Evaluation of the virtual reality usage for the promotion of historical heritage

Chaowanan Khundam

Communauté UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA

COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

Spécialité : GI : Génie Industriel : conception et production

Arrêté ministériel : 25 mai 2016

Présentée par

CHAOWANAN KHUNDAM

Thèse dirigée par **Frédéric NOEL**, Professeur, Grenoble INP

préparée au sein du **Laboratoire des Sciences pour la Conception, l'Optimisation et la production de Grenoble** dans **l'École Doctorale I-MEP2 - Ingénierie - Matériaux, Mécanique, Environnement, Energétique, Procédés, Production**

Evaluation de l'usage de la réalité virtuelle pour la promotion de patrimoine historique

Evaluation of the virtual reality usage for the promotion of historical heritage

Thèse soutenue publiquement le **25 mars 2019**,
devant le jury composé de :

Monsieur Frédéric NOEL
Professeur, Grenoble-INP, Directeur de thèse
Monsieur Florent LAROCHE
Maître de Conférences, Ecole Centrale de Nantes, Rapporteur **Monsieur Frédéric MERIENNE**
Professeur, Institut Image - Arts et Métiers, Rapporteur
Monsieur Georges DUMONT
Professeur, École Normale Supérieure de Rennes, Président
Monsieur Jean-François BOUJUT
Professeur, Grenoble-INP, Examinateur
Madame Franca GIANNINI
Directeur de Recherche, CNR-IMATI: GENOVA, Examinateur

Evaluation de l'usage de la réalité virtuelle pour la promotion de patrimoine historique

Evaluation of the virtual reality usage for the promotion of historical heritage

Chaowanan Khundam

Acknowledgements

First, I would like to thank my advisor Professor Frédéric Noël for his supervision, support and suggestions during the PhD programme. He provided his professional advice and experienced guidance of the PhD research. Especially, when I need encouragement and inspiration to overcome the difficulties and problems encountered during my PhD study.

Second, I would like to express my sincere gratitude to the many friends, colleagues and experts at G-SCOP laboratory who took part in the research supported. Special thanks also go to the staff of Grenoble-INP for their good reception. I would like to thank Franco-Thai scholarship program and Walailak University for supporting me the three years awarding and providing with the facilities to complete this thesis.

Last but not least, thanks would not be complete without mentioning my family in Thailand and my Thai friends in Grenoble for providing me infinite love and supporting as always.

Chaowanan Khundam

Evaluation of the virtual reality usage for the promotion of historical heritage

Abstract

Digital heritage applications have been widely developed through Virtual Reality (VR) technologies as known as Virtual Museum (VM). Devices and digital contents are significantly increasing which support interaction system to immerse users into VM. In order to develop interactive application, interaction is always defined before creating application up to selected platform. It depends on a given content and devices which is limited when considering switching devices. Nowadays VR technologies is rapid changing, a platform to develop application should support devices changing and also to optimize interaction system to be used in VM. However, both devices organization and contents structure on a platform is still lacking efficient management to support alternative interaction in general.

We proposed a novel method for developing a VM application providing digital storytelling template to create interactive content and adaptive interaction system where an application is exportable into any device. We provide flexible editing tools for developer to manage the content structure. The interaction usage will be interpreted into a high-level abstraction and run on a low-level hardware device where interactions have been adapted. Storytelling will specify interaction behavior which can drive interaction in a virtual scene even device may be switched. Our adaptive interaction system supports to identify devices capabilities and storytelling expectation which could be selected for efficiency learning system and improve level of immersion. We claim that the development of adaptive interaction system will help us to find good presentation and interaction for any given content and devices capacities can be evaluated. This implementation is useful to deploy not only for the development of digital heritage applications but also for industrial engineering where interactive content and collaborative working are required.

Keywords: Virtual Reality, Virtual Museum, Interaction system, Interactive content

Evaluation de l'usage de la réalité virtuelle pour la promotion de patrimoine historique

Résumé

Les applications de patrimoine numérique ont été largement développées grâce aux techniques de réalité virtuelle, également appelées Virtual Museum (VM). Les appareils et les contenus numériques augmentent considérablement, ce qui permet le système d'interaction de faire immerger les utilisateurs dans la VM. Afin de développer une application interactive, une interaction est toujours définie avant la création d'une application sur la plate-forme sélectionnée. Cela dépend du contenu et des appareils, ce qui est limité s'il y a le changement d'appareil. A l'heure actuelle, les technologies de la réalité virtuelle évoluent rapidement. Une plate-forme de développement d'application doit supporter l'évolution des périphériques et optimiser les systèmes d'interaction à utiliser dans les ordinateurs virtuels. Cependant, l'organisation des périphériques et la structure du contenu sur une plateforme ne font toujours pas l'objet d'une gestion efficace pour soutenir une interaction alternative en général.

Nous avons proposé une nouvelle méthode pour développer une application de VM qui fournit un modèle de narration numérique pour créer un contenu interactif et un système d'interaction adaptatif dans lequel une application est exportable dans n'importe quel appareil. Nous fournissons aux développeurs des outils d'édition flexibles pour gérer la structure du contenu. L'utilisation des interactions sera interprétée dans une abstraction de haut niveau et exécutée sur un périphérique matériel de bas niveau où les interactions ont été adaptées. La narration spécifie le comportement d'interaction qui peut conduire à une interaction dans une scène virtuelle, même un périphérique peut être commuté. Notre système d'interaction adaptatif supporte l'identification des capacités des appareils et des attentes en matière de narration qui pourraient être sélectionnées pour un système d'apprentissage efficace et pour améliorer le niveau d'immersion. Cette mise en œuvre est utile à déployer non seulement pour le développement d'applications de patrimoine numérique, mais également pour l'ingénierie industrielle où un contenu interactif et un travail collaboratif sont nécessaires.

Mots-clés: Réalité virtuelle, Musée virtuel, Système d'interaction, Contenu interactif

Contents

List of Figures

List of Tables

Grossary

Virtual Reality (VR) The experience of being along with virtual environments.

Virtual Environment (VE) 3D models in a space displayed to user using real-time computer graphic.

Virtual Museum (VM) A digital entity refers to the organized presentation of museum items displayed in the form of exhibitions to complement, enhance, or augment the museum experience through personalization, interactivity and richness of content.

Virtual museum application A digital content of virtual museum applied VR technology.

Interactive content A content that needs user to participate more than simply reading or watching.

Interaction technique (IT) A method that user performs a task on the device.

Interaction system A group of devices connected together to the central controller having an organization about interaction in order to use an application with several devices.

Collaborative Virtual Environment (CVE) A platform to manage content and devices connected to an application which allows multiple users to manipulate virtual object asynchronous in the same VE.

User People who participate with an interaction system.

Visitor People who participate in a museum or a VM application.

Chapter 1

Introduction

1.1 Introduction

In digital era, thousands physical bricks of information are converted into digital content. There are many devices and technologies which allow users to access data and to visualize and interact with the virtual world. Virtual reality (VR) technologies have been used for various purposes. Virtual Museum (VM) is one branch that applies VR for demonstrating actual objects or architecture with three-dimensional models. VM applies VR concepts with multimedia information and computer graphic technology. The design and development of VM exhibition consider interactivity in the system to allow users to learn about collection of history with content that they are interested in (Falk and Dierking, 2016). VM becomes a large multimedia center where users can easily access information and convey content through interaction (Walczak et al., 2006).

Web-based VM or Web3D exhibition (Martínez et al., 2016) is a kind of VM so far representing the content in term of multimedia online. Interaction with these applications is normally facilitated through traditional input devices by mouse and keyboard. Nowadays, there are a lot of devices available for supporting VR application such as touch screen, tracking devices, head mounted display (HMD) etc. These devices have been applied to VM for more interesting content and increased learning performance. VM has more abilities to immerse users into the context of museum to be a new format called interactive VM (Carrozzino and Bergamasco, 2010). That means users are able to interact with various devices instead of using a desktop device as a web-based VM. Having interactive content allows users to gain knowledge of the museum more easily, but the keystone remains a good storytelling. We need a storytelling to drive virtual environment behavior while the interaction is the modality to follow this story. Storytelling becomes a massive part to create interactive content for a VM and use stories as instruments for suspenseful knowledge transferring (Marchiori et al., 2012). Interactive content is a kind of storytelling where users get involved in the story by interacting with content.

In order to create a VM application, the platform will be the tool to help developers defining elements in a scene and makes stories called “Storytelling platform”. To design and develop an application as interactive content, activities will be considered in term of user interaction where an interaction system is concerned. Here, devices and interaction techniques are also considered to design a story as an interactive content. Storytelling platform assists to organize the story and provides tools to design a story with interaction. However, to develop an interactive VM application concerned about using an application on various devices different from web-based VM which runs on desktop device only. Interaction with content is not just point and click any more, but related to interaction techniques that depend on the

selected device. Most of VM applications have been designed for the selected devices. There is no management for an application to support technology change over time. Maintenance and service of interaction system will be restricted to the content of all development process which is a complex application management. Storytelling platform provides interaction management with story design of interactive VM application development. We propose a high-level abstraction to define all events and actions in a scene to represent into low-level user interactions. Entity, Event and Action are the main components of our storytelling model that support scene creation and connection to devices of high abstraction level. Thus, interaction techniques are applied at high abstraction level which allows the system change devices and support interaction management.

In this thesis we develop a Storytelling platform for interactive VM application. Then our application is allowed to change devices and interaction techniques with the same interactive content. Here, we can study the effect of interaction of each device capability for learning the provided content. This process will help us to analyze devices and interaction techniques to classify their specific potential to support user learning in VM exhibition and also for industrial engineering where interactive content and interaction system are required.

1.2 Research motivation

I am a lecturer at Multimedia Technology and Animation (MTA), School of Informatics, Walailak University, Thailand. There are many subjects I learned from experts such as 3D modeling, 3D Animation, Motion Capture, Special Effect, Mathematics and Physics for Programming, etc. This knowledge (very useful to do research) works on multimedia and cross disciplinary to other fields. In particular, I am interested in VR focus on VR interaction applied for historical heritage site. Wat Phra Mahathat Woramahawihan is the archaeological temple located in Nakhon Si Thammarat province that we would like to preserve and publish into a digital content using VR technology. As the first project with my students, we develop a walkthrough system on the desktop device using the game engine Unity3D. This engine already provides first-person assets and third-person viewpoint to develop application like shooting game and also provides real-time rendering engine. Thus, the main part is about creating virtual environments to let users visit and perceive cultural art of this temple by using desktop device. This project finished and we have a walkthrough system, then we plan to make more interaction for the next project by reusing virtual environments and are looking for funding to have more interaction abilities.

Thailand Digi Challenge 2014 competition organized by SIPA (Software Industry Promotion Agency) has accepted my project proposal to create a VR project about archaeological site and ancient monument in the south of Thailand. In this project we desired to improve virtual environments more effective and realistic. We are aware of the realistic simulation engine Unreal Engine 4 (UE4) published using for free. I have experience with UDK (Unreal Development Kit) before and UDK has a good real-time graphic rendering that could be more interesting for high-end game engine UE4. UE4 also support the devices Oculus Rift and Leap Motion that we would like to study in this project. Even though Unity3D supports these devices as well, but we point to realistic simulation and UE4 is better. So, we decided to move all resource imported to new engine UE4.

Here, we found that even using the same 3D model assets, the process to build virtual environments in a new engine is totally different. That means we have to develop a new project with the same virtual environments. However, assets organization should revise again due to the fact that we produced high polygon models in the previous project. When the virtual environment part was finished, devices connection and interaction were developed. We introduced Head Mounted Display (HMD) to visualize immersive virtual environments. Oculus Rift is a HMD device supported by UE4 that can be connected to the project directly

without any installation just enabling a VR mode in preferences.

So, we observe virtual environments of the temple with first person view point 360 degrees all around using Oculus Rift by moving head. The next device, the Leap Motion is introduced to be an avatar controlled by hand gesture. As mentioned, Leap Motion is supported by UE4 and capture hand gesture (either left or right hand). We attached this device in-front of the Oculus Rift and user is able to make a command through hand gestures like body-free while using Oculus Rift. We put a different type of gesture for movement control in VR. So, user can use these devices to observe all virtual environments of the temple by moving and interaction as a first person perspective view. This project finished, we have a more realistic virtual environments of the temple with simple interaction system which allows users immerse to the virtual temple like they visit in the real cultural heritage site.

At school of informatics, Walilak University, we have collaboration with Institut Polytechnique de Grenoble (INPG) in France. Every year there will be master students from INPG to do projects as internship students. Our major Multimedia Technology and Animation (MTA), we facilitate many tools support for project about AR/VR but also 3D printing. It opened an opportunity to meet Dr. Frédéric Vignat who was the supervisor of those projects. Many times I discussed with him about my ambition to apply for a PhD about VR and interaction especially in the topic as mentioned before, but he was not a specialist of VR and interaction field. So anyway, he suggests me to contact Prof. Frédéric Noël who is better in this field. After that I introduced myself to him what I want to do and share an idea about VR and interaction problems. Then we concluded that VR and interaction applied to VM is an interesting project. In 2015, I was funded by the Franco Thai Scholarship for a PhD at INPG with Prof. Frédéric Noël as my advisor. He introduced GSCOP laboratory where it exists a platform to develop an application called CVE (Collaborative Virtual Environment) where many devices can connect together. CVE is used to develop projects which various devices sharing the same VE and synchronized in real-time. With CVE performance we can apply to solve the problems about devices organization and storytelling for a VM which are interesting for my thesis topic. By the way CVE can solve some parts of devices connection, but the main part of the PhD is about interaction and storytelling synchronization.

1.3 Research context

With our research motivation, we thought that our VR temple is not good enough for learning. We should provide more historical information and details. Due to the interactions put into the project, users could do something better than just walkthrough and observe. We would like to provide interactive virtual environments that users are able to interact with. Moreover, recently we have many new devices to work with VR project and supporting user interaction. VM uses digital content and interaction system to let visitors learn by participating with content. We would like to improve our project to be a VM where users can interact with the story by various devices. Here, we found there are two components to develop VM which are storytelling and interaction models. The storytelling model is the main part to create interactive virtual environments that should synchronize with interaction design. The interaction model must derive the connection of devices with the application. The problem is that game engine doesn't have the tool to support story making directly which necessary for interactive content creation. Anyway, there are some authoring tools providing story making, but they do not support device connection and interaction. They do not allow to switch from one device to another one transparently.

When we developed an application the content is linked to the devices. Interaction techniques that used an application have been designed for a specific device. When expect to change a device, we must to reproduce all process of the interaction assignment while the

content basically remains the same. For Wat Phra Mahathat Woramahawihan temple we used an Oculus Rift and a Leap motion.We encoded hand gesture for movement control driven by the Leap motion device and synchronized with Oculus Rift (head tracking controls gaze direction). When Oculus Rift or Leap motion is switched to other devices, we must reprogram again all interaction techniques for the new selected devices. Even if the content changed while all devices are still the same, such as walkthrough virtual tour content switched to the manipulation of virtual objects all interaction techniques are not available any more. So, there are issues to be considered:

- Most VR platforms are proposed to be a tool for VEs development and support developer to handle devices configuration. However, storytelling to support and to organize story structure is still lacking storytelling and interaction management should be considered for complete proposal.
- High-level abstraction model should support the definition of object behaviors to ease switching between devices.

We expect to develop an interaction system, but providing a similar experience whatever the device finally used.

Eventually, storytelling model will make a good communication with our interaction system. In this system, we make any project into a single system and we connect any device with very few adaptation to use devices and lower competencies. It allows users to interact with the interactive content and devices which boost up the abilities of learning. The added value of this work is to identify the limit between a model used with any kind of device and a model which need specific development for working on a new platform. And the objectives of this work are:

- To develop an adaptive interaction system providing high-level abstraction for switching devices, to investigate suitable devices for manipulating virtual environment.
- To develop a complete process of storytelling for designing interactive content dedicated to VM which allows users interact efficiently with the proposed knowledge.

1.4 Research questions

Development of interactive VM application that can change devices while maintaining the same content needs understanding devices interactions and interactive story. The interaction issues concern device capabilities, interaction techniques and interaction tasks. Interactive story issues are related with historical data, 3D models, scenario, all about story making and how to connect event within the scene to enable interaction. This thesis addresses the platform development and the study of devices capabilities to create interactive content. Then the questions about platform development are raised:

- How can we define a high-level abstraction interaction system development where interaction tasks to support displays and devices switching adaptability?
- How to define a storytelling model which provides interactive content creation associated with interaction tasks?

To study devices capabilities, we need the platform to create the content in term of interactive VM application as mentioned above. We introduce a top-down design platform where all interaction issues and storytelling model are defined at top abstraction. Device implementation will be a low-level function and is driven by the high-level abstraction. The research issues are then divided into the following questions:

- How do different interaction systems affect user's performance to accomplish interaction tasks?

- How do different interaction systems affect user's need of interaction?
- How do different interaction systems affect user learning?

Each interaction system may have different characteristics that affects the user engagement. When considering all research issues, we state the research question to study and explore device usage and user engagement as follows:

"How do different interaction systems affect user engagement in VM applications and what are the key factors?"

1.5 Methodologies

The overall steps of this research from initialization of the Storytelling platform for experimentation to evaluation and conclusions are the steps of the research process.

1. Development of Storytelling platform, high-level abstraction and event editor

First, an information system is developed to create a "Historical model" linked to the Storytelling model where the data is transformed to become an interactive content using classes and properties defined from formal ontology. Event editor applied an ontology to build a story using data from the Historical model. In the Event editor events and actions have been designed and implemented for a visual setting in the Storytelling platform.

2. Development of an interaction system, low-level functions connection to device for implementation

The low-level connection is a part of a top-down design workflow. High-level abstraction is decomposed into a lower level for physical implementation. This is compatible with vertical transformation as proposed in the Model Driven approaches (Calvary et al., 2003; Coutaz and Calvary, 2012). High-level abstraction related to the entity as defined while event and action allow developer to connect the device which is independent of event setting anyway. Here, storytelling platform enables device changing through events and actions by the event editor and low-level functions are applied by interaction following device characteristics.

3. A case study of historical story is transformed to be an interactive content

We expect to build a VM application about the history of Wat Phra Mahathat Woramahawihan, the crucial temple model of South East Asia. The storyteller expects to make a VM application to tell this story with the Storytelling platform by transforming history to be an interactive content. The story is modelled at a high-level abstraction by the event editor.

4. Interaction system evaluation

To investigate device potential, quantitative and qualitative variable will define performance metrics for experiments and for each device we evaluate device potential with different IT. Then for each application, we define an efficient interaction system to be used in a VM. Finally, the Storytelling platform and interaction system will be deployed to maintain a VM application and to support user learning in VM. The goals of this evaluation are to find device potential, to find the appropriate interaction system and to find the factors that engaged user into VM.

1.6 Thesis report organization

The structure of this report is organized into six chapters:

In Chapter 1, we describe the research motivation, the research context, the research questions, research methodologies and the overall structure of the thesis.

In Chapter 2, we describe the current components in the Virtual Museum (VM). The classification of VMs is proposed which are digital media, VM online, interactive Web3D VM, mobile application VM and interactive on-site installation. VM development process of each classification is then explained. Finally, devices and interaction of each type are described.

In Chapter 3, the Storytelling platform is introduced. The VR framework and authoring framework are described to be the tools for VM development. Then, we proposed the Storytelling conceptual model with design and development of story model. Finally, the development of the Storytelling platform is described.

In Chapter 4, we use the Storytelling platform to VM application as a case study. First, we prepare three interaction systems by designing interaction techniques for each system. Then, the case study of the story of Wat Phra Mahathat Woramahawihan is developed. We will use this case study to investigate how different interaction system and story affect the learning outcomes.

In Chapter 5, two experiments of virtual tour and semi-guided story are evaluated. Some statistics are used to analyze the results of these experiments. The conclusions explore contribution to better immerse museum visitors into VM application.

In Chapter 6, the conclusions and perspectives of this research, we show the summary of the results starting from the proposed research question to the contributions to knowledge. We discuss the limitations in this research, our perspectives on this research and future works.

Chapter 2

Current Components in Virtual Museums

2.1 Introduction

The state of the art in this chapter presents the context of Virtual Museum (VM) development. The aim is to gather and share conceptual knowledge of VMs and identify existing museum formats to investigate new opportunities for VM development. The definition of VM, including its meanings and the various formats of VMs, as well as their evolution are discussed. Each VM format provides examples how technology is used to present information to VM visitors. This section provides an overview of the evolution of the VMs, with the introduction of technology to support VM development. This section also provides information in a way that makes it easier for VM visitors to access content, which will be beneficial for the adoption of new technologies.

Then the section about VM development process describes the tools and platforms that are used to develop VM applications. Because of the different technologies and devices used to develop VM applications, the platforms to develop VM are also different. In this section, we present the evolution of platforms, including features that make platforms suitable for developing VM applications.

VR technology associated with adapted interaction devices increases the level of immersion of the user and thus increases interest for VM development. Then the development of VM must be connected to a variety of interaction devices. The section about devices and interaction describes existing devices, as well as a classification, respect to their characteristics affecting the learning of the user.

The final section discusses the results of the review issues just mentioned to study the problems and gaps for the development of VM. It provides keys to assess the performance of the device on the learning of the user through VM development.

2.2 Virtual museum

According to Virtual Multimodal Museum (ViMM) definition (Polycarpou, 2018), a Virtual Museum (VM) is a digital entity that draws on the characteristics of a museum, in order to complement, enhance, or augment the museum through personalization, interactivity, user experience and richness of content. Both the physical museum and the VM share a common commitment to the institutional validation of content and quality of experience

through curatorial process, inherent in the definition of museum defined by the International Council of Museums (ICOM) (Anfruns, 2014).

As with traditional museums, a VM can be designed using specific objects, which can consist of online exhibitions created from primary or secondary resources. Moreover, a VM also refers to mobile application offering traditional museums to display digital representations of its collections or exhibits. VMs are usually, but not exclusively delivered electronically as a digital content such as virtual 3D environments and digital artwork which are denoted as online museums, hyper museum, digital museum, cyber museums or web museums (Hazan et al., 2015).

This section discusses the classification of VMs in terms of format, content design and access methods, which are digital media, VM online, interactive Web3D VM, mobile application VM and interactive on-site installation. Details of each VM development and examples of applications are described in the next subsections.

2.2.1 Digital media

Before the existence of web browser and fast network connectivity, multimedia technology has played a role in making media accessible. In traditional museum, physical data has been digitized to be a new media of cultural heritage collections. Personal computer (PC) and desktop devices are the tool to access these media. It allows visitors to navigate a 3D simulation. In order to develop a digital media content, software to handle media is expected. We call it the multimedia framework. A good multimedia framework offers an intuitive user interface and a modular architecture to easily add support for integration of audio, video, container formats and transmission protocols.

The platforms, such as QuickTime (*QuickTime and the Rise of Multimedia* n.d.) and Macromedia (Bouhlel, 2010), are an example of multimedia platform for building interactive multimedia applications. They were used for the creation of the majority of educational CD-ROMs. Increasing computational power of PCs, the popularization of 3D-accelerated graphic cards and real-time 3D navigation enable further pre-calculated digital environments and create a quasi-real experience of virtual spaces. One famous application was introduced in 1992 by Apple Computer Inc. (Huhtamo, 2010). The educational CD-ROM was developed through their software QuickTime, entitled “The Virtual Museum”. Scientific exhibits were displayed in a pre-rendered environment and the user could navigate from room to room and view detailed information about the exhibits. Interacting with objects in this space is called virtual navigation (Kang and Desikan, 1997). This application uses real-time video compression to display and interact with high-quality computer animations. 3D objects are also used in animation sequences, which allow the selection of 3D objects to trigger movement within the 3D space and to examine animations or play a digital movie or soundtrack.

2.2.2 VM online

The development of WWW (World Wide Web) in the early 2000 made the data available in the form of a web page to make it easier for user access. At that time, web pages looked simple, bandwidth was lower, the concepts of online museums were still under development and there was limited multimedia technology available within the web browser. Some online museums reproduce existing physical museums. They start to publish within electronic formats thank to the increasing digitization of cultural heritage answering to an increasing demand to access collections. Museums are increasingly using their websites to communicate with their visitors and found that experience of online VM significantly and ultimately improve visits of the museums (Padilla-Meléndez and del Águila-Obra, 2013). The goal of online VM is to preserve content for future generations and support its use and management

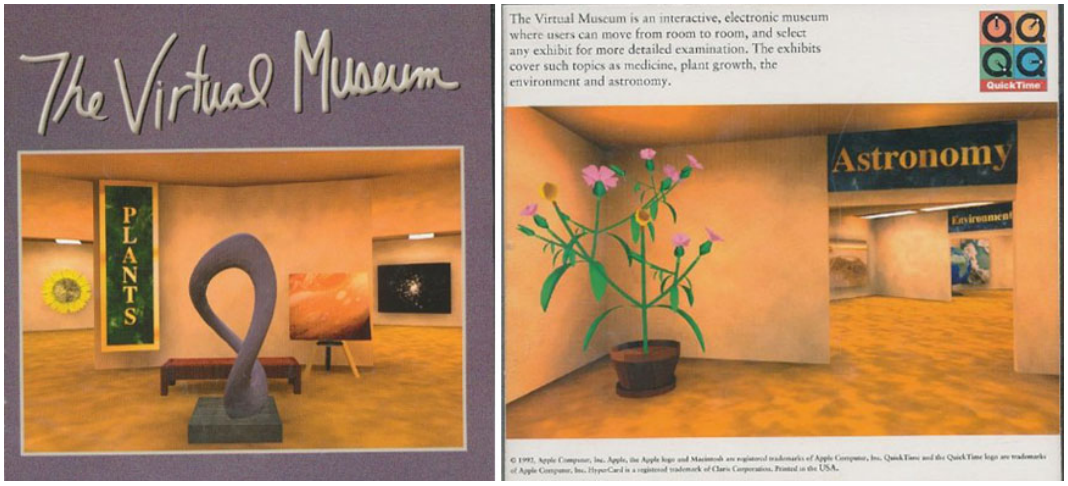

Figure 2.1: The first digital VM, released on CD-ROM in 1992 by Apple Computer Inc. (Retrieved from https://www.inexhibit.com/case-studies/virtual-museums-part-1-the-origins/)

over time.

Information in term of hypertext and hypermedia is applied to present the data interconnected by hyperlinks, which are typically activated by a mouse click, keypress sequence or by touching the screen. The concept of WWW has hypertext and hypermedia implemented on web pages which is mostly written in the Hypertext Markup Language (HTML). HTML enables information publication over the internet. Internet connection is required to access this information via desktop device. Point and click remain the principle interaction for this modality. A web browser receives HTML document from a web server and interpret into text, images, and other material on web pages where user accesses these content. HTML also embeds programs written in a scripting language such as JavaScript which affects the behavior and content of web pages; it makes the content more dynamic.

The following examples recall some online VM developed in the early 2000, at the beginning of hypermedia and online media development. Users interact with the media via hyperlinks to access the information and choose to watch the media as needed.

The Virtual Museum of Computing[1] (VMoC) is an eclectic collection of links and online resources including links to other related museums around the world, as well as having its own virtual galleries of information concerning the history of computers and computer science. This VM was founded by Jonathan Bowen in 1994.

The Virtual Museum of New France[2] is an online resource created by the Canadian Museum of History. The site includes interactive maps, photos, illustrations and information which proposed to share knowledge and raise awareness of the history, culture and legacy of early French settlements in North America. This VM was launched in 1997 and expanded in 2011.

WebExhibits[3] is a VM of science, humanities, and culture that uses information, virtual experiments that encourages visitors to think about and explore scientific and cultural phenomena from a variety of angles in new ways. The site launched with two exhibits on Calendars and Daylight Saving Time, as a complement to www.time.gov. This VM was founded in 1999 by Michael Douma at IDEA, a nonprofit organization working in the area of scientific and cultural literacy.

[1] http://www.historisches-centrum.de/vlmp/computing.html
[2] https://www.historymuseum.ca/virtual-museum-of-new-france/
[3] http://www.webexhibits.org/

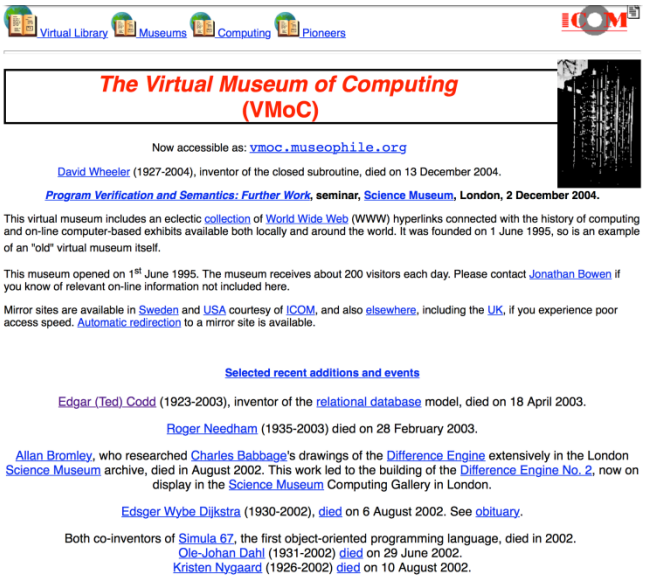

Figure 2.2: The Virtual Museum of Computing on webpage (Retrieved from http://www.historisches-centrum.de/vlmp/computing.html)

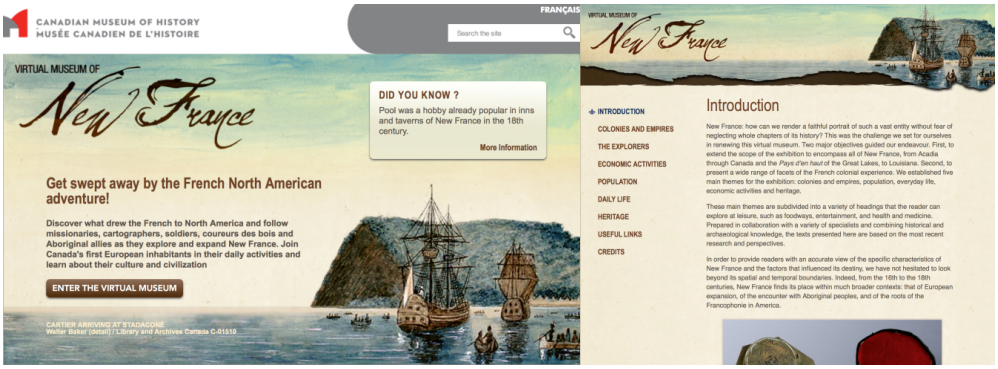

Figure 2.3: The Virtual Museum of New France on homepage and introduction page (Retrieved from https://www.historymuseum.ca/virtual-museum-of-new-france/introduction/)

Figure 2.4: The VM of WebExhibits on homepage and link to "Why are things colored" page (Retrieved from http://www.webexhibits.org/causesofcolor/)

2.2.3 Interactive Web3D VM

Following 2000, internet technology developed new potential to offer virtual visitors access through the WWW to a VM environment. In addition, the enhancement of Internet connectivity such as ADSL, 3G, 4G enables the delivery of big size media files related to the invention of VM exhibits. The most popular technologies for WWW visualization is Web3D which can be used to create interactive VMs.

Web3D Consortium[4] is an international, non-profit, member-funded, industry standards development organization. It supports the X3D specification, designed for sharing interactive 3D graphics on the Web between applications and across distributed networks and web services. Web3D offers formats and tools such as VRML and X3D to display and navigate web sites using 3D. All interactive 3D content are embedded into HTML web pages through this extension.

VRML (Virtual Reality Modeling Language) is a standard file format for interactive 3D vector graphic designed particularly for web pages supporting 3D geometry, animation, and scripting. The text file format defines vertices and edges for a 3D polygon with specific surface color, UV-mapped textures, shininess, transparency and so on. The URL can be associated with a graphic element and the web browser can retrieve a new VRML web page or file from the Internet when a user clicks on the specific graphic element. Animations, sounds, lighting and other aspects of the virtual word can interact with a user or may be triggered by external events such as timer. In 1997, VRML was ISO certified and continued to attract a large community of artists and engineers which widely support 3D format for tools and viewers.

X3D (Extension3D) is a standard for representing 3D scene offering real-time communication of 3D data across applications. In 2001, X3D became the successor VRML using XML encoding enhanced application programming interfaces (APIs) to encode the scene. The X3D extension supports shading with light map, normal map, multi-texture rendering and also other cutting edge 3D features along with custom support for users. Use of X3D is growing with content and applications across multiple platforms which provide portability, interoperability and durability to interactive 3D information. It is however just a scene representation for user and it does not provide end user tools to render the corresponding scenes.

Panoramic imagery is another form of media that is widely used on the web for virtual tour because it displays a wide angle of view using a special combination of panoramic shots. Web3D supports panoramic image rendering as well through Background nodes called Universal Media Panoramas. Another extension is QuickTime VR (QTVR) which provides the creation and rendering of photographically-captured panoramas plus the exploration of objects through images taken from multiple viewing angles. It is an option for museums; it allows panning and zooming in high quality and also adds hotspots that connect QTVR and panoramas to other files. It functions as a plugin for the standalone QuickTime Player and works as a plug-in for the QuickTime Web browser plugin. Watching a 360 degree of virtual tour, navigation within a single panorama is provided by two main actions: dragging and zooming. We interact with the panorama using mouse, rotating in all directions from floor to ceiling, in some cases to go from one room to another, to zoom in the images, to read information, to watch videos, etc.

Implementing Web3D extensions in conjunction with HTML makes more dimensional of VM development on the web. 3D visualization and, in particular, a panoramic view enhancement enable users better access and interaction with information. When combined with the use of hypermedia and hyperlinks, it makes online VM visits more attractive. The following examples demonstrate the development of interactive Web3D VM.

[4] http://www.web3d.org/

Figure 2.5: The NMNH Virtual Tour at beginning with navigation tools and icons (Retrieved from https://naturalhistory.si.edu/VT3/NMNH/z_NMNH-016.html)

The Smithsonian Institution claims to be the world's largest museum and research complex. A panoramic virtual tour of the Smithsonian National Museum of Natural History (NMNH)[5] allows visitors to navigate into the whole museum, walking from room to room, checking out fossils of countless dinosaurs, specimens of early sea life, exhibits on the ice age, and much more.

THE VALENTINO GARAVANI VIRTUAL MUSEUM[6]: The legendary fashion designer, Valentino Garavani, has defined a unique world of high couture. His achievement takes radical new form, in keeping with the creative traditions of the house, over 5000 documents have been installed in a spectacular 3D Palazzo, which is visited through web browser directly via a Unity Web Player. The application may be downloaded for an optimized experience, and create unique route through the galleries, to discover and enjoy every aspect of Valentino's extraordinary world.

2.2.4 Mobile application VM

Recently, mobile devices such as smartphones and tablets are widely used with touchscreen that can be used by touching it with a finger or a stylus pen. These devices enabled users to directly surf the Internet through its mobile browser with a relatively fast and simple modality. Then mobile devices became a common device in everyday life, replacing a desktop computer. Interactive VM on web sites are also deployed on these devices instead of using mouse and keyboard. Mobile devices use the gyroscope, the accelerometer, the magnetometer and the GPS to determine the position and orientation of the device. These sensors can be applied with the touchscreen to run applications on device. Interactive virtual tours can be viewed on mobile devices, allowing visitors to enjoy virtual tours anywhere at any time. The mobile can also be used in physical museum to create augmented reality experience. Then the museum is extended with context and virtual content on the mobile device. This following example is running a VM online on a mobile device. User can interact with the content through the touchscreen.

[5] https://naturalhistory.si.edu/
[6] http://www.valentinogaravanimuseum.com/

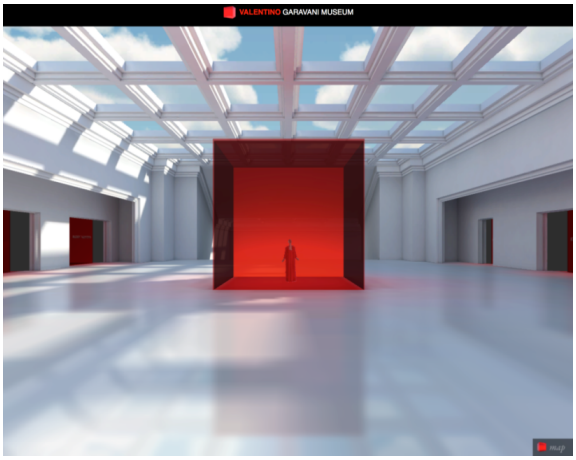

Figure 2.6: The Virtual Museum of Valentino Garavani run on Unity Web Player (Retrieved from http://www.valentinogaravanimuseum.com/online-museum)

Figure 2.7: The appearance of application on mobile of AMNH, personal tour guide to discover the museum (Retrieved from https://www.macworld.com/article/1153037/amnhexplorer.html)

New York's American Museum of Natural History (AMNH)[7] provides mobile application to explore for photos and information on exhibits at AMNH as well as a detailed map of a current location of visitor in the museum.

This example of VM tour on mobile device applied all sensors to enable user to interact with panoramic image by tilting the device and touching the screen. Moreover, the accelerometer, the magnetometer and other mobile device sensors allow the application to get user position (Seco and Jiménez, 2018) which can be applied for contextual navigation inside the museum.

Here, we observe a change of interaction from desktop devices by point and click to mobile device with touchscreen and tilting to interact with the content. Many applications try to link the features of touchscreen and gyroscope to create new interaction. More gadgets and mobile applications were developed to support interaction and immerse user to the next level of learning in VM. Google Cardboard[8] and Google Arts & Culture[9] are an example of gadget and application which support user experience in VR.

[7]https://www.amnh.org/apps/explorer
[8]https://vr.google.com/cardboard/
[9]https://play.google.com/store/apps/details?id=com.google.android.apps.cultural&hl=en

Figure 2.8: The Google Cardboard VR headset and the "Paris VR" Android app specifically conceived for it (Retrieved from https://play.google.com/store/apps/details?id=com.cardboard360images.parisvr&hl=en)

Google Cardboard is a VR tool to make a head mounted smartphone. This gadget is intended as a low-cost system to encourage interest and development in VR applications. Simple cardboard glasses with two lenses in combination with any mid-range smartphone providing limited quality are accessible for almost everyone. Users can run Cardboard-compatible applications on their phone, place the phone into the back of the viewer, and view content through the lenses. One of the first applications released for such device features a virtual tour of the Louvre Museum in Paris[10].

Google Arts & Culture is an online platform (Yáñez et al., 2015) through which the public can access high-resolution images of artworks housed in the initiative's partner museums, from over 2000 leading museums and archives who have partnered with the Google Cultural Institute[11] to bring the "world's treasures" online. The platform enables users to virtual tour in partner museums' galleries, exploring physical and contextual information about artworks, and compile their own virtual collection. The "walk-through" feature of the project uses Google's Street View[12] technology. The images of many of the artworks were reproduced in a very high quality. Additionally, improved version of the application with new features enhanced search capabilities and a series of educational tools.

There is an explore option that uses Google Street View technology to let user explore a museum gallery (Csapó et al., 2015) with the same way using the Street View in Google Maps[13]. The trolley specially designed Street View took 360-degree images of the interiors of 385 selected galleries within the museums. Its design makes it easier to use on mobile devices, while also adding new features such as support for VR via the Google Cardboard.

2.2.5 Mobile application AR

Augmented Reality (AR) technology overlays a computer image created into a real-world user perspective with virtual information. A simple AR system consists of a processor, display, sensors and input devices such as camera. The camera captures the image, then the virtual object is added to the top of the image and displayed. The fundamental difference compared to other imaging tools is that in AR a virtual object is moved and rotated in 3D coordinates rather than using 2D imagery coordinates (Siltanen, 2012). The most important requirements regarding the performance of an AR system are related to the effectiveness of visualization and interaction techniques (Teyseyre and Campo, 2009). Nowadays, mobile devices such as smartphones and tablet have these components, which typically include a camera and MEMS (Microelectromechanical systems) sensors such as GPS accelerometers

[10]https://www.youvisit.com/tour/louvremuseum
[11]https://en.wikipedia.org/wiki/Google_Cultural_Institute
[12]https://en.wikipedia.org/wiki/Google_Street_View
[13]https://en.wikipedia.org/wiki/Google_Maps

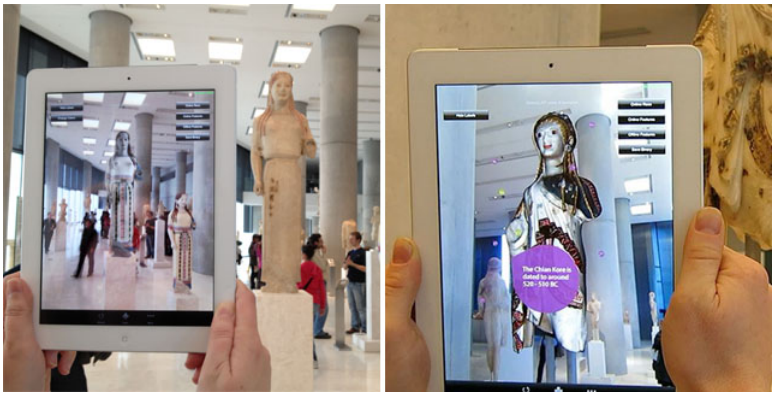

Figure 2.9: The Acropolis Museum applied mobile devices bring augmented colors and augmented stories to art pieces through AR (Retrieved from https://www.youtube.com/watch?v=DUPLPwOVE-M)

and solid state compasses making mobile devices an ideal AR platform. Combining powerful sensors, mobile device became an irreplaceable part of everyday life. The popularity and capabilities of mobile devices coupled with widespread and affordable Internet access and advances in the areas of cooperative networking, computer vision and mobile cloud transformed these devices into AR environment, also usable for museums.

Many computer vision methods of AR are inherited from camera position and orientation. Usually those methods consist of two stages. The first stage is to detect interest points, markers or optical flow in the camera images. This step can use feature detection methods such as corner, edge or thresholding and other image processing methods. Then it restores a real world coordinate system from the data obtained in the first stage. Some methods assume objects with known geometry or markers are present in the scene.

There are two types of AR mobile applications: marker-based and location-based. Marker-based AR mobile applications are based on image recognition, using a camera of a portable device to detect certain patterns or markers, such as QR codes or images. Once a pattern is recognized, the app overlays digital information on this marker. The orientation of the AR object depends on the position of the marker. Location-based AR applications use GPS and other position detectors to establish location and to create augmented reality objects. These applications can show directions where the interest point is located or mapped virtual object to the place or physical objects enable user to get more information.

For location-based, Simultaneous Localization and Mapping (SLAM) allows applications to map an environment and track their own movements in it. For example, an AR mobile app can remember the position of different things in a room and, thus, keep a virtual object in a certain place while a user moves around the room. Also, this technology can go far beyond adding AR objects to a room. Thanks to SLAM, it's possible to create maps for indoor navigation. Note that GPS doesn't work indoors, but SLAM does and this technology has enormous potential, even if SLAM is still reduced to small area.

New interaction techniques have been developed offering a greater degree of freedom compared with traditional windows style interfaces (Billinghurst et al., 2001; Zhou et al., 2008). On the other hand, AR has an interface technology that aims to take advantage of a way to combine information generated from the device with the real world. The success of an AR exhibition is highly related to the level of realism achieved (Liarokapis and White, 2005). Especially, AR for museum exhibition, visitors expect the visualized information to be naturally presented in entertaining manner. Following example is AR mobile application for museum offering visitors device to explore exhibits with AR technology. The AR applica-

tion uses a back camera to visualize, integrate and add virtual objects or data to provide useful information to users. This helps finding information in the museum easier and more interesting.

An example in Figure 2.9 shows how visitors interact with the content using AR application by their handheld devices. Augmented Reality Stories at Acropolis Museum (Keil et al., 2013) shows how mobile devices bring augmented colors and augmented stories to art pieces at the Acropolis Museum through AR developed by CHESS (Cultural Heritage Experiences through Socio-personal interactions and Storytelling).

2.2.6 Interactive on-site installation

Museum visitors have come to expect a high level of interactivity in museum exhibitions. Interactive on-site installation is a new way of using emerging digital technologies to enhance the visitor experience. On-site installation is a full-scale interaction of VMs, with the most interactive devices in this format. VM has various forms depending on the application, scenario and end-user. It can be a 3D reconstruction in the physical museum where cultural heritage site is an entirely imaginary environment, in the form of various rooms where cultural artefacts were placed. On-site virtual heritage exhibition is one of the computer-based interactive technologies where the user immerses and understands the content (White et al., 2007). The development of new interactive technologies affects all aspects of teaching. It is obvious in the case of new interactive technologies for the public. The user-oriented design learning process and its design implications on how digital and tangible interactions can be used for cultural learning in museum exhibits. Interactive exhibition will become a link between the user and the cultural heritage where users learn about the culture by interacting with the virtual environment (VE).

A projection mapping also known as video mapping or spatial AR is used to transform an object into a display surface for projection. These objects may be complex landscapes, such as building a small indoor object or theatrical stage. By using special software, two or three dimensional objects are mapped to areas in virtual applications that mimic the real environment. The software can interact with the projector to fit the desired image onto the surface of the object. The video is often integrated with audio stream to create more complete immersion. Many museum exhibitions use this method to present content in a place where some devices can be combined.

Nantes 1900 (Billen et al., 2012) is an exhibition for highlighting the model of the port of Nantes (Figure 2.10). The Nantes History Museum decided to improve the layout of the port model, using interactive, educational and technological tools. The goal is to promote this heritage object through digital technologies. The museum provides visitors to the museum with multi-touch screens located in front of the model, allowing, through a specific interface, to navigate in different ways. The system also provides through video projectors, a bright return on the model depending on the actions of the user.

The River of Grass exhibit[14] was commissioned for the Frost Museum of Science in Miami, Florida along with two Shark Sense 4D VR exhibits and the Immersive Gulf Stream Experience (Figure 2.11). The highly immersive experience teaches children about the Everglades' unique ecosystem. The interactive VR exhibit room is a large, floor-to-ceiling room with technology to allow kids and their parents to experience the full interactive experience in the Everglades. The VR exhibit provides a host of playful interactions within the VE to stimulate understanding of Florida's precious wetland ecosystem, using up to 16 projectors and 7 network motion detectors to control motion pictures in a custom 3D application.

A Cave Automatic Virtual Environment (CAVE) is an immersive VE where the virtual room renders 3D projection from three up to six walls. All users entering a CAVE will

[14]https://www.frostscience.org/exhibition/river-of-grass/

Figure 2.10: The enhancement device for the mock-up of Nantes in 1900 using interactive and technical tools which are multi-touch screens and 3D mapping (Retrieved from http://www.club-innovation-culture.fr/chateau-de-nantes-maquette-nantes-1900-ecrans-tactiles-crowdsourcing/)

Figure 2.11: The interactive River of Grass exhibit with VR technology and interaction to give children and their parents the experience of the Everglades (Retrieved from https://www.youtube.com/watch?v=m9tK4W0xevs)

Figure 2.12: The Dassault Systèmes CAVE with full-room spaces bring virtual worlds of D-Day history to users via projectors and 3D glasses (Retrieved from http://blogs.3ds.com/perspectives/dday-innovation-day/)

Figure 2.13: The VR Museum of Fine Art, available for HTC Vive, the HMD device with controller for interaction in VR experience (Retrieved from https://www.youtube.com/watch?v=lUEaLdnJhNs)

wear lightweight stereo glasses that allow them to see both virtual and physical worlds. Some systems may interact with a portable device called a wand. The CAVE system offers unlimited opportunities for exploring virtual worlds. The system also provides the ability to provide real-time experience to large groups such as guided tour, group teleconferencing, and interactive simulation (Roussou, 2001).

