uAdventure: Simplifying Narrative Serious Games Development

Pérez-Colado Víctor Manuel, Pérez-Colado Iván José, Freire-Morán Manuel, Martínez-Ortiz Iván, Fernández-Manjón Baltasar
Software Engineering and Artificial Intelligence Department
Faculty of Informatics, Complutense University of Madrid
Madrid, Spain
e-mail: {victormp, ivanjper}@ucm.es, {manuel.freire, imartinez, balta}@fdi.ucm.es

***Abstract*—Game engines and developer-friendly authoring tools have greatly simplified entertainment game development. However, this does not extend to serious games which, in particular, require involvement of non-developer stakeholders. We present the first evaluation of uAdventure, an easy-to-use game development environment for narrative point-and-click graphic adventure games. uAdventure is a re-implementation of the previously-validated eAdventure environment on top of the Unity game engine. The idea is to get most of the advantages of the Unity professional environment at a fraction of its complexity, and without requiring programming knowledge to use it. uAdventure include educational-oriented affordances, such as assessment and learning analytics; and has been formatively evaluated by heterogeneous users with different degrees of technical knowledge. The results of our evaluation show much simpler story creation for profiles that had no previous knowledge of the engine, and positive feedback from more technical profiles which would use the tool as a prototyping tool for complex projects.**

Keywords- Serious game, authoring system, narrative games, learning analytics.

I. INTRODUCTION

Serious Games (SGs) are games where the main purpose is not entertainment, being education one of the most extended. There are multiple benefits for students when it comes to learning through SGs and gamified experiences [1], most notably, a better engagement. However, despite of multiple initiatives having demonstrated positive impact in both educational institutions and companies, SGs are not yet widely used in education [2]. One of their biggest drawbacks are the high costs of using SGs for educational institutions, that must either purchase already validated games, or must develop their own games involving non-developer stakeholders in their creation, increasing development costs.

The popularity of free game engines with fully featured authoring tools has opened new possibilities in the SGs market, as it has happened with the indie game market [3]. However, the growth in freely available game authoring tools has not had as much impact for SGs, at least when compared to more mainstream types of video games. The use of SGATs (SG-specific authoring tools) can greatly simplify SG development [4], [5], making it accessible for non-programmers and non-technical SG stakeholders, most notably teachers, in the creation process. This paper describes uAdventure, an SGAT implemented on top of the Unity game engine, fully affordable for non-experts, and with geolocalization and learning-oriented features such as learning analytics.

To simplify the development process and make it affordable to non-experts, SGATs should focus on facilitating high-level tasks, instead of overwhelming authors with unwanted flexibility in game mechanics. This can be achieved by specializing the tool in a single game genre. Specific genres support different types of learning [6]. Game-like simulators allow authentic learning through recreating real-life situations [7], but the complexity of their mechanics can be expensive to develop. Adventure games are adequate for creating educational games as they combine a strong narrative with puzzles and problem solving, promoting learning and reflection [8].

The rest of this introduction provides greater background on adventure games, how they can benefit SGs, and the general process of authoring adventure games. The rest of the paper is structured as follows: Section 2 explores uAdventure's (uA) features to simplify adventure SGs development for non-experts; Section 3 details the pilots performed with uA; Section 4 discusses the results; and finally, Section 5 comprises conclusions and future work.

A. *Adventure genre for serious games*

Adventures are story-driven games, and hence, they inherit elements from novels, including *characters*, which are the elements (normally persons) that populate the story and perform the different actions; *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 is placed inside of this narrative context, and must interact with the environment (scenarios) and the characters or items it contains, occasionally solving mini-game puzzles. These features are especially useful in educational games, as players need to

Figure 1. Broken Age (2014), a "point and click" game. The image shows two characters having a conversation and the conversation options given.

reflect on their knowledge to solve in-game puzzles and advance the story. According to [8], graphic adventures of the "point and click" (Figure 1) sub-genre, where the mouse or touch is used to select the protagonist actions, are one of the best genres for SGs.

