DESIGNING A CLOUD PLATFORM FOR INTERACTIVE GAME ACTIVITIES IN WEB-BASED E-LEARNING

A. Gradinaru, F. Moldoveanu, A. Moldoveanu
University Politehnica of Bucharest (ROMANIA)

Abstract
Games are one of the most common activities among young people and for a good reason: they are fun, entertaining and usually challenging. Recent advancements in computer graphics allow fast creation of computer games with stunning visuals and advanced interaction in rapid development environments such as Unity or Unreal Engine 4 editors. Still, these technologies are requiring lots of technological knowledge like graphics programming, networking, 3D modelling or animation. For this reason, educational or serious games are not frequently developed and used in the e-learning processes. Advanced applications are currently hard to build and publish, requiring serious investments of time, money and professional human resources.

This work describes a prototype system designed to help teachers to easily build, publish or integrate games or simulations into existing web-based Learning Management Systems. The applications can be developed using a simple drag and drop interactive editor and features a rich library of educational resources. The educator can start a project from templates like puzzles or 3D games and then use pre-made complex objects and interactions to design the game logic. The data is stored in a cloud platform and can be either published and directly accessed online in a web-browser or it can be integrated through an API into a web-based LMS for private or public multi-user classes.

Keywords: E-Learning, Serious Games, Cloud, LMS.

1 INTRODUCTION
Nowadays digital games are part of everyone’s life. Young children start playing games on mobile phones or tables even before they know how to say or to spell “game”. Therefore, high quality educational content in games is a major concern for parents and educators.

There are voices that are against using games in classrooms [1], the design of the game is limiting students’ actions because they come with predefined goals and course of actions, and they are usually losing the educational purpose – for example students will learn only how many mouse click they must make in order the get the best result.

Still, the main problem is the quality of the educational content in games used in classes, and not the games themselves. Games can be a useful tool to stimulate learning and teach in a more interactive an immersive way. If games have high-quality content and a game design that is focusing on education then it can help students to understand better, faster and in a fun way. It is scientifically proven that learning from fun activities is much more productive.

Games are complex pieces of software that are hard to make, and their development is clearly not accessible to educators, teachers and professors, that should be the main content creators and supervisors.

The paper describes the design of an innovative cloud platform aimed at helping teachers with non-technical skills to develop custom games or interactive activities and to use them easily in classes by integrating them with their preferred Learning Management System (LMS). The design of such a platform is complex and requires lots of testing and deep understanding of user experience aspects. The game building process must be easy to use and understand for any non-technical user to enable educators to create good educational content for their classes in any field of study. If any step is too complex or difficult to use then the platform won’t be used, and if it doesn’t allow for rich customizations, it won’t allow users to create useful content.

Starting from the design of one of the most popular toy company, Lego, we proposed a similar approach to the game building editor. Giving the users the basic building blocks, like Lego bricks, and a simple way to create free combinations of elements and interactions, could lead to a variety of
impressive creations. Therefore, we proposed a platform that relies on templates of pre-made content, that can be customized or combined in an intuitive user interface.

A prototype system that has been developed to test and validate the design is also presented in the Result chapter.

2 RELATED WORK

We propose a novel approach for the development of interactive game activities and their integration into existing popular LMS. Relevant publications show that the e-learning and VLEs (Virtual Learning Environments) are starting to be heavily researched and developed [2][3][4][5][6]. They present various solutions and experiments in using VLEs for online teaching and assessment. These environments are usually offered to students in dedicated environments and require additional login credentials or accounts. Also, it is a fact that development is still a serious problem, many research studies [7] trying to find a design solution that could simplify the process and to increase the ease of creating modern educational content.

There are many game editors like Unity [8] or Unreal Engine (UE) [9] that have rich marketplace resources and starter content, but the development process is still requiring lots of technical skills like graphics programming, graphical design, network knowledge and so on. Buildbox [10] is a recent example of a simplified game editor that gained popularity by offering pre-made objects, special effects and interactions, making it fairly simple to use even for non-technical users.

