Scenario-based Serious Games Repurposing

Aristidis Protopsaltis
Serious Games Institute
Coventry University Tech Park
Cheetah Road, Coventry, CV1 2TL, UK
aprotopsaltis@gmail.com

Laurent Auneau
Succubus Interactive
17 rue Paul Bellamy
44000 Nantes, France
laurent.auneau@succubus.fr

Ian Dunwell, Sara de Freitas, Panagiotis Petridis,
Sylvester Arnab, Simon Scarle, Maurice Hendrix
Serious Games Institute
Coventry University Tech Park
Cheetah Road, Coventry, CV1 2TL, UK
{IDunwell, SFreitas, PPetridis, SArnab, SScarle, MHendrix}@cad.coventry.ac.uk

ABSTRACT

Serious games are very content-rich forms of educational media, often combining high fidelity visual and audio content with diverse pedagogic approaches. This paper introduces scenario-based serious games repurposing and demonstrates repurposing a serious game into new learning objects. The process uses the scenario editor called “mEditor”. Two case studies based on the Happy Night Club serious game are presented. The article describes exploratory work which continues the work that started within the mEducator project regarding repurposing serious games in order to enable their use and reuse in the same or different educational contexts.

Categories and Subject Descriptors

J.m Computer Applications: Miscellanesous

General Terms

Experimentation

Keywords

Information systems education, Computer-assisted instruction (CAI), Computer science education, Collaborative learning, Serious Games; Interactive learning environments; Virtual reality; Simulations; Media in Education

1. INTRODUCTION

Serious games usage within the current educational context has been growing steadily over the last decade. Serious games are very suitable for learning and they are a very good environment for improving the learning experience, by increasing motivation and diversify the content delivery media. Considering however the time, effort, resources, cost and complexity of developing serious games it is imperative that such a game can be repurposed effectively adaptively into educational practices and curricula. This includes updating, changing, enriching serious games to reflect new functionalities, amending to different pedagogies, technologies, representations, cultures, contexts and learners. Such repurposing therefore, is a desirable activity, reducing organisational resource consumption and opening up new opportunities for learning, maximising the capabilities of existing learning objects. Therefore, being able to repurpose game content reduces the need to continually rebuild content, and offers potential to efficiently adapt serious games and game elements to meet the needs of wider audiences and application areas.

The term repurposing refers to the changing of a learning resource or object initially created for a specific educational context, to a new educational context (or contexts) [1], and should be distinguished from reuse, which refers to the use of the same learning resources or objects without any changes [2].

Work in the field of repurposing learning objects has been focused greatly on automatic learning object repurposing. For example, Zaka et al. [3] work on creating new PowerPoint presentations from a repository of existing presentations. Furthermore, the ALOCOM project created an ontology and a framework that was used to disassembles PowerPoint presentations and then reassembles them in a more meaningful way [4-6]. Further, the TRIAL-SOLUTION project defines an ontology for learning objects which aims to produce and deliver adaptive teaching materials from existing documents on mathematics at undergraduate level [7]. While Singh’s [8] work had focused on repurposing text.

On a slightly different direction other researchers focused on web-based technologies. For example, the DART project focused on the subject of anthropology has investigated how web-based technologies and digital resources can be used and reused to enhance student learning [9]. One important conclusion from this project was that repurposing aids the successful reuse of learning objects, and that involving the teacher in the design process is an essential part of repurposing since he is the one to use the new object to fulfil his learning objectives.

Multimedia repurposing has also been an area which received considerable attention by researchers. Steiger et al. [10] have worked on an MPEG-based personalized multimedia content delivery system while Hjelsvold, et al. [11] have showed a web-based interactive video repurposing framework. Hossain et al. [12, 13] have introduced a multimedia content repurposing framework using Web Services.

Contrary to the area of multimedia repurposing the area of games repurposing is still in its infancy and there are no exhaustive works addressing the issue. In one study, Burgos et al. [14] describe the use and the repurposing of what they called "generic" games, referring to commercial games. The focus of their work was mainly on the different pedagogic approaches in game repurposing. They described two different approaches where in one a game is fully integrated in the learning flow while in the other a game is used as an autonomous learning object disconnected from the learning flow.

