Extending Narrative Serious Games Using Ad-Hoc Mini-games

Víctor M. Pérez-Colado(✉), Iván J. Pérez-Colado, Iván Martínez-Ortiz, Manuel Freire-Morán, and Baltasar Fernández-Manjón

Department of Software Engineering and Artificial Intelligence, Complutense University of Madrid, C/ Profesor José García Santesmases, 9, 28040 Madrid, Spain
victormp@ucm.es

Abstract. Narrative games have proven their effectiveness as serious games in different domains and for different purposes, such as promoting learning or increasing user awareness. However, there are many situations where the narrative model falls short and can benefit from being extended with puzzles or mini-games to afford more flexibility or explore non-narrative mechanics more adequate to the task at hand. In our uAdventure game authoring environment, a narrative serious game provides support for the driving narrative, managing stories, conversations, and other narrative elements, together with an integrated game learning analytics support. We present how we have extended uAdventure to support the inclusion of mini-games within a host narrative game, while allowing hosted mini-games to access uAdventure services through a streamlined interface. As a case study, we describe how this extension has been used by students learning serious games development to create narrative games with ad-hoc puzzles that use alternate mechanics to achieve game-specific goals.

Keywords: Serious games · Narrative games · Mini-games · Game authoring · Game analytics

1 Introduction

The use of video games in teaching has attracted great interest from teachers and researchers for their ability to improve the interest and retention of players by keeping them motivated [1, 2]. As their primary goal is not that of entertainment, such games are often called serious games; for example, players may face a series of challenges to improve knowledge or cognitive skills [3]. Genres such as action games allow players to improve their dexterity and reflexes, while simulation games use realistic environments where players can test and apply their knowledge without being exposed to the risks of real environments [4].

Due to the different characteristics of serious games, for example to their genre, the requirements for their design and development also vary. For example, simulation games, which are very effective for specific learning, are also very costly to develop due to the high level of detail required to emulate the target domain. Therefore, when

W. Zhou and Y. Mu (Eds.): ICWL 2021, LNCS 13103, pp. 63–74, 2021.
https://doi.org/10.1007/978-3-030-90785-3_6

developing a serious game, the genre chosen is key for both its effectiveness and educational applicability as well as for its development feasibility in terms of requirements and costs. While development costs can sometimes be reduced by using pre-built assets (such as those available at Unity's Asset Store), such assets will require significant effort and expertise to customize for any sufficiently-specific domain.

Among the different types of games, narrative games offer a good balance between simplicity and flexibility, as they allow players to play roles and perform meaningful tasks in an environment not excessively expensive to create, but rich enough to achieve immersion. In particular, the "point and click" subgenre of narrative games [5] often relies on interactive conversations to deliver content, using characters who play different roles in the story; together with different objects and interactions to solve logical puzzles where players can apply their knowledge and learn from their mistakes [6]. For this reason, narrative games have been applied in very different domains and with very different purposes (e.g., learning, awareness).

However, narrative mechanics also have limitations. For example, they are not suitable for the development of other types of cognitive skills, such as pattern recognition, reflexes, or physical skills in general, which can be taught through mechanics present in other genres (e.g., action games to develop reflexes) [7]. Despite the possibility of simulating the mechanics from other genres, the adaptation needed to implement them is complex and costly. For example, developers that intend to use a narrative engine to develop a visual puzzle would need to manually analyze the different states of the puzzle and configure a series of narrative elements (and their corresponding graphical representations) to represent the state of the puzzle. This is impractical, complex and, above all, unnatural for a game developer; and is the case of commercial narrative content authoring tools such as Adobe Captivate[1], Articulate 360[2] and ITyStudio[3].

