Cost-effective Virtual World Development for Serious Games

Hao Liu, Yasmine Arafa, Cornelia Boldyreff and Mohammad Dastbaz

University of East London

Abstract – Developing a virtual world environment from scratch normally involves a large number of model creations, animations and event simulations. Such undertakings generally require a large amount of man-hours and expensive software. This paper introduces an open-source approach that will enable developers to easily and quickly create scenario-driven collaborative environments for serious games. A case study based on the development of a virtual crisis room in the Pandora project[1] is provided to demonstrate the effectiveness of the approach.

I. INTRODUCTION

A virtual world (VW) is a genre of online community that often takes the form of a simulated 3D environment, through which users, represented by avatars, can interact with one another, and use or create virtual objects [2]. Virtual worlds are normally used for entertainment purposes, such as socializing as in Second Life [10] and massively multiplayer online games as in World of Warcraft [14]. Recently VWs have become more popular for serious games for medical purposes, professional training, and military applications [5]. For example, Starlight Children's Foundation has been using VWs to help hospitalised children (suffering from painful diseases or autism for example) to create a comfortable and safe environment which can expand their situation, experience interactions they may not have been able to experience without a virtual world [4]. VWs have also been applied to training for emergency management. Simulation and modelling technology can recreate major disasters and emergencies from severe flooding and earthquakes, to large-scale terrorist attacks and CBRN dispersal events [3]. There are existing projects employing VWs such as TruSim [11] and Pandora [1], [9].

Developing a virtual world environment can involve creation of a large number of models, animations and event simulations, which requires a large amount of time and expensive software. Due to the current economic recession, many Small and Medium Enterprises (SMEs) and universities are looking for a more cost-effective way to develop VWs. In this paper, we introduce an open-source approach that will enable developers to easily and quickly create scenario-driven virtual worlds to achieve their goals.

II. BRIEF OVERVIEW OF VW DEVELOPMENT

Developing a VW from scratch is a very complex process. At the minimum level, it involves these key component developments:

- 3D Modelling: model 3D objects and avatars.
- Animation: temporal description of an object and an avatar, i.e., how it moves and deforms over time.
- Physics: this involves the introduction of the laws of physics (e.g. Newtonian) for the purpose of making the effects appear more real to the observer.
- Object Engine: it controls and gives behaviour to objects.
- Scenario Engine: it generates discrete-events that represent specific storylines (e.g. training scenarios) within the VW.
- Online Hosting: A VW is a distributed system that users can access through the Internet. Development includes constructing server-client architectures, addressing security issues, communication mechanisms, etc.

To complete these developments would consume a large amount of human resources (e.g. skilled programmers are needed for complicated programming from networking and physics to graphic and feature implementation), and also there is a need for expensive commercial software (e.g. 3D Max and Massive for modelling and simulation). For example, the well-known massively multiplayer online game World of Warcraft took roughly 4 years to develop, with a price tag of over 63 million US dollars [13]. Its development team size as of 2009 was around 900 [15].

A. Build VW based on existing Platforms

Where the development goal can be satisfied by a slightly small-scale virtual environment, fortunately, there are many existing platforms providing basic features of virtual worlds that can be used straightaway. They provide basic 3D models for simulating a real world (land, sky, the sun, avatars, etc.), basic animations (walking, smiling, sun rising, etc.) and on-line hosting capabilities. Developers can expand these virtual world stubs by creating objects, customising animations and adding scripts to control objects or avatars. This can be easily done through their built-in object creation tools and supported script languages, for example, LSL (Linden Script Language [12]) and OSSL (OpenSim Script Language [7]). Developing VWs based on these platforms significantly simplifies the development process.