In previous work [15] some of the authors started identifying the key issues faced when repurposing serious games. They proposed a theoretical framework for the repurposing of serious games in medical education and three case studies based on the Climate Health Impact serious game were developed. These case studies demonstrated the ability to repurpose a serious game into new learning objects, covering three different paradigms of content repurposing, language, content and pedagogy. The different paradigms for content repurposing have been developed by the mEducator consortium [16, 17]. This work has shown that the need for programming skills by the educators in order to be able to repurpose serious games is a limiting factor for widespread repurposing of game content. As a result, the authors formulated recommendations that readily assist serious games repurposing. Having the content separated is really important in order to facilitate the efficient repurposing of a game. A general recommendation would consider eliminating the need for programming skills by separating the content from the game engine changing the balance between what an educator cannot manipulate (i.e. due to a lack of programming skills or access to the source files) and what they can (i.e. eXtended Markup Language (XML) files) towards the latter. Content can be stored in separate files and educators would only have to access and modify those files and nothing else. Such tasks do not require advanced programming skills and therefore educators would be able to complete them with relative ease. Another suggestion that could simplify serious games repurposing and even games repurposing that came from this work [15] refers to scenario-based games repurposing. Scenario-based games are branching scenarios, where users' decisions lead to effects, both immediate and simulation wide, altering the events, characters and situations encountered. Players are usually faced with multiple scenarios and they can change the situations and challenges they are involved in, in the game. The scenario can be modified to fit individual needs and situations and feedback is provided seamlessly as a natural progression throughout the experience. Usually, the scenario in serious games is scripted inside the game dynamics and therefore quite often inaccessible for repurposing after being compiled. However, sometimes the script files are accessible after the game is being compiled, as they are actually only interpreted during runtime and therefore one will still need programming skills to understand them. A solution therefore could be to present the scenario loaded inside a bespoke editor. Therefore, the scenario can be edited and manipulated without programming knowledge and then directly integrated into the game. However, this bespoke editor would be entirely dedicated to the game engine in use, and thus scenario data could not be reused in another technology.

Examples of authoring applications that use environment editing are UnrealEd, the level editor of the Unreal 3 engine; and e-Adventure [18, 19]. UnrealEd was developed by and for professional game developers, and is as such very powerful, but also very complex. Another approach was adopted by van Est et al. [20], which allows instructors to define and edit scenarios, using high-level actions and events and some basic logic. However, these authoring applications are focusing more on providing alternative scenarios rather than repurposing serious games and learning objects.

Bar the work from Burgos and colleagues [14] and our own within mEducator (www.meducator.net) [15], our research uncovered no other current studies regarding repurposing of serious games content. Our current mEducator study therefore aims to provide the mechanisms with which to discuss and analyse the repurposing of game-based content using case studies as a research method. This work is an extension of our previous work [15] on serious games repurposing and focuses on scenario-based repurposing.

2. GAMES DESIGN AND SERIOUS GAME DESIGN

Regarding game design in general we can identify two main aspects, one technical which contains the game engine while the other one contains game rules, scenario, behaviours, background etc. and is the game mechanics.

In this way modern entertainment game development very much follows a data-driven software engineering paradigm. In that the underlying software, the Game Engine, is intended with minimal alteration to be useable on multiple game projects. The game's playable content, scenario and mechanics are codified as so-called runtime objects and these are then the data which is acted upon by the engine. This is an important part of the modern entertainment game industry business model, in that it allows for maximum re-use of software and more rapid development of game content. An ancillary component of the software for the Game Engine is its "Asset Pipeline", this will be a suite of software applications which take various forms of input from the artist and designers (e.g. 3-D objects, 2-D images, sound, description of AI and possible interactions) and converts them into templates for the required runtime objects. The tools which constitute this pipeline are increasingly becoming more generically useable, but are still primarily aimed at being used by professional games designers and artists.

Serious games design approaches derive from these general game design approaches. Work undertaken by one of the authors outlines how serious games design principles can be mapped against traditional game design processes and principles. From their review of game-based learning, de Freitas and Jarvis [21] derive five game-based learning development principles: foster positive attitudes, appropriate selection of games, learning and usability design criteria, adopt a participatory design approach, use formative evaluation methodologies. This is mapped against games principles and supported by specific tools and techniques based upon the four dimensional framework [22]. Pre-prototyping, human factors analysis, scenario creation tools and learning needs analysis approaches are adopted. Tate et al., [23] also outline design principles including defining fun as a main criterion for serious game design.

3. REPURPOSING AND PEDAGOGY

Pedagogy is a very important aspect of serious games repurposing. When repurposing to different pedagogical approaches one needs to consider that learning objects usually are created with specific teaching method in mind. However, repurposing can change the teaching approach, or the learning objectives or the assessment method and so on. Pedagogical repurposing might or might not require adaptation of the learning object itself.

The approach of the instruction methodology for example can be changed from a simulation based to a combination of a problem-based and exploratory learning. Learners are active during the learning process by constructing their own knowledge through exploration and action and serious games seem to aid such exploration.

From a pedagogy perspective well designed games have different learning theories integrated in the design and take advantage of their characteristics. These pedagogic approaches include Problem-Based Learning (PBL), as well as contextual and experiential learning models [24]. However, recent pedagogic trends have seen situative and experiential approaches move to the forefront [25]. In such approaches an emphasis is placed on indirect learning which occurs through exposure to situations that mirror real-world problems and environments.