We describe an extensible model that allows enriching narrative games with other mechanics, such as puzzles; and an implementation of this model in a tool to demonstrate the feasibility of this approach. For the integration of mini-games in narrative games we have taken as a basis uAdventure [8], an open-source authoring tool for serious narrative games built on the Unity game platform. The challenge is to incorporate other mechanics into uAdventure's narrative game model, which increases its versatility and allows new capabilities to be addressed, all without significantly increasing the difficulty of game creation. However, the inclusion of these new mechanics through the mini-games requires game creators to know how to program the mini-games. This approach has been tested in two case studies with students within the serious games course of the Videogame Development degree at Complutense University of Madrid.

The following sections of this article describe the mini-game integration model, the use cases of this integration model, lessons we have learned from our experiences, and finally conclusions and future lines of work.

[1] https://www.adobe.com/es/products/captivate.html.
[2] https://articulate.com/360.
[3] https://itystudio.com/.

2 Extending the uAdventure Narrative Model Through Ad-Hoc Mini-games

uAdventure is a tool that simplifies the prototyping and development of games for non-experts, such as most teachers, as no programming is required. The authoring metaphor with uAdventure is based on creating a series of scenes, where the author will use the narrative for the player to develop a role through the game. Players learn new content or develop additional awareness by interacting with other players, participating in in-game conversations, or solving logical puzzles. For situations where it is too complex to adapt the content to the narrative model, uAdventure also provides other types of scenes where it is possible to include videos and interactive animations that can be interspersed with the main narrative content [8].

Furthermore, uAdventure includes other educational-specific features, such built-in assessment of learners and the capability to deploy and integrate games with other educational platforms by means of widely used educational standards [9]. In addition, player actions can be recorded and used internally in the game (through a mechanism of variables and conditions to change game behavior), as well as externally through the built-in learning analytics mechanism. Use of built-in learning analytics allows creators to not only access the final results at the end of the game (e.g. score, completeness, progress), but also to analyze the full sequence of actions that each player took to achieve that score, with minimal work required from game authors [10, 11].

Despite the benefits of uAdventure for most developers, the point-and-click narrative model it implements may fall short of the requirements of some games. For example, the narrative model ill-suited for puzzles with a very large number of states. For example, in a Rubik's Cube representation, where pattern recognition and spatial recognition skills can be learned, a direct three-dimensional representation would be much simpler than a representation through narrative elements. Our proposal provides support for including self-contained, alternate game mechanics that are more appropriate for the type of skills to be worked on in those parts of the serious narrative game that require it. The challenge is to achieve the integration of those alternate mechanics within the narrative game model of uAdventure, therefore increasing its versatility and allowing new skills to be addressed while avoiding a large increase in the difficulty of creating such games. While the integration of these new mechanics would certainly require a programming effort, once created, our goal is for the mechanics to be freely reusable by other authors without needing to program them themselves.

The process of adding new mechanics is done by programming mini-games integrated into uAdventure [12, 13]. The mini-games are launched from uAdventure, which delegates full control of the game execution to them (Fig. 1). With this model it is possible, for example, to develop an action mini-game that can read the state of the game to adapt its difficulty to the narrative moment in which it is found and use conversations during the mini-game itself to explain it or as part of its operation. The mini-games are developed as new components integrated with uAdventure, so mini-game authors must know how to program for the Unity platform. In this regard, mini-games can exploit all the capabilities of Unity to create the new mechanics, but they must also be integrated with the services offered by uAdventure: the core of the game; the narrative engine, and the learning analytics engine.

