Entertainment Computing 38 (2021) 100429

Contents lists available at ScienceDirect

Entertainment Computing

journal homepage: www.elsevier.com/locate/entcom

A novel UML-based methodology for modeling adventure-based educational games☆

Rafael P. De Lope [a], Nuria Medina-Medina [a,*], Matías Urbieta [b], Alejandra B. Lliteras [b,c], Antonio Mora García [a]

[a] University of Granada, Granada, Spain
[b] National University of La Plata, La Plata, Argentina
[c] CICPBA, Argentina

ARTICLE INFO

Keywords:
Computers and education
Adventure-based educational games
Design tools and techniques
Modelling language

ABSTRACT

In the last years, serious games have been successfully exploited in different areas. However, in spite of the powerful tool that this kind of video games has proved to be in the classroom, a few methodological efforts have been conducted in order to involve pedagogues or educators in the design loop of these games or, one step further, include students in the co-design process to promote learning through design as a context. With this objective, this paper presents a set of metamodels that could facilitate the conceptual design of this type of games. To this end, a complete graphical notation based on the UML standard (with adaptations) is defined for representing the components of this type of games; having in mind to improve the collaboration between the team of educators/students and the technical team during the design process. Finally, the design diagrams defined during the production of a specific serious game, titled Uranus, are illustrated, in order to show the feasibility of the proposal. In addition, a validation experience was conducted with educators and computer engineering students in order to test the value of the proposed graphic notation to design educational games.

1. Introduction

Nowadays the world of video games is one of the biggest entertainment industries and its growth will be increased in the next years, according to several market analyses [21]. This success is mainly due to its visual and interactive richness, which enables multiple ways of entertainment. In the same way, among the possibilities that digital games offer, serious games are an option that is growing in popularity [37]. Although, there is not a unique definition for serious game, sixteen years ago Zyda [59] and Michael and Chen [41] formalized the concept of this kind of games, followed by many others [1,39,57], and all these definitions agree on the term 'serious' refers to the purpose of the game and not to its contents (plot, characters, challenges, etc.). Thus, a serious game can be defined as a type of game in which the main objective is the player to get a true profit, further than just to have fun. Indeed, the enjoyment is used as a mean to obtain this benefit, which is normally focused on health or education [32] (among others application areas).

Precisely, there are many studies that highlight the benefits of using serious games in the classroom [27,49,32] as they aid to improve the spatial abilities, promote interactive learning, motivate learning through challenge or, even, can act as simulators (players can practice without real consequences). Due to these reasons, education is one of the most prolific areas of application of serious games [20,26,28] and the domain of interest of this work. Accordingly, the goal of this work is to facilitate the design of game-based systems, integrating non-technical personnel such as educators in the team, to create effective games that help students raise their own knowledge from the playing experience, given them the responsibility of solving a series of tasks in an effective way to progress in the game (and, consequently, in their learning).

At this point, the difficulty of developing educational games applications is an undisputed fact. When designing an educational game there are an abundance of design decisions that must be performed (platform, deployment, narrative, interaction, adaptation, evaluation, etc.) [15,40]. Oftentimes, these decisions are made unconsciously, in an

☆ This paper has been recommended for acceptance by Pierre Jouvelot. (Where Pierre Jouvelot is the name of the Handling Editor).

* Corresponding author.
E-mail addresses: rprieto@ugr.es (R. P. De Lope), nmedina@ugr.es (N. Medina-Medina), matias.urbeita@lifia.info.unlp.edu.ar (M. Urbieta), lliteras@lifia.info.unlp.edu.ar (A.B. Lliteras), amorag@ugr.es (A. Mora García).

https://doi.org/10.1016/j.entcom.2021.100429
Received 4 March 2020; Received in revised form 5 March 2021; Accepted 4 April 2021
Available online 15 April 2021

inappropriate order and they are not reflected anywhere. However, it would be better if these design decisions were consciously contemplated [9]. In this line, a software engineering methodological approach offers a framework to organize and document the enormous diversity of design alternatives that should be considered during the creation of the educational game. Game design is not a science, but common elements exist for each game and having a methodology that anticipates the steps of the game's life cycle simplifies the task [8]. Thus, the use of a design methodology is recommended to perform a complete analysis so that no aspect is omitted in the solution, and to reduce the video game designing difficulty, considering that this difficulty is increased in the case of educational serious games [2,30]. Additionally, this design involves a multidisciplinary approach [35] and the complexity of managing a multidisciplinary team to develop the game [24], it is added the necessity of working with a team of stakeholders in the serious topic (for example, educators and students in educational games). In addition, a methodology could ensure compliance with a set of desirable design principles, getting a better ludic-educational balance.

On the other hand, at the methodological level, the game genre determines the procedure for generating the product, as well as the role that the members of the team must play to conduct their tasks; existing a clear correspondence between the game genre and these characteristics [48]. For this reason, the work in each iteration, phase or activity defined in the game's design methodology must be described according to its genre. In addition, the game genre must be aligned with the target concepts and their associated teaching methods [2]. Hence, due to its flexibility and extensive use of narrative, the graphic adventure seems to be one of the most adequate genres to address the creation of educational games [42,56] and its special characteristics must be considered in the methodological approach to generate educational serious games. However, in spite of the popularity of educational games and the benefits they offer, especially the educative graphic adventures, there are not many contributions focused on the development of a specific design methodology for this type of serious games [12]. With this aim, in [12] authors proposed a design methodology specific for adventure-based educational games, where a graphic notation was suggested to reduce the 'communication barrier' than often exists between the educationist and technical teams. Subsequently, authors matured the idea of using UML [55] to model the educational adventure and in [14] a correspondence between the elements defined in the proposed design methodology and the UML diagrams more appropriated to their representation was outlined. Nevertheless, the application of that proposal to a broader base of games revealed the need to adapt and extend the original UML to accommodate the special characteristics of the educational games. As a result, a new graphic notation based on UML has been defined, which is presented and formalized for the first time in this article.

Thus, the main contribution of this work is to propose a graphic notation language, based on UML, to assist the design of adventure-based educational games. The principal advantage of choosing this language over other graphical notations is that it is simple, popular and easy to use. UML is a standardized language that could be understood by everyone and it is also based on graphics which simplifies the task of the user [23]. Therefore, it can enable a co-design procedure, carried out by a multidisciplinary team. UML can be an adequate instrument to help programmers and analysts to 'speak' the same language as educators/students or people unfamiliar with software engineering [50]. This is also supported by the conclusions reached by [18], where nearly 200 software professionals and their customers were surveyed; concluding that "UML should not be considered exclusively as a language for software professionals" and evidencing that its use is not only for software engineering.

For all the above, the UML-based graphic notation proposed to model adventure-based educational games could be used to involve educators and students in the game design; what is necessary to get the game to provide students with opportunities to build their own understanding instead of only embedding lessons. In fact, this integration of educators and students in the design process enables both instructional (students use the game) and constructivist (students make the game) learning models [33], allowing to exploit the computational thinking in education [36]. In order to facilitate the interpretation of it, the notation has been defined at a syntactic and semantic level and enriched with metamodels that make explicit the structural elements that make up the game and its relationships. In addition to the proposed graphic notation (main contribution of this work), other contributions of the work are the methodological approach to apply the notation, the use of a use case that illustrates the application of the notation and a validation experience conducted with educators and students to evaluate the understanding and usability of it.

The rest of the paper is structured as follows: Section 2 describes related works studying other graphical approaches to the design of serious games, mainly educational games. Section 3 details the proposed graphical notation. It also describes, as a use case, its application to the design phase of a real educational game. Section 4 presents the validation of the usability of the proposal, which has been evaluated by educators and computer engineering students. Finally, in Section 5 the main conclusions, implications, limitations and future lines of the work are stated.

2. Related work

It's nothing new that more and more educators are betting on using serious educational games. "Games expose players to deeply engaging, visually dynamic, rapidly pacer, and highly gratifying pictorial experiences that make almost any sort of conventional schoolwork seem boring by comparison [22]." One step further, some authors argue that for a successful integration of this technology in the classroom, the convergence of three types of knowledge is necessary: content knowledge, pedagogical knowledge and technological knowledge. Educators (also named pedagogues) already possess the two types of knowledge; therefore, if the technological barrier is reduced, they could have great potential to create their own educational games. According to it, in [4] an experience was conducted, where a set of educators created their own games assisted by a software tool (after a process of technological formation). During this task, they followed a simple process that began by identifying the topic for the game and followed with the definition of the flow of events, the narrative and the characters to generate storyboards that were the basis for implementing the game. The educators acted as alpha and beta testers [29], but "overtime game development for these educators became cumbersome".

Still further, some authors propose that, according to a constructivist approach [44], the students themselves are who design the educational game to improve their own knowledge and create new cognitive relationships by connecting a new experience to a prior experience. That is, "making games for learning instead of playing games for learning" [34]. With this aim, in [5] an incremental model with six layers is proposed. Each layer deals with a conceptual element related to the design of the educational game: identity, immersion, interactivity, increasing complexity, informed teaching and instructional. From a software engineering perspective, these frameworks are insufficient since they move in a conceptual plane rather than at a level of technical design or production. But the experiences where educators and students collaborate to create games [38] serve to support the thesis that the design and creation of educational video games is not an exclusive competence of engineers and that the existence of specific methodologies for educational video games that facilitate the integration of the non-technical team is a necessity.

On the other hand, UML is a general-purpose language for specifying and visualizing complex software by means of graphic notations, which has widely proven to be valid for large projects [47]. UML is a most-used and flexible language where you need to know only the part of the language that you are using; it is not necessary to learn a formal notation

and the complexity is ascending [23]. Therefore, as argued in the introduction, it has demonstrated its feasibility to be used by technical personnel and also by non-technical personnel (such as educators and students in educational video games). In addition, the benefits of using UML to model games have been previously stablished by other authors [17,7,23,3]. However, there are not many studies in the scientific literature describing a systematic and comprehensive methodology for developing educational games, and even there are fewer works using UML or other graphic notation as a tool to facilitate the complete design of the adventure-based educational game, which is the intention of the present proposal. To illustrate this statement, the main related works using UML to model games are analysed below.

The reviewed works have been classified in three categories: UML to describe the game architecture, UML to model the game requirements and UML to model the game elements. None of the works revised uses UML to create and document the structural design of the complete game, which is precisely the contribution of this work: a set of UML-based metamodels and graphical notations to generate the set of game's structural diagrams, which will be obtained during the design process according to a methodological approach for educational games. Due to the special characteristics of the educational game, our proposal requires adapting UML to be more expressive. This is also a differentiating feature of our work compared to most of the reviewed works where UML is applied to model the game (like any other software system) without adaptations for games. Other differentiation is that our proposal explicitly defines the association between educational competencies and game elements (scenes, actions/dialogues, etc.).

In the line of the first category, Hu's work [30] defined a *general architecture* for educational games (EGA, Educational Game Architecture), focusing on the genre of adventure (Sim-Eduventure). This proposal used UML to describe the architecture, which is basically divided into four subsystems: subject, activities, tasks and feedback. This high-level architecture intended to serve as a guideline for all the stakeholders in the development of an educational game. However, the work did not define the game design process itself whereas our approach also clarifies the design steps (design of acts, design of scenes, design of scenarios, etc.). Following this architecture-based proposal, Amaral et al. [2] applied EGA to a different genre: strategy and simulation. Thus, it was necessary to modify the initial EGA architecture to adapt it to the new genre; specifically, they changed the Game Flow or Game Cycle [25]. In the same line, a later proposed architecture was used to develop a simulation video game, Rural Management.

According to the second category, Cooper and Longstreet [10] proposed the use of UML to model the *requirements* of a serious educational game. To this end, they first divided the video game into acts (a game has one or more acts), scenes (an act has one or more scenes), screens (a scene has one or more screens) and challenges (a screen has associated an optional challenge); and then, they applied an adaptation of use cases and tabular specification templates to model every act of the video game. Their main contribution is the creation of a tool, named SimSYS, which allows loading a XML file to test the design game [11]. This is very interesting for video games that are not of a large wingspan (not very complex in story/acts), as it permits going through three steps: (1) an informal stage, where basically the narrative and the storyboard are included, (2) a semi-formal step, where the requirements are specified in UML, and (3) a formal phase, in which the XML is obtained from UML (tailored UML Use Case). This XML can be loaded into their own game engine, SimSYS. The authors applied the proposed notation to the design of a particular serious game, as a test of its utility and value. This work has several points in common with the proposal made in this article (for example, the structuration of the game in acts and scenes); however, it operates with a different abstraction level, since it was focused on the requirements analysis rather than on the design phase. In addition, they only used the UML use cases, restricting the UML's potential to represent other aspects of the game, such as certain actions and dialogues.

On the other hand, in [6] it is proposed to use the methodology MDA (Model-Driven Architecture) [43] to create a metamodel, based on the standard e-learning IMS-LD [31], to integrate serious games and e-learning environments. Additionally, the metamodel was implemented using an authoring tool, which allows the educators to define models that represent their requirements. Although this approach has some highlights aligned with this article, such as the facilitation of the communication between designers and educators, or the easy inclusion of the pedagogical and ludic aspects of the game, their model is also focused on the requirements phase. For its part, Dowd's approach [19] recognized the importance of using UML and, particularly, building activity diagrams to improve the design of the initial stage of the serious game, using the CAT method (Creative Agents and Triggers) for the elicitation of game's domain knowledge. CAT is not a methodological approach, although it aims to improve the communication during the design of serious games. However, there is no adaptation of UML and some important elements of the game such as the dialogues, characters or educational challenges are not considered. Additionally, this work differs from the objective outlined in our proposal because it did not focus on educational games.