Therefore the challenge of serious games repurposing does not only relates to the technical aspects of a serious games but also involves the pedagogic and instructional methods used to deliver the material. However, it is outside the scope of this paper to cover this aspect since the work presented here focuses on the technical challenges of serious games repurposing.

4. THE REPURPOSING CONTEXTS

Educational content repurposing is a very popular activity between educators and it can broadly be distinguished into ten different categories as proposed by the mEducator consortium [1, 15, 17]. The categories include: 1) Actual content, 2) Languages, 3) Cultures, 4) Pedagogical approaches, 5) Educational levels, 6) Disciplines or professions, 7) Delivery content types, 8) Technology, 9) Educational context, 10) Different abilities.

As shown in Figure 1 using these 10 definitions of repurposing types, we begin to deconstruct the various elements common to serious games and illustrate the strongest links between repurposing types and game elements. This provides a preliminary guide outlining which elements of a game are likely to be needed to effectively perform a given type of repurposing. For example, language repurposing may require a simple translation if the content are kept into separate files while cultural repurposing may require animations, sounds and game dynamics to be changed to reflect differences in gameplay and gestures between cultures, which demands for a lot more effort and specialized knowledge.

5. THE mEducator APPROACH

The scenario in a game can be split into two types: narrative and behavioural. The narrative scenario refers to dialogs and texts and how they will be displayed successively to the player. This part is easy to repurpose as only few things needs to be touched, and most of them could be available in separate non binary files. The behavioural part of a scenario describes how the game will react to user input (i.e. if I click on a blue door and I have the blue key in my inventory, then the blue door should open) and is very hard to repurpose because it is described as source code, not as a model, and thus needs specific skills (programming skills) to be understood. Some solutions exist, even if they are quite limited:

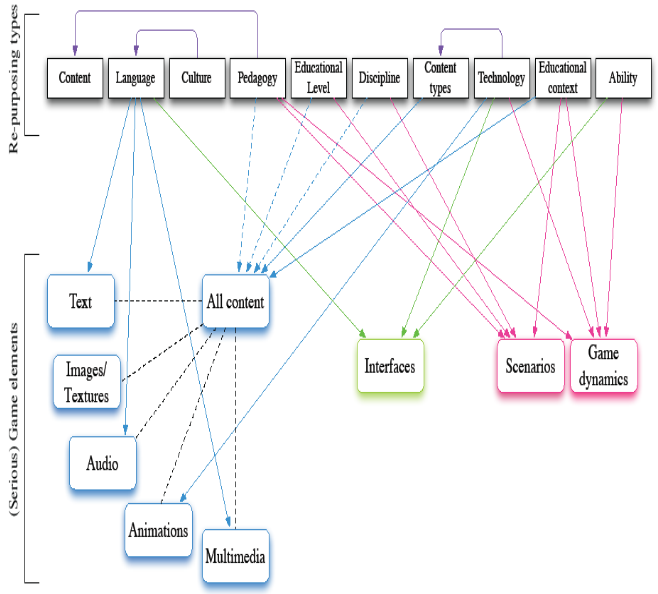


Figure 1. Hypothesised relationships between repurposing types and common game elements

- The first one consists in using a dedicated graphical editor that presents the model to the game designer, and hides the implementation. The limitation comes from the fact that this authoring tool is tightly bound to the game engine, and thus scenario files can't be used in a separate project using a different technology.

- The second approach lets developers use script files instead of compiled source code. The scenario is thus available, but still very hard to understand when you're not a computer scientist.

- The last solution consists in opening an API so that users can "mod" the game. It is the most open solution, letting other developer entirely modify and reuse the original creation, but is also the most difficult one, as it is mandatory to perfectly understand everything from the original work.

To handle all problems related to scenario repurposing, the mEducator approach aims to consider scenario as content, and not as plain text and source code. And so on, this content should be available in separate non binary files, should be described as a logical and understandable model, and a graphical editor should be provided to edit and author the model. More importantly, the model itself should be technology agnostic, meaning it doesn't need to be aware of the game engine that will run it.

6. THE MEDITOR

In order to have a generic tool that could allow game designers to directly integrate their designed scenario content into a game, the model needs to be extremely generic and technology agnostic. A mix between visual programming and workflow management was created, letting designers model how the game should react to user input. The model itself is represented as a directed acyclic graph, and is stored in simple XML files.

The agnosticity is achieved by declaring to the editor the features of the game engine, allowing the effective cross compatibility between game engine if they both share the same features. The XML files store references to game engine features as well as to branching generic features (such as “IfThenElse”). These files are read by the game engine that will “run” the scenario, calling at the right moment its own dedicated functions that were declared to the scenario editor.