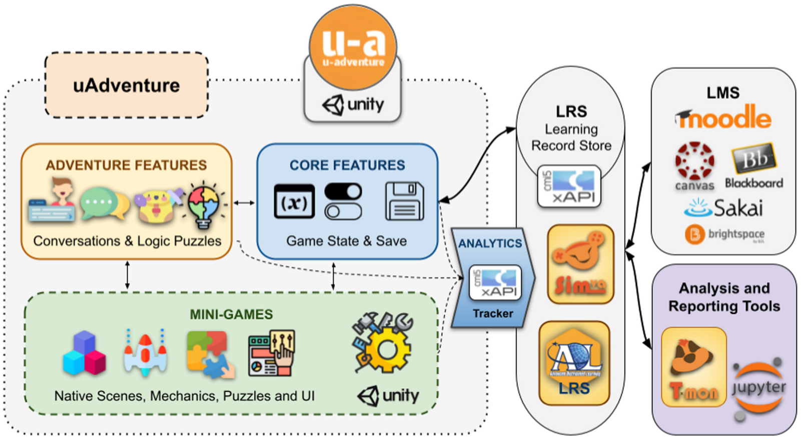

Fig. 1. Mini-game integration into the uAdventure game authoring tool. Mini-games allow additional game mechanics to be added to uAdventure, and can make use of uAdventure-provided features such as conversations, persistence, or xAPI-SG analytics.

The core game system is the system in charge of maintaining and managing the global state of the game. Using it, the mini-game can inspect the current state of the main game as a result of the previous actions that the player has carried out up to this moment. This allows the mini-game to adapt to its context of use within the current session; for example, by adjusting its own difficulty, or the style or visibility of certain features. In addition, mini-games can also save significant actions performed by players to the game state. In summary, through the game state, it is possible to communicate bidirectionally with uAdventure and thus read the different parameters that will define the configuration of the game; and write in uAdventure those parameters that are relevant once the main narrative game resumes.

uAdventure's narrative engine provides support for narrative content and interaction as well as corresponding visual effects. Through the narrative engine it is possible to launch conversation sequences and customizable options without leaving the mini-game. The mini-game developer has access to this uAdventure functionality, for example, to show tutorials or to pause the mini-game when it reaches a milestone, displaying a conversation at that moment. In addition, it allows scenes to be changed, and different events to be triggered once the mini-game ends, continuing with the flow of the game while taking into account the results of the mini-game. Finally, the learning analytics engine provides mini-game developers with a means to generate logs of the player's progress through the mini-game, which can be stored locally or sent to a remote storage location for further analysis. The main component of the analytics engine is the xAPI tracker [14] which provides a high-level interface to hide the details of the xAPI-SG specification and allows developers who are not experienced in analytics to generate traces to analyze the player's progress through the mini-games. Using this feature, a mini-game developer can, for example, generate traces that describe the interactions of the player with the interactive elements present in the mini-game.

From a more technical point of view, the uAdventure mini-game feature is based on Unity's scene mechanism. The developer of a mini-game must create a Unity scene into which to place the mini-game. In it, by means of the various features available in Unity (such as the physics engine, lighting, or its user interface system) and its scripting interfaces, the scene and its behavior will be shaped, thus defining its concrete mechanics and the layout of the mini-game. For example, in a serious game involving perception evaluation, the user could use the physics engine to create a realistic environment in which different particles move more naturally. Once the scene is created and an identifier is assigned, the developer will be able to reference the scene from the main game as any other native uAdventure scene, thus allowing the application of the uAdventure's authoring metaphor that does not require programming.

This trade-off allows advanced users (programmers) using uAdventure to use native Unity scenes without great effort; while offering non-expert users the possibility of reusing previously-developed mini-games within their own uAdventure games. Thanks to this extensibility model, it is possible to create an ecosystem where programmers and teachers can collaborate to improve the educational experience through the resulting game. In addition, thanks to uAdventure's built-in services provided to the mini-game, it is possible to create parameterized mini-games so that the experience is not unique to specific games and can be customized by teachers using uAdventure's visual editor.

Because this integration architecture is quite flexible, it can be adapted to all kinds of situations with different requirements. Although mini-games do not need to comply with any particular structure, Fig. 2 describes our recommended structure for mini-games, focusing on allowing the mini-game to be executed in a subrogated way to the host game, thus allowing the game to be decoupled from the main game and thus promoting a simpler distribution of roles in its development. In order to establish an independent execution of the mini-game, the input and output parameters must be defined beforehand. The input of parameters should be defined before launching the mini-game through the visual programming system of uAdventure, and will be read at the beginning of the execution of the mini-game through the core; while the output of parameters will become available at the end of the mini-game, to be incorporated into the state of the game for later use.

