Virtual Reality Applications with User Interface for Dynamic Content Development

NIKOLAOS PAPASTAMATIOU,
THEOFANIS ALEXANDRIDIS,
KONSTANTINOS TSERGOULAS,
ALEX MICHOPOULOS,
Omega Technology
4 El. Venizelou Av., 17676, Kallithea,
GREECE
www.omegatechnology.gr
nikos@omegatech.gr,
info@omegatech.gr

NIKOLAOS V. KARADIMAS
Dept of Mathematics and Engineering Sciences,
Hellenic Military Academy
Varis-Koropiou Av., 16673, Vari,
GREECE
nkaradimas@sse.gr

Abstract: This work presents a methodology for the development of dynamic virtual reality applications for commercial use. Through a combination of state of the art technologies and methods, thee-dimensional representations of the real world (stores, buildings, etc) result in being independent of the objects they embody. The location and allocation of the objects in the virtual world and all information about them are dynamically set by administrators. These users in a "What You See Is What You Get" (WYSIWYG) web environment can add/change/rearrange the objects in the virtual world, along with the information about them. This methodology has been implemented with the combination of common software for developing 3D models, JavaScript, a custom .net web application and an XML data source. A paradigm of a virtual store is presented, where the owner can dynamically upload and manage its own products online (description, prices, multimedia material).

Key-Words: - Virtual Applications, Virtual Environments, Virtual Reality, 3D e-Shops.

1 Introduction

The usage of Information and Communications Technologies (ICT) has been one of the main drivers of changes within society and businesses since more than a decade. High speed internet has created new opportunities for the Information Technology (IT) sector to exploit state of the art technologies for commercial use.

In this work, the authors managed to combine different technologies and know –how to develop a virtual environment integrated with an authoring application aimed at managing objects into the virtual space. Through an ActiveX interface**,** the objects in the virtual space can be manipulated through java scripts. All the interactivity is accomplished through scripting that is provided by the 3D software, custom JavaScript, and a custom .net web application that updates an XML database. The interactivity is also integrated with flash files to add additional features to the solution. A paradigm of a virtual store is presented , where the owner can dynamically upload and manage its own products online (description, prices, multimedia material).

The paper is structured in the following sections: In Section 2 the need for dynamically managed virtual environments for commercial purposes is described. Section 3 presents the methodology and the platform functionality of the integrated system. In Section 4, the technologies used are justified and finally in section 5 conclusions are summarized and future work is discussed.

2 Problem Formulation

The evolution of the internet has led to the birth of Virtual Environments that offer a graphical thee-dimensional representation that simulates the real world. In some of these environments users are identified by a digital version of themselves: the avatar [1]. These virtual environments pose a new challenge for Firms. They offer Interactivity, Physicality and Persistence. There are several studies on how firms may advance their business models by developing and running a proprietary virtual environment platform [2]. Arakaji and Lang [3] have described a method for the evaluation of business value creation in virtual commerce. Cagnina, and Poian [4] made a study on how virtual worlds and second life can have an impact on the

business models of the firms. Guo and Barnes [5] answered the question why people buy virtual items in virtual worlds with real money and Susan Wu [6] provided some evidence on the amount of money spent on virtual goods giving real life examples.

The fact is that firms cannot ignore the business opportunities that surface. There are, however, two key aspects that virtual environments must tackle in order to succeed; ease of use and interoperability. Ease of use due to the target markets and interoperability with existing software applications the firms use.

Nowadays, several virtual worlds can be found in the web. The majority of them has been developed for gaming and a web browser is all that is required for the user to participate in the virtual world [7, 8, 9]. In such virtual worlds there is no need for a "thin client" to be installed in the user's machine. In the e-commerce sector, similar solutions for virtual stores exist as well as three dimension representations of their products. For example, *enjoy 3D Toy Store* [10] powered by amazon.com is a flash approach for simulating 3D experience for the customers.

3 Problem Solution

The proposed work managed to combine existing technologies, making a parallel study on the available software for presenting 3D models, to finally conclude to common techniques that can be applied to the majority of them. The goal was to develop a virtual environment platform, accessible through the most popular web browsers, where the owner of the virtual environment will have the possibility to alter and manage the objects within it through a user friendly interface.

A number of widely used software tools for presenting 3D models were assessed like Turntool [11], Unity3D [12] and Blink 3d [13], in order to select the most suitable one. Turntool [11] was preferred for testing our methodology since it offers a comprehensive scripting environment, interoperability with third party software (3D max, maya and daz 3d) and ability to run from every web server without the need of additional installations. The latter eases the transferability of the platform to the users' environment (e.g a customer's web server) or any hosting provider they use.