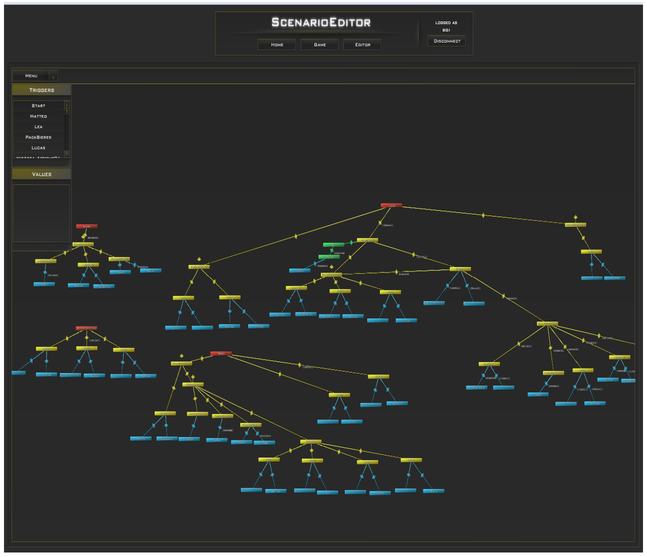


Figure 2. SUCCUBUS meditor for scenario-based repurposing

The model is split into six elements:

- Triggers (red nodes) are the entry points to the graph. They represent events fired by the game engine, themselves commonly created upon user input (i.e. “Player clicked on the door”, or “Player gained a level”)
- Branching nodes (yellow nodes) are used to choose between several paths in the scenario (ie. “IF player has the blue key THEN open the door, ELSE display a hint message”) or to go into multiple paths at once (do all of these following actions at once, or sequentially)
- Engine node (orange nodes) are references to engine features such as “Walk to a specific place”, “Display a message”, “Trigger a door open / close”, etc. These nodes needs to be declared at first so that the game designer has a set of features he or she can use to create some actual scenario
- Functions and operators (green nodes) are quite the same as orange nodes the main difference being that they return a value upon execution. They can be specific engine features (“HasItem” returns a boolean value indicating if the player has a given item in his inventory) or more generic (“+” adds two integer values together and returns the result)
- Values (blue nodes) are used to store text, number and boolean values that can be constant or change over time.
- Groups (purple nodes) contain a collection of other nodes and provide a simple way of using multiple times the same behaviour.

7. METHOD

The research method used in the current research is case study.

7.1 Material: the game

The case studies are based on the Happy Night Club game[1] developed by SUCCUBUS Interactive, a partner in the mEducator project. Happy Night Club is a serious game that explains to teenagers the dangers associated to binge drinking. It is a part of a bigger communication campaign (print, web, TV, radio) that was launched in the city of Nantes, France. Furthermore, the Happy Night Club scenario itself works as a test bed for the scenario editing and repurposing tools accessible on www.succubus.fr/meditor.

Figure 3. Happy Night Club game

All the files that can be found in the Happy Night Club game directory can be sorted into three different categories: content files, interface elements and game dynamics.

7.1.1 Content files

Content files contain every data used by the game, from parameters defining the rules of the game to text or images to be displayed on the screen. Objects other than text like images, clips or sounds are often referred to as assets in the context of a game project. Content is represented as XML files in this format allow for the structuring of heterogeneous types of content. Contents in the game are represented as a single record, and structured into many fields that can be text elements or numbers. Assets cannot be included directly in such files and are therefore referred to by their URL.

7.1.2 Interface elements

Interface elements consist of panels, graphic widgets, buttons and containers ready to display the content. As Happy Night Club is a web-based game, the interfaces are written in Adobe Flash. Adobe Flash is a programming environment for developing animations and graphics. A flash player is then required to play the files. In Happy Night Club, the entire game screen is a Flash panel that recursively contains other Flash panels. All these panels have been designed by web-designers.

The interfaces integrate and present the content from the XML files (texts, images, etc). They may also include labels for titles,

[1] The game is available online at http://www.succubus.fr/ and http://www.meducator.net/

tabs, buttons, etc. which are usually not represented as separated content but hard coded inside the Flash files.

7.1.3 Game dynamics

Finally, the game dynamics bring the interfaces and the content together to create the game. Game dynamics are written using scripts and algorithms in a specific programming language (e.g. C++, Java, C#, Flex etc.). The Happy Night Club game has been programmed in the Flash environment, from Adobe, which allows the production of rich internet applications (RIAs) which includes collections of predefined components and the ActionScript scripting language. Game dynamics have three roles; first, they are responsible for loading the content and the interfaces and building the main game screen, second, they load the content into the interfaces, and third, will listen to the interfaces for user events (button clicked, text input, etc.) and trigger specific behaviours accordingly (change a panel, update a text in the interface, etc.).

The game elements and the workflow presented in this section are commonly found in games and they are a benefit when designing a game. However, they can also be an essential step towards repurposing the content.