While the mini-game is running, it should not manipulate game variables, thus avoiding collateral effects. In order to return control from the mini-game to the main game, we have introduced a new special command that returns to the previous scene, available using uAdventure's visual programming system. Although mini-games have access to the narrative engine, and can execute conversations and provide other visual feedback, it should rely on this command to switch scenes. Finally, in terms of learning analytics, the mini-game should generate its specific traces by calling the analytics tracker directly. Mini-games developed following these recommendations act as uAdventure components and thus are reusable across games.

3 Pilots of uAdventure and Ad-Hoc Minigames

The mini-game integration model in uAdventure has been tested with students in two case studies within the Serious Games course of the Video Game Development Degree at the Faculty of Computer Science of the Complutense University of Madrid. In other words,

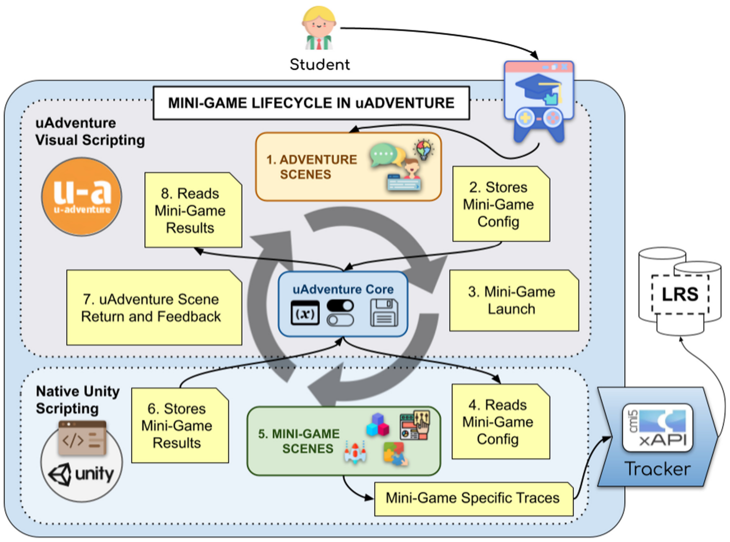
Student
MINI-GAME LIFECYCLE IN uADVENTURE
uAdventure Visual Scripting
u-a
u-adventure
1. ADVENTURE SCENES
2. Stores Mini-Game Config
3. Mini-Game Launch
4. Reads Mini-Game Config
5. MINI-GAME SCENES
6. Stores Mini-Game Results
7. uAdventure Scene Return and Feedback
8. Reads Mini-Game Results
uAdventure Core
Native Unity Scripting
unity
Mini-Game Specific Traces
LRS
xAPI
Tracker

Fig. 2. Mini-game lifecycle within uAdventure. The transition between the uAdventure and mini-game scenes allow exchange of inputs and outputs, as well as the use of the tracker for analytics. The launching process of the mini-game is customizable through uAdventure's visual editor. The actual mechanics of mini-games must be developed using Unity scripts

it has been tested with students who know how to program games, although they have limited experience in the creation of narrative games. The first case was a pilot test with the objective of addressing a need for a more flexible environment in uAdventure; and analyzed the strengths and weaknesses of the proposed integration model. The second case is more complex in terms of educational goals and narrative depth, using multiple mini-games as part of an escape room-like narrative game whose objective is to teach competencies related to computational thinking.

3.1 Pilot: Serious Game for Airport Protocols Training

In the 2019–20 academic year, students used uAdventure to prototype and create narrative games, but since they knew how to program with Unity, they were also asked to enhance these games with other mechanics and visual effects. To do so, they used the proposed mini-game system, which provided them with the flexibility of the Unity engine without losing the advantages of uAdventure.