The D-Day program[15] is a VR center on Ouistreham Beach with interactive 3D CAVE experience. Dassault Systèmes[16] built a bridge between yesterday's engineers and today by preserving memories of outstanding technological innovation. For the 70th anniversary of the invasion of Normandy (Figure 2.12), well known as D-Day, their passion for innovation foundation has built a partnership with landing engineers in Normandy.

Head Mounted Displays (HMDs) is a display device worn on the head or as part of a helmet. The HMD generally has one or two small displays with transparent lenses and glass embedded in the device. A VR system with HMD provides real-time head-tracked perspective with a large angle of view, interactive control, and stereoscopic display. User moves in the 3D space and use motion-tracked handheld controllers to interact with the environment. The wireless controllers are the hands of VR, making a more immersive experience for the user. The controller has multiple input methods included a track pad, grip buttons, and a dual-stage trigger which has infrared sensors that detect the base stations to determine the location of the controller.

The VR Museum of Fine Art[17] is an example application that applied HMD device and controller to get into new way of museum experience. To explore a VM at room-scale, seeing famous sculptures in fully 1:1 scale and also famous paintings without the limitations of glass and security guards. This VR experience get user hands all over the priceless artifacts. The initial release of The VR Museum includes 15 high-fidelity sculptures with famous paintings which are scanned and rendered in crisp detail.

Sketchfab[18] is a platform to publish and share 3D model, VR and AR content providing a viewer based on the WebGL and WebVR technologies that allows displaying 3D models

[15]https://www.3ds.com/dday/
[16]https://en.wikipedia.org/wiki/Dassault_Syst%C3%A8mes
[17]https://store.steampowered.com/app/515020/The_VR_Museum_of_Fine_Art/
[18]https://sketchfab.com/

Figure 2.14: The Van Gogh Room: the completely 3D replica of a Van Gogh painting is freely accessible with HTC Vive and Sketchfab VR for an impressive VR experience (Retrieved from https://sketchfab.com/models/311d052a9f034ba8bce55a1a8296b6f9)

on the web, to be viewed on any mobile browser, desktop browser or HMD device. Users can get into the platform and upload 3D models, support over 30 file formats, or view other users' content directly in the browser with a 3D viewer. More than 750,000 3D models are already online that will be a digital library for 3D content.

Here we can see that applications development is based on devices and new devices will come in the future. Furthermore, there is an aspect of learning ability in the VM alongside with interaction issue that should be investigated to find the relation between each other. VMs as On-site Installation are the VM style that we would like to investigate to better manage interaction system with the three following research questions:

- What are the needs of interaction and devices potential? There are so many opportunities when a museum invest in very expensive devices, it will expect to be able to apply all.
- How different interactions affect learning and what are the factors? We expect to help museum finding the best interaction solution for thier specific application.
- How to better immerse museum visitors into VE thanks to interaction? We expect to use new cutting edge technologies to make visitor active rather than an observer.

2.3 VM development process

Based on usage patterns, as mentioned above, VMs are divided into 5 types: digital media VM, online VM, interactive Web3D VM, mobile application VM and interactive on-site installation VM. Each type has different patterns of development and interaction with different users depending on the technology and devices used to access the data in the VM. For each development, each VM will be built on a different platform depending on its usage and content. Considering the platform, there are details on how to produce the work, including platform development, as described in the next subsections.

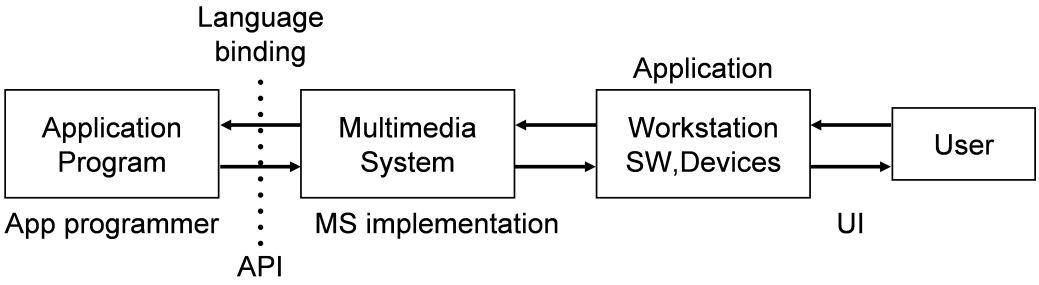

Figure 2.15: The architecture of multimedia-authoring tool, an application development is designed in MS concept where authoring tools are supported, users often use an application on their desktop devices and interact through the UI.

2.3.1 Development of digital media VM

The multimedia-authoring tool is the main platform to develop digital media content on PC with desktop device (Kaskalis et al., 2007). Multimedia authoring tools provide the framework to organize and edit the elements of a multimedia project, including an integrated environment for combining the content and functions of a project. It enables the developer to combine text, graphics, audio, video, and animation into an interactive presentation. The applications are an offline software installed on personal computer or delivered by CD-ROMs.

Authoring systems include editing tools to create, edit, and convert multimedia elements such as animation and video clips. There are two main components of this tool: authoring part and programming part. Authoring involves the assembly and integration of multimedia with possibly high level graphical interface design and some high level scripting, while programming involves low level assembly, construction and control of multimedia. It involves languages like C and Java. Features of authoring tools edit and organize content, performance tuning and playback, interactivity and programming. Visual overview facility illustrates project structure at a macro level to organize, design, and produce multimedia. It involves storyboarding and flowcharting. The authoring interface is referred as the authoring metaphor. It basically means the way applications are structured or the methodology used by the system to accomplish its task (Ayub et al., 2005).

In multimedia-authoring tool, there are two major metaphors: the timeline metaphor and the icon metaphor. Timeline metaphor works with the time diagram and places objects and events over time scale in the correct relationship. This is especially useful when dealing with a variety of activities. Some of the authoring tools that use this metaphor are Adobe Director[19] (previously Macromedia Director), Toon Boom[20] and Moho[21].

With the icon metaphor the application structure is built by dragging icons from an icon palette into the work-place to form a flow-line of icons. These icons represent objects and events that will occur according to the sequence of the arrangement of the icons. Authors can now concentrate on the application logic and content rather than on the programming details. Developing sophisticated application is still a difficult and complex task. For example Adobe Authorware[22] (previously Macromedia Authorware) and Adobe Captivate[23] are tools based on this metaphor.

[19]https://www.adobe.com/products/director.html
[20]https://www.toonboom.com/
[21]https://my.smithmicro.com/anime-studio-debut.html
[22]https://www.adobe.com/products/authorware
[23]https://www.adobe.com/products/captivate.html#item-1-11

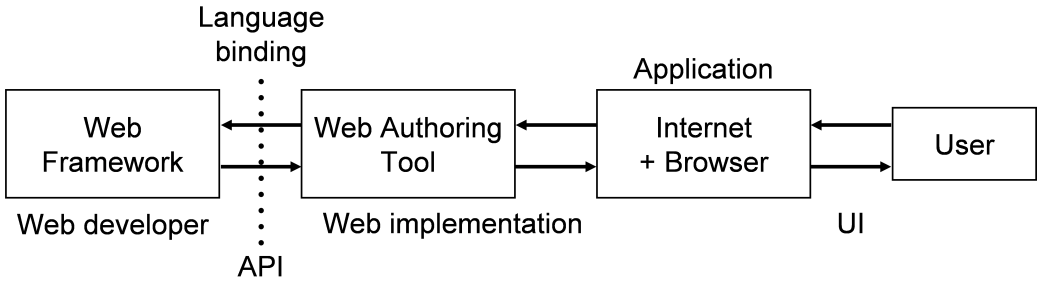

Figure 2.16: The architecture of web authoring tool, a development of web application is designed on web authoring tools which support 3D graphic, users use a web browser to access interactive content.

2.3.2 Development of online VM

The platform used to develop the online VM is still the same multimedia-authoring tool, but it requires the ability to export work in a web-based format, with many of this software capabilities. Some multimedia-authoring tool intended to create a particular web called web page authoring tool. Exporting multimedia online VM into web sites requires converting the multimedia format into HTML format. By the past, because the speed of data transmission on Internet was not as fast as it is today, most data were still in a simple graphical form. The main purpose of these platforms for online development of VMs is to create hypermedia and hyperlinks to present information. However, the main device used to connect to Internet to visit the site remains a PC desktop with a mouse and keyboard. Content design is not complex and uses point and click interaction or even drag and drop. Since the development of the online VM on the website requires HTML to interpret the data on web browser. Content designers may not have the knowledge of HTML language, so the design of the platform comes with tools to help developer in designing the website without the need of HTML skill. Adobe Dreamweaver and Microsoft Front page are examples of platforms allowing developer to create web pages without learning the underlying HTML.

2.3.3 Development of interactive Web3D VM

3D graphics are widely used and this use is increasing on web sites. Most authoring tools offer 3D modeling, 3D animation and 3D rendering. The web page authoring tool is still the main platform used to develop interactive Web3D VMs. However, content is changing from hypermedia and hyperlinks to interactive 3D environments. Web authoring tools add the ability to insert interactive 3D environments including 3D computer graphics, as well as panoramic images where users interact with content. Here, JavaScript is applied to make content smarter, increase the interest of the content, and increase interaction with users. With the JavaScript library X3DOM, X3D scenes become part of the HTML Document Object Model (DOM). Standard DOM manipulation and event handling are used to represent and interact with 3D scenes and all the objects therein.

WebGL (Web Graphics Library) is a cross-platform API that brings OpenGL to the web as a 3D drawing context within HTML. It is a low-level DOM interface that uses shading languages and can be cleanly combined with other web content that is layered on top or underneath the 3D content (Parisi, 2012). It is suited for dynamic 3D web applications in JavaScript programming language and integrated into modern web browsers. X3DOM uses WebGL to do its rendering directly in the browser, without any plugins.

WebVR is an experimental JavaScript API that supports virtual devices such as HTC Vive, Oculus Rift, Google Cardboard, or OSVR in a web browser. The WebVR API provides

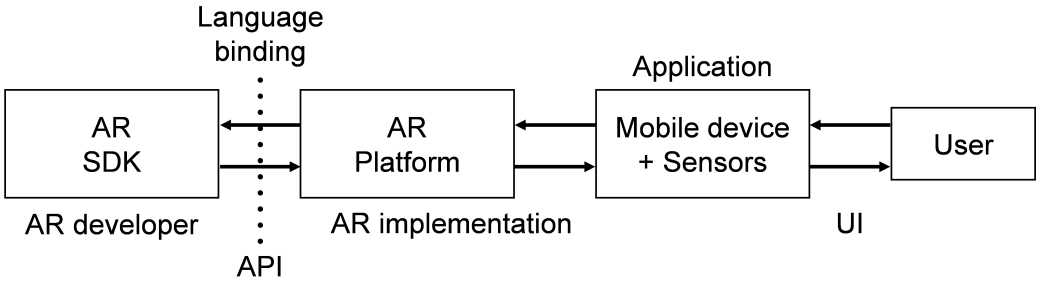

Figure 2.17: The architecture of AR platform to develop AR application on mobile, a development of AR application is designed on AR platform, mobile devices are used to display the content and allow user interaction through the sensors.

new interfaces that allow web applications to render VR content using WebGL with the necessary camera settings and device interactions (Neelakantam and Pant, 2017). The API is designed to use some paths similar to other intrusive Web APIs such as the Geolocation API.

2.3.4 Development of mobile application VM

With mobile VM application, we focus on AR application development on mobile. AR application development is different from mobile applications, it needs an appropriate AR software development kit (SDK) to support content creation under platform to work with. Augmented Reality Markup Language (ARML) is a data standard developed within the Open Spaces Geospatial Consortium (OGC), which contains XML syntax to describe the position and appearance of virtual objects in the scene, as well as binding script to provide dynamic access to virtual objects properties. Some software development kits (SDKs) emerged to quickly develop AR applications. AR SDKs are offered by Vuforia, ARToolKit, Catchoom, CraftAR, Mobinett AR, Wikitude, Blippar, Layar, Meta, and ARLab. One can also develop AR application with direct access to OpenGL without using these AR SDKs. However, creating 3D objects and VEs to be added to the real world still requires a platform to facilitate the handling of these virtual objects. A game engine is an ideal platform for creating AR applications, as it allows AR SDKs and also support VEs designing.

The game engines are platforms that extend features of the AR application, as it supports the creation of AR content. The details of the game engine will be discussed about the interactive on-site installation. The following examples are the game engine with SDKs usable for an AR application development.

Unity3D[24] is one of the most advanced game engines used to create games for computers and consoles, but also capable of powering AR applications that can be exported to both iOS and Android. The top AR SDKs widely used compatible with Unity3D are Vuforia, Wikitude, ARToolKit, AppleARKit, etc.

Unreal Engine 4 (UE4)[25] is another game engine enabling AR application development with support for Apple's ARKit and Google's ARCore augmented reality frameworks. The workflow for using ARKit has been streamlined and improvements made to the fidelity and performance of projects developed using Apple's tech with the potential of high-quality AR experiences. UE4 also supports Google's developer ARCore and enables AR development across the Android ecosystem, giving developers the ability to make compelling AR experiences without the need for any additional hardware.

[24]https://unity3d.com
[25]https://www.unrealengine.com/en-US/what-is-unreal-engine-4

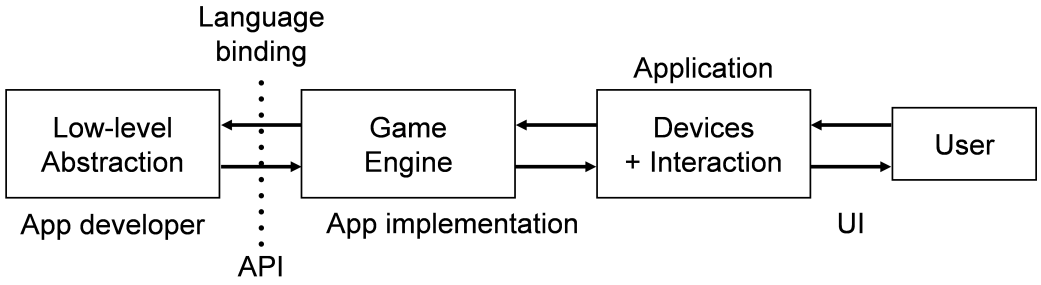

Figure 2.18: The architecture of platform to develop on-site installation VM, the application is developed with a low level abstraction on the platform, various devices are connected to allow user interaction.

2.3.5 Development of interactive on-site installation VM

An interactive on-site installation is a system where user can directly interact with. Most are real-time interactive agent-based computer simulations applied VR technology. With this feature, the game engine is a platform that is suitable for use as a tool for application development. A game engine is a collection of extensible software which can be used as the foundation for many different games without major modification. Game engine combines project manager, text-editor, compiler and debugger into a single tool. Facilitates rapid-prototyping, extensible integrated development environments (IDEs) can be configured with plug-ins for automating many other development tasks.

The development of virtual exhibition in case of interactive on-site installation consists of two main parts. The first part is the development of the content or VE of the media, which is processed in real-time. The second part is the part of devices connection that allows the user to use those devices to interact with the media in the first part. Considering VR technology, devices are constantly evolving to support technology changing. To further enhance the user experience, interaction with immersive VR is increasingly applied to virtual exhibitions. The platform used to develop the virtual exhibition needs to connect a variety of devices which can provide access to the fully use of these devices.

Although the purpose of the game engine is to develop games, its features support VE design and device connectivity, it makes the game engine an ideal platform for developing interactive content. The following examples are the game engine that supports the creation of VR content with user interaction.

As already mentioned above, Unity3D is an intuitive game development engine that offers features for both 2D and 3D development with real-time rendering engine to produce realistic scenes. The Unity Editor provides multiple tools that enable rapid editing and iteration in development cycles, including play mode to preview the result of work in real-time. Moreover, it is a creative hub for artists, designers, developers including scene design tools, storytelling and cinematics, lighting, audio system, and animation system.

Unity VR allows the connection of some VR devices directly from Unity, without any external plug-ins. It provides a base API and features set with compatibility for multiple devices. Enabling VR support mode can enable VR devices at runtime. The first device that initializes properly is the one enabled and this feature is very useful to develop VR application on direct device testing.

UE4 is the next version of UDK (Unreal Development Kit) released by Epic Games[26]. This game development kit comes with huge number of options for mobile, PC as well as

[26]https://www.epicgames.com/site/en-US/home?lang=en-US

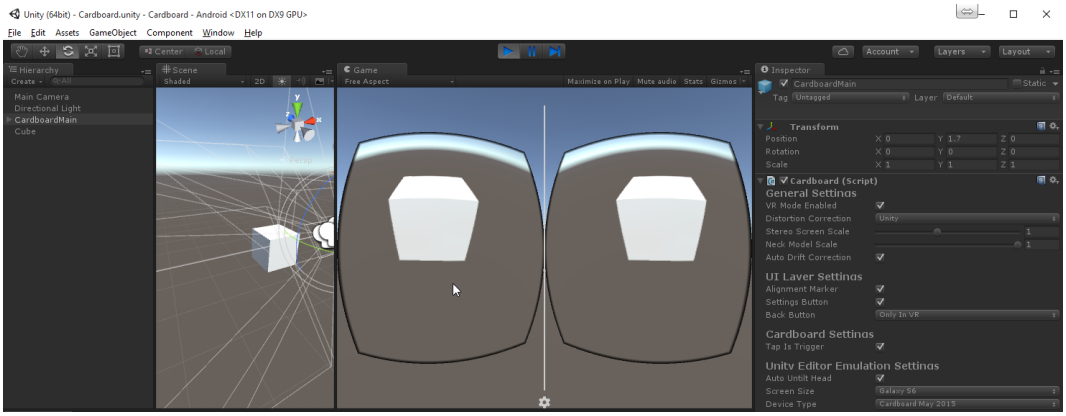

Figure 2.19: The appearance of Unity game engine in VR play mode (Retrieved from http://mammothinteractive.com/activating-vr-split-mode-unity-tutorial/)

console game development. UE4 encompasses stunning graphical capabilities like advanced dynamic lighting and a new particle system that can handle up to a million particles in a scene simultaneously.

UE4 is designed for realistic visualization, it meets these requirements and provides a solid foundation to build content on all VR platforms. Advanced Optimization designed for high performance, CPU/GPU profiling tools and flexible renderer equips developers to efficiently achieve quality VR experiences.

VR requires complex scenes rendered at very high frame rates, Unreal Engine developed a rendering solution specific to VR. The Forward Renderer supports high-quality lighting features, Multisample Anti-Aliasing (MSAA) and instanced stereo rendering to produce crisp.

Moreover, with the power of Unreal Engine to build VE in VR, developer can reach out, grab and manipulate objects on their scene. The full Unreal editor runs in VR with advanced motion controls so that you can build in a "what-you-see-is-what-you-get" environment. It's the most robust, complete and capable feature VR for development solution.

Blender[27] is the free and open source 3D creation suite. It entirely supports 3D pipeline such as modeling, rigging, animating, simulating, rendering, compositing and motion tracking, even video editing and game creating. The Blender Game Engine (BGE) is a component of Blender used for making real-time interactive content. The game engine was written from scratch in C++ as a mostly independent component, and includes support for features such as Python scripting and OpenAL 3D sound.

BlenderVR is an adaptation of the BGE to support VR devices which are CAVE, VideoWall, Head-Mounted Display (HMD) and external rendering modality engines. BlenderVR allows the BGE on any VR architecture and supports adaptive stereoscopy and communication protocols such as VR Private Network (VRPN) and Open Sound Control (OSC) with minimal effort. The goal of the BlenderVR project is to offer a cross-platform solution for VR applications which benefits from the Blender environment and associated community for creating quality interactive scenes. Porting of scenes from one VR platform configuration to another should be transparent and require no editing of the actual scene.

[27]https://www.blender.org/

Figure 2.20: The appearance of Unreal Engine in VR runtime mode to build VR scene through HMD device and controller (Retrieved from http://unreal1249.rssing.com/chan-60225994/all_p1.html)

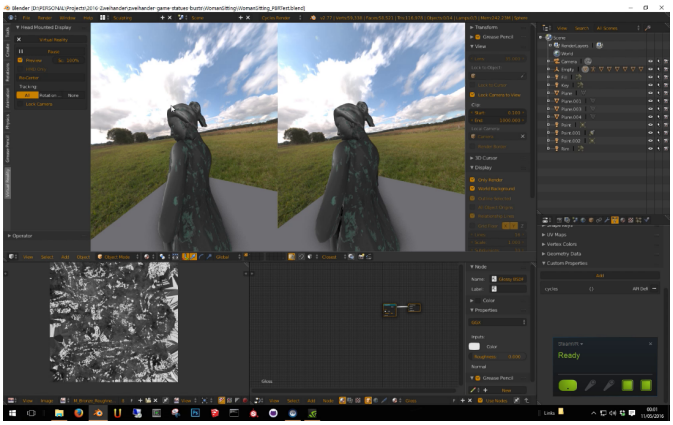

Figure 2.21: The appearance of Blender game engine in VR runtime mode (Retrieved from https://www.youtube.com/watch?v=UK4RmnCOOv8)

2.4 Devices and interactions

VM application development, especially in the form of interactive on-site installation, is connected to devices and application development across platforms, accessing them through the device SDK. Therefore, device selection is important for application development because the type of selected device is directly affected to the design of the interaction with the user.

The advancement of VR technology enables a variety of devices to be applied for the application so that users feel engaged with the interaction system. Understanding the usability and capabilities of the device will make application design easier and more consistent with the features of the device. Considering the features of the device used in the interactive on-site installation VM, devices are divided into two major types: output devices and input devices, the details are as follows.

2.4.1 Output devices

An output device refers any device used to send data from a computer to render information to one or more users through the human perception system. In this topic, we will focus on the output devices that can be applied to the on-site installation VMs that are used to deliver information and focus on user interaction. These hardware devices are usually focused on stimulating the visual, audition, or haptic senses as described in the next sections.

2.4.1.1 Visual displays

Display devices are electromechanical systems capable of full motion graphic displays to present the information produced by the system through the human visual system. The cathode ray tube (CRT) was used for several decades until being replaced by plasma, liquid crystal (LCD), solid-state devices such as LEDs and OLEDs and even laser sources now. With the advent of microprocessors and microelectronic devices, many individual picture elements called pixel, could be incorporated into one display device.

The performance of these devices depends on the following parameters: field of view, spatial resolution, screen geometry, light transfer mechanism, refresh rate and comfort of use. Another way to characterize these devices is to follow to the different categories of depth perception cues used to achieve that the user can understand the three-dimensional information (Wits et al., 2011). The following examples are visual displays used to present information: monitors, surround-screen displays, hemispherical displays, head-mounted displays (HMDs), arm-mounted displays, stereoscopic displays and CAVE.

2.4.1.2 Audio

These devices present sound information through the human auditory system, its objective is to generate and display a spatialized 3D sound allowing user to determine the location and direction of the sound.

OpenAL is a cross-platform 3D audio API applied for VR applications. The library models a collection of audio sources moving in a 3D space that are heard by a single listener somewhere in that space (Dong, 2016). The basic OpenAL objects are a Listener, a Source, and a Buffer. There can be a large number of Buffers, which contain audio data. Each buffer can be attached to one or more Sources, which represent points in 3D space which are emitting audio. There is always one Listener object with audio context, which represents the position where the sources are heard and rendering is done from the perspective of the Listener. There are different localizations cues: binaural cues, spectral and dynamic cues, head-related transfer functions, reverberation, sound intensity and vision and environment familiarity (Zotkin et al., 2004).

2.4.1.3 Haptic feedback

Haptic devices simulate physical interaction between users and virtual objects. The human haptic system has two fundamental kinds of cues, tactile and kinesthetic (Hergenhan et al., 2015). Tactile cues include textures, vibrations, and bumps, while kinesthetic cues include weight, impact, etc. In VR where users can interact with VE in real time, haptic feedback enables state changing to increase realism. Such interactions are sometimes carried out with the help of haptic interfaces, allowing participants to exchange tactile and kinesthetic information with the VE. There are three different types of 3D haptic displays: providing a sense of force, simulating the sense of touch and using both (Luciano et al., 2005). The main features that distinguish these devices are: haptic presentation capability, resolution and ergonomics (Sharma et al., 2011).

2.4.2 Input devices

An input device is any device used to capture and interpret the actions performed by the user. In this section, we will focus on the interaction using input devices that can be applied to the on-site installation VMs. These hardware devices are either audio and video input devices, desktop input devices or tracking devices as described in the next paragraphs.

2.4.2.1 Audio input devices

Audio input devices are used to capture sound which allows a user to send audio signals to a computer for processing, recording, or carrying out commands. Devices such as microphones allow users to speak to the computer in order to record a voice message used with speech recognition software.

2.4.2.2 Video input devices

Video input devices are used to digitize images or video from the real world into the computer. The information can be stored in a multitude of formats depending on the application requirement. Pattern recognition is a processing method to convert images or videos into various commands applied to application. Devices such as webcam, depth camera are examples of video input devices.

2.4.2.3 Desktop input devices

These input devices are designed for an interaction on a desktop PC with 2D monitor. Many of them have an initial design thought in a traditional interaction in 2D. However, desktop input devices work perfectly in 3D with an appropriate mapping between the system and the device. There are different types of device such as keyboard, mouse, touch pad, touch screen and game controller.

The keyboard is used to enter text information into the computer. The keyboard can also be used to type commands directing the computer to perform certain actions.

The mouse is used for pointing, movement of mouse will transmit the information to the computer. A mouse also includes buttons and possibly a scroll wheel to allow users to interact with the graphic user interface (GUI) such as cursor to be a pointer on screen.

The touch pad is used on many laptop computers today as an alternative to the mouse. Sliding finger along the surface of the touch pad moves cursor on the screen. The buttons are located below the pad, but most touch pads allow you to perform mouse clicks by tapping on the pad itself.

The touch screen is an input device normally layered on the top of a visual display. The touchscreen enables the user to interact directly with displayed objects. A user gives input or controls the information processing system through simple or multi-touch gestures by touching the screen with one or more fingers (Walker, 2012). Users can use the touch screen to respond to what is displayed if the software allows controlling the display method. For example, zoom to increase the font size.

The game controller is a device used with games or entertainment systems typically to control an object or character in the game. A controller is usually connected to PC. Special purpose devices, such as joystick, steering wheel and gamepad are also game controllers. Game controller can have a number of action buttons combined with one or more omnidirectional control sticks or buttons.

2.4.2.4 Tracking devices

In VR technology, a tracking system usually track the user or parts of user coordinates. The degrees of freedom (DOF) are one of the main features of these systems applied to get position and orientation. A tracking device is used to observe body movement of user or some parts of body, supplying a timely ordered sequence of location data for further processing. Tracking technologies will obtain all the necessary information from the user through the analysis of their movements or gestures. User interaction is related to basic parameters such as the relative position of the user, the absolute position, angular velocity,

rotation data, orientation or height. The collection of these data is achieved through systems of space tracking and sensors in multiple forms, as well as the use of different techniques. The tracking techniques can be categorized by methods of position tracking: markers tracking, body tracking, inertial tracking and magnetic tracking.

1. Markers tracking

Infrared light sources serve as markers for optical tracking with tracking devices such as a camera of infrared (IR) range. In this method, a target made of markers form a known pattern. A camera (or multiple cameras) constantly seeks the markers and then use various algorithms to extract the position of the object from the markers. The algorithms need to contend with missing data in case one or more markers is outside the camera view and are thus temporarily obstructed. By synchronizing the time with the camera, it is easier to block out other IR lights in the tracking area and reflect the IR light back towards the source almost without scattering. Then 3D movements are captured. Such technology was implemented by OptiTrack[28], MotionAnalysis[29], Advance Realtime Tracking (ART)[30], etc.

HMD devices also use this tracking technique. Oculus Rift uses Constellation, an IR-LED array that is tracked by a camera while HTC Vive uses Valve's Lighthouse technology which is a laser-based system. A real-time head tracker provides feedback to the central processor, allowing for selection of appropriate head-related transfer functions at the estimated current position of the observer relative to the environment.

2. Video tracking

This method represents a set of computer vision algorithms and tracking devices such as a stereo camera and a depth camera. The captured data recovers the pose of an articulated body, which consists of joints and rigid parts using image-based observations. These data are possible to perform tracking which continuously searches and compares the image with the known 3D model if the geometric characteristics of the target are known. These systems track the user to render their position, in addition to perform tasks like gesture recognition to enable the user to interact with the application. Such technology was implemented by Leap Motion[31], Microsoft Kinect[32], RealSense[33], etc.

3. Motorized tracking

This tracking system is controlled by three direct DC motors that have sensors and encoders attached to them. The number of motors corresponds to the number of degrees of freedom. Force feedback can be associated and this tracking system is usually a haptic system. The encoders track the user's motion or position along the x, y and z coordinates and the motors track the forces applied on the user along the x, y and z axis. A gimbal is a device that permits a body freedom in any direction.

Degrees of freedom are the directions the user moves in. For a user to touch all sides of a virtual 3D object the haptics system needs 3 degrees of freedom. Another 3 degrees of freedom are needed if a user wants to rotate the object freely. The haptics system is able to analyze and sense the forces applied by the user and then deliver a sensation back in real time. For good perception quality an about 1000 Hz frame rate is expected. The haptics system will interpret force and motion information to determine how objects move when forces are applied and also determine the geometry of the object. Such technology was

[28] https://optitrack.com/motion-capture-virtual-reality/
[29] https://www.motionanalysis.com/
[30] https://ar-tracking.com/
[31] https://www.leapmotion.com/
[32] https://developer.microsoft.com/en-us/windows/kinect
[33] https://realsense.intel.com/

implemented by Phantom Premium[34], Touch3D[35], Virtouse6D[36], etc.

4. Inertial tracking

This tracking method is usually implemented within mobile devices. The GPS serves to provide a location of the mobile device with a threshold about 1 meter, while the inertial sensor allows good orientation (Zendjebil et al., 2008). Inertial tracking systems do not require external reference such as those based on movement, but use data from accelerometers and gyroscopes. Accelerometers measure linear acceleration while gyroscopes measure angular velocity. Since the derivative of position with respect to time is velocity and the derivative of velocity is acceleration, the output of the accelerometer could be integrated twice to find the position relative to some initial point. Angular velocity can be integrated as well to determine angular position relatively to the initial point. Accelerometers and gyroscopes allow tracking the orientation with high update rates and minimal latency. With a reference to a compass, the rotation integration parts are connected and accelerometers are sharp for rotation tracking. However, they usually provide bad quality for positioning because of cumulative errors in double integration. Another tracking device using this method is Nintendo Wii Remote[37].

5. Magnetic tracking

Magnetic tracking is based on measuring the intensity of inhomogeneous magnetic fields with electromagnetic sensors. A base station is associated to the system transmitter or a field generator generates an alternating or a static electromagnetic field. The magnetic fields are generated by three electromagnetic coils which are perpendicular to each other to cover all directions in the three dimensional space. These coils should be put in a small housing mounted on the moving target. Current, sequentially passing through the coils, turns them into electromagnets, which allows determining their position and orientation in space. They provide good relative movement but experience demonstrates that they are very poor tracking system for exact location without a complex calibration task. Magnetic tracking was implemented by Polhemus[38] and in Razor Hydra[39] by Sixense.

2.4.3 Interaction techniques

Combination of input and output devices makes an interaction system. Interaction techniques are the methods for user interaction with the VE to execute different tasks directly related to input . The selection of the right input devices and interaction methods is important for a successful VR system. Development of VM application should consider the necessary components for an effective learning, which includes input devices and interaction techniques for the purpose of navigation, selection and manipulation in the VE. These three key interaction modes were identified by Bowman (Bowman et al., 2001a).

2.4.3.1 Selection and manipulation

Interaction techniques for selection and manipulation in VEs must accomplish at least one of three basic tasks: object selection, object positioning, and object rotation. To accomplish these tasks usually the system provides to the user a 3D cursor often represented as a human hand whose movements correspond to the motion of the hand tracker. This virtual hand technique involves touching an object, then positioning and orienting this virtual hand within the VE. The following examples are others selection and manipulation techniques that are used to access the VE.

[34]https://www.3dsystems.com/haptics-devices/3d-systems-phantom-premium
[35]https://www.3dsystems.com/haptics-devices/touch
[36]https://www.haption.com/fr/products-fr/virtuose-6d-fr.html
[37]https://en.wikipedia.org/wiki/Wii_Remote
[38]https://polhemus.com/
[39]https://support.razer.com/console/razer-hydra/

Ray-casting technique (Bowman and Hodges, 1997) allows users to cast a light virtual ray in a 3D virtual world and afford the ability to interact with whatever was attached to that ray.

Spotlight and aperture technique (Forsberg et al., 1996) improved the ray-casting technique by replacing the virtual ray with the spotlight circle representation. Objects are selected in the cast of the light in the direction it is pointing at. The advantage of this technique is that it allows to select objects from a far distance with high precision.

Image-plane technique (Pierce et al., 1997) gives users the affordance for manipulation of a 3D object that has been projected onto a 2D image plane. This is the same using a mouse to interact with a 3D representation on a 2D screen.

Go-Go technique (Poupyrev et al., 1996) improves the simple virtual hand technique by allowing the user to modify the length of the virtual arm.

World in miniature (WIM) technique (Stoakley et al., 1995) provides an alternative approach to extend user's arm to interact with objects, but instead scales the entire world to bring it within the user's reach. Once in reach the user can directly manipulate any 3D virtual within reach.

Hand-centered Object Manipulation Extending Ray-casting (HOMER) technique (Bowman and Hodges, 1997) allows user to select an object in the VE by extending ray-casting technique, an object is attached to virtual hand when it is selected.

Scaled-World Grab technique (Mine et al., 1997) is similar to HOMER, but uses the image-plan technique for selection instead of ray-casting.

2.4.3.2 Navigation

Navigation is used by the user in large-scale 3D environments and presents different challenges as supporting spatial awareness, providing efficient and comfortable movement between distant locations, and making navigation lightweight and natural. Navigation techniques are divided into the travel motor, and the wayfinding (Bowman et al., 2004).

Travel is defined as the control of the user's viewpoint motion from one location to another in the 3D environment. Further, viewpoint orientation is usually handled in immersive VEs by head tracking, so only techniques for setting viewpoint position need to be considered. There are several travel techniques: physical movement, manual viewpoint manipulation, steering, target-based travel and route planning.

Wayfinding is defined as the cognitive navigation action in a physical space using the users' space knowledge in relation to objects, landmarks, road signs, etc. Wayfinding supports user during the VE travelling to facilitate the constraints from the virtual world. This concept is the same in virtual space, but enhanced with navigation overlays.

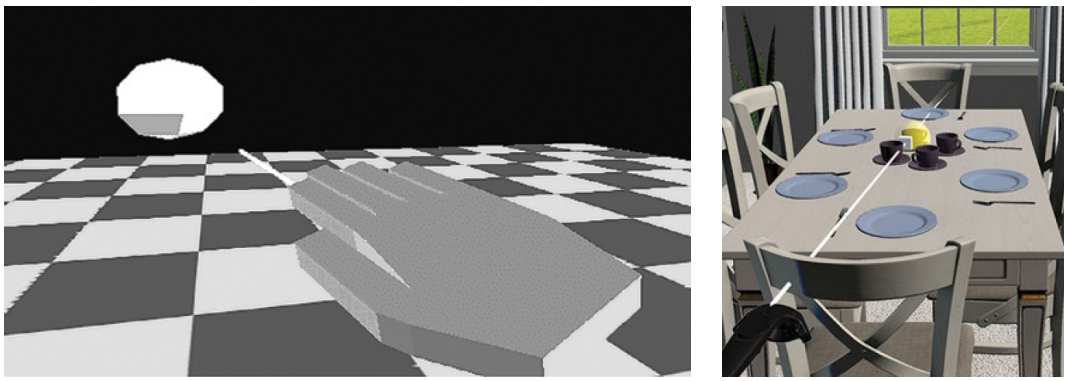

Figure 2.22: Ray-casting technique with virtual hand (left) and virtual controller (right) (Bowman and Hodges, 1997)

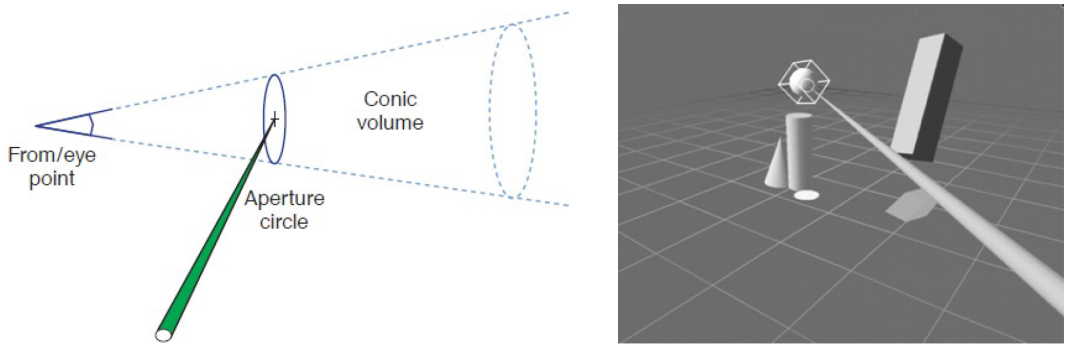

Figure 2.23: Aperture selection technique (left) and an example of use (right) (Forsberg et al., 1996)

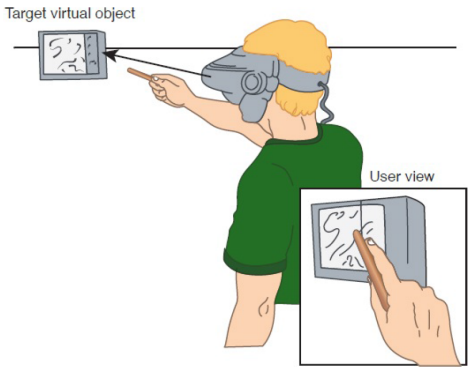

Figure 2.24: The "sticky finger" image-plane pointing technique (Pierce et al., 1997)

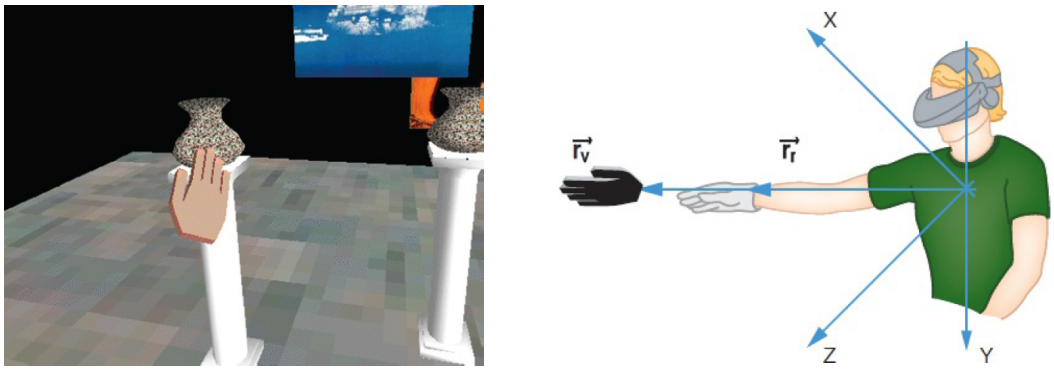

Figure 2.25: The Go-Go technique, the virtual hand (left) and egocentric coordinate system (right) (Poupyrev et al., 1996)

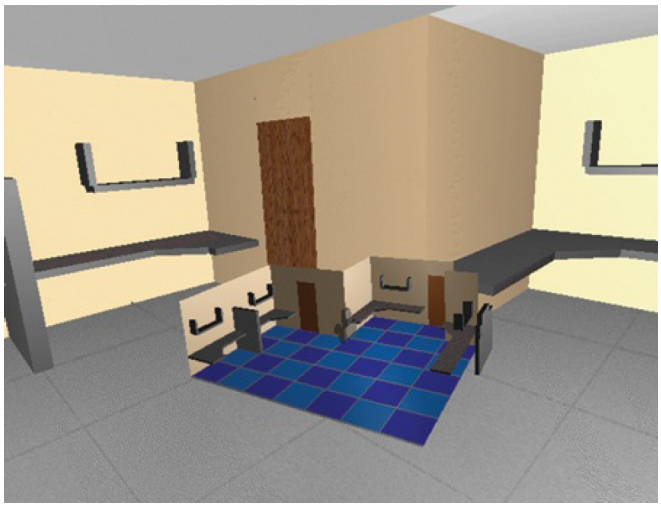

Figure 2.26: The WIM technique, an exact copy VE at a small scale where user can indirectly manipulate virtual object in the WIM (Stoakley et al., 1995)

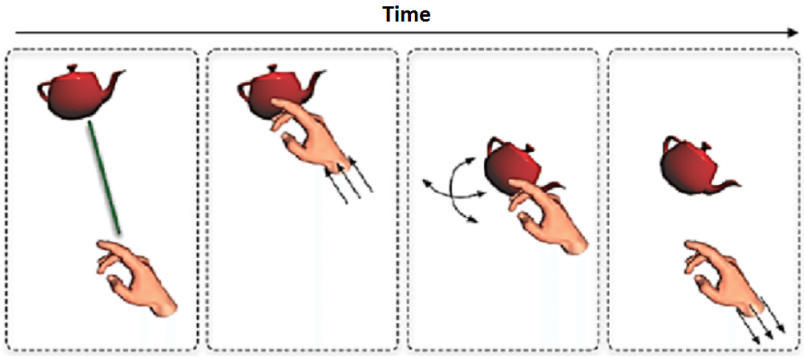

Figure 2.27: The HOMER technique allows user to pick a virtual object, bring it close and return to the original position when user release it (Bowman and Hodges, 1997)

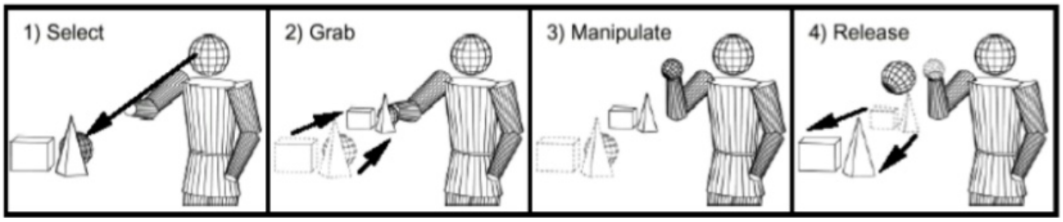

Figure 2.28: The Scaled-World Grab technique scales down entire VE, a user can manipulate using the simple virtual hand (Mine et al., 1997)

2.5 Conclusions

Device technologies provide significant areas to deploy tools with advanced interaction for the generation of VMs. Design of VM will use devices as conveyors of information for knowledge construction, acquisition and integration. According to museum visitor study (Chittaro and Ieronutti, 2004) virtual visitor's behaviors in the VEs were identified as similar to the behaviors of real visitors in a real museum environment. Visitor behavior is an important indicator about interaction in VM which provides learning ability that devices inevitably relate to. Visitors will interact to access the content that they are interested in. The appropriate interaction and devices must be selected thanks to a knowledge adapted to digital heritage applications. Then the research question could start from this point to investigate the needs of interaction and devices potential.

The difference between devices capabilities is part of this knowledge. The contents of VM application would be efficiently accessed with appropriate devices providing better knowledge to the users. Moreover, interaction with the content may imply to user learning and we would like to investigate the factors of interaction affecting user learning. The classification of interaction techniques provides interaction task to access the VE. The study of devices capabilities defines interaction flexibility and limitation.

In Section 2.3, VM development process is presented to show various platforms on it usage where we can conclude that all of them are device dependent platform. Especially, on-site installation is the type of VM where VR technologies are used and play an important role for VM development because high level interaction is expected. Devices will be connected to application as the interaction media with VM content. Therefore, the system used to develop the VM application is based on the selected device and the structure of the application is device dependent. As a result, when the VM application is developed, it is hard to upgrade the device. If new devices are introduced, it will not be available for the original application without modification.

In order to deal with this problem, we provide a device-independent VM development. The application will not depend on the device. The content of the application must be used with a new device without changing its structure. The framework for the new VM development is divided into a high and low abstraction levels. High abstraction level is a part of content development, which is designed to be easy to create content supported to VM. And the low abstraction level is designed to contribute the management of device connectivity and interaction. This enables VM application development to be built at high abstraction level, with the choices of devices at the low abstraction level in last decision steps.

In addition, this framework does not only provide flexibilities of VM applications in a wide range of devices, but it can also be applied to compare the performance of various devices in the same content. It also includes the study of the interaction between users and content through the device, how each device is used in the application. This framework will be useful for designing a VM and selecting the appropriate device that should be used in the

application. This state of the art highlights a very wide set of technologies and options. None of them can solve every VM issue. When investing in technology we would like to provide applications taking benefit and evolving with technology. We need a very nimble system.

Research question 1

- What are the needs of interaction and devices potential?
- How different interactions affect learning and what are the factors?
- How to better immerse museum visitors into VE thanks to interaction?

Research question 2

The research question should start from user interaction with interactive content to investigate the needs of interaction and devices potential.

Research question 3

We would like to investigate what are the key factors of interaction affected to user learning.

Proposition 1

We provide a device-independent VM development to create interactive content and adaptive interaction system where an application is exportable into any device.

Proposition 2

We evaluate the learning capacity by using the device independent VM.

Chapter 3

A Storytelling Platform

In Chapter 2, we discussed the current components in VMs. A VM definition was provided and types of VMs were identified: digital media VM, online VM, interactive Web3D VM, mobile application VM and interactive on-site installation VM. Each VM will use a different platform to develop applications for storytelling and interaction design. In this chapter, we propose a platform dedicated to the interactive on-site installation VM; a specific exhibition is connected to a variety of devices to support user interaction. This platform develops user interactive applications by designing story narrative as well as interaction. However, there are gaps in the development of applications that cannot be interchanged with other devices because the application is attached to the device. We create a flexible platform to enable a variety of devices for effective VM application development.

This chapter firstly discusses devices connection and 3D graphic libraries, then reviews the frameworks used to develop VR applications. There are two different frameworks: the VR framework and the authoring framework. Both are core modules of the overall platform development. After the description of these two frameworks, we describe the components to detail the interaction system and the story transformation into an executable environment.

3.1 Introduction

As already mentioned, interactive on-site installation are composed of VE content development and devices connection to interact with the content. The design and development of VM also consider system interactivity to allow users to learn the content they are interested in (Johnson et al., 2016). We need a storytelling to drive VE behavior while the interaction is the modality to get control on a story. There is a kind of disconnection between the initial specification of the exhibition and its implementation and delivery.

Storytelling leads the creation of interactive content for a VM and use stories as instruments for engaging knowledge transfer (Göbel and Mehm, 2013). In order to create an interactive on-site installation VM, storytelling platforms will be the main tool. It helps developer defining elements to put in a scene and to build interaction. This VM is an interactive content where users are involved interacts with the story.

If storytelling is the main trigger, devices and interaction techniques also contain the design of interactive contents. Most VMs are designed for specific selected devices. Maintenance and service of interaction system will be restricted to the contents of the whole development process. Exhibition management is complicated because there is no lifecycle management for an exhibition to support technology change over time (Khundam and Noël, 2017). But indeed history does not change so much. The VM story should exist independently of the

device. We propose a high-level abstraction to define all events and actions in the scene that is translated into low-level technical user interactions in a second step. Thus, VM developer respects the high-level abstraction story whatever devices are finally used. We thus quite reach a device independent development processes. Devices connection and 3D graphic libraries are included in this development process which is explained first.