8. PROCEDURE

The process of repurposing was completed in five steps. First step included extraction, from the source code of the game, of all "string" variables. Second step included filtering the strings, to keep only those that needed to be translated. Next, the translation and the adaptation of the text file took place. After translation the reintegration took place: the translated text strings were placed back into the source code, at the appropriate position. Finally, testing and debugging made sure that all errors that might have been introduced during the reintegration were fixed.

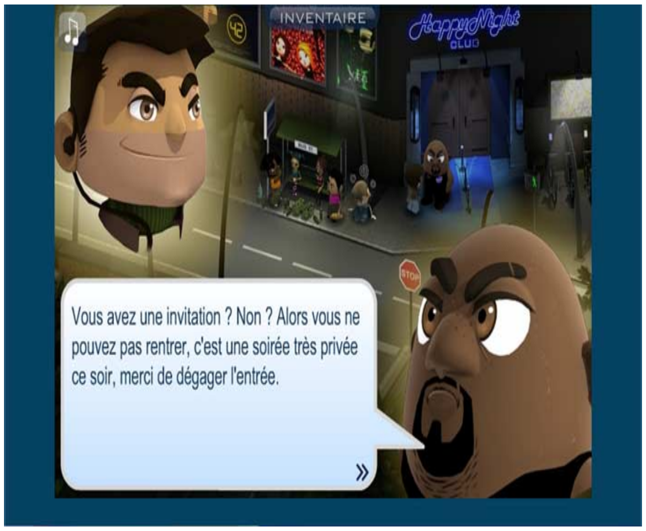
The work started by translating the text in the XML files. The work was conducted using Adobe Flex.

9. THE CASE STUDIES

9.1 Happy Night Club: Language Repurposing

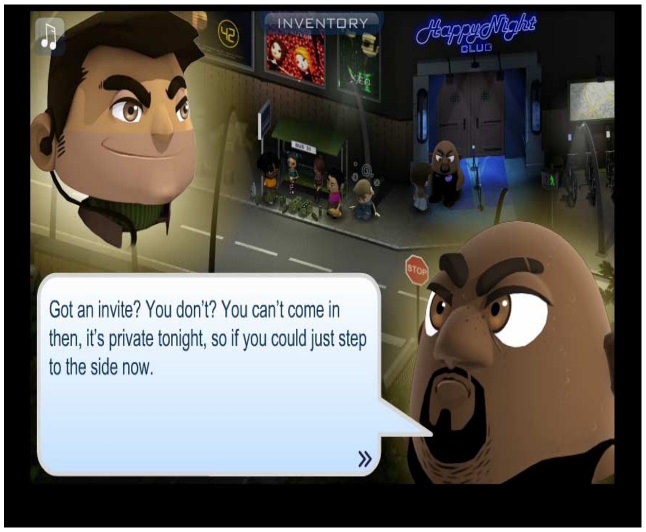
The first case study considers repurposing the game to a different language, which basically involves the translation of every text displayed in the game. The content-related parts were fairly easy to process as XML files can be edited by any text processor and translated without requiring any programming skills. Once translated, the content was seamlessly integrated into the game by the interfaces provided the right fonts have been embedded in the Flash panels.

The labels and tags on the Flash based interfaces are different. Not only does editing a Flash file require some programming skills but these files also come compiled as Small Web Files (SWF), whereas the FLA (Flash file in the Macromedia Flash authoring program) source files are needed to perform any modification. Although some software allow users to de-compile a SWF file into a Flash source file, it may not work properly and re-compiling the source file into a new SWF file that can be integrated back into the game can be problematic. The main suggestion consists of replacing the hard-coded labels by variables that can be separated in XML files just as any type of content. Figure 4 shows the French repurposed version of the Happy Night Club game. The text has been translated by editing all the XML content files using the mEditor.



Vous avez une invitation ? Non ? Alors vous ne pouvez pas rentrer, c'est une soirée très privée ce soir, merci de dégager l'entrée.

Figure 4. Happy Night Club game in French



Got an invite? You don't? You can't come in then, it's private tonight, so if you could just step to the side now.

Figure 5. Happy Night Club game in English

9.2 Happy Night Club: Scenario repurposing

The second case study consisted of repurposing the scenario of the game. Using the scenario editor we created a totally new scenario using existing characters and manipulating their actions. We made Jacques, the main character, to talk to Lucas the security at the club's door and we placed the entrance point at Lucas. In its properties we selected "Mouth"; an arrow appears and then we defined the actions that will follow the choice "act with mouth" on Lucas. Several actions will be performed sequentially; first move to be in front of Lucas and then a text bubble will pop up. Blue arrows indicate actions' parameters; *MoveNpc* needs the name of the characters to move to its destination, while *Info* requires the name of the character on whom the bubble will show up with the text content.