Finally, in the third category (where we work would be included), we can find a few works that model a specific genre of game to later facilitate the design of games of this type. For example, in Tlili et al. [54] a generic UML class diagram for educational role games is proposed, which is instantiated for three different games showing its feasibility. This model represents the main elements of role games (classes: character, mission, combat, etc.) and the game activities (association relations: select, collect, win, etc.), being useful for those who wish to design a new game of this genre. Classes "Learner" and "Skills" are used in that class diagram to represent the educational component of the game. However, a correspondence between playful challenges and educational challenges/competences is not made explicit, nor the complete structure of the game is addressed. In the same line, Taylor et al. [53] did not either focus on educational games, but their proposal is very remarkable since it used UML and pseudocode to facilitate the design of video games, focusing on the game-flow inside the scenes to design computer game levels. The proposal uses game-flow diagrams (extending and adapting use-case diagrams), which show the game objects with which the game player's character must interact. However, it does not contemplate the educational part of the game nor does it address the complete structural design of it.

Thus, our work comes to fill the aforementioned gaps in the design of the educational serious games. To this end a graphical notation, which includes diagrams to generate and document the artefacts in the conceptual design phase, is presented. These design artefacts, which are integrated in a methodology to design adventure-based educational games [13], are defined as an adaptation of UML diagrams and are based on a set of formal metamodels. The structure of the game is explicitly detailed in the models constructed according to these metamodels. The proposal ensures a balance between the ludic and educational parts, since it is possible to objectively measure the number of ludic and educational challenges. The diagrams are also used as a tool to facilitate the communication between all the stakeholders (clients, designers, programmers, analysts, and educators) involved in the development of an educational game. Consequently, the main research questions addressed in this work are four:

(1) Is it possible to model the complete structure of an adventure-based video game using UML-based diagrams? I.e. is it possible to define appropriate diagrams for specifying the acts, scenes, and scenarios of an educational game, including actions and dialogues? This issue is important because getting the overview of the game is a vital design task to develop successful games.
(2) Is it possible to include the educational purpose of the game in these UML-based diagrams to reflect the skills worked in the game actions? This is not a trivial goal either as many serious games do not have this necessary educational-ludic balance.

(3) Are the proposed diagrams usable and understandable for computer engineers and educators? This issue is crucial to facilitate the co-design of the game.
(4) Does the proposed approach present benefits against other UML-based notations to model educational video games?

Through a real use case (questions 1 and 2) and a validation experience (questions 3 and 4), these research questions will be answered in the following sections of the paper, where the proposal is described and formalized.

3. A graphical approach to design educational games

In this section, we introduce an approach for the design of educational games (especially suitable for educational graphic adventures) which relays on a graphical notation. The design diagrams used to model the game are integrated within a complete methodology previously proposed in [13]). The methodology establishes a five-stage life cycle: startup (type of video game is defined), design, production (tasks of programming, modelling of characters, objects, and scenarios), test (evaluation) and post-production (patches and/or updates to improve the game). In addition, in order to support the phases of startup and design, a comprehensive taxonomy for serious games has been proposed in previous works [15]. In the current work, the authors present a set of UML-based diagrams which could be used in any design process of an educational game; that is, the proposed graphic notations pretend to be useful regardless of the methodology.

With the spirit of facilitating the design of the game, the game is structured into acts, scenes, scenarios, actions and dialogues (theatrical metaphor of play) following a similar approach as Cooper and Longstreet [10]. Keeping in mind that structure, a set of graphic notations which adapt UML to fit educational video games is offered. From a higher to a lower level of abstraction, these artefacts (models and metamodels are: (a) the acts diagram, which shows all the acts in which a game is divided; (b) the scenes diagrams, a set of diagrams representing what happens inside each act (decomposed into scenes); (c) the scenarios diagrams, which present the transitions between the different places within each act, (d) the actions diagrams, which show all actions related to each of the scenes; and finally, (e) the dialogue diagrams, which specify the exchange of messages within each dialogue (being a dialogue a special type of action). It is worth mentioning that this structuration offers a cognitive structure that allows the integration of knowledge in an associative way, which means that the educational content is inserted into the plot and narrative of the game, so that it is easier to assimilate and fix it in the long term. The key elements supporting the proposal were defined in [13].

Consequently, the proposed approach implies an incremental design of the adventure-based educational video game. This design is divided into seven phases as detailed in Fig. 1. The main artefacts generated in each phase are the models representing the acts, scenes, scenarios, actions and dialogues of the game, which will be specified by means of the UML-based diagrams presented in the next sections.

Thus, authors formally define the graphic notation using UML metamodels that enables the syntactic and semantic correctness of models. According to the structure specified for the game, in first place, different UML metamodels were defined with its corresponding semantic description to support each design model (acts diagram, scenes diagram, etc.). In second place, authors have based the graphical notation on the UML standard so that the reader with UML experience can find them friendly. However, in some exceptional cases, it was required to perform some extensions, modifications and simplifications in order to adapt UML to educational concepts and vocabulary of the game.

3.1. Use case

Furtherly, as a case of study, authors will use the game titled *Uranus: The invaders of Times* to illustrate the application of the proposed graphic notation (research questions 1 and 2). This educational game was developed in 2017, and has a mobile version available for Android platform[1] and other for PC.[2] A team of researchers including some of the authors of the present work designed the game, which was developed by the Spanish company Greyman Studios. Authors modelled the complete structure of the video game (research question 1) using the proposed UML-based diagrams and they had no difficulties to represent the scenes or to include the educational component in them (research question 2). The effectiveness of the game was tested with a sample of 237 Spanish children, obtaining satisfactory results [58]. For example, an average value of 3.83 (out of 5) was obtained about the general satisfaction with the game (gameplay), only 24% participants identified some aspect to improve in the game and only 8% of the children detected the serious purpose of the game (ludic-educational balance).

Uranus is a 2D adventure video game with a point & click interaction designed for practicing Spanish reading comprehension (for this reason, the entire narrative of the game is in Spanish language), having as target user children aged between 7 and 9 years. The game follows the story of a child on who it depends the future of planet Earth. In order to save the planet, the player must travel back in time and meet some historical characters (e.g. Cleopatra, Julius Caesar or Marco Polo) to get several objects of historical significance. Finally, the player must deliver these objects to the aliens of planet Uranus, for the aliens to clone the historical characters through these objects instead of kidnapping them (as it was their intention).

The game is divided in four periods, so the player must perform four time jumps: Ancient history, Middle ages, Modern history and Contemporary history (although currently not all periods are operational). Within each period, the player visits different locations (e.g. Egypt or Rome in the Middle ages) where he/she must overcome a lot of challenges of reading comprehension using resources such as inference or deduction. Everything that happens in one place, during a period, has been called a *mini-story*. Fig. 2 shows four screenshots of the game (mini-story of Rome). From left to right and top to bottom: the avatar ("Boy") is speaking with an old man in the countryside ("Repair chariot" scene); he is trying to fix the chariot ("Repair chariot" scene); he is in prison with Julius Caesar ("Visit Julius Caesar in prison" scene); and he is in the Roman senate about to start talking with one of the senators ("Visit the Senate in search of Crassus and Pompey" scene).

The educational key of this adventure is that during the game experience, the player is not a passive receiver of ready-made knowledge, but builds his own knowledge by means of the resolution of problems, according with constructivist approaches defended by other authors [5]. In addition, the integration of the educational skills (including acquisition of knowledge) into ludic challenges makes students not realize the learning embedded content, allowing them to connect and engage in an active and implicit process of learning.

The following subsections present the different types of diagrams proposed for the design and modelling of educational games. Uranus's design models instantiate the conceptual elements defined in the metamodels.

3.2. Acts diagram

The acts diagram represents the complete story of the game, which will be structured in acts, at the highest level of abstraction. This type of diagram is based on UML activity diagrams but with some adaptations (explained below) to facilitate the design of the adventure-based educational video game.

[1] https://apps.ugr.es/informacion/apps/invasores
[2] http://bios.ugr.es/~nmedina/

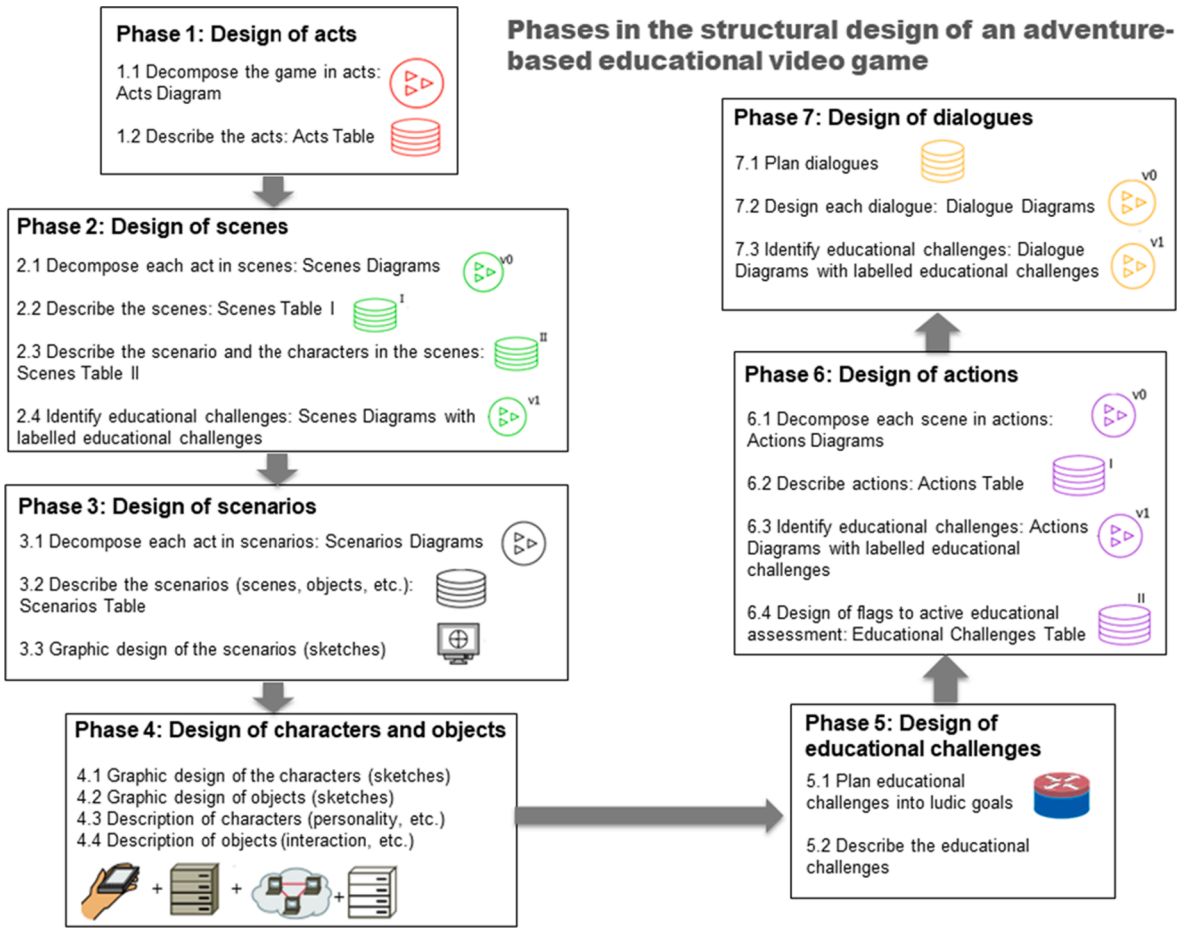


Fig. 1. Approach for the structural design of an adventure-based educational video game.

Fig. 2. Screenshots of Uranus game.

3.2.1. *Metamodel of the acts diagram*

Fig. 3 shows the metamodel of the acts diagram which describes the elements present in every acts diagram and the existing relationships between them. The representation of the metamodel is made using a UML class diagram, where there are five types of meaningful objects (classes, which are represented with rectangles): serious game, act,

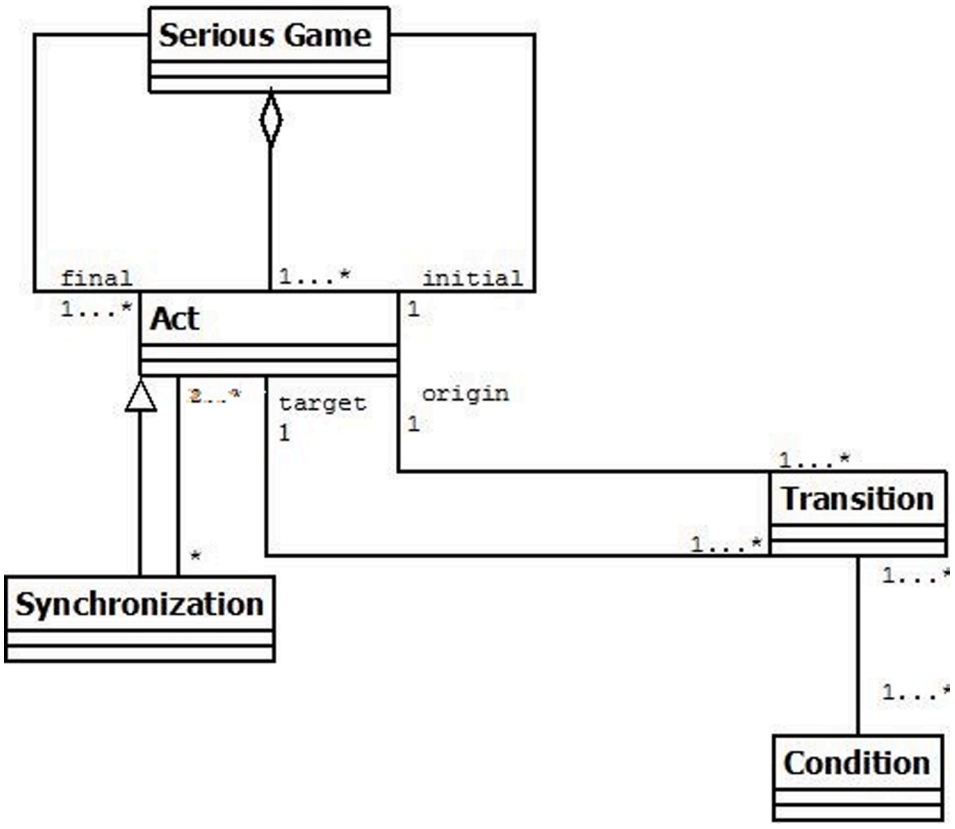


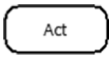
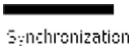
Fig. 3. Metamodel of an acts diagram.

transition, condition and synchronization; and three types of connexions between these classes of objects (relationships, which are represented with lines): aggregation (a serious game is composed by acts), heritage (a synchronization is a type of act) and associations (a serious game has an initial act and one or several final acts, a transition has an origin act and an target act, etc.).