3.1.1 Devices connection

Immersive VR can be achieved by focusing on device connection with simultaneous different devices. Each device has distinct characteristic data which must be transferred to the application. Data transmission between input devices and application are mainly divided into three types: tracker, analog and button. The tracker type holds a position and an orientation. The analog type is used for any type of axis such as joystick axis, mouse axis, etc. The button type is used for any type of binary button such as joystick button, mouse button, etc. For instant a mouse has a 2 analogic channels and a 3 button channels. A wand, a typical VR device, has a tracker, an analog data for a joystick, and buttons.

We need a tool to exchange data from different devices towards VR application in a standard model. VRPN (Virtual-Reality Peripheral Network) is widely used to deal with this issue. VRPN provides a network-transparent interface to virtual-reality peripherals (Taylor et al., 2001). VRPN does not aim to provide an overall VR API, but focuses on providing a uniform interface to a wide range of devices, keeping low-latency, robustness, and network-transparent access to devices. VRPN is complementary to VR toolkits, since several users of existing toolkits integrated VRPN as a device-interface layer. It maps communication channels and device drivers transparently and efficiently. A lot of VR framework use VRPN to handle device connection. VRPN remains a key technology to connect interaction device, but the three device types (tracker, analog and button) do not take in change the connection with a rendering device (display): VRPNdoes not support communication with the display. This is managed by 3D graphics libraries.

3.1.2 3D Graphic libraries

Graphics library supports computer graphics rendering to a monitor. Typically, this involves providing optimized versions of functions that handle common rendering tasks. By using these functions, the program can compile images to be swapped to the monitor. This program simplifies the creation and optimization of these functions and allows programmer to focus on creating graphical applications. In general, the graphic library is divided into two types according to the implementation: low-level 3D graphic libraries such as OpenGL and DirectX, and high-level 3D graphic libraries such as OGRE, OpenSceneGraph, VTK, etc. These graphic libraries will be used as low-level toolkit to create VR application.

3.1.2.1 OpenGL

OpenGL is a well-known environment for developing interactive graphics applications on 2D and 3D. OpenGL was initially developed by Silicon Graphics, Inc. (SGI) in 1991 and was published in 1992 (*The Programming Languages Beacon* 2012). It has become the most widely used and supported graphics application programming interface (API) in the fields of computer-aided design (CAD), virtual reality, scientific visualization, simulation, and video games, bringing thousands of applications to a variety of computer platforms. OpenGL speeds up application development by incorporating a broad set of surface rendering, texture mapping, special effects, and other visualization functions.

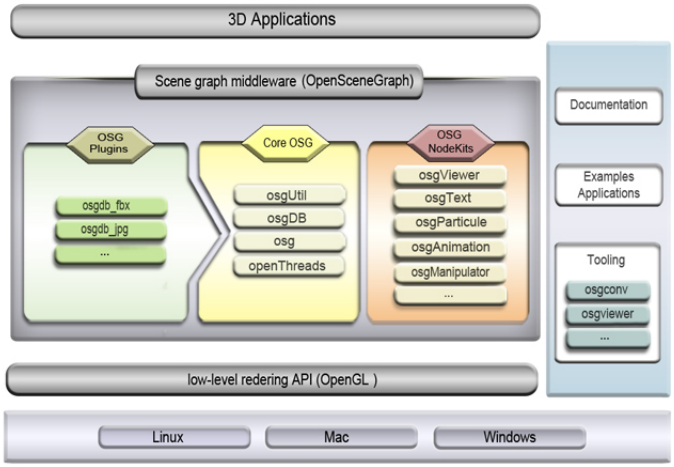

Figure 3.1: Overview of OpenSceneGraph architecture (Retrieved from https://en.wikipedia.org/wiki/OpenSceneGraph)

3.1.2.2 DirectX

DirectX is a collection of APIs designed to allow development and management of multimedia-related tasks, especially for games and video programming on Microsoft platforms. Initially, all of API function names begin with Direct, such as Direct3D, DirectDraw, DirectMusic, DirectPlay, DirectSound, and so on. DirectX is coined as a short term for all of these APIs, the name X standing for the particular API name and became the name of the collection. Direct3D is the 3D graphics API within DirectX which is widely used in the development of video games for Microsoft Windows. Direct3D is also used by other software applications for visualization and graphics tasks. Direct3D can implement advanced features (McMullen, 2014) such as antialiasing, tessellation, interleaved rendering, deferred contexts, compute shaders and instancing. It is indeed a concurrent to OpenGL.

3.1.2.3 OpenSceneGraph (OSG)

OSG is an OpenGL-based high performance 3D graphics toolkit for visual simulation, games, virtual reality, scientific visualization, and modeling. It provides high-level rendering features not found in the OpenGL API (Martz, 2007). The toolkit is written in standard C++ runs on a variety of operating systems and also supports application development for mobile platforms, namely iOS and Android.

The core OSG functionality consists of four libraries: OpenThreads, osg, osgDB and osgUtil. The OpenThreads library is intended to provide a minimal and complete interface used by OSG as the threading implementation. The OSG library provides basic elements used to build scene graphs, such as nodes, geometries, rendering states and textures. The osgDB library provides a plugin mechanism for reading and writing 2D and 3D files. The osgUtil library is designed for building the OSG rendering backend, which traverses the scene tree, performs culling in each frame, and finally converts the OSG scene into a series of OpenGL calls.

The scene graph also has additional modular libraries known as NodeKits (Wang and Qian, 2010). They have been delivered to meet specific development requirements, which can be used to advance 3D application components and graphics algorithms such as osgAnimation, osgFX, osgManipulator, osgParticle, osgQt, osgShadow, osgTerrain, osgText, osgVolume, osgWidget.

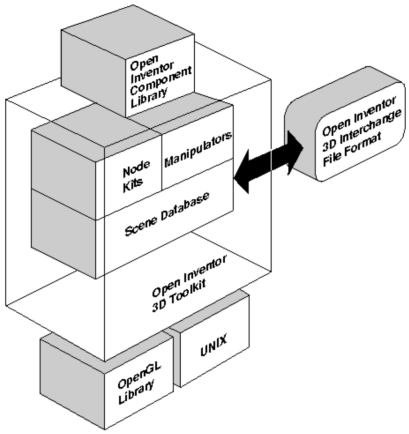

Figure 3.2: Open Inventor architecture (Retrieved from https://developer90.openinventor.com/content/11-what-open-inventor)

3.1.2.4 Object-Oriented Graphics Rendering Engine (OGRE)

OGRE is a 3D rendering engine written in C++ within an object-oriented design. OGRE is cross-platform and abstracts from the underlying system libraries such as Direct3D and OpenGL. The main purpose is to provide graphics rendering, while it also comes with other facilities with a plugin architecture that allows addition of features, thus making it highly modular. The libraries also feature memory debugging and loading resources from archives.

OGRE is a scene graph based engine, with support for a wide variety of scene managers. The landscape scene manager supports progressive level of detail (LOD), which can be created automatically or manually. OGRE also has a compositing manager with a scripting language and full screen video post-processing for effects such as high dynamic range rendering (HDR), blooming, saturation, brightness, blurring and noise. OGRE is designed to make it easier to write programs that divert use of hardware-accelerated 3D graphics supporting vertex and fragment programs along with custom shaders written in OpenGL Shading Language (GLSL), High-Level Shading Language (HLSL), Cg, and Assembly language.

3.1.2.5 Open Inventor (OI)

OI is a C++ object oriented retained mode 3D graphics toolkit designed by SGI to provide a higher layer of programming for OpenGL. The goal of OI is to create a toolkit that made developing 3D graphics applications to better programmer convenience and efficiency. The strategy is based on difficulty of 3D application development because it is time-consuming to do with the low-level interface of graphic language.

OI is written to address this issue and provide a common base layer to start working with object-oriented 3D toolkit (Wernecke, 1994) offering a comprehensive solution to interactive graphics programming problems. It presents a programming model based on a 3D scene database that dramatically simplifies graphics programming. It includes a rich set of objects such as cubes, polygons, text, materials, cameras, lights, track balls and handle boxes. Objects could be sub-classed from a number of primitive shapes and then easily modified into new shapes. OI also defines a standard 3D file format for scene data interchange providing the construction of scene graphs in ASCII files which can be viewed by using the viewers from OI or any common modelling tool.

3.1.2.6 Performer

Performer as known as IRIS Performer (Rohlf and Helman, 1994) is a C/C++ based graphics library of utility code built on top of OpenGL for enabling hard real-time visual simulation applications which can be used to create high performance VR applications. Performer was developed by SGI which continues Open Inventor project. Open Inventor delivered easy-to-use objects and various UI elements to interact with them, while Performer focused on a scene graph system, allowing the various passes of a rendering task to be performed in parallel through multiple threads.

Performer is also based on internal events graphs, but is allowed to correct for even better speeds, even when leaving insignificant objects and polygons to maintain guaranteed performance. Performer used a parallel execution process to increase performance. Unlike Open Inventor, Performer continues to modify the API as needed to keep in step with the latest hardware enhancements.

The scene graph holds a complete representation of all objects in the virtual world. This means that all geometric data within the scene is constructed from node objects. Performer provides a wide array of node objects that applications use to create a scene description. Connecting these nodes in a directed acyclic graph forms a scene graph. It has basic scene graph nodes such as transformation nodes and geometry nodes, but also supports nodes that allow more complex behaviors.

3.1.2.7 Conclusion about graphic libraries

Low-level graphic libraries focus on creating 3D objects. Object information such as shape, size, and location in 3D space, is stored in a scene database providing the necessary functionality to interact with objects and to change the objects in the scene. Those libraries are referred as operating in retained mode where all the data describing a model needs to be specified in advance using predefined data structures. They internally organize the data in a hierarchical database.

Open Inventor offers a data-driven API, only re-render when something changes in the scene, for example when the user changes the viewpoint of the scene. In general a data-driven approach fits better for a general purpose 3D API where constant frame rate is not the main concern. Application-driven APIs are OGRE, Performer and OpenSceneGraph used for simulation software where high and constant frame rates are desirable.

3.2 VR frameworks

When developing VR applications, it is necessary to connect the devices together and boost up the device abilities to enhance realistic graphic or other capabilities through technical process such as cluster rendering. A VR framework connects and access the devices in order to interact with the VE. VR framework is a complete solution that covers the process of preparing, adding interaction, adding animation, viewing, and distributing VR models to create VR experience. A VR framework is also a group of APIs that abstracts multiple domains of resource and extends them by various support tools. The key features of the VR framework are VEs creations and device interfaces to create connections between devices and VE.

In this chapter, we separated VR frameworks into four kinds of toolkit which are VR toolkits, CVE toolkits, toolkits for research and game engine. All toolkits are related with interaction and rendering system. For interaction system, device connection is concerned and some toolkits are related to connect the application and devices using the ready-made class of service. For rendering system, many libraries and software components are created to build VR frameworks. Firstly, we mention device connection tool and then 3D graphic libraries

are used to simulate VE. The toolkit will apply these graphic libraries and connect devices to create application with common API using graphic libraries and the device service. Then, examples of VR toolkit, CVE toolkit, toolkit for research and game engine are described.

3.2.1 VR toolkits

VR toolkits provide reusable components to create VR applications by avoiding starting from scratch and reducing low-level programming and scripting. VR toolkits include functions for handling, displaying, distributing, or managing input devices. The primary requirements for VR development system are performance, flexibility and ease of use. Moreover, capabilities of the environment including cross-platform development, support for VR hardware, rapid prototyping, runtime flexibility and development interface are also considered. Here, we mention the following VR toolkits which are still active and representative of exciting solutions: VRJuggler (Bierbaum et al., 2001), AVANGO (Kuck et al., 2008), Vrui (Kreylos, 2008), FreeVR (Sherman et al., 2013), CalVR (Schulze et al., 2013).

3.2.1.1 VR Juggler

VR Juggler (Bierbaum et al., 2001) is a C++ framework using OpenGL and OpenSceneGraph for rendering. It is middleware designed for the creation of cross-platform, cross-VR-system immersive software applications which provides an abstraction layer between the hardware of a VR system and the VE created in software. It is a virtual platform providing a virtual reality environment independent of operating system. This platform supports many graphics engines and network distribution through the NetJuggler module (Allard et al., 2002), but no high-level support for application distribution. VR Juggler is designed to be an object-oriented system that is divided into components called Managers. Each manager is designed to encapsulate and hide specific system details. There is a manager that links the window system with the graphics library to encapsulate input devices and to configure and control the system. The kernel connects all managers together to communicate within the system.

VR Juggler provides many features such as run-time reconfiguration: system and script can be reconfigured while they are running. It supports multi-pipe hardware and a variety of input devices used in VR systems, including passive and active stereoscopic rendering. It has cross-platform abstractions to make VR applications much more portable across operating systems. Support for use of different graphics programming interfaces such as OpenGL, Performer or OpenSceneGraph. However, VR Juggler does not provide a custom scene graph for application programmers. Moreover, it has no model loader, no collision detection and does not have built-in scene navigation.

3.2.1.2 AVANGO

AVANGO (Kuck et al., 2008) is a free software framework designed for interactive, distributed applications. It supports a large range of displays from standard desktop applications to large-scale immersive VR installations. It is a distributed scene graph framework applying a generic field container. It is encoded based on OpenSceneGraph and develops an entire application with Python scripting enabling rapid prototyping of applications.

One of AVANGO feature develops distributed applications for VR environment. Distributing tasks to multiple machines not only makes the algorithm work more efficiently, but also allows projection to multiple monitors without overloading a single computer. AVANGO does not have a kernel like the VR Juggler to handle multiple graphics libraries, but is built at the top of the scene framework called OSG to display the virtual world. The node object of the graph structure represents the environment, which may be a geometric object, a user command, or an object that generates 3D sound. AVANGO extends the concept of

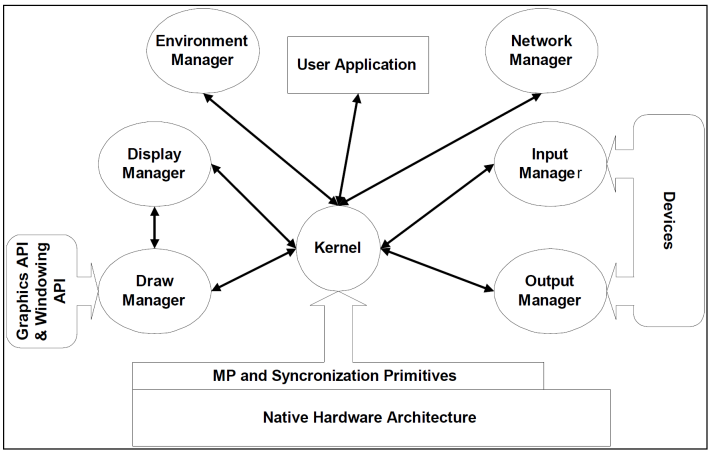

Figure 3.3: The kernel and managers connection within VR Juggler (Bierbaum et al., 2001)

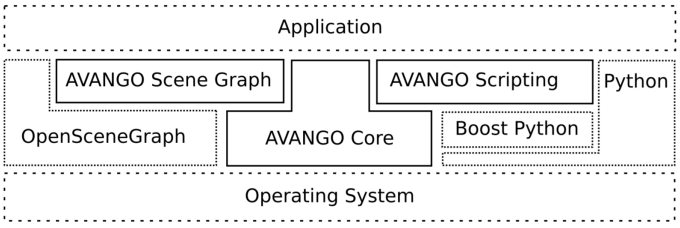

Figure 3.4: System overview of AVANGO based on OpenSceneGraph and Python scripting (Kuck et al., 2008)

a scene graph with a field connection. These fields show the state of the object and may be associated with other object fields, allowing developers to determine complex behavior through the network of objects (Moreira et al., 2011).

With AVANGO, developers can map script language features using Python, allowing users to quickly and easily create and manipulate virtual worlds using immersive 3D. Despite these features, AVANGO is restricted to the OSG library for graphics. Compatibility with other clients that do not use OSG is not possible.

3.2.1.3 Vrui

Vrui (Virtual Reality User Interface) is a C++ development toolkit for highly responsive and interactive VR applications aimed to create completely environment-independent software (Kreylos, 2008). The concepts of Vrui VR toolkit have three important parts: encapsulation of the display environment, encapsulation of the distribution environment, and encapsulation of the input device environment. For the display abstraction, Vrui provides OpenGL rendering contexts that are set up to render a model in user-specific coordinates. For distribution abstraction, the detail aspects of distribution are hidden by the toolkit. For input abstraction, Vrui will hide these differences in hardware and will provide a uniform view of the set of connected input devices. Furthermore, it provides mechanisms not only to hide the hardware details of the input device environment, but also the number and configuration of input devices.

Vrui applications work with an intermediate tool layer that expresses interaction with input devices at a higher semantic level, which separates applications from the input devices available at any environment. System integrators provide input devices that are available at semantic level such as location selection, drag, navigation, menu selection, etc., using the

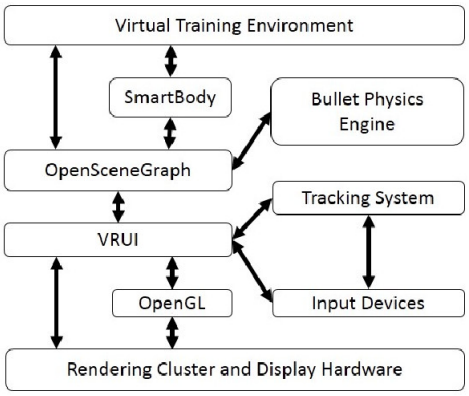

Figure 3.5: Architecture of the simulation environment applied Vrui (Wischgoll et al., 2018)

most efficient and easy-to-use method. By this reason, Vrui applications work effectively in a wide variety of VR systems ranging from desktop systems with keyboard and mouse to fully-immersive multi-screen systems with multiple 6-DOF input devices.

3.2.1.4 FreeVR

FreeVR (Sherman et al., 2013) is an open-source and a cross-platform support for multi-processing. It integrates a variety of VR systems and configurations. Graphics rendering is handled by the OpenGL and OpenSceneGraph libraries. FreeVR is compiled on various UNIX platforms, OS X, and Microsoft Windows which is suitable for use in highly immersive environments and also deployed in desktop-VR settings. It has been designed to work with a wide variety of input and output hardware, with many device interfaces already implemented. FreeVR employs runtime configuration files to specify system details. An administrator can change an execute program for purposes of debugging, fine tuning, or interfacing with the user differently. One of the design goals was for FreeVR applications to be easily run in existing virtual reality facilities, as well as newly established VR systems. The other major design goal is to make it easier for VR applications to be shared among active VR research sites using different hardware from each other.

However, FreeVR is not a VR content library which does not provide a scenegraph layer, or other features often associated with such libraries like intersection testing and collision detection. It does not provide a physical simulation for objects in the virtual world.

3.2.1.5 CalVR

CalVR (Schulze et al., 2013) is a VR middleware system. It is implemented through an object-oriented class hierarchy which is written in C++. CalVR implements the typically used VR functionality of middleware such as CAVElib, COVISE, Vrui or FreeVR and adds to it by supporting nonstandard VR systems such as autostereoscopic displays, as well as multi-user support for viewing and interaction.

CalVR supports several non-standard VR system configurations, multiple users and input devices, sound effects, and high level programming interfaces for interactive applications. It uses the OpenSceneGraph library for graphics output and supports native OpenGL code. Functionality and entire VR programs can be added through a simple plug-in system which allows compiling new modules separately from the main code. They have several built-in navigation methods, an extensible 3D menu system, and support for a variety of 3D display and tracking systems and also for collaborative work at different network-connected sites. All

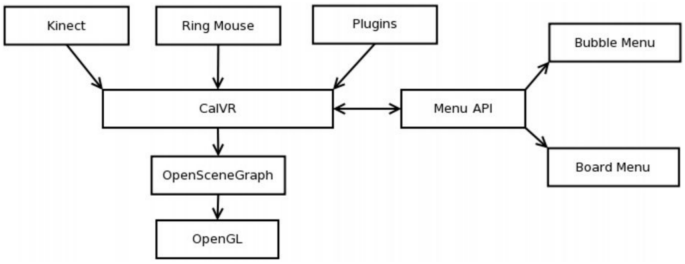

Figure 3.6: CalVR software design (Schulze et al., 2013)

display configuration, tracking, framework and plugin parameters are specified in an XML configuration file. CalVR comes with a customizable menu system, allowing straightforward integration of user-adjustable parameters into plugins.

3.2.1.6 Conclusion about VR toolkits

The purpose of the VR toolkit is to provide reusable components that can be used to create VR applications by avoiding starting from scratch. It reduces the amount of programming effect. VR toolkits may include functions for handling, displaying, distributing, or managing input devices. This includes the ability to use the application and how to tell which system meets those needs. However, these features are just the basis of the VR application development. Especially, considering the properties of the VM narrative, the VR toolkit does not explicitly support this feature. It lacks scene graph design that emphasizes interaction between objects. In the next section, we consider the VR toolkit for specialized research to see how the development is different from the original toolkits.

3.2.2 Toolkit for research

These toolkits created for specialized research will have features similar to the VR toolkit, but are designed for specific tasks. However, these toolkits are interesting in the way they are designed for use with VR devices as well as for further development, so that they can be used to create other VR applications. Examples of toolkit for research are DIVE (Carlsson and Hagsand, 1993), Alice (Conway et al., 1994), inVRs (Anthes and Volkert, 2006), VARU (Irawati et al., 2008), and RUIS (Takala, 2014).

3.2.2.1 DIVE

The Distributed Interactive Virtual Environment (DIVE) is an experimental software environment for the development of multi-user VR applications based on UNIX and Internet networking protocols (Carlsson and Hagsand, 1993). DIVE provides a dynamic VE where applications and users can enter and leave the environment on demand. Several user-related abstractions have been introduced to ease the task of application and user interface construction. In the experiment focused on multi-user and 3D interaction aspects, they found that the effort of designing VEs and applications is essential to develop tools, but also that abstractions support the design of VEs. The use of active replication has also resulted in a degree of fault-tolerance and persistency. They introduced some higher level concepts such as vehicles, visors and behaviors helpful to design VR applications.

3.2.2.2 Alice

Alice (Conway et al., 1994) is an open-source object-based educational programming language with an integrated development environment (IDE) using drag and drop to create

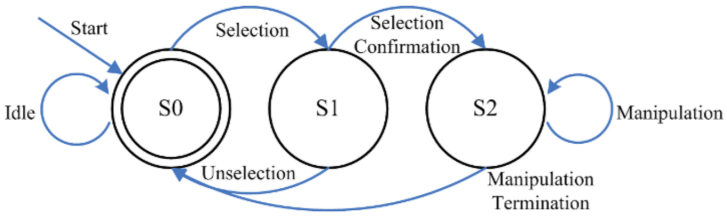

Figure 3.7: Interaction state machine of InVRs framework (Anthes and Volkert, 2006)

computer animations using 3D models. Alice was a VR toolkit that over the years has morphed into a 3D teaching tool for programming education, shedding support for spatial input devices or immersive displays in the process. Alice encourages developers to experiment and to explore 3D programming techniques in an interpreted, object-oriented environment. Developer interact with objects via Alice's 2D GUI tools and Python code. It also supports setting the camera position, activating and deactivating trackers and head mounted displays, and getting statistics back from the simulation such as frame rate.

3.2.2.3 InVRs

InVRs (Anthes and Volkert, 2006) implements Collaborative Virtual Environments (CVEs) approach in the form of a highly extensible, flexible, and modular framework with pre-defined navigation and interaction techniques. Configurable via XML, it has a network distributed virtual world using OpenSceneGraph as a scene graph engine. The high flexibility of different input and output devices is generated through the usage of the interfaces. The advantages of inVRs lie in the abstraction of input and output centralized replicated databases for a high responsiveness. Navigation and interaction methodologies allow high reusability and expandability by using the inVRs libraries or by altering the XML-configuration files. However, this framework does not support advanced animation, scripting possibilities, or menu driven interaction.

3.2.2.4 VARU

VARU (Irawati et al., 2008) is an integrated VR, AR and Ubiquitous Computing (UC) framework designed to make the development of a tangible space application easier and more efficient. Depending on the available resources, the application developer could design an application which involves virtual, physical, or mixed spaces. A single object may have multiple representations in the different interaction spaces. Instead of defining them as multiple objects, they introduced the extension to handle the object consistency across the different interaction spaces. This makes the object management across the different spaces easier to maintain and easier to extend to another interaction space. This framework is implemented to explore different types of mixed-space collaborations and take advantage of the benefits of each collaboration type.

3.2.2.5 RUIS

RUIS (Reality-based User Interface System) (Takala, 2014) is a VR toolkit to address challenges of interfacing with exotic VR devices and their equally exotic drivers, having to rely on low-level input data, issues with compiling software and linking programming libraries, etc. VR applications commonly use a 3D user interface (3DUI), where the user operates in a spatial 3D context involves the 3D position tracking and orientation of user's hands and head. RUIS toolkit provides 3DUI building blocks for creating immersive VR applications with spatial interaction and stereo 3D graphics, while supporting affordable VR peripherals like Kinect, PlayStation Move, Razer Hydra, and Oculus Rift. They implemented a spatial

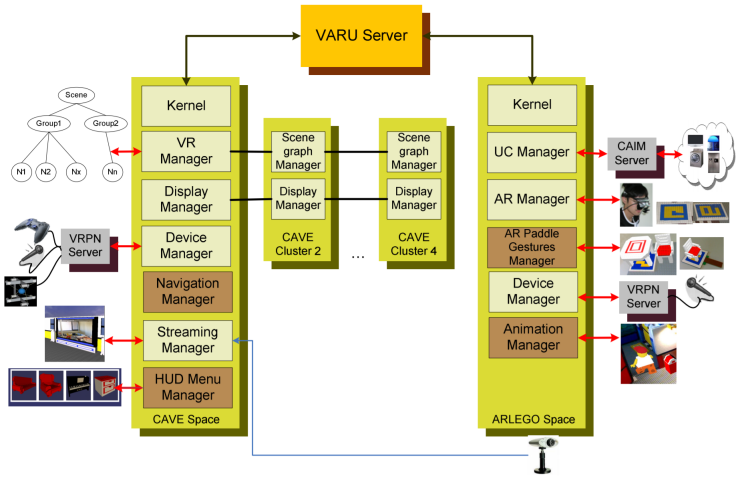

Figure 3.8: The design and implementation of VARU framework (Irawati et al., 2008)

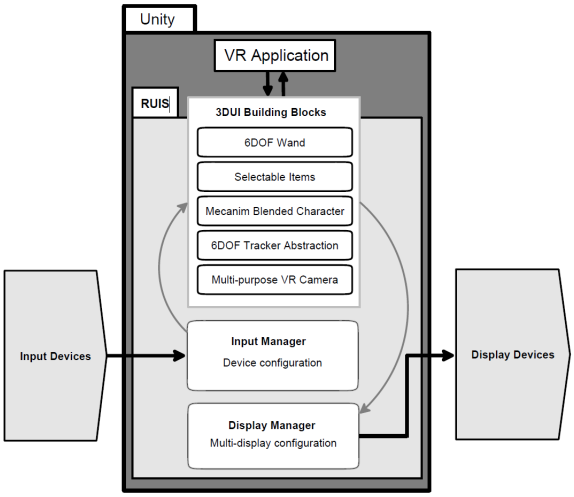

Figure 3.9: The Overview of RUIS of Unity architecture with 3DUI Building Blocks (Takala, 2014)

interaction scheme that combines freeform, full-body interaction with traditional video game locomotion.

3.2.2.6 Conclusion about toolkits for research

Toolkits for research are created for explicit application with use of specific list of devices. Making application development does not differ from the VR toolkit. One thing that has been added is the creation of more IDEs for easier management. It is evident that toolkits for research focuses on VE management rather than on device connection and processing for display. In addition, resource management is accessible of low-level, but it focuses on user-to-system interaction. This is a design that can be applied to the on-site installation VM that focuses on user interaction.

3.2.3 CVE toolkit

Collaborative Virtual Environment (CVE) is relevant for multiple user application. CVE is used to distribute device organization. Most existing CVEs support developer to handle

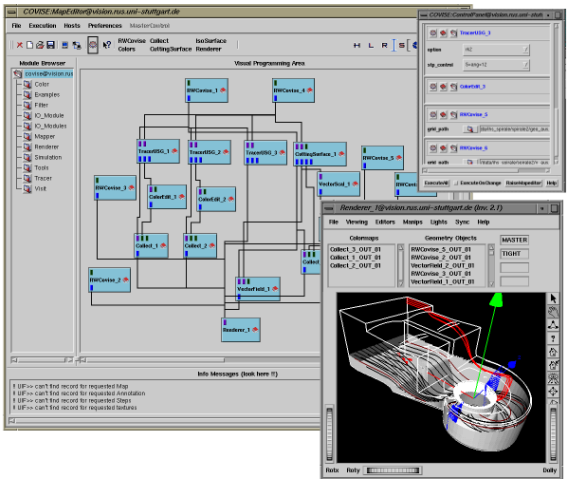

Figure 3.10: Data visualization with COVISE (Retrieved from https://fs.hlrs.de/projects/covise/doc/html/usersguide/intro/intro.html)

device configuration. In order to study interaction process, CVE manages device connection in the system where collaboration and sharing features of VE are considered. Typically, CVE is used for collaboration and interaction of multiple users working together within VR application (Wright and Madey, 2008). However, CVE does not provide only multiple users in the system, but it enables also multiple devices together in the same application. Thus, a CVE is the flexible tool which facilitates device use. User behavior will be studied when content and devices are changed. COVISE (Rantzau et al., 1996), COLLAVIZ (Dupont et al., 2010), and G-SCOP CVE are such CVE systems. In addition, Second Life is also an interesting example of CVE platform with capacities to create a virtual environment and share user interaction.

3.2.3.1 COVISE

COllaborative VIsualization and Simulation Environment (COVISE) (Rantzau et al., 1996) is an extendable distributed software environment to integrate simulations, post-processing and visualization functionalities in a seamless manner with high-speed network architectures. The implementation of the system architecture is done in C++ with the basic communication functionality. COVISE is designed for collaboration, enabling engineers and scientists work together on the infrastructure network. The implementation process can be distributed in different machine platforms to make optimal use of distinct features. Simulation codes can be easily integrated into this distributed software environment by wrapping the code into a COVISE module, allows easy extension of the COVISE architecture. Visualization packages support many features such as high performance computer usage, distributed and collaborative working, integration of custom codes, time dependent simulation and virtual reality.

In COVISE, modules are used to visualize data. Each module is operated as a separate operating system process and communicates with the central controller and local data request broker by sending or receiving control messages via a TCP/IP sockets. The Map Editor with Qt based user interface is the tools to survey all required interactions. The modular approach allows flexibility in distributing certain parts of the visualization applications to a particular computer. A high-speed network is necessary to manage network connections depending on the nature of transferred data. The database approach makes a data request broker necessary. This combination defines the COVISE architecture.

The Controller is the central part of this architecture which supervises the distribution of

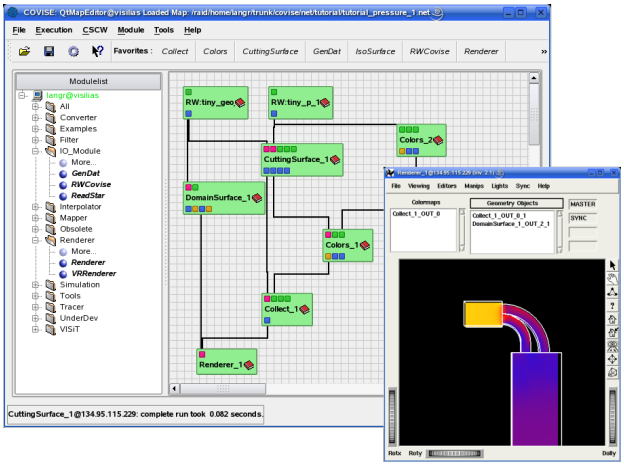

Figure 3.11: The Map Editor and Render window of COVISE (Retrieved from https://fs.hlrs.de/projects/covise/doc/html/usersguide/mapeditor/mapeditor.html)

modules across the associated computer, as well as the management of the application. Therefore, application modules require only a connection to the Controller and COVISE request broker (CRB). The data to be exchanged between subsequent application modules will be kept under the control of the CRB.

The Map Editor is the tool to connect module ports for mapping and modify module parameters with the visual programming on a canvas window. Module icons can move around and connect lines between module ports. The execution of modules is indicated by highlighting the icon boundaries of currently executed modules using the Renderer window through the processing pipeline. Data objects are created when a map is executed and information is shown in the Data Viewer.

The Renderer supports collaborative working with Master/Slave Mode. Every partner in the session has the same viewpoint respect to the rendered geometry objects. Only the master has the ability to change the view in the other renderers. On the slave side it is possible to change the camera position independently from the others as long as the master does not change anything. As soon as the master performs an interaction, the slaves are synchronized and updated automatically. In addition, the renderer supports stereo viewing mode for the shutter glasses and various autostereoscopic viewing devices. It also supports tracking devices such as 6Degree of Freedom devices.

COVISE Virtual Environment Renderer (COVER) is the rendering module for VR with intuitive interaction. COVER is also used as a VR viewer for 3D geometry with 3D menu. It is then a very complete CVE system with the corresponding interaction techniques:

- Navigation modes: xform, scale, view_all, freeze, fly, walk, drive, collide, speed, viewpoints
- View options: coord_axis, specular, spotlight, stereo_sep
- Part manipulation: snap, remove, undo, move_parts
- Animation: forward, backward, anim_speed, steady_cam

3.2.3.2 COLLAVIZ

COLLAVIZ (Dupont et al., 2010) is an innovative multi-domain remote collaborative platform for the simulation-based design applications. Bringing together academic partners and industrialists allow collaborators of the same structure or the same project to work remotely

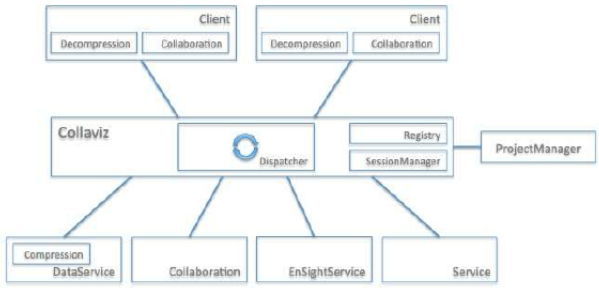

Figure 3.12: General architecture of the COLLAVIZ platform (Dupont et al., 2010)

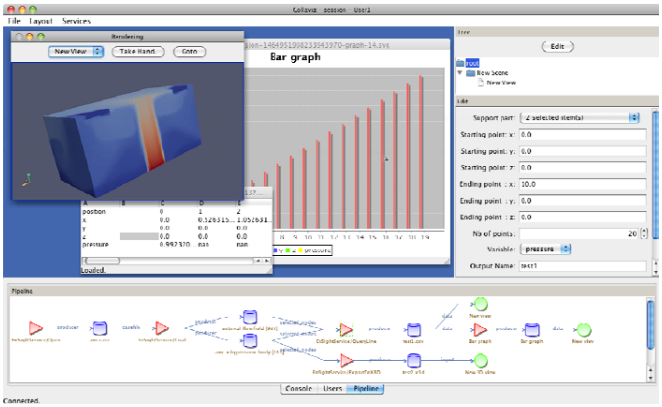

Figure 3.13: The COLLAVIZ client GUI (Dupont et al., 2010)

and simultaneously on the same data and numerical models and to share their results from the simulation. Based on open architecture and open source software bricks, the platform allows users visible an amount of data when dealing with existing resources with using only network capacity devoted to high-performance computing.

The objective is to offer an intelligent solution to destroy the bottleneck of production and data volume processing using mainstream technologies for accessing visualization services with standard hardware. The content is an interactive collaboration with remoting and sharing visualization. This platform provides the proper tools to manage all services with a full transparent access to these scalable resources such as visualization clusters, grid computing, etc.

COLLAVIZ architecture is designed based on SOA (Services Oriented Architecture) concept with clients connected to distributed systems on different servers. The middleware is the core component of this architecture, ensuring the consistency of the information across the whole system which is composed of four main modules:

- Dispatcher, the operation of the communication subsystem is an event loop allows customers, services and core components to exchange commands and notifications.
- Registry, the service description database that keeps tracking of all functionalities provided by plugged-in services.
- Session, the session management component that handles client connections and the processing pipeline.
- Data proxy, the data exchange sub-system which compresses and adapts data to the client and network capabilities.

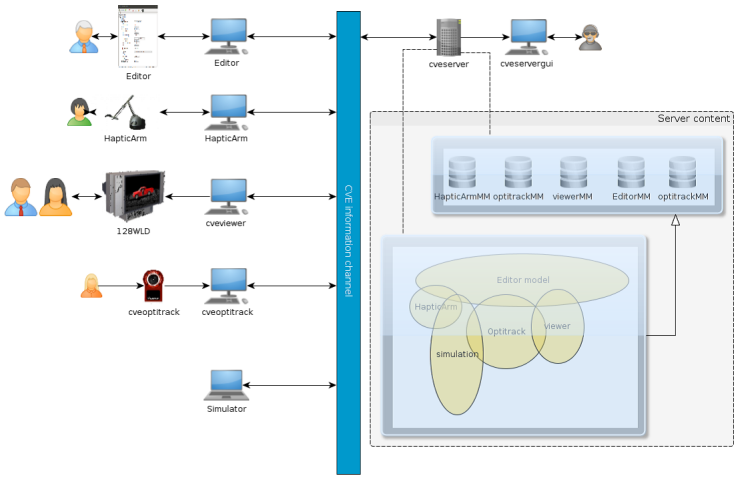

Figure 3.14: The G-SCOP CVE Architecture

The Dispatcher component provides the communication modules for clients and services to connect to the platform. The Collaboration Service enables users to interact on the same pipeline processing with other clients, but also uses real processing services. By the way, even if the client application is running on a simple laptop, they will benefit from all the processing and computing power of the infrastructure that the COLLAVIZ platform is connected.

3.2.3.3 G-SCOP CVE

G-SCOP laboratory provides CVE system dedicated to collaboration within a virtual environment. The CVE system is a multi-agent based system developed as a tool with several VR modules. There is a server to monitor the communication between every agent is defined and connected to the server. A network communication is used to ensure the communication between agents. The modules must be connected to support an expected task and the agents may be any system interacting with the virtual scene. An agent is also called device because it is mainly related to a physical interaction device and rendering display.

3.2.3.3.1 CVE Architecture CVE is designed to handle interaction between user and virtual environment. There are several agents expected to specific behavior of objects. A CVE device may be simulator, software observing the states of some objects and reacting by modifying these or other objects states. It operates a model clustering the states of every agent. Agents must be compatible with their meta-model. It implies the definition of a plug-in and the encoding of its dynamic behavior.

3.2.3.3.2 CVE Server The server is in charge to monitor the behavior of all virtual environments connected several agents together. It has a simple graphic user interface to start and stop the agents. It provides a simple list of connected agents also called devices. Every agent may be distributed on a specific computer by defining the IP address of the associated computer.

3.2.3.3.3 Device connection An agent, a device connection in CVE, is client software for ensuring a part of the behavior of the scene. It has a bilateral connection to the CVE channel which illustrates the synchronization with the server. The agent has a role configured by the analysis of an associated part of the server model. Firstly, to update an internal model reacting to server events and controls the state of an external device. Furthermore, to react

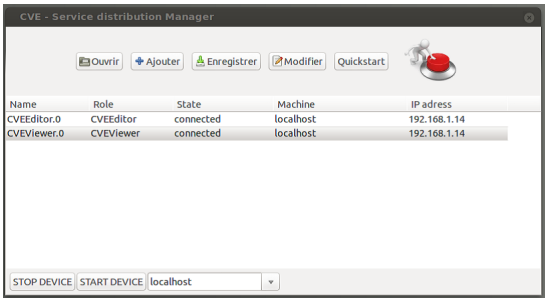

Figure 3.15: The G-SCOP CVE Server interface

some external events updating the model consequently and makes the interaction available through physical devices.

3.2.3.3.4 CVE Model The CVE model is a semantic graph made of nodes and relations between nodes. Each node may have distinguished states values called items of the graph. The details of all CVE models are described as follows:

- cveitem is an abstract class as a basic node which may have attributes and states values. Every item inherits from cveitem.
- cveserver is a cveitem to define the devices which are connected by the server in the application, a list of sub-behaviors are provided.
- cvedevice is a single agent run as a specific process on a dedicated machine. The CVE will connect the agents through the server and interacts with users by changing a display or reporting an action to the user. Every cvedevice have a subbhevior attribute which associates cverules.
- cverule is a logical code that executed when an event occurs on the graph belong to a cvedevice. It is in charge of its management when the cvedevice process is running.
- vstates is a specific attribute values that can be associated with a cveitem. Values can be boolean, real, integer, transformation, vector, string and file. The state values are defined by inheritance of vstate item.
- cveclassifier is a cveitem that describes the template of a cveitem. Then cveitem is associated with a cveclassifier and become an instance of the cveclassifier. The classifier defines the type of an item and provides the semantic of the graph to identify the interpretation of the item and its associated links.
- cveattribute is a cveitem that identifies the name and type of attributes of a cveclassifier.

The server notifies the event towards the devices whenever any point of this model is changed. Any cveitem may reference a classifier which identifies the attributes expected for this node. The classifier definitions are identified through a plug-in system which is a set of classifiers allowing new semantic graphs. It can be associated with specific software implementing the real-time behavior of a cvedevice instantiated one of the defined classifiers.

3.2.3.4 Second Life

Second Life (SL) is an online virtual world developed and owned by Linden Lab with approximately 57 million accounts and 500,000 active users per month in 2018 (Schultz, 2018). In many ways, SL is similar to a massively multiplayer online role-playing game. However, Linden Lab emphasizes that it is not a game because there is no manufactured conflict and

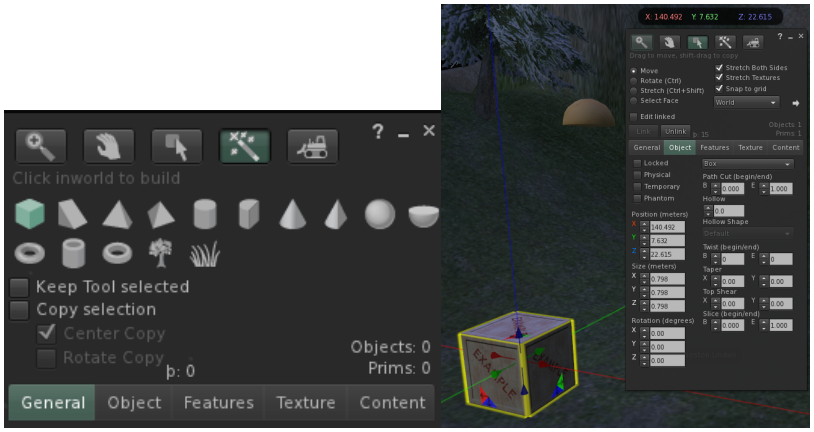

Figure 3.16: Build window and primitive shape in the Second Life and adjusting transformation value in virtual world (Retrieved from https://community.secondlife.com/knowledgebase/english/build-tools-r12/)

no set objective (Hebbel-Seeger, 2013). The software with a 3D modeling tool, using simple geometric shapes allows users to create virtual objects. There is also a procedural scripting language, the Linden Scripting Language (LSL), which can be used to enhance interactivity with objects. Models, mesh, textures for clothing or other objects, animations and gestures can be created using external software and imported. In SL, we can create objects only on land that permits building, object creation is marked building or dropping not allowed if it does not in permit area. Models are created using built-in tools and collaborate in real-time. The virtual worlds can be accessed freely through the Linden Lab's own client programs or alternative third-party viewers. SL users called residents can create their own virtual avatar called avatars and can interact with objects, places and other avatars. They explore the virtual world (grid), meet other residents, participate in either individual or group activities, build, create and trade virtual property and services with others. Second Life has been used in various fields including education and simulation, as well as VM creation.

3.2.3.4.1 Second Life for educational research and simulation SL is applied to several educational research including immersive teaching, real-time collaboration, global community. With the potential of engaging students in fun through interactive 3D environments enable users to collaborate, teach, and create together using multimedia in virtual classroom. A virtual campus created using SL (De Lucia et al., 2009). SL environments and objects have been designed and programmed to support synchronous lectures and collaborative learning (Figure 3.17). Texas A&M University and Florida Institute of Technology uses SL to facilitate and enhance the learning process (Merchant et al., 2013). The 3D virtual world enhances undergraduate student learning of a vital chemistry concept (Figure 3.18). SL features accommodate learners' cultural knowledge simulate real-life scenarios (Chen, 2016) can optimize learners' virtual learning experiences (Figure 3.19). Medical schools and other health-care training programs are using the simulation scene on SL (Krueger et al., 2017). The 3-D virtual world in which users can create avatars and interact in realistic spaces and communities (Figure 3.20).

3.2.3.4.2 Second Life for virtual museum development Museums have been exploring the use of multi-user virtual environments often in the form of proprietary, non-persistent virtual worlds designed and developed. Users are laying the foundations for widespread adoption of museum-like activities occurring in SL (Urban, 2007). These experiments in a new medium can tell us a lot about what real-life museums should consider,

Figure 3.17: A virtual classroom and a detail of the slide presenter in Second Life (De Lucia et al., 2009)

Figure 3.18: The chemistry virtual classroom laboratory in Second Life (Retrieved from http://www.science.tamu.edu/news/story.php?story_ID=1117#.W3vAx84zaAk)

Figure 3.19: A class held in Rose Garden on the VIRTLANTIS Island and students role playing (Chen, 2016)

Figure 3.20: An operating room in Second Life (Courtesy of Imperial College London) (Retrieved from http://discovermagazine.com/2009/jul-aug/15-can-medical-students-learn-to-save-real-lives-in-second-life)

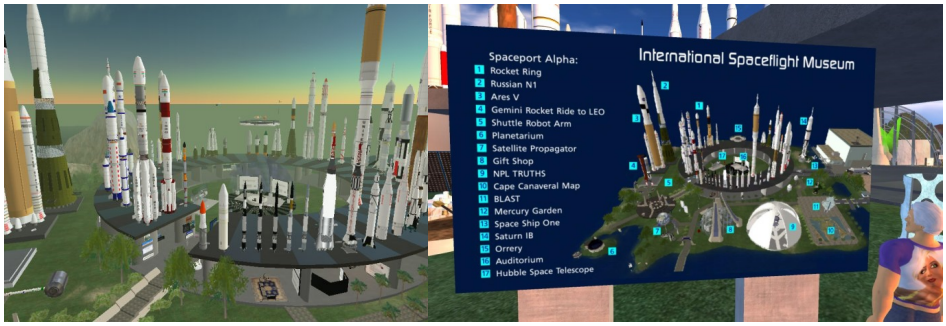

Figure 3.21: International Spaceflight Museum in Second Life (Retrieved from https://virtualoutworlding.blogspot.com/2012/08/edu-international-spaceflight-museum.html)

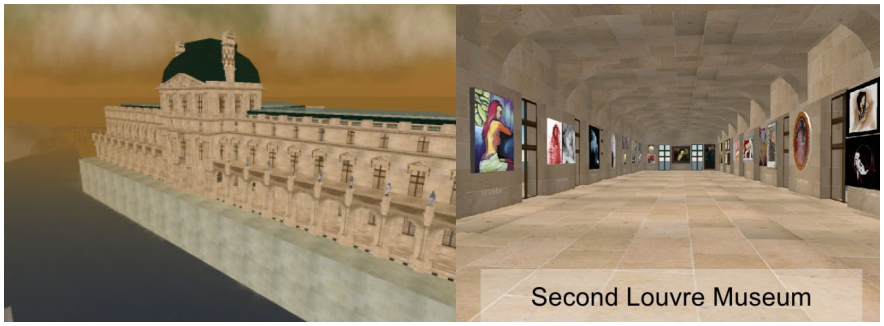

Figure 3.22: The Second Louvre Museum (Retrieved from https://www.slideshare.net/musebrarian/a-second-life-for-your-museum-3d-multiuser-virtual-environments-and-museums)

as well as how to inform and re-merge with physical resources. SL museums can be about dressing up and walking around with other residents acting in character. There are SL museums containing digitized artifacts including labels, lectures, tours, audio guides, docents, even museum shops. Some of the museums created in SL are now seeking affiliations with real-life artists and others are actively establishing themselves as non-profit organizations. The following examples are virtual museum in SL.