3.1 Developing the 3D Environment.

The development of the 3D environment has to follow some basic steps.

Initially in a vector editor software (CorelDRAW, Adobe Illustrator, Xara Xtreme, Adobe Fireworks, Inkscape), the floor plan of the virtual environment is designed. The floor plan is important in order to have enough room for the objects that will be placed later dynamically by the user. There are some key issues that should be taken into consideration during this phase. These are the width of the paths, so that the user may navigate easily in them and the heights (walls of rooms, or a building) that should have to be higher than normal, so that the user does not get the feeling of being huge in a small place.

Having finished the development of the floor plan, the next step is to import it into a modelling, animation and rendering package (3d studio max, Maya, Cinema4d, etc), where it can be easily turned into 3D with the existing tools that such programs offer. In this step, textures of real objects are used, in order to achieve a realistic and not "fake" feeling of the environment. An also important feature is the lighting, which has to be handled very carefully.

The next important step is to create the boxes where the products will be placed. Each box should be covered with transparent textures with unique names. The last step is to export the file choosing a number of parameters according to each case. The file is then loaded in the 3D software that will undertake the presentation of the virtual environment, where each object is adjusted depending on the desired functionality (being invisible, clickable and many other functions).

3.2 Loading the Objects

For the loading of objects in the virtual environment, a method has been developed that reads an xml data source with metadata for these objects. The method, for each object it reads, fires a relevant method in the 3D environment that loads the object in the designated by the metadata position of the virtual environment. Objects can be referenced either by name or by index. The name of an object is the original base filename of the object's file. The only prerequisite is that identifiers of objects in metadata should be mapped with transparent textures in the virtual environment.

The **loadObjects** method uses some code to read the XML data source along with a logical function to offer interoperability with all web browsers. Once the XML data source is read, the calculation of the number of objects to load follows. For each loaded object, the method retrieves the relevant metadata that will determine its behaviour in the virtual environment (e.g the available stock of a product

will determine if this product is visible, hidden or marked as currently not available). Some of these metadata appear also as basic information in the virtual environment (e.g. title, description, price). When the desired information is read, the method refreshes the virtual environment with the object in the designated place accompanied by its metadata. Fig 1, presents an instance of the aforementioned method for loading products in a virtual store.

loadObjects Method

Read XML Data source
Get the number of Objects to load
For Each Object in Number of Objects
- Get the physical file of the object
- Get the Quantity in the stock
- Get other metadata (Price, Title, Usage, etc)
- Load it in the virtual environment

Loop

End loadObjects Method

Fig 1: Example of loading objects in a virtual store.

While the Objects are being loaded in the virtual environment, one of the issues that arise is whether the virtual environment understands when this load has finished in order to allow user interaction (navigation in the virtual space and interaction with objects). It is possible to start navigation even if not all objects are loaded, but this has not been proved to be efficient, since the user sees empty places and the navigation slows down, due to the downloading of the objects (depends of the size and number of objects to load).

Timer Method

- Read XML Data source
- Get the last object in the list that should appear in the virtual environment
- Check the location in the virtual space where the object should be loaded to see if the source of the label matches the physical path in the data source.
- If Yes,
  - Inform the environment that loading has finished.
  - Exit Method
- Else
  - Continue
- End IF

End Timer Method

Fig 2: Method checking if all objects are loaded

To solve this issue, an additional method (Fig. 2) is required that runs repeatedly within a given time interval. This method checks to see if the last object of the list (XML data source) was loaded in the virtual environment. If the method succeeds, the virtual environment is informed that all objects were loaded and the navigation can start. Since the objects come dynamically from external XML files, the virtual environment cannot by itself identify when this has been completed.

The timer method should be called immediately after the loadObjects method with the desired timer interval. In Fig 3 this sequence is depicted.

On Load

Load Objects
Timer (Time Interval)

End

Fig 3. Firing the methods

3.3 Interacting with Objects in the 3D environment

Once all objects have been loaded, the navigation and the interaction with them in the virtual world is possible. The challenge again here is that the virtual environment does not really know the object the user decides to explore. The only information that it captures is the transparent textures (an area) the user has clicked in the virtual world. To show information of the selected object a kind of a reverse algorithm of the **loadObjects** method is used.