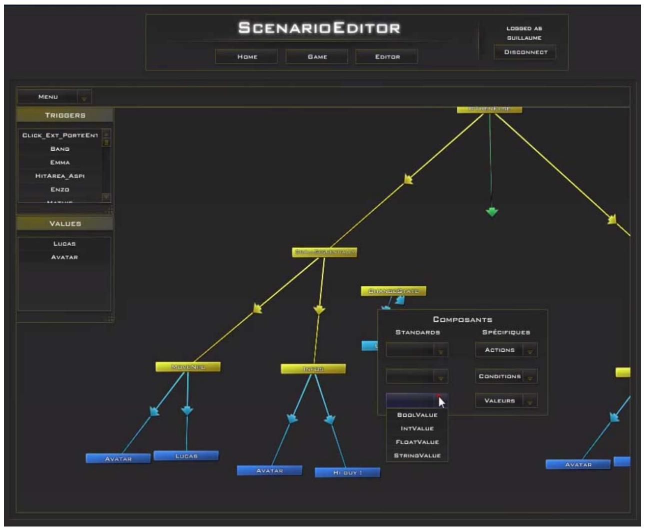


Figure 6. Happy Night Club new game using the Scenario Editor

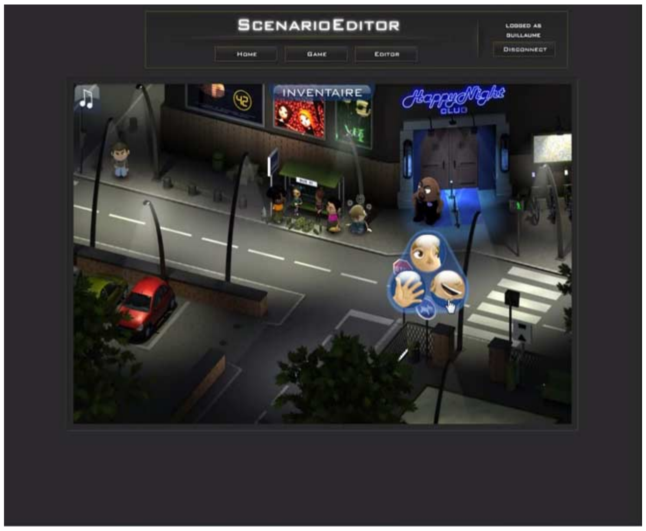


Figure 7. Happy Night Club new game

When playing the game, we act on Lucas by selecting the mouth icon available to us, as it can be seen in figure 7. Jacques then moves to Lucas' and when he arrives, the text shows (see Figure 8).

We create three values: the character's name, which will be used twice, so we select "module" in its properties, this value now shows up in the left hand side menu. We call that value "Avatar", its value is Jacques: the name recognized by the renderer. The name won't be reused so it's a temporary variable. For the text in the bubble when the character talks: the text constant used once (temporary). For the destination of our character, here we tell to the renderer to use the Lucas' position on the scene. We then save and test our scenario.

We can always go back to the editor, to improve or complete our scenario in order to introduce more dialogue or actions for the characters. We can copy the actions of our previous scenario so we do not have to start from the beginning and use them again.

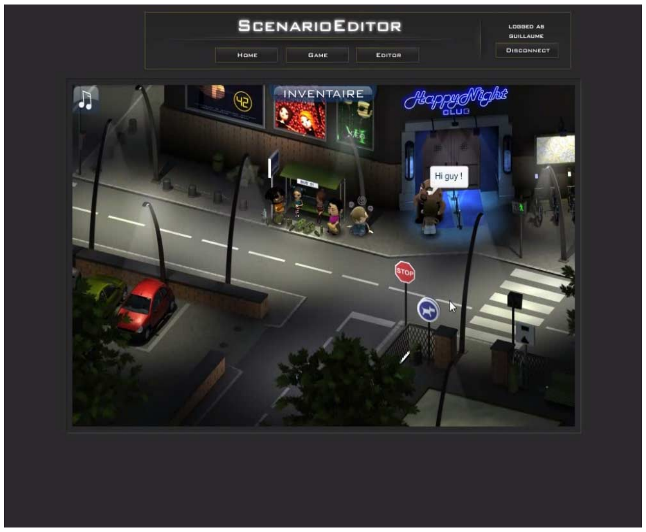


Figure 8. Happy Night Club new game with text

Now we can insert a condition with the *IfThenElse* action. This action leads to different actions following a conditional statement, if that statement is true, we will then have to introduce a *Then* statement to the left hand side action sequence, *Else* it will execute the right hand side sequence. To check that we have talked once to Lucas we can add a *ChangeState* action to the first sequence that will change the state of a Boolean value. We will put "true" from a temporary constant to a temporary variable which default state is false.