The International Spaceflight Museum (Figure 3.21) is an example of a large-scale SL museum with many opportunities for social interaction. The exhibitions include a replica of the lunar landing module, photo galleries, and a large ring circling the island on which a series of rockets and other spaceflight vehicles are displayed. The Second Louvre Museum is a traditional museum installation displaying both classic and modern works of art in a setting specifically designed to replicate a wing of the Louvre Museum in Paris (Figure 3.22). Inside, the museum is divided into different galleries, and the artifacts on display often cover the walls. The Sci-Fi Museum is an interactive museum that uses a variety of multimedia technologies as part of its display. The exhibitions focus on science fiction, television and film, as well as TV and movie posters with regular science fiction films. The USS Defiant and Klingon Prey from the movie Star Trek are floating above the museum (Figure 3.23).

There are numerous of VM in SL which implies that SL supports VM development such as: Art Center, Aho Museum, Crescent Moon Museum, Second Life Science Center, Museum of Flip Animation, SL Computer History Museum, SL Historical Museum, Bayside Beach Galleria-Museum, Star Trek Museum of Science, Tarot Card Museum, Fort Malaya Malay

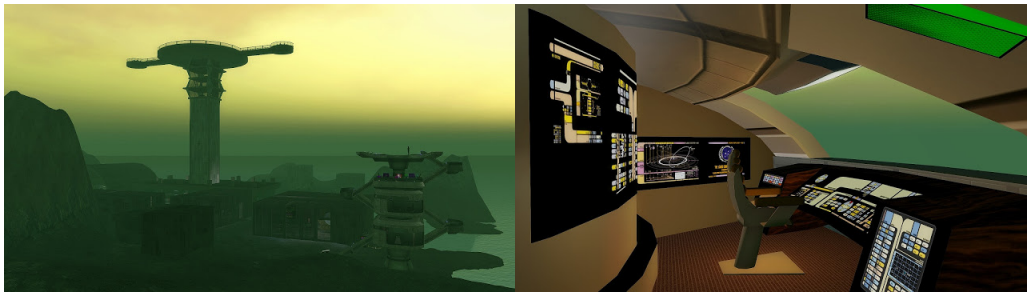

Figure 3.23: The Sci-Fi Museum in Second Life and cockpit of captain Yacht from Star Trek TNG (Retrieved from http://secondlife.com/destination/sci-fi-museum-hub?sourceid=dgw1)

History Museum, Virtual Starry Night-Vincent van Gogh Second Life, Avnet Technology Museum, Museum of Robots, Tech Museum of Innovations, Exploratorium, Kirsti Aho Art Museum on NMC Campus.

3.2.3.5 Conclusion about CVE toolkits

Taking into account the device management features of the CVE toolkit, it is found that each example is different in its use. COVISE is designed for collaborative working on network, supports pre-session access control using multi-server to distribute workload. COVISE modules are several processing step implemented application as separated processes. The objective is to integrate together simulation, post-processing and visualization functionalities. COLLAVIZ focuses on web-based technology developed on the top of shared high-performance visualization platform for scientific and industrial work to make remote analysis. Large data set can be handled on a single workstation, allows engineers and researcher collaboration with only connection is needed. While COVISE and COLLAVIZ are interested on visualization, G-SCOP-CVE provides access to high abstract level resources for visualization and interaction. The platform is a multi-agent system enabling to connect any device to be connected together through a model driven engineering mode. A device plugin is modelled like environment associated with a synchronization algorithm. VR application can be developed on this platform with visualization and interaction support.

Second Life is a research platform for multi-user virtual environments (MUVEs). Their functionality is similar to 3D games with no predefined goals. Second Life can be used to support collaborative scientific visualization (Mouton et al., 2011) considering it as an interesting environment satisfying many of the requirements for collaborative visualization. However, this platform has limited interaction capabilities and it was designed as a social communication rather than a general-purpose MUVE.

3.2.4 Game engine toolkit

A game engine toolkit is the software development environment designed to provide the necessary set of features to build games and brings together several core areas such as graphics, audio, networking, physics, GUI, and scripting. We can import 2D and 3D assets from other 3D software and assemble these assets into scenes as VEs. In the VEs, we also add lighting, special effects, physics and animation, interactivity into an application and optimize the content on the target platforms.

Rendering engine is the important feature of game engine that could be distinct from CVE toolkit. Instead of being programmed and compiled to be executed on the CPU or GPU directly, most often rendering engines are built upon one or multiple rendering application

programming interfaces (APIs), such as Direct3D or OpenGL which provide a software abstraction of the GPU.

The objective of game engine is obviously to create games, but with the capacities as mentioned we can apply game engine to create VR application as well. The difference between game engine and CVE toolkit is the characteristic to create and organize VE. CVE toolkits focus on importing 3D model to the platform and render it to display specification, while game engine is able to customize VE with high-level tool support. The physics engine is another feature usually found on game engine toolkit. It provides with the components that handle physical simulation created in the scene by the game elements, or collisions between components. Moreover, game engine provide facilities including networking, streaming, memory management, threading, localization support, scene graph, and support for cinematics that make it useful for interactive content. Following game engine toolkits are considered the most popular (NewYorkFilmAcademy, 2017) for game developer: Unity3D, Unreal Engine 4, and Blender.

3.2.4.1 Unity3D

Unity3D (https://docs.unity3d.com/Manual/index.html) is a game engine developed by Unity Technologies used to create interactive 2D and 3D virtual environments. Although it involves most video game development, it has also been used to develop other systems, including simulation capabilities and visualization environments for use in education, medicine, and engineering applications. The Unity game engine support game creation system which provides object-oriented scripting frameworks available in three languages: Boo, JavaScript, and C#. These languages can be used to create custom code components that come from a generic class called MonoBehaviour. It gives an access to override method in various stages of game execution, such as during game update loops. Scripts with classes derived from MonoBehaviour can be attached to game objects to control their behavior at runtime. It can also be used to respond to user input, implement custom user interface, store and load data, and use general mechanics. The script components inherited from MonoBehaviour can be associated with various game objects for use of interoperability, and that multiple scripts can be used with the same object to create interoperability.

Unity has an Integrated Development Environment (IDE) that provides extensive support for the development of virtual environments. It includes a number of windows, such as the Scene, Game, Hierarchy, Project and Inspector windows. The Scene window enables a developer to visualize the current state of virtual environments, while the Game window plays and shows the result of testing the scene state. Each element of virtual environment called game objects are tuned in the Inspector window. In the Project window, all project assets will be stored here such as models, textures, audio files and scripts. Developer can access the assets and manage their project in this window.

One of the features that make Unity attractive is the effort to integrate the system by supporting multi-platform development. Unity offers over 20 platforms for publishing. It is used to create applications including web and personal computer such as Windows, Mac OS and Linux, on game console such as Xbox, PlayStation, Wii, and Nintendo, on mobile OS such as iO, Android, BlackBerry, Tizen, on TV targets such as Apple TV, Android TV and Samsung Smart TV. The Unity game engine also offers technical support, a community-based property inventory, an impressive list of game development features, and compatibility with Microsoft's .NET Framework. In addition, the latest version of the game engine is free to use for research and development.

In addition, Unity has close collaboration with leading device manufacturers for VR development experience. Native support is now available for Oculus Rift, Steam VR/Vive, Playstation VR, Gear VR, Microsoft HoloLens, and Google's Daydream. With super-high frame rates thanks to a highly-optimized stereoscopic rendering pipeline and the tools to help

Figure 3.24: Unity IDE with appearance of a number of windows (Retrieved from https://static.filehorse.com/screenshots-mac/developer-tools/unity-screenshot-03.jpg)

developer further optimize their content. Built-in support for numerous platform-specific features with a versatile VR/AR API and provides rapid iterative development.

Advantages

- Unity is a very popular game engine used by a wide range of developers with great communities support through the Forums and Unity Answers.
- Unity provides an exhaustive documentation where everything is given a full description supplied by many tutorials.
- Unity offers choice of scripting languages (C#, JavaScript, Boo) depending on preference or knowledge of developer.
- There are a lot of assets in the Unity Asset Store providing variety of models, audio files, scripts, or modules which support developer to create their projects.
- The GUI editor is powerful and intuitive which allows pausing gameplay and manipulating the scene at any time as well as progress gameplay frame by frame.
- Unity is cross platform deployment, the same source can be deployed on several platforms and also many plugins supporting.
- Unity is supported by many official VR devices. Developer can deploy and publish without paying for anything.

Disadvantages

- Garbage collection in Unity has built on Mono can impact performance or cause stuttering.
- Unity is proprietary, closed source game engine. There is a free version with limited features.
- As a game engine, it is initially restricted to applications with fully defined content. Content evaluation at exaction time is less easy to integrate.

3.2.4.2 Unreal Engine 4

Unreal Engine 4 (UE4) (https://docs.unrealengine.com/en-us/Engine/Editor) is a complete suite of development tools developed by Epic Games , made for working with real-time

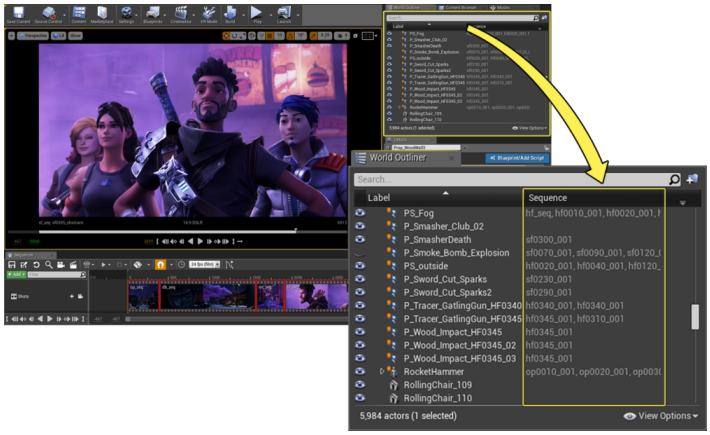

Figure 3.25: The Sequencer in UE4 on editing process (Retrieved from https://api.unrealengine.com/images/Engine/Sequencer/Overview/Outliner.jpg)

technology. From enterprise applications and cinematic experiences to high-quality games across PC, console, mobile, VR and AR. Development platforms that can run the Unreal Editor are on Windows, OS X and Linux. Developers have a toolset and accessible workflows to quickly iterate on ideas and see real-time results without coding, while full source code access gives in the UE4 community the freedom to modify and extend engine features.

High-level shading language (GLSL, Cg, HLSL) are used to support physically-based rendering, advanced dynamic shadow options, screen-space reflections and lighting channels provide the flexibility and efficiency to create realistic content. The built-in Cascade visual effects editor enabled to completely customize particle systems using a wide variety of modules. Leverage particle lights to impact the scene and build complex particle motion with vector fields to mimic reality and create professional levels of VFX polish. UE4 provides Sequencer, professional cinematic designed by film and TV experts to unlocks creative potential with a fully non-linear, real-time cinematic editing and animation tool built for collaboration. It supports to define and modify lighting, camera blocking, characters and set dressing on a per-shot basis and also to create dynamic cut scene variations using cinematic cameras and live gameplay recordings.

Blueprints are authoring tools designed for non-programmers so designers and other team members can help tweak and prototype. UE4's Blueprint scripts resemble flowcharts where each box represents a function or value, with connections between them representing program flow. This provides a better indication of game logic than a simple list of events, and makes complex behaviors easier to accomplish and games a lot faster to prototype. This system is extremely flexible and powerful as it provides the ability for designers to use virtually the full range of concepts and tools generally only available to programmers. In addition, Blueprint-specific markup available in Unreal Engine C++ implementation enables programmers to create baseline systems that can be extended by designers.

In UE4, Content Browser is used to import, organize, search, tag, filter and modify project assets within the Unreal Editor. Create asset collections to be used for individual work or shared with other developers. There is the Marketplace that has thousands of high-quality assets and plugins to accelerate production and bring new functionality which can access new environments, characters, animations, textures, props, sound and visual effects, music tracks, Blueprints, middleware integration plugins, add-on tools and full starter kits. UE4 enables to deploy projects to Windows PC, PlayStation 4, Xbox One, Mac OS X, iOS, Android, AR, VR, Linux, SteamOS, HTML5. Tools and code are available with no additional cost to developers.

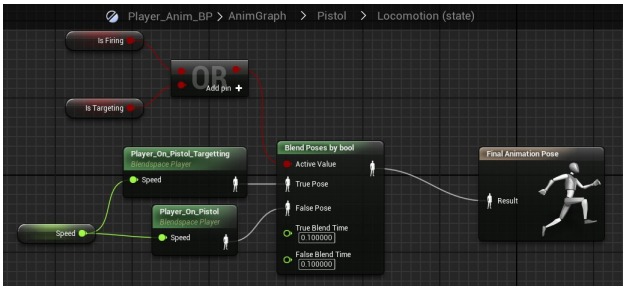

Figure 3.26: The Blueprints Visual Scripting in UE4, a complete scripting system based on the concept of using a node-based interface (Retrieved from https://antonblog.s3.amazonaws.com/2016/ue4-impressions-1.jpg)

Their collaboration with global leaders in hardware and software provides the high quality solution for creating VR/AR experiences with features such as forward renderering, multi-sample anti-aliasing (MSAA), instanced stereo rendering, and monoscopic far field rendering. It provides rendering pipeline up to 90 Hz stereo framerate or faster at high resolutions for VR, no code changes required. Tools that scale from simple to extremely detailed scenes, environments and characters with advanced cinematic and post-processing. UE4 is available for the latest AR/VR device and software including Oculus Rift, Steam VR/HTC Vive, PlayStation VR, and Mac, iOS/ARKit, Google ARCore, Samsung Gear VR, Google VR, OSVR and Leap Motion. Moreover, in VR Mode, the full Unreal Editor runs in VR with advanced motion controls which can build in a WYSIWYG (what you see is what you get) environment to reach out, grab and manipulate objects.

Advantages

- Providing the advance tools such as Sequencer, Cascade VFX etc. to create realistic content.
- A visual scripting system with Blueprints for non-coders enables quick prototyping.
- Fast compilation for quick iteration. Recompiling an entire game to test a small change takes up a lot of time. UE4 quickly compiles in seconds instead of minutes improving iteration time by an order of magnitude.
- Dynamic global illumination with voxel cone tracing similar algorithm to ray tracing, but uses thick rays instead of pixel thin rays to be able vastly decrease the amount of computational power needed.
- Free development license, including full access to source code.

Disadvantages

- Royalty based, even Unreal gives supporting anything but 5% on resulting revenue will go to Unreal.
- Tutorials do not go in-depth enough more documentation is needed.
- In order to start the engine, opening the editor, opening a project, rebuilding shaders, updating references, calculating light maps, saving projects, etc. take long time compared to other engines.
- As a game engine, like Unity3D, it is not really open to content created and evolving at execution time.

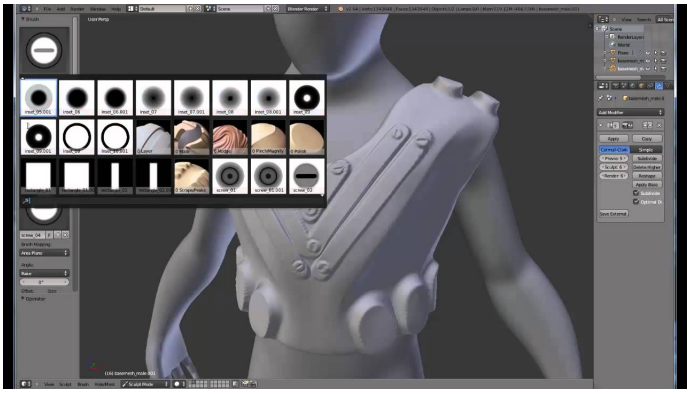

Figure 3.27: Sculpting tool in Blender (Retrieved from https://www.youtube.com/watch?v=aGyPHaPGz4Q)

3.2.4.3 Blender

Blender (https://docs.blender.org/manual/en/dev/game_engine/index.html) is an open-source 3D computer graphics software toolset licensed under the GNU GPL, all code is written in C, C++ and Python. Blender is used for creating animated films, visual effects, art, 3D printed models, interactive 3D applications and video games. Blender's features include a complete 3D pipeline modeling, rigging, animation, simulation, rendering, compositing and motion tracking, video editing and game creation. Blender has its own built-in Game Engine (BGE) that allows developer to create interactive 3D simulations in real-time. The major difference between BGE and the conventional Blender system is in the rendering process. In the normal Blender engine, images and animations are built off-line cannot be modified after rendered. Conversely, the BGE renders scenes continuously in realtime and incorporates facilities for user interaction during the rendering process.

Blender contains features that are characteristic of high-end 3D software with a number of different editors for displaying and modifying different aspects of data. There are Timeline Editor, Graph Editor, Dope Sheet and NLA Editor to support animation editing. In addition, UV/Image Editor, Movie Clip Editor and Video Sequence Editor to support video editing process. For scripting and preference in Blender, they provide Text Editor, Node Editor and Logic Editor to support scripting customized interactions in Python. Several Blender features are provided in details:

Modeling support for a variety of geometric primitives, including polygon mesh, surface modeling and also sculpting. Procedural and node-based textures are implemented, as well as texture painting, projective painting, vertex painting, weight painting and dynamic painting.

Rendering and ray tracing there is an internal render engine with scanline rendering, indirect lighting, and ambient occlusion integration with a number of external render engines through plugins. A path-tracing render engine called Cycles, which takes advantage of the GPU for rendering with a fully integrated node-based compositor within the rendering pipeline accelerated with OpenCL.

VFX Blender offers simulation tools for soft body dynamics including mesh collision detection, LBM fluid dynamics, smoke simulation, Bullet rigid body dynamics, and ocean generator with waves. There is a particle system that includes support for particle-based hair. The BGE offers interactivity features such as collision detection, dynamics engine, and programmable logic. It also allows the creation of stand-alone, real-time applications ranging from architectural visualization to video games with real-time control during physics

Figure 3.28: Motion tracking tool in Blender (Retrieved from https://en.blender.org/index.php/Dev:Ref/Release_Notes/2.64/Motion_Tracker)

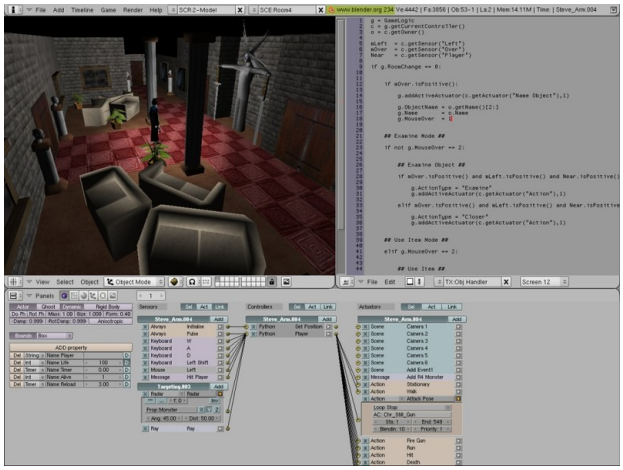

Figure 3.29: Logic Bricks and Python Scripting in BGE (Retrieved from https://www.moddb.com/engines/blender-game-engine/images/screenshot1)

simulation and rendering. Blender now includes production ready camera and object tracking which allows importing raw footage, tracking the footage, masking areas and seeing the camera movements live in 3D scene.

Animation and video editing Blender's animation feature set offers automated walk-cycles along paths, character animation pose editor, nonlinear animation (NLA) for independent movements, IK forward/inverse kinematics for fast poses, and sound synchronization. Furthermore, it offers an impressive set of rigging tools for keyframe animation setting and also basic non-linear video/audio editing.

Scripting BGE provides development of an interactive simulation with a flexible graphical interface called Logic Brick to run scripts on objects in the scene and to define variables called Logic Properties associated with the same object. One of the Logic Brick also allows for the use of Python scripts that interact with the Blender world with dedicated APIs, additional modules can be programmed using Python or C/C++ using SWIG wrappers.

Blender enables to deploy projects to Windows, macOS, Linux, FreeBSD, iOS, Android and also on the Web applications with HTML5 by using Armory SDK.

Advantages

- Blender is free and open source. It is developed to be free for any purpose, including commercially or for education without licensing or payment.
- Blender is being actively developed by hundreds of contributors including professional 3D designers, animators, artists, VFX designers, etc. and integrated all features which can do all processes of 3D production in one platform.
- Blender has a high number of libraries, free character download, and tutorials.
- The Blender cloud library serves thousands of textures for modeling, 3D printing, VFX, rigging and even advance digital painting.
- There is a huge community for support Blender developer.

Disadvantages

- A lot of casual users have to relearn a whole new interface due to its customizable.
- Blender is not deeply used in the professional or commercial work place.

3.2.4.4 Conclusions about game engines

Game engine is designed specifically for video game creation. It contains many different internal systems such as a physics engine, audio engine, rendering engine, AI, animation, and more. There are plugins or APIs available to customize a game, and some game engines come with asset libraries to make it easier to create a game. With these capabilities, VR applications can be easily designed and developed using the game engine. It also has connectivity to VR devices and software SDK to develop VR application. However, there are limitations on accessing specific VR devices and switching devices when interacting with VEs are defined by the selected device in advance.

3.2.5 Conclusions about VR frameworks

All VR frameworks focus on developing 3D applications that support interconnecting devices and interactions between users and systems allow applications to run on those devices. A VR toolkit focuses directly on the device, uses APIs that are close to the hardware, and focuses on networking. It can be developed and upgraded to higher tools to simplify the design and editing of applications. There are high-level features that help to develop more robust VR applications, simplify device tuning, and has more complete graphics interface that provides better access to VE. The toolkit for research is a framework that has been developed for specific researches, and therefore has limitations in its use, but is a good example of how to design and develop a new framework. CVE toolkits are proposed to provide user collaboration in the same VE. This involves the use of VR devices and interaction between users as well as immersive 3D rendering. Thus, these toolkits can be applied to design platform for interaction.

The Table 3.1 shows that VR toolkits have the main approach to develop VR application by scripting without high-level support for editing. They will be useful for developers who need to access in the low-level abstraction. Some toolkits for research support high abstract level with more specific devices and features like physic engine. CVE toolkits aim to support collaboration to share environments focus on visualization. Game engines are high performance tool and provide the most high-level support for developer. However, some features are limited due to commercial issue. They fit well proof of concept development, but less adapted for long term development application. These frameworks have developed emphasize on devices and interaction without the features of VE design. There are other frameworks that offer this feature which called the authoring framework.

Table 3.1: Comparison characteristics and features between all types of VR frameworks

	Framework	**Language**	**Main approach**	**High level**	**Devices**	**CVE**	**Graphic**	**Physic engine**
VR toolkit	VRJuggler	C++	Scripting	No	VR immersive	Support	OpenSceneGraph	No
	AVANGO	Python	Scripting	No	VR immersive	No	OpenSceneGraph	No
	Vrui	C++	Scripting	No	High-interactive	Support	NA	No
	FreeVR	C	Scripting	No	Desktop, CAVE	Support	OpenSceneGraph	No
	CalVR	C++	XML configuration	No	VR immersive	Support	OpenSceneGraph	No
Toolkit for research	DIVE	C	Scripting	No	HMD, wand,glove	Support	SGI	No
	Alice	Java	IDE	Yes	HMD, tracking	No	NA	Yes
	inVRs	C++	XML configuration	No	CAVE, controller	Yes	OpenSceneGraph	Yes
	VARU	C++	XML configuration	Yes	VR immersive	Support	OpenSceneGraph	Yes
	RUIS	C++	Building blocks	Yes	VR immersive	No	on Unity	Yes
CVE toolkit	COVISE	C++	Visual programming	Yes	VR immersive	Yes	SGI	No
	COLLAVIZ	Java/ J2EE	IDE	Support	Visualizations	Yes	VTK	No
	G-SCOP CVE	Python	Scripting	Support	VR immersive	Yes	OpenGL	No
	Second Life	C/ C++	LSL/GUI editor	Yes	Desktop	Yes	OpenGL	Yes
Game engine	Unity3D	C++/ C#	IDE	Yes	Commercial VR devices	Support	Cg/HLSL	Yes
	UE4	C++	IDE/ Blue-prints	Yes	Commercial VR devices	Support	GLSL/ HLSL	Yes
	BGE	C/ C++	Logic Brick	Yes	Commercial VR devices	Support	GLSL	Yes

3.3 Authoring frameworks

Most VR frameworks are proposed to be a tool for VEs development and support developer to handle devices configuration. However, storytelling supporting and to organizing story structure are still lacking. There are some frameworks providing high-level abstraction to define object behaviors in general for interaction which could be improved as educational game tools or authoring framework. An authoring framework is used to preset for interactive multimedia development which can be defined as software that allows developers to create multimedia applications to manipulate multimedia objects. It usually consists of an authoring language, which is a programming language (or extension) that has functionality for displaying instructional systems. There is overlap between authoring languages with domain representation functionality and domain-specific languages.

In educational application development, authoring frameworks allow non-programmers to easily create application with programming features. The features are included behind the interface or tools, so the author does not need to know how to program. In general, authoring frameworks will have interactive graphics and other tools needed for education. The three main components of the authoring frameworks are content organization, content control, and content assessment. The content organization allows users to structure and sequence content or other multi-media. Content control refers to the ability to determine presentation of content and storytelling to engage learners with the content. Content assessment refers to the ability to test the learning outcomes within the system. The following authoring frameworks are developed to support the creation of a learning application that uses storytelling as a component: Adventure Author (Robertson and Nicholson, 2007), Storytelling Alice (Kelleher et al., 2007), StoryTec (Göbel et al., 2008), <e-Adventure> (Torrente et al., 2010), WEEV (Marchiori et al., 2010; Marchiori et al., 2012), Thinking Worlds, Scratch.

3.3.1 Adventure Author

Adventure Author (Robertson and Nicholson, 2007) focuses on young learners as the target group and uses a set of plug-ins to create games. This system is good at encouraging young students to create stories without real interactivity. The system is composed of two primary components: an authoring interface which enables authors to specify an interactive story and a game engine in which the story specification is rendered. Story details generated in the authoring tool are stored in XML format which used Java data structure to display the story. When a story is saved, the XML format is automatically generated from the Java object. The advantage of storing in XML format is to separate the use of the authoring tool from any particular platform. The approach is based on the visual language, but focused on the development of linear stories.

3.3.2 Storytelling Alice

Storytelling Alice (Kelleher et al., 2007) is a variant of Alice 2.0 which proposed to teach programming concept for students. Storytelling Alice provides 3D character and scenery with custom animations. It includes high-level animations that enable users to program interaction between elements and also a story-based tutorial that introduces users to programming through story building. Alice is designed to teach programming theory without the complexity of languages and also to appeal exposed to computer programming by encouraging storytelling. Users can place objects from Alice's gallery into the virtual world that they imagine, and then they can program by drag and drop tiles showing the logical structures. In addition, users can manipulate Alice's camera and lighting for further enhancements used for 3D user interfaces. However, it supports the full object-based programming, event driven model of programming.

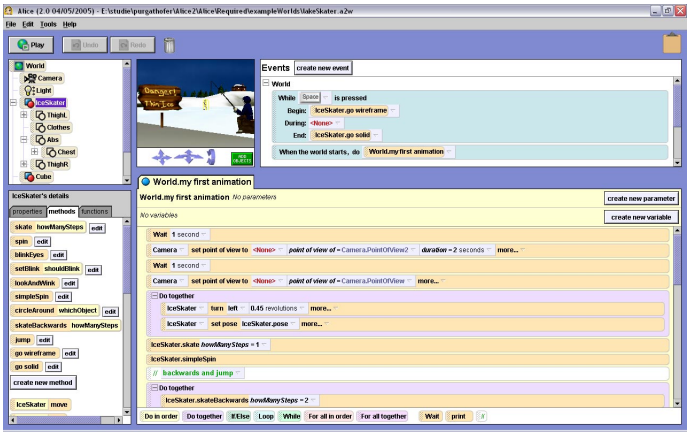

Figure 3.30: Storytelling Alice's interface allows for the creation of animated movies using drag and drop elements while teaching programming concepts (Retrieved from https://en.wikipedia.org/wiki/Alice_(software))

3.3.3 StoryTec

StoryTec (Göbel et al., 2008) is a visual programming language for authoring and experiencing interactive or non-linear stories to encourage creativity rather than educational game. This framework introduces storytelling concepts into the game or the story development model which includes both an authoring environment and a runtime engine. StoryTec is a powerful expressive story creator based on a pluggable framework composed of different parts: Story Editor, Stage Editor and Action Set Editor. Interactive storytelling for creative people (INSCAPE) is applied to a hierarchically organized story graph in the Story Editor which used to manage the story structure while the Stage Editor is used to edit scenes of the story. The Action Set Editor is a visual editor of high-level story logic in a scene basis and the Asset Manager used to import different assets into the scenes.

3.3.4 <e-Adventure>

<e-Adventure> (Torrente et al., 2010) is a framework proposed to facilitate the integration of educational games and simulations in educational processes developed by <e-UCM> the e-learning research group at Universidad Complutense de Madrid. This tool has features to support point and click (action of user interaction using mouse) adventure games and interactive fiction, graphical editor for authoring and XML notation for the description of the games. Developers can use the graphical editor to create the games or directly access the documents that describe the adventures using XML markup. The scene editor supports developer to define and configure settings of the VEs and also the camera and lights settings. Moreover, it provides an editor to define the interactive elements inside the VEs. The end-user will interact with predefined elements by performing certain tasks and actions. Shareable Content Object Reference (SCORM) provides a runtime API and data model used for communication between content objects and learning management systems. SCORM packages in XML files are used to enable content sharing with other systems.

3.3.5 WEEV

WEEV (Writing Environment for Educational Video games) (Marchiori et al., 2010) is a framework to create narrative point and click educational games which is built upon <e-Adventure> framework. WEEV implemented three tools to edit main element which are Actor Editor, World Editor and Story Editor using visual programming language to represent

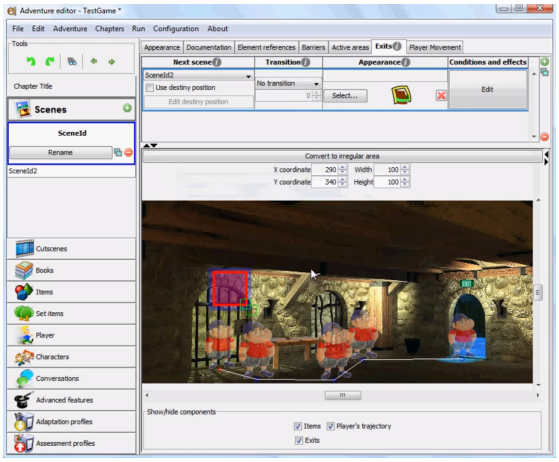

Figure 3.31: Screen-shot of the <e-Adventure>educational video game editor (Retrieved from http://e-adventure.e-ucm.es/tutorial/minitutorial.php)

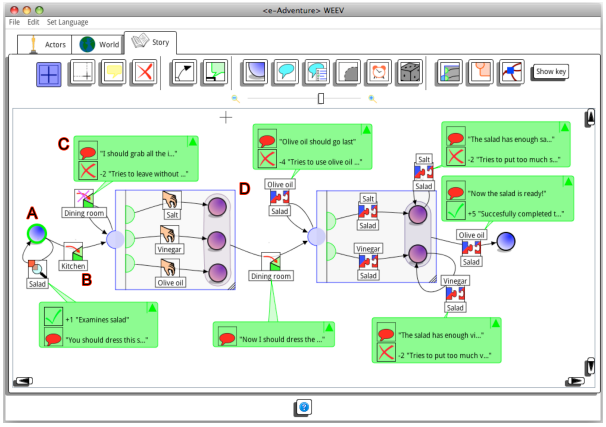

Figure 3.32: The story edition panel of WEEV presents a toolbar on top to add new expressive elements to the story (Marchiori et al., 2010)

a story based on interaction between user and game (Marchiori et al., 2012). The actors are the 3D objects upon which the player can perform an action and interact with them. The Actors Editor enables developers to view a list of all the actors in the game in a specific panel to define the interactive elements in the game. The World Editor uses a Domain Specific Visual Language (DSVL) to represent an abstraction of the story-flow which allows the developers to define the space of VEs including the different interactive elements and non-interactive elements of the educational game. On the other hand, the Story Editor is used to define the story-flow of adventure games that must be edited separately to define the player's interactions and the game feedback.

3.3.6 Thinking Worlds

Thinking Worlds (http://www.thinkingworlds.com) is an authoring framework for VEs focused on creating structured learning experiences specifically designed for education which publishes highly immersive simulations or games rapidly. This tool attempts to create more complex scenes which provide 3D rendering engine and also camera and character control through the scene using a 3D editor that allows the combination of existing resources to create 3D interactive worlds. Although the interactivity of the game has been relatively

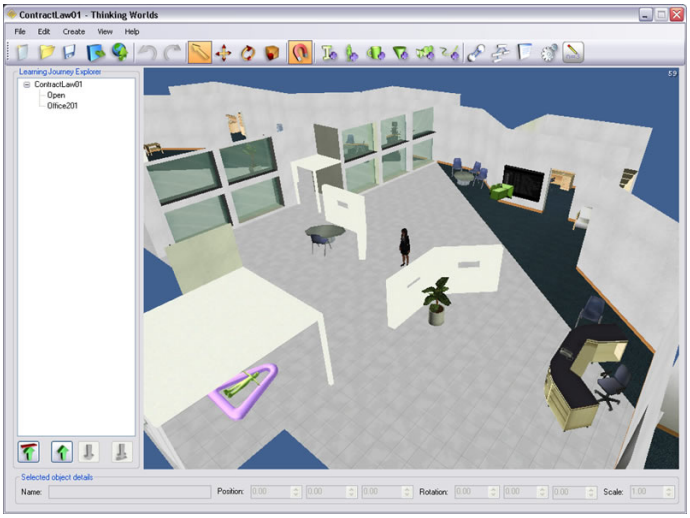

Figure 3.33: The Thinking Worlds 3D environment authoring tool (Retrieved from https://www.dontwasteyourtime.co.uk/games/creating-games-with-caspian-thinking-worlds-software/)

limited, the flow editor offers a new approach to game development. This flow editor uses hybrid text and visual systems to present the flow graphically, but different elements within this flow use a textual representation. This makes a rather complex representation, but allows many specific details for directly editing specific elements.

3.3.7 Scratch

Scratch (https://scratch.mit.edu/) is a visual programming language that makes it easy to create simple interactive stories, animations, games, music, and art which can be shared on website. Scratch is often used in teaching coding, computer science, and computational thinking. To let children learn important mathematical and computational ideas, while also learning to think creatively, reason systematically, and work collaboratively. There are many ways to create story by personal sprites and backgrounds. First, developers can create sprite manually with Paint Editor provided by Scratch. Then, choose a sprite from the library and blocks of commands can be applied to it by dragging them from the Blocks Palette onto the stage. There are several available blocks which categorized as the Motion, Looks, Sound, Pen, Data, Events, Control, Sensing, Operators, and More Blocks. Scripts associated with the selected sprite can also be individually tested under different conditions and parameters via double-click. The blocks-based grammar of Scratch has influenced many other programming environments and is now considered a standard for introductory coding experiences for children. With Scatch teenagers can develop specific games or interactive stories.

3.3.8 Conclusions about authoring frameworks

In summary, each framework has a different story generation model depending on its implementation such as linear stories, non-linear stories, event-driven and story-driven. Standards for the development of frameworks are used to design system appearance including: INSCAPE, SCORM and DSVL. Furthermore, some frameworks provide the tools to support the user's learning assessment which usefully to evaluate efficiency of applications. These issues can be summarized as in the Table 3.2.

VR framework is concerned to the device connection and interaction techniques, whereas

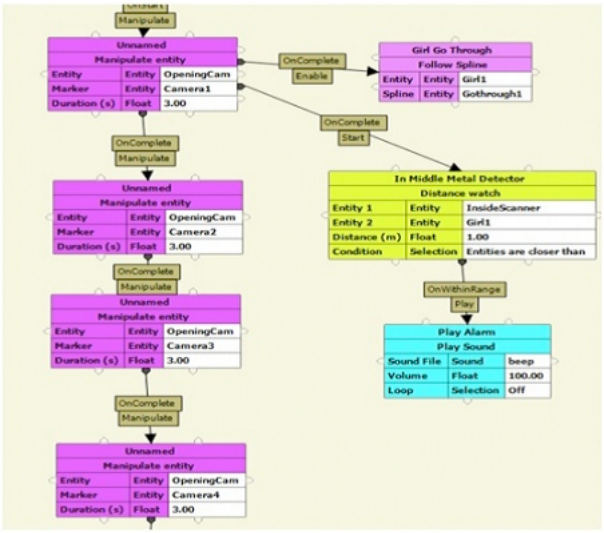

Figure 3.34: The Storyboard editor in Thinking Worlds allows the definition of the flow of the game, mostly concatenating predefined actions and resources modifications (camera positions, NPCs, etc.) (Retrieved from https://www.youtube.com/watch?v=UtHbRc_Ju4w)

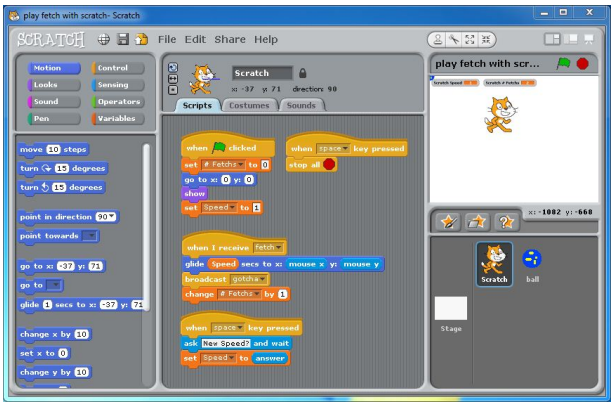

Figure 3.35: Simple program logic created using Scratch (Retrieved from https://onedublin.files.wordpress.com/2010/01/scratch-computer-programming-example.jpg)

Table 3.2: Comparison characteristics and features of authoring frameworks

Framework	**Paradigm**	**License**	**Main approach**	**Educational features**	**Standards**	**Edition complexity**
Adventure Author	Linear stories	NA	GUI editor	NA	NA	High
Storytelling Alice	Event-driven	Free	Visual Language	NA	NA	Low
StoryTec	Non-linear stories/ UML	NA	Visual Language	Evaluation	IN-SCAPE	High
<e-Adventure>	Story-driven	LGPL	GUI editor	Evaluation, Adaptation	SCORM, IMS CP	High
WEEV	Story-driven/DSVL	LGPL	Visual Language	Evaluation, Adaptation	SCORM, IMS CP	Low/ High
Thinking Worlds	Event-driven	Commercial	Hybrid Visual Language	Evaluation	SCORM	Low
Scratch	Event-driven, Block-based	GPLv2	Visual Language	NA	NA	Low

content design is not the main approach for this framework. By the way, authoring framework is handled on the content organization which can be applied to create VEs and build into stories, but this framework usually remains working on the desktop device with simple interaction.

In term of interactive on-site installation VM, we need the tool handle with the user interactions and also the content design to create applications. We aim to build a tool for assisting developer to define environment in a scene and to make stories in term of storytelling as well as device connection in term of interaction system.

Storytelling plays an important role to drive VE behaviors while interaction system deals with low-level device connection. The tool that supports both storytelling and interaction system is related to the functions such as high-level abstraction to define the story, animation tools, physic engine, device connection and interaction techniques. According to the existing frameworks both VR and authoring frameworks, these functions are not fully provided on each platform to address development of interactive on-site installation VM application. We compare the available technologies with these requirements as showed in the Table 3.3.

As we mentioned about the storytelling and interaction system, those kinds of requirements show the differences (advantages and disadvantages) of each framework. The main approaches for VR frameworks are scripting with some xml configuration support. The system has been set up to adjust the parameters according to the desired values. CVE toolkit and game engine have their own IDE or specific visual language which allows scripting. However, some VR frameworks have been developed with GUI editor, whereas all authoring frameworks have GUI which may be a feature of visual language. With abilities of visual language, high-level visual components have been developed to promote low-level user interaction such as an icon in the flow representation which could be found in Storytelling Alice, StoryTec, <e-Adventure>, WEEV and Thinking Worlds. About the VR framework, depending on the nature of the use, some of them can be a middleware supported to implement a higher-level tool on top of them.

Storytelling abilities are the modules to support VE creation including animation and mul-

Table 3.3: Comparison characteristics and features of both VR frameworks and authoring frameworks

	Framework	Main approach	Storytelling	Animation support	Physic engine	Devices	Interaction
VR toolkit	VRJuggler	Scripting	No	No	No	VR immersive	Device dependent
	AVANGO	Scripting	No	No	No	VR immersive	Device dependent
	Vrui	GUI	No	No	No	High-interactive	Device dependent
	FreeVR	GUI	No	No	No	Desktop, CAVE	Device dependent
	CalVR	XML configuration	No	No	No	Limited	Device dependent
Toolkit for research	DIVE	Scripting	No	No	No	HMD, wand,glove	Device dependent
	Alice	IDE	Yes	Yes	Yes	Desktop, HMD	Device dependent
	inVRs	XML configuration	No	Yes	Yes	CAVE, controller	Pre-defined
	VARU	XML configuration	No	Yes	Yes	VR immersive	Pre-defined
	RUIS	Building blocks	No	Yes	Yes	VR immersive	Device dependent
CVE toolkit	COVISE	Visual programming	No	Yes	No	VR immersive	Device dependent
	COLLAVIZ	IDE	No	No	No	Visualization	Device dependent
	G-SCOP CVE	Scripting	No	No	No	VR immersive	Device dependent
	Second Life	LSL/GUI editor	Support	Yes	Yes	Desktop	Point and click
Game engine	Unity3D	IDE	Support	Yes	Yes	Commercial VR devices	Device dependent
	UE4	IDE/Blueprints	Support	Yes	Yes	Commercial VR devices	Device dependent
	BGE	Logic Brick	Support	Yes	Yes	Commercial VR devices	Device dependent
Authoring framework	Adventure Author	GUI editor	Support	Yes	No	Desktop	3DUIs
	Storytelling Alice	Visual Language	Yes	Yes	Yes	Desktop	3DUIs
	StoryTec	IDE/Visual Language	Yes	Yes	No	Desktop	3DUIs
	<e-Adventure>	GUI editor	Yes	2D	No	Desktop	Point and click
	WEEV	Visual Language	Yes	2D	No	Desktop	Point and click
	Thinking Worlds	Hybrid	Support	Yes	Yes	Desktop	3DUIs
	Scratch	Visual Language	Support	2D	No	Desktop	Point and click

timedia. VR toolkits and CVE toolkits do not support these features by default, while game engine support storytelling with the features of cinematic processes and video editing. Alice framework is the only one providing this feature since the framework is developed for research on programming education. Even though authoring framework is concerned to narrative and storytelling, but some of them do not have this feature directly such as Adventure Author, Thinking Worlds, and Scratch. They support storytelling by visual language logic as well as game engine can do.

Most VR frameworks do not have animation features since they focus on connecting to devices rather than being a tool to create a VE. However, the frameworks for specific research such as Alice, inVRs, VARU and RUIS are designed to support animation making. CVE toolkits support just only simple animation without specific tool. All game engines are available to use the animation features, especially, they provides specific editor to edit key frame for animation making and also rigging for complex model movement. Authoring frameworks are especially designed to create content which has the animation tools as a component of story flow, but some of them support only simple 2D animation.

Physics engine is used to simulate the real-world characteristic for VE through collisions or gravity. This feature is applied to enhance the realism of the application which is applied for interacting with the 3DGUIs. Considering the devices support issues associated with the VR frameworks for the low-level toolkit and toolkit for research. Most of them focus on specific implementation, thus supporting relatively limited device connection, but also increasing connectivity later. The high-level features toolkit is different from others. They support a broader range of devices and most devices are fully immersive. Physic engines are provided by game engine, not only collision detection and dynamic engine, but also particle simulation for VFX. The game engine is useful for this feature. Most CVE toolkits also provide this feature, but they focus on simulatiosn that are not realistic.

Devices and SDK are concerned to develop VR application to go beyond 2D perception. Immersive VR devices are applied to use in various projects. VR frameworks support device connectivity, depending on setting and related tasks. While game engine focuses on supporting commercial devices to keep collaboration with their users, usually HMD devices and AR features on mobile. Another interesting point is that all authoring frameworks only support desktop devices because they do not focus on applying VR technology. This issue is a direct impact on the interaction system. There are two types of interaction will be found: point and click interaction that supports simple manipulation in 2D and 3DGUIs is applied to manipulate with VEs in 3D on desktop device.

Considering interaction in VR framework, some VR toolkits have limited interaction because they only focus on connecting devices to the system, but they can be configured later by scripting. While some toolkits provide high-level features which have more flexibility to increase interactivity depending on the device being used. Toolkit for research may be pre-defined in advance because of its clear purpose and can change later such as inVRs and VARU framework. All VR frameworks will have the same interaction based on basic 3DGUIs, but they have different features depending on its usage. Some toolkits provide a flow editor, a higher abstract level to approach object interactivity instead of scripting. This allows us to specify the details that can be set up for any object with interaction. We will apply this feature to proposed platform.

Table 3.3 shows that VR framework supports devices organization and CVE supports users' collaboration, but most of them do not contribute for storytelling, animation and simulation. Game engine supports game creation useful for develop VR application which provides animation and physic engine for simulation, but complete tool for storytelling is still needed. Authoring framework has abilities for developing structured learning, but applications remain on desktop device and need more interaction.

The state of the art demonstrates that main expected functions were developed which are device organization and authoring features, but remain disconnected and no system is complete. Especially, the creation of an interactive on-site installation VM needs storytelling abilities to support educational features for user learning. It does not only advance interaction system, but also enable device switching to study various devices performance when using it. A new system and integration layer are proposed to combine all features together. In order to investigate the appropriate devices to develop interaction system which is needed to complete the development of on-site installation VM. Therefore, it would be great to combine good features of each framework together.

3.4 Storytelling conceptual model

The development of a storytelling platform expects a story model. Story model is used to describe the elements of a story. History is transferred from a exhibition designer into simple text or media forms into sub-elements. Then, it should be converted into VR application. This model must enable mechanism for interpreting and generating stories in different media carried out by the basic concepts related to ontologies. Ontologies provide the formal expression of various components of narratives which is the basic requirement to generate storytelling. Furthermore, semantic web is often used to refer the formats and technologies which enable the collection, structuring and recovery of linked data. Semantic web provides a formal description of concepts, terms, and relationships within a given knowledge domain. These technologies are specified as W3C standards including: Resource Description Framework (RDF), a general method for describing information and Web Ontology Language (OWL), a family of knowledge representation languages for authoring. Ontology and semantic web technologies are related in the fields of existing knowledge management and derive additional information to user.