Semantic: An *Act* is defined as the highest-level element used to structure the video game in self-containing parts. At least, there should be one act in the video game, but normally, there will be several. During the design of the acts diagram, the designer has only a sketchy idea about how the video game will be, so the scenarios, characters, dialogs, etc. are not completely defined. In the same way, based on this, educators could think about how to balance the educational competences they want to achieve between the different acts, which will be later specified in the scenes diagram of each act. The *Transitions* help defining how the player can navigate the acts, specifying a source *act* and a target one. Therefore, there will be an initial act and one or more final acts in the video game, and a sequence from the opening to the ending acts, moving through intermediate ones defined by *transitions*. As it happens in UML, *Synchronization* is a specialization of Act class used to synchronize several acts. This synchronization means that all the acts must be completed before the flow of the story can continue. Moreover, one or more *Conditions* could be required in order to perform a transition. Conditions can be expressed using the language for the formal description of expressions in UML (OCL) although also using natural language or other language previously agreed by the stakeholders.

Graphical notation: The graphical notation is borrowed from the UML Activity diagram. In Table 1 we define the mapping between UML Activity diagram elements (Activity, transition, Fork & Join, etc.) and the Acts metamodel elements (Act, Transition, etc.). Thus, for each element of the acts diagram (first column) a graphic representation is provided (last column). This notation is based on the activity diagram of UML (second column). However, the UML terms have been changed to adapt

Table 1
Graphical notation of the acts diagram.

Element in the acts diagram	Base UML element	Adaptation of UML	Graphic representation
Act	Activity diagrams → Activity	*Simplification:* An act is an Activity node fused with its Activity edge *Name change:* The union of Activity node plus Activity edge has been renamed as Act	
Synchronization	Activity diagrams → Control nodes → Fork and Join nodes	*Name change:* Fork and join nodes has been renamed as Synchronization *Simplification:* Simultaneity is not required	
Condition	Activity diagrams → Control nodes → Merge node	*Name change:* Merge nodes has been renamed as Condition	
Final/Initial	Activity diagrams → Control nodes → Final node/ Initial node	None	
Transition	Activity diagrams → Control flow	*Name change:* Control flow has been renamed as Transition *Simplification:* Behaviors 'selection', 'transformation' and 'pin' have been eliminated	

them to the vocabulary of the game world (third column). Additionally, several elements have required extending or simplifying the corresponding original UML definition in order to adapt it to the needs of the design of adventure-based educational video games (third column). Thus, the second column shows the original UML element which has served as a basis for defining the proposed graphic notation whereas that the third column describes the adaptation performed in each case. For example, the act is simplified since only the activity node is used (fusing it with the activity edge for the sake of simplicity) and the transition has been also simplified, by removing some behaviours as the use of pins, since these are directly linked to the management of objects, in this case, the flow of objects. In addition, fork and join nodes have been renamed as "Synchronization" and the meaning of this element has been relaxed since concurrence is not required. This means that the acts entering in the Synchronization bar must all occur before advancing, but they will not occur simultaneously (unless it is a multiplayer game this would not be possible).

In addition, there is a supplementary table associated with the acts diagram. It has two columns: the first one indicates the name of each act of the game, and the second one shows a brief description of it. In this supplementary table, educators could include notes (in natural language) on the pedagogical approach, the competences and the educational contents that will be worked within each act.

3.2.2. Acts diagram in Uranus

According to the metamodel, the diagram in Fig. 4 shows that Uranus game is composed of nine acts. The initial act (*Introduction*) and the final act (*The end*) are annotated, respectively, with the initial and final UML nodes. Once the third act (*Time warp*) has been completed, the player can choose a mission to perform, travelling back in time to Egypt, Greece, Rome or Granada. Then, he/she must choose another one and so on. Once all these mini-stories are completed (synchronization requires that all acts delimited between the two bars be performed), the player can move on to *Back to the Alhambra* act before the final act.

This visual representation facilitates the understanding between designers and educators, providing a mental model of the game at a high level of abstraction on which both can discuss. In addition, on this basis, educators can decide, for example, to work more deeply on critical comprehension and *meta*-comprehension in the last two acts, while inferential comprehension skills are practiced mainly in the four central acts (these and other considerations will be noted in the supplementary table).

3.3. Scenes diagrams

The scenes diagrams represent the complete story of each act, i.e. as in theatre plays, there will be a presentation, development and denouement scenes for each act. In order to facilitate the complex task of designing the scenes, the concept of sequence is introduced. A sequence is a set of scenes which must take place in a predefined order. The sequences can be 'navigated' by the player in different ways, depending on the defined conditions and synchronizations. For the sake of clarity, the metamodel has been separated in two UML class diagrams in Fig. 5. An scenes diagram is very similar to an acts diagram in its structure; consequently, it will also be represented using an adaptation of the UML Activity diagram.

3.3.1. Metamodel of the scenes diagram

The following metamodel (divided in two parts: Fig. 5 left and Fig. 5 right) describes the elements present in the scenes diagrams, and their relations.

Semantic: Each *Act* is divided into *Sequences*, which subdivide the scenes diagram, and every one of these is composed of *Scenes*. So every act must have at least one sequence and this, in turn, must have at least one scene. A scene will include a series of actions that are performed by a set of characters on one or more scenarios (the scenarios and characters involved in each scene will be defined in a supplementary table). Actions will be described in the actions diagram for each scene. Conceptually, a sequence has the same elements as an act in the acts diagram (as it can be seen in Fig. 5 left), also sharing the same syntax and graphical notations. With regard to scenes, a *Transition* involves a change of one scene to another, following the defined sequence. A *Condition* can be associated with a transition as a premise to change the scene. *Educational competences* represent a set of skills and attitudes in a particular context or concrete curriculum. For example, the Spanish education system defines eight basic competences [51], some of which are: linguistic communicative competence, cultural and artistic competence or mathematical competence among others. Every scene can be associated with one or more educational competences. However the aim is not to define the concrete educational challenges, but just to analyze the number of competences, so that the team of pedagogical experts can distribute the educational work among the different scenes.

Graphical notation: Table 2 shows the representation of all the elements of the scenes diagram (Fig. 5), which is based on the UML Activity diagram. Some components have been modified from the original UML

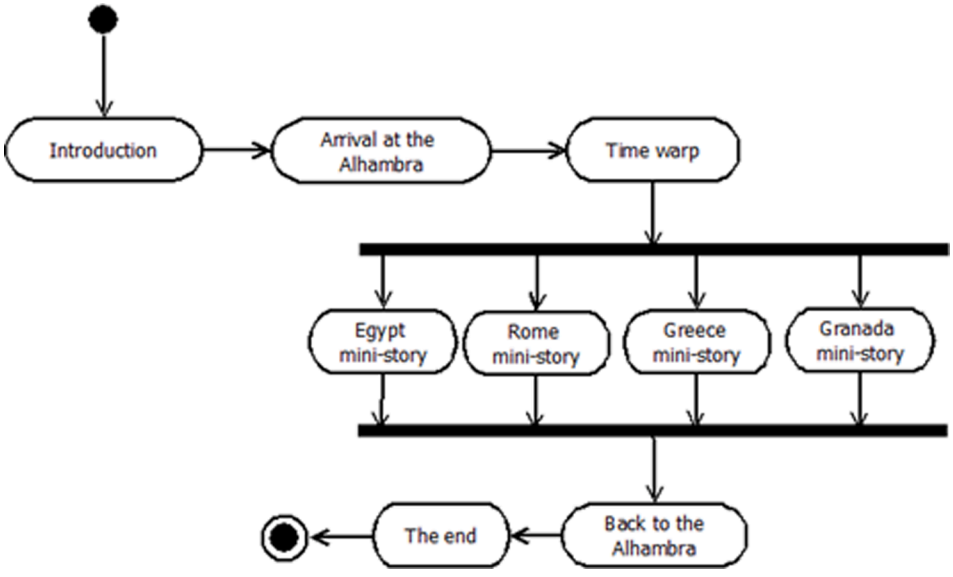


Fig. 4. Acts diagram in Uranus.

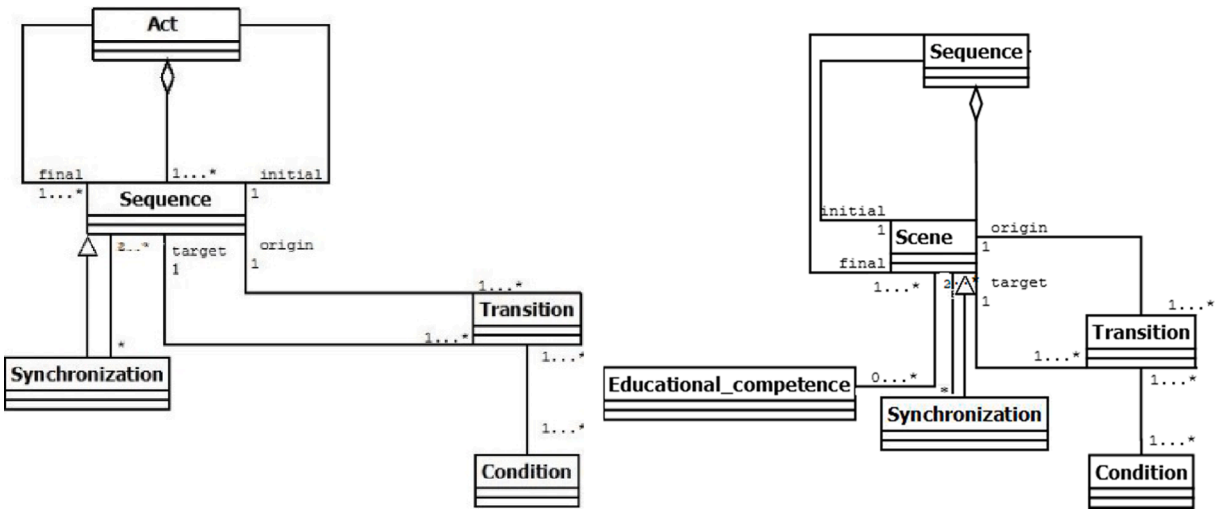


Fig. 5. Metamodel of a scenes diagram: relationship act-sequence and sequence-sequence (left); relationship sequence-scene and scene-scene (right).

Table 2
Graphical notation of scenes diagram.

Element in the scenes diagram	Base UML element	Adaptation of UML	Graphic representation
Scene	Activity diagrams → Activity	*Simplification:* An scene is an Activity node fused with its Activity edge *Name change:* The union of Activity node plus Activity edge has been renamed as Scene	Scene
Optional scene	Activity diagrams → Activity	*The same simplification and name change done for the Scene.* *Extension:* The symbol "O" is placed in the upper left corner to indicate optionality.	Scene
Sequence	None	*Extension*: Sequence is a new element (not included in the original UML)	Sequence
Educational competence	UML note	*Extension:* Using UML note to specify an educational competence	Educational challenge
Synchronization	Activity diagrams → Control nodes → Fork and Join nodes	*Simplification:* Simultaneity is not required *Name change:* Fork and join nodes has been renamed as Synchronization	Synchronization
Condition	Activity diagrams → Control nodes → Merge node	*Name change:* Merge nodes has been renamed as Condition	Condition
Final / Initial	Activity diagrams → Control nodes → Final node/ Initial node	None	Final Initial
Transition	Activity diagrams → Control flow	*Name change:* Control flow has been renamed as Transition *Simplification:* Behaviors 'selection', 'transformation' and 'pin' have been eliminated	Transition

(similar to how it was done with the acts diagram). In addition, as stated, two new elements are added: *Sequence* and *Educational competence*. The first one is represented as a box with dashed line border, whereas the educational competence is represented as a UML note because it is an element that is easily associated with any activity. Finally, an additional notation has been added to mark those scenes that are not mandatory to move from this scene to the following one (which is very common in video games: scenes that may not be played because they are not essential to advance in the game). These scenes have been called *Optional scenes*.