Stories themselves can be non-linear, that is, allow access to a whole "sequence of possibilities considering changing world states." [9]. Where a linear story follows a strict sequence, non-linear stories can result in different states (endings, scores, etc.) depending on the players. This is not without drawbacks, since, for example, in the "King's Quest" game series, some interactions could lead to unresolvable game states named "dead ends" [10]. However, non-linearity is ideal for SGs, as it provides a greater sense of immersion for players, and the resulting game state can be used as an assessment measure, combined with the players' responses to in-game questions and problem-solving history, among other sources of analytics.

B. *eAdventure*

The eAdventure (eA) SG engine (Figure 2) is a previous SGAT for the creation of "point and click" games [11]. The key goal of eA was to provide a multiplatform adventure game engine and reduce the complexity of using high-level concepts such as scenes, characters, interactions, and conversations and allowing to package games as educational learning objects using different educational standards like IMS Content Packaging and ADL SCORM [12]. For interaction, eA provided a high-level model based on *conditions* and *effects*, where the former had to be fulfilled to apply the changes in game-state described by the corresponding effects. At the time eA was created, game engine & authoring tool licenses were expensive. As a free and open-source project, eA was a good fit for educators and small teams.

eA games were mostly delivered via Java applets, which are now largely obsolete. At the time, applets allow to package and distribute a SG using a Learning Management Systems such as Moodle, but nowadays we have lost this advantage. In addition, the growth in the numbers of mainstream game platforms opened new opportunities, allowing us to focus on the distinctive eA's features. Relying on game engines such as Unity to handle low-level tasks can greatly reduce development and maintenance costs and allow future development to focus on educationally-valuable features, that make uA unique.

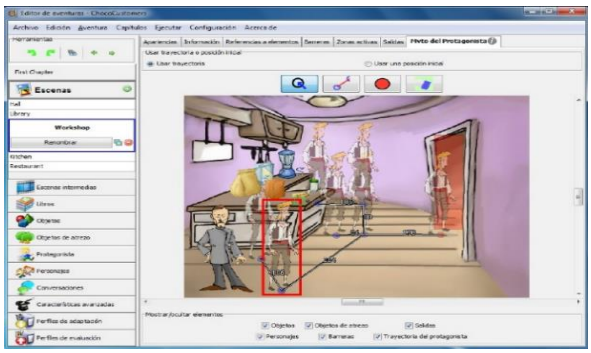

Figure 2. The scene editor in the eAdventure SGAT.

C. *Unity*

The Unity game engine is multiplatform, supporting the creation of games for all major operating systems, mobile devices (including iOS and Android) and consoles. It provides generic functionalities such as animations, physics, cameras and lightning systems, and support for almost any conceivable game mechanic. However, scripting is needed to create genre-specific mechanics, making it inaccessible to non-experts. Unity itself is designed to be highly customizable and supports a "Unity Store" where developers can share and plugins that add or modify its default behavior.

II. UADVENTURE FEATURES

uAdventure (uA) is a framework built on top of Unity with the goal of simplifying the creation of narrative "point and click" SGs by non-experts. By building on top of Unity, uA solves eA's major problems: uA can always support the latest technologies and platforms, reusing most of Unity's subsystems and focusing instead on providing features of a narrative and educational nature, making it more maintainable and less likely to become obsolete [13]. This is key to extending the lifecycle of the games created with uA and eA as it includes backwards compatibility.

This section provides an introduction of distinctive features and capabilities that uA offers. Sections II.A and II.B describe the high-level narrative model editor, which allows easy creation of narrative content to present to the players, promoting exploration and creativity through non-linearity, and allowing the addition of challenges through logic-puzzles and quizzes. Section II.C and II.D describe the simplified created to isolate non-experts from the full complexity of Unity (uA replaces most of the Unity interface with custom windows), designed to be used during narrative model creation. Hence, uA is not just a Unity tool, but a whole engine that redefines Unity's purpose. Finally, subsections II.E and II.F describe additional features, such as learning analytics.