Examples of such existing VW platforms are Second Life [10], OLIVE (On-Line Interactive Virtual Environment) [6], OpenSim (or OpenSimulator) [7] and Openwonderland [8]. Second Life is a popular online virtual world developed by Linden Lab which was launched in 2003. To host a customized VW, developers need to buy a land from Second Life. OLIVE, owned by Forterra Systems, has a range of license pricing models or hosting services for small-scale or enterprise-grade virtual worlds. OpenSim, in contrast, is an open-source server platform for hosting virtual worlds. It is compatible with the client for Second Life and can host alternative worlds with differing feature sets with multiple protocols. Openwonderland is yet another open-source

[1] Project FP7-ICT-2007-1- 225387. The project was co-funded by the European Commission under the mixed call on ICT and Security.

platform for creating collaborative 3D virtual worlds, developed by fully in Java.

After considering the above VW platforms, we chose OpenSim to be the development platform. The following reasons led to this choice: 1. it is open-source and free to use; 2. since it is developed by a public community there is no dependence on a single commercial vender; 3. it is compatible with Second Life, which has already been accepted by the public; and 4. instead of buying lands from Second Life, OpenSim gives us more flexibility in terms of hosting own servers, with full management capabilities and no limitation on the number of islands created.

III. Scenario-Driven VW Development

A virtual world used for serious games is usually a controlled environment where in-world events are governed by pre-set training or gaming scenarios. This is different from a public open environment used for socialising or online community building, where users have no constraints and no specific guideline for playing the virtual world.

However, OpenSim and other VW platforms were initially designed for on-line societies, not for scenario-driven virtual worlds. We need to add a scenario engine to make the virtual environment to adapt to our serious game usage. To achieve flexibility, the scenarios should not be hard-coded, instead, they should be easily re-configured as needed, i.e., plug-and-play.

A. *Scenarios*

A scenario is a synthetic description of a series of actions and events. It can be a linear storyline where the users cannot change the story line or ending of the story, or a branching scenario, where the users' decisions can affect the events and situations encountered. The events will drive the associated objects of the VW, for example: how NPCs (Non-Player Characters) move, when to play a video, how to respond depending on user's decisions and actions. Fig. 1 shows a part of a storyline used for training in crisis management. It is a series of events that define when and how to deliver the crisis information into the VW. In this scenario, the crisis information is delivered into the VW through in-world videos, maps, NPCs. etc. For example, the first event defines that the VW will play a video showing the weather news bulletin at timestamp 1. The storyline also contains interactive events that will ask the users to respond with their decisions via text input, multiple-choice, etc. According to users' responses, the scenario can correspondingly generate new crisis events.

A scenario can be stored into a XML file (Fig. 2). Once we have the scenario story, we need to send those events into the VW to manipulate objects and avatars.

B. *Add Scenarios into VW*

We use LSL script to give behaviours to objects and avatars in the OpenSim powered VW. There are three types of LSL functions that allow VW to communicate with programs running outside OpenSim: XML-RPC, HTTP-In, and HTTP-Request. LSL XML-RPC sends XML-RPC function calls [16] that remote system then handles. HTTP-In creates a temporary server in Opensim and enables outside sources to request data from scripts. While HTTP-Request enables LSL scripts request data from outside HTTP-accessible servers. Here we use the LSL HTTP-Request function because of the reliability limitations of other two with which we have experienced i.e. XML-RPC and HTTP-In.

Serial	Timestamp	Sender	Subject	Description	Channel
0	1	Media	Weather News Bulletin	The rain lasting for several days is rising the waters all o…	VideoContent
1	14	<PANDORA>	Fatalities amongst patients due to their evacuation from the Hospitals	Condition of several more patients worsens - Severe criticis…	DocumentContent
2	16	Met office	Weather Data Sheet	Weather Data Sheet	DocumentContent
3	21	Environment agency	Flood Map	Detailed river flood warning map	MapContent
4	51	EDFEnergy	Power Cuts	We are receiving a large number of calls reporting total los…	NpcContent
……					
11	261	<PANDORA>	Group Decision / Discussion Point	Participants should come together for a formaised collective…	TextInput ActionContent
12	301	EDF energy	Flooding of Switching Station Krsko	After flooding of the switching station an urgent decision i…	YesNo ActionContent

