Simplifying the Creation of Adventure Serious Games with Educational-Oriented Features

Pérez-Colado Víctor Manuel*, Pérez-Colado Iván José, Freire-Morán Manuel, Martínez-Ortiz Iván and Fernández-Manjón Baltasar

Software Engineering and Artificial Intelligence Department, Faculty of Informatics, Complutense University of Madrid, Spain // victormp@ucm.es // ivanjper@ucm.es // manuel.freire@fdi.ucm.es // imartinez@fdi.ucm.es // balta@fdi.ucm.es

*Corresponding author

ABSTRACT: Developer-friendly professional authoring tools have greatly simplified entertainment videogame development. However, this simplification had a limited impact on serious games, which require the active collaboration of developers with educators and other stakeholders. To address this issue, we present uAdventure, an easy-to-use game development environment for narrative "point-and-click" graphic adventure games. uAdventure started as a re-implementation of the previously validated eAdventure authoring tool, now built on top of the Unity game platform. The idea is to enjoy the advantages of the Unity professional environment while reducing its complexity to allow non-developers to author serious games. uAdventure is designed to simplify the creation of serious games that include educational-oriented capabilities such as learning analytics without requiring programming knowledge and has been formatively tested with different types of users and in different settings. Initial testing was carried out with a group of heterogeneous users with different computer usage profiles in a vocational environment. In the second round of testing, it was used by students of two different university courses to develop serious games that include educational features such as location-based mechanics and learning analytics. The results of these formative evaluations show that uAdventure can be used as a serious game teaching tool and that it simplifies the creation of serious games with educational features by non-expert authors.

Keywords: Serious game development, Authoring tool, Narrative games, Learning analytics, Location-based games

1. Introduction

Serious Games (SGs) are videogames where the main purpose is not entertainment, with education as one of the most notable examples. There are multiple benefits for students when it comes to learning through SGs and gamified experiences (De Freitas, 2018), among which an increased engagement is especially important. However, although multiple initiatives have demonstrated a positive impact on both educational institutions and companies, SGs are not yet widely used in education (Almeida & Simoes, 2019). The difficulty of formally proving their benefits and return of investment; and to the need to involve other stakeholders (e.g., educators, domain experts) with key responsibilities in their development process are two of the main reasons hindering more widespread adoption of SGs in educational institutions.

Nowadays, there is wide range of tools to create serious games from professional platforms to research tools. At the professional level, we can find all-purpose but complex game platforms such as Unity, or more narrowly-scoped and consequently simpler platforms such as Game Maker Studio. Certain commercial game authoring tools are even accessible to non-programmers, including GameSalad, Genie, or The Training Arcade. Besides, there are also tools created as a result of research projects that were specifically designed as authoring tools for SGs. Some of them are StoryTec (Göbel, Salvatore, Konrad, & Mehm, 2008) that is a story-telling authoring tool or Emergo (Nadolski et al., 2008) that is a scenario-based authoring tool.

The use of genre-specific game engines such as Adventure Game Studio can greatly simplify game development and thus SG development (del Blanco et al., 2012; Marchiori et al., 2012), making their creation accessible for non-programmers and simplifying the participation of non-technical SG stakeholders, among which teachers are particularly important. This paper describes uAdventure (uA), an SG authoring tool implemented on top of the Unity game environment, designed to simplify the creation of serious games that include educational-oriented capabilities such as learning analytics without requiring extensive programming knowledge.

To simplify the SG development process for non-experts, uA focuses on facilitating a high-level authoring metaphor, instead of overwhelming authors with the highly flexible but complex functionalities offered by all-purpose platforms. We achieve this by specializing in the adventure game genre. Specific game genres support different types of learning (Aust, Nitsche, & Pelka, 2014). For example, game-like simulators allow authentic learning through recreating real-life situations (Center for Technology Implementation in Education, 2014), but the specificity and complexity of their mechanics can be expensive to develop and will usually require extensive ad-hoc programming. We chose to focus uA on authoring only "point-and-click" adventure SGs because the genre provides a good balance between power and complexity for creating educational games, which combine a strong narrative with puzzles and problem solving, promoting learning and reflection (Dickey, 2006). After testing uA with different users and in different settings, we consider that uA can be used as a serious game authoring and teaching tool. Additionally, uA also allows non-expert authors to add educational-oriented features, such as analytics, to the SGs that they develop.

The rest of the paper is structured as follows: we first introduce narrative adventure games and eAdventure (eA) the predecessor of uA and its limitations. Then, we introduce uA features that both address eA's limitations and further simplify adventure SG development for non-experts. Finally, we detail the experiments used to validate and guide the development of uA, discuss the results, and present our conclusions and future work.

2. Narrative adventure games: From eAdventure to uAdventure

Adventure games are story-driven (Rollings & Ernest, 2003) and inherit their main elements from the narrative metaphor (Marchiori et al., 2012). The main elements of adventure games are (Dickey, 2006): *characters*, which are the entities that populate the story and perform the different actions; *objects* that the player must collect or interact with; *scenes*, where the action occurs; and finally, the use of *narrative* and *actions* to encompass narration, conversations, and interactions of characters with their environment. In adventure games, the player advances through the story by exploration and solving logic puzzles, normally involving characters or items (Amory, 2001). These features are especially useful in educational games, as players need to reflect on their knowledge to achieve their in-game goals.

Game stories can be *linear* or *non-linear*. Linear stories follow a strict sequence, while non-linear stories can result in different states (endings, scores, etc.) depending on the players' choices, allowing them to access a whole "sequence of possibilities considering changing world states" (Spierling, 2009). Non-linearity is ideal for SGs, as players feel that their actions have meaning and understand that they must investigate and use deduction to reach the desired ending. Non-linearity is also very useful from an analytics perspective, as game states, player responses to in-game questions, and past problem-solving actions can be analyzed and used for assessment.

The eAdventure (eA) SG platform (**Error! Reference source not found.**) is a previous authoring tool for the creation of "point-and-click" games (Torrente, del Blanco, Marchiori, Moreno-Ger, & Fernandez-Manjon, 2010). The key goals of eA were:

- To provide a multiplatform java-based adventure game authoring tool for non-programmers, and even accessible to users without high levels of computer literacy.
- To allow authors to package games as educational learning objects using different educational standards such as IMS Content Packaging usable in Learning Management Systems (LMS) such as Moodle.
- To allow authors to create games as an assessment tool. eA's authoring tool included a simple assessment tool based on flags and variables and a reporting tool based on the (limited) capabilities provided by SCORM standard (del Blanco, Marchiori, Torrente, Martínez-Ortiz, & Fernández-Manjón, 2013).

At the time eA was created, game engines and authoring tool licenses were expensive or did not provide then-current features. This situation made a Java-based free and open-source project such as eA a good fit for educators. At the time, applets allowed easy packaging and distribution of SGs using any web-based Learning Management System such as Moodle. The java technology multiplatform approach was initially adequate for the diversity in operating systems and devices, but the fast change in mobile technology made eA maintenance a challenge. Thus, relying on commercial game engines, such as Unity, as a middleware for an SG development tool, ensures cross-platform compatibility and ongoing support as the platforms evolve, and can greatly reduce development and maintenance costs. However, using Unity alone requires significant programming skills, making it inaccessible to non-experts.