A. *Narrative model editor*

The model of uA is heavily based on narrative game elements [14]: scenes, interactive / non-interactive elements, 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 zone. Moreover, scenes can be connected through areas of interest, as the basic means to build the 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/or manipulated. Finally, Cutscenes can be used to introduce linear narrative sequences, such as to provide necessary exposition when a player reaches a milestone or enters a scene for the first time.

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, an item in the scene can be examined, grabbed or used, and, depending on each action's configuration, this can lead to different states,

endings or story branches. Instead of scripting, uA provides high-level effects that easily manipulate the game context, including the current scene, game-state variables, game elements, or 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, at least when compared with conventional scripting.

The conversation system provides a tree-based editor to manipulate all the different narrative branches that a non-linear story can require. Without the conversation system, dialogues could be created using effects, but the amount of effects would be just impossible to handle for large conversations with different branches. The conversation system also manages all the runtime tasks required to display the conversation, such as displaying different dialog bubbles from the character that is speaking, changing character animations, and displaying the different options in the screen for the player to select them.

B. Simplified scene editor

The uA scene editor allows the author to create scenes and manages their elements, including characters and items. Although Unity supports 3D, we have chosen to limit uA to 2D scenarios because they are more common in adventures and, due to their greater simplicity, are easier to author. Additionally, the capability of managing elements directly from the scene editor is a breaking change compared to Unity's scenario editor, where users must deal with different rendering components and behavior scripts. uA also sets scene sizes and boundaries automatically and handles the camera configuration and transitions. In addition, the scene itself may contain game-relevant elements that visually exist in the scene background. Hence, they can be handled by defining their region in the scene and defining their functionality such as exits or barriers.

Finally, the scene editor manages the properties of each element by displaying a Unity-like inspector, making it more affordable for native Unity users.

C. Integrated experience

uA provides different editors that simplify the content creation in an adventure. To provide a streamlined experience for non-experts, uA automatically switches Unity's layout, by closing unrelated windows such as the hierarchy window, and replacing them with SG-relevant views, including the model editor, context variables and flags, a gameplay window, and a project file explorer.

uA also inserts itself to respond to 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 opened back. Saving and loading projects works in a similar fashion.

Finally, uA uses the asset importing pipeline to automatically handle asset configurations when users copy assets into their project folders. For example, any image placed in the scene backgrounds folder will be automatically configured as a readable sprite, ready to be included into an actual uA scene. This allows authors to ignore the technicalities involved in configuring each asset type.

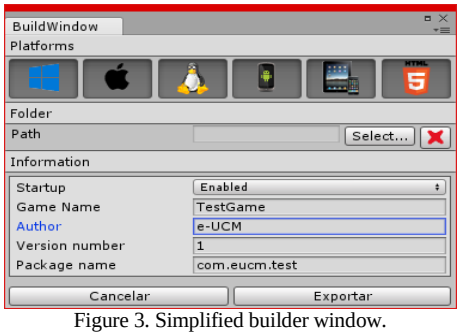

Figure 3. Simplified builder window.

D. Simplified Builder

One of the most important features of Unity that helped it gain popularity is its multiplatform support (up to twenty-five platforms by 2019), reducing project costs by avoiding expensive platform ports, and maintenance costs by keeping a single code-base. Building a game is the process of generating executable distributions for each of the target platforms, and the builder is the Unity tool that allows it to be configured.

Supporting multiple platforms requires significant tuning to achieve good performance in each of them, for example in terms of quality or lighting. For this reason, uA provides a simplified builder that allows choosing only those platforms relevant to the educational field, and exporting the game in one step (see Figure 3), while automatically configuring all platform-dependent settings by assuming that the target audience (education) will either use the students' devices (according to "bring your own device" format [15]), or rely on institutional computers, which will frequently be running outdated software or hardware [16]. Therefore, the simplified builder enables Windows, Linux, MacOS, WebGL, Android and iOS platforms, discarding non-school-feasible platforms (e.g. consoles), and aiming for minimum requirements.