Also, there are a couple of Visual Programming Languages (VPL), like Scratch [11], that allow the creation and the logic scripting of simple games. Still, the language is restrictive and complex or multiplayer interactive applications are very hard or impossible to develop with it.

The deployment and the access to educational content is also an important issue. The DECAMP project demonstrates in several articles [12][13] that the use of VLEs can be integrated into existing LMSs, simplifying the access for students and providing a quick assessment method.

3 SOLUTION DESIGN AND ARCHITECTURE

The first step in designing our solution was to define an architecture that allows a straightforward editing access to modern interactive game applications, without the need to install dependencies or build systems, that can be used and distributed as educational content. Furthermore, it must allow for a seamless integration with popular LMSs so that the content is easily accessed by students and then graded or assessed automatically based on the player's performance.

In order to cover all the requirements, we concluded that a web-based cloud platform that can be accessed in a web-browser is the most suitable approach and has several major advantages:

- Ease of development: nowadays anyone has a modern browser that supports WebGL, so there are no additional installs or requirements
- Deployment: browser-based applications are multi-platform ready, so you can run your applications from a PC, a Mac, an Android phone etc.
- Ease of access: you only need to open a browser, create an account and log in
- Data is stored in the cloud: no need to worry about resources, backups or distributing the resulting games
- Collaboration: editing in a collaborative mode is very common, and this is possible only with online cloud apps

Fig. 1 presents the main components of the proposed cloud platform:

- Editor: create and edit game or interactive activities and publish them
- Game: store and distribute the published game data, and manages the play instances
- Player: web player for games, used as standalone and in the editor preview
- LMS plugin: a custom plugin that instantiate as an activity in a LMS and can play a game using the LMS local account data
We addressed the issue of simplifying the creation and configuration of game activities as much as possible. We faced the challenge of many available game types, each with their own particularities and variations. We created a relevant classification for educational purposes, starting with a major game type selection from Puzzle, Action, Role-Playing, Strategy, and Simulation. A design mockup for user interaction is shown in Fig. 2. Each major game type covers multiple genres or ramifications, such as 2D or 3D puzzles, First-Person or Third-Person RPGs, Real-time or Turn-based strategy games, etc. Each major type can also have multiplayer versions, or start from scratch with a blank base.

The next step is to select a starter template from a wide range of predefined fully-functional game activities that better define the result (Fig. 2). Selecting a template will auto-generate the game based on that initial template with ready-to-run content and interactions. This allows the user to play or test the result immediately after initial configuration, in contrast to the traditional game development process, where the first playable result is usually done after many hours of intensive programming. The templates are to be presented with a relevant name and a preview image displayed at selection, as shown in Fig. 3.
Selecting the template will define the playable game. Still, the resulting game application will require a main menu and a runtime user interface. To simplify this process, we proposed the use of predefined style templates, that can be applied to any game type or template. This is possible for two reasons: most of the main menu functionality is common to all the games, and the same style can be applied to all in-game user interface, usually comprised of buttons, windows or status bars. This will easily allow for automatic generation of all the menus and user interfaces. The template style will cover both basic elements in the game runtime such as score or progress, with some variations depending on the game type, and a basic main menu with user details, score, settings and game specific functionalities: save, load and start. The styles will have a clear and relevant name, alongside preview images, directly applied to the game template previously selected, if possible, as shown in Fig. 4.

Further on, the platform will allow adding multiple levels for the game, each level being edited individually. Each level will start from the selected base template, so the level will be guaranteed to run. This will allow educators to just apply changes to the templates and then to test them instantly.

The level editor was the most challenging part to design, because it must be interactive and simple to use for non-technical users and at the same time very complex to cover a large variety of games and to provide support for customization. The editor must allow educators to be creative and to easily develop inspiring and innovative games or activities to be used in classes in order to assess the students or to assist them in learning in an interactive and immersive way.
Many game editors nowadays use game objects as the base of their level design, a concept that abstracts the content found inside a game. A game object can be a controllable avatar, a tree, a source of light, a menu item and even a script. Game objects can also act as a parent group for other game objects. Each game object usually has a base type and can have additional attachments like meshes, textures, effects and scripts to extend its basic appearance or functionality.