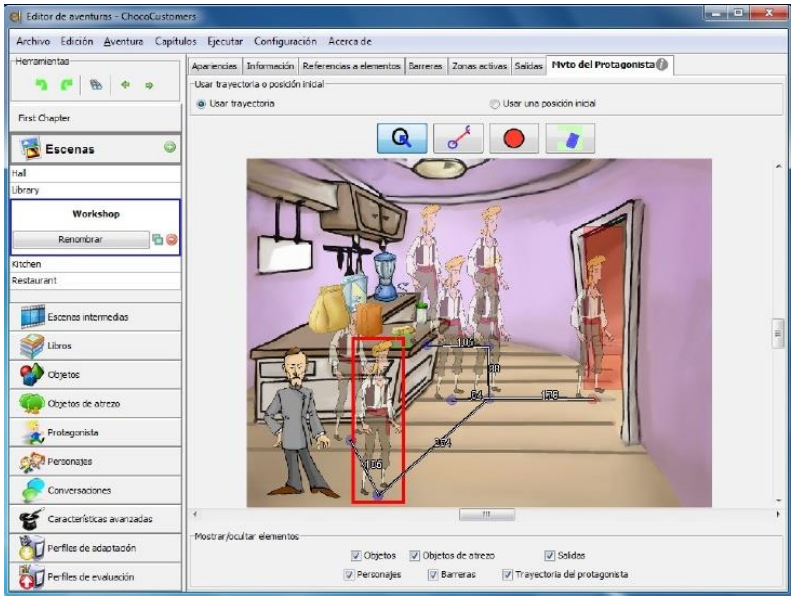

Figure 1. The scene editor in the eAdventure serious game authoring tool

To increase the added value of SGs, and to support their use as assessment tools, we consider that evaluation should be embedded as part of the game in a non-intrusive manner, and if possible, based on the evaluation of players' in-game actions, as opposed to requiring players to answer intrusive post-tests. Assessment capabilities were already included in eA, based on author-specified flags and variables that represented milestones or important events. This kind of assessment was fixed at authoring time, and affected by the limitations of the ADL SCORM data model and of the LMSs that hosted the games.

Since eA was developed, the use of Learning Analytics (LA) has become much more widespread. LA information is usually collected via *traces*, where each trace represents an event produced during the interaction with the educational tool. ADL eXperience API (xAPI) is a new LA specification designed to address the limitations of previous standards such as SCORM, providing an open common structure and an extensible data model that can be adapted by different communities of practice; for example, there are application profiles for SGs, videos, and other media. The inclusion of xAPI can provide more flexible and advanced analytics for SGs, which, for example, need not be completely fixed at game creation and can instead be improved based on experience.

Moreover, technology-consuming behavior has changed, transitioning from the use of single to multiple devices, and from the preference of using PCs and laptops to mobile devices such as tablets and smartphones. Mobile devices, in particular, are preferred by children (Lauricella, Cingel, Blackwell, Wartella, & Conway, 2014) – and also include new sensors and capabilities that can be of value to SGs. In particular, location-based (LB) games are a popular trend in both commercial (Laato, Rauti, Pietarinen, & Laine, 2019) and educational (Xanthopoulos & Xinogalos, 2018) sectors. In terms of game mechanics, location-based games use real-world locations as game elements both outdoors (using the GPS) and indoors (using QRs codes or beacons). From an educational perspective, location-based SGs can offer new learning experiences such as gymkhanas or historical tours that require interacting with the real world to progress in the game.

3. uAdventure

uAdventure (uA) is a SG framework built on top of Unity that reuses our previous experience with eA to simplify the creation of narrative "point-and-click" SGs by non-experts. We decided to build uA on top of Unity to solve eA's major technical problems: uA will always support the latest technologies and platforms, as long as Unity keeps being updated to support them. Reusing most of Unity's subsystems allows us to focus the development on educationally-oriented features that make uA unique, making it more maintainable and less likely to become obsolete (Perez-Colado, Perez-Colado, Martinez-Ortiz, Freire-Moran, & Fernandez-Manjon, 2017a). Still, uA maintains the

interoperability spirit of eA and facilitates the transition between tools. We also developed an eA-to-uA conversion tool that can convert old eA games into equivalent uA versions.

The design of uA is based on emphasizing narrative game elements (Salter, 2014) and was mainly adopted from eA: scenes, characters, items, and cutscenes. Scenes, where the game action takes place, contain all elements (interactive or not) that the players will explore and interact with. Moreover, scene definitions include areas of interest and walkable zones, and scenes can connect through areas of interest, as the basic means to build the SG story. Characters (representing the player or other important characters of the game's plot) may, among other actions, converse, move, and receive objects. Items are passive elements that can only be examined, and occasionally picked up and manipulated. Finally, cutscenes can be used to introduce linear non-interactive narrative sequences, such as to provide necessary exposition when players enter a scene for the first time or fulfill certain conditions.

The uA narrative model includes a built-in set of actions, which, once triggered, can change the game state in both linear and non-linear stories. For example, items in the scene can be often be examined, grabbed, or used; and depending on each action's configuration, this can lead to different game states, endings, or story branches. Instead of scripting, uA provides high-level effects to easily manipulate the game context and feedback displayed to the player. Since effects have a high-level meaning and are documented within the interface, this makes it easier to trace the flow of the story, especially when compared with conventional code scripting.

The conversation system provides a tree-based editor (**Error! Reference source not found.**) to manipulate the different narrative branches that a non-linear story can require. The editor is geared towards the agile edition of conversations, as it displays all the narration in one view as a node graph that can be easily modified and extended. The conversation system also manages all the runtime tasks required to display the conversation, such as displaying dialog bubbles for characters, changing their animations, and displaying the different available responses for players to choose from.

The uA scene editor (**Error! Reference source not found.**) allows authors to create scenes and manages their elements, including characters and items. Although Unity supports 3D, we have chosen to limit uA to 2D because it is more common in adventure games, and much easier to author and to navigate for players. Additionally, uA's scene editor allows scene elements to be managed directly, without ever having to handle their associated components and behavior scripts as done in Unity's scenario editor. For example, uA handles scene sizes, boundaries, camera configurations, and transitions automatically. The uA scene editor provides a much more understandable overview of what will be displayed in a scene, how it can be navigated (in third-person games where player-controlled characters move around the scene), and how it can be interacted with, including characters, items, and areas that act as exits or barriers.