The conditional block is composed of an *IsTrue* operator that evaluate the variable's state and a *Not* operator, so that while the variable is false, the condition is true and we execute the left hand side sequence. Finally we change the text of the right hand side block.

To test the game we need to save and load the game again. We then can talk to Lucas however because it is the first time to do so Jacques says "hi", using the previously created action, the variable takes the "true" state and then we can talk to Lucas again, since the system knows that we have already talked to him once and therefore shows a different text.

In the same way we can carry on introducing more dialogues and actions to create the scenario that fits our aims. All the processes in the scenario editor are automated either in menu or drop down lists.

10. DISCUSSION

Due to the technical nature of game content, a limited knowledge amongst educators regarding how games are designed and programmed is still a limiting factor for widespread repurposing of game content. Consequently, game developers should realise the needs of educators who using serious games content as a relevant source of educational material and provide the tools and interfaces required to realise its potential. Therefore the aim of this work is to provide an alternative method of game repurposing based on the serious games scenario editor created within the mEducator project. To supply game developers and game companies with standards and recommendations that will help them to create serious games that can be easily repurposed and reused by educators either within the same context or not. The role of researchers is to provide game developers with recommendations and frameworks that would enable this repurposing at the lowest cost. To that end, future research should focus on developing technical solutions that can bridge the gap between technical skills required for the repurposing an external module, serious game, simulation or virtual world and an e-learning system.

These two case studies have been realised in the specific context of the mEducator project but they have led us to formulate different recommendations that readily apply beyond this context. Having the content separated is really important in order to facilitate the efficient repurposing of a game. A general recommendation would consider pushing the limit between what an educator cannot manipulate –due to a lack of programming skills or access to the source files– and what they can towards the latter. For example, regarding the interfaces, it is possible to have Flash considering a text field as a variable (a Dynamic Text Field in Flash terminology) by assigning it an identification name. Consequently, the same way content is loaded into the interfaces at runtime, the interfaces labels, buttons or titles can be separated into a file, preferably XML, and loaded by the game dynamics.
The same recommendation applies, albeit to a lesser extent, to the game dynamics. In some cases, they can be considered as content and stored in separate files, provided appropriate ways of representing them are found. Scenario-based games provide a perfect illustration. Usually, the scenario of such a game is scripted inside the game dynamics and therefore inaccessible for repurposing after being compiled. In the context of the mEducator project, research has been initiated to find ways of representing scenarios as content in order to ease their repurposing. A solution is to present a scenario loaded inside a bespoke editor such us mEditor. Therefore, the scenario can be edited and manipulated without programming knowledge by manipulating the menu and the actions via the drop down lists provided in the editor and then directly integrated into the game. Additionally, these elements can be reused to form a basis for the game without having to start from scratch. Educators' actions and decisions lead to effects, both immediate and simulation wide, altering the events, characters and situations encountered.

Another, great advantage of the editor is that it was designed so that it can be used with any game engine. However, "bindings" of the game engine must be provided to the editor, and that is made possible using xml files. Inside the game engine, some code must be able to read and interpret the xml files coming out of the scenario editor, but this is fairly easy to be developed, and runtimes for Flash, JavaScript, PHP and C++ can be provided.

Finally, the work on repurposing will assist towards reducing the development costs of serious games and expand the target market by supporting the process of localisation and by creating reusable games and game tools which in turn appeals both, to game developers as to the organizations that implement them.

11. CONCLUSION

This paper has presented a framework for serious game-based learning design, and interrogated how game content can be repurposed. Additionally, we presented the scenario based editor called mEditor created to assist in serious games repurposing. The case studies demonstrate that two new serious game learning objects were created by repurposing the Happy Night Club serious game. The conclusion makes clear that a separation between the content and the behaviour of the game is a great facilitator to such repurposing and the serious game editor offers the flexibility and the ability to the educators to create bespoke serious games to serve their learning objectives. With the scenario based editor, educators and users' decisions lead to effects, both immediate and simulation wide, altering the events, characters and situations encountered.

12. ACKNOWLEDGMENT

This paper has been fully supported by the mEducator project which is funded by the eContent*plus* programme by the European Commission.

REFERENCES

[1] Dovrolis, N., et al., "Depicting Educational Content Re-purposing Context and Inheritance", in *International Conference on Information Technology and Applications in Biomedicine (ITAB)*: Larnaca, Cyprus, 2009.

[2] Meyer, M., et al., "Requirements and an Architecture for a Multimedia Content Re-purposing Framework", in *Innovative Approaches for Learning and Knowledge Sharing*, W. Nejdl and K. Tochtermann, Editors, Springer-Verlag: Berlin / Heidelberg. p. 550-556, 2006.

[3] Zaka, B., et al., "Topic-Centered Aggregation of Presentations for Learning Object Repurposing" in *World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education (E-Learn)*, Las Vegas, USA, 2008.