One of the games designed by the students aimed to teach different protocols typically found in air travel (Fig. 3). More specifically, it had three educational objectives: to learn to differentiate between the objects that can be carried in the flight cabin and those that must be carried in the cargo hold; to examine the different pieces of documentation required on a flight and the different problems associated with these documents; and to remind flyers of the different safety rules that must be followed during flight.

To address the first objective, the students chose to develop a mini-game in which, through the use of "drag and drop" mechanics, players had to overcome the challenge of

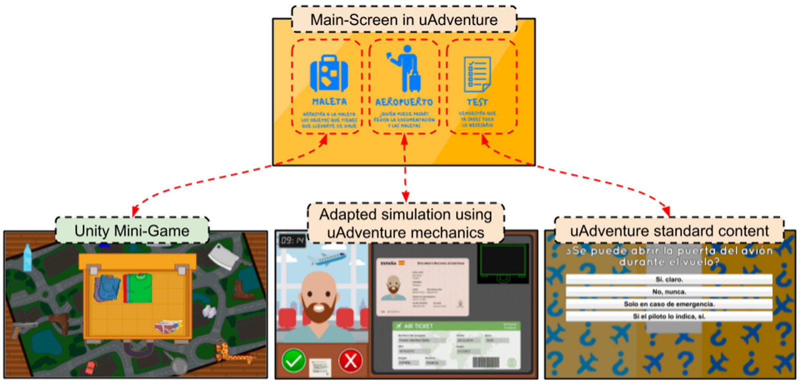
Main-Screen in uAdventure
MALETA
AEROPUERTO
TEST
Unity Mini-Game
Adapted simulation using uAdventure mechanics
uAdventure standard content
¿Se puede abrir la puerta del avión durante el vuelo?
Si, claro.
No, nunca.
Solo en caso de emergencia.
Si el piloto lo indica, sí.

Fig. 3. Integration of different mechanics in uAdventure in the serious game "Vuela" (Fly) to teach flying protocols

being able to make the most of the space in the suitcase, by including only the essential (and valid) objects for cabin carry-ons. In this case, the mini-game not only taught players to differentiate objects by their visual appearance, but also took advantage of the limited space mechanics to infer which objects should actually travel in the checked luggage due to size constraints.

Regarding the educational goal of teaching the rules related to documentation, the developers opted, utilizing the different narrative resources of uAdventure, to create a simulation in which players temporarily take on the role of an officer who must review the different documents of would-be passengers. This mechanic is particularly interesting, as it represents the case in which the mechanics of a narrative engine are used to develop a limited simulation that is at the limit before being necessary a mini-game due to the complexity of its development and the large number of elements involved in its different states. In this case, the uAdventure mini-game used a simulation in which the face and shoulders of passengers are visible to one side and, to the other side, the passenger's documentation appears. Players must then decide whether or not each passenger can board the plane, identifying possible errors or contradictions in the documentation and thus favoring deductive learning. Finally, the educational objective related to safety rules was addressed through a questionnaire built using uAdventure's conversation system.

Despite being a first proof of concept of mini-game integration, the students were able to successfully use mini-games to develop their own mechanics extending uAdventure's narrative model. Additionally, during development, multiple shortcomings were identified, including lack of details regarding the different APIs to take advantage of uAdventure's mechanics, the communication flow between the mini-game and the main game, and the integration of analytics from the mini-games. As a result, documentation was significantly improved in time for the second round of testing.

3.2 Developing an Escape Room Game for Promoting Computational Thinking

During the 2020/2021 academic year, the enhanced version of uAdventure was used to create narrative games, while placing more emphasis on the importance of learning analytics. Unlike the previous year, as a lesson learned, students received additional guidance on the use of the different uAdventure systems within mini-games, including two-way communication, the use of the narrative engine, and the use of the tracker.

The uAdventure mini-games model allowed a group of students to develop an escape room-themed narrative game named "The Paranormal Mansion", teaching competencies related to computational thinking. To this end, the game focuses on the investigation of paranormal events in a mansion and intersperses different mini-games focused on developing specific computational thinking skills, including decomposition, pattern identification, abstraction, and algorithms. These skills would have been either not possible or practical to convey using only standard uAdventure mechanics.