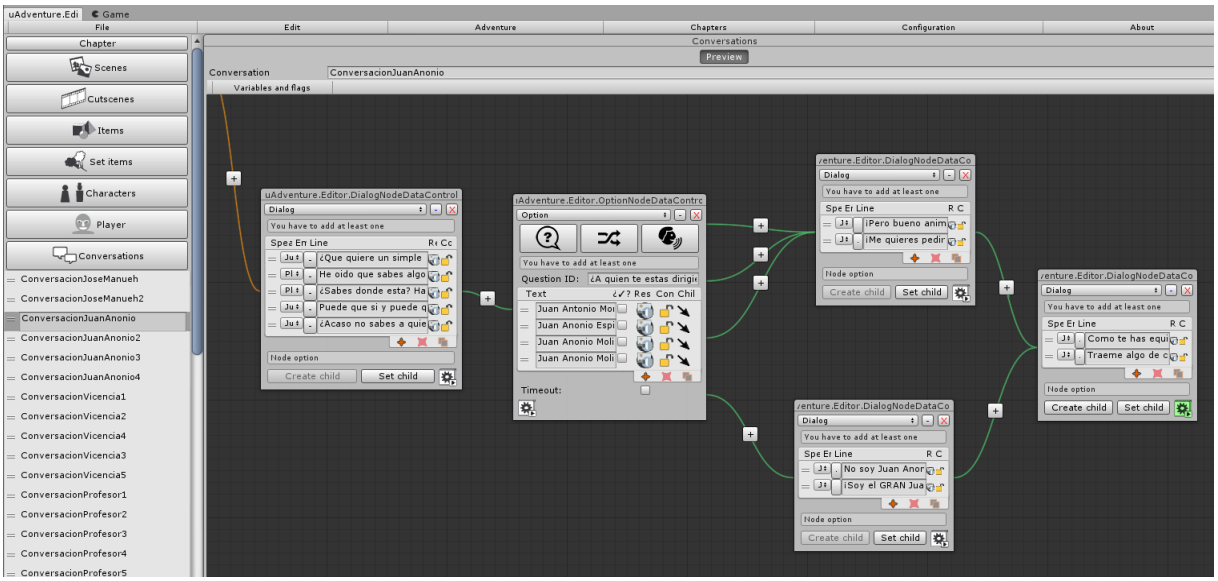

Figure 2. uAdventure's narrative conversation editor

A new element inspector in uA simplifies element modification: instead of having to navigate to the underlying elements themselves to view or change their properties, the inspector allows authors to directly access them in the scene editor without navigating away from the scene itself. The design of this inspector (**Error! Reference source not found.**) is based on the Unity inspector, making it more affordable to users that are already familiar with Unity. We expect it to be useful in three distinct scenarios: where development teams include both educational experts and Unity experts, the latter of which will often have to switch between the different perspectives; when Unity users want to use uA as a prototyping tool; and when uA users want to step into Unity to achieve more complex results. In any case, authors can switch between the uA and Unity interfaces at any time in just a couple of clicks.

To simplify SG content creation by non-experts, uA provides its editors, replacing Unity's default interface with SG-relevant views such as a model editor, context variables and flags, a gameplay window, and a project file explorer. uA also handles some of Unity's default functionalities. For example, when the "play" button is pressed to test the game, the corresponding uA scene is loaded transparently, without forcing users to deal with Unity scenes at all. Conversely, when the game stops, the uA window is reopened. Saving and loading projects work similarly. Also, uA uses the asset importing pipeline to automatically handle asset configurations when users copy assets into their project folders, allowing authors to ignore the technicalities involved in configuring each asset type.

Supporting multiple platforms requires significant tuning to achieve good performance in each of them, for example, in terms of quality or lighting; and is, therefore, a multi-step, knowledge-intensive process in Unity. For uA, we have significative simplified the generation of executable distributions for each of the target platforms by providing a simplified builder. This builder asks authors to select the platforms where the games are to be played, chooses sensible default values, and creates executable versions of games for each selected platform.

3.1. uA extensibility

Unity, and game engines in general, rapidly evolve as new trends and technologies emerge. To leverage this evolution, uA is easily extensible, allowing the creation of new modules to add support for features ranging from low-level and technical to high-level and educational. uA added two additional educational capabilities to the core features adapted from eA: learning analytics and location-based games. Both features are described in the following sections.

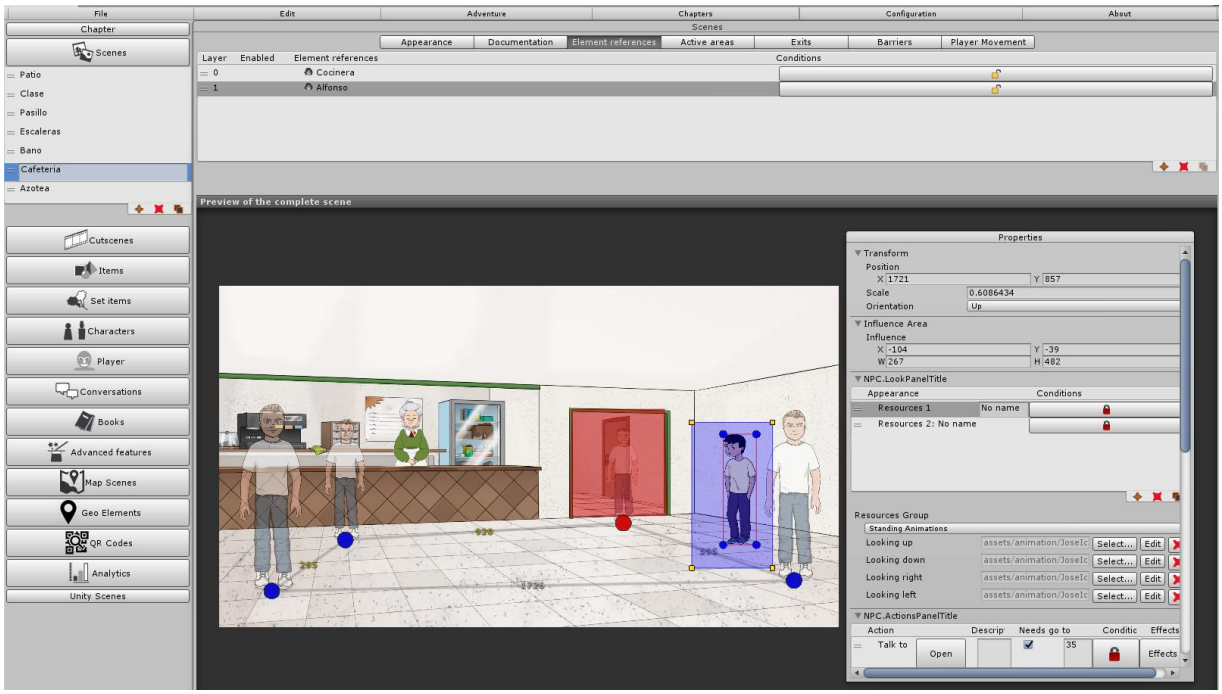

Figure 2. uAdventure' simplified editor for a third-person game, showing several characters in a scene and the Unity-like inspector at the right

3.1.1. Learning analytics extension

The Learning Analytics (LA) educational extension included in uA is far more flexible and powerful eA's assessment system, which was based on the SCORM specification. Using xAPI standards-compliant LA allows us to consider all in-game user actions and to represent in-game relevant facts as analytics data traces. To simplify the application of LA, uA allows easy authoring of a Game Learning Analytics model (Freire, Serrano-Laguna, & Iglesias, 2016), translating in-game events into analytics data transparently. In our implementation, we use the xAPI Serious Games (xAPI-SG) Profile (Serrano-Laguna et al., 2017), which is a close fit for our game model. Since xAPI is event-based, an xAPI-compliant server can gather the different events and states generated while playing through non-linear stories to later reconstruct the entire gameplay. During analysis, traces that provide evidence of in-game actions related to learning objectives are essential to evaluate player knowledge of those objectives. For this reason, authors that wish to use LA correctly must first identify such learning objective-relevant traces in the game and ensure that they are correctly reported via game-process traces to later inform the analysis process.