Fig. 1 Example of scenario storyline, from Pandora project [9]

```
<?xml version="1.0" encoding="UTF-8"?>
<timeline>
  <event>
    <serial>0</serial>
    <main>
      <Event subject="Weather News Bulletin">
        <sender>Media</sender>
        <desc>The rain lasting for several days is rising the waters all over the country...</desc>
        <autoPlay>true</autoPlay>
        <includeControls>true</includeControls>
        <modal>false</modal>
        <timestamp>1</timestamp>
        <content class="Video Content">
          <url>http://127.0.0.1/TempAssetStore/e9.mp4</url>
          <loopPlay>false</loopPlay>
          <volume>50</volume>
        </content>
        <classEvent>true</classEvent>
      </Event>
    </main>
    <recvtime>June 13, 1:15pm</recvtime>
  </event>
  <event>.....</event>
</timeline>
```

Fig. 2 XML representative of scenario storyline

Fig. 3 shows the architecture of the scenario-driven virtual world. The Scenario Engine firstly generates events according to the storyline timestamps at runtime. Once an event is generated, it will pass the event to the Scenario Web Server. The LSL Object Engine continuously pulls new events from the web server using *llHTTPRequest()*, and then manipulates related objects or avatars according to the events. In this way, what the users eventually see is a scenario-driven virtual world which is continuously developing on the basis of the scenario. Once the users perform some in-world actions, this information will also be passed back to the Scenario Engine through LSL HTTP-Request. An intelligent engine will possibly generate different events according to users' different reactions.

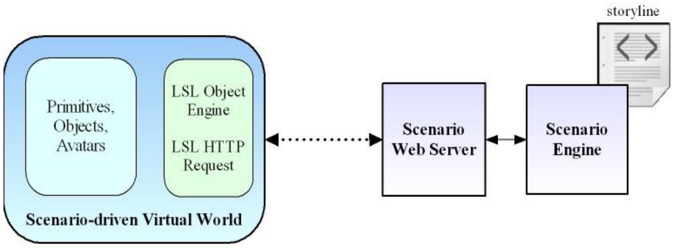

Fig. 3 Scenario-driven virtual world architecture

In this architecture, the virtual world hosting and the scenario setup are loosely-coupled. Users can easily reconfigure the storyline (which is suitable for the game scene) as needed, without making changes to objects and scripts in the virtual world.

IV. CASE STUDY - PANDORA VIRTUAL CRISIS ROOM

We have applied the approach to develop Pandora [9] virtual crisis room (Fig. 3). Pandora is a European FP7 ICT project, which is aimed to bridge the gap between table-top exercises and real world simulation exercises, providing a near-real training environment for Gold Command level crisis management at affordable cost. It authentically simulates all the dynamic events of the entire disaster environment and emulates an immersive and stressful crisis room.

The Pandora virtual crisis room is the "place" where a training exercise is conducted. It is comprised of a selection of audiovisual components and appropriate displays, communication and data delivery channels (Fig. 4). In this room, trainees (Gold Commanders) will receive a variety of crisis information over time and decide how to deal with the crisis.

By using the virtual crisis room, the trainees can log in remotely and engage in training exercise scenarios from geographically distributed locations. Participating trainees will each control a 3D avatar, represented within the virtual room, using their keyboard, mouse and/or microphone. Crisis events (e.g. Fig. 1) within an overall crisis scenario are generated over time by the Pandora scenario engine and passed to the VW.

When a training session starts, the VW continuously receives events from the scenario engine. As Fig 4 shows, when a new event comes, each trainee will firstly be notified by a pop-up event message. If this is a public event which is shared by all the trainees, the event and its multimedia contents will be immediately shown on these public big screens. Otherwise, if it is a private event, the information will be only shown on the trainee's in-world personal computer screen. For some events the VW requires trainees to bring up the built-in browser to view the contents. Those contents are normally maps or html documents which are not suitable to be shown on the virtual screens. In this case, trainees can click the screen to open the built-in browser and explore the document inside of the browser (Fig. 5). For interactive events, trainees (or a nominated chair) need to make a response within a required time (Fig. 6). Trainees can collaborate with each other by in-world typing, speaking or emailing as needed to discuss the crisis situation and make a group decision.