The mini-games (Fig. 4) address different topics and visual styles and use Unity's three-dimensional capabilities, visual effects, and user interfaces. Also, all mini-games have three levels of difficulty. The implemented mini-games are: shapes and colors (3), which develops abstraction by representing numbers using geometric shapes and debugging by requiring players to use shapes to create a specific sequence of numbers; ethical hacking (4), where players must combine a set of cards with color bars to generate a new card with a specific pattern of color bars, by first identifying how cards combine their patterns and later using trial and error until they achieve the correct pattern; crazy rings (5), where players must rotate several groups of rings until the colored flags of each ring are aligned, by identifying the rotation patterns and decomposing the problem into smaller problems associated with each ring; electrician (6), where players illuminate a series of boxes by abstracting the pattern of changes that occur when interacting with each box, exercising both pattern identification and abstraction; and finally number maze (7), where payers must input a path of numbers that connects the blue box to the red box by following a series of rules, while both developing both algorithms that adhere to the rules and abstraction by representing the path with numbers.

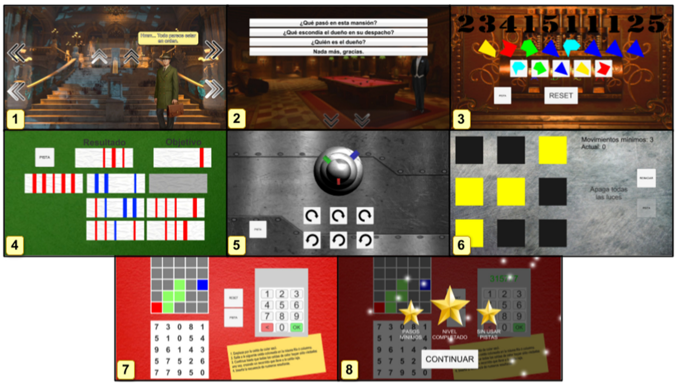

Fig. 4. "The Paranormal Mansion", a narrative game with integration of mini-games for the development of computational thinking skills. Images (1) and (2) show game scenes created with uAdventure in the point and click style with conversations and options. The rest of the images are mini-games: (3) shapes and colors (4) ethical hacking, (5) crazy rings, (6) elec-trician, (7) number maze. The image (8) shows the score obtained from one to three stars.

The interaction mechanisms used in the puzzles and their high number of states (and steps) needed to solve them, were the motivations for the students to develop them as mini-games. For example, in the crazy rings game (5), due to the large number of buttons and turning possibilities, the titular rings can reach a large number of different configurations, making it extremely complicated to simulate this behavior exclusively through narrative mechanics.

In this case, in addition to using its capabilities as an engine, the students take advantage of the different uAdventure systems so that, at the end of the execution of the mini-games, they can store a score which ranges from 1 to 3 stars. This score is then used in the credits screen and in the different analytics that the game generates. In addition, each time a mini-game ends, students made use of uAdventure's narrative engine to create appropriate conversations.

During the development of this game and its associated mini-games, students used uAdventure's learning analytics system to create and send traces that would allow the progress of players through mini-games to be examined. Their goal was to detect possible problems in the difficulty of the games and the progress of levels, and to find possible improvements for future iterations. To this end, the students sent enough traces from the mini-games to be able to recreate the players' actions during the completion of the mini-games; as well as to measure the progress/completeness of the mini-games themselves.