3.4.1 Ontology Technologies

The Suggested Upper Model Ontology (SUMO) (Pease et al., 2002) is a large formal ontology with detailed axiomatization of general concepts and specific domains. SUMO was created by merging a number of existing upper-level ontologies. The SUMO is mapped to WordNet (Miller, 1995) a huge structured lexicon to promote the use of SUMO in natural language understanding applications. However, SUMO has not emphasized the large numbers of simple facts due to the huge space of human knowledge. Therefore, it has only limited knowledge of the world. The Yet Another Great Ontology (YAGO) (Suchanek et al., 2007), on the other hand, is a large and extendable ontology of high quality. It contains 1 million entities and 5 million facts. YAGO and SUMO were merged into YAGO–SUMO (De Melo et al., 2008) providing very detailed information, including entities (agents and objects), actions, and events which are accessed through a lexical resource.

The DRAMMAR (Lombardo and Pizzo, 2013) refers to external large–scale semantic resources for the description of the common sense knowledge applied YAGO-SUMO. The DRAMMAR ontology is designed to provide an instrument for the conceptual modeling of story facts, and a formal tool for the devise of an annotation scheme for building the metadata of a narrative document. The interface is provided for supporting the manual selection of meaning, extending the vocabulary to a multilingual setting through the lexical database MultiWordNet (Gangemi et al., 2001), to increase the interoperability of the annotation data across languages. The procedural aspects of drama generation are intended as the starting point for specifying, implementing and evaluating practical storytelling systems. The top level of the presented ontology consists of five disjoint classes: Unit, Entity, Dynamics, DescriptionTemplates, and Relation. The action level models the intentional behavior of the characters in a plot by enforcing a BDI (Belief-Desire-Intention) perspective on characters.

Character mental states concern one of the following classes: Belief, Emotion, Value and Goal.

CultureSampo (Hyvönen et al., 2009) provides online access to Finnish cultural heritage through the encoding of the background knowledge in a set of domain ontologies. It supports the connections over the items in terms of geographical and chronological relations, or authorship. The vision and design of CultureSampo goes beyond semantic web portals for cultural heritage which has many aspects and has been made over the years. The thematic perspective consists of several processes including maps search and browse views, relational search, organization, collections, history, skills and cultural narratives, biographies and semantic annotation.

There is also a formal ontology intended to facilitate the integration, mediation and interchange of heterogeneous cultural heritage information created by the International Committee for Documentation (CIDOC) of the International Council of Museums (ICOM). The CIDOC Conceptual Reference Model (CRM) is a high-level ontology to enable information integration for cultural heritage data and their correlation with library and archive information. The CIDOC-CRM (Doerr, 2003) becomes an ISO standard to analyze and develop the common concept behind data and metadata structures for an effective design, in particular when dealing with knowledge from the past in any domain. This ontology is widely used in order to make data connection and plays an important role for defining the composition and operation of historical stories.

The DECHO system (Aliaga et al., 2011) constructs the conceptual model of archaeological domain to support the exploration of cultural heritage objects, including digital images, 3D environments and narratives by applying CIDOC-CRM to establish data connection. The objective is to interact with the 3D virtual environment for accessing the original document about the displayed object.

3.4.2 Event Model

Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) (Gangemi et al., 2002) is an upper level ontology, the first module of a Foundational Ontologies Library being developed within the WonderWeb (Masolo et al., 2002) project. DOLCE is complemented from OntoClean (Guarino and Welty, 2002) with ontology of taxonomic links. Entities have no instances in the sense that its domain of discourse is restricted to them. Properties and relations are usually considered as universal as formally separated from taxonomic links. It is based on a fundamental distinction between enduring and perduring entities. DOLCE is based on a fundamental distinction between enduring and perduring entities related to their behavior in time. Space and time are considered as qualities and quality regions. The basic components are Substantial, Aggregates, Objects, Features, Non-physical substantial, Agentive/Non-agentive and Occurrences. DOLCE improves communication among agents in most cases of information exchange as a reference for agents to commit to certain theories, as a set of formal guideline for domain modeling or a tool for making heterogeneous ontologies.

Due to the expressive limitations of OWL (Web Ontology Language), these semantics are often defined outside the ontology language. The activity patterns ontology implements the descriptions and situations (DnS) ontology pattern of DOLCE Ultralite to formally represent the relationships. The modelling activity classes of domain ontologies as instances drive the derivation of complex activities in terms of the activity types and temporal relations to allow the formal representation of activity interpretation models over activity classes. DOLCE + DnS Ultralite (DUL) (Meditskos et al., 2013) define the Event class next to the disjoint upper classes which are Object, Abstract, and Quality. The Event definition is specific to the DOLCE formal definition as an entity that exists in time. The Object class refers to entities that exist in space such as living thing including non-living and abstract things like social and cognitive entities. Quality is the characteristic of an object or event where the

values represented as a point or area in some Abstract. The Abstract classes represent space value such as space of natural numbers or time.

Event-Model-F (Scherp et al., 2009) is a formal model of events based on the foundational ontology DUL and provides comprehensive support to represent time and space, objects and persons, as well as mereological, causal, and correlative relationships between events. This event model prevents interoperability in distributed event-based systems. Event-Model-F also provides a flexible means for event composition, modeling event causality and event correlation, and representing different interpretations in the same event. Event-Model-F is designed and implemented with the functional requirements aligned to the DUL ontology represented by specialized instantiations of the Descriptions and Situations (DnS). The DnS pattern allows for representing different opinions about events and their participating objects. This feature is not provided by the DOLCE participation relation. For the functional requirements, they introduced specific ontology patterns based on the DnS model including: Participation pattern, Mereology pattern, Causality pattern, Correlation pattern, Documentation pattern and Interpretation pattern. This allows representing arbitrary occurrences in the real world and formally models the different relations and interpretations of events. The full support for the structural aspect as well as different event interpretations distinguishes the Event-Model-F from existing event models.

3.4.3 Interaction Classification

Many VR devices are used to support user interaction (Carrozzino and Bergamasco, 2010) that will be implemented for learning with interactive content (Pinto et al., 2015). Our work aims to advance Interactive VM focus on the development of virtual environments in term of interactive content. The point is many devices and interactions can produce various applications up to the objectives of content. We attempt to define a high-level abstraction for interaction model related to adaptability on different devices. The same interaction tasks should be applied with the same result even using different devices. The designing of interactive content must ensure that interaction is involved in the story as a part of an application. The device capacity should be classified for interaction supporting that will be essentially issued to normalize interaction behavior. Taxonomies of interaction techniques (LaViola Jr et al., 2017) are introduced for task classifications which are selection, manipulation, navigation and system control. We applied interaction tasks and subtasks to organize interaction techniques for various devices. Then we can determine the interaction model on the platform which will be a high-level interaction abstraction.

Furthermore, we need to define a storytelling model cooperating with interaction model. The common language to describe a story in the field of interactive content should organize the structure of object interactivity in general. According to content production and authoring process, there are several tools on the market which are commonly used to create a story in term of virtual environment (Kybartas and Bidarra, 2017). Most of them are restricted to drive the story without interaction (Gaeta et al., 2014) and there is a lack of storytelling instrument for interactive virtual environment contribution. This is a kind of disconnection between virtual environments and interaction organization. We expect a platform to support story making on the adaptive interaction system. To enhance the development of VM application in the context of storytelling and interactive content designing, we propose a platform to be a tool connecting story and interaction together with top-down design. High-level abstraction of storytelling and interaction are defined and then translated to low-level device connection.

3.5 Design and development of a story model

We offer content creation for the VM in a new way that must be consistent with the VM design which can be connected to a variety of devices. Therefore, the design and development of the story model will be different and must include interaction into a story. In storytelling design, the main process is narration of what the characters do and story-based action, but we are introducing a new kind of storyline about interaction with the user. From researches related to conceptual design and narrative modeling, there is also a shortage of interaction between the user and the story in the narrative model. Thus, the novel platform development still has a similar structure as the historical model and storytelling model and will add an interaction model to deal with this issue. A new design and development of story model will adapt the elements of related ontologies and concepts to historical model and storytelling model consistence to the user interaction.

Figure 3.36 shows in a UML diagram the main design and development of our story model to describe these concepts in details. The overall model integrates three main packages: the Historical model, the Storytelling model, and the Interaction model. The Historical model describes historical facts which is the basic knowledge and artefacts reference of a museum to tell a story about a selection of these facts. This leads to the second part, the Storytelling model, to create relation between component in term of VE and interaction. Then this story is accessed through an interactive system by the Interaction model.

3.5.1 Historical Model

The historical model has been developed based on an analysis and abstraction of historical story based on the CIDOC-CRM, the ontology for cultural heritage applications with support of hierarchical relationships. It provides a generic structure for the definition of history facts and is extensible to the requirements in the most different concrete applications and domains. The main classes in the model are the “historical fact” and the “component”. The object of the component class is the elements that may be referred in a story which is created or modified by historical facts (or events). A component is an input or output of events which can be a single element or a group of elements. Components are of two types: actor and artefact. The actor is a representation of a historical human being in the scene, while the artefact is a physical object which is either a building or an environment. Each component may have representation allowing reference to them, which can be documents or other models.

The historical fact is the core process to present an event. It takes a component as input and returns a new modified component as output. There are two types of the historical fact: existence and modification. We applied the concept of existence using CIDOC-CRM in E5:Event. There are subclass of event which are E63:Beginning of Existence and E64:End of Existence, those subclass have E81:Transformation as a subclass. We implemented all subclass together as events of existence for more understanding and easier application in our model. For modification facts, we directly applied the concept of E11:Modification. There are either E12:Production, E80:Part Removal and E79:Part Addition as subclass as same as our model. Here the existence will describe an event of static component in a scene which can be a transition or movement, while the modification describes an event of dynamic component such as group component which are addition or reduction of component. Historical model allows event to be linked to external event by triggers. The trigger is called when the event of historical fact is entered referring to the storytelling model.

3.5.2 Storytelling Model

The Storytelling model is driven by the storyboard with subsequent elements to outline the story and to design scenarios with their situations. The storyboard is described with

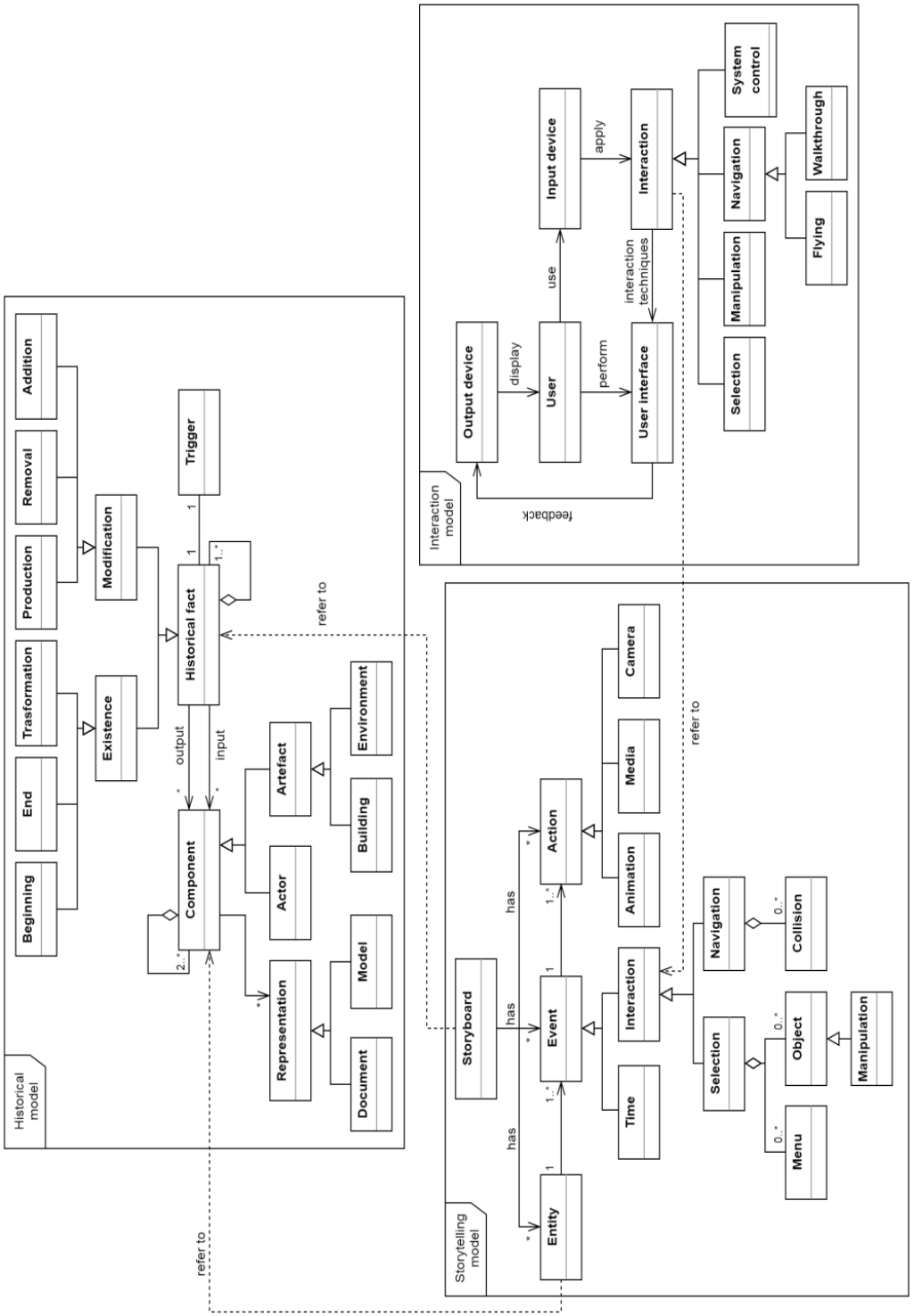

Figure 3.36: UML class diagram of the generic story model

three concepts: entity, event and action. One entity is associated with many events and one event is also associated with many actions. This structure allows many choices of actions to describe situations of the entity. The main part of storytelling model is entity defined as an object in the scene. The entity represents the identity of component in the historical model and the story will be driven through this component.

In order to drive the entity, the event is set up to define a situation for the entity. There are two types of event: time and interaction. Here, we can schedule what event happened by the time or interaction. Event scheduled by time is the condition that occurs at a specific time. By the way, interaction event is the condition to run the event triggered from a user interaction. Three kinds of interaction modes are derived from interaction event: selection, manipulation and navigation (Bowman, 1999). The selection provides choices of interaction by menu or object selection. Object selection is the event trigger when a user select an object in the scene, while menu is triggered when a user interact with a system menu on the display. Manipulation is an extension of object selection to let user interact on object transformation. Navigation is the method to change viewpoint and position in a scene. For example, the interaction event should provide first-person walkthrough mode and collision box to make an event of navigation when the collision box is entered by the user.

The action is the way to present an entity through event condition. It is a static or dynamic story depending on how the events setting or action are displayed. Animation, media and option are the components of action to control the story. The animation is a transition and movement assignment related to the existence and modification in the historical fact. The media is considered as assistance in conveying the story to be fully presented with the detail of information. Moreover, the media is a good response of interaction to offer information by user interaction along the scene. The camera class is associated to the camera scene to allow viewpoint changing by camera transformation. This function enhances the visual organization, supports user attraction and facilitates access to the scene.

3.5.3 Interaction Model

An interactive on-site installation VM is a VM which applies VR technologies to develop an application. Many VR devices are used to support user interaction that will be implemented for learning with interactive content (Carrozzino and Bergamasco, 2010). We need to design a new platform for advance Interactive VM. We focus on the development of VEs in term of interactive content, but the point is that many devices and interactions can produce various applications. We define a high-level abstraction for interaction model flexible for different devices. The same interaction tasks should be applied with the same result even using different devices. The design of interactive content must ensure that interaction is involved in the story as a part of an application. The device capacities should be classified for interaction support that will be essentially issued to normalize interaction behavior. We applied interaction tasks and subtasks to organize interaction techniques for various devices. Then we determine an interaction model for our platform which will be a high-level abstraction of interaction. Furthermore, Model-Driven Engineering (MDE) is the conceptual models for abstract representation (Calvary et al., 2003) that will be useful to implement the platform. We developed the framework and applied MDE to define metamodels, transformations and mappings to address the problem of interaction plasticity and to support multiple devices and context of uses (Coutaz and Calvary, 2012).

We would like to promote a user-centered story driven by interaction. Thus, the user will be the center of the interaction model. Devices are the tools that will be used to receive and display data: they are input or output devices. The input device is able to determine its settings such as buttons or other functions, while the output device will focus on various types of display such as stereoscopic display, HMD display, or CAVE display. User interface is provided to be a communication between user and the scene. The selected device will

apply interaction techniques to perform user interaction with the user interface, and the result of interaction will return to user through output device.

The interaction tasks for VR applications are considered to be selection, manipulation navigation and system control. Obviously, there are other tasks which are specific to an application domain, but these are some basic building blocks that can often be combined to create a more complex task in the story. Selection and manipulation are often used together, but selection may be a stand-alone task. Manipulation occurs after a selection has occurred to manipulate the selected object. Navigation is the common task to explore the scene composed of two tasks: Flying refers camera navigation as a physical movement along the scene without restriction. Walkthrough refers navigation of human being. The avatar is created with physical movement restricted to the ground. The camera attached to the avatar, allows first-person perspective. System control allows the user to send commands to an application to change the system state or the mode of interaction.

Interaction technique is the method to realize a task to apply device control. All devices are based on interaction techniques that will be predefined and each device has different interaction techniques, thus, it is necessary to define the relationship between the device and the interaction techniques before use to allow device change. Figure 3.36 defines our proposal for a complete storytelling and interaction model for VM. Next section discusses the proof of concept we developed.

3.6 Development of the Storytelling platform

We use the previous model to implement a story model to create an operational platform. Historical model is used to transform a history into components of story in term of relation. Storytelling model creates a theme in term of VE where user interaction can be added. The Interaction model is used to determine the use of devices including interactive techniques. The details of the platform development are described to see how these models are applied to create VM application.

3.6.1 Components of storytelling platform

The storytelling platform provides story organization and high-level abstraction to define object behaviors in general for interaction. We develop the platform to assist an interactive content creation which provided devices connection to develop VEs along with interaction setting. The high-level abstract model will be defined into a scene with events and actions corresponding to storytelling needs. When all entities in a scene have been determined, the timeline will be addressed by a runtime engine to perform interactive stories in real time. The translation between the high-level abstraction and low-level connections will automate interaction with the selected device. We provide various assistance tools to design such as the Viewer, the Asset manager, the Event editor and the Timeline. All objects are presented in the 3D Viewer and handled in the Asset manager while defining event and action will be specified in the Event editor. The result of relations in the Event editor could be a sequence of animation observed in the Timeline panel. The details of each component are described here after:

The Viewer is a display window with 3D visualization which provides camera modes, navigation modes and many API features connected to the application. This viewer was developed on the top of the CVE G-SCOP environment. It can be configured seamless to display content on a 2D desktop, within a HMD or in a CAVE. 3D transformation: translation, rotation and scale are also supported to create VEs and cooperate with Asset manager for objects editing. There are two modes that use the same window which are the storytelling edition mode and the runtime mode to execute the result of storytelling mode.

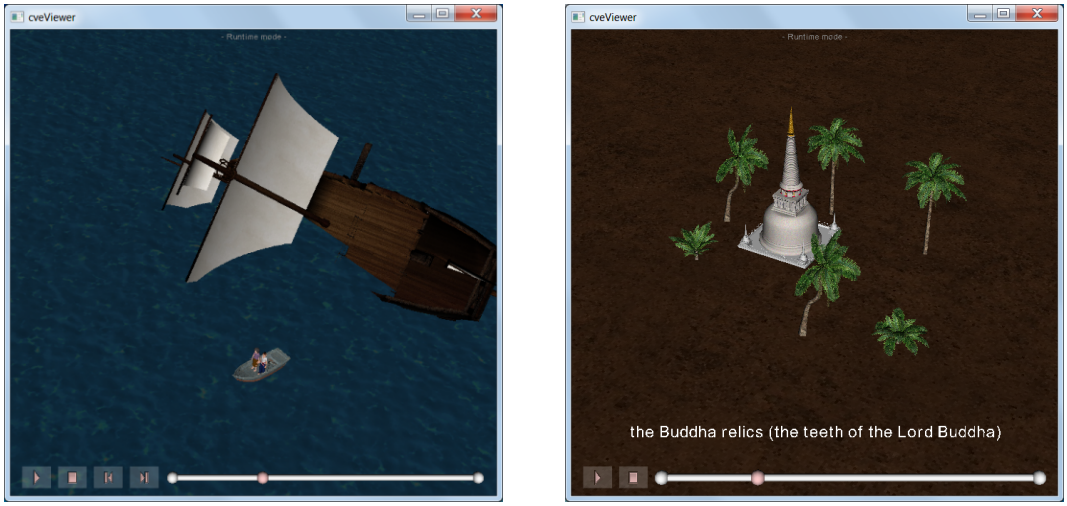

Figure 3.37: The viewer in runtime mode with additional timeline

The Asset manager is a tool panel to carry out all objects in the scene which can be 3D models or terrain. The two components of Asset manager are the list of objects within the scene and the transformation attributes management. The objects are representations of historical facts model. All objects within the scene are represented in the list of objects which can be in the models list or the terrains list. Storytelling platform does not support modeling creation, so all objects in the scene are imported from the historical model. When a model is imported to the scene, the name of the model is presented on the models list as well as a terrain that will present the terrain name in the terrains list when the terrain is created. Both model and terrain can be selected to modify their transformation values through the transformation attributes management and the result of transformation is displayed on the Viewer. In the models list, we provide additional functions such as Create_group, Create_empty, Show_all and Hide_all to more easily manage scene components. For each imported model there will be a command such as Edit_name, Hide, Show and Delete to control the model appearance. Moreover, it can be added an icon by the function Add_icon. When the icon is added the icon name will be presented under the model name on the models list. The icon plays a role of avatar in the VR scene. The asset manager is an original development for this study.

The Event editor is the tool for defining high-level abstraction events. It provides a common template for every story. There are two parts which are the components list on the left hand side and the canvas on the right hand side of of Figure 3.40. Story structure can be created by dragging the name from components list and then drop on the canvas. In the Figure 3.40, a created node follows the selected name of the list which is an entity, an event or an action. Parameters and setting are adjusted directly by double clicking on the node on the graph canvas. The relations of links between entity, event and action will be interpreted to make a story.

- Entity is a list of imported 3D model except Camera which will be available from the beginning to control the camera. Many models can be grouped together as a group of entities to easily control, a group entity is considered as one entity. Moreover, some models may have additional components called icons, which are added from the Asset manager. These icons are considered to be an entity that is created as a node to define event and action as well.
- Event is the list of all events which trigger story evolutions. When entering the specified

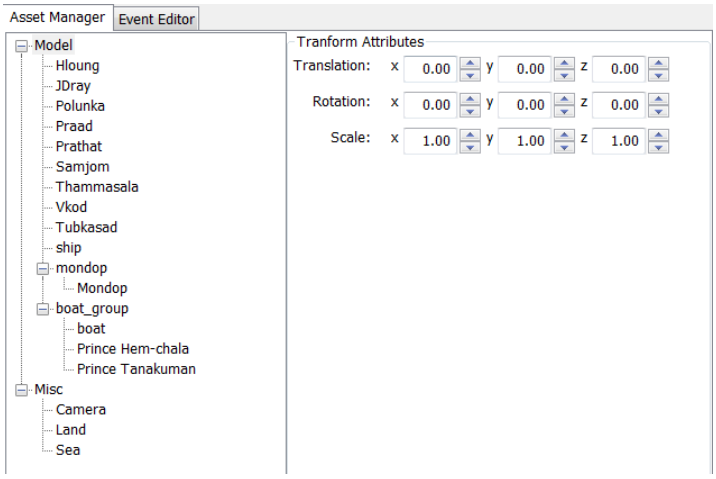

Figure 3.38: Asset manager appearance and transformation attributes editor

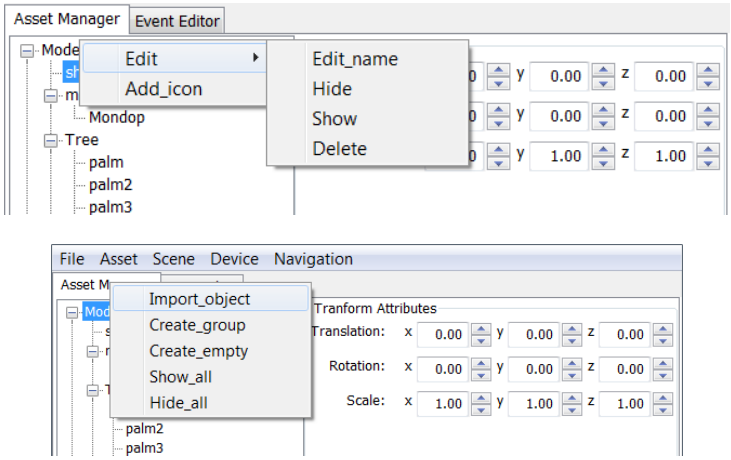

Figure 3.39: The object commands on the Asset manager

conditions, it will execute the connected action. Events and conditions are:

- Timing: is a condition of time; there are Start and Stop value to define this event. When the specified start time is reached, the connected action will be launched and will end when the stop time is reached.
- Collision: is a condition of box collider which is triggered when the user's avatar get inside a given box. This event is a condition of the first-person navigation, which is checked by the position of the user's avatar.
- Selection: is a condition of interaction when the assigned object (entity) is selected then the assigned action is executed. This event is concerned about device connection to determine interaction techniques which is already defined and is selected from the device menu.

• Action is a list of actions that are used to create a story, including animation making, multimedia control, user interaction, and camera control.

- Transition: is to set the display pattern of the object to achieve the order of entry and exit. Fade-in and fade-out parameters are available to control transition.

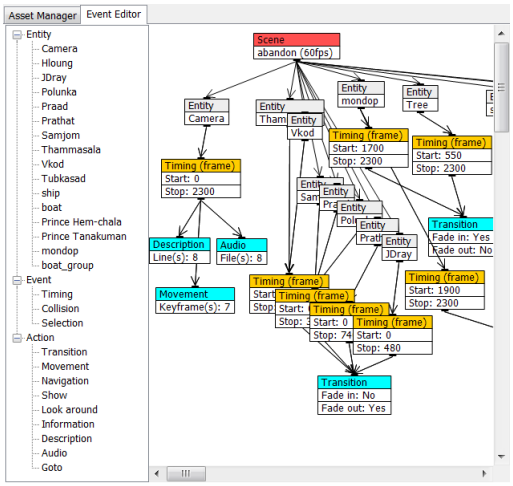

Figure 3.40: Event editor appearances with structure of linked nodes on canvas

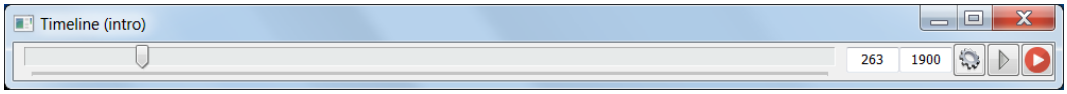

Figure 3.41: The appearance of the Timeline panel

- Movement: is an animation of the object from one position to another, which is defined as a keyframe with potential multiple keyframes for completed motion. There is a separate keyframe window to easily manage the details of the moving position.
- Navigation: allows to enter the first-person walkthrough mode, which will set the default location Start (x, y, z) for the avatar.
- Show: displays the camera viewpoint for showing only the selected object and hiding the rest of the objects in the scene. The distance, up and angle are parameters to adjust the camera viewpoint as same as Look-around.
- Look-around: is a visual representation of objects by rotating the camera viewpoint to make the object interesting. There are variables: distance, up and angle to adjust the camera viewpoint as needed.
- Information: is a pop-up window to display the information details, including image and description.
- Description: is a line caption that can be put whether at the top or bottom of the display. This action provides the window to manage the captions and the times setting to be displayed in sequence as desired.
- Audio: is used for defining the sequence of soundtracks. This action also provides the window to manage audio file and time setting.
- Goto: is the action to jump to another scene while the current scene is running. In general, there is a Multiple scenes command in the menu to play multiple scenes respectively. However, when we need to jump to another scene while the current scene is not over, this action stops the current scene and the new scene starts.

The Timeline schedules some events when the structure of linked nodes is interpreted and the story is stored on the timeline. In the storytelling mode we can use the Timeline window

for animation checking. In the runtime mode, we create new virtual timeline into the scene to move all interactions including scene controlling on the Viewer. A developer can interact only in the Viewer with the timeline. The Timeline interaction allows:

- To compile: for interpreting structure of linked nodes from the Event editor to frame by frame animation and keep it on the timeline.
- To play: when the compilation is completed a recorded animation is set on the timeline, the story results in playing the animation. The play button automatically plays the frame rate that was set from the scene node in the Event editor.
- To run: is used to change the mode of operation to runtime mode. User interaction can be performed on the Viewer when entering this mode and a new timeline will be created on the scene.
- To move the slider: is used to scroll through the animation, which let jump to a desired frame as required.

The Menu is a command bar that categorizes the same form of command to make the platform easy to use. The menu bar contains buttons to control the storytelling platform including File, Asset, Scene, Device and Navigation.

3.6.2 Storytelling platform usage

The storytelling platform is divided into two parts for story design and execution by using the same display on the Viewer. There are two modes to control objects in a scene which are storytelling mode and runtime mode.

In Storytelling mode we prepare, define animation and interaction of each object separately. The Viewer and Event editor will be used together to define the story using a graph association of entity, event, and action. Firstly, all entities must be imported to the scene, and then we use the Event editor to assign event and action to the entity. Transformation setting of each entity is organized on the Viewer linking nodes on the Event editor. In this mode, when an object is selected in the Viewer, it changes the color of the model to present active object and it displays current value of transformation attributes. The object is selected directly from the Asset manager list, which will display the same selection as the selected objects in the Viewer. The Viewer in the storytelling mode is responsible for displaying the orientation of elements within the scene, which acts in conjunction with the Asset manager to adjust the transformation of the object. It also helps to see the results of the animation when it is used together with the Timeline slider.

In Runtime mode if entities, events and actions have been linked correctly from graph nodes in Event editor, the animation and interaction of all objects are executed into the timeline. The timeline registers the storytelling result frame by frame to present the story as a video playing that user can play, pause or even approach to interact with any object in the scene at any time. When entering runtime mode, the Viewer window changes to the animation display and interacts with the user, without the object selection to adjust the transformation attribute value anymore. The object selection triggers an interaction with that object instead. Runtime mode has an event listener for each frame. When the event is triggered, the object (entity) will run an action or jump to the next frame. User interaction: selection, manipulation and navigation are connected to the event trigger. Thus, the process of runtime mode is looping synchronization of the event listener and user interaction then act on the object if event is triggered. We exit the runtime mode by pressing the stop button (when entering runtime mode, the run button will change to stop button), then return to storytelling mode to edit the story again.

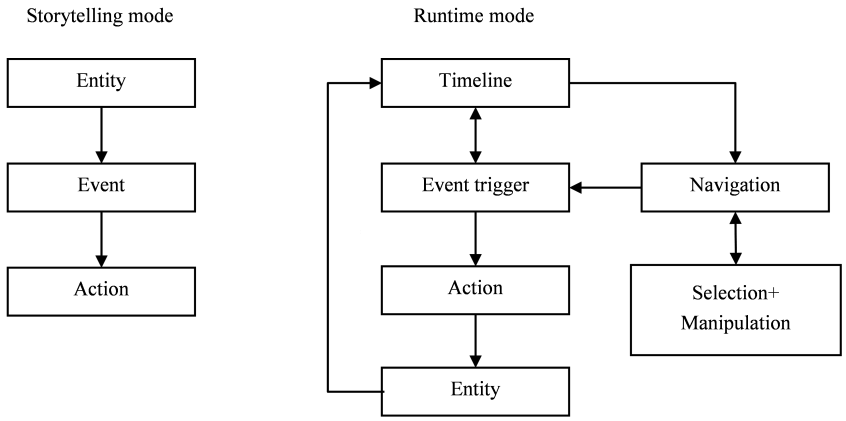

Figure 3.42: Working process on storytelling mode and runtime mode

3.6.3 Story creation

Storytelling model has three components: entity, event and action which are introduced to be a high-level abstraction of the story. All components are the graph nodes linked in the Event editor. The entity represents imported 3D model of each object in the scene which is the main component to assign event and action. The event is the component to control entity: when event is triggered a related action is executed. Every part of story is defined in term of entity-event-action to determine animation and interaction in the scene.

In order to make a story with the Event editor we can link each node as graph relation as a tree structure. At the beginning there will be a root node on the canvas. We can name it as a scene name and set the frame rate here. The scene node is always on the top of the tree.

As a top-down structure design, scene node is connected only to entity nodes. The entity nodes are connected to the event nodes and the event nodes are connected to the action nodes. Therefore, each relation will start at the entity node and end at the action node. However, our tree structure has flexibilities to set each node which allows an infinite variety of situations in the story. There will be a setting window to set various values by double clicking on the node. If all nodes on Event editor have been connected correctly the story result will be parsed into the timeline seen in sequential sorted frames.

3.6.4 Interaction system

The interactive content that we already have created can apply the same story on another device without any structure or programming modification. Device and interaction techniques are defined before the story is created, we can select the device in a menu and thus any device can be used without affecting the story. When compiled, the selected device is used as an agent for interaction with the system. However, we are able to switch to another device while the structure of the story remains the same and re-compile again to confirm the device is enabled. Our platform allows changing device or interaction techniques applying interaction model as defined at high abstraction level when devices are installed. Event editor is the combination of methodology between storytelling model and interaction model which is the core component of the platform to create interactive content with the adaptive interaction system.

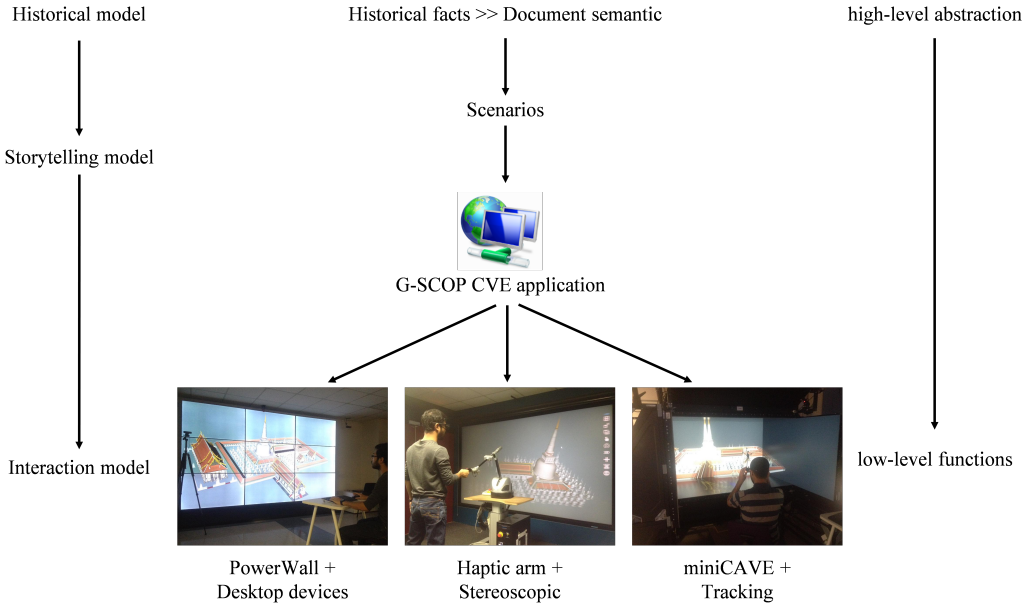

Figure 3.43: Transformation of storytelling model to low-level device connection

3.6.5 Story transformation

The process of model transformation begins with the device installation and interaction behaviors of each device. We keep all interaction definitions on the CVE application[40] which are retrieved when the interaction model is transformed in term of the low-level device connection. As a top-down design workflow, high-level abstraction is decomposed into a lower level for physical implementation. The storytelling model related of high-level abstraction through the definition of entities, events and actions allowing a developer to create a story description by graph nodes connection. Interaction model will be used through event node on Event editor to define interaction behavior along the scene. Here, storytelling platform enables device and interaction change. If all nodes connection is correctly linked, then the storytelling model is executed as an animation into the timeline while the interaction model is executed for every specific object. The result of execution of the animation is seen on the Timeline panel while the result of interaction is seen at runtime mode and only depends on the defined situation. On runtime mode, specification of interaction techniques will be transferred into low-level functions applied by selected devices. When the interaction event is triggered the action will be launched as defined. Figure 3.43 explains how high-level abstraction is transformed into a VE that can be deployed on various virtual device set.

3.7 Conclusions

This chapter proposed the Storytelling platform to support digital interactive content for application of on-site installation VM. Storytelling model and interaction model are introduced to be a high- level abstraction of storytelling in term of VEs. The platform provided the tool to organize the story and interaction which are the Viewer, Asset manager, Event editor and Timeline. The main process to generate the story abstraction is in Event editor combined storytelling and interaction model together which has device changing abilities. Device connection and interaction techniques are handled on CVE application related to the story execution on Event editor. Model transformation allows device and interaction

[40]CVE G-SCOP application

techniques change while the story structure remains unchanged. Developers are able to apply their interactive content to various devices without much structure or programming modification.

By the way, we develop storytelling ontology useful for modelling the specification of expected interactions for VM visitors. It is assumed that visitors will get more understanding about digital heritage contents by storytelling model. Storytelling can greatly improve the access to the information and knowledge stored in museums appropriately linking information according to the user preferred interaction metaphors (Uijlings, 2006). For this purpose semantic annotation, a mental model and user interests are expected. However, ontology for storytelling needs to improve and adapt some concepts for using in VM.

The overall proposal has been converted in an operational proof of concept. We then used this platform to apply it on specific use-case. Next chapter demonstrate such a complete use-case.

Chapter 4

Development of a VM Case Study

This chapter describes the creation process of an on-site installation VM, which may use several devices for user interaction. The case study is the model of Wat Phra Mahathat Woramahawihan, a temple in the south of Thailand, which is considered as a World Heritage Site. Thus, this is a great opportunity to create an on-site installation VM using VR technology for interactive learning. A story in term of virtual environment is developed on the Storytelling platform connected to the interaction system. Different interaction systems are set up with immersive devices from G-SCOP laboratory to study how different interaction systems affect learning.

4.1 Interaction system preparation

The first step describes technical devices that will be used to build the interactive system. They are divided into two types: the output device and the input device. Then the interaction technique will be defined as a method to interact with each system.

4.1.1 Output device

Output devices are the different display environment used, such as the UHD Powerwall, the stereoscopic display and the CAVE from G-SCOP laboratory.

4.1.1.1 Powerwall

The Powerwall is a special multi-monitor setup that consists of multiple monitors tiled together contiguously or overlapping in order to form one large screen. We use the Powerwall instead of a single large screen to get tile layouts, to get a greater screen area per length, and greater pixel density per area. The abilities to display the large size image with ultra-high resolution face the low resolution of a single monitor which seldom supports 6K images or videos. Continuous technological advancements enhanced performance and flexibility of the Powerwall for different applications that need visualization beyond the traditional display, especially, to support user perception through high resolution and high angle of view. This wide display is coming in many exhibition centres and are good technology candidate for VM.

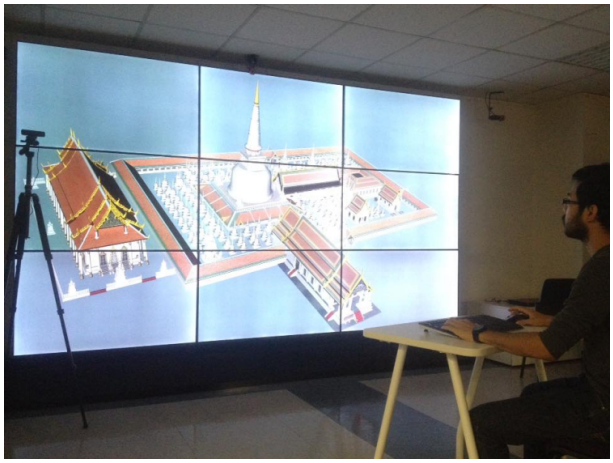

Figure 4.1: The Powerwall with 3x3 tiled displays

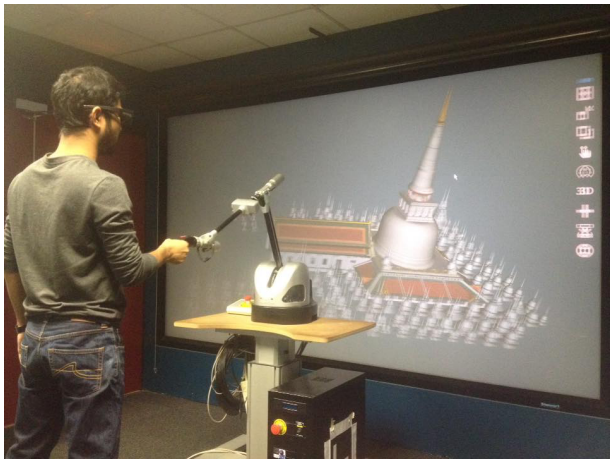

Figure 4.2: The stereoscopic display with shutter glass

4.1.1.2 Stereoscopic display

The stereoscopic display provides a perception of depth to the user by means of binocular vision. An active or passive shutter 3D system is used to display stereoscopic images. It works by presenting only the image intended for the left eye while blocking the right view of the eye, then presenting the right eye image while blocking the left eye. A 120 Hz repetition of this process does not interfere with the fusion of the two images into a single 3D image.

4.1.1.3 MIHRIAD miniCAVE

A CAVE (Cave Automatic Virtual Environment) is a fully immersion environment that usually occupies a room or place where up to six projectors create a complete immersive cube box. The MIHRIAD miniCAVE is such an immersive virtual reality platform at G-SCOP laboratory with five stereoscopic projections assembled as an open box. The user wears a stereoscopic shutter glasses inside the miniCAVE to see 3D graphics generated and synchronized with the stereoscopic projectors so that each eye only sees the correct image as left and right eye gaze. User's movements are tracked by a motion capture system which records the real time position of the 3D glasses tracking the user gaze. The computer generates a pair of images, one for each of the user's eyes, based on the motion capture data. The user perceives objects apparently floating in the air getting a proper view of what

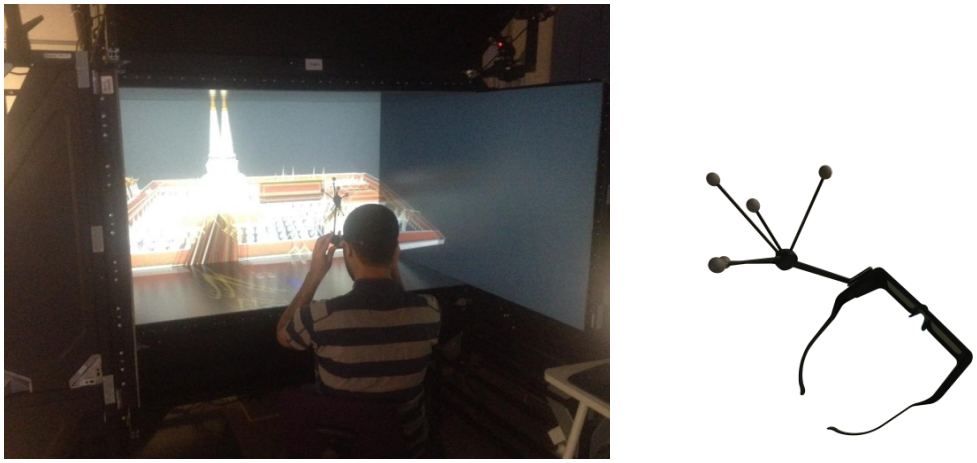

Figure 4.3: MIHRIAD miniCAVE and a shutter glasses with tracking markers

it looks like in reality.

4.1.1.4 Software display control

G-SCOP CVE was developed to support various displays configuration with the process to generate immersive visualization such as 2D display, 3D stereoscopic display and CAVE display. That means our application can provide the options to visualize the content for whatever required display device. Hence, all types of interaction system can be combined with the 3 visualization output devices.

4.1.2 Input device

An input device receives data from the user to be processed and converted to interact with the system, such as desktop device, haptic arm, 3D pointer and HMD controller.

4.1.2.1 Desktop device with mouse and keyboard

We assign the keyboard as an input device and mouse as a pointing device. The keyboard is used to get button-based input and then send the data to execute the command to the application. The mouse allows user to input spatial data into the application. Movements of the pointing device are transformed to the screen by movements of the pointer creating a intuitive way to navigate on a 2D GUI (Trackball like behavior).

4.1.2.2 Haptic arm

The Virtuose 6D is a 6DOF haptic device designed for VR applications. It enables a scale one interaction with digital models coming from virtual environments with a large workspace and its force feedback. The Virtuose 6D is composed of two main articulated components: fixed on a rotating base and articulated wrist. The haptic interface is a 6DOF device, with force-feedback in all directions. The user holds the haptic device using a gripper at the end of handle placed. The gripping tool is equipped with three push-buttons. One of the push-buttons is dedicated to the offset function for wider 3D space control. The state of the other buttons can be accessed using the Virtuose API.

Figure 4.4: The Virtuose 6D haptic device

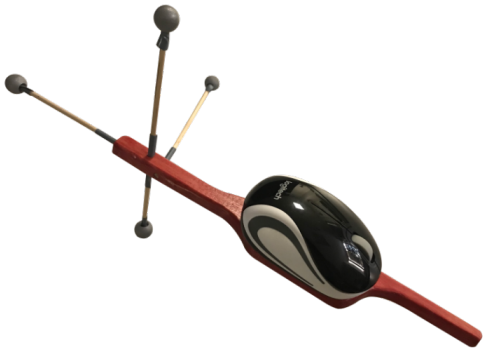

Figure 4.5: The G-SCOP 3D pointer for miniCAVE

4.1.2.3 3D wand

A wand was designed at G-SCOP laboratory to be a pointer and selector on a 3D GUI which composes a handle, a small mouse and markers for 3D tracking. The markers attached on it to be tracked in 3D which allows spatial 3D pointing when the wand is moved. The small mouse is used to be a selector with left, right and middle clicking and also scrolling. It disables movement on the surface and attached to the handle stick instead to enable spatial 3D pointing. This device is especially set up to operate with the CAVE because the CAVE size is handly compatible with usual commercial wand, however, it can apply to other interaction system working on 3D environment. It works as a usual wand system, but it is smaller and thus is adapted to the G-SCOP miniCAVE.

4.1.3 Design of the interaction systems

We set up three types of interaction systems with distinct user interface depending on the display device. The interactors for each system are designed to support interaction techniques implementation.

The 2D interaction system is set up with the Powerwall and on a desktop device. The user interface uses point and click and has menus on the top left. The standard interactor is a pointer.

The 3D interaction system is set up on the stereoscopic projection with the Virtuose 6D haptic arm. The user interface uses 3D environment and has menus on the top left. The standard interactor is a 3D ball avatar, representing the hand position.

The CAVE interaction system is set up on MIHRIAD miniCAVE with the 3D wand. The user interface uses 3D environment and has menus on the bottom space. The standard interactor is a 3D stick avatar which is aligned with the wand pointer.

4.2 The design of interaction techniques

In Chapter 2, we presented information about input and output device technology that enables interaction. However, choosing or designing good interaction devices is not sufficient to produce a good user experience (LaViola Jr et al., 2017). This section clarifies interaction techniques for interactive tasks. We remind that interaction techniques are methods used to accomplish a given task via the user interface depending on the devices of interaction system. Interaction system expects to set up hardware for application. Interaction techniques define the functions that will be applied to interact in the application.