E. Learning analytics

Learning analytics (LA) are an important trend in learning understanding and assessment. For serious games, LA can allow game success (or failure) to be measured and help to understand its causes. LA applied to SG usually comprises the gathering of the main relevant game events (or, at worst, every interaction), in order correlate these relevant events with educational objectives. Instead defining a custom protocol, uA uses the the xAPI standard, more precisely, uses the xAPI serious game's profile as interchange format for the events [17]. These events include element interaction, player movement, dialog choices and game progress.

uAd implements the xAPI profile out-of-the-box, without forcing authors to deal with analytics concepts. Since xAPI is event-based (although in xAPI terminology events are called statements), an xAPI-compliant server, such as an LRS, can gather the different events and states the non-linear stories pass through and later reconstruct the gameplay. Most major uA elements, such as scenes, can generate and send xAPI statements automatically; but to measure game progress and

success in so-called "completables" (milestones such as quests or puzzles), uA provides an additional system to submit progress reports to the analytics component when the necessary conditions are met.

F. Extensions

Since game engines and SDKs are a very competitive environment, Unity adds novel functionalities pertaining to new technologies, such as Augmented Reality, in each new version. To leverage this evolution, uA is easily extensible, allowing the creation of new modules as trends emerge. Modules can extend the main editor, the effect system or the scene runner, among other entry points.

Two extensions have been developed so: GPS and QR. Both extensions are served in a pack of location-based game mechanics [18]. The GPS extension includes functionalities for outdoor positioning and a new type of map-based scenes, while the QR extension complements this by using QR codes and a scanner to determine player position indoors. The location-based extensions are best used in mobile platforms, allowing players to embark in geo-positioned adventures, a new genre of game that allows exploration and investigation in augmented real-world environments.

III. TESTING USER EXPERIENCE

To study the extent to which uA's goals have been achieved we performed a preliminary evaluation with users from different profiles, including non-technical users such teachers; artists and programmers. The purpose of the evaluations was mainly finding out the difficulties and issues in uA but also to verify that with uA users are capable of: i) developing point and click game examples, ii) create non-linear stories with multiple endings, iv) develop their own stories autonomously and v) build their games.

A. Experimental design

To prove the objectives above, participants with no prior experience with uA were provided with the latest uA software and manual to teach them the narrative elements and uA concepts, which is going to be the average use case in uA developments. To make the users create a story and use the tool, the manual contains eleven guided examples, through which the user can create a simple non-linear story with multiple endings but excluding advanced tasks such as timers or macros. In the examples, the user creates a story based on existing emergency procedures in case of fire. Into the game, the player can either win or lose depending on how they choose to follow the protocol. Finally, participants were encouraged to develop their own stories to test tasks not included in the examples, and to build and test their games for the Windows and Android platforms.

After the intervention a survey was used to build a user's profile and to measure the difficulty of each performed task in four levels: easy, normal, difficult or not-accomplished. In total, 110 simple tasks (e.g. "create a scene") were identified to test the main uA functionality. The user profile consists in a self-classification into non-technical users, artists and programmers and a set of questions to create a narrative profile, regarding experience creating stories and development of narrative games. Finally, users were also asked about their success (and doubts) in examples and goals and if the manual helped them through the learning process.

B. Results

The results were grouped by the different profiles and general overall success is calculated. Table 1 shows different sections regarding user's background, manual and concept understanding, goals completed, and tasks completed. The last section analyzes task complexity and hence, uA success in terms of interface simplification.

Regarding the first goal, the results in Table 1 show that participants have finished 95% of the examples successfully, which is corroborated by the percentage of incomplete or "found hard" tasks, which shows that most of tasks are rated simple (~65%). Individually, artists have the lowest success in completing examples, with a 15% over the average value in tasks not tried. For goals two and three, regarding users' creation of non-linear stories and multiple endings, the results show a 70% of success.

