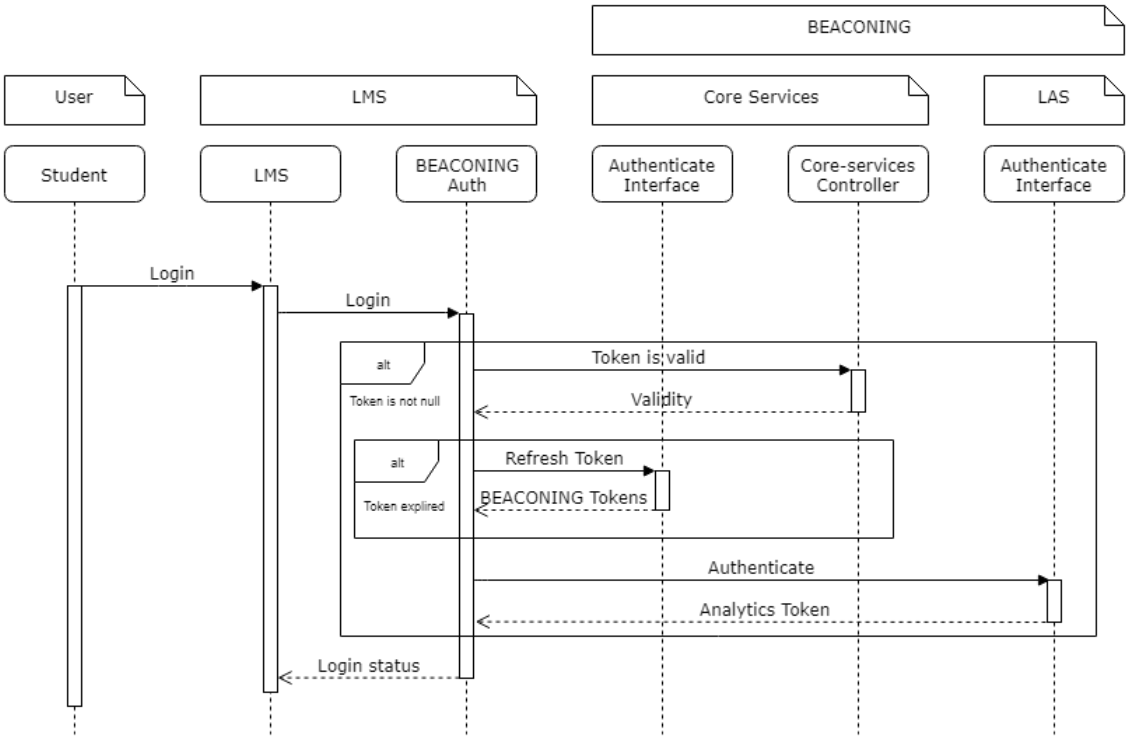
After requesting the analytics from the LAS, the LMS saves them for grading purposes.

4.2.2 Student

The Student is able to:

- See the list of available games.
- Launch a game through the LMS.
- See the results he obtained so far in the game.

4.2.2.1 Authenticate



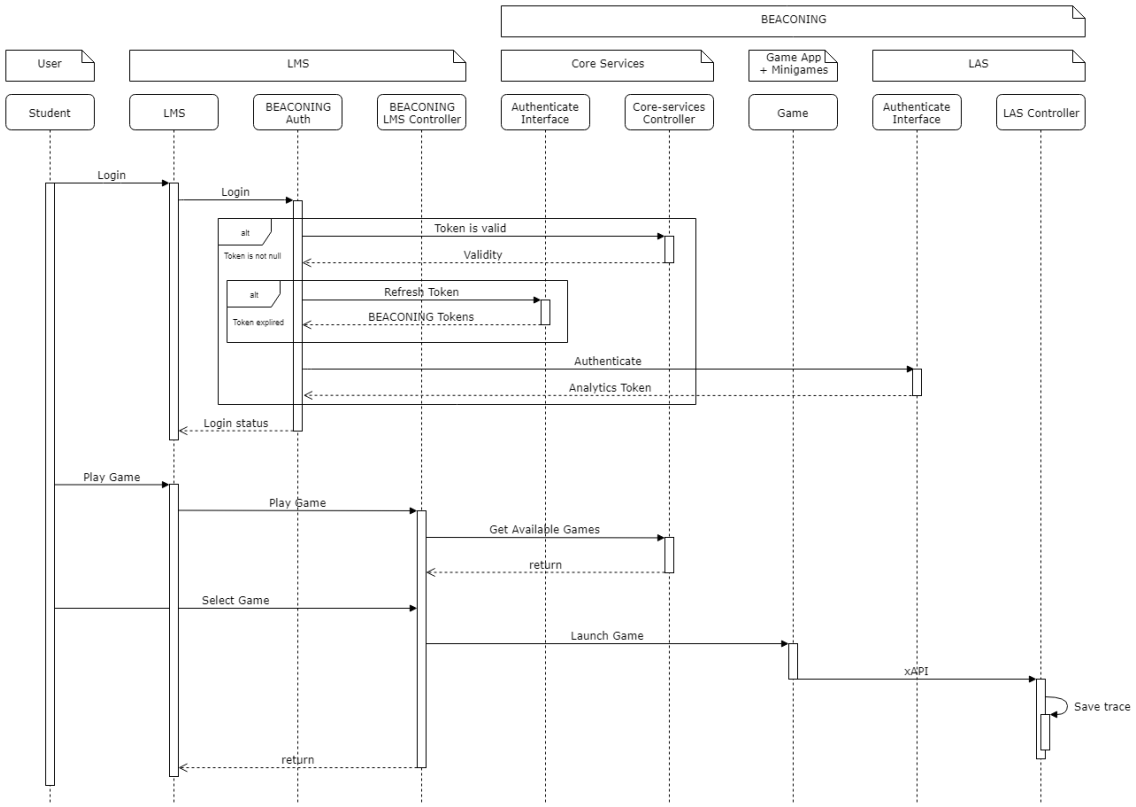
Through BEACONING Auth module, LMS authenticates the Student in the BEACONING ecosystem.