By having out-of-the-box support for the xAPI-SG profile, and not forcing authors to deal with low-level analytics details directly, uA greatly simplifies the use of LA for novice authors. Most major uA elements, such as scenes or conversations, can generate xAPI statements automatically. To measure game progress and identify educationally-relevant in-game events, we implement *completables*, as contructs that are configurable in a high-level editor (Figure 3) based on milestones. The milestones link in-game element interactions or game-status changes to progress reports for completables; and configure the score calculation based on current game variables or on previous scores.

Analytics data generated from uA games are handled by a *tracker* component embedded into games; and can be sent to an external server and stored locally for later retrieval, an option which is useful when there is no network or network disruptions are to be expected. Consequently, uA includes a section to configure tracker settings, covering both the format used to report the traces and the reporting method, including offline storage, LRS settings, and credentials, or both.

3.1.2. Location-based (LB) games extensions

Two extensions have been developed for both outdoors and indoors positioning: GPS and QR. Both extensions are part of a Location-Based (LB) game mechanics pack (Pérez-Colado, Pérez-Colado, Martínez-Ortiz, Freire-Morán, & Fernández-Manjón, 2017b) to be used in mobile environments. The GPS extension includes functionality for outdoor positioning and a new type of map-based scene, mixing both physical elements (e.g., points-of-interest or regions) and adventure elements (e.g., characters). These elements include a new set of actions for physical areas such as entering, exiting, looking towards certain points (using the mobile compass), or in certain directions. Indoors positioning is handled by an integrated QR code and scanner system. Among other functionalities, the extension includes navigation to guide the player in the scenes and the possibility to trigger the opening of uA scenes based on player location. Lastly, to simplify the development process, we have implemented a location-setting debug window to manage and simulate player location from the editor, allowing games to be tested locally.

To guarantee a good player experience in location-based games, we have identified several best practices. First, a companion-character figure should be placed in the map scenes to assist players (e.g., when they get lost or forget their pending tasks), using conversational menus and setting up navigation if needed. Second, we always recommend using moderate distances in navigation in order not to lose players' interest in the game. This aspect was already pointed out by players of our first location-based pilots (Pérez-Colado et al., 2017b). Another recommendation to keep players' interest is to enhance paths by adding enemies (such as zombies) and/or treats (such as extra points). Lastly, we always recommend using zoned uA scenes to represent indoor environments – where QR codes might, if needed, be placed.

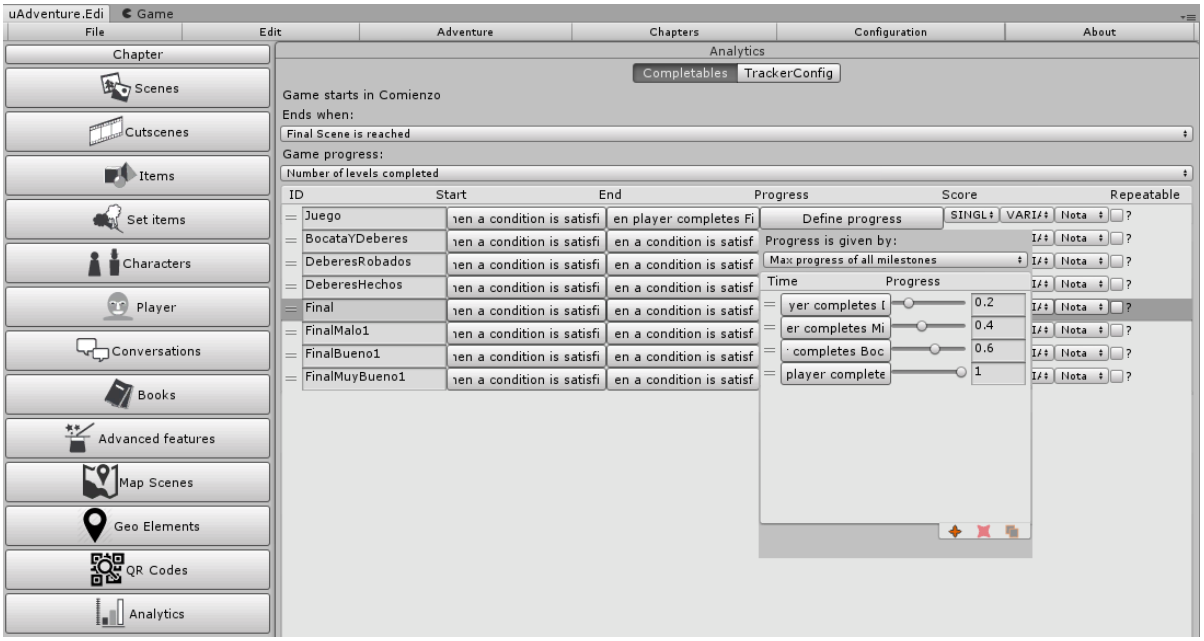

Figure 3. Completables editor in uAdventure's learning analytics extension, showing a completable's progress editor open with several completable steps

4. Testing uAdventure in different settings

To study the extent to which uA's goals have been achieved, we have conducted several tests. The goals of these tests are two-fold: to measure the degree to which uA allows non-experts to author SGs; and to measure and improve software quality towards a full uA 1.0 release, including both the LA and LB extensions. Additionally, these experiments should be understood as pilots guiding the development of uA itself: after each round of testing, we used the results to identify and address bugs and to improve the user experience for subsequent versions.

Testing of uA has been performed in an iterative process divided into two main stages: (i) pilots, testing uA using a small and controlled group of authors with varied profiles, and (ii) use uA in actual undergraduate and graduate courses. The pilot testing helped us to fix bugs and identify possible improvements in uA, while the courses used uA as a teaching tool to introduce different SG aspects, and allowed us to study how users approach SG creation that satisfied different design requirements such as achieving certain learning objectives, using branching and multiple endings, or using geo-localization features. Figure 4 depicts the relationship between the different experiments carried out and the uA features under test.

4.1. uAdventure pilots

We performed a preliminary evaluation with users from different profiles, including both non-technical users (e.g., teachers and artists) and programmers. The purpose of these pilots was to find usability problems in uA and guide the development of uA and its supporting materials (e.g., user's manual); and also to verify that its users were capable of: (i) developing narrative adventure game examples, (ii) create non-linear stories with multiple endings, (iii) develop their own stories autonomously and (iv) build actual games.

Participants with no prior experience with uA were provided with uA's user manual and uA itself. The manual introduced the main SG narrative elements, concepts, and features of uA. Then, participants were asked to implement the eleven self-guided examples contained in the manual by themselves. By completing all examples, participants created a simple non-linear story with multiple endings. Finally, participants were also encouraged to develop their own stories to test high-level actions not included in the examples and to build and test their games on the Windows and Android platforms.