Regarding the manual, all users reported finding it helpful. However, all groups have also reported difficulties using uA. For the computer users, doubts with the model are related to the lack of narrative experience, as they are probably less familiarized with the concepts of characters, linear and non-linear stories and they haven't authored stories before, which is also present in programmers. On the other hand, the doubts in effects and conditions are more visible in the groups with no programming background as expected. Finally, another relation is present between the doubts in users and the lower simplicity reported, being more evident in average computer users that only reported ~26% as simple.

Additionally, participants also reported 29 different issues, misconceptions and difficulties to be fixed in the software and its documentation for the final release.

TABLE I. SUMMARY OF RESULTS FOR DIFFERENT PROFILES (NON-EXPERTS, ARTIST, PROGRAMMER).

	User category			
	Non-tech.	***Artist***	***Prog.***	***All***
User profile				
Amount	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
Doubts about model elements?	1	0	1	2
Doubts in 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
Individual use cases (110 in total)				
Completed (%, n=110)	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	1,01	0,9
Not completed	0,0	3,6	0,90	1,2
Not needed / tried	34,5	45,0	24,6	30,6

a. *30% played narrative games, 40% created narrative content, 30% created narrative or SG*
b. *Avg. between users. Scale from 0% (I do not know what they are) to 100% (I know them, and I have used them)*

IV. Discussion

The software development industry has grown big and strong the last few years. However, the SG industry has not followed this trend, specially, due to costs. With uA we are trying to drastically break this trend providing a simple tool to easily create SGs ready for multiplatform support.

For this reason, uA takes Unity (a next-gen engine) and replaces all the functionalities to focus in high-level concepts without having to interact with native Unity concepts. Even so, it is impossible to hide all the Unity features and results show that users get distracted and confused by these options (e.g. the top menu in Unity). Although this may lead to some issues, results show that non-experts have been able to generate narrative stories in Unity.

Our development concluded with a pilot that reported good results on the users' objective consecution, and therefore, verified that even users with neither Unity nor programming skills can create at least simple examples. However, some results show the importance of narrative experience by users (prior to the development) to reduce the doubts. Also, some users have reported to complete all the examples but also denied having been able to create non-linear stories and multiple endings, which was the purpose of the examples. Therefore, we expect to generate a revised version of the manual, providing more background on narrative structure in the introduction, and improving uAdventure with additional in-software documentation.

V. Conclussions and future work

This paper presents the preliminary evaluation of the game authoring environment uAdventure (uA) as an evolution of a pre-existing system called eAdventure (eA). With uA, it is now possible for non-experts to start developing adventure SGs without having to invest in expensive software and expert developers. uA simplifies game development by abstracting the main narrative elements of "point and click" adventure games (e.g. scenarios, characters, conversations) into a simplified, easy-to-understand model, allowing non-expert game developers to focus on the features they need.

uA substitutes the default Unity interface with genre-focused windows, simplifying Unity's editors and processes. Therefore, uA is not a Unity extension, nor a standalone editor, but an environment that transforms Unity into a "point and click" SG editor affordable to non-experts.

To improve the authoring tool, the uA environment has been formatively evaluated with different kinds of users. This evaluation has reported very promising results especially in the creation of simple narrative games by people with no previous experience in game development. This evaluation has also helped in creating a better user experience for different kinds of users.

As a next step, we would like to do a new iteration based on the analysis and observations gathered in the experiment presented in this paper and, later on, to use the updated uA version in a formal course on interactive narrative for graphical designers in a design school in Madrid. Finally, we are developing additional uA extensions to allow the creation of location-based narrative adventure games and improving the game analytics obtained from those games. Location-based SGs creation can be greatly simplified with uA, producing games both connected to the real world and easier to integrate into the learning cycle through LA.