The new method(**ObjectClicked**) again reads the XML data source, searches in the XML file to find an entry with the same name as the id of the transparent texture clicked and retrieves the metadata of that object (pictures, videos, links to web sources, manuals, etc). Fig 4, depicts an instance of the aforementioned method for a selected product in a virtual store.

ObjectClicked Method

- Read Xml File
- Find the item clicked by label name
- Get the requested information
- Show the requested information

End ObjectClicked Method

Fig 4. Selecting an object in the virtual store.

To give a real life example, consider the case of a virtual store where the user, through the virtual reality, will have the feeling that he/she is in the real store, seeing products in the same shelves/areas as in real life. He/she will also be able to select a product by clicking on that particular product in order to have all the available information about it. In Fig. 5 a screen shot of a virtual store shows an instance when a user has selected a product from a shelf.

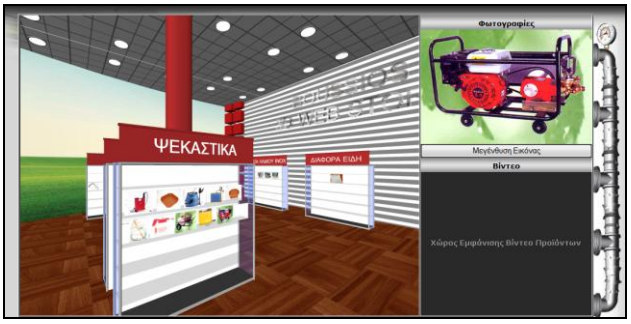

Fig 5. Example of selecting a product from a shelf in the virtual store.

3.4 Managing the Objects in the 3D environment

The advantage of the proposed work is that each virtual environment platform is totally independent of the objects it embodies. Changing the XML data source, that could be easily derived from any current application a firm uses (ERP, E-SHOP, etc), would result in the virtual environment showing different objects and info about them. So changing entirely the content of the virtual environment is a matter of some clicks. On the other hand, changing the virtual environment is quite simple. If the new environment has its transparent textures similarly named, the objects will be automatically loaded in the new environment without any additional overhead.

Apart from reading info from other applications, the proposed work provides a user friendly interface, in a WYSIWYG web environment, where the owner of the virtual environment can click on the objects inside the virtual space and can add/change/rearrange them at will.

This is achieved in a way similar to the **ObjectClicked** method that is fired when the visitor of the virtual space clicks on an object in it. The same method is called (Fig 4.) with the difference that the last line is replaced with one calling a web application that provides a user interface for reading and changing (delete, insert, update) the XML data about the objects and their positions (transparent textures) in the virtual environment. In the case of the virtual store (Fig 6.), the owner of the store can click on an empty space on one of its shelves to add a new product. Info about the product is added through a Web HTML editor, where the author can add text, links to web sources, images and other multimedia information. When finished, the information is added in the XML data source.

Fig 6. Example of changing/adding a product in a virtual store.

4 Technologies Used

The specific technologies used for the development of a platform to test our methodology was a software for presenting the 3D models (Turntool), JavaScript, a custom .net web application, an XML data source and some graphics tools.

TurnTool [11] is a real time 3D graphics solution targeted at the growing market for 3D on the Internet. TurnTool can be used for interactive viewing and manipulation - responding to user input from mouse and keyboard. It uses JavaScript as the main program script. TurnTool works with 3D Studio Max, Autodesk Maya and Cinema4D which enables the developer to export 3D scenes and make them interactive in the web.

JavaScript is an object-oriented scripting language used to enable programmatic access to objects within both the client application and other applications. In our work, it was primarily used in the form of client-side JavaScript that allowed the communication of the dynamic virtual environment.

The .NET framework introduces a new suite of XML APIs built on industry standards such as Document Object Model (DOM), XML Path Language (XPath), XML Schema Definition (XSD), and eXtensible Stylesheet Language Transformation (XSLT). The .NET Framework XML classes also include innovations that offer convenience, better performance, and a more familiar programming model that eases the manipulation of xml files. In our work, we used .NET to build a dynamic Web application that offers scalability, security and reliability.

Using the aforementioned tools, the floor plan of a store's building was initially designed with Illustrator software tool, in order to determine the location of the shelves and other objects. Afterwards, the design of the floor was imported into 3d studio max software tool, where it was turned into a three-dimensional environment with the tools that 3d studio max offers. After that, boxes on the shelves where the products would be placed were created. Transparent image files with the prefix "label" followed by a number from 1 to N were then placed on the boxes. Using the Turntool tab in 3d max, each label was set as clickable. Then the executable file was produced by Turntool and could be executed on any web browser. The JavaScript functions developed according to the aforementioned methods, were used to manage from then on the loading of the objects and the clicks of the users in the virtual environment.