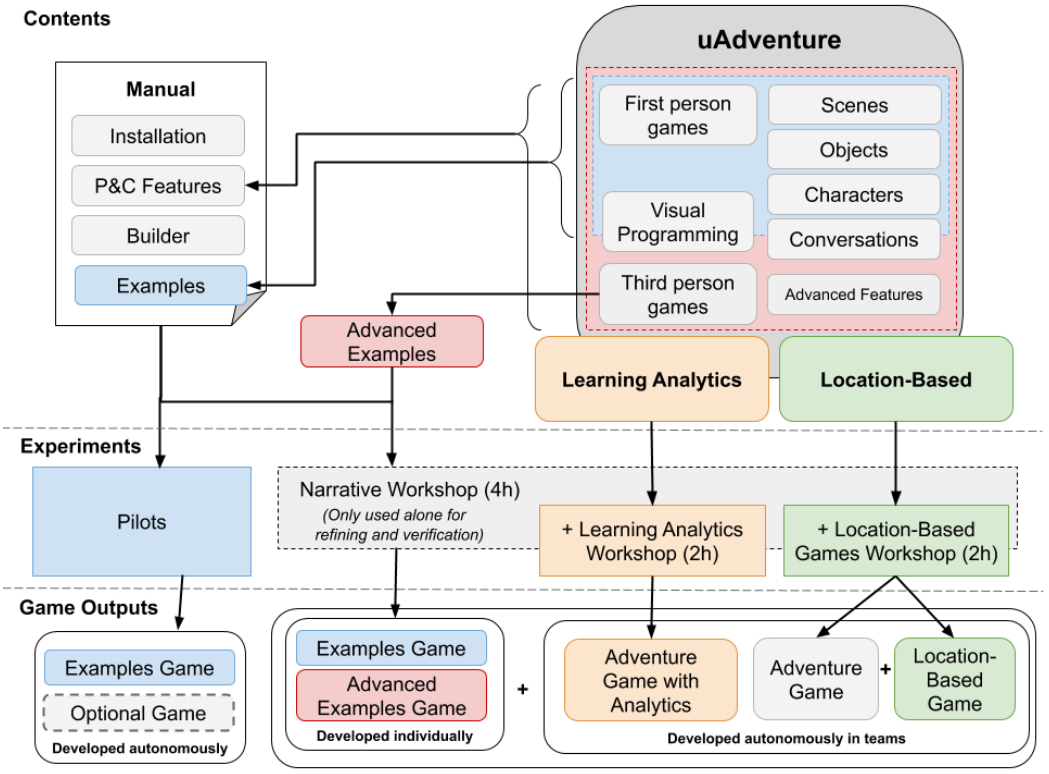

Figure 4. Relationship between different experiments (mid), uA features (top), and outputs (bottom)

After the experiment, a questionnaire was used to build profiles of participants, and to measure the perceived difficulty of each task that they performed into four levels: easy, normal, difficult, or not-attempted. In total, 110 simple high-level user actions (e.g., "create a scene") were identified to test the main uA functionality (of which 59 were covered in the examples). The profiles questionnaire consisted of a self-classification as a non-technical user, artist, or programmer; and a set of questions regarding the participant's experience creating stories or narrative games to estimate their "narrative experience" (see Table 1). Finally, participants were also asked about their success and questions while following the examples, their goals, and the degree to which the manual helped them through the learning process.

We first grouped the results by user profiles depending on their answers regarding computer use: artists used the computer mainly for image processing or 3D software, programmers mainly for programming, and the remaining users were classified as non-technical. Then we calculated the average success of the goals and high-level actions. Table 1 shows participant backgrounds, their level of understanding of the manual and uA's main concepts, and the number of goals and high-level actions they completed. The last section analyzes task complexity and hence, uA's success in terms of interface simplification.

According to Table 1, usability was high: participants finished 95% of the examples successfully, which was corroborated by the 87,4% of the examples actions completed; the percentage of incomplete or "found hard" high-level actions is correspondingly low, and most high-level actions were rated as simple (~65%). Individually, artists had the lowest success in completing examples: they had 15% more untried high-level actions than the global average. Regarding the goals of allowing users to create non-linear stories and author multiple endings, the results show an average success rate of 70%.

All users reported finding the manual helpful. However, they also reported multiple difficulties using uA. For non-technical users, doubts regarding uA's model were probably related to their lack of narrative experience. Programmers also reported similar questions. On the other hand, questions regarding effects and conditions are, as expected, more frequent for users with no programming background. Finally, another relation is present between the questions asked by users and how complex they reported to have found the high-level actions. For example, non-technical computer users only reported ~26% of completed actions as simple, compared to 79-72% for artists and

programmers. Finally, participants also reported 29 different issues, misconceptions, and difficulties, of which 27 were considered to have merit and fixed in a subsequent release.

Table 1. Summary of results for different profiles (non-technical, artists and programmers)

	Participants by category			All
	Non-tech.	Artist	Programmer	
User profile				
Number of users	2	2	6	10
Narrative experience (%) [a]	50	100	78	77
Experience with SGs (%) [b]	80	80	70	74
Manual				
Read entire	1	1	4	6
Found helpful	2	2	6	10
Asked questions on model elements	1	0	1	2
Asked questions on effects and conditions	1	1	2	4
Goals				
Examples completed (%, n = 11)	95	86	97	95
Non-linear story created	2	2	3	7
Multiple endings?	2	2	3	7
Have built the game	1	1	2	4
High-level actions (110 in total, in %) [c]				
Completed	65.4	51.3	74.4	68.0
… of which found simple	25.7	79.8	72.1	64.5
… of which found normal	74.3	18.4	26.9	34.6
… of which found hard	0.0	1.7	1.0	0.9
… actions covered in the examples	85.6	72.6	92.9	87.4
… actions out of the examples	43.3	27.6	53.5	46.3
Not completed	0.0	3.6	0.9	1.2
Not needed / tried	34.5	45.0	24.6	30.6

Note. [a] = Avg. between users. Each user value is calculated with the sum of experiences: 0-30% in playing narrative games, 0-40% in creating narrative content (e.g., novels) and 0-30% in creating narrative or serious games. [b] = Avg. between users. Scale from 0% (I do not know what they are) to 100% (I know them, and I have used them). [c] = There were 110 possible high-level actions, such as "select a scene background," that could be performed in the version of uA used in this experiment.

4.2. uAdventure as SG teaching tool in university courses

The previous pilots helped us to identify areas for improvement before attempting more ambitious experiments. By the end of the previous round of pilots, users could generate simple non-linear stories with multiple endings just by following the guided examples in the manual. We then improved both manuals and examples based on the results of the first testing round. However, the examples in the manual were relatively simple, and all participants implemented the same stories: we wanted to test uAdventure on a more realistic scenario, where a new set of participants with no uA knowledge could learn it and explore larger stories, different from each other's, in a more agile development environment.

For the second round of testing, we used uA as part of both undergraduate and graduate courses, dedicating several sessions to first teach the use of the tool, and then allow students to build their games. For this second round, we adopted a workshop-based approach, instead of the self-taught approach of the first round. This new structure offers a faster learning process, where users have a better foundation in both authoring and the serious game development process, and allowed us to carry out experiments to explore more advanced goals in smaller periods when compared to a self-taught approach. However, a workshop-based approach can also lead to false successes due to the influence of the learning process. In the case of uA, both the manual and examples had been already proven to be effective, so a workshop based on the manual's updated contents and examples allow us to start up with a realistic foundation not too different than self-taught users.