The most interesting from an educational point of view is that educators can connect educational competencies with scenes on this diagram. This is possible since the proposed graphic representation facilitates the understanding of the structure of scenes of the game and it will be also important for the designers, who can obtain a fairly specific idea of the educational purposes pursued in each scene. This does not mean that it is not possible to design purely playful scenes.

Finally, the scenes diagram has associated a supplementary table with four columns containing descriptions of the act's scenes in natural language (which may be accompanied by sketches and images): the first column defines the scene, the second column details the scenarios where the scene is elapsed, the third column shows the main characters present in the scene and the four column serves to provide information on the educational competencies that will be worked in the scene. There is a supplementary table for each act and an entry in the table for each scene in the act.

3.3.2. Scenes diagram in Uranus

The diagram in Fig. 6 shows the scenes of the Rome act in Uranus game, where there are five sequences grouping the scenes. Sequence 3 was detailed in other diagram due to its high number of scenes. As shown in the first scene of Sequence 2, the boy (our character/avatar in the game) arrives at a prison (S2.1 - *Arrive to prison*) where he meet to Julius Caesar (S2.2 - *Visit Julio Caesar in prison*). In this second scene, the design team (advised by educators) labeled the educational competencies: LC1 (literal comprehension - ability to identify specific data in texts), IC1 (inferential comprehension - ability to infer non-explicit content by applying deduction) and GC14 (global comprehension - ability to understand discontinuous texts (i.e. tables, graphs, maps, images, etc.). The complete list of competences used in Uranus can be consulted in the competences' document[3] (in Spanish).

[3] http://bios.ugr.es/~nmedina/competencias.pdf

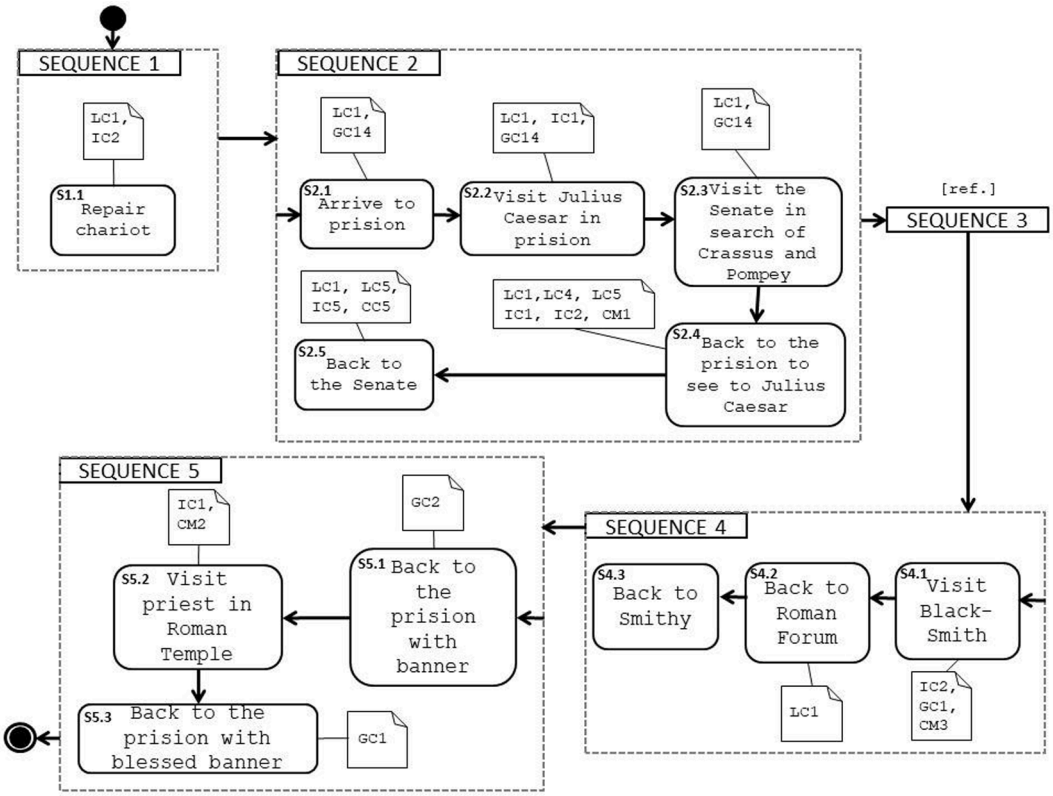


Fig. 6. Scenes diagram in Uranus.

The following three scenes in sequence 2 (which also have associated competences) are: the first visit of the avatar to the Senate looking for Crassus and Pompey (S2.3); come back to the prison to get evidence that the avatar has met Julius Caesar (S2.4); and finally, he returns to the Senate with a letter to Crassus and Pompey (S2.5). Then, the sequence 3 will be executed.

It is important to note that at this point, educators and designers do not have to define how the educational competences are going to work within the scene, only that they want to work these skills in that scene. However, if they want to point out an idea, they can do so in the corresponding supplementary scene table.

3.4. Scenarios diagrams

The scenarios diagrams represent the set of virtual places where the acts take place. This type of diagram is based on UML state diagrams since the change between scenarios is conceptually aligned with the change of states.

3.4.1. Metamodel of the scenarios diagram

The metamodel shown in Fig. 7 describes the elements present in the scenarios diagram of one act and the connections or interdependences between them.

Semantic: A *Scenario* is the location where an act is played, but there may be different scenarios associated with an act. Each scenario is composed of *interactive* and *static objects*. Each *Act* is performed in one or various scenarios, depending on the transitions (a transition can be triggered by a change of scene or for another reason). Thus, a *Transition* implies a change of scenario, and could be associated with one or more *Conditions*, which must be met before this change. Moreover, the transition could have linked a *Cutscene*, which can be a kind of effect (for example a cinematic) in which the player intervenes little or nothing at all or a mini-game where the player moves to another environment where he/she must solve a little complex subgame within game. Finally,

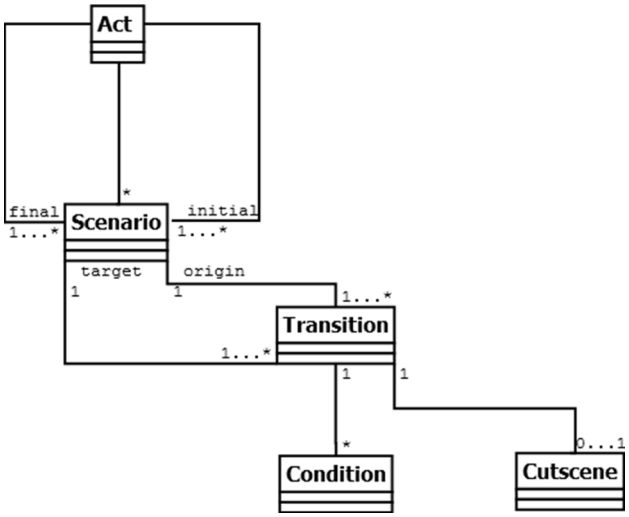


Fig. 7. Metamodel of a scenarios diagram.

the scenarios of the start or finish of an act are associated with a transition to the initial and final nodes, respectively.

Graphical notation: Table 3 shows all the elements of the scenarios diagram. There are no new elements. For its part, the *Cutscene* is represented as a UML note because it is an element that is easily associated with any state. In addition, the *Scenario* has been simplified by ignoring unnecessary elements of the UML state and just considering a UML simple state, which has been renamed as scenario. The UML state includes, for example, composite states and submachine states which present a complexity not necessary to model the scenario of a game and, consequently, it has been discarded.

Similarly to the scenes diagrams, the scenarios diagrams have

Table 3
Graphical notation of scenarios diagram.

Element in the scenarios diagram	Base UML element	Adaptation of UML	Graphic representation
Scenario	State diagrams → Simple state	Simplification: A scenario is a UML simple state (UML states are not used) *Name change:* Simple state has been renamed as Scenario	Scenario
Cutscene	UML note	*Extension:* Using a UML note to specify a cutscene	Cutscene
Final / Initial	State diagrams → Final State / Initial State	None	Final Initial
Transition Condition	State diagrams → Transition	None	[Condition] Transition

associated a supplementary table which provides a description of the game scenarios using natural language and images. This table has three columns: the first one defines the scenario, the second describes the existing objects in that scenario, including interactive objects and the static objects with semantic relevance, and the third column details the functions of each interactive object (how the object is used and what it is for). There is a supplementary table for each act and an entry in the table for each scenario where that act takes place.

3.4.2. Scenarios diagram in Uranus

Fig. 8 shows, as an example, the scenarios diagram of Rome act. This diagram includes the required conditions to enable the scenario transition, and several cutscenes (presented as interactive maps). For example, as it is shown in the figure, in order to move from *Country estate* to *Prison*, a necessary condition is *repair the chariot*. Subsequently, there is a map (cutscene) where the player must select that destination. In addition, the scenario *Country estate* has various interactive objects with different functions, which are described in the supplementary scenario table associated with the act. For example, one of them is a *hammer* that can be picked up by the avatar, and will serve to get a backrest of an old chair, or put the nails to fix the support of the chariot.

It should be noted that this type of diagram provides a dynamic spatial representation of the game easily understood by the design team (designers, educators, etc.) and that, like the rest of the diagrams, also facilitates communication with the development team. Specifically, this type of diagram allows visualizing the spatial dimension of the act, in terms of the places where it takes place and the conditions that must be met to move between these scenarios.

3.5. Actions diagrams

The actions diagrams represent the complete set of actions and interactions which are performed in each scene. This type of diagram is based on activity diagrams of UML due to its dynamic character.

3.5.1. Metamodel of the actions diagram

The metamodel in Fig. 9 describes the elements present in the actions diagram of one scene and the relations between them.

Semantic: Each *Scene* is divided into *Actions* to be performed on it. The actions describe the events which are part of that piece of the game story. For instance some actions could be: pick up objects, use interactive objects, or talk with a character (a dialogue is a type of action). Every action may be associated with one or several *Educational challenges*, which are addressed in that action. At the same time, each challenge serves to achieve a set of specific *Educational competences*. An educational challenge is any task or educational exercise designed by the team of educators that is masked in any action (or dialogue as a type of action), and which aims to address some competences of the student's curriculum. In this point, it is essential the collaboration of the teaching team and the technical team, since the action must be designed to meet the educational challenge in a playful way and in many cases the design of the educational challenge implies a redesign of the action and/or vice versa. In addition, each educational challenge has a *Progress state*. Three different states of progress are considered: initiating the challenge, working the challenge and assessing the challenge. For example, to work the empathy in children (a type of educational competence), the scene must present a situation (action) where something happens to a third person (initiating the challenge) and give the player the option to choose between several answers/actions (working the challenge); so that one (or several) of the possible alternatives supposes understanding of the feelings of that third person in that situation while the others not (assessing the challenge).

Like in the acts diagram, a *Transition* implies a change of action, and *Synchronization* and *Dialogue group* are types of actions. Synchronization and *Condition* are analogous to those already described in Section 3.2.1. The dialogue group will be described in Section 3.6.1. The *Flag* implies

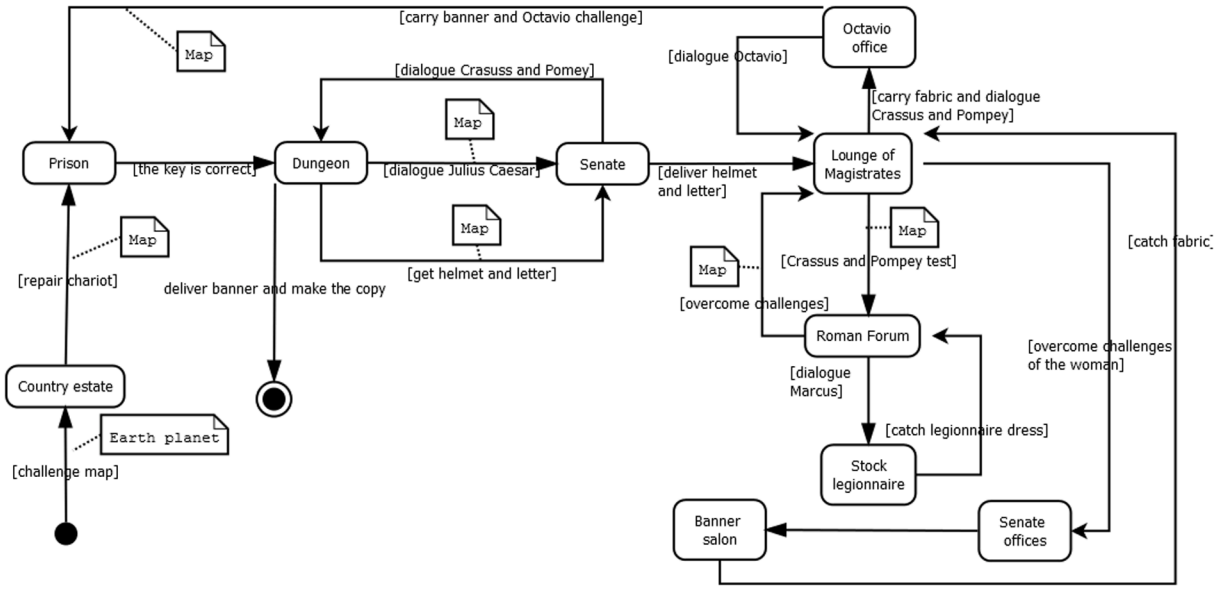


Fig. 8. Scenarios diagram in Uranus.