Fig. 4 The virtual crisis room. Area 1 is a pop-up text to show the description of a new coming event. Area 2 shows the description of new event on the screen; and area 3 shows the event's multimedia contents such as videos, images, maps, etc. Area 4 is the personal computer screen which will show private information for the user.

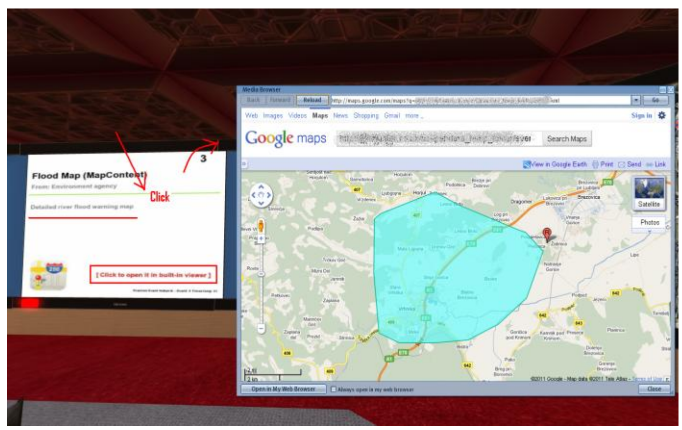

Fig. 5 Open the built-in browser to view the map content.

Fig.6 Ask trainee's decision by a pop-up multi-choice menu.

V. CONCLUSION

In this paper we have introduced a cost-effective approach based on OpenSim to easily develop a scenario-driven virtual world, and demonstrated its effectiveness by a real world example, the Pandora virtual crisis room. Although the quality of 3D models cannot compare with those developed by commercial software such as 3D max or Maya, this approach using freely available open-source software can be a good choice for SMEs and universities allowing them to develop small-scale virtual worlds for training and education purposes.

VI. ACKNOWLEDGEMENT

This work is funded by the EU Pandora project (FP7-ICT-2007-1-225387). We would like to thank the project partners for their collaboration.

REFERENCE

[1] Boldyreff, C., Dastbaz, M., Liu, H., and Arafa, Y., "Engineering advanced training environment for crisis management: The Pandora project", Proceedings of Advances in Computing and Technology Conference, pp.125-131, 2011.
[2] Bishop, J., "Enhancing the understanding of genres of web-based communities: the role of the ecological cognition framework", Int. J. Web Based Communities 5, pp. 4–17, November 2009.
[3] Cole, J., "Virtual emergencies: can technology replace live exercising?", Contingency Today Online Magazine, March 21, 2011.
[4] Dallas, U., "Avatars help asperger syndrome patients learn to play the game of life.", http://www.utdallas.edu/news/2007/11/18-003.html
[5] De Freita, S., "Emerging trends in serious games and virtual worlds", Emerging Technologies for Learning, volume 3, 2008.
[6] Olive, http://www.saic.com/products/simulation/olive/.
[7] Opensimulator. http://opensimulator.org/wiki/Main_Page.
[8] Openwonderland. http://openwonderland.org/.
[9] Pandora project. http://pandora.eupm.net/public.
[10] Second life. http://secondlife.com/.
[11] Trusim. http://www.trusim.com/.
[12] Heaton, J., Introduction to Linden Scripting Language for Second Life, Heaton Research, Inc., 2007.
[13] World of Warcraft Cost $63 Million, digitalbattle.com, http://www.digitalbattle.com/2006/06/15/world-of-warcraft-cost-63-million/
[14] Walter Simonson, World of Warcraft, Wildstorm, 2010.
[15] GDC Austin: An Inside Look At The Universe Of Warcraft, http://www.gamasutra.com/php-bin/news_index.php?story=25307
[16] XML-RPC, http://www.xmlrpc.com/