The interaction techniques must be compatible with every interaction system and thus with the different devices. In a first step the interaction technique is defined a priori with the model proposed in Chapter 3. It is easy to manage and change the interaction system over time. We divided the interaction techniques into three main categories: selection, manipulation, and navigation. These categories as used in our use case are detailed in the next subsections.

4.2.1 Selection and Manipulation

To enable selection and manipulation techniques, we refine selection and manipulation into basic tasks achieved in all the interaction workspaces. This section develops a possible set of canonical manipulation tasks designed to a specific interaction technique for a small set, which can be deduced from all areas of manipulation activities as proposed by (Argelaguet and Andujar, 2013).

Selection and manipulation can be an interaction on one or more objects, but this work focuses on individual object for supporting user learning on each object instead of group of objects at the same time. Thus, we assign the following tasks as a basic manipulation.

Selection acquires or identifies a specific object from the entire set of objects available in the scene and sometimes it is called a target acquisition task (Zhai and Milgram, 1993). The selection task is either pointing to an object or indicating an object or picking up an object.

Positioning is the method to change the 3D position of an object. The positioning task moves an object from its initial position to a target location, and refers to translation.

Rotating changes the orientation angle of an object. The rotation task rotates an object from its initial orientation to a target orientation. In some case positioning and rotating can be merged into a single gesture, but some techniques may lead to separated actions.

Scaling changes the size of an object. The scaling task enlarges or reduces an object size from its initial size to a target size.

Each technique was designed respect to each device. Table 4.1 describes the details of selection and manipulation technique for the used interaction system.

Table 4.1: Selection and manipulation tasks for each interaction system

Interaction	2D	3D	CAVE
Selection	Lclick	Lclick+ 3D ball collision	Lclick+ 3D wand collision
Positioning	Lclick+ dragging	Selection+ translation	Selection+ translation
Rotating	Lclick+ dragging	Selection+ rotation	Selection+ rotation
Scaling	Rclick+ dragging	Rclick+ 3D ball collision+ dragging	Rclick+ 3D wand collision+ dragging

4.2.2 Navigation

Navigation is a fundamental task for movement in virtual environment including both travel and wayfinding.

4.2.2.1 Travel

Travel is a motor component of the low-level navigation system for user control of his position and direction (Bowman et al., 1999). In the virtual world, the travel technique allows the user to translate, rotate the camera viewpoint and modify the conditions of movement such as the speed (Von Kapri et al., 2011). Depending on the system, the camera or the whole scene is moved respect to a fix observer.

Another type of hand-based object manipulation metaphors using manipulation-based travel technique (Bowman et al., 2001b) is proposed to manipulate either the viewpoint or the entire world. It will be used in situations where both travel and object manipulation tasks are frequent and interspersed. In a standard size CAVE the observer is moving.

In the MIRHIAD CAVE, the scene is moved and the camera reflects head tracking only. It is a matter of relative position of the camera respect to the scene. This task involves object manipulation tasks to move virtual objects and frequent travel tasks to see the virtual world from different viewpoints. If the same metaphor can be used for travel and object manipulation, then the interaction with this environment will be smooth and simple from a user perspective.

In this work we provide two types of travel navigation which are camera free and walk-through. Camera free is a free camera viewpoint control, while walkthrough is a first-person movement in the virtual environment. Both camera free and walkthrough mode has translation and rotation for navigation engine. On camera free mode, translation can move freely within six directions: forward, backward, left, right, up and down. While up and down are skipped in the walkthrough mode because the avatar is hold on the ground. The rotation of both camera free and walkthrough are defined only yaw and pitch. The roll axis is skipped due to the effect may be confusing. Table 4.2 explains the details of travel technique on different interaction system.

4.2.2.2 Wayfinding

Wayfinding is an advanced cognitive component of high-level navigation system to plan and decide user movements. It involves spatial understanding and task planning to determine the current position in the routing environment from the current location to the target location. It expects a mental map of the environment (Bacim et al., 2009). At a storytelling

Table 4.2: Navigation by travel tasks for each interaction system

		Navigation	2D	3D	CAVE
Camera free	Translation	Forward Backward	Mouse FBscrolling	Lclick+ FBtranslation	Lclick+ FBtranslation
		Left Right	Mclick+ LRdragging	Lclick+ LRtranslation	Lclick+ LRtranslation
		Up Down	Mclick+ FBdragging	Lclick+ UDtranslation	Lclick+ UDtranslation
	Rotation	Pitch	Lclick+ FBdragging	Lclick+ UDrotation	Look up Look down
		Yaw	Lclick+ LRdragging	Lclick+ LRrotation	Turn left Turn right
Walkthrough	Translation	Forward Backward	Keyboard FB buttons	Push and pull gripping tool	Lclick Rclick
		Left Right	Keyboard LR buttons	LRtranslation of gripping tool	FBscrolling
	Rotation	Pitch	Mouse FBdragging	UDrotation of gripping tool	Look up Look down
		Yaw	Mouse LRdragging	LRrotation of gripping tool	Turn left Turn right

level, there is no interaction definition, thus, no way to specify wayfinding directly. It must be designed at the event editor level. A menu to jump user avatar into specific location in a scene is an example of using wayfinding technique that means all systems have the same technique organized in the event editor.

4.3 Device switching

In general, when developing a VR and also VM application, the devices and interaction techniques used are defined as a first step of the development process. To change devices or to apply new interaction techniques, to change the software code, as well as to determine new organization corresponding to the new update, modification is needed in some parts of previous development process. For this reason, a VM application development is based on the device and cannot be adjusted easily without modification when new devices are replaced. Here the storytelling platform is developed to study the learning outcomes of different interaction systems and to evaluate the learning performance of each system to find the optimal interaction system that will be used to create a VM application. Thus, the storytelling platform is designed to accommodate different interaction systems and interaction techniques. To change the device should be simple as selecting it like it is illustrated in a menu (see Figure 4.6).

In the Storytelling platform a toolbar provides the interaction system button to determine the devices that will be used to interact with the system such as the 2D interaction system, 3D interaction system, CAVE interaction system and HMD interaction system. All interaction systems are set up as designed in the section 4.1 and each system determines interaction techniques as defined in the Section 4.2 (Table 4.1 and 4.1 stand for the overall configuration of device behavior). The interactive content that we already have created should apply the

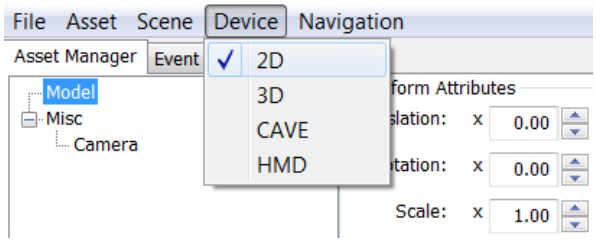

Figure 4.6: The Toolbar menus for interaction system changing

same story connected to any interaction system without structure or program modification. Our platform allows changing the interaction system with pre-defined interaction techniques applying interaction model as defined in a high-level abstraction configuration when devices are installed. Event editor is the combination of methodology between storytelling model and interaction model which is the core component of the platform to create interactive content with the adaptive interaction system. When the interaction event is triggered the action will be launched with the same result as defined in the storytelling model and for any selected system. Storytelling becomes technology independent.

4.4 Events and actions assignment

The storytelling model is applied in the event editor with three connection parts: entity, event and action, as described in the Section 3.6.3. The action determines how the entity will be edited while the event triggers the actions.

4.4.1 Events assignment

There are two types of events which are timing event and interaction event. If timing events are defined within the scene, then the story is a linear story. By the way, if interaction events are defined within the scene, then the story is a non-linear story. Actions are based on user interaction, which allows the user to participate in the story presentation. Therefore, the interaction system affects the interaction between the user and the virtual environment. The interaction model is responsible for managing the interaction of different interaction systems. Thus, we can define event and action independently and can use any type of interaction system as described in the Section 4.3.

In order to assign events for each entity, we provide parameters and constraints to handle specification of event as shown in the Figure 4.7. Timing event has two parameters: start and stop to define action starting frame and ending frame. Interaction event has input parameter depending of its nature, but always has the same output parameters to execute an action. Selection has the interactor parameter to be an input listener which can be 3D entity or an icon of any avatar in the scene. The selection event will launch the action when the interactor is selected. Collision has the box collider as input to the listener. The box can be selected from box collider list created in the asset manager panel as described in the Section 3.6 and the collision setting of box collider is shown in the Figure 4.8. The collision event will launch the action when navigation mode is walkthrough and when the box is hit by the user avatar. Menu has a button presented in the user interface created by naming in the menu settings. The result of menu event is shown in the Figure 4.9. The menu event will launch the action when the menu button is selected.

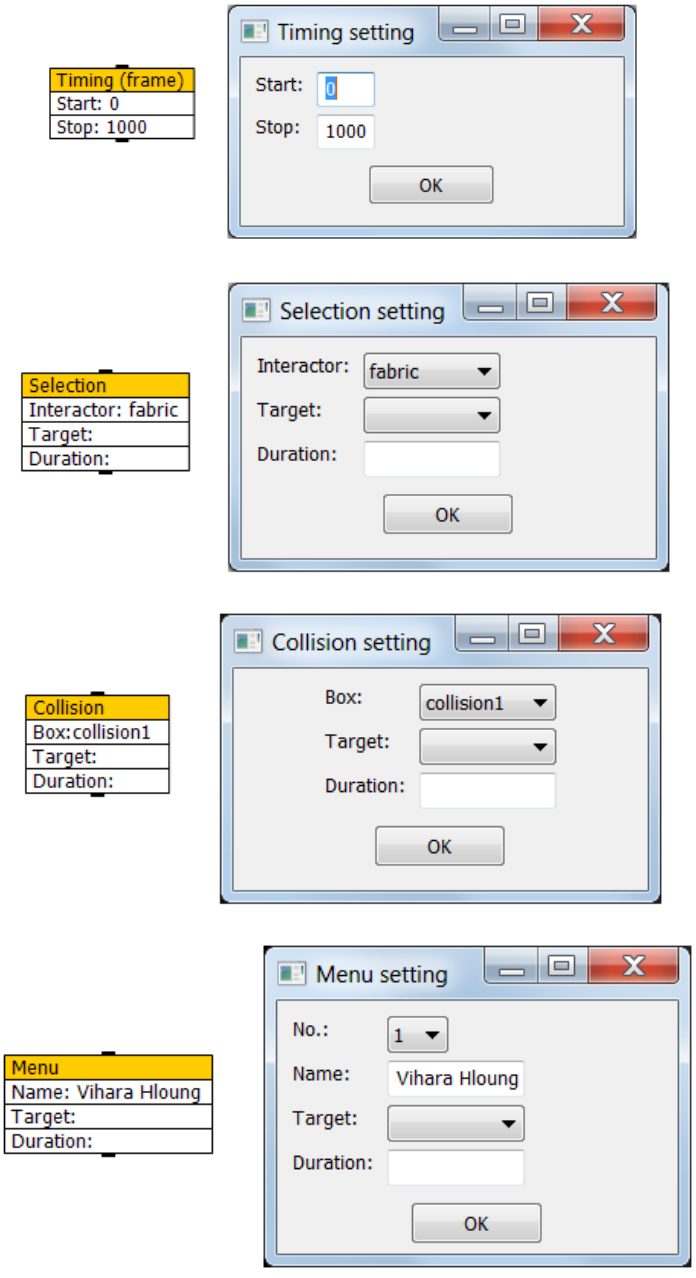

Figure 4.7: All event nodes appearance in the event editor with their parameters setting

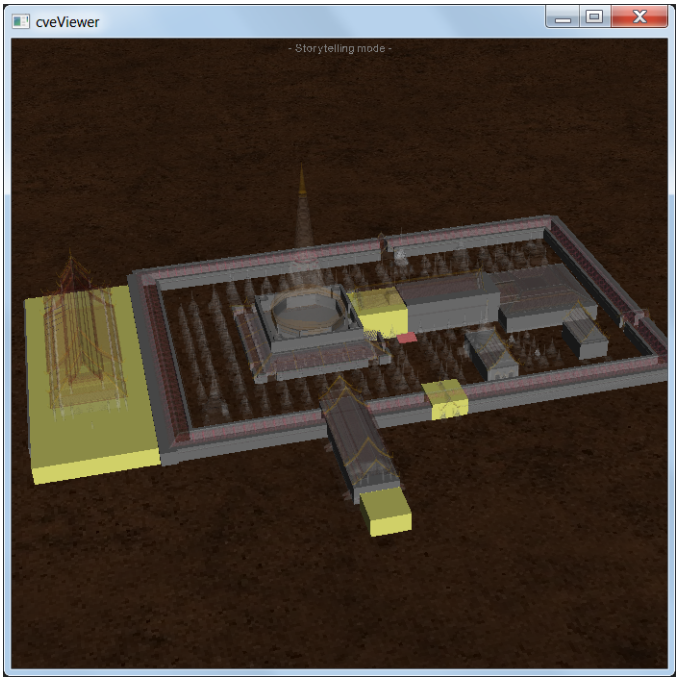

Figure 4.8: The collision setting in asset manager provides collision lists to use with the collision event

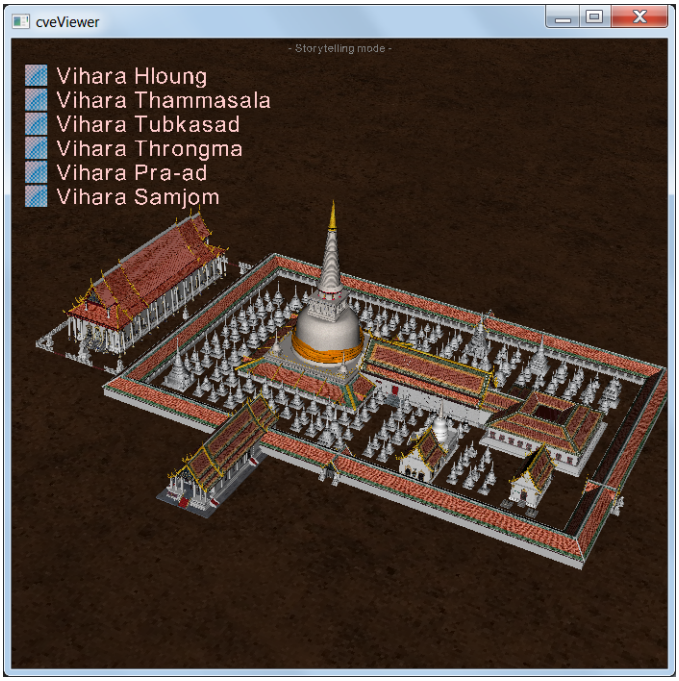

Figure 4.9: The result of menu event will show in the viewer and launch the action when selected

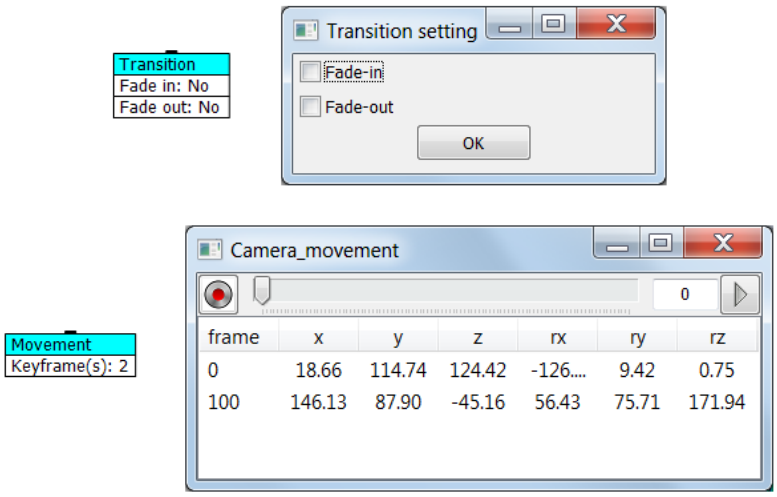

Figure 4.10: Action nodes appearance for animation assignment with their parameters setting

4.4.2 Actions assignment

Actions in an event editor are divided into four groups: animation assignment, media assignment, camera action and optional action. Each group of action has different options to create interactive story.

4.4.2.1 Animation assignment

Animation has two actions to define whether transition or movement is implemented. Transition is used to display the entity, with fade-in and fade-out functions. Movement is used to move an entity. Keyframes are defined to determine where entity is moved. It moves continuously from the previously assigned keyframe. Figure 4.10 shows the appearance of both animation assignments.

4.4.2.2 Media assignment

Media may be information, a description, a caption or an audio media, their appearance windows as shown in the Figure 4.11. Information leads to a window presentation to display the content that comprises either image or text. There are buttons on the window to show next or previous content. Thus, various contents may be stored into one information action as shown in the Figure 4.12. Description is a narrative subtitle that allows conservator to record the display time. The subtitles can be played continuously at any given time. Caption is a text box that shows the position of an object in the virtual environment. This allows conservator to put the caption following the object position in the scene. The result of description and caption are shown in the Figure 4.13. Audio is a sequential play of audio files, allowing conservator to set voice for a specific time and also can be a sound effect up to specific event.

4.4.2.3 Camera action

Camera action is an action to control the camera behaviors. It is either a show or look-around. Show is an action to present a single entity. It cuts off all the components inside the entire scene so that the object is presented alone. All parameters are defined to adjust camera viewpoint such as distance, high and angle of camera. The show action will allow the user to manipulate the displayed object for interaction. Look-around is a 360-degree

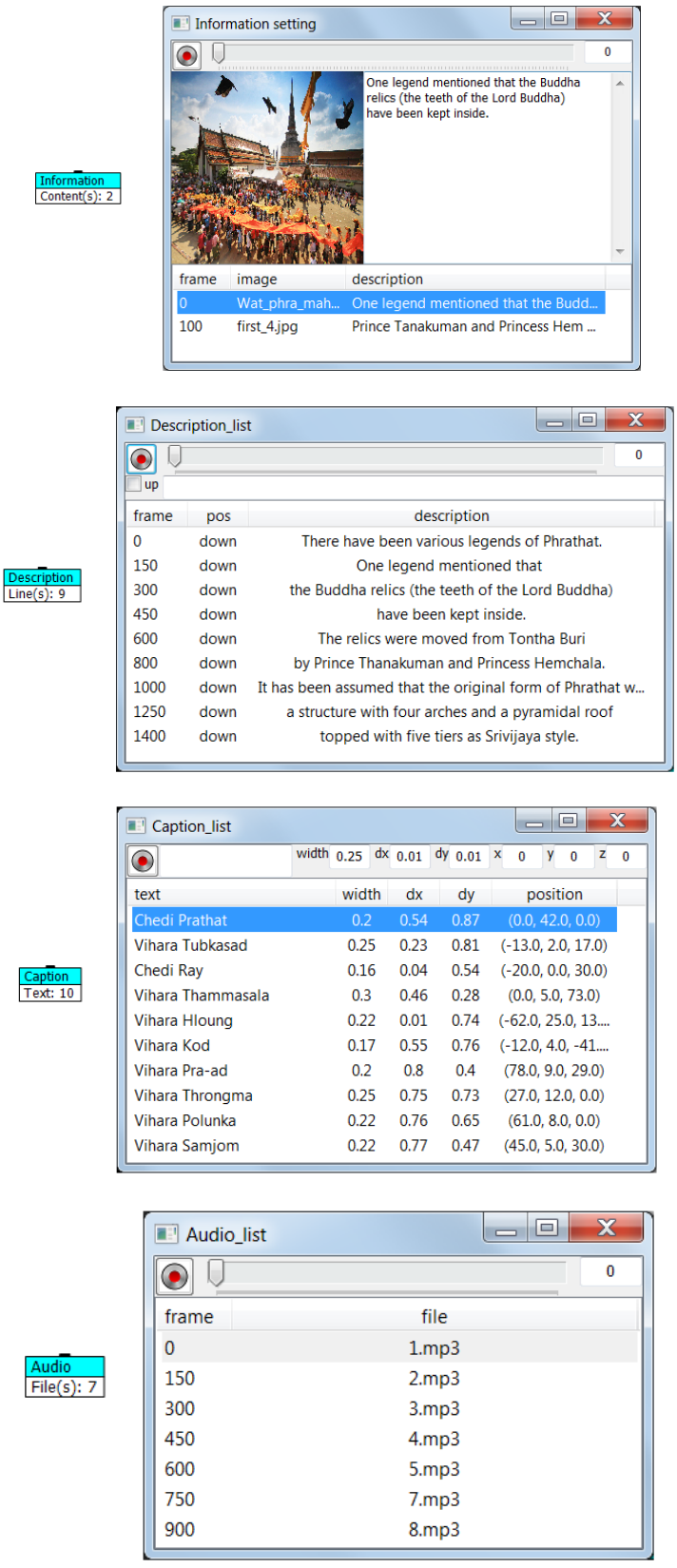

Figure 4.11: Action nodes appearance for media assignment with their parameters setting

Figure 4.12: The appearance of information window presents content as defined

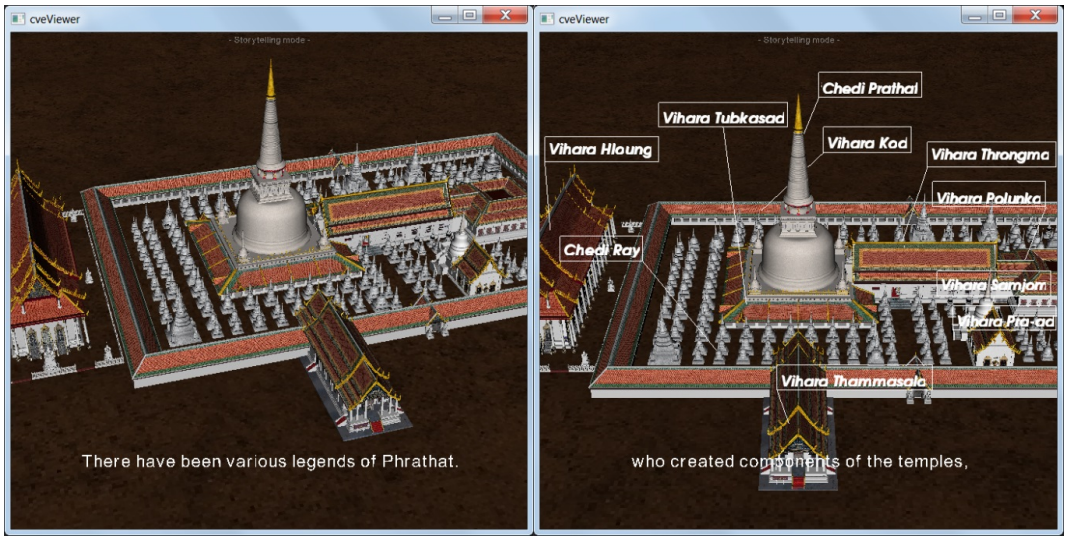

Figure 4.13: The appearance of description and caption in the scene

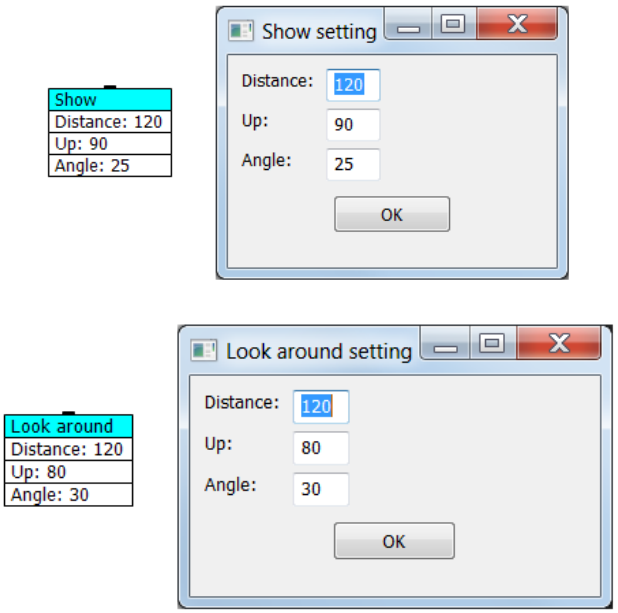

Figure 4.14: Action nodes appearance for camera action with their parameters setting

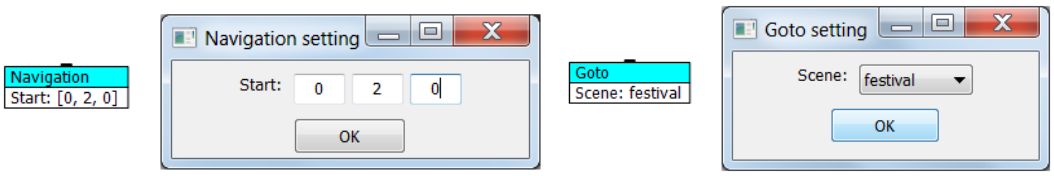

Figure 4.15: Action nodes appearance for optional action with their parameters setting

rotation of the object to see the detail of the object around it. There are parameters to set the camera viewpoint like the show action. Figure 4.14 shows appearance of two camera actions.

4.4.2.4 Optional action

Optional action supports scene organization. It comes up with two distinct options which are navigation and goto as shown in Figure 4.15. Navigation is an action to allow walkthrough mode with specific position that can be a warp portal for user. Goto is an action to support scene change when user triggers an event.

It creates a storyboard with a variety of stories triggered by interaction. For a storyboard creation within the event editor, there are four events: timing, selection, collision and menu and there are ten actions: transition, movement, information, description, caption, audio, show, look-around, navigation, and goto. This will allow each entity to create 40 different types of actions associated to events as details in the Table 4.3.

Table 4.3: Different types of results by combining the events and actions

		Events			
	Actions	**Timing** • Start • Stop	**Selection** • Interactor • Target • Duration	**Collision** • Box colider • Target • Duration	**Menu** • Menu name • Target • Duration
Animation assignment	Transition • Fade-in • Fade-out	Show the entity (fade-in) from the start frame until the stop frame then hide the entity (fade-out)	When the interactor is selected then show/hide the target (with fading as setting)	When user avatar hits the box then show/hide the target and when out of box hide/show the target (with fading as setting)	When the menu name is selected then show/hide the target (with fading as setting)
	Movement • Keyframes	Set the keyframes between start and stop frame then animate the entity follows these keyframes	Set the keyframes according to the duration setting, when the interactor is selected then animate the target follows these keyframes	Set the keyframes according to the duration setting, when user avatar hits the box then animate the target follows these keyframes	Set the keyframes according to the duration setting, when the menu name is selected then animate the target follows these keyframes
Media assignment	Information • Contents	Show information window from the start frame until the stop frame then close the information window	When the interactor is selected then show the information windows (closed by user)	When user avatar hits the box then show the information windows and when out of box close the window	When the menu name is selected then show the information windows (closed by user)
	Description • Lines	Show description from the start frame until the stop frame then hide description	When the interactor is selected then show/hide the description	When user avatar hits the box then show the description and when out of box hide the description	When the menu name is selected then show/hide the description
	Caption • Texts	Show caption from the start frame until the stop frame then hide caption	When the interactor is selected then show/hide the caption	When user avatar hits the box then show the caption and when out of box hide the caption	When the menu name is selected then show/hide the caption
	Audio • Files	Play the audio file from the start frame and stop playing at the stop frame	When the interactor is selected then play/stop the audio file	When user avatar hits the box then play the audio file and when out of box stop playing	When the menu name is selected then play/stop the audio file

continued ...

	Actions	Timing	Selection	Collision	Menu
Camera action	Show • Distant • Up • Angle	At the start frame change the camera viewpoint to the entity position (manipulation is not allowed)	When the interactor is selected then change the camera viewpoint to the target position (manipulation is allowed)	When user avatar hits the box then change the camera viewpoint to the target position (manipulation is allowed)	When the menu name is selected then change the camera viewpoint to the target position (manipulation is allowed)
	Look-around • Distant • Up • Angle	From the start frame rotate the camera around the entity until the stop frame	When the interactor is selected then rotate the camera around the target follow duration setting	When user avatar hits the box then rotate the camera around the target follow duration setting	When the menu name is selected then rotate the camera around the target follow duration setting
Optional action	Navigation • Start(x,y,z)	At the start frame go to navigation point and change navigation mode to walkthrough	When the interactor is selected then go to navigation point and change navigation mode to walkthrough	When user avatar hits the box then change avatar position to the navigation point	When the menu name is selected then go to navigation point and change navigation mode to walkthrough
	Goto • Scene	At the start frame change running scene to the selected scene	When the interactor is selected then change running scene to the selected scene	When user avatar hits the box then show the selected scene until its end	When the menu name is selected then change running scene to the selected scene

All interaction events have two output parameters which are the target and the duration to define more constraints on action. Some actions are activated through an object which is called the target. For example, a transition action is the action to show or hide a target. Some actions need duration definition. A movement action is the action to animate an object. It follows the keyframes along the duration time. However, some actions do not need to assign target and duration. Thus, we leave these parameters without any assignment. For example, information action shows an information window that does not expect further action on an object then target and duration are avoid.

Events and actions assignment are designed to support all interactions which are selection, manipulation, navigation and system control as defined previously. Selection is used when the selection event is assigned. An interactor (set as an input parameter) allows user to select it. Manipulation is applied when the "Show" action is assigned to the interactive event whether selection, collision or menu. The target (output parameter) is an object for this interaction. The target object allows user to manipulate it individually such as positioning, rotating and scaling.

The two types of navigation are "travel" and "wayfinding". The travel navigation may use a collision check. The box collider is thus an input parameter of the travel navigation. It allows user to travel in the virtual environment and launch action whenever the box is collided. Wayfinding navigation is used when a menu event triggers a navigation action or a goto action. This allows user to go to new position in the virtual environment. Furthermore, the system control is used when the menu event is assigned also depend on objective of developer.

Thanks to all this high level abstract actions we expect a more natural description of the

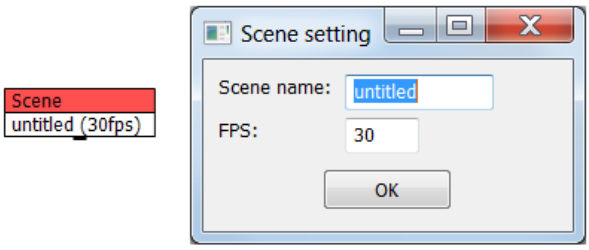

Figure 4.16: Scene setting in the event editor

story by any VM designer even if he is not a computer scientist.

4.5 Scene organization

The whole story may be too long leading to a very complex graph in the event editor. We should divide storyboards into easy and convenient ways to create and manage storytelling models in the event editor. In the design of the event editor, the whole story is divided into various scenes created on different canvas. The first storytelling model starts with three components: entity, event, and action then connect the three nodes together to define action for each entity according to events. Actions of each entity can be combined to make a story. Each scene have a single scene node created as a root element to determine which entities are displayed in the scene. The scene node has a name, a number of frames (frames per second) set on event editor as shown in Figure 4.16.

In order to build and organize each scene of the story, the Storytelling platform provides a scene button in the toolbar with several commands such as multiple scenes, new scene, save scene, timeline data and open scene as described in the Section 3.6.1. These commands are designed to support scene organization in relation with the event editor and the timeline panel as shown in Figure 4.17. In general, the timeline panel works with the active scene. If the multiple scenes command is activated the timeline panel will interpret the scenes from the player settings (Figure 4.18) in sequence and generate stories from various scenes simultaneously.

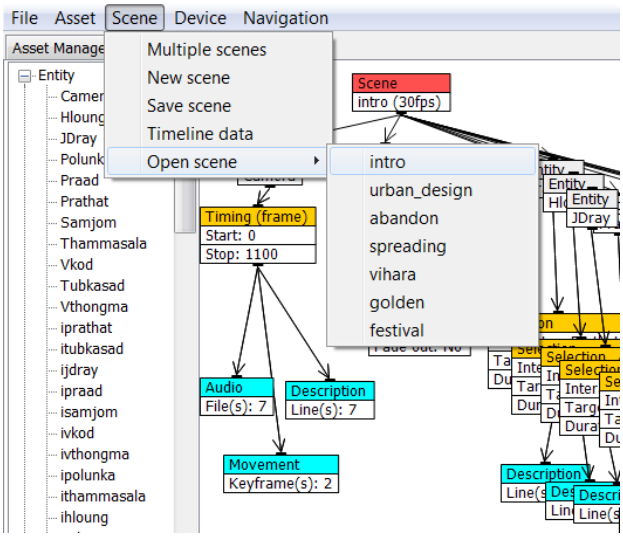

Figure 4.17: Commands in the toolbar to support scene organization

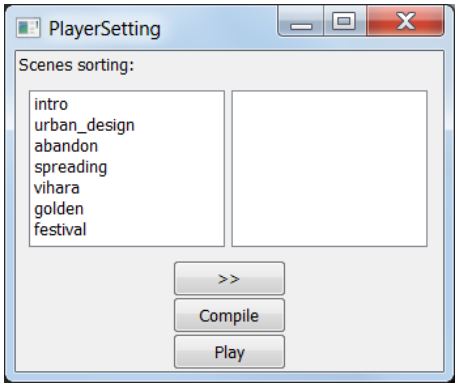

Figure 4.18: Player setting when multiple scenes running is required

4.6 Case study: Wat Phra Mahathat Woramahawihan

There are many cultural heritage sites and museums in the world waiting for people to visit. Thailand has several cultural heritage sites including temples that have interesting histories. Wat Phra Mahathat Woramahawihan is a temple in southern Thailand that has a unique history. Here, we expect to tell this story to visitors who come to this place. We would like to make a contribution to let visitors have more understanding about this place using VR technology. We have 3D models from the real place and the history of this temple as a resource and also the platform and modeling concept are ready to develop the virtual environment and content. Therefore, it is a good case study to apply our platform. The Storytelling platform is implemented using historical model, storytelling model and interaction model to develop VM application. In order to develop an application, we need to simplify the resource and parse it into a virtual environment with interaction. The processes are a guideline for any kind of VM application.

4.6.1 Introduction

Wat Phra Mahathat Woramahawihan (Wat Phrathat) is the main Buddhist temple (Wat) of Nakhon Si Thammarat Province, the largest province in Southern Thailand. It is located on the main sand bar of Nakhon Si Thammarat on which the ancient town and the present town of Nakhon Si Thammarat were built. The ancient town of Nakhon Si Thammarat was developed from the early state of Thailand called Tambralinga. Its name of which is mentioned in the Pali canon of the Buddhism as one of the prosperous port towns of the Eastern world, and thereby archaeological evidence found at many sites in Nakhon Si Thammarat supports the literary evidence. Tambralinga became a flourishing port town and was ruled independently since the 5th century CE and continued onwards. At some points of times it joined a union with Sri Vijaya, the Mahayana Buddhist Kingdom, which was famous for the world maritime trade networks during the 8th to the 12th century CE. The main stupa of the temple called Phra Borommathat Chedi (Great Noble Relics Stupa) was built by King Sri Dhammasokaraja in early 13th century CE to establish a symbol for the Theravada Buddhism sect in the province. It is believed that the temple houses a tooth relic of Gautama Buddha. The temple was nominated in the additional list of UNESCO World Heritage Sites[41] in 2012.

[41] https://whc.unesco.org/en/tentativelists/5752/

Figure 4.19: The main stupa which called Phra Borommathat Chedi (Retrieve from https://en.wikipedia.org/wiki/Wat_Phra_Mahathat)

Figure 4.20: Location of Nakhon Si Thammarat and Srivijaya Kingdom in the past (Retrieve from https://en.wikipedia.org/wiki/Srivijaya)

4.6.2 Brief history of Wat Phra Mahathat Woramahawihan

Here, we write the story of the temple in natural language to prepare its translation into the Storytelling model.

Wat Phrathat has been the center of the locals with a history of more than 1800 years and significant art and cultural aspects. Wat Phrathat is also considered as the origin of people's faith towards Buddhism in the south of Thailand. It has influenced the form of architecture and communication and existence of Theravade doctrine of Lanka. There have been various legends of Phrathat. One legend mentioned that the Buddha relics (the teeth of the Lord Buddha) have been kept inside. The relics were moved from Tontha Buri by Prince Thanakuman and Princess Hemchala. It has been assumed that the original form of Phrathat was Mondop, a structure with four arches and a pyramidal roof topped with five tiers as Srivijaya style. Then, in 13th century CE, Theravade doctrine of Lanka has been prosperous and was spread to Nakhon Si Thammarat, as a result Chedi of Theravade was constructed to cover the original, but existing one. It becomes a typical model of Theravade in the south of Thailand and influentially spreads to adjacent areas. The value of Wat Phra Mahathat Woramahawihan also portrays creativity of human race in architecture via

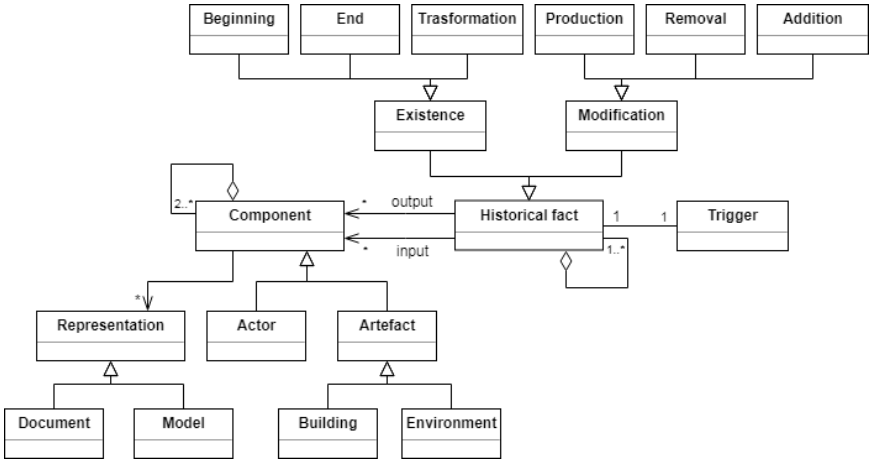

Figure 4.21: Historical model to define conceptual idea for the story making

artistic Buddhism which, moreover, identified and clearly explained through the location of Wat Phrathat. Wat Phrathat is situated at the ideal position which reflects urban design, with mountain ranges and streams at both sides. In addition, Wat Phrathat is situated on the sand hill, hence this was recognized as the landmark by sailors. The architecture, furthermore, reflects the integration between Buddhist philosophy and creativity of artisans who created components of the temples, as various viharas with their identity. Faith towards Buddhism has been playing roles which influence and reflect the locals' faith since the old days. In the past, the locals collected their valuable belongings and kept them at the top part of the Chedi. Hence the Chedi was called Phrathat Yod Thong, which means the golden top. Important activities reflecting faith and ways of life of the people are held twice a year, i.e. on Make Buja's Day and Visaka Buja's Day. The locals will buy white, red or yellow fabrics and sew them together as a long cloth to symbolize Phra Bot, the sacred cloth representing the Lord Buddha. There is the procession, an activity when the locals come together to hold the cloth and put it around the Chedi. Wat Phra Mahathat Woramahawihan directly associates with the locals' ways of life. In other words, a religious center, along with cultural and architectural heritage of Buddhism, whose types of value has been retained until the present days.

4.6.3 Extracting on historical model

Based on the brief history, we can divide the whole story into a series of scenes and define a group of information that is used to carry the story. For each scene we can analyze historical facts and differentiate components for an easy understanding and a comfortable organization. Both historical fact and component will be an important part for the implementation of a VM exhibition. Following scene description are examples of using historical model uses as basic components of the story told by the VM.

Scene1: introduction
Wat Phrathat has been the center of the locals with a history of more than 1800 years and significant art and cultural aspects. Wat Phrathat is also considered as the origin of people's faith towards Buddhism in the south of Thailand. It has influenced the form of architecture and communication and existence of Theravade doctrine of Lanka. To describe such a scene we will need:

- Historical facts: communication and existence of Theravade
- Components: Wat Phrathat, landscape

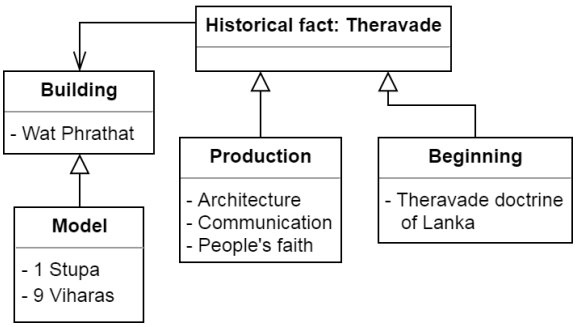

Figure 4.22: Historical model for introduction scene

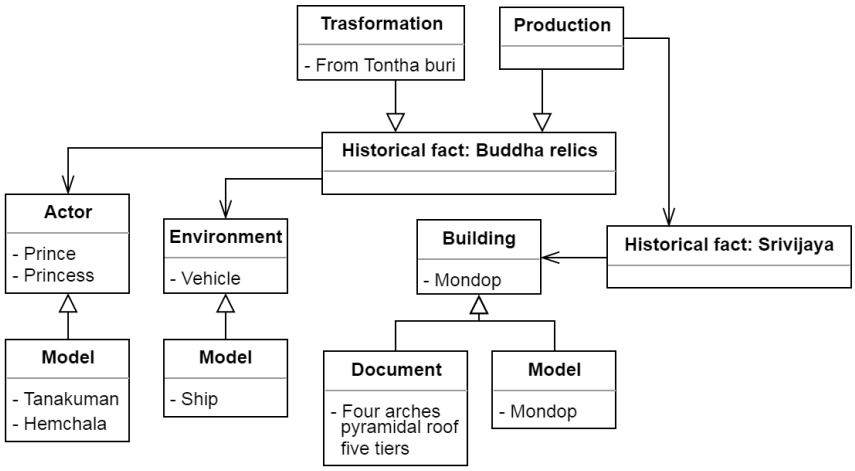

Figure 4.23: Historical model for abandon scene

Scene2: abandon
There have been various legends of Phrathat. One legend mentioned that the Buddha relics (the teeth of the Lord Buddha) have been kept inside. The relics were moved from Tontha Buri by Prince Thanakuman and Princess Hemchala. It has been assumed that the original form of Phrathat was Mondop, a structure with four arches and a pyramidal roof topped with five tiers as Srivijaya style. Here, this scene will use:

- Historical facts: Buddha relics, Srivijaya style
- Components: Prince Thanakuman and Princess Hemchala and their vehicle, Mondop

When the history is designed within the historical model for each scene, the implementation of the story to create virtual environments is possible. Both historical facts and components of each scene from the historical model are used to design a storytelling model. An interaction model to determine characteristic of the virtual environment follows the historical model. Examples of historical model of introduction scene and abandon scene are shown in the Figure 4.22 and 4.23.

4.6.4 Storytelling model and interaction model

The next step is to set up a story from various information and to format presentations. Historical fact and component are used to create a storytelling model that includes entities, events and actions. A component is converted into an entity by associating a 3D model which can be a single entity or a group of entities. A historical fact is presented through

the component, where every component has two main parts: existence and modification. Events and actions are the behavior mechanisms of the story. Existence allows to enter a scene and to expose it. Components in the scene are distributed into beginning, end and transformation. Beginning and end are supported by "Transition" action nodes. A transformation is translated into a node named "Movement" to create the behavior in the storytelling model. A modification is either an addition, a removal or a production. Unlike the existence, this action is not specific, but depends on the narrator wish. Components grouping and scene organization are provided to support component modification with action nodes.

In summary, when considering a historical model, historical facts and components are used to design a storytelling model. In addition, the design of a virtual environment requires interaction with the user. In this part, there is an event node that supports information management derived from different conditions due to various interaction systems. The Interaction model handles devices interacting with users and virtual environments and manage pre-defined interaction techniques. To study how interactions affect user learning, three types of experiments were designed to study interaction with different virtual environments. We call them fully-guided story, semi-guided story and non-guided story. There are defined in the next sub-sections.

4.6.5 Fully-guided story

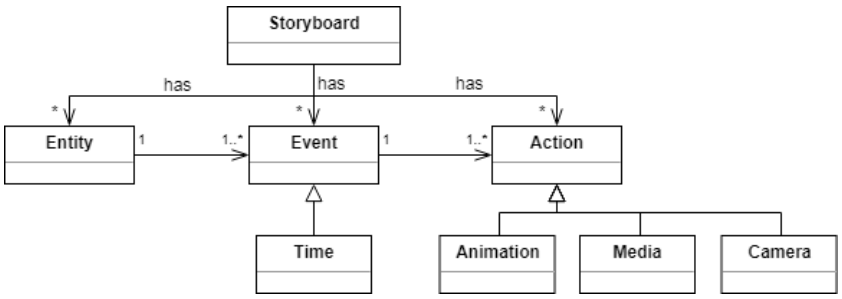

Figure 4.24: Storytelling model of the fully-guided story

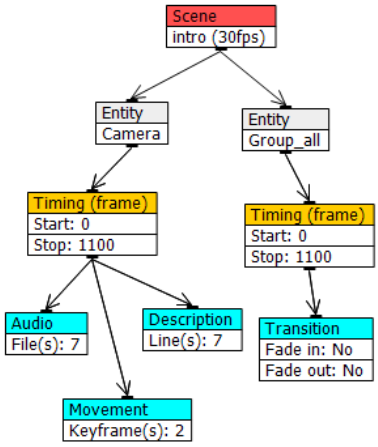

Figure 4.25: Fully-guided story of introduction scene in the event editor

Fully-guided story is a specific story created without any human interaction within the scene. It creates a story that looks like a movie. It only allows user to watch, focus on animations and narrative production. Users follow what is going on. Each entity uses

an event called “Timing”, which defines the starting and ending frames, depending on the action. Transition and movement create animations. Audio and description create a multimedia content. Camera viewpoint control assists the story follow-up and defines attention points in the scene. We create a fully-guided story to evaluate the user’s learning in case of passive media. The fully-guided story is thus content pushing. Users can only attend and get information from the media. The examples of fully-guided story in the event editor are the introduction scene and abandon scene as shown in the Figure 4.25 and 4.27.

The introduction scene in Figure 4.25 consists of two entities: Camera and Group_all. The entity Group_all is a group of all models in this scene. It linked to the Timing event and Transition action to appear these models in specific time interval without any movement. The Camera entity is set to use the same time linked to three actions. Audio and Description are the action to assign media, where the Movement is set to pan the camera as specified. The result of this scene graph produces a story with a simple information.