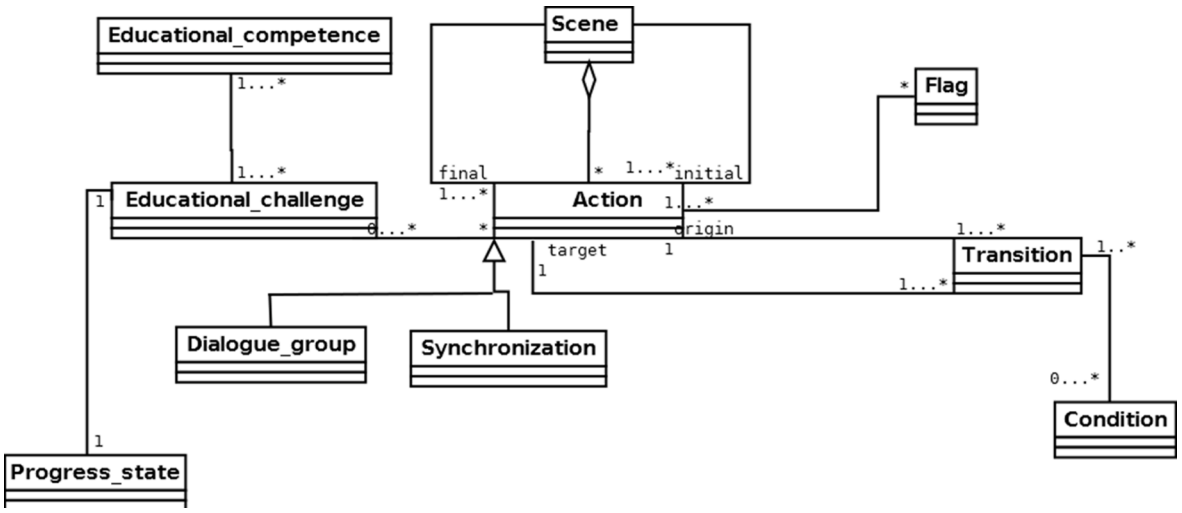


Fig. 9. Metamodel of an action diagram.

the change of state of one or more variables used to assess conditions. For example, it could just mean that a transition is activated, or the arising of a new element in the scene (such as a dialog, object or character).

Finally, the progress state of an educational challenge can be defined by the progress state of each one of its associated competencies. The identifier of an educational competence has three parts: the identification code of the competence, the progress state of the competence regarding that particular challenge and the identification code of the educational challenge. The progress state is labelled as *I* if the competence has been *initiated* in the challenge, *W* if the competence is being *worked, A* if the competence is being *assessed or* a combination of these states (concatenating the corresponding letters). Fig. 10 shows an example where the competence is LC3 (a type of literal comprehension), the state is working and assessing and the identifier of the educational challenge is 023. Each educational challenge is defined by the set of the associated educational competences using the mentioned identification system (examples are shown in Fig. 11).

Graphical notation: Table 4 shows all the components of the actions diagram. The elements *Action*, *Synchronization*, *Condition*, *Final and initial node* and *Transition* are based on the similar elements of the UML Activity diagram and there is a new element named *Educational challenge*. The educational challenge, as in previous cases, is represented as a UML note. Similarly, as in the acts diagram (and the scenes diagram), *Synchronization*, *Condition*, and *Action* modify their names with respect to UML. The action is simplified and only an activity node is considered (activity edge is discarded). Finally, the transition is also simplified discarding unnecessary behaviors not needed for the game design (behaviors of selection, transformation and pin). Synchronization has also been simplified since actions are not required to run in parallel.

Each actions diagram has associated two supplementary tables, where the corresponding actions (supplementary actions table) and the

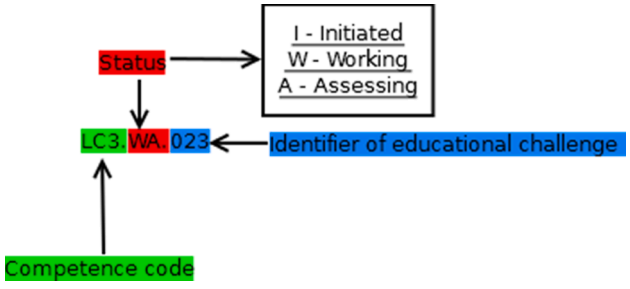


Fig. 10. Label for an educational challenge.

educational challenges (supplementary educational challenges table) in the scene are respectively described in natural language (pseudo-code, formulas, etc. can be also used). In the educational challenges table, the challenges included in the scene are described in terms of purpose, methodology and evaluation (no educational model is required). The actions table details each action in the actions diagram. If the action has associated one or more educational challenges, the way in which the action develops each educational challenge will also be specified. For example, if the educational challenge is to make a sum, and the action requires the player to stack blocks with this purpose, it will be specified that the relationship of the action with the challenge is produced by adding the blocks until a given number (the sum) is achieved. If the action has associated one or more flags, each flag is described indicating the value before and after executing the action. For example, if an action involves obtaining an object that is required for a global challenge, a flag will indicate that the player has already achieved that object so that this requisite can be verified in another action where that global challenge is continued.

3.5.2. Actions diagram in Uranus

The actions diagram of the scene *Repair chariot* is shown in Fig. 11 as an example. It represents the actions and the educational challenges associated with this scene. For example, when the avatar dialogue with an ancient (*Dialogue with the old man*), the competence LC1 (described in Section 3.3.2) associated with the challenge 002 is started (LC1.I.002). This challenge (LC1.002) is worked during several actions (for example, *Collect nails*) and it is not evaluated (as you can see with the state = *A*) until the action *Tie up the horses* (LC1.A.002) where the player completes the reparation, showing that he/she has understood the instructions provided by the elder (literal comprehension).

The labeling of educational challenges and competences permits to implement the benefits of the problem-based learning, which the literature indicates that can be challenging [5]. Students are actively immersed performing actions in the game when the educational challenges are presented, so learners engage in an active cognitive process where a challenge does not necessarily provide them all the needed information. A challenge can start with an action and be worked during several actions and even for several scenes. Flags are used to register the achievements of the player and will be used in the action or actions where each competence of the challenge is evaluated. In our example, the chariot is broken and, in order to repair it, the avatar must perform several actions including literal comprehension (LC) (as indicated above –LC1) but also inferential comprehension (IC) (for example, the avatar must deduce that the wood of the backrest of an old chair serves him/her and then use the appropriate tool to extract said backrest – IC2). It is

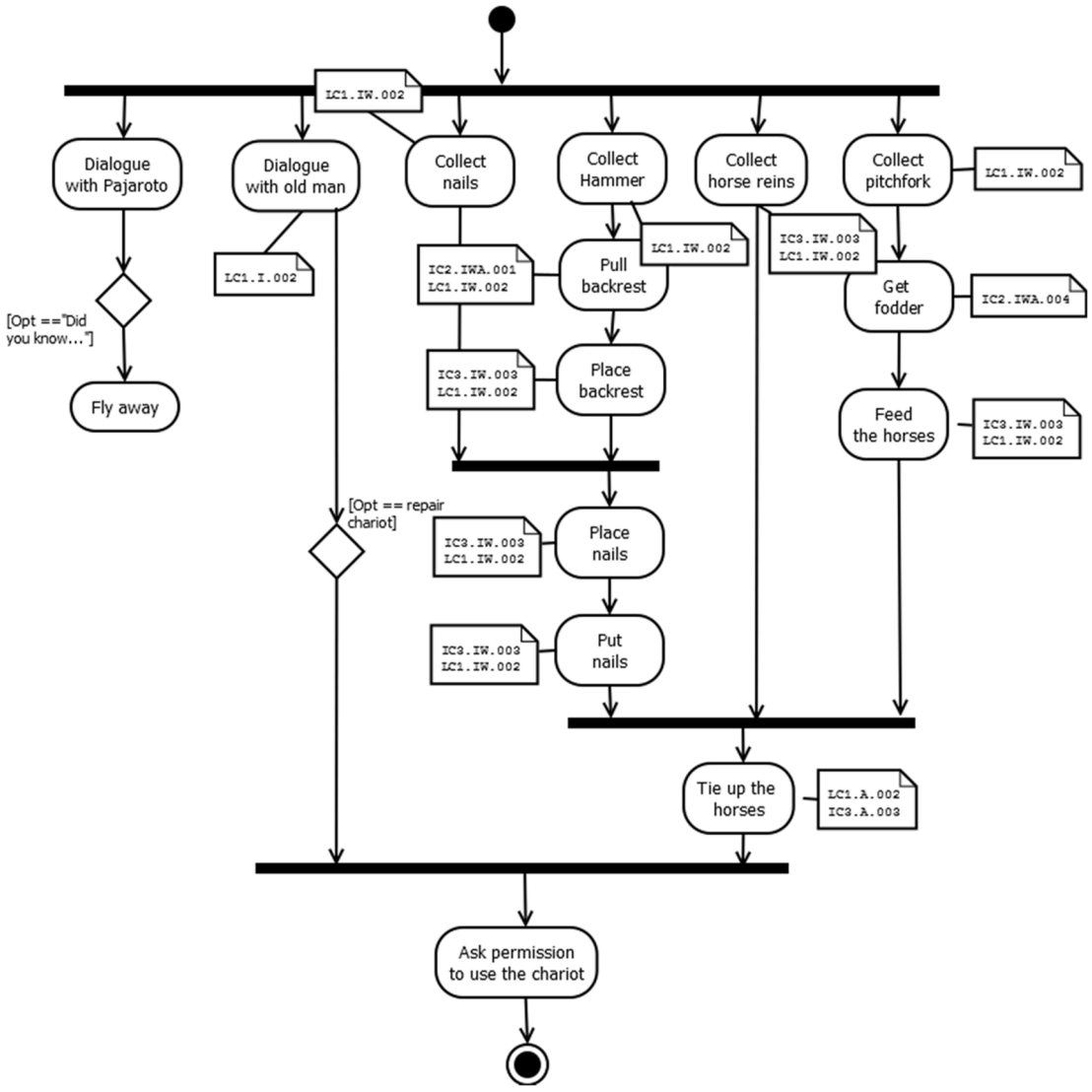


Fig. 11. Actions diagram in Uranus.

evident that the interrelation of challenges and actions is a joint work where designers and educators must work hand in hand. The supplementary tables, which includes the description of the challenges and actions, allows obtaining an overall vision of the educational challenges that in the diagram appear distributed along the different actions that involve them.

In addition, it can also be seen in the diagram that there are actions that require synchronization, for example, before *Place nails* it is necessary to conduct *Place backrest* and *Collect nails*. Moreover, there are two actions in the diagram that depend on a condition. Concretely, during the action *Dialogue with Pajaroto* (a bird) if the character selects the condition *"Did you know …"*, the bird flies away.

3.6. Dialogue diagrams

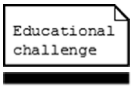
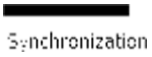
The dialogue diagrams allow representing every dialogue in the video game. This type of diagram is based on UML sequence diagrams which seem to be the most appropriate to represent dialogues since they describe message sequences.

3.6.1. Metamodel of the dialogue diagram

The metamodel of the dialogue diagrams is presented in Fig. 12. It describes the elements present in these diagrams and their relations.

Semantic. An *Actor* represents one of the characters participating in the dialogue (including the avatar). In the metamodel, the dialogue between actors will be represented by *Messages*, each of which models a spoken (or written) intervention of a character (that is, a conversation thread). Every actor has associated a sequence of interventions (*Intervention_line*), which includes all messages of an actor in a concrete dialogue; besides, a dialogue has at least two lines of intervention. On the other hand, the *Fragments* are bounded regions within a dialogue that carry at least one *Condition* (*loop* or *alt* in UML). Thus, a condition represents any branch or loop that could happen in certain conversation between characters. For example, a fragment with an alt condition is used to model when the avatar has several possible answer options to a question asked by another character and depending on the answer chosen by the avatar, the other character tells him/her one thing or another. A fragment with a loop condition is used when a sequence of messages is repeated until a valid response is obtained. The dialogues can be grouped if they involve the same actors (*Dialogue_group*). A very typical way of grouping dialogues is to represent the menu of the dialogues, where the player can to choose between several options of dialogue with a concrete character. Finally, *Educational challenges* may be associated with the messages. This will allow specifying the exact point in the conversation where the educational competences are addressed.

Table 4
Graphical notation of actions diagram.

Element in the actions diagram	Base UML element	Adaptation of UML	Graphic representation
Action	Activity diagrams → Activity	*Simplification:* An action is an Activity node (activity edge is eliminated) *Name change:* Activity node is renamed as Action *Comment:* Dialogue_Group is represented as an action	
Educational challenge	UML note	*Extension:* Using UML note to specify an educational challenge	
Synchronization	Activity diagrams → Control nodes → Fork and Join nodes	*Simplification:* Simultaneity is not required *Name change:* Fork and Join nodes has been rename as Synchronization	
Condition	Activity diagrams → Control nodes → Merge node	*Name change:* Merge nodes has been renamed as Condition	
Final / Initial	Activity diagrams → Control nodes → Final node/Initial node	None	
Transition	Activity diagrams → Control flow	*Simplification:* behaviors of selection, transformation and pin have been eliminated	

Graphical notation. Table 5 shows all the components of the dialogue diagrams. As in previous cases, UML notes are used with a different purpose for which they were created, namely to label educational challenges. Intervention line is conceptually similar to the UML life line. UML messages are also simplified, being used only one kind of message (synchronous message). UML messages as asynchronous messages or lost messages are options that would not make sense in a video game.