The student introduces his/her LMS credentials for a regular Login.

After the regular LMS Auth, the BEACONING Auth LMS plugin checks if the student has an associated BEACONING Token. If so, the plugin checks if the token is still valid and in case the validity check fails, it requests a new Token.

After obtaining the BEACONING Token, an endpoint on LAS is used to retrieve the Analytics

4.2.2.2 Play Game

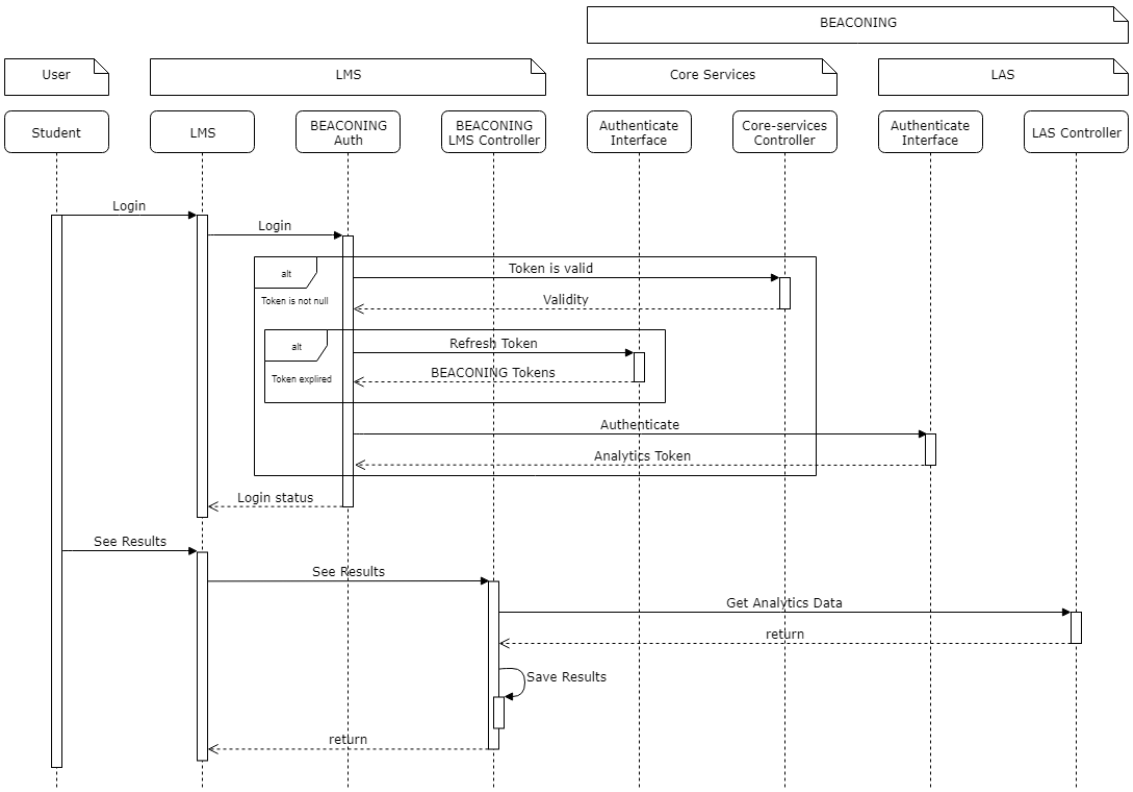


When a student wants to play a game, the LMS requests the available games from the core-services.

After the student selects the game, the LMS launches an URL present in the response from the core-services that runs the game. In the end, the student must manually return to the LMS.

During the game, LAS receives traces that are used to create statistics about the Student.

4.2.2.3 See Results



At any time, the Student can see his/her current state in the game and matching results.

To achieve this, the LMS asks the LAS for the current status and saves it for grading purposes