Finally, Macromedia Flash tool was used to animate the process of loading until all objects were placed in the virtual environment as described in the timer method in section 3.

5 Conclusion

The presented methodology can be used with the majority of the 3D presentation engines, since it uses transparent textures, JavaScript and a standalone application (.NET) that undertakes the management of the information about the objects.

So no matter which environment one may use, this methodology can be customized to offer a solution in managing virtual environments. All 3D software for presentation of virtual worlds is somehow integrated with modelling, animation and rendering packages [14]. The XML extends these abilities and offers integration of these worlds with real data from other applications (ERP, CRM, e-shops) with only prerequisite to know the names of the clickable areas. Virtual World administrators, through an interface identical with the one the visitor uses, can rearrange the objects in the virtual space (e.g. products in a store's shelves), change the information about them, delete and replace them with other objects.

Further work in the area includes the research of ways to integrate some or all of the aforementioned features into the virtual environment. That means to manage all the information about the objects from within the virtual application without the need of the development of third applications. As technology evolves, such functionality is expected to be included in all 3D software.

Acknowledgments

This research has been supported by own funds of Omega Technology in the framework of the project with title "Research and Development of a dynamic virtual reality application". Examples of demonstration products of this research can be found for virtual stores [15, 16, 17, 18] and a virtual museum [19].

References:

[1] Castronova E. "Theory of the Avatar". *CESifo Working Papers Series*, No. 863, 2003.

[2] Cagnina M. R. and Poian M. "Beyond e-Business Models: The Road to Virtual Worlds". *Electronic Commerce Research*, Springer Netherlands, Vol. 9, No. 1-2, 2009, pp. 49-75.

[3] Arakaji Y. R. and Lang R. K. "Avatar Business Value Analysis: A Method for the Evaluation of Business Value Creation in Virtual Commerce". *Journal of Electronic Commerce Research*, Vol. 9, No. 3, 2008, pp. 207–218.

[4] Cagnina M. R. and Poian M. "How to Compete in Metaverse: The Business Models in Second Life". *Working Paper Series in Management and Organizational Studies*, No. 01-2007, Department of Economics, University of Udine, http://ssrn.com/abstract=1088779 (Accessed 15 November 2009).

[5] Guo Y. and Barnes S. "Why People Buy Virtual Items in Virtual Worlds With Real Money". *ACM SIGMIS Database*, Vol. 38, No 4, 2007, pp. 69–76.

[6] Wu S. "Virtual Goods: The Next Big Business Model". *TechCrunch*, 2007, http://www.techcrunch.com/2007/06/20/virtual-goods-the-next-big-business-model (Accessed 13 November 2009).

[7] Unity List of Games, http://unity3d.com/gallery/game-list/ (Accessed 1 December 2009).

[8] Case Library, http://www.turntool.com/showcases/case-library.html (Accessed 1 December 2009).

[9] Customer Showcase Metaplay, http://www.pelicancrossing.com/CustShowMetaPlay.htm, (Accessed 1 December 2009).

[10] 3D Toy Store, http://enjoy3d.com/toys/ (Accessed 1 December 2009).

[11] Turntool, www.turntool.com (Accessed 1 December 2009).

[12] Unity3D, www.unity3d.com (Accessed 1 December 2009).

[13] Blink 3d, www.pelicancrossing.com (Accessed 1 December 2009).

[14] Bukvic, I. and Kim Ji-Sun. "μ Max-Unity 3D Interoperability Toolkit", *International Computer Music Conference*, Montreal, Canada, 2009, pp. 375-378.
[15] Ramfos 3D store, http://www.ramfosmg.gr/virtual/virtual.asp (Accessed 1 December 2009)
[16] Vasarmidis 3D store, http://www.vasarmidis.gr/virtual/virtual.asp (Accessed 1 December 2009)
[17] Karamalegos 3D store, http://www.karamalegos.gr/virtual/virtual.asp (Accessed 1 December 2009)
[18] Boussios 3D store, http://www.emplonet.gr/3dstores/boussios (Accessed 1 December 2009)
[19] Museum of the Committee for Pontian Studies, http://www.emplonet.gr/3dstores/epm3d/k/ (Accessed 1 December 2009).