Finally, each dialogue diagram has associated a supplementary message table where the first column identifies each message of a dialogue, the second column shows the complete text of the message and the third column details the associated educational challenge if it exists (or the associated challenges if there are several of them). In order to improve the readability of the dialogue diagram, the messages are only shown completely in the diagram if they are short, otherwise only the beginning of the message is shown in the diagram and it must be searched in the supplementary message table to obtain the complete text.

3.6.2. Dialogue diagram in Uranus

An example of dialogue diagram is shown in Fig. 13 (*Dialogue with old man*). The diagram represents a dialogue group between the avatar and the old man. First, there is one dialogue where five messages are exchanged. Then, there are three dialog-groups grouped with a loop where if the option "*Thanks for everything*" is chosen, the avatar will finish the dialogue. Inside the loop, three fragments can be seen; each one with three conditions (alt), so depending on the option chosen by the player, one branch or another will be followed.

On the other hand, in the second fragment (alt="*How to repair…*"), the old man explains to the boy how to repair the chariot and an *educational challenge* is associated. In this challenge, the educational competence (LC1 - Literal Comprehension of data) is *initiated* and, as it was explained, it will be assessed when the avatar finish repairing the chariot.

It is necessary to emphasize that the narrative of the game is a very important educational resource, which is often used to integrate the challenges. The dialogue diagrams have the aim to provide educators and designers with a graphic language to specify the educational challenges and their work/evaluation in specific points of the dialogues. Complementary, the supplementary table makes it possible to specify any additional detail about a challenge initiated, worked or evaluated in a point of a dialogue (that is, in a message) using a no technical language.

4. Method

In this section, it is described a validation experience which was conducted with educators and computer engineering students in order to test the value of the proposed graphic notation to design adventure-based educational games.

4.1. Research questions

In this evaluation we address the research question 3: Are our diagrams understandable and usable for computer engineers and educators? and the research question 4: Does our approach present benefits (in terms of understanding and usability) against other UML-based notations to model educational video games? To answer the question 4, it has been decided to compare the proposal presented in this article with other two proposals that include UML-based graphic notations to design video games. In accordance with the analysis in the related works section, the notations proposed by Cooper and Longstreet [10], and Taylor et al. [53] were chosen since they also use UML graphical notations and, although the approach is not exactly the same as ours, both proposals permit to represent the flow inside the scenes (as in our proposal). Thus the three proposals will be presented to the participants during the experiment (as explained below).

4.2. Participants

The sample included computer science students and educators to check its validity for both profiles (technical and non-technical). Specifically, the population of the experiment comprised eight educators from three Spanish schools (*Sagrada Familia*, *Sagrado Corazón* and *IES José María Infantes*) at Utrera (Sevilla, Spain), and twelve students of Computer Engineering (second course) from the University of Granada, in Spain. The whole group included nine woman and eleven men, with an average age of twenty-seven years old.

4.3. Materials

Three different graphical representations of a part of the Uranus game were presented to the participants, one for each proposal. The materials used are available at the experiment's document[4]. Different scenes was chosen to avoid that the participants become familiar with a scene of the game. All the selected scenes belong to the Rome mini-story, one of the most complete and with more educational challenges in the game. Following our proposal, the acts diagram representing the complete structure of the game (nine acts) and the actions diagram of the scene "Repair chariot" (fifteen scenes and four educational challenges) were modelled. In addition, the supplementary table with the educational challenges in this scene was provided to the participants.

[4] http://bios.ugr.es/~rprieto/gsg/questionnaire.pdf

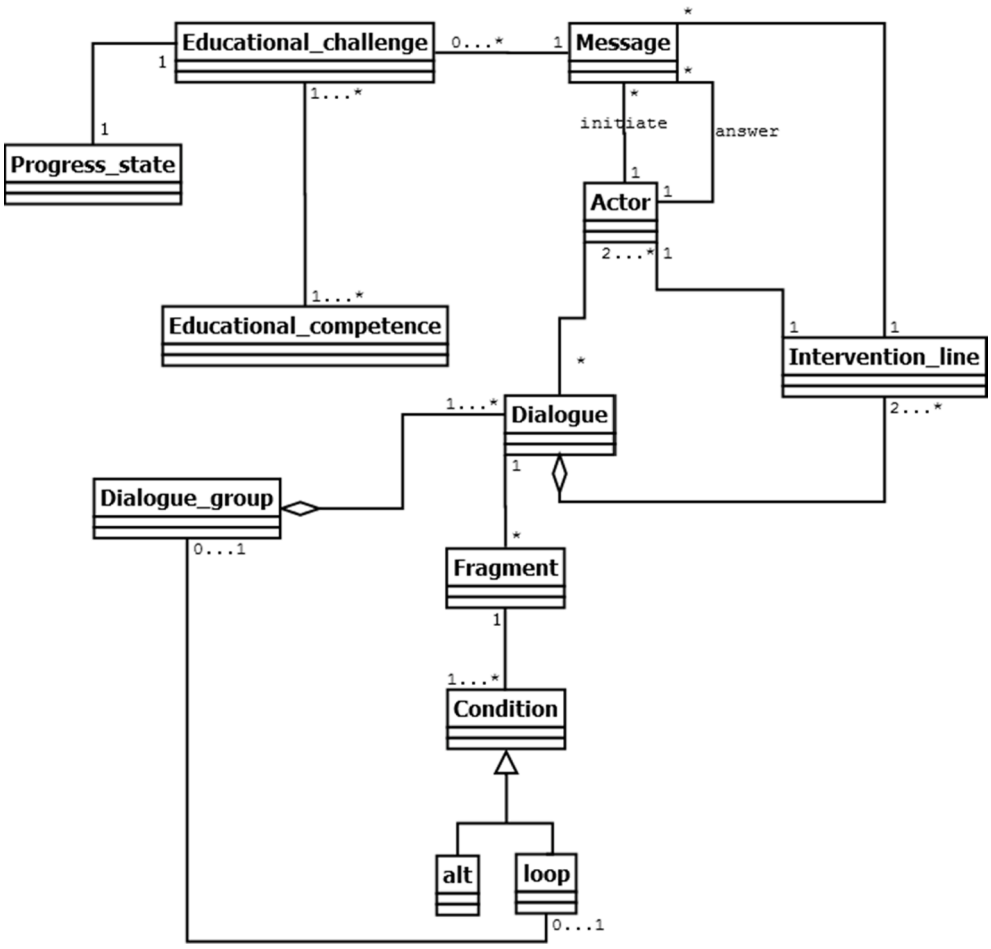


Fig. 12. Metamodel of a dialogue diagram.

Table 5
Graphical notation of dialogue diagram.

Element in the dialogue diagram	Base UML element	Adaptation of UML	Graphic representation
Actor	Sequence diagrams → Actor	None	
Educational challenge	UML note	*Extension:* Using UML note to specify an educational challenge	Actor Educational challenge
Intervention line	Sequence diagrams → Life line	*Name change:* Life line has been renamed as Intervention_line	Life_line
Message	Sequence diagrams → Message	*Simplification:* There is only one kind of message (synchronous)	[Description] Message
Fragment Condition loop	Sequence diagrams → Combined fragment (loop)	None	loop [condition]
alt	Sequence diagrams → Combined fragment (alt)	None	alt [condition]

Applying the proposal by Cooper and Longstreet, a use case diagram representing the structure of the game (thirteen nodes) and a table with the description of the educational challenges were provided. In the table, real screenshots of the game instead of a storyboard were used. Due to the way in which the educational challenges of Cooper and Longstreet's proposal are represented, it was necessary to select a scene with few educative challenges (only one, "Arrive to prison") to reduce the size of the representation. Similarly, the global diagram of the game with the representation of Cooper and Logstreet could not show the whole story of the game for becoming unapproachable for the

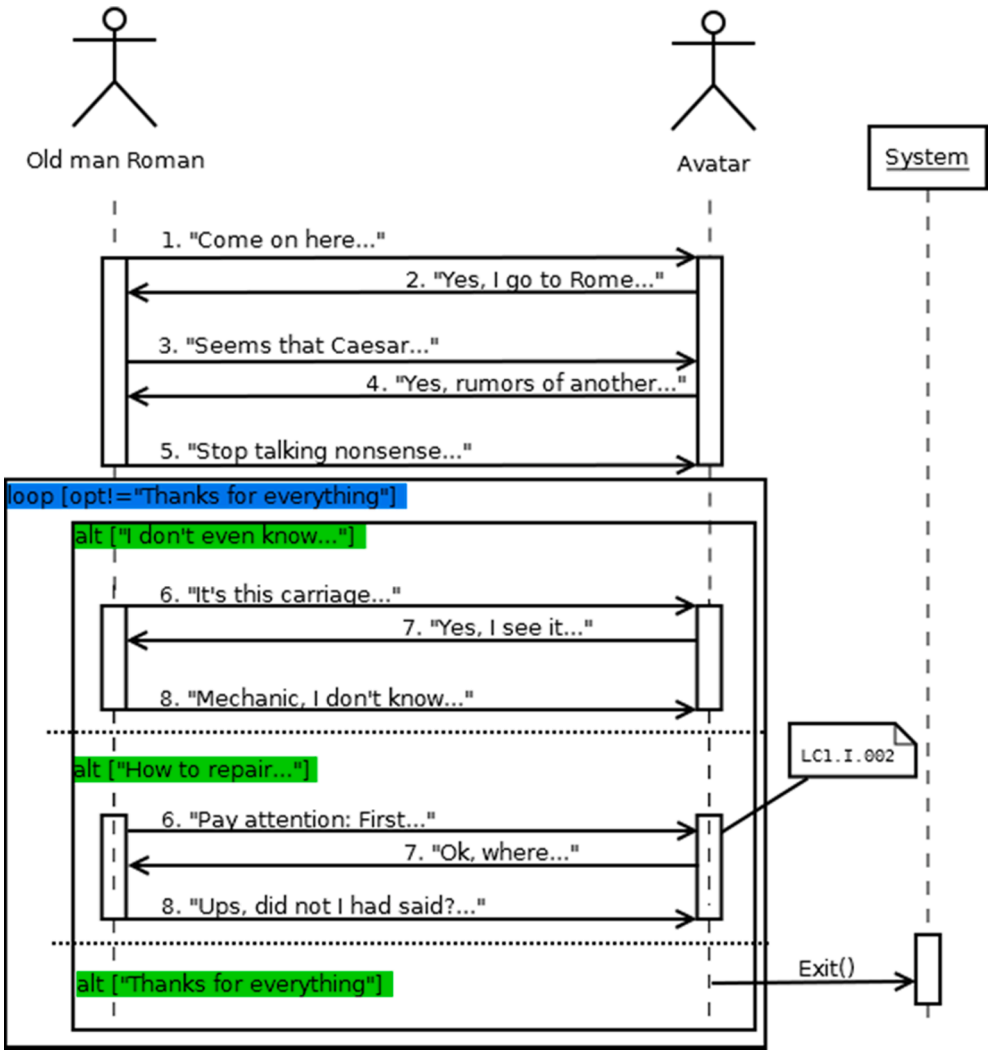


Fig. 13. Dialogue diagram in Uranus.

experiment, so only a part was represented. Finally, the proposal by Taylor et al. was adapted to include educational challenges in the pseudocode used to represent the interactions since originally it does not have explicitly said concept. Following this proposal (with the mentioned modification), a game-flow diagram (an adaptation of UML use cases) to structure the scene "Visit Julius Caesar in prison" (including five nodes and two educational challenges) and the corresponding pseudocode (ten lines) were provided. In this case, the proposal does not include a diagram with the overall structure of the game because in their work is not proposed.

4.4. Procedure

The validation experience was carried out in person and individually with each participant. For each proposal included in the validation, the participants received a brief tutorial (approximately ten minutes) and, then, they were given the set of design documents generated according to the proposal (materials have been described in the previous section). Between ten and fifteen minutes were given to analyze each proposal. After this process, participants completed a questionnaire about the diagrams and complementary resources represented with each proposal (approximately ten minutes). Finally, participants answered a usability questionnaire comparing the three proposals (approximately ten minutes more). Therefore, each session took between one and two hours. The order in which the proposals were presented to each participant was random with the aim of not introducing bias in the results.

4.5. Questionnaire

The questionnaire included questions to detect if the participants had understood the design of the game represented in each diagram and questions to assess how easy and pleasant had been that process. The complete survey with all the questions is available at the experiment's document[4]. The questions were organized in four parts (and are summarized below):

(1) Personal data: age, gender, previous knowledge in UML (yes/no), and frequent game player (yes/no).
(2) Comprehension test: Five comprehension questions associated with each of the three evaluated approaches in order to test user comprehension about the design documents created with each graphic notation. The aim of these questions was to assess whether the surveyed people had correctly understood the game elements described by each representation. The questions were referred to the structure of the game, the educational challenges and the evaluation of these challenges.
(3) Specific usability: Five opinion questions using a Likert-type scale [52] to compare the three proposals in terms of usability. In this way, the surveyed people were asked to compare the proposal in terms of: assimilation (Q1), comprehension (Q2), ease of use

(Q3), ability to represent educational challenges (Q4), and ease of structuring the game (Q5).

(4) General usability: Ten Likert-type questions following the System Usability Scale [SUS, 17]. So, these ask about: like to use (SUS1), unnecessarily complex (SUS2), easy to use (SUS3), need the support of technical staff (SUS4), well integrated (SUS5), too much inconsistency (SUS6), most people would learn to use (SUS7), very cumbersome to use (SUS8), felt very confident (SUS9), and needed to learn a lot of things (SUS10).