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 (BEACONING H2020-ICT-2015-687676, Erasmus+ IMPRESS 2017-1-NL01-KA203-035259).

References

[1] S. De Freitas, "Are Games Effective Learning Tools? A Review of Educational Games," *Educ. Technol. Soc.*, vol. 21, no. 2, pp. 74–84, 2018.

[2] R. Dörner, S. Göbel, W. Effelsberg, and J. Wiemeyer, *Serious Games Foundations, Concepts and Practice*. Cham: Springer International Publishing, 2016.

[3] A. Kovanto, "The Improvements for Indie Game Development," KARELIA UNIVERSITY OF APPLIED SCIENCES, 2013.

[4] E. J. Marchiori, J. Torrente, Á. Del Blanco, P. Moreno-Ger, P. Sancho, and B. Fernández-Manjón, "A narrative metaphor to facilitate educational game authoring," *Comput. Educ.*, vol. 58, no. 1, pp. 590–599, 2012.

[5] Á. del Blanco, J. Torrente, E. J. Marchiori, I. Martínez-Ortiz, P. Moreno-Ger, and B. Fernández-Manjón, "A framework for simplifying educator tasks related to the integration of games in the learning flow," *Educ. Tech. Soc.*, vol. 15, no. 4, pp. 305–318, 2012.

[6] R. Aust, M. Nitsche, and J. Pelka, "Digital game-based learning and video games in teacher training. Conception, evaluation and results from Leipzig University," *Perspect. Innov. Econ. Bus.*, vol. 14, no. 3, pp. 113–131, Sep. 2014.

[7] Center for Technology Implementation in Education, "Learning with Computer Games and Simulations," *Am. Institutes Res.*, 2014.

[8] M. D. Dickey, "Game Design Narrative for Learning: Appropiating Adventure Game Design Narrative Devices of Interactive Learning Environment," *Educ. Technol. Res. Dev.*, vol. 54, no. 3, pp. 245–263, 2006.

[9] U. Spierling, "Models for Interactive Narrative actions," in *Proceedings of the Sixth Australasian Conference on Interactive Entertainment - IE '09*, 2009, pp. 1–8.

[10] K. Q. Omnipedia, "Dead end." [Online]. Available: http://kingsquest.wikia.com/wiki/Dead_end.

[11] J. Torrente, A. del Blanco, E. J. Marchiori, P. Moreno-Ger, and B. Fernandez-Manjon, "<e-Adventure>: Introducing educational games in the learning process," in *IEEE EDUCON 2010 Conference*, 2010, pp. 1121–1126.

[12] Á. del Blanco, E. J. Marchiori, J. Torrente, I. Martínez-Ortiz, and B. Fernández-Manjón, "Using e-Learning standards in educational video games," *Comput. Stand. Interfaces*, vol. 36, no. 1, pp. 178–187, 2013.

[13] I. J. Perez-Colado, V. M. Perez-Colado, I. Martínez-Ortiz, and B. Fernández-Manjón, "uAdventure: The eAdventure reboot," in *IEEE Education Engineering EDUCON 2017 Conference*, 2017.

[14] A. Salter, *What Is Your Quest?: From Adventure Games to Interactive Books*. University of Iowa Press, 2014.

[15] G. Disterer and C. Kleiner, "BYOD Bring Your Own Device," *Procedia Technol.*, vol. 9, pp. 43–53, 2013.

[16] I. Škorić and M. Milić, "Computers in school: A student's perspective," in *MIPRO 2010 - 33rd International Convention on Information and Communication Technology, Electronics and Microelectronics, Proceedings*, 2010, pp. 1056–1061.

[17] M. Freire, Á. Serrano-Laguna, and B. Iglesias, *Game learning analytics: Learning analytics for serious games*. 2016.

[18] V. M. Pérez-colado, I. J. Freire-morán, I. Martínez-Ortiz, M. Freire-Morán, and B. Fernández-Manjón, "Simplifying location-based serious game authoring."