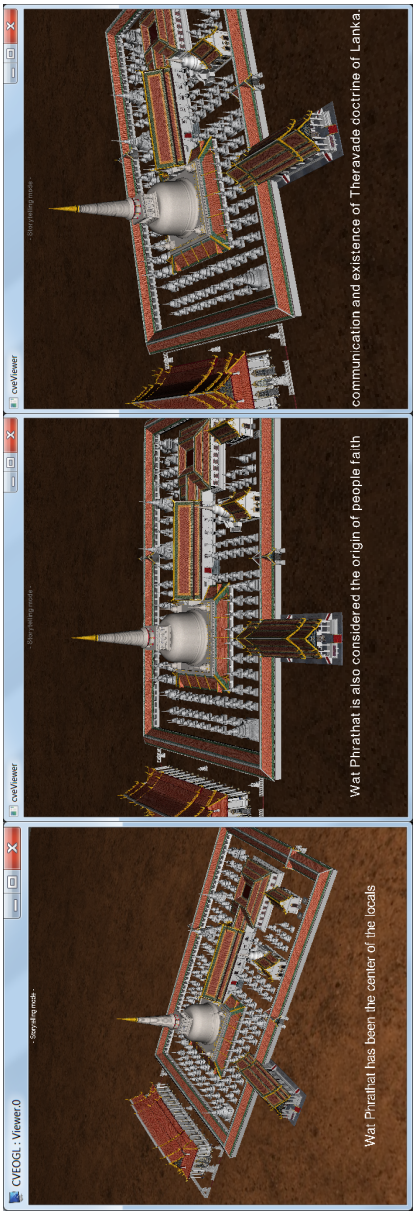

Figure 4.26: Result of fully-guided story of introduction scene on the viewer

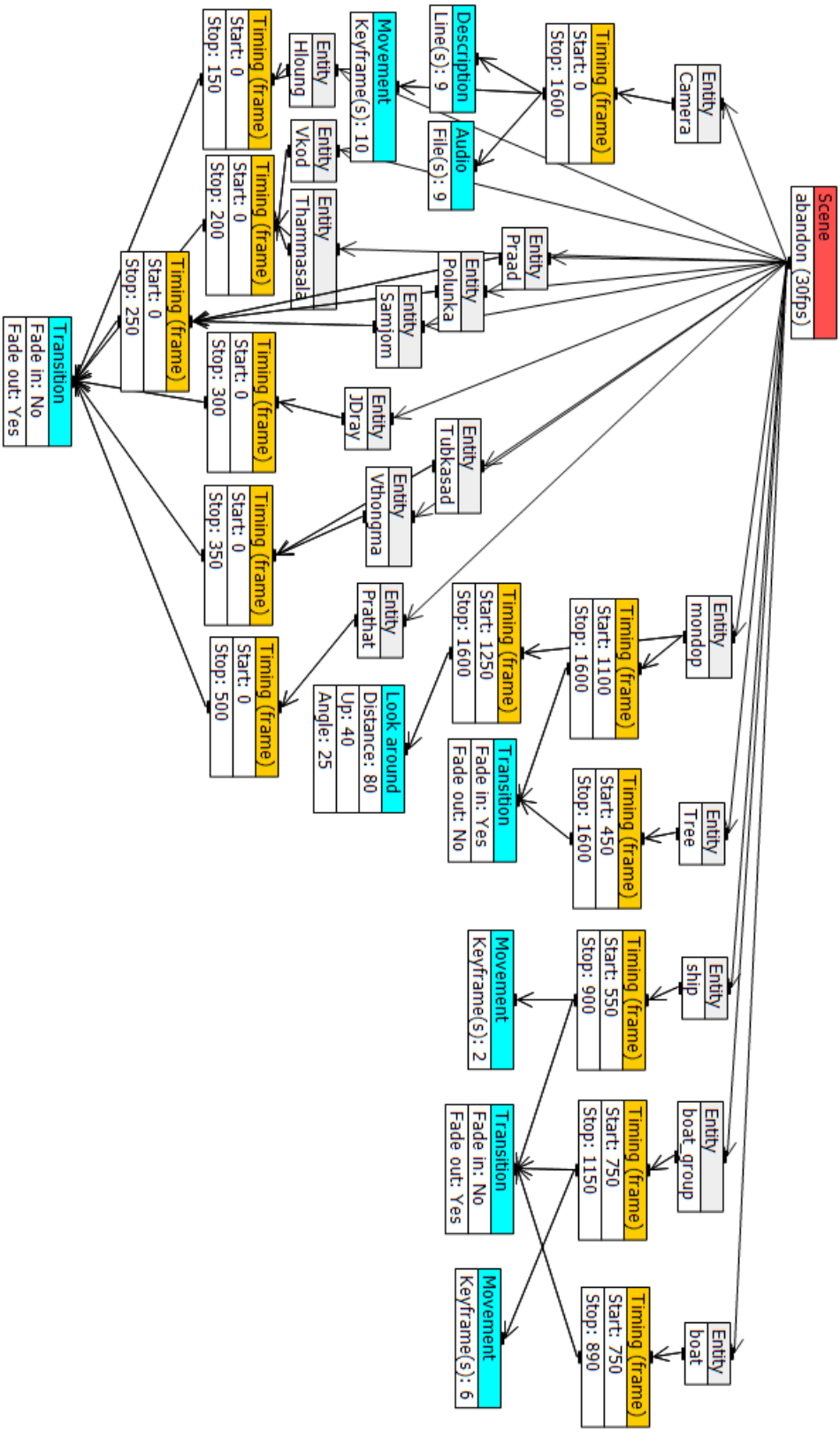

Figure 4.27: Fully-guided story of abandon scene in the event editor

Figure 4.28: Result of fully-guided story of abandon scene on the viewer

4.6.6 Semi-guided story

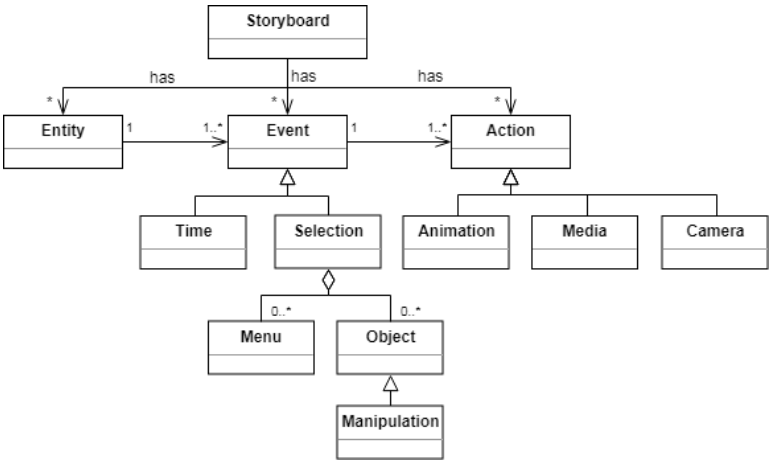

Figure 4.29: Storytelling model of the semi-guided story

The semi-guided story is similar as the fully-guided story, but it adds interactive event selection and manipulation to allow users to interact with the story. “Selection” and “Menu” events are used within the event editor to allow user interaction with objects and widget menus in a scene, but a main stream follows timing events. By adding selection and menu events, it is possible to create more display options and let user focus on specific items. Details were defined in Section 4.4.

We integrate manipulation mode, which allows direct object interaction. It adds this characteristic through the “Selection” event that is connected to the “Show” action. When an object is selected, it displays a single object so that users manipulate the object and get more details and information about it. The semi-guided story uses a camera-free navigation mode, which allows the user to freely control the camera viewpoint to interact with objects within the scene, but he cannot navigate freely.

The purpose of the semi-guided story design is to study the differences in user learning when adding interactivity. It does not specify which objects the user must interact with, so the user behavior on the virtual environments in the semi-guided story can be observed. The system will be able to store interaction data between users and virtual environments. This information is used to analyze user interest. VM developers should design interactive objects and virtual environment corresponding to the needs of users. The examples of introduction scene and abandon scene for semi-guided story as designed in the event editor are shown in the Figure 4.30 and 4.32.

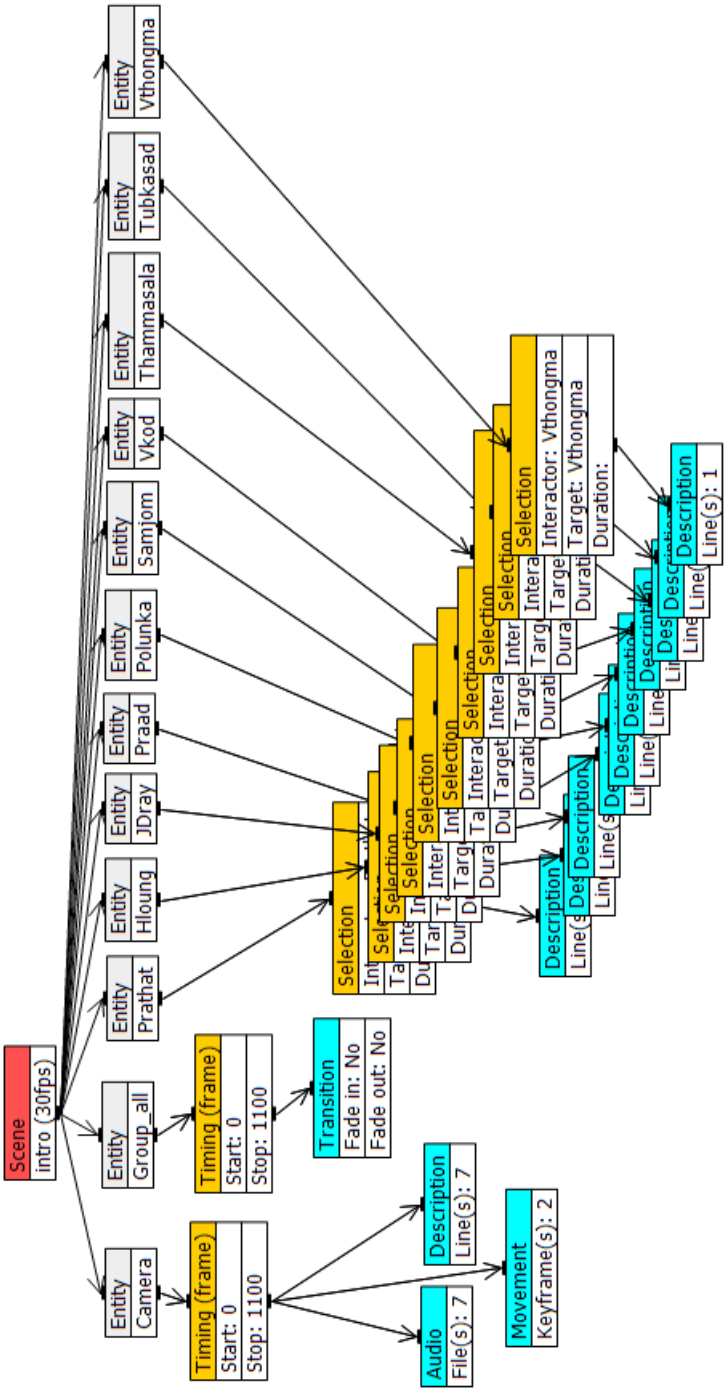

Figure 4.30: Introduction scene in the event editor of the semi-guided story

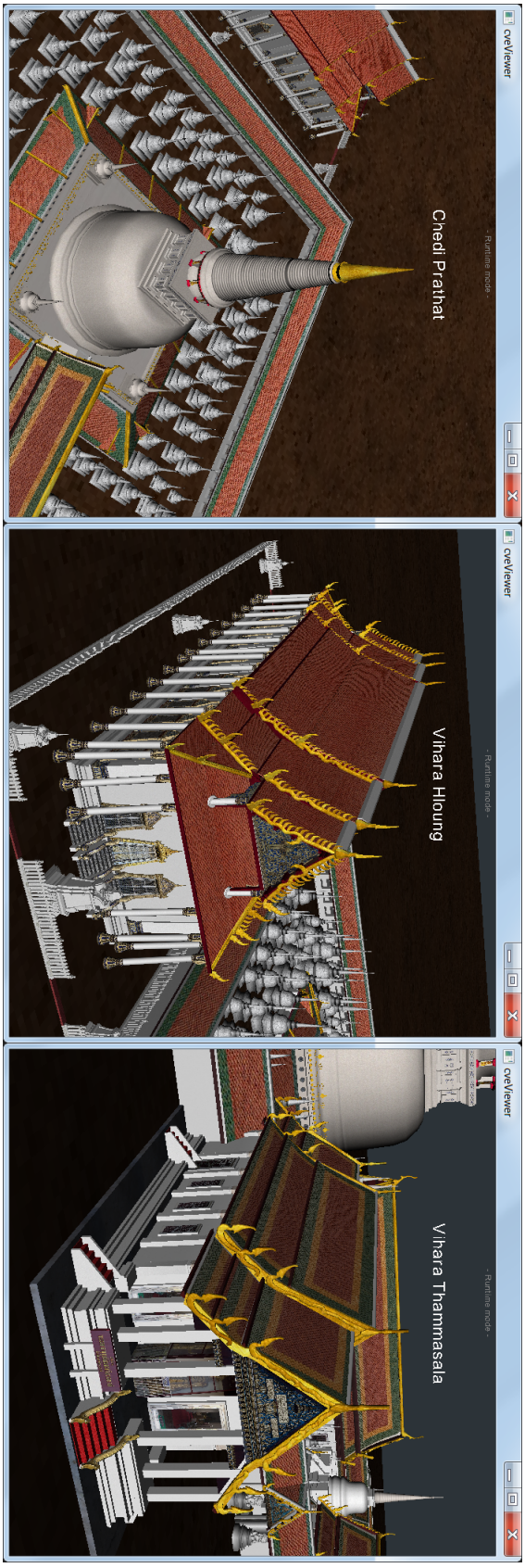

Figure 4.31: Examples of semi-guided story of introduction scene when user interacts on objects

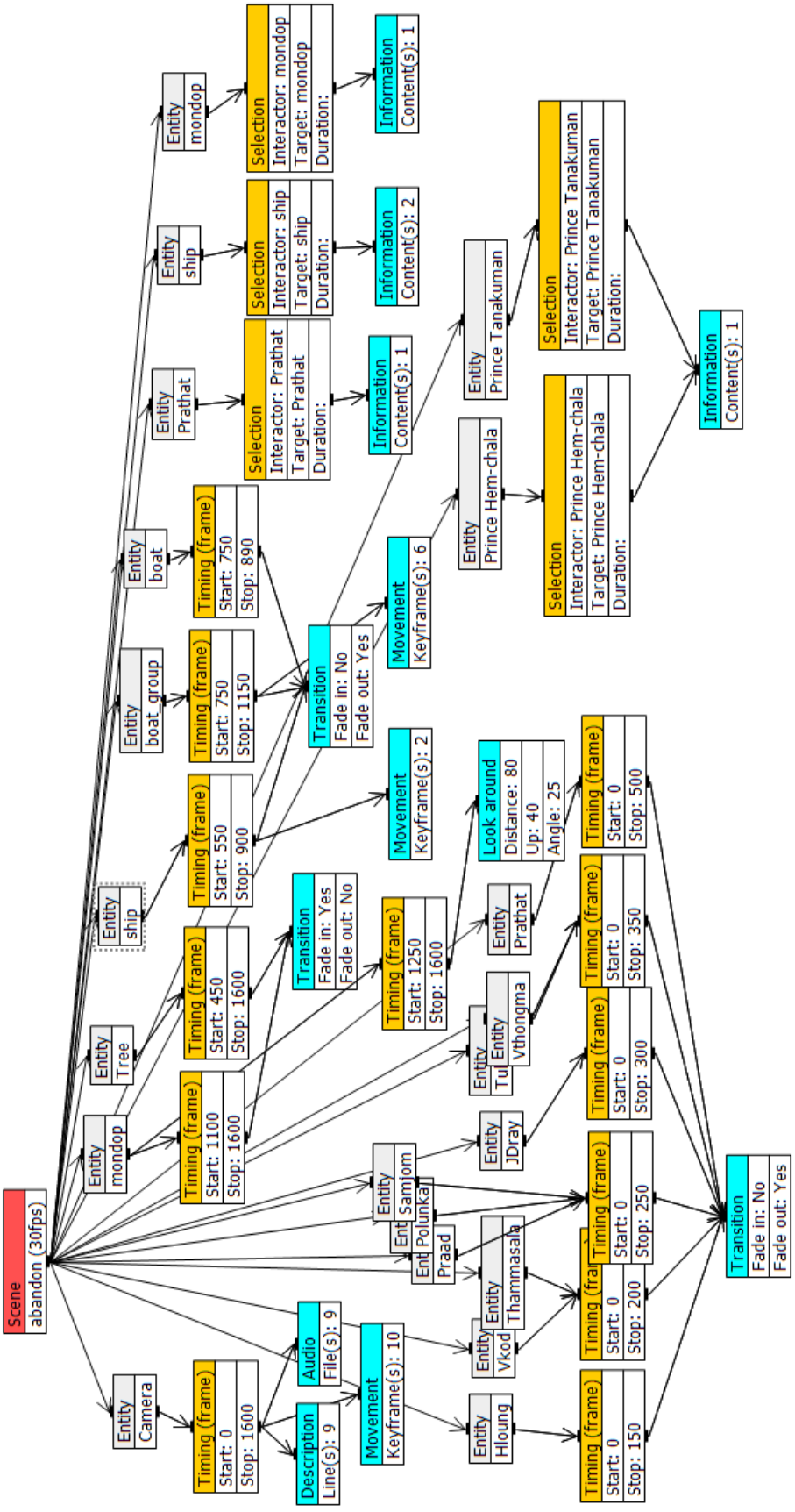

Figure 4.32: Abandon scene in the event editor of the semi-guided story

Figure 4.33: Examples of semi-guided story of abandon scene when user interacts on objects

4.6.7 Non-guided story

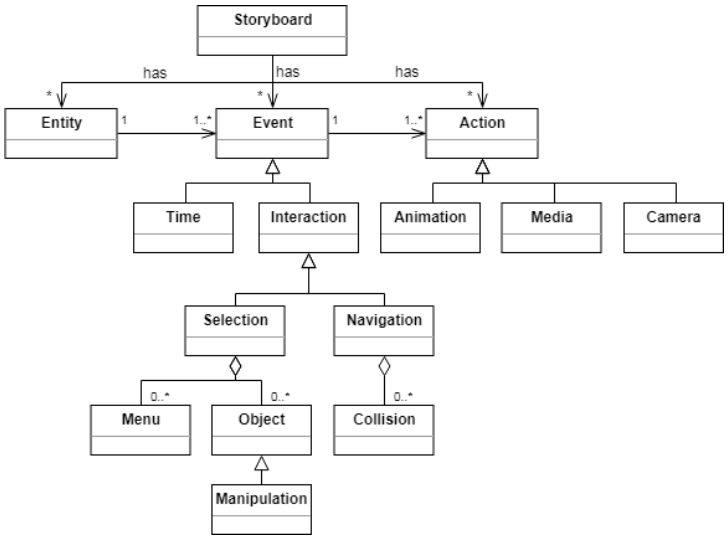

Figure 4.34: Storytelling model of the non-guided story

The non-guided story is a step further allowing a walkthrough navigation mode. A first-person perspective camera attached to the avatar enables user lives inside the virtual environment. The user can freely control the camera viewpoint and movement to interact with objects within the scene.

Selection and manipulation event still plays an important role for interacting with the virtual environment. "Collision" events are added according to the walkthrough navigation to support the event of navigation. When a user navigates into a given area, we customize the actions as a result of the collision. It is used to create a new style of storytelling which allows story fully driven by user interaction with the virtual environment. It is also referred as a non-linear story, because the story is not in the sequence of the scene as it is in a fully-guided story. It depends on user exploration and the areas of interaction which produces uncertain story. With a freedom walkthrough, the user may not have a target for finding information. Therefore, it is important to highlight some key areas that will be linked to the main content to help users access to required and available information.

The purpose of the non-guided story design is to study the user's learning ability when there is no direct presentation of content and information. Instead, it will allow users to manually search for information by direct interaction with the virtual environment. The interaction between the user and the virtual environments will also be collected. The data will be analyzed for the user's behavior against the other models in order to improve the structure of the virtual environment to meet users interesting points and to increase the potential for learning.

The Figure 4.35 shows non-guided story with single scene on the event editor. A collection of interactions are placed on this scene where the story is run by user interaction with walkthrough mode navigation. The interactions lead to various results which can be animation, information or jumping to another scene. All interactions selection, manipulation and navigation are fully used with non-guided story.

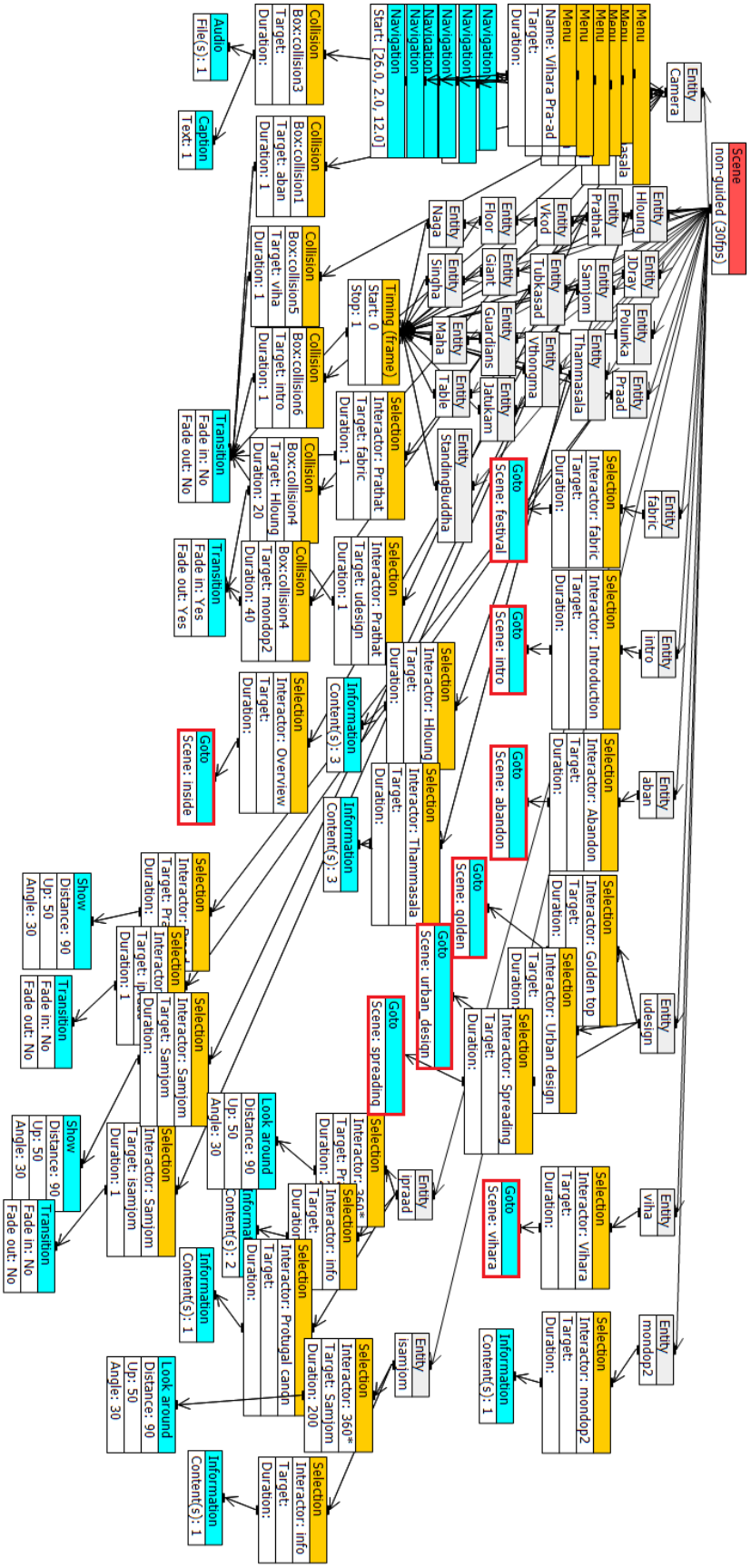

Figure 4.35: Non-guided scene in the event editor for non-guided story with ability to go to another scene (red frame action nodes)

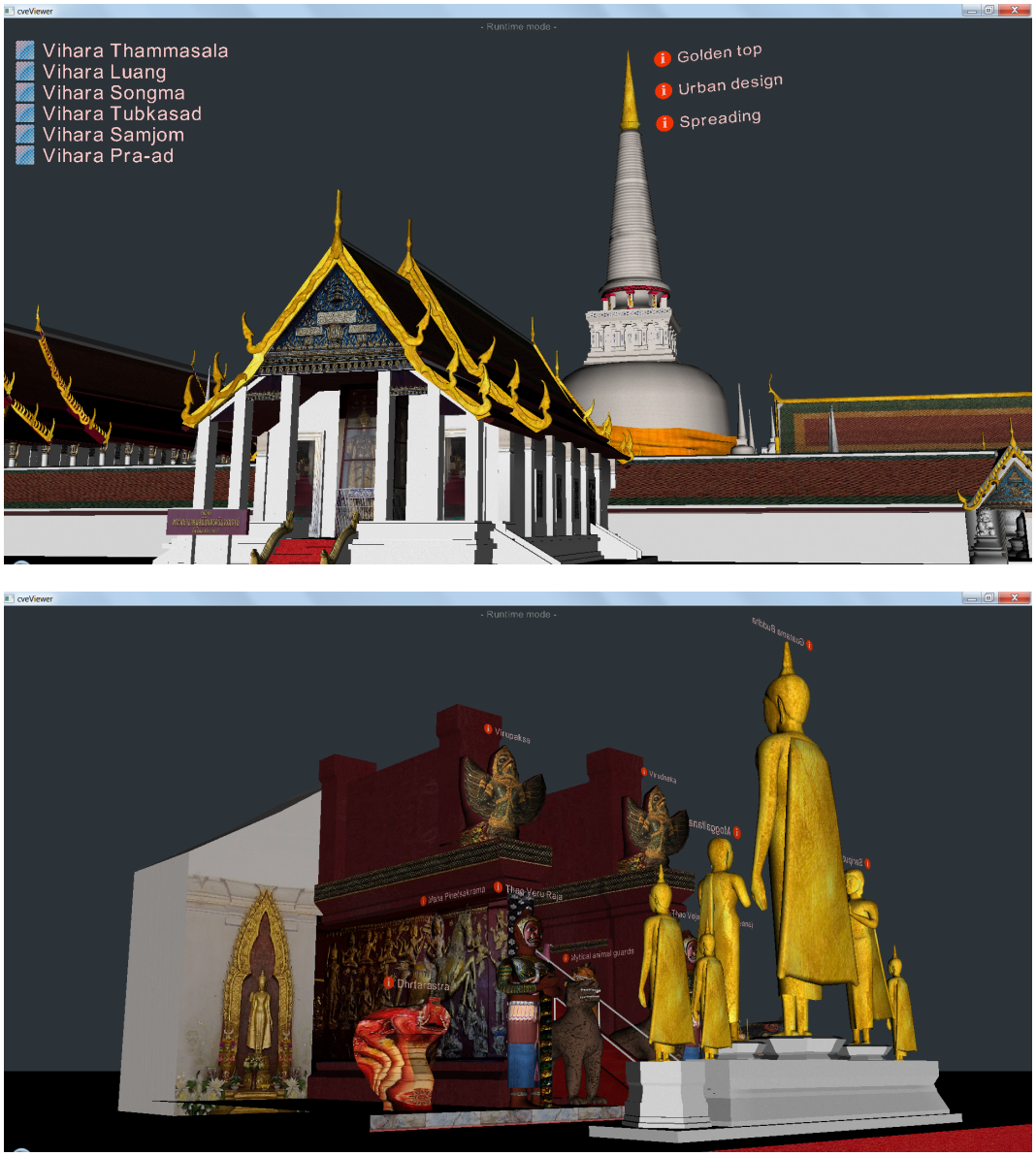

Figure 4.36: Examples of non-guided story where user needs to interact within the scene

4.7 Conclusions

In this chapter, we described the creation of a VM on our Storytelling platform, starting with the installation of three interaction systems: a 2D interaction system, a 3D interaction system, and a CAVE interaction system. Since the development of the VM application was previously based on the device used, switching to another device required a reorganization of the interaction resulting in time consuming development. In order to solve this problem, G-SCOP CVE was used to manage and design the Storytelling platform with the ability to connect different devices and displays. All interaction systems are connected to the CVE server using different devices, so there are different interaction techniques which are designed adapted to the device.

The first part of this chapter defines the interaction system and interaction techniques based on the interaction model which were defined in the Chapter 3. Thus, the Storytelling platform has the ability to select interaction systems as required. This allows us to study the use of the different interaction systems of VM applications for the same story. The Storytelling platform enables independent interaction system by creating interactive content that uses interaction without interfering with the device. Nevertheless, it depends on the interaction techniques that are pre-defined including selection, manipulation and navigation.

In the event editor, the structure of the story is composed of entity, event and action. It is an interactive content that manages both the content structure and interactivity based on the storytelling model and the interaction model. Moreover, it also has a scene organization working with the timeline panel to assist complex VM development.

The case studies were designed to be used in conjunction with the interaction system to investigate how different interaction system and story affect the learning outcomes. The case study is derived within 3 different interaction modes: fully-guided story, semi-guided story and non-guided story. The first fully-guided story was created to test user's behavior without the need of interaction with the story. The semi-guided story was built to test user's behavior when some interactions are available, selection and manipulation should be observed here. The non-guided story will not provide any information to users so we can observe user's behavior to find information especially by navigation. These three case studies demonstrate the ability of a Storytelling platform to create a wide range of interactive content including interaction system which is freely altered. This can be used to design the content and to investigate the appropriate interaction system for VM applications.

Chapter 5

Experiments

5.1 Research question

According to the state of the art of VM discussed in the Chapter 2, VM development has evolved from past to present from digital media, online VM, interactive Web3D VM, mobile application VM and interactive on-site installation. Most application developments are device dependent. For example, digital media applications are developed for desktop devices. Online VM and interactive Web3D VM applications are also developed for desktop devices, but run on web browsers. Mobile applications are developed using mobile devices such as smartphones or tablets. VM applications based on interactive on-site installation are developed on various devices because these applications are using VR technology which remains non-standardized. It allows user to immerse into virtual content with multiple devices by tricking their visualization and interaction for more interactive VM.

This enables a wide range of interactive on-site installation developments that can be designed to interact with users, depending on the content offered. The content is a virtual model and its interaction with users via devices and interaction techniques. Since VR technologies are more efficient and now cheaper, various devices are available and the VE content is freely designed. It is a real challenge to develop an interactive on-site installation VM and to decide which types of device should be used and how the content should be designed to maximize user engagement. With these problems in mind, we state the research question to study and explore device usage and user engagement as follows:

"How do different interaction systems affect user engagement in the application of on-site installation VM and what are the key factors?"

This research needs a conceptual framework to develop an interactive on-site installation of VMs and to solve these problems following these objectives:

- To develop a complete platform to design interactive VEs and to manage device connection dedicated to VM which allows users interact efficiently with the proposed knowledge.
- To develop an interaction system that provides suitable devices and interaction techniques to activate and support users' interaction in VM.

In Chapter 3, a Storytelling platform is proposed. It provides the tool to create interactive content which is connected to various devices. It is supporting digital interactive content for application of on-site installation VM. Storytelling model and interaction system are applied to be a high- level abstraction of story making in term of VEs. The platform provides a tool to organize the story and interaction which are handled on a CVE application. This

application enables the story execution thanks to an event editor. Model transformation allows device and interaction techniques change without story structure or programming modification.

In the Chapter 4, we described the creation of a VM on a Storytelling platform. We proposed three case studies: a virtual tour, a semi-guided story and a non-guided story were designed to investigate how different interaction system and content affect the learning outcomes. Interaction systems and interaction techniques are designed based on the interaction system which was defined in Chapter 3 to be applied within on-site installation VM. Various display devices are designed for different interaction systems which are 2D, 3D and CAVE. All systems are connected to the input device and designed interaction techniques are managed by the device.

The Storytelling platform is a tool to create the content in term of interactive on-site installation. Different styles of stories with three case studies are prepared to study user engagement on each content style and also to investigate needs of interaction for these case studies. Device usage and performance is observed for different interaction system. A case study may be connected to any interaction system. A methodology to assess device efficiency, usability of interaction and content design is now proposed to investigate the initial research question. The result of this study will help us to re-design the content for this story as well as to identify user's behavior on each interaction system. The goal is to contribute to a better immersive museum experience for visitors.

5.2 Experience creation

The research question is about devices usage and user engagement. Then the objective of this study is proposed to compare the device performance of each interaction system and user behavior for different kinds of content. To answer the research question, we designed the experience using three types of content: a virtual tour, a semi-guided story and a non-guided, each type will be tested with three interaction systems.

A virtual tour scene will test the performance of the device and interaction techniques. Users have the mission to navigate within the VE, which requires interaction techniques including selection, manipulation and navigation. The time spent on the mission and the satisfaction with each device is evaluated as the device performance.

A semi-guided story is a linear story where users interact with the desired VE to access additional information. If the user has no interaction with the VE, the semi-guided story becomes a fully-guided story as a streaming video, but in term of 3D immersive animation. In this experience, the needs of interaction are explored to see what particular part of the story the user is interested in. Furthermore, using the semi-guided story with a different interaction system will assess the impact of each interaction system on user behavior including needs of interaction.

Non-guided story is a non-linear story where behavior depends on the user interaction. Story opportunities are presented to the users based on their interaction with the VE. In this experience, we examine the holding power of the system, but also how the interactive content attracts to user interest. The difference between a semi-guided story and a non-guided story is interaction enabling. In a non-guided story, the user interacts with VEs, while in a semi-guided story no interaction is required. The experience compares the holding power of interaction system with either passive visit or active visit.

The user's behavior on each interaction system is evaluated to indicate good interaction system. The evaluation of user engagement assesses the content used in experience with different interaction system. The interaction performance on each type of content is also evaluated. The performance of interaction system is divided into four categories according

Table 5.1: The evaluation of user engagement and device usage on comparison between each interaction system and the three types of content

	2D	**3D**	**CAVE**		
Virtual tour	2D performance	3D performance	CAVE performance	⇒	Device performance
Semi-guided story	Needs of 2D interaction	Needs of 3D interaction	Needs of CAVE interaction	⇒	Needs of interaction
Non-guided story	Holding power of 2D	Holding power of 3D	Holding power of CAVE	⇒	Holding power
	⇓	⇓	⇓		
	Users' behavior on 2D	Users' behavior on 3D	Users' behavior on CAVE		

to devices and interaction techniques. Obviously each interaction system has a different performance that affects the user engagement.

This study is based on four hypotheses:

- **H1 (Usability hypothesis):** participants have different performance to accomplish the tasks on different interaction system.
- **H2 (Needs of interaction hypothesis):** participants have different needs of interaction when visiting with different interaction system.
- **H3 (Learning hypothesis):** participants learn differently when interacting on different interaction system.
- **H4 (Holding power hypothesis):** the visit time depends on the interaction system. A short time could demonstrate modality efficiency, but a long time may be due to a high interest.

5.3 Evaluation and measurement

Usability of interaction defines the ability of users to accomplish the specific tasks with effectiveness, efficiency and satisfaction in a specified context of use (Dix et al., 1998; Nielsen, 2003). Usability testing is an assessment paradigm, especially in the later stages of design (Bevan et al., 2015) to ensure consistency in the structure of interaction and how the system responds to the user. Usability testing is concerned with measuring user's performance on typical tasks (McCracken and Wolfe, 2004). Generally, this is done by user observation and by recording the time taken to complete the task when performance expects short time. User satisfaction questionnaires and interviews are also used to express user feedback. The characteristic and the format of usability measurement are defined as follow:

- Performance is observed directly by capturing the user within the specific task including the time required to complete the task with effectiveness and efficiency. This type is a quantitative and objective measure. Such measurements concern navigation time, manipulation time and selection time.
- Preference measures are an indication of a user perception about the context of use of interaction which is not directly observable. This type is a qualitative measure which is determined by questionnaires or interviews. We measure the ease of use, immersion perception and overall satisfaction.

Need of interaction and interactive designs are applied for learning process in physical museums. Visitor engagement in museums (Boisvert and Slez, 1995) is identified by attractiveness and holding power as necessary steps for learning. Attraction is the measurement of visitors who stops at the exhibit while holding power is time spent by visitors in the exhibit. Attraction and holding power are important variables for understanding the learning environment of the museum (Yahya, 1997). In addition, holding power affects the visitor participation during visiting the exhibition. In the study (Chittaro and Ieronutti, 2004), visitor's behaviors in a real museum environment were identified as similar to the behaviors of visitors in VM. The information regarding visitor behaviors is an important indicator of VM abilities to engage visitor's attention and maintain their interest. Therefore, we use attractiveness and holding power for evaluation, which will be adapted to the context of VM as follows:

- Attractiveness: the number of users who interact with the VE for additional information.
- Holding power: amount of time spent by users interacting with the content. Holding power is higher when the user stays longer.

Evaluation consists of formative and summative measurement. Formative measurement is used to monitor user's behavior to provide ongoing feedback that is used to improve interaction system and content design. Summative measurement is used to assess interaction performance at the end of an experience by comparing system usage with quantitative and qualitative data.

In our study design, both formative and summative measurements are used to validate performance of interaction system, but also the designed content. The evaluation methods are classified to be the quantitative objective measurement and qualitative subjective measurements. The details of these measurements are described as follows:

Quantitative objective measurements are concerning usability of interaction system in term of performance to accomplish the tasks. This measure relates to the content which is designed in terms of needs of interaction and interactive design. These data has to be in numerical form to run statistical analysis. Thus, we convert them in numerical measurements:

- Navigation time: the time spent to change the position from a start position to a target position.
- Manipulation time: the time spent to manipulate objects to dock it into an expected position.
- Selection time: the time spent to select objects.
- Holding power: amount of time spent by users to use the system.
- Amount of interactions: the number of interactions with the object.
- Comprehension scores: the scores of a quiz comprehension test about the story.

Qualitative subjective measurements are concerning the impact on the visitor. Questionnaires are used to assess visitor preferences. We wish to measure:

- Ease of use: the capability of an interaction system to enable the user to learn how to use it.
- Immersion: user's perception of being physically present in a non-physical world.
- Satisfaction: user's acceptance of the interaction system performance.

NASA-TLX (Hart and Staveland, 1988), a standard questionnaire was used to estimate the effort for the user self-assessment of the visiting task in VM. The addition of scores for these items gives the estimated workload score to investigate user perspective on each interaction

Table 5.2: Criteria of hypotheses and measurements

Experience	Hypothesis	Quantitative	Qualitative
- Virtual tour	- H1 (Usability hypothesis)	- Selection time - Manipulation time - Navigation time	- Ease of use
- Semi-guided story	- H2 (Needs of interaction hypothesis) - H3 (Learning hypothesis)	- Amount of interactions - Comprehension scores	- Immersion - Satisfaction
- Non-guided story	- H4 (Holding power hypothesis)	- Holding power	- Ease of use - Immersion - Satisfaction

system. These ratings are on a 7 level Likert scale (Likert, 1932) from very low (1) to very high (7). Table 5.2 summarizes our analysis grid of our experience.

5.4 Statistics analysis

In our experience, we will use data from all three sample groups from three interaction systems using different content to find out the performance difference. The average and deviation of the data from three interaction systems are compared to analyze the results.

For each experience, all three interaction systems will have three tested scenarios. Dependent samples occur on all scenarios are assessed three times. The participants' evaluations of each interaction system are influenced by the same user. We assume that the data of one sample affect the others. Therefore, type of sample will be considered to be dependent sample.

Statistical analysis methods (Figure 5.1) are adopted to compare statistical data of dependent samples. Before choosing experience testing, we need to examine if parametric assumptions are valid. It depends on distribution of samples data which affect the experience test selection. The Shapiro-Wilk test (Shapiro and Wilk, 1965) is used to check parametric assumption. If the data respect the normal distribution, 1-way ANOVA is used as a statistical model and to analyze the differences among group means in a sample. If the data do not respect a normal distribution, the Friedman test (Friedman, 1937) will be used. The Friedman test is used for one-way repeated measures analysis of variance by ranks. The parameters used to detect the difference between samples across multiple test attempts.

The post hoc analysis consists of analyses that are not specified before seeing the data. When there are significant differences between two samples, the post-hoc test is used to compare the difference of each condition. If 1-way ANOVA is applied, there are several different post-hoc analyses and Tukey HSD (Honestly Significant Difference) (Tukey, 1949) is used because we need to compare simultaneously all pairs of means. We compare the means of every sample to the means of every other sample. If Friedman test is applied, the Wilcoxon's test will be used for post-hoc analyses of samples with nonparametric approach (Derrick and White, 2017). This method compares two related samples and repeated measurements on a single sample to assess whether their populations mean ranks differ.

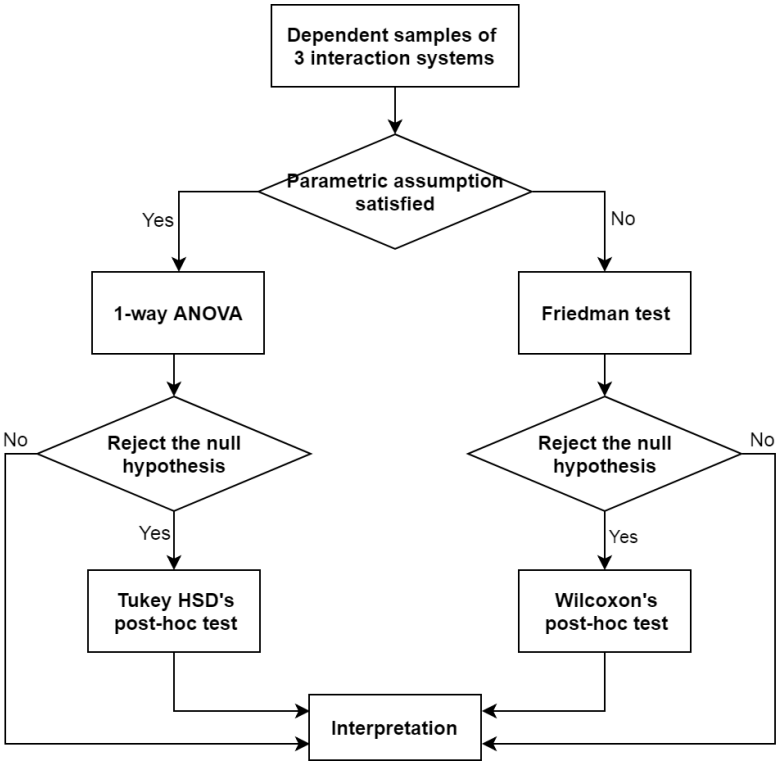

Figure 5.1: Statistical analyses of dependent sample data of three interaction systems

5.5 Virtual tour experience

The virtual tour experience was set up to investigate the performance of each interaction system following the criteria of usability hypothesis (H1). The quantitative measures are monitored to check the interaction time for navigation, manipulation and selection. In order to design VM application, we have to know the performance of interaction when a user performs assigned tasks. This experience also investigates the design of interaction techniques through the qualitative measurement of usability.

5.5.1 Introduction

We designed this experience by placing three buildings at a distance enabling participants to move from one building to another. Participants use first person perspective navigation mode to move in the virtual scene. When they enter into a building the application will change from navigation mode to manipulation mode. Participant manipulates the object to find hidden checked points using translation, rotation and scaling applying on the building. When a checked point is reached, the visited building disappears and the application returns to navigation mode. Participant needs to go to a new building until they have visited the three buildings.

5.5.2 Hypothesis

We will investigate the criteria related to interaction performance. These criteria are used to prove usability hypothesis (H1) with following sub-hypotheses.

- **H1:** Participants have different performance to accomplish the tasks on different interaction system.

Figure 5.2: The virtual tour experience on three interaction systems: 2D system, 3D system,and CAVE system from left to right

- **H1.1:** Participants have different navigation time to complete the mission.
- **H1.2:** Participants have different manipulation time to complete the mission.
- **H1.3:** Participants have different selection time to complete the mission.

5.5.3 Research protocol

We have three interaction systems the 2D Powerwall, the 3D stereoscopic display and the CAVE and each system follows the same protocol. Participants are invited to use all systems to accomplish the same mission, but there must find different checked points to avoid the bias of task repetition. Firstly, we spend around 5-10 minutes to introduce how to use devices to control an avatar for navigation, manipulation and selection. Then we launch the virtual tour application and explain the mission to the participant. At the end of the mission, users complete a questionnaire to provide a qualitative score for each system.

Before testing, participant is trained how to use devices and how to apply interaction techniques. The test scene with a simple building and terrain will be launched to let participant try to navigate with camera free mode at first. Then selection of objects and icons is introduced to see what happens when an object or an icon is selected. Some instructions appear to allow them to understand how to control the device for navigation, manipulation and selection. Then the first person navigation mode is enabled to let them test it. The scene allows user to get into the building to go to a manipulation mode and let them learn how to manipulate the object.

In the virtual tour experience, the first step explains the mission to participants. When the participant is aware the time recorder starts. Interactions are monitored to capture how long each device is used for navigation, manipulation and selection. We divide the experience into three groups to avoid results bias due to modality order.

At the end, each participant completes a questionnaire to assess the usability of each system. There are six questions of usability to accomplish the interaction techniques. The questionnaire will be evaluated separately to prevent them from comparing with the previous assessment scores. The questionnaire is used to assess user perception through a qualitative value corresponding to a usability evaluation: i.e., ease of using and learning interaction techniques.

5.5.4 Results

This section compares and discusses the performance of interaction of each system. This experience, involved 15 participants (11 males and 4 females) volunteers to take part in the study. The participants were aged from 23 to 42 (mean=29.87). All participants are right-handed except one, and all of them had an experience with mouse and keyboard and joystick controller. Five of them had already an experience with the Powerwall and the stereoscopic display. Six of them had an experience with the CAVE system. Eight of them had no experience with any device. Each five participants started with the 2D system, the 3D system and the CAVE, respectively, and test next system until they complete all systems.

5.5.4.1 Interaction time evaluation

Since we have dependent samples of three interaction systems, we need to test parametric assumption first. The Shapiro-Wilk, W test is used for normality testing and the data respect to normal distribution then 1-way ANOVA will be used to analyze the differences among group means in a sample.

Figure 5.3 shows the time spent for the virtual tour experience. Navigation time is quite similar for all three systems. While the CAVE system takes more manipulation time than

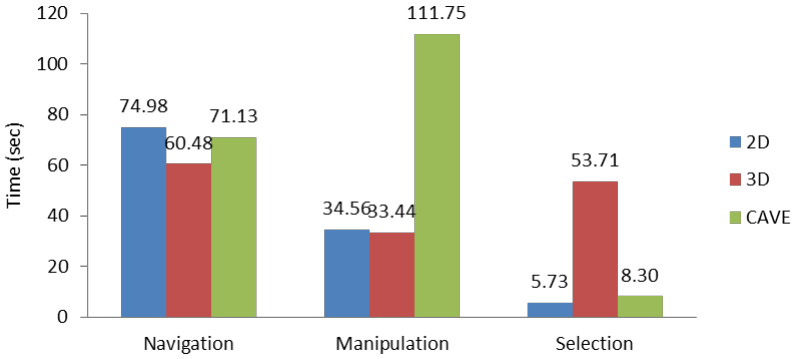

Figure 5.3: Interaction time average in the virtual tour experience

Table 5.3: Interaction time analysis of the virtual tour experience (* significant p<0.05, ** highly significant p<0.01)

	Navigation time (sec)	**Manipulation time (sec)**	**Selection time (sec)**
2D system	74.98	34.56	5.9
3D system	60.48	33.44	56.8
CAVE system	71.13	111.74	8.53
p-value	0.137787	5.38E-07	1.84E-07
Significant		**	**

other systems and the 3D system lead to higher selection time. The 1-way ANOVA is used to confirm this observation, results are shown in Table 5.3.

The results confirm that the navigation time is not significantly different (p=0.137787>0.05). The manipulation time is different with highly significant (p=5.38E-07<0.01) because the CAVE system obviously has a higher average of manipulation time. While the selection time is also different with highly significance (p=1.84E-07<0.01) because of 3D system has a higher average selection time. Tukey's post-hoc analysis is used to check which system makes manipulation time and selection time different. The results confirm that the difference among manipulation time is the CAVE system (Table B.1) and the difference among selection time is the 3D system (Table B.2). We accept H1.2 and H1.3, where H1.1 is rejected. Thus, H1 is significant and we conclude that participants have different performance to accomplish the tasks on different interaction system.