4.6. Data analysis

During the sessions, some informal annotations were made by the interviewer, obtaining a very positive impression about the usefulness of these UML-based diagrams to communicate the design of the video game. Logically, the educators asked more questions during tutorials, but none of the participants had problems to perform the experience. With the intention of interpreting the obtained numerical data, a quantitative analysis was carried out. Therefore, once all the polls had been completed, the data was collected, cleaned, filtered and homogenized to conduct the statistical analysis. The obtained result of this analysis are presented and commented in the following section.

4.7. Results

The results will be detailed and discussed according to the research questions (which have been decomposed to facilitate the analysis). The interpretation of these results present preliminary evidence about the utility of our approach, which shows a better usability and less error-proneness than the proposals with which it has been compared.

(1) Are our diagrams understandable for computer engineers and educators?

Five questions were asked to each participant to check if they understood the design of the part of the game represented with the proposal's diagrams (questions can be seen at the experiment's document[4]). As shown in Fig. 14 (an error implies an incorrect interpretation of the provided diagrams), only one user made three mistakes, revealing a significant lack of understanding. Fourteen of the twenty respondents had no errors. The remaining five participants had only one or two errors. The question with more errors was about the educational competence worked and assessment in a challenge represented in the actions diagram (six of the twenty participants failed this question). This may be because the way to label educational challenges is a bit complicated and it may require more training. Even so, from these results, we can say that our proposal was well understood by computer engineers (computer science students, u1 to u12 in the Fig. 14) and the educators (teachers of the three mentioned schools, u13 to u20 in the Fig. 14). Nevertheless, the number of participants who made mistakes was double in the case of educators (four participants failed some question) compared to computer science students (only two participants failed some question). This could indicate that in the case of the educational team, since they do not have previous knowledge of UML, a more extensive training would be necessary. Although even so, in regard to the proposal presented, the educators showed a high understanding (mean = 1error/educator); which is very positive for us considering that the training last ten minutes. In addition, it should be noticed that three educators (u15, u16 and u18) committed 70% of all the errors (7 errors /8 errors in total) while the other five educators committed very few errors (only 1 error made by u14).

(2) Does our approach present benefits (in terms of understanding) against other UML-based notations to model educational video games?

The first point to remark, on the basis of the statistical study, is that the second part (comprehension test) of the survey determines that the authors' proposal presents the least number of errors in the answers (Fig. 14). Thus, there have been just 10 errors in the 100 made questions (5 questions * 20 surveyed) regarding the interpretation of the authors' proposal. On the other hand, the results of Cooper and Longstreet's model and Taylor's one have been very similar among them, 24 and 25 errors respectively. These numbers can be seen in Fig. 14 which shows the number of errors for every user regarding each proposal. Moreover, as mentioned, the majority of surveys (14/20) answered without any error with respect to the authors' proposal while with the Cooper and Longstreet's proposal only five participants had no errors and with the Taylor's proposal only one participant had no errors. Analyzing the questions with more errors, in the case of Cooper and Longstreet, two questions accumulated the majority (89%): the structure of the game (16/27) and the evaluation of the educational challenges (8/27). Concerning Taylor et al., the question on the structure of the scene represented by the diagram accumulated the greatest number of errors (14/25); the interpretation errors in other aspects were more distributed.

If the groups of students and educators are separately analyzed, there is an increase in the fail rate of the educators (75%) with respect to computer engineering students (55%). This confirms that educators require extra training to interpret the UML diagrams with the three evaluated proposals.

(3) Are our diagrams usable for computer engineers and educators?

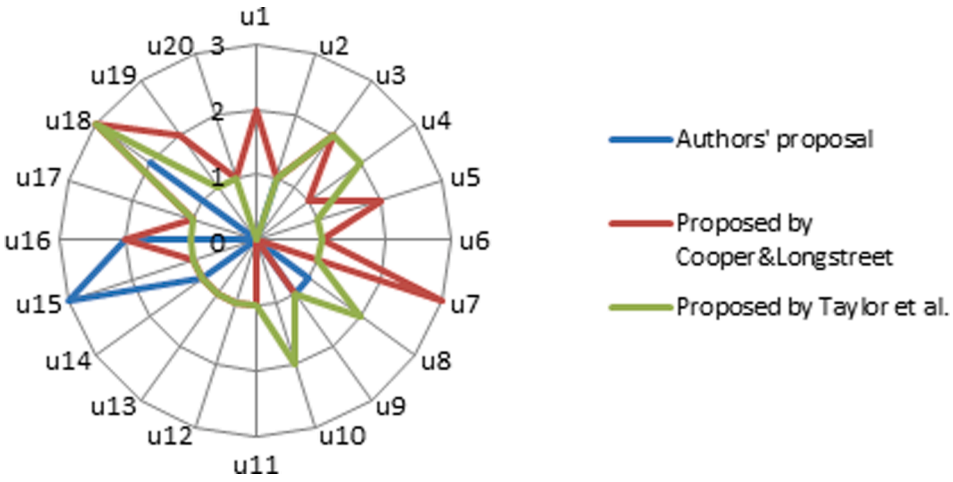


Fig. 14. Number of errors in the survey presented to 20 users regarding each proposal.

In the third and fourth part of the survey (specific usability and general usability) a total of 15 questions were asked using a Likert scale. Thus, in order to better visualize and analyze the results, some questions whose best response from the point of view of design was '1' in the Likert scale have been inverted, so that '5' is always the most positive answer in any question now. For all the answers the average has been obtained rounding to two decimals. Fig. 15 shows the results, from left to right, of the engineering students, the educators and the total sample. The results obtained for the authors' proposal are shown on the left. The five questions corresponding to specific usability have been represented as Q*i*, where *i* is the question number ($i \in [1, 5]$). And the ten questions of general usability that belong to [SUS, 17] have been represented as SUS*j*, where *j* is the question number ($j \in [1, 10]$). Questions can be seen in the experiment's document[4].

The authors' proposal has been rated as very usable, obtaining an average of 4.24/5. The usability average of the students has been 4.33/5 and the educators' usability average has been 4.11/5. The most poorly rated questions were SUS2, SUS3 y SUS8, referencing unnecessary complexity, easy to use and cumbersome to use (even so the obtained value was good, 4/5). In comparison, educators have perceived the proposition slightly less usable than computer science students. This is probably due to the need for further training in some cases; although, both for educators and students, the proposal is easy to learn and pleasant to use.

(4) Does the approach present benefits (in terms of usability) against other UML-based notations to model educational video games?

As it can be seen in the Fig. 15, the students have rated as more usable the authors' proposal (4.33/5). The other two proposals have obtained similar qualification among them (3.03/5 and 2.80/5), although slightly better Cooper and Longstreet's results. With respect to the educators, the results are more similar, especially the proposal of Cooper and Longstreet and the authors 'proposal, although the authors' proposal have obtained a better average (4.11 vs 3.70). In the whole sample of users, as expected from the two previous graphs, the authors' proposal has been chosen as the most usable. The total averages have been: authors' proposal → 4.24, proposed by Cooper and Longstreet → 3.30, and approach of Taylor et al. → 2.85.

Focusing on the five questions about specific usability, there are three (Q3: 4.15, Q4: 4.45 and Q5: 4.60) where authors' proposal show a larger difference with respect to Cooper and Longstreet (Q3:3.10, Q4: 3.00 and Q5: 3.25) and Taylor et al. (Q3: 2.80, Q4: 2.30 and Q5: 2.50). These issues referred respectively to ease of use, ability to represent educational challenges, and ease of structuring the game. These are very positive data in view of the validity of the authors' proposal. These numbers are in agreement with the errors committed by the surveyed. The difference is lower for questions Q1 and Q2 (assimilation and understanding), showing that the three proposals are assimilated and understood correctly in a similar way. Positive data also since the three proposals have been correctly assimilated by technical personnel and educators.

Regarding general usability (SUS*j*), there is a greater difference between the authors' proposal and the other two in SUS1, SUS2, SUS4, SUS5, SUS6, SUS9 and SUS10 issues, the results are very favorable for the proposal of the authors. Then, the results of the three proposals are shown in parentheses, first the authors' proposal, next Cooper and Longstreet, and finally Taylor et al. Questions that refer to preference and understanding (SUS1: 4.45 vs 3.20 and 2.65 and SUS2: 3.85 vs 3.20 and 2.90), the need for technical support (SUS 4: 4.50 vs 3.75 and 3.00), integrity and inconsistencies (SUS5: 4.30 vs 3.45 and 2.80, and SUS6: 4.50 vs 3.20 and 3.10), confident and completeness (SUS9: 4.00 vs 3.00 and 2.65, and SUS10: 4.05 vs 3.10 and 2.85). However, the comparison is more adjusted in SUS3 (ease of use), SUS7 (assimilation) and SUS8 (cumbersome), especially in the case of educators and with respect to the proposal of Cooper and Longstreet, which may be understandable by their non-technical profile.

5. Conclusions and future work

This paper proposes a graphical notation based on UML to design educational games, which is especially suited for video games where the story defines the structure of the game. This happens, for instance, with the genre of adventure, where the narrative is essential. According to [5] (based on [45] and [46], learning cannot occur unless an individual is in a mental state of disequilibrium -because the true learning is born of the constructive process that takes place during the resolution of the conflict-, so the graphic adventure -a constant source of conflicts that the player must overcome- is revealed as an very powerful learning resource. Thus, a design language to define educational adventure and others types of video games based on narrative can be an interesting tool that should allow educators, designers and developers to work together. Accordingly, the proposed notation is very simple and visual to make it easier for educators to understand designers and vice versa.

In addition, this work presents a set of metamodels to describe and structure educational games, formalizing the proposed graphic notation. The metamodels are represented by means of UML class diagrams and establish the main elements of the game and their interrelations. This conceptual effort aims to help the designers and educators to structure the game and also to facilitate understanding for non-technical users since metamodels define a common game design vocabulary. An added advantage of the proposal is that the designers can, if they want it, instantiate these metamodels and create their own graphical notations.

In this respect, the graphical notation proposed in the article is an

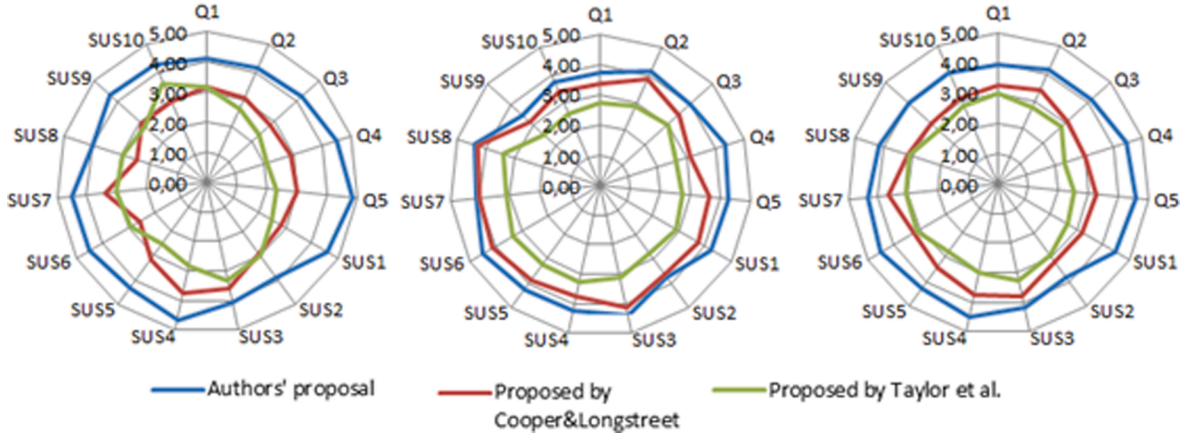


Fig. 15. Specific usability (Q1-Q5) and general usability (SUS1-SUS10). Answers from Students (left), Educators (center) and All the participants (right) are shown.

adaptation of the UML standard, which establishes a set of design diagrams based on it. The diagrams created following the proposed notation aim to ensure an easy understanding by both the technical staff and the team of educators, following the ideas highlighted in some previous studies, such as Cooper and Longstreet [10]. This notation can also be crucial to enable the communication between the different actors of the stakeholders. The set of proposed diagrams structure the game in acts, scenes, scenarios, actions and dialogues and are included in a development methodology for educational games [12], although can also be applied independently of this methodology.

In order to show the feasibility and value of the proposal, a real educational game, titled *Uranus*, has been modeled following the proposed methodology and integrating educators in the process of design of the game. Thus, the design diagrams of *Uranus* are described in the paper as a proof of concept. In addition, the video game *Uranus* has been successfully finished; with very few required clarifications from the video game development team (Greyman Studios Company). Additionally, an experience has been conducted with a group of 20 participants including technical (computer sciences students) and pedagogical people, who successfully interpreted a set of design diagrams of *Uranus* generated according to the proposal (only 10 errors in 100 comprehension questions). In addition, the proposal has been compared with two other proposals for game design [10,53]. The obtained results, if they are not statistically significant, have been encouraging, since the authors' proposal has been selected by students and educators as the most adequate. Thus the authors' proposal has obtained a smaller number of errors in the understanding of the different diagrams and supplementary tables. With regard to the other two proposals, the one by Cooper and Longstreet has obtained slightly better results, although the three proposals have proved sufficiently suitable in certain aspects of usability such as ease of use, assimilation or cumbersome. In the usability comparisons, the authors' proposal has again shown better results. The main conclusion from the results could be that the educators have correctly understood and assimilated the proposal of this paper, with a good difference in results with respect to the other proposals.