After the first round of experiments, we released an updated version (uA Beta in Figure 5), addressing all identified limitations. We also designed a narrative adventure game training workshop (*narrative workshop* for short) to address the most common conceptual issues found in the first round. This workshop requested participants to build more elaborated games and to build the example game described in the manual and used in the initial pilots in just one 2-hour session. It also included a second 2-hour session, where more advanced uA aspects such as third-person games were covered. As shown in Figure 5, the narrative workshop was tested in two courses with a diverse set of participants, including both design school and post-graduate students. The two following workshops included many more sessions and allowed students to games in teams based on their ideas. While student teams worked independently, they still had expert assistance available. Besides, these longer workshops included LA or location-based development as each plugin became ready for testing.

The rest of this section describes the common aspects of the courses where the experiments took place, while the following sections describe the particularities of each of them. All courses took place in the Faculty of Informatics at the Complutense University of Madrid, and therefore participants can be considered programmers following the classification used in the first round of experiments.

In all cases, participants were introduced to the following topics: adventure games, narrative mechanics, uA's narrative model, and features, and learning-objectives-based game design. This last topic was introduced through a demo session implementing a SG based on the examples used in the pilots. After this introductory lesson, participants were asked to implement the example game with a teacher nearby to help with any issues, ensuring that all of them fully completed the example, and closing the initial two-hour session. After this, participants were introduced to the features and examples required to build third-person games, after which they were asked to create a small third-person game example. This second session also lasted two hours.

After these two sessions, participants were asked to form teams of 4-5 people and use uA to implement one or more SGs covering different aspects and features during the next weeks. During the development, additional development-only sessions took place, with uA experts assisting the participants and providing guidance when requested. In general, the expected length of games ranged from 5 to 10 minutes and had to include multiple endings. To implement the game, participants were provided with graphical resources from an existing school-based third-person game; but were also encouraged to create or use external resources or plots.

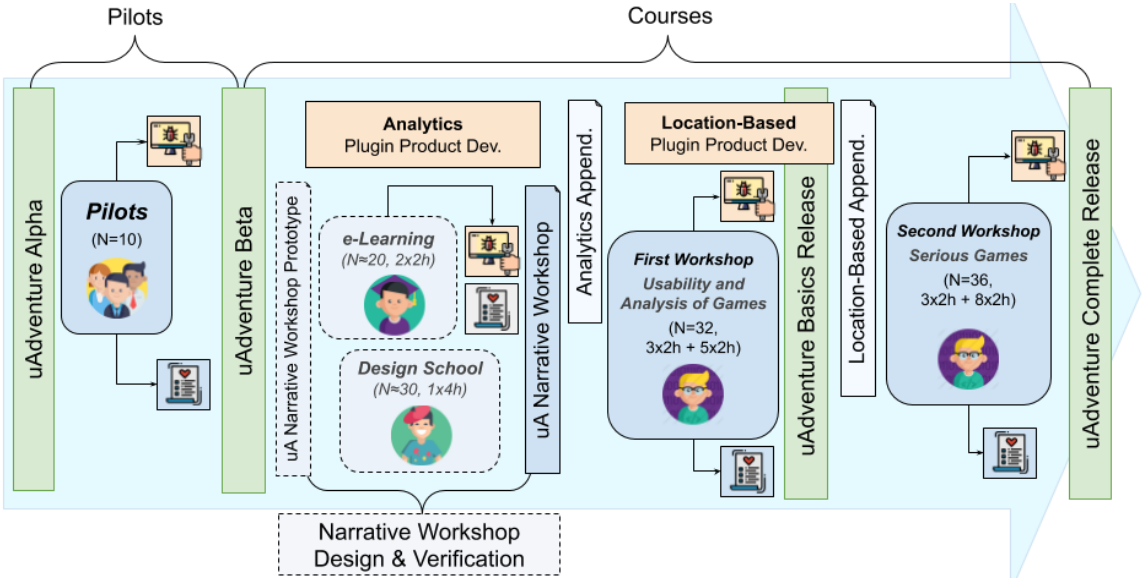

Figure 5. Timeline of the experiments (blue rounded boxes), uA releases (green), and development efforts (orange) and short workshops (dashed borders)

Finally, to improve uA software quality, participants were asked to report and submit issues for any bugs, ideas, or new features that they found or thought of while using uA. Reporting and triaging were performed through the GitHub issue system.

4.2.1. First (LA) workshop: introducing game analytics

The first workshop was designed for the "Usability and Analysis of Games" course for undergraduate students. For this reason, it included an additional two-hour session specializing on the LA extension in addition to the four-hour foundation sessions. The additional lesson introduced LA concepts, and the analytics generated by uA games, including the different traces and formats, completables, and how to configure the tracker.

This workshop focused not only on proving that users could create SGs with learning objectives, but also that they could generate analytic data from the game to whether to understand if players had learned. Hence, participants were also requested to create games with multiple endings and to include a list of analytics traces generated from the game and a document describing how the traces would be used to assess the learning objectives.

4.2.2. Second (LB) workshop: creating narrative serious games with location-based mechanics

The second workshop was designed for the "Serious Games" course for undergraduate students. After the initial four-hour foundation sessions, participants received an additional two-hour lesson covering uA's location-based features, location-based SG best practices, and LB authoring and debugging. The location-based features included augmented map-based scenes, region-based interactions, QR-based positioning, and navigation. Besides, participants were instructed on the differences in positioning accuracy indoors versus outdoors, the use of companion characters, and guided navigation to help players not to get lost in LB games, and the importance of not asking players to travel long distances with insufficient feedback. Finally, while the workshop teacher demonstrated LB game creation, participants were asked to follow and reproduce all authoring steps on their own copy of uA.

Participants were requested to develop a SG covering uA narrative features, including a simplified game design document (GDD) describing the learning objectives and how uA elements cover them; and to develop a location-based SG using the main location-based mechanics, including a simplified game design document mentioning educational objectives and how location-based elements and mechanics in the game cover each objective. For each game, teams were also asked to create three-minute gameplay videos.

5. Results

Around 50 students participated in the two short workshops (in "e-Learning" and Design School courses) to refine and validate the four-hour foundation sessions explaining basic and advanced uA concepts and use; and 68 students participated in the two main workshops (in the "Usability and Analysis of Games" and "SGs" courses), which also introduced Learning Analytics (LA) and Location-Based games (LB), respectively. In these two workshops, students received 6 hours of lessons each and were divided among 15 teams, with a team size average of 4 people for the first workshop and 5 for the second workshop. Participants in the LA-focused workshop had around 10 hours of labs during which they could contact uA experts, and those in the LB-focused workshop had 16 hours of similar labs.

A total of 22 SGs were created with uA based on each teams' ideas (5 of them third-person games), 7 of them covering uA's narrative "point-and-click", advanced and educational features, 8 covering also uA's LA features, and 7 focused on location-based mechanics. All teams implemented SGs with a learning objective-based design and multiple endings. The game topics were mostly unique for each team. Since participants in the LA workshop were provided with a set of assets for a school-centered game, they mostly focused on school life, tackling issues such as bullying, violence, drugs, plagiarism, and women in STEM. LB games were, on the other hand, mostly based on historical events in Madrid, featuring kings, writers, and wars.

To analyze the change in game complexity, we counted the different model elements in both the example game and the games created by student teams. Table 2 shows the comparison of these counts, where the first two rows reflect counts of game elements in the example and the average of team-built games. The last row displays average change, as a multiplier.