In order to simplify the development process, we also proposed a system based on templates, so the educators can select from a wide range of predefined game objects with integrated functionality and interactions. Furthermore, each game object is basically a collection of attachment components as presented in Fig. 5. The components can be mesh objects, physics interactions, generation scripts, various special effects and many other. All these attachments can be simply added to an object by using a drag and drop user interaction.

![Game object configuration](image)

**Figure 5. Game object configuration.**

For example, a basic game object could have a component to make it scroll down continuously from the top, a component to make it responsive to collision events, and can have a component that defines a random value for it, that could be a random mesh, or a texture or a text label.

The components are also highly configurable, and educators can even create and save their own custom content. Fig. 6 presents the three types of component customization. The first type (a) is the simplest to use, and requires only visual configuration from a predefined set of options. For example, a “Get random value” component can have a mesh object selection from a list (like Meteors, Donuts etc.) and can have also a text label data (for example Kings of Old). Custom content can be uploaded to each of this lists. The second type of possible customization (b) is by using a visual scripting tool similar to the UE Blueprint. Functions are represented as nodes in an interactive diagram, and can be connected through wires in order to create the workflow of the component. This way, educators can define custom scripts even without much programming skills. The most technical customization type (c) is by using the JavaScript language to fully control the functions, the parameters and all the logic of the component.
Additional configurations can be added, for example to customize the multiplayer experience, to provide access control or to create time-based events.

Finally, after the game has been developed, polished and tested, users have the option to publish their game and access it online or deploy it into an LMS.

4 RESULTS

A prototype system was developed to validate the design. All the proposed main components were implemented with basic functionality. We used Laravel, a popular PHP framework, for the managing projects and users and to provide a RESTful API to communicate with other components. The game data is serialized in JSON (Javascript Object Notation) objects and distributed easily through the provided API. A Javascript player was built using the ThreeJS library, allowing to play the game in any modern browser on any device.

We also defined a game template for a “Galaxy shooter” game, that requires the player to shoot incoming asteroids that match the current mission request, for example a roman emperor as seen in Fig. 7.

One of the main challenges was to develop a real-time level editor in browser. We started from an open-source ThreeJS-based editor and added custom functionality to simplify the creation process accordingly to the proposed design. The editor has complex game objects and ready-to-use components that can be attached to game objects. At the moment, the components can only be added and edited as Javascript coded scripts. Fig. 8 shows the current state of the editor development.
The user can then save and publish the game, generating a unique id to easily access it in the online web player as seen in Fig. 9.

The next step was to integrate the resulting games into a LMS. We choose Moodle, an open-source web learning platform, that is very popular and provides easy extensibility through custom mods. Therefore, we developed a Moodle plugin, that allow educators to add game activities directly into their courses. The plugin requires an initial setup in order to authenticate and safely communicate with the game APIs, so the Moodle administrator must add a username and an API key generated from our cloud platform. After the setup, a game can be added as any other regular Moodle activities, by configuring a name for the activity and selecting an available published game as shown in Fig. 10.

As the user start to play, the Moodle plugin will send player data to the game server (name and email), in order to collect the current progress into his own account. After finishing a level, the progress is sent back to Moodle and updated in the gradebook automatically.
5 CONCLUSIONS AND FUTURE WORK

Nowadays there is a critical need of high-quality educational content in games. The presented solution for the development of educational interactive game activities enables non-technical users to easily develop content based on pre-made components and templates and integrate them into standard LMSs.

By making content creation available to teachers, and educational games available to students with one cloud platform, it has a real chance to greatly improve and revolutionize the educational process.

Future work will involve finishing the prototype system according to the proposed design and extending it to virtual reality and multiplayer social games that can assess and grade a team of students.

REFERENCES