Therefore, the main implication of the work falls on the specification and documentation phase of the educational video game design. Although the proposal also implies a design vocabulary for technical and non-technical personnel at a conceptual level and it implies the incremental structuring of the game in acts, scenes, actions, etc. at a methodological level. Regarding to the limitations of the work, the proposal is especially suitable for graphic adventures, and may not fit some peculiarities of other genres of video games (games of different genres have been modelled with the proposal, but this validation has not been extensive [16]. On the other hand, there are a lot of UML diagrams to create, which can generate a high workload in complex video games. For example, for video games with extensive and rich dialogs, creating all dialog diagrams could be tedious. However, in return, the advantages of using UML would be obtained. That is, the set of UML-based diagrams proposed in this work is a complete documentation of the video game's structure that facilitates the development of the game without ambiguity (it is important because the development teams are usually different from the design teams) and assists the subsequent evolution of the game. However, as it is an incremental approach, designers may decide to make some diagrams and others not in each case. Not all diagrams are required. Finally, the fact that the proposed notation makes an adaptation on the standard UML can generate some confusion for expert users of this language. For example, a fork node in a UML diagram is a control node that splits a flow into multiple concurrent flows but in the proposed diagrams is used to indicate that several acts (scenes or actions) can happen in any order, but they must all happen before moving forward (which is quite common in video games). These adaptations are necessary to fit the special characteristics of video games and we hope that the derived limitation could be reduced with adequate user manuals and training.

There are some future work lines open. In the first place, we should analyse possible simplifications or modifications in the proposed graphical notation from those questions where there have been made more misinterpretations during the comprehension tests (reviewing the labelling of educational competencies seems to be necessary) and then repeat the evaluation experience with a larger number of participants. During this second evaluation would also be interesting to increase the type and variation of questions in the validation survey in order to obtain firmer conclusions. Then, the proposal could be used in the design of future video games, by other work teams unfamiliar with the proposal. This task was previously carried out with a couple of games [16], but a more massive use of the proposal will reinforce our certainty about its feasibility. Secondly, since the proposal focuses on the design of the complete structure of the game in acts, scenes, etc., it should be extended to other facets of the game. Thus, it will be necessary to define the graphical notation concerning the design of the sound, gameplay and adaptation (to the player); as well as the possibility of including collaboration (between players to achieve a goal) and emotions (emotional reactions during the game for analysis). Finally, existing tools for automatic source code generation from UML diagrams will be studied to analyse if it is possible to create a tool to generate some code of the video game from the proposed UML-based diagrams. This would have a positive impact on development times and encourage video games companies, usually reluctant to spend time on UML designs, to use the proposal (at least for the most critical or complex parts of their educational games).

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is supported by the Andalusian Research Program under the project P11-TIC-7486 co-funded by FEDER, together with TIN2014-56494-C4-3-P and TEC2015-68752 from the Spanish Ministry of Economy and Competitiveness and FEDER also.

References

[1] J. Alvarez, O. Rampnoux, J.P. Jessel, G. Methel, Serious Game: Just a question of posture, Retrieved from, Artificial Ambient Intelli. 420–423 (2007).
[2] H. Amaral, J.L. Braga, A. Galvão, Game Architecture for teaching-learning process: An application on an undergraduate course, in: Proceedings of IEEE International Games Innovation Conference (IGIC), 2013, 1–6.
[3] C.S. Ang, R.K. Rao, Designing Interactivity in Computer games a UML approach, Int. J. Intell. Games Simul. 3 (2) (2005) 62–69.
[4] L.A. Annetta, W.M. Frazier, E. Folta, S. Holmes, R. Lamb, M.T. Cheng, Science teacher efficacy and extrinsic factors toward professional development using video games in a design-based research model: The next generation of STEM learning, J. Sci. Educ. Technol. 22 (1) (2013) 47–61.
[5] L.A. Annetta, The "I's" have it: A framework for serious educational game design, Rev. General Psychol. 14 (2) (2010) 105–113.
[6] N. Aouadi, P. Pernelle, J.C. Marty, T. Carron, A Model Driven Architecture MDA Approach to Facilitate the Serious Game Integration in an e-Learning Environment, in: Proceedings of European Conference on Games Based Learning, Academic Conferences International Limited, 2015, 15–24.
[7] E. Bethke, Game Development and Production, Wordware Publishing Inc., 2003.
[8] H.M. Chandler, The Game Production Handbook, Jones & Bartlett Publishers, 2009.
[9] A. Collins, Design Issues for Learning Environments. In International perspectives on the psychological foundations of technology-based learning environments, 1996.
[10] K.M. Cooper, C.S. Longstreet, Towards model-driven game engineering for serious educational games: Tailored use cases for game requirements, in: Proceedings of 17th International Conference on Computer Games (CGAMES), IEEE, 2012, 208–212.
[11] K.M. Cooper, E.S. Nasr, C.S. Longstreet, Towards model-driven requirements engineering for serious educational games: Informal, semi-formal, and formal models, in: Proceedings of International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer International Publishing, 2014, 17–22.

[12] R.P. De Lope, J.R.L. Arcos, N. Medina-Medina, P. Paderewski, F.L. Gutiérrez-Vela, Design methodology for educational games based on graphical notations: Designing Urano, Entertainment Comput. 18 2016 ((available online 20 August 2016)), 1–14. https://doi.org/10.1016/j.entcom.2016.08.005.
[13] R.P. De Lope, N. Medina-Medina, R. Montes Soldado, A. Mora García, F. Gutiérrez-Vela, Designing educational games: Key elements and methodological approach, in: Proceedings of 9th International Conference on Virtual Worlds and Games for Serious Applications, VS-Games 2017, 2017, 63–70. https://doi.org/10.1109/VS-GAMES.2017.8055812.
[14] R.P. De Lope, N. Medina-Medina, Using UML to model educational games, in: Proceedings of 8th International Conference on Games and Virtual Worlds for Serious Applications, VS-Games 2016, 2016, 1–4. https://doi.org/10.1109/VS-GAMES.2016.7590373.
[15] R.P. De Lope, N. Medina-Medina, A comprehensive taxonomy for serious games, J. Educ. Comput. Res. 55 (5) (2017) 629–672, https://doi.org/10.1177/0735633116681301.
[16] R. De Lope, Doctoral Thesis "Development Methodology for Educational Videogames Based on Graphic Notations". University of Granada, June 2018.
[17] T. Demachi, Extreme game development: right on time, every time, 2003. Retrieved January 29, 2021, from https://www.gamasutra.com/view/feature/131236/extreme_game_development_right_on_php.
[18] B. Dobing, J. Parsons, How UML is used, Commun. ACM 49 (5) (2006) 109–113.
[19] C. Dowd, Creative Agents and Triggers (CAT) Game Design Method: For Crisis Communication, in: Proceedings of Conference on Games and Virtual Worlds for Serious Applications, 2009, IEEE, 2009, 215–216.
[20] K. Durkin, J. Boyle, S. Hunter, G. Conti-Ramsden, Video games for children and adolescents with special educational needs, Zeitschrift Für Psychologie 221 (2) (2015) 79–89.
[21] ESA, Entertainment software association, 2020. Retrieved January 29, 2021, from https://www.theesa.com/press-releases/report-video-games-contribute-90-billion-to-u-s-economy/.
[22] J. Foreman, Next-generation educational technology versus the lecture, Educause 38 (4) (2003) 12–22.
[23] M. Fowler, UML distilled: a brief guide to the standard object modeling language, Addison-Wesley Professional, 2004.
[24] D.L. García, Metodología ontológica para el desarrollo de videojuegos (doctoral dissertation), Universidad Complutense de Madrid, 2014.
[25] R. Garris, R. Ahlers, J.E. Driskell, Games, motivation, and learning: A research and practice model, Simulation Gaming 33 (4) (2002) 441–467.
[26] I. Granic, A. Lobel, R.C. Engels, The benefits of playing video games, Am. Psychol. 69 (1) (2014) 66.
[27] M. Griffiths, The educational benefits of videogames, Educ. Health 20 (3) (2002) 47–51.
[28] J. Groff, C. Howells, S. Cranmer, Console game-based pedagogy: A study of primary and secondary classroom learning through console video games, Int. J. Game-Based Learn. (IJGBL) 2 (2) (2012) 35–54.
[29] S. Hai-Jew, Alpha testing, beta testing, and customized testing, in: Designing Instruction For Open Sharing, 2019, pp. 381–428.
[30] W. Hu, A common software architecture for educational games, in: Proceedings of International Conference on Technologies for E-Learning and Digital Entertainment, Springer Berlin Heidelberg, 2010, 405–416.
[31] IMS Global Learning Consortium, Learning Design Specification, 2017. Retrieved July 15, 2017, from http://www.imsglobal.org/learningdesign/index.html.
[32] V. Janarthanan, Serious video games: Games for education and health, in: Ninth International Conference on Information Technology: New Generations (ITNG), 2012, 875–878. Retrieved from https://ieeexplore.ieee.org/abstract/document/6209109/.
[33] Y.B. Kafai, C.C. Ching, Talking science within design: Learning through design as a context for situating children's scientific discourse, J. Learn. Sci. 10 (2001) 323–363.
[34] Y.B. Kafai, Playing and making games for learning: Instructionist and constructionist perspectives for game studies, Games Cult. 1 (1) (2006) 36–40.
[35] G. Kalmpourtzis, L. Vrysis, A. Veglis, Teaching game design to students of the early childhood through Forest Maths, in: Proceedings of 11th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP), 2016, 123–127.
[36] S.C. Kong, H. Abelson, Computational thinking education, Springer Nature, 2019, p. 382.
[37] F. Laamarti, M. Eid, A. Saddik, An overview of serious games, Int. J. Comput. Games Technol. 11 (2014). https://dl.acm.org/citation.cfm?id=2728206.
[38] R. Lamb, L. Annetta, D. Vallet, The interface of creativity, fluency, lateral thinking, and technology while designing Serious Educational Games in a science classroom, Electronic J. Res. Educ. Psychol. 13 (2) (2015) 219–241.
[39] T. Marsh, Serious games continuum: Between games for purpose and experiential environments for purpose, Entertainment Comput. 2 (2) (2011) 61–68.
[40] N. Medina-Medina, P. Paderewski, N. Padilla-Zea, R. López-Arcos, F. Gutiérrez-Vela, Model for the integration of educational processes in a graphic adventure, Campus Virtuales 7 (1) (2018).
[41] D. Michael, S. Chen, Serious games: Games that educate, train, and inform (Muska & Li), 2005. Retrieved from https://dl.acm.org/citation.cfm?id=1051239.
[42] P. Moreno-Ger, D. Burgos, I. Martínez-Ortiz, J.L. Sierra, B. Fernández-Manjón, Educational game design for online education, Comput. Human Behav. 24 (6) (2008) 2530–2540.
[43] OMG, MDA® - The Architecture Of Choice For A Changing World, 2019. Retrieved May 30, 2019, from https://www.omg.org/mda/.
[44] S. Papert, An exploration in the space of mathematics educations, Int. J. Comput. Math. Learn. 1 (1) (1996) 95–123.
[45] Phillips, Piaget's theory, a primer. WH Freeman, 1981.
[46] J. Piaget, M. Cook, The origins of intelligence in children, New York: Int. Universities Press 8 (5) (1952) 18.
[47] R. Pooley, P. Wilcox, Applying UML. Elsiever, 2003.
[48] K. Rapeepisarn, K.W. Wong, C.C. Fung, M.S. Khine, The relationship between game genres, learning techniques and learning styles in educational computer games, in: Proceedings of International Conference on Technologies for E-Learning and Digital Entertainment, Springer Berlin Heidelberg, 2008, 497–508.
[49] R. Rosas, M. Nussbaum, P. Cumsille, V. Marianov, M. Correa, P. Flores, P. Rodriguez, Beyond Nintendo: design and assessment of educational video games for first and second grade students, Comput. Educ. 40 (1) (2003) 71–94.
[50] G. Schneider, J.P. Winters, Applying Use Cases: A Practical Guide, Pearson Education, 2001.
[51] Spain, G. of., Organic Law 8/2013, of December 9, for the improvement of educational quality, 2013.
[52] G.M. Sullivan, A.R. Artino Jr, Analyzing and interpreting data from Likert-type scales, J. Graduate Med. Educ. 5 (4) (2013) 541–542.
[53] M.J. Taylor, D. Gresty, M. Baskett, Computer game-flow design, Comput. Entertainment (CIE) 4 (1) (2006) 5-es.
[54] A. Tlili, F. Essalmi, L.J.B. Ayed, M. Jemni, Towards a generic UML model to support designing educational role playing games, in: 2016 IEEE 16th International Conference on Advanced Learning Technologies (ICALT), IEEE, 2016, pp. 153-157.
[55] UML, OMG Unified Modeling Language (UML). Version 2.5. Retrieved July 17, 2017, from http://www.omg.org/spec/UML/.
[56] R. Van Eck, Building artificially intelligent learning games, in: Games and simulations in online learning: Research and development frameworks, 2007, pp. 271–307.
[57] J.A. Vargas, L. García-Mundo, M. Genero, M. Piattini, A systematic mapping study on serious game quality, in: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, ACM, 2014, 15.
[58] F. Zurita Ortega, N. Medina Medina, F.L. Gutierrez Vela, R. Chacon Cuberos, Validation and psychometric properties of the gameplay-scale for educative video games in Spanish children, Sustainability 12 (6) (2020) 2283.
[59] M. Zyda, From visual simulation to virtual reality to games, Computer 38 (9) (2005) 25–32. http://wiki.arl.wustl.edu/images/4/47/Zyda-2005-computer.pdf.