After the game was played by 8 users, all the traces generated by the mini-games were analyzed in order to compare their complexity based on the number of steps required to complete each of them. In Fig. 5, we can observe how the electrician mini-game was particularly complex, closely followed by the crazy rings mini-game and the one on ethical hacking. These results corroborate feedback from the players, who had already pointed out that these mini-games could be too complex. In addition, an unexpected result that was not pointed out by the players was that the complexity in the levels was

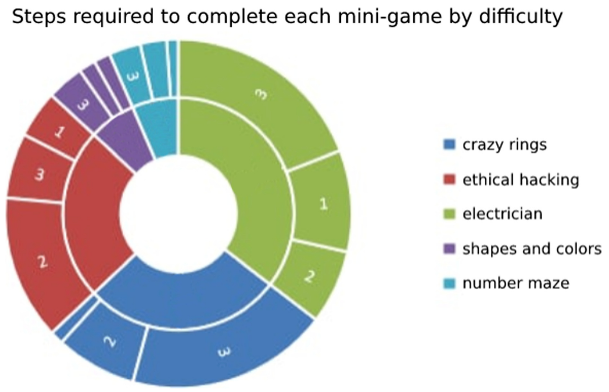

Fig. 5. Comparison of the number of steps required to pass a difficulty level in one of the mini-games. The legend shows the different mini-games names. In the graph, the inner layer represents each mini-game, while the outer layer represents each difficulty level in each mini-game.

not progressive in all cases, for example in the electrician mini-game, level 1 requires on average more steps than level 2.

4 Conclusions and Future Work

The extension model based on mini-games presented in this article increases the flexibility of the narrative model implemented in uAdventure. This versatility allows, on the one hand, to include new game mechanics to address skills that were previously very difficult to address within the narrative model. On the other hand, this approach enables a new development model where game developers can benefit from narrative aspects while, at the same time, making greater use of the full capabilities of the Unity platform.

The mini-games are built using uAdventure's native capabilities, such as the narrative engine or learning analytics. In particular, the use of the learning analytics capability is especially relevant for the evaluation of the capabilities and skills to be trained in the mini-games, in line with uAdventure's philosophy of providing analytics by default and with little effort. As seen in the case studies, easy access to analytics also facilitates the development and validation of the mini-games.

The proposed extension model has been validated in two case studies with game development students, where two serious games have been developed in which mini-games have played major roles. In these two serious games, the mini-games allowed cognitive skills to be addressed such as space management or computational thinking, which would have been difficult to tackle using pure narrative elements. We consider that these two case studies validate our proposal, but do not fully exploit its potential. Some of the mini-games developed in the case studies could, with minimal modifications, be reused in other games as new uAdventure game components.

Currently, uAdventure games can be reused (i.e. it is possible to reload a game into the uAdventure editor), with the aim of customizing the educational experience. For example, by adapting the resources or texts to the group of students who will use it. This has been one of the notable features of uAdventure and we believe it can also be applied to mini-games. As future work, we are working to be able to package and exchange mini-games (Fig. 6) through Pumva, a repository of games and mini-games created with and for uAdventure. For ease of use, a mini-game will be packaged together with a descriptor of the input and output variables configuration, as well as its different analytics traces. Using this descriptor, the uAdventure editor will display a friendly interface through which to configure the imported mini-game. On the other hand, we also want to improve learning analytics, with the goal of offering default analytics also for mini-games; and to offer templates for the xAPI traces generated by the mini-games, thus complying with xAPI's goal of keeping traces/statements self-contained. The traces will also include information about the context in the game where the mini-game has been integrated, further simplifying certain analysis tasks. Thanks to this new extension model, the separation of responsibilities between game and mini-game developers will allow educational experts to create games in uAdventure using mini-games in an even simpler way.