Table 2. Comparing game complexity in terms of discrete game elements between the example game and games built by teams

Game	Scenes	Cutscenes	Refs.	NPCs	Items	Actions	Conv.	Lines	Cond.	Effects
Example	4	4	12	1	1	5	1	17	14	22
Team Avg.	10	3	47.3	8.5	10.5	30.8	15.5	138	75	227
Increase	×2.5	×0.75	×3.9	×8.5	×10.5	×6.16	×15.5	×8.1	×5.3	×10.3

The complexity of team-created games was, in general, several times greater in all but one game element categories. The "Refs." column of Table 2 counts element references to NPCs, items and atrezzos (3 in the example, 35 in avg. game), areas (1 in the example, 1.6 in avg. game), and exits (8 in the example, 10.5 in avg. game).

In terms of narrative complexity, games had an average of 15 conversations and 138 dialogue lines per game. This represents an estimated 10-to-15 minutes of gameplay for "point-and-click" adventure games when compared to the duration of plays of the same length, without accounting for replayability – as players are generally interested in exploring the different available endings. Compared to the example game, this is an 8-fold increment in narrative content.

In terms of game and branching complexity, we see a six-fold increment in actions, five-fold in conditions, and ten times more effects than the example game. Resulting games had on average, at least two endings (good and bad), with a few of them, including up to four different endings.

In games with LA, there were 7 completables with 11 milestones per game on average. In location-based games, at least one map-scene was used with 13 geo elements, 5 zone scenes, and an average of 5 navigation paths.

Table 3 provides an overview of software-quality related results: a total of 70 issues were reported, including 47 bugs, 9 enhancements, and 14 new features. Most of the bugs were solved, and many of the enhancements were implemented or planned while the workshops were ongoing. We prioritized bugs, and therefore only a small number of new features were implemented in this period, mostly centered on usability improvements in effects and conversations.

Table 3. Issues reported, accepted, and solved after the second round of testing

Issues	Bugs	Enhancements	New features	Total
Reported	47	9	14	70
Accepted	43	9	13	65
Solved	35	4	3	42

6. Discussion

The experiences described in this paper helped us to conduct a continuous process of improvements for uA. The results of the pilots described in the first testing round, in which almost no participant tried or succeeded in creating their games, helped us to mature the software (e.g., fixing bugs or implementing new features) for the second, and more successful, test pilots.

Participants in the pilots generally achieved the main goal of proving that even users with no prior experience in Unity, and to a lesser extent also non-programmers, can create at least simple serious games examples. However, our results also show the importance of having a prior narrative experience to be more proficient during the authoring process. Also, several users that completed all the assigned tasks reported having problems building non-linear stories or multiple endings, which was one of the purposes of the examples. Therefore, we carefully revised the manual, providing more background on uA's narrative structure in the introduction, and generally improving uA's internal help system.

The workshop results for teams developing SGs with uA and assisted by researchers are highly positive. They prove that uA allowed participants to: (i) create longer stories (with up to 15 minutes of gameplay) with more narrative complexity; (ii) work in small teams; (iii) work in more diverse game genres; and (iv) develop serious games using an agile methodology.

These last results were more successful (besides the bug fixing of the tool as a result of the experiment) due to the fact that new features implemented after pilots (and from first to the second workshop). Besides, the longer and guided formative sessions (in contrast to the pilots where participants only had the manual, and they were on their own for few days), the workshop had hours of guided formation and weeks of free work. Last, the planning of the sessions was modified to mitigate the students' difficulties. For instance, we focused more on model the design and the way conditions are planned to prevent mistakes.

Regarding complexity, we have seen in the results a great increase in terms of conversations, lines, effects, and conditions. The example game was already based in a real serious game named Fire Protocol with several endings, and our results show that most of the participants created games that were, on average, eight times more complex than the example in the manual. We found several promising games created by participants, such as the STEM awareness game for girls, and many of the location-based games that were historical-themed and could be joined to become a complete historical tour of Madrid showing very promising results for the location-based games extension.

During the experiments, we noticed that uA elements such as scenes, characters, objects, and conversations were easier to understand than game-state aspects such as conditions and effects. A possible explanation is that during the demonstration sessions more emphasis was done on demonstrating the narrative elements (due to the low experience of participants on these aspects) than on explaining conditions and effects, which were supposed to be more familiar for participants. This last assumption seems to have been wrong, maybe due to the higher cognitive load required to understand and apply the rule-based execution model of conditions and effects.

Since uA allows the development and testing of even early prototypes, it is a good fit for teams using agile development approaches. Most of the enhancements and new features suggested and later implemented were oriented towards improving this workflow, including the usage of dropdown interfaces to make windows simpler; making conversation-editor interfaces more flexible for extending nodes and options; or having more flexible dialogs that can include images to avoid the need for using effects for such simple tasks.

Throughout the tests, participants used uA to quickly prototype their game ideas, easily linking educational value to game elements. This experience highly contrasts with the full Unity experience, which is much more difficult to learn and where SG development requires significantly more effort. In this sense, our integrated experience approach is successful, as it simplifies authoring for participants. Regarding their feedback, they liked the tool and found it very inspiring. Many proposed ideas and new features, but in general, they were satisfied with current features and even impressed by their amount and quality. Artists, in contrast, requested more graphic flexibility, while programmers asked for proper ways to hook with the tool.

An important limitation is that, even if all participants were able to follow the guided example lessons successfully, many of them required expert assistance while authoring games. Having this assistance probably had a positive impact on the success of our results, but also accepts additional interpretations. On the one hand, it is normal for developers to explore different solutions while authoring, and in these cases, expert assistance helped them choose among several possible technical solutions, of which several could have been valid from a player's perspective. On the other hand, by providing this assistance, we were able to find aspects where users require more help (as issue-reporting has shown), and thereby reduce the need for assistance in subsequent experiments. Therefore, while we cannot state the extent to which uA can achieve its goals in unsupervised environments, it is certain that a newer version based on the lessons learned in the latest experiments would be an improvement over previous versions of the tool.

7. Conclusions and future work

This paper presents uAdventure, an SG game authoring tool, and describes how uAdventure was tested in different settings. With uAdventure, non-experts can start developing adventure SGs without having to invest in expensive software or the time to learn general game development. Moreover, with the eAdventure to uAdventure conversion tool, the lifespan of pre-existing eAdventure users can be expanded with minimum effort.

Implemented on top of the well-known Unity game environment, uAdventure allows authors to use high-level concepts (e.g., scenarios, characters, conversations) to develop "point-and-click" adventure games instead of dealing

with the complexity of the full-featured Unity. This way, uAdventure makes use of Unity's advantages in terms of performance, support for emerging game technologies, and ability to package its games for multiple platforms. The uAdventure tool brings together the main narrative elements of "point-and-click" adventure games in a simplified, easy-to-understand model, allowing non-expert game developers to focus on the educational features they need most. To this end, uAdventure hides or replaces most of Unity's user interface, which could be overwhelming for non-experts. Therefore, uAdventure is not just a Unity extension, nor a standalone editor, but an environment that transforms Unity into a "point-and-click" SG editor affordable to non-experts.