[4] Verbert, K. and E. Duval, "ALOCOM: a generic content model for learning objects", International Journal on Digital Libraries, 9 (1): p. 41, 2008.

[5] Jovanović, J., et al., "Ontology of learning object content structure" in *12th International Conference on Artificial Intelligence in Education*, Amsterdam, The Netherlands, 2005, 322–329.

[6] Verbert, K., et al., "Ontology-Based Learning Content Repurposing: The ALOCoM Framework", International Journal on E-Learning, 5 (1): p. 67-74, 2006.

[7] Lenski, W. and E. Wette-Roch, "The TRIAL-SOLUTION Approach to Document Re-use Principles and Realization", in *Workshop on Electronic Media in Mathematics*: Coimbra, Portugal, 2001.

[8] Singh, G., "Content repurposing", IEEE Multimedia, 11 (1): p. 20-21, 2004.

[9] Bond, T.S., C. Ingram, and S. Ryan, "Reuse, repurposing and learning design – Lessons from the DART project", Computers & Education, 50 (2008): p. 601–612, 2008.

[10] Steiger, O., T. Ebrahimi, and M.D. Sanjuan, "MPEGBased Personalized Content Delivery", in *IEEE International Conference on Image Processing (ICIP)*: Barcelona, Spain, 2001.

[11] Hjelsvold, R., S. Vdaygiri, and Y. Ldautd, "Webbased personalization and management of interactive video" in *10th International Conference on World Wide Web*, Hong Kong, 2001, 129-139.

[12] El Saddik, A. and S. Hossain, "Content repurposing: Multimedia", in *Encyclopedia of Wireless and Mobile Communications*, B. Furht, Editor, Taylor & Francis, 2008.

[13] Hossain, S.M., A.M. Rahman, and A. El Saddik, "A Framework for Repurposing Multimedia Content" in *CCECE 2004/CCGEI 2004*, Niagara Falls, 2004.

[14] Burgos, D., C. Tattersall, and D. Koper, "Re-purposing existing generic games and simulations for e-learning", Computers in Human Behavior, 23 (6): p. 2656-2667, 2007.

[15] Protopsaltis, A., et al., "Repurposing Serious Games in Health Care Education" in *XII Mediterranean Conference on Medical and Biological Engineering and Computing (MEDICON 2010)*, Chalkidiki, Greece: Heidelberg, Springer, 2010, 963-966.

[16] Dovrolis, N., et al., "Depicting Educational Content Re-purposing Context and Inheritance" in *International Conference on Information Technology and Applications in Biomedicine (ITAB)*, Larnaca, Cyprus, 2009.

[17] Kaldoudi, E., C. Balasubramaniam, and D.P. Bamidis, "mEducator D.3.1 Content Repurposing: Definition of Repurposing Reasons & Procedures", 2010.

[18] Moreno-Ger, P., et al., "Rapid Development of Game-like Interactive Simulations for Learning Clinical Procedures" in *Fifth International Game Design and Technology Workshop and Conference (GDTW2007)*, Liverpool, UK, 2007, 17-25.

[19] Moreno-Ger, P., I. Martínez-Ortiz, and B. Fernández-Manjón, "The <e-Game> Project: Facilitating the development of educational adventure games" in *Cognition and Exploratory Learning in the Digital age (CELDA 2005)*, Porto, Portugal: IADIS, 2005, 353–358.

[20] van Est, C., R. Poelman, and R. Bidarra, "High-Level Scenario Editing for Serious Games", in *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Springer: Vilamoura, Algarve, Portugal, 2011.

[21] de Freitas, S. and S. Jarvis, "Towards a development approach to serious games", in *Games-Based Learning Advancements for Multi-Sensory Human Computer Interfaces: Techniques and Effective Practices*, M. Stansfield, T. Connolly, and L. Boyle, Editors, IGI Global: London. p. 215-231, 2009.

[22] de Freitas, S. and M. Oliver, "How can exploratory learning with games and simulations within the curriculum be most effectively evaluated? Special Issue", Computers and Education, 46 (3): p. 249-264, 2006.

[23] Tate, R., J. Haritatos, and S. Cole, "HopeLab's Approach to Re-Mission", International journal of Learning and Media, 1 (1): p. 29-35, 2009.

[24] Pappa, D., et al., "Game-based learning for knowledge sharing and transfer: the e-VITA approach for intergenerational learning", in *Handbook of Research on Improving Learning and Motivation through Educational Games: Multidisciplinary Approaches*, P. Felicia, Editor, IGI Global, 2011.

[25] Hulme, K., et al., "Experiential Learning in Vehicle Dynamics Education via Motion Simulation and Interactive Gaming", International Journal of Computer Games Technology, (2009): p., 2009.