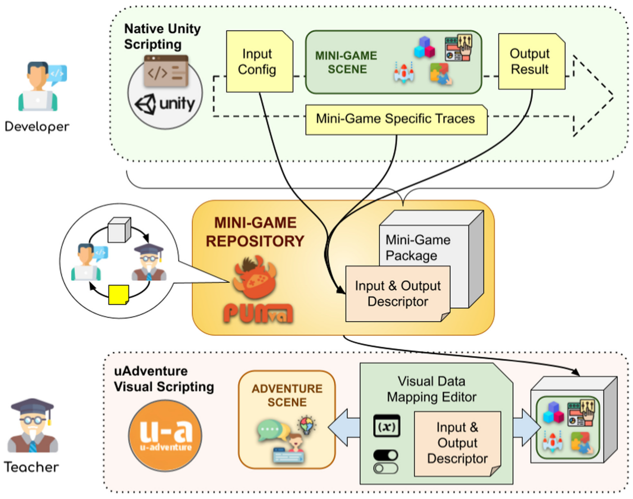

Fig. 6. uAdventure mini-game exchange model through Pumva. The mini-game must be packaged together with a descriptor that identifies its input and output parameters and ana-lytics traces.

Acknowledgements. This work has been partially funded by Regional Government of Madrid (eMadrid S2018/TCS4307, co-funded by the European Structural Funds FSE and FEDER), by the Ministry of Education (TIN2017–89238-R, PID2020-119620RB-I00), by MIT-La Caixa (MISTI program, LCF/PR/MIT19/5184001) and by the Telefonica-Complutense Chair on Digital Education and Serious Games.

References

1. Susi, T., Johannesson, M., Backlund, P.: Serious games – an overview. Elearning **73**(10), 28 (2007)
2. Michael, D.R., Chen, S.: Serious Games: Games that Educate, Train and Inform. Thomson Course Technology (2006)
3. Mitchell, A., Savill-Smith, C.: The Use of Computer and Video Games for Learning: a Review of the Literature, Learning and Skills Development Agency, London (2004)
4. Center for Technology Implementation in Education: Learning with Computer Games and Simulations. American Institute Research (2014)
5. Dickey, M.D.: Game design narrative for learning: appropriating adventure game design narrative devices and techniques for the design of interactive learning environments. Educ. Technol. Res. Dev. **54**(3), 245–263 (2006)
6. Amory, A.: Building an educational adventure game: theory, design and lessons. J. Interact. Learn. Res. **12**(2), 249–263 (2001)
7. Baptista, R., Coelho, A., Vaz de Carvalho, C.: Relation between game genres and competences for in-game certification. Significance **13**(6), 28–35 (2016)
8. Pérez-Colado, V.M., Pérez-Colado, I.J., Freire-Morán, M., Martínez-Ortiz, I., Fernández-Manjón, B.: Simplifying the creation of adventure serious games with educational-oriented features. Educ. Technol. Soc. **22**(3), 32–46 (2019)

9. Pérez-Colado, I.J., Pérez-Colado, V.M., Freire-Morán, M., Martínez-Ortiz, I., Fernández-Manjón, B.: e-learning standards in game-based learning? In: 21th International Conference on Advanced Learning Technologies (ICALT 2021), pp. 81–82 (2021)
10. Hauge, J.B., et al.: Implications of learning analytics for serious game design. In: IEEE 14th International Conference on Advanced Learning Technologies (ICALT 2014), pp. 230–232 (2014)
11. Serrano-Laguna, Á., Martínez-Ortiz, I., Haag, J., Regan, D., Johnson, A., Fernández-Manjón, B.: Applying standards to systematize learning analytics in serious games. Comput. Stand. Interfaces **50**, 116–123 (2017)
12. Bellotti, F., Berta, R., De Gloria, A., Primavera, L.: Adaptive experience engine for serious games. IEEE Trans. Comput. Intell. AI Games **1**(4), 264–280 (2009)
13. Barbosa, A.F.S., Pereira, P.N.M., Dias, J.A.F.F., Silva, F.G.M.: A new methodology of design and development of serious games. Int. J. Comput. Games Technol. **2014**, 1–8 (2014)
14. Perez Colado, I.J., Perez Colado, V.M., Martinez Ortiz, I., Freire Moran, M., Fernandez Manjon, B.: Simplifiying serious games authoring and validation with uadventure and SIMVA. In: 21th International Conference on Advanced Learning Technologies (ICALT 2020), pp. 106–108 (2020)