The first testing round included a formal evaluation with different kinds of users, yielding promising results for in simple narrative adventure game development. However, its results did not extend to more realistic games, although they were very helpful in improving uA and its ancillary materials. In the second testing round, we tested uAdventure through workshops in four courses, with two short workshops focusing only on narrative features, and two much longer workshops focusing on one extension each: learning analytics and location-based games, respectively, over many more sessions. In these last two workshops, students were asked to create full SGs, with very impressive results: all participants were able to successfully create their games in an autonomous development with expert's assistance, and several were of high quality. The second testing round also serves to improve uAdventure's quality and user experience, as participants identify many issues and enhancements that were considered.

Next steps in the project are focusing on three aspects: first, to verify that the uAdventure games can be actually used in real environments (e.g., schools); second, to test uAdventure with more diverse user profiles in non-assisted environments, and specifically with projects where participants contribute with specific but diverse skillsets, such as artists collaborating with educators and programmers; and third, to create new uAdventure extensions designed to improve the experience of expert Unity developers when using uAdventure as a prototyping tool. To address the first goal, we are currently developing a location-based SG (including learning analytics features) for the vice-rectory of Sustainability and Environment of the Complutense University of Madrid, planned to have a high volume of participants testing uAdventure's features in a complex and diverse environment. For the second aspect, we are planning a hackathon next year using the uAdventure final release in the Designer School of Madrid, oriented at multi-profile teams. Finally, to achieve the third goal, we are currently developing plugins that integrate Unity scenes as mini-games into uAdventure.

Acknowledgment

This work has been partially funded by Regional Government of Madrid (eMadrid P2018/TCS4307), by the Ministry of Education (TIN2017-89238-R), by the European Commission (Erasmus+ IMPRESS 2017-1-NL01-KA203-035259).

References

Almeida, F., & Simoes, J. (2019). The Role of serious games, gamification and Industry 4.0 tools in the education 4.0 paradigm. *Contemporary Educational Technology, 10*(2), 120–136. doi:10.30935/cet.554469

Amory, A. (2001). Building an educational adventure game: Theory, design, and lessons. *Journal of Interactive Learning Research, 12*(2), 249-263.

Aust, R., Nitsche, M., & Pelka, J. (2014). Digital game-based learning and video games in teacher training. Conception, evaluation and results from Leipzig University. *Perspectives of Innovations, Economics and Business, 14*(3), 113–131. doi:10.15208/pieb.2014.14

Center for Technology Implementation in Education. (2014). *Learning with Computer Games and Simulations. American Institutes of Research*. Retrieved from http://www.ctdinstitute.org/sites/default/files/file_attachments/CITEd%20-%20Learning%20with%20Computer%20Games%20and%20Simulations%20FINAL.pdf

De Freitas, S. (2018). Are games effective learning tools? A Review of educational games. *Educational Technology & Society, 21*(2), 74–84.

del Blanco, Á., Marchiori, E. J., Torrente, J., Martínez-Ortiz, I., & Fernández-Manjón, B. (2013). Using e-Learning standards in educational video games. *Computer Standards & Interfaces, 36*(1), 178–187. doi:10.1016/j.csi.2013.06.002

del Blanco, Á., Torrente, J., Marchiori, E. J., Martínez-Ortiz, I., Moreno-Ger, P., & Fernández-Manjón, B. (2012). A Framework for simplifying educator tasks related to the integration of games in the learning flow. *Educational Technology and Society, 15*(4), 305–318.

Dickey, M. D. (2006). Game design narrative for learning: Appropriating adventure game design narrative devices of interactive learning environment. *Educational Technology Research and Development, 54*(3), 245–263. doi:10.1007/s11423-006-8806-y

Freire, M., Serrano-Laguna, Á., Manero, B., Martínez-Ortiz, I., Moreno-Ger, P., & Fernández-Manjón, B. (2016). Game learning analytics: learning analytics for serious games. *Learning, design, and technology*, 1–29. doi:10.1007/978-3-319-17727-4_21-1

Göbel, S., Salvatore, L., Konrad, R. A., & Mehm, F. (2008). StoryTec: A digital storytelling platform for the authoring and experiencing of interactive and non-linear stories. *Lecture Notes in Computer Science* (vol. 5334, pp. 325-328). doi:10.1007/978-3-540-89454-4_40

Laato, S., Pietarinen, T., Rauti, S., & Laine, T. H. (2019). Analysis of the Quality of Points of Interest in the Most Popular Location-based Games. In *Proceedings of the 20th International Conference on Computer Systems and Technologies - CompSysTech '19* (pp. 153–160). doi:10.1145/3345252.3345286

Lauricella, A. R., Cingel, D. P., Blackwell, C., Wartella, E., & Conway, A. (2014). The Mobile generation: Youth and adolescent ownership and use of new media. *Communication Research Reports, 31*(4), 357–364. doi:10.1080/08824096.2014.963221

Marchiori, E. J., Torrente, J., Del Blanco, Á., Moreno-Ger, P., Sancho, P., & Fernández-Manjón, B. (2012). A Narrative metaphor to facilitate educational game authoring. *Computers and Education, 58*(1), 590–599. doi:10.1016/j.compedu.2011.09.017

Nadolski, R. J., Hummel, H. G. K., van den Brink, H. J., Hoefakker, R. E., Slootmaker, A., Kurvers, H. J., & Storm, J. (2008). EMERGO: A methodology and toolkit for developing serious games in higher education. *Simulation and Gaming, 39*(3), 338-352. doi:10.1177/1046878108319278

Perez-Colado, I. J., Perez-Colado, V. M., Martinez-Ortiz, I., Freire-Moran, M., & Fernandez-Manjon, B. (2017a). uAdventure: The eAdventure reboot: Combining the experience of commercial gaming tools and tailored educational tools. In *Proceedings of IEEE Global Engineering Education Conference - EDUCON 2017* (pp. 1755–1762). doi:10.1109/EDUCON.2017.7943087

Pérez-Colado, V. M., Pérez-Colado, I. J., Martínez-Ortiz, I., Freire-Morán, M., & Fernández-Manjón, B. (2017b). Simplifying location-based serious game authoring. In *Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality - TEEM 2017* (pp. 1–9). doi:10.1145/3144826.3145395

Rollings, A., & Adams, E. (2003). *Andrew Rollings and Ernest Adams on game design.* San Francisco, CA: New Riders Publishing.

Salter, A. (2014). *What is your quest?: From Adventure games to interactive books*. Iowa City, IA: University of Iowa Press.

Serrano-Laguna, Á., Martínez-Ortiz, I., Haag, J., Regan, D., Johnson, A., & Fernández-Manjón, B. (2017). Applying standards to systematize learning analytics in serious games. *Computer Standards & Interfaces, 50*, 116–123. doi:10.1016/j.csi.2016.09.014

Spierling, U. (2009). Models for interactive narrative actions. In *Proceedings of the Sixth Australasian Conference on Interactive Entertainment - IE '09* (pp. 1–8). doi:10.1145/1746050.1746062

Torrente, J., del Blanco, A., Marchiori, E. J., Moreno-Ger, P., & Fernandez-Manjon, B. (2010). <e-Adventure>: Introducing educational games in the learning process. In *Proceedings of IEEE Global Engineering Education Conference - EDUCON 2010* (pp. 1121–1126). doi:10.1109/EDUCON.2010.5493056

Xanthopoulos, S., & Xinogalos, S. (2018). An Overview of location-based game authoring tools for education. In *Advances in Intelligent Systems and Computing* (pp. 201–212). doi:10.1007/978-3-319-75175-7_21