By this performance results, 3D system is the best for navigation and manipulation. While 2D system is the best for selection. However, there is something interesting on manipulation time and selection time where CAVE system and 3D system have poor performance. As the results of related work (Basset and Noël, 2018), CAVE system should be the best for manipulation. Then, the experience in the Section 5.7 is set up to investigate relevance of this result. The problem of selection time of 3D system, we can summarize by the results in the next subsection.

Table 5.4: Usability questionnaire result of 2D system in the virtual tour experience

No.	Usability questionnaires of 2D system	1	2	3	4	5	6	7	Avg. scores
1.	I understand clearly how to navigate in the scene.	0%	0%	0%	0%	6.67 %	46.67 %	46.67 %	6.4
2.	I found the navigation in the scene is easy to do.	0%	0%	0%	0%	0%	46.67 %	53.33 %	6.53
3.	I understand clearly how to select the model or icon.	0%	0%	0%	0%	0%	26.67 %	73.33 %	6.73
4.	I found the selection of the model or icon is easy to do.	0%	0%	0%	0%	6.67 %	20%	73.33 %	6.67
5.	I understand clearly how to manipulate the model.	0%	0%	0%	6.67 %	6.67 %	40%	46.67 %	6.27
6.	I found the manipulation of the model is easy to do.	0%	0%	0%	6.67 %	20%	26.67 %	46.67 %	6.13

Table 5.5: Usability questionnaire result of 3D system in the virtual tour experience

No.	Usability questionnaires of 3D system	1	2	3	4	5	6	7	Avg. scores
1.	I understand clearly how to navigate in the scene.	0%	0%	0%	6.67 %	26.67 %	33.33 %	33.33 %	5.93
2.	I found the navigation in the scene is easy to do.	0%	0%	13.33 %	13.33 %	20%	33.33 %	20%	5.33
3.	I understand clearly how to select the model or icon.	0%	0%	0%	6.67 %	20%	40%	33.33 %	6
4.	I found the selection of the model or icon is easy to do.	0%	6.67 %	13.33 %	6.67 %	26.67 %	13.33 %	33.33 %	5.27
5.	I understand clearly how to manipulate the model.	0%	0%	0%	6.67 %	20%	20%	53.33 %	6.2
6.	I found the manipulation of the model is easy to do.	0%	0%	6.67 %	6.67 %	26.67 %	13.33 %	46.67 %	5.87

Table 5.6: Usability questionnaire result of CAVE system in the virtual tour experience

No.	Usability questionnaires of CAVE system	1	2	3	4	5	6	7	Avg. scores
1.	I understand clearly how to navigate in the scene.	0%	0%	0%	13.33 %	20%	33.33 %	33.33 %	5.87
2.	I found the navigation in the scene is easy to do.	0%	6.67 %	13.33 %	13.33 %	13.33 %	13.33 %	20%	5.13
3.	I understand clearly how to select the model or icon.	0%	0%	0%	13.33 %	6.67 %	40%	40%	6.07
4.	I found the selection of the model or icon is easy to do.	0%	13.33 %	6.67 %	13.33 %	6.67 %	33.33 %	26.67 %	5.2
5.	I understand clearly how to manipulate the model.	0%	6.67 %	6.67 %	6.67 %	26.67 %	40%	13.33 %	5.27
6.	I found the manipulation of the model is easy to do.	6.67 %	20%	6.67 %	26.67 %	13.33 %	26.67 %	0%	4

Table 5.7: Usability analysis of the virtual tour experience (* significant $p<0.05$, ** highly significance $p<0.01$)

	Navigation understanding (max=7)	**Navigation easiness (max=7)**	**Manipulation understanding (max=7)**	**Manipulation easiness (max=7)**	**Selection understanding (max=7)**	**Selection easiness (max=7)**
2D system	6.4	6.53	6.27	6.13	6.73	6.67
3D system	5.93	5.33	6.2	5.87	6	5.27
CAVE system	5.87	5.13	5.27	4	6.07	5.2
p-value	0.21225	0.01713	0.0986	0.00155	0.12043	0.01742
Significant		*		**		*

5.5.4.2 Results of interaction usability questionnaire

Each answer was a seven-point Likert Scale from strongly disagree (1 point) to strongly agree (7 points). The average score for each question is the total scores divided by the total number of participants. The questions number 1 – 6 are about interaction usability. The results are shown in the Table 5.4 – 5.6, respectively.

The manipulation easiness score of CAVE system is significantly lower than the other values, which is consistent with the time spent in the experience from Figure 5.3. The average manipulation time of the CAVE system is higher than other systems. Tables 5.4 to 5.6 clearly demonstrate that the 2D system has the highest score and is thus perceived as the easiest system. The 3D system comes after and was perceived as better than the CAVE. This is confirmed by the summary in Table 5.7.

A Linkert Scale data is collected from the feedback of participants for analysis. Since the data do not respect to normal distribution, a non-parametric Friedman test is used to analyze the difference among the three systems. The understanding of navigation, manipulation and selection of the three systems is not different, which means users understood how to use each system (Table 5.7). By the way, in terms of easiness: navigation and selection are significantly different ($p=0.01713<0.05$ and $p=0.01742<0.05$, respectively), while manipulation is different with highly significance ($p=0.00155<0.01$). Although in the actual usage of the virtual tour experience in the Table 5.3 found that the navigation time is not different because of the user perspective of navigation of each system is different.

Wilcoxon's post-hoc analysis is used to check again which system is different from navigation, manipulation and selection easiness. For navigation easiness, 2D system is different with highest easiness score, where 3D system and CAVE system are not different (Table B.8). For manipulation easiness, CAVE system is different with lowest score, where 2D system and 3D system are not different (Table B.9). For selection easiness 2D system is the easiest, where 3D system and CAVE system are not different (Table B.10).

Here, we conclude that a part of manipulation and selection problems on the CAVE system and the 3D system is about the design of interaction techniques. Because users felt that the CAVE system was not easy to manipulate and the 3D system was not easy to select. Then we improve the design of interaction techniques and investigate another related factors in Section 5.7.

From the results of system performance, we explain into two points. The first point form user performance in the virtual tour experience, 3D system is the best for navigation and manipulation, where 2D system is the best for selection (Table 5.3). The second point from user perspective, 2D system is the best system with the highest usability score (Table 5.7) of selection, manipulation and navigation. The next section we study about immersion and

satisfaction when applied these interaction systems with a story.

5.6 Semi-guided story experience

The semi-guided story experience is set up to investigate the impact of interaction system on visitor point of interest (H2) and on visitor learning (H3). Quantitative measures count the number of actions of participants with each object. Each object in the scene has an attractiveness score defined by this number. Learning capacity is also assessed in this experiment by a post-questionnaire about the story comprehension after using each system. We have thus a qualitative measurement of participant satisfaction and immersion.

5.6.1 Introduction

As designed in Chapter 4, the story was divided into three scenes to assess every interaction system. Users use the different interaction systems following different sequence, but every interaction system has a different scene to run. The story is designed base on interactions, while animation is used to support story flow. We do not specify interaction points, participants must interact randomly on objects in the story.

5.6.2 Hypotheses

We investigate satisfaction, immersion and learnability. These criteria are used to prove the needs of interaction hypothesis (H2) and the learning hypothesis (H3) with following sub-hypotheses:

- **H2:** Participants have different needs of interaction when visiting on different interaction system.
  - **H2.1:** Participants have more interactions when the interaction system is easy to use.
  - **H2.2:** Participants have more interactions when the interaction system is immersive.
  - **H2.3:** Participants have more interactions when the interaction system is satisfied.
- **H3:** Participants learn differently when interacting on different interaction system.
  - **H3.1:** Participants have more comprehensions when the interaction system is easy to use.
  - **H3.2:** Participants have more comprehensions when the interaction system is immersive.
  - **H3.3:** Participants have more comprehensions when the interaction system is satisfied.

5.6.3 Research protocol

The three interaction systems (2D, 3D and CAVE) use the same protocol. The whole stories have three scenes. Each scene interacts on a different interaction system. Participant starts the story with the first scene on one of interaction system. At the end of the first scene, participant will move to the next interaction system and continue the next scene. When the second scene is finish, the last scene is launched on the last interaction system. Every participant is sequentially tested with all interaction systems. Hence, there are three groups

Figure 5.4: The semi-guided story experience on three interaction systems: 2D system, 3D system,and CAVE system from left to right

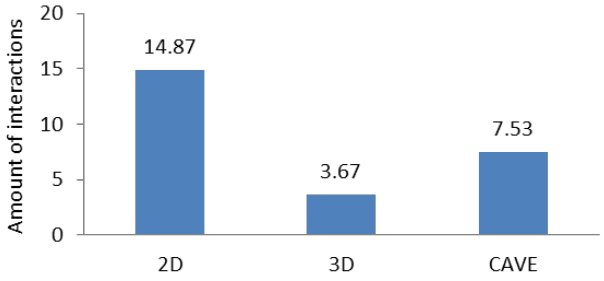

Figure 5.5: Average of amount of interactions in the semi-guided story experience

of participants, some start with the 2D, some with the 3D and some with the CAVE. All groups have the same number of participants to avoid the statistical bias.

Firstly, we introduce how to play/pause animation including how to jump to any frame on demand. The navigation mode is camera free. Participant stops whenever in the scene and moves camera to get closer and to interact with any object. Participant interaction is allowed whenever something interested to them. When an object is selected the animation automatically stops and needs to be played again to continue.

The semi-guided story shows the story of Wat Phra Mahathat Woramahawihan. Participants must understand the content presented through the interaction system. It is used to analyze the interaction between the system and the user, as well as the user's learning and the interaction with the different devices. Participant will start the first scene with a specific interaction system. The story will continue with the next interaction system until the end of the scene. Each interaction system spends around 2-3 minutes to play the full animation. Participants have no limit of time to interact with the scene until they are satisfied. When the experience starts, numbers of interactions with the scene are counted.

After every modality experience, participants answer questions about their comprehension of the story. Two questions are asked with multiple choices. If participants answer correctly, they get a +1 score. If they answer wrong, they get a -1 score. The last choice is "I don't know", participant can choose this choice if they cannot remember the answer or are unsure and they get a 0 score. The summations of scores are used to evaluate learning performance with the interaction system. Furthermore, participants must complete a questionnaire about immersion perception and corresponding satisfaction.

5.6.4 Results

In this experience, we used the same 15 participants as for the previous experience just continuing with the semi-guided story.

5.6.4.1 Interaction capturing evaluation

Participants used the three interaction systems. Then the measure of each scene and each interaction system were average. Collected data does not correspond to a normal distribution, thus, non-parametric Friedman test was used to analyze data differences.

The average of amount of interactions that user perform on different interaction system is shown on the Figure 5.5. Participants interacted on average 14.87 times in the 2D system, 3.67 times in the 3D system and 7.53 times in the CAVE system. The result of ANOVA shows the highly significance difference of interactions among the three systems ($p=0.000764<0.05$) and the Tukey's post-hoc analysis shows that the 2D system is different from others (Table B.3).

Table 5.8: Amount of interactions analysis in the semi-guided story experience (* significant p<0.05, ** highly significance p<0.01)

	Amount of interactions (average)
2D system	14.87
3D system	3.67
CAVE system	7.53
p-value	0.000764
Significant	**

Table 5.9: Immersion and satisfaction questionnaire result of 2D system in the semi-guided story experience

No.	**Immersion and satisfaction questionnaires of 2D system**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Avg. scores**
7.	I perceived the model in 3D by the quality of the display.	0%	0%	0%	13.33 %	33.33 %	40%	13.33 %	5.53
8.	My interactions with the virtual environment were responsive.	0%	0%	0%	6.67 %	26.67 %	33.33 %	33.33 %	5.93
9.	I felt stimulated by the virtual environment.	0%	0%	0%	33.33 %	13.33 %	33.33 %	20%	5.4
10.	I am not suffered from visual tiredness during my interaction with the VE.	0%	0%	0%	0%	13.33 %	33.33 %	53.33 %	6.4
11.	I am not suffered from physical tiredness during my interaction with the VE.	0%	0%	0%	0%	6.67 %	26.67 %	66.67 %	6.6
12.	I am not suffered from headache during my interaction with the VE.	0%	0%	0%	0%	13.33 %	26.67 %	60%	6.47

5.6.4.2 Post-questionnaire about immersion and satisfaction

The post-questionnaire about usability of interaction was the same Linkert Scale as the previous experience. The questions number 7 – 9 concerns immersive perception, while the questions number 10 – 12 are about system satisfaction. The results of post-questionnaire according to the participant are a subjective measure of the semi-guided story experience for 2D system, 3D system and CAVE system. They are shown in the Table 5.9 – 5.11, respectively.

5.6.4.3 Difference of interaction system

Each system is significantly different respect to the amount of interactions. We compare the results of the six questionnaires to examine the difference between systems. Immersion questions ask 3D perception, interaction responsive and VE stimulation. The satisfaction questions are about visual tiredness, physical tiredness and headache. The ease of use score we collected from virtual tour experience. We investigate which characteristics make the interaction system different.

The result of Friedman test can summarize that all questions of immersion are not significant which mean user perspective of 3D perception, interaction responsive and VE stimulation are not different. By the way, all questions of satisfaction are strongly significant on the visual tiredness ($p=0.00158<0.01$), the physical tiredness ($p=0.00402<0.01$), and the headache suffering question is significant ($p=0.00543<0.01$). Wilcoxon's post-hoc analysis shows that 2D system has different visual tiredness and physical tiredness (Table B.11, Table B.12).

Table 5.10: Immersion and satisfaction questionnaire result of 3D system in the semi-guided story experience

No.	Immersion and satisfaction questionnaires of 3D system	1	2	3	4	5	6	7	Avg. scores
7.	I perceived the model in 3D by the quality of the display.	0%	0%	0%	6.67 %	40%	33.33 %	20%	5.67
8.	My interactions with the virtual environment were responsive.	6.67 %	0%	0%	20%	20%	33.33 %	20%	5.27
9.	I felt stimulated by the virtual environment.	0%	6.67 %	0%	26.67 %	13.33 %	40%	13.33 %	5.2
10.	I am not suffered from visual tiredness during my interaction with the VE.	13.33 %	6.67 %	26.67 %	20%	6.67 %	13.33 %	13.33 %	3.93
11.	I am not suffered from physical tiredness during my interaction with the VE.	13.33 %	0%	6.67 %	6.67 %	33.33 %	6.67 %	33.33 %	5
12.	I am not suffered from headache during my interaction with the VE.	6.67 %	6.67 %	20%	13.33 %	13.33 %	20%	20%	4.6

Table 5.11: Immersion and satisfaction questionnaire result of CAVE system in the semi-guided story experience

No.	Immersion and satisfaction questionnaires of CAVE system	1	2	3	4	5	6	7	Avg. scores
7.	I perceived the model in 3D by the quality of the display.	0%	6.67 %	0%	0%	20%	46.67 %	26.67 %	5.8
8.	My interactions with the virtual environment were responsive.	6.67 %	6.67 %	6.67 %	6.67 %	6.67 %	46.67 %	20%	5.2
9.	I felt stimulated by the virtual environment.	0%	0%	13.33 %	0%	13.33 %	46.67 %	26.67 %	5.73
10.	I am not suffered from visual tiredness during my interaction with the VE.	0%	6.67 %	6.67 %	26.67 %	6.67 %	33.33 %	20%	5.13
11.	I am not suffered from physical tiredness during my interaction with the VE.	0%	0%	13.33 %	26.67 %	6.67 %	26.67 %	26.67 %	5.27
12.	I am not suffered from headache during my interaction with the VE.	0%	0%	6.67 %	20%	20%	20%	33.33 %	5.53

Table 5.12: Immersion and satisfaction analysis of the semi-guided story experience (* significant $p<0.05$, ** highly significance $p<0.01$)

	3D perceiving (max=7)	Responsive interaction (max=7)	VE stimulated (max=7)	Visual tiredness (max=7)	Physical tiredness (max=7)	Headache (max=7)
2D system	5.53	5.93	5.4	6.4	6.6	6.47
3D system	5.67	5.27	5.2	3.93	5	4.6
CAVE system	5.8	5.2	5.73	5.13	5.27	5.53
p-value	0.72857	0.2865	0.38674	0.00158	0.00402	0.00543
Significant				**	**	**

Table 5.13: Usability of ease of use, immersion and satisfaction analysis of the experience (* significant p<0.05, ** highly significance p<0.01)

	Ease of use (max=7)	**Immersion (max=7)**	**Satisfaction (max=7)**
2D system	6.39	5.33	6.41
3D system	5.76	5	3.93
CAVE system	5.20	5.07	4.93
p-value	0.00026	0.2865	0.00053
Significant	**		**

Table 5.14: Regression analyses between user perspective and amount of interactions in the semi-guided story (* significant p<0.05, ** highly significance p<0.01)

Regression statistics	**Ease of use**	**Immersion**	**Satisfaction**
Multiple R	0.4442	0.4905	0.2926
MS	3.9117	9.3184	5.3014
F	6.1462	7.9222	2.3413
p-value	0.0203	0.0094	0.1385
Significant	*	**	

For headache score, all systems are different (Table B.13).

We considered results by grouping the questions about ease of use, satisfaction or immersion. Each question was used to collect average values for each system and use Friedman test again to see the statistical data. According to the result in the Table 5.13, ease of use and satisfaction scores are highly significance (p=0.00026<0.01 and p=0.00053<0.01, respectively). On the other hand, the immersion scores are not significant. Post-hoc analysis also shows that all systems have different on ease of use (Table B.14), while satisfaction is almost different where 3D system and CAVE system quite the same (Table B.15). Therefore, the ease of use and satisfaction are the keys of user perspective that make the interaction system different.

5.6.4.4 Stimulation of interaction

Ease of use and satisfaction are the significant key difference between the interaction systems by user perspective. Then we examine what stimulates the interaction with a story. We use the same variables: ease of use, satisfaction and immersion. Then we compare the amount of interactions participants have in the semi-guided story experience. The relationships between the three variables compared to the amount of interactions are shown in Figure 5.6. The regression statistics are used to find correlation between each variable. Results are shown on Table 5.14. The result of regression statistics on the Table 5.14 shows that ease of use is significant to the amount of interactions (p=0.0203<0.05), while satisfaction is not significant. In contrast, the immersion becomes highly significance to the amount of interactions (p=0.0094<0.01).

We conclude that H2.1 is significant, participants have more interactions when the interaction system is easy to use. Thus, we accept hypothesis H2, participant have different needs of interaction when visiting on different interaction system.

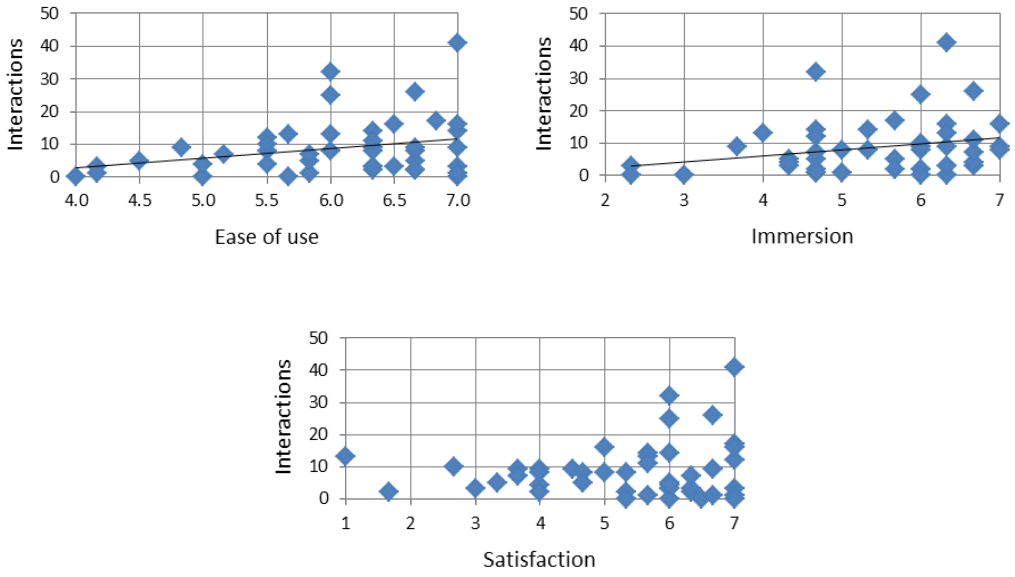

Figure 5.6: Relation between user perspective and amount of interactions in the semi-guided story experience

5.6.4.5 Content learning

Ease of use and immersion impact to the amount of interactions. We examine what factors between ease of use, satisfaction and immersion impact user learning by analyzing the answers to the two comprehension questions of the story. Each system will have scores from the participants. If both answers are correct, they will get a +2 score. If both answers are incorrect, they will get a -2 score. They can answer "I do not know" and get a 0 score, thus, the score of each system can be -2, -1, 0, 1, and 2. From the scores, we compare and find a relationship between the comprehensive questionnaire score and user perspective about ease of use, satisfaction and immersion on each system. The relationship between the scores and user perspective sorted from low to high value as shown in Figure 5.7.

The results of the regression analysis in the Table 5.15 show that there is no correlation between the ease of use, satisfaction and immersion that impact the comprehension scores. This means that user perspectives do not affect the understanding of the content. Moreover, when we compare the scores with the amount of interactions there is no significant relation. According to this result it implies that the comprehension scores are independent from the interaction system. We conclude that H3.1, H3.2 and H3.3 are not significant. Thus, we reject the hypothesis H3, participants do not learn differently when interacting on different interaction system.

5.6.4.6 The design of VE

The development of the semi-guided story experience, we designed interaction with objects into the scene following the story and information to users. However, in the actual usage, the user may have a different interaction from the design. We should capture the user behavior, how often do users interact with each object within the designed scene. If the design does not meet the needs of the user, it will know which object or location should be updated to meet the needs of the user.

The storytelling platform is designed to support this idea. We kept all interaction data

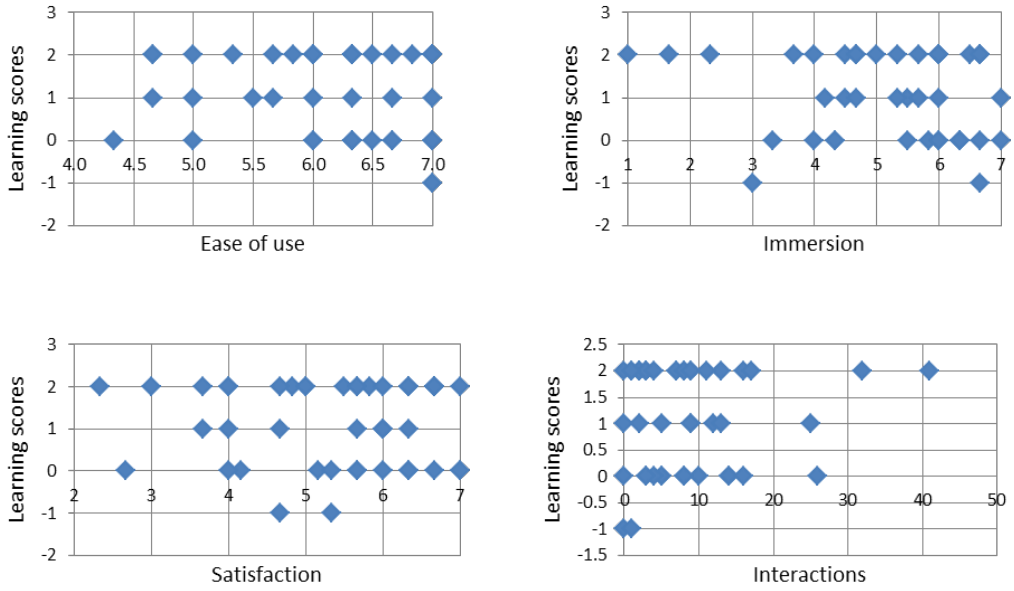

Figure 5.7: Relation between user perspective and comprehension scores in the semi-guided story experience

Table 5.15: Regression analyses between user perspective and comprehension scores in the semi-guided story experience (* significant $p<0.05$, ** highly significance $p<0.01$)

Regression statistics	**Ease of use**	**Immersion**	**Satisfaction**	**Amount of interactions**
Multiple R	0.3488	0.3239	0.0018	0.2717
MS	2.4111	4.062	0.000206	150.9449
F	3.4619	2.9295	0.0001	1.9933
p-value	0.0746	0.0994	0.9928	0.1703
Significant				

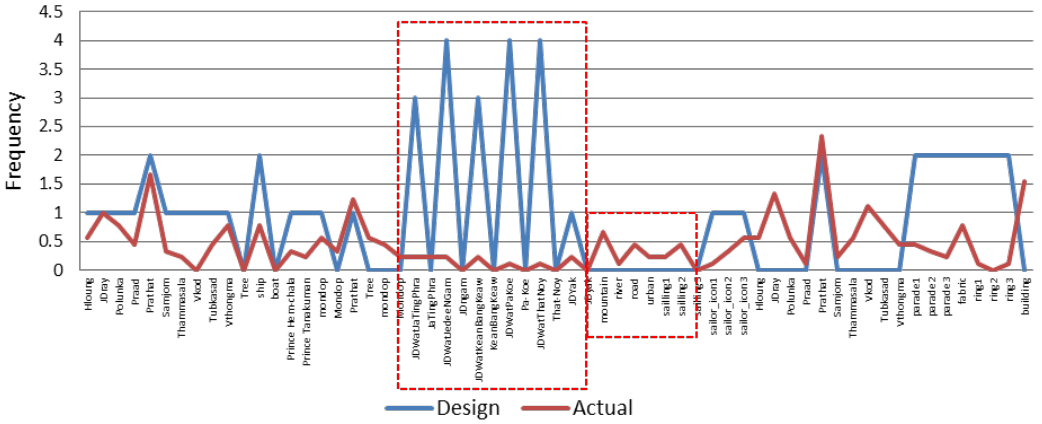

Figure 5.8: Object interactions from actual usage compares with the interaction design of the semi-guided story experience

which can make a report to the developer after user finished using the system. The report will save the data of amount of interactions on the scene and where is the interesting point for the user. Virtual museum developers use this report to improve their design for more attractive and interactive meet to the needs of users.

The Figure 5.8 shows the interaction design within the scene compared to the actual usage. It is found that some object interactions are quite different. Within the large dashed rectangle, we proposed large number of interactions, but the user interacts rather less. May be the arrangement is not interesting or the story in this part does not appeal to the user. Within the small dashed rectangle, we do not propose any interaction, but many users expect more information by trying to interact on objects. From the user feedback, they are interested in the mountain and river and want to know the name and information about these objects. This report addresses the needs of users and can be used to improve the design of interactions for user benefit. VR becomes a user centered design environment for VM.

From previous results, we accept H2 and reject H3 these also effect to the design of VE. In order to let users have more participations, interaction system should easy to use. Although, it is not affected to comprehension of a story, but the design of VE affects user attention.

5.7 Non-guided story experience

The non-guided story experience is set up to validate the results from the virtual tour experience about the usability hypothesis (H1). Observation of the animation in the semi-guided story experience is quite far to the scene leading to the immersive scores. A new controller and interaction techniques for the CAVE system was designed to improve system usage for manipulation during the virtual tour experience. This new controller is shown in Figure 5.9.

In this experience, the observer is inside the scene with a large scale rendering to increase the impact on immersion. The holding power is measured by amount of time the user uses the system. More time means more power to hold the user within the content. Time spent for both navigation and manipulation are captured to check the holding power (H4) of each interaction system. Each object in the scene is still monitored to count the amount of interactions and compare the different interaction systems.

Because immersion perception is checked we decided to compare the CAVE (assumed to be the immersion device), a wide Powerwall where gaze is fully included in the view angle and

a single 2D display (Figure 5.10).

Figure 5.9: New controller and interaction techniques are applied to use with the CAVE system

5.7.1 Introduction

The non-guided scene of Chapter 4 is used. We set up interactions into the scene for the different interaction systems. We have designed the scene that contains artifacts with interaction. Users are able to select artifacts they are interested to focus on by selection and get more expected information. In this experience, we do not fix time limit, thus users are allowed to interact freely.

5.7.2 Hypothesis

We investigate immersion and interaction time. These criteria are used to prove the holding power hypothesis (H4) with the following sub-hypotheses.

- **H4:** The visit time depends on the interaction system. A short time could demonstrate modality efficiency, but a long time may be due to a high interest.
  - **H4.1:** Participants have more navigation time when the interaction system is more immersive.
  - **H4.2:** Participants have more manipulation time when the interaction system is more immersive.

5.7.3 Research protocol

From virtual tour experience, we were not confident on the results of the 2D system and the CAVE system, where manipulation time of 2D system better than CAVE system. And also from semi-guided story experience, the VEs of the story was presented quite far from user point of view which affected to user immersion. Here, we set up a new research protocol to investigate interaction time and immersion of user.

Three interaction systems: 2D with a single display full HD 55”, the 9 screens Powerwall, and the CAVE are used. Participants are invited to use all systems for interacting with the scene. Participants test every system with the same story. Since the order of use has an effect on the experience, every participant tests the environments on different order.

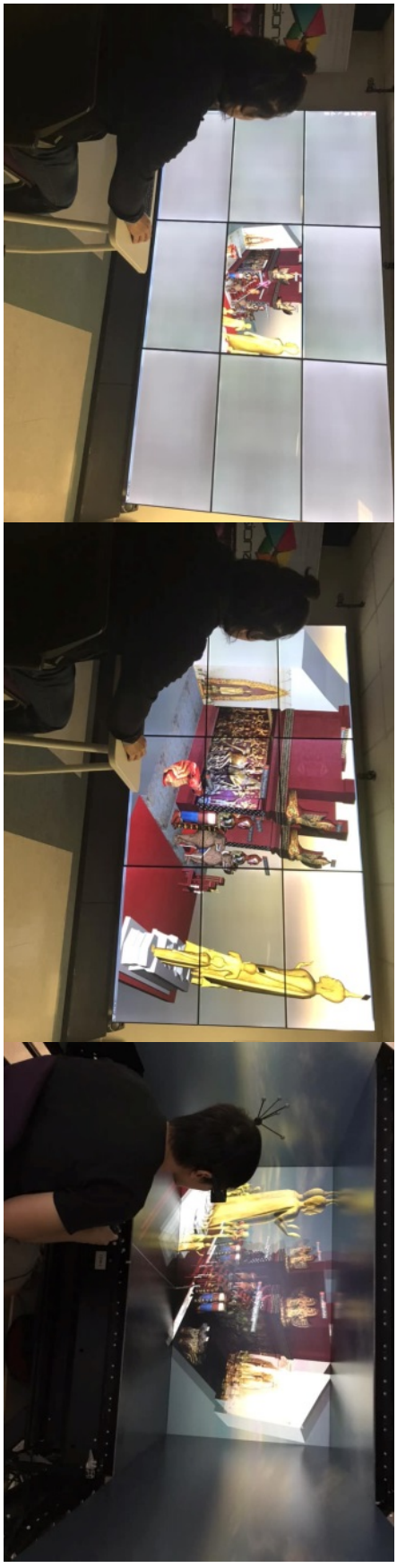

Figure 5.10: The non-guided experience on 2D single screen system, 2D full screen system and CAVE system

We introduce how to control navigation and how to select artifacts in the scene. In the non-guided story experience, the navigation mode is camera free. Participant can change point of view as they wish and can select whatever artifact they are interesting in the scene. This experience has no time limit, and participants can stop at any time.

The non-guided story shows story inside the Vihara Thongma, one of the buildings in the Wat Phra Mahathat Woramahawihan, which depicts various gods associated with artifacts. In the semi-guided story, participants are allowed to watch animation without interaction. However, they can interact to the scene if they are interested in something. In contrast, non-guided story needs interaction to drive the story. Participants need to interact with various artifacts within the scene to obtain information. Hence, they are forced to use the interaction system.

After every interaction system experience, participant is asked the same questionnaire as previous experiences. There are 12 questions about usability, satisfaction and immersion. By the end of the experience, participant is asked to sort the three interaction systems according to their usage of ease of use, satisfaction and immersion.

We included a script to record the following objective measures:

- Total: all time spent for each experience.
- Navigation time: travel time on the scene.
- Manipulation time: the time spent of each manipulation task on the object.
- Interaction of navigation time: the time spent of interaction of navigation task only.
- Interaction of manipulation time: the time spent of interaction of manipulation task only.
- Amount of interactions: the number of actions that user interacts with the artifact in the scene.

The questionnaire is used to assess user perception about:

- Usability: ease of using and learning.
- Immersion: ability of the system to immerse user into VE.
- Satisfaction: user's acceptance of system using.

5.7.4 Results

This section compares and discusses the holding power of each system. In this experience, we have 15 participants (11 males and 4 female) volunteered to take part in the study. The participants were aged from 22 to 36 (mean = 28.93). Twelve participants are right-handed and three participants are left-handed. Each participant will test each system until they complete all systems.

5.7.4.1 Interaction time evaluation

Here again we have dependent samples of three interaction systems. We need to test parametric assumption first by W test for normality testing. Data respects normal distribution then 1-way ANOVA will be used to analyze the differences among group means in a sample.

Figure 5.11 shows the interaction times for the non-guided story experience (Nav-time = total navigation time, Mani-time = total manipulation time, Nav-int = interaction time of navigation, Mani-int = interaction time of manipulation). The CAVE system seems to get more interest. The experience compares all time spent, navigation time, manipulation time, interaction of navigation time, and interaction of manipulation time. The experience

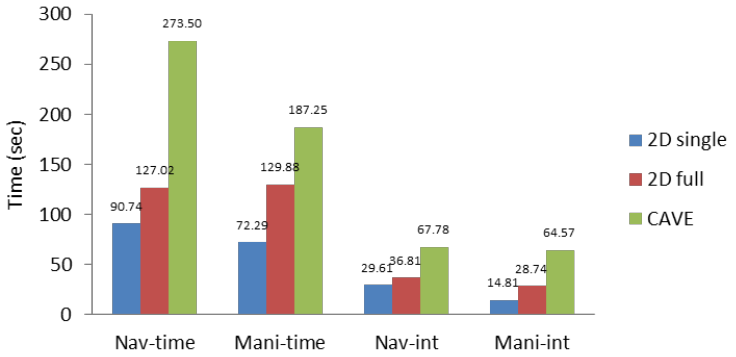

Figure 5.11: Interaction time average in the non-guided story experience

Table 5.16: Interaction time analysis of the non-guided story experience (* significant $p<0.05$, ** highly significance $p<0.01$)

	Nav-time (sec)	**Mani-time (sec)**	**Nav-int (sec)**	**Mani-int (sec)**
2D single screen	90.74	72.29	29.61	14.81
2D full screen	127.02	129.88	36.81	28.74
CAVE system	273.50	187.25	67.78	64.57
F	12.05775	7.134847	3.530768	14.82644
F crit	3.219942	3.219942	3.219942	3.219942
p-value	7.28E-05	0.00215	0.038254	1.34E-05
Significant	**	**	*	**

shows that the interaction time of the CAVE system is more than the 2D full screen system and the 2D single screen system, respectively. The 1-way ANOVA is used to confirm this observation, results are shown in Table 5.16.

The 1-way ANOVA is used to test the null hypothesis that the means of several interaction times are all equal. Table 5.16 shows for all interaction times F>F crit and p-value<0.01 (except Nav-int p-value<0.05), we reject the null hypothesis. The results confirm that all interaction times of each system are significantly different. These systems affect user behaviors for navigation time and manipulation time. And users pass more time in the CAVE than on the Powerwall and than the single display. Tukey's post-hoc analysis is used again and confirms that CAVE system is different from others (Table B.4-Table B.7). Thus, we accept H4, participants have more interaction time when the system is more immersive.

5.7.4.2 Amount of Interactions

Figure 5.12 shows the amount of interactions on the different systems in the non-guided story experience. There are 21 artifacts in the scene and participants have in average 7.87 interactions on the 2D single screen system, 11.33 interactions on 2D full screen system and 9.4 interactions on CAVE system. We found that the amount of interactions of 2D full screen is the highest. Then we will investigate what are the factors of difference of the amount of interactions and the corresponding interaction time.

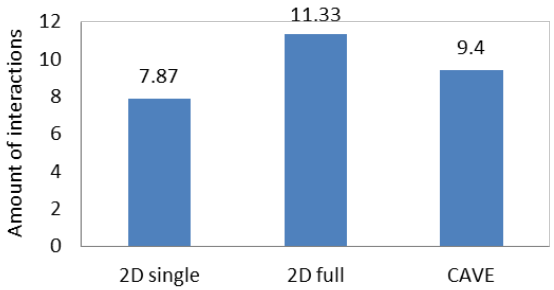

Figure 5.12: Amount of interactions of each system in the non-guided story experience

5.7.4.3 User perception assessment

There are twelve questions as in the previous experiences using a seven-point Likert Scale from strongly disagree (1 point) to strongly agree (7 points). The questions 1 – 6 ask about usability of interaction, questions 7 – 9 ask about immersion, and questions 10 – 12 ask about satisfaction. The results of the post-questionnaire according to the participant's perspective about the non-guided experiences are shown in Table 5.17 to 5.19.

From the Table 5.17, the results of the 2D single screen system show that the participants rated the highest score of satisfaction, but this system has the lowest immersion score as well. From the Table 5.18, the results of the 2D full screen system show that the participants rated the usability score highest, but not much difference from the 2D single screen system. However, the satisfaction score is slightly less than the 2D single screen system, but the average of the immersion score has increased. The Table 5.19 shows the results of the CAVE system that the participants rated the score of usability and satisfaction less than other systems, but it comes with the highest immersion score.

The Linkert Scale data does not respect the normal distribution, thus, a non-parametric Friedman test is used again to analyze the difference of usability, immersion and satisfaction among the three systems. The Table 5.20 demonstrates that the usability of the all systems is not so different just only navigation and selection easiness have some difference ($p=0.02599<0.05$, $p=0.04736<0.05$). That means the users understand how to use system and their perspective of system usage is not different. Especially, when we improve manipulation techniques on the CAVE system, user does not feel different compared to other systems. However, Wilcoxon's post-hoc analysis shows that CAVE system is different from others on navigation (Table B.16) and selection easiness (Table B.17). By the way, immersion on 3D perceiving and VE stimulated are significantly different ($p=0.01057<0.05$, $p=0.01925<0.05$) and also satisfaction on visual tiredness and headache are highly significance different ($p=0.00329<0.01$, $p=0.00221<0.01$), only interaction responsive and physical tiredness that users feel no different.

Relationship between immersion and interaction time is investigated by regression analysis. The results are shown in Table 5.22, Nav-time and Mani-time are significant related to immersion. Then we accept H4.1 and H4.2.

The post-hoc analysis shows that 3D perceiving of 2D single screen system is different with the lowest scores (Table B.18), while the CAVE system makes the VE stimulated, visual tiredness and headache scores different from others (Table B.19 – Table B.21). This implies that immersion and satisfaction affect the interaction usage of each system. Users have different interaction times and amount of interactions, which are related to the immersion and satisfaction capacity.

Table 5.17: Usability questionnaire result of 2D single screen system in the non-guided story experience

No.	Questionnaires and results of 2D single screen system	1	2	3	4	5	6	7	Avg. scores
1.	I understand clearly how to navigate in the scene.	0%	0%	0%	0%	0%	27%	73%	6.73
2.	I found the navigation in the scene is easy to do.	0%	0%	0%	0%	13%	33%	53%	6.4
3.	I understand clearly how to select the model or icon.	0%	0%	0%	0%	13%	20%	67%	6.53
4.	I found the selection of the model or icon is easy to do.	0%	0%	0%	0%	13%	40%	47%	6.33
5.	I understand clearly how to manipulate the model.	0%	0%	0%	0%	0%	47%	53%	6.53
6.	I found the manipulation of the model is easy to do.	0%	0%	7%	0%	13%	33%	47%	6.13
7.	I perceived the model in 3D by the quality of the display.	0%	7%	13%	20%	13%	40%	7%	4.87
8.	My interactions with the virtual environment were responsive.	0%	7%	13%	7%	7%	33%	33%	5.47
9.	I felt stimulated by the virtual environment.	0%	13%	13%	20%	20%	13%	20%	4. 7
10.	I am not suffered from visual tiredness during my interaction with the VE.	0%	0%	0%	7%	0%	20%	73%	6.6
11.	I am not suffered from physical tiredness during my interaction with the VE.	0%	0%	0%	13%	0%	13%	73%	6.47
12.	I am not suffered from headache during my interaction with the VE.	0%	0%	0%	7%	0%	20%	73%	6.6

Table 5.18: Usability questionnaire result of 2D full screen system in the non-guided story experience

No.	Questionnaires and results of 2D full screen system	1	2	3	4	5	6	7	Avg. scores
1.	I understand clearly how to navigate in the scene.	0%	0%	0%	0%	0%	27%	73%	6.73
2.	I found the navigation in the scene is easy to do.	0%	0%	0%	0%	7%	40%	53%	6.47
3.	I understand clearly how to select the model or icon.	0%	0%	0%	0%	0%	33%	67%	6.67
4.	I found the selection of the model or icon is easy to do.	0%	0%	0%	0%	7%	33%	60%	6.53
5.	I understand clearly how to manipulate the model.	0%	0%	0%	0%	7%	33%	60%	6.53
6.	I found the manipulation of the model is easy to do.	0%	7%	0%	7%	13%	33%	40%	5.87
7.	I perceived the model in 3D by the quality of the display.	0%	0%	20%	0%	13%	40%	27%	5.53
8.	My interactions with the virtual environment were responsive.	0%	0%	13%	0%	7%	53%	27%	5.8
9.	I felt stimulated by the virtual environment.	0%	7%	13%	13%	27%	20%	20%	5
10.	I am not suffered from visual tiredness during my interaction with the VE.	0%	0%	0%	0%	13%	33%	53%	6.4
11.	I am not suffered from physical tiredness during my interaction with the VE.	0%	0%	0%	7%	7%	20%	67%	6.47
12.	I am not suffered from headache during my interaction with the VE.	0%	0%	0%	0%	0%	47%	53%	6.53

Table 5.19: Usability questionnaire result of CAVE system in the non-guided story experience

No.	Questionnaires and results of CAVE system	1	2	3	4	5	6	7	Avg. scores
1.	I understand clearly how to navigate in the scene.	0%	0%	0%	7%	13%	53%	27%	6
2.	I found the navigation in the scene is easy to do.	0%	0%	0%	13%	47%	27%	13%	5.4
3.	I understand clearly how to select the model or icon.	0%	0%	0%	7%	20%	40%	33%	6
4.	I found the selection of the model or icon is easy to do.	0%	0%	0%	13%	27%	40%	20%	5.67
5.	I understand clearly how to manipulate the model.	0%	0%	0%	7%	27%	53%	13%	5.73
6.	I found the manipulation of the model is easy to do.	0%	0%	0%	13%	40%	20%	27%	5.6
7.	I perceived the model in 3D by the quality of the display.	0%	0%	7%	0%	13%	13%	67%	6.33
8.	My interactions with the virtual environment were responsive.	0%	0%	7%	13%	13%	47%	20%	5.6
9.	I felt stimulated by the virtual environment.	0%	0%	0%	0%	20%	33%	47%	6.27
10.	I am not suffered from visual tiredness during my interaction with the VE.	0%	0%	13%	7%	40%	33%	7%	5.13
11.	I am not suffered from physical tiredness during my interaction with the VE.	0%	0%	7%	7%	20%	27%	40%	5.87
12.	I am not suffered from headache during my interaction with the VE.	0%	0%	7%	20%	13%	53%	7%	5.33

Table 5.20: Usability analysis of the non-guided story experience (* significant $p<0.05$, ** highly significance $p<0.01$)

	Navigation understanding (max=7)	Navigation easiness (max=7)	Manipulation understanding (max=7)	Manipulation easiness (max=7)	Selection understanding (max=7)	Selection easiness (max=7)
2D single screen	6.73	6.4	6.53	6.13	6.53	6.33
2D full screen	6.73	6.47	6.53	5.87	6.67	6.53
CAVE system	6	5.4	5.73	5.6	6	5.67
p-value	0.08629	0.02599	0.0598	0.19856	0.15464	0.04736
Significant		*				*

Table 5.21: Immersion and satisfaction analysis of the non-guided story experience (* significant $p<0.05$, ** highly significance $p<0.01$)

	3D perceiving (max=7)	Responsive interaction (max=7)	VE stimulated (max=7)	Visual tiredness (max=7)	Physical tiredness (max=7)	Headache (max=7)
2D single screen	4.87	5.47	4.67	6.6	6.46	6.6
2D full screen	5.53	5.8	5	6.4	6.47	6.53
CAVE system	6.33	5.6	6.27	5.13	5.87	5.33
p-value	0.01057	0.76593	0.01925	0.00329	0.63763	0.00221
Significant	*		*	**		**

Table 5.22: Regression analyses between interaction time and immersion in the non-guided story (* significant p<0.05, ** highly significance p<0.01)

Regression statistics	Nav-time	Mani-time
Multiple R	0.1923	0.1803
MS	2.6675	2.3439
F	1.6511	1.4440
p-value	0.0206	0.0236
Significant	*	*

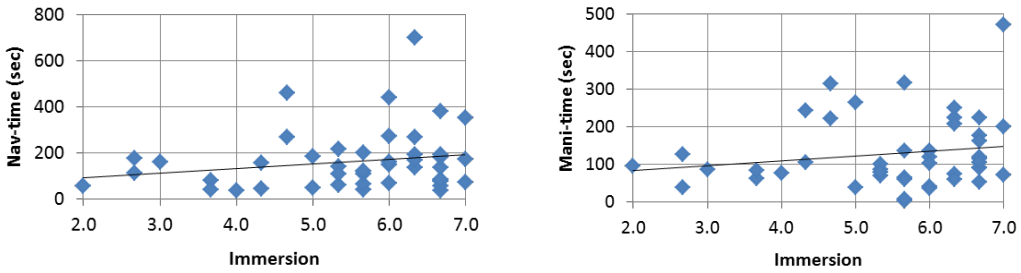

Figure 5.13: Relation between interaction time and immersion in the semi-guided story experience

5.8 Results and conclusions

5.8.1 Results

From the results of the virtual tour experience, we found that in the actual usage in the virtual tour experience. Manipulation time and selection time of each system significantly difference and we accept H1.2 and H1.3. While navigation time is not different, therefore, we reject H1.1. However, the results of user perspective in Table 5.24 we found that users understand how to interact on each system with no difference, while they feel that ease of use with each system is different whether navigation, manipulation or selection. This implies that user feeling on navigation easiness is different, but in actual usage navigation time is not different.

From the results of the semi-guided story experience, we found that in the actual usage the amount of interactions significantly difference. The results from Table 5.14 show that ease of use and immersion are significantly related to number of interactions, thus, we accept H2.1 and H2.2. Especially, the 2D system is the highest number of interactions which is assessed the easiest to use and the CAVE system is assessed the most immersive. However, we found that satisfaction is not affected to the number of interactions, thus, H2.3 is not significant.

By the way, based on the results from Table 5.13, the immersion of the all systems is not different. This influenced to the results in Section 5.6.4.4 which found that whenever the users feel the system is immersive, no matter which system is used, they will have a greater amount of interactions considered by user perspective. For example 2D system is not the most immersive system, but with the wide screen may influence user interaction. Another point is about user satisfaction. Users felt that the satisfaction of each system is different, but it does not make a greater number of interactions.

Table 5.23: Results of experiences following all hypotheses (✓ accept, ✗ reject)

H1: Interaction performance			H2: Needs of interaction			H3: Learning			H4: Holding power	
H1.1	H1.2	H1.3	H2.1	H2.2	H2.3	H3.1	H3.2	H3.3	H4.1	H4.2
✗	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓

Table 5.24: Results of user perspective on usability in the virtual tour and semi-guided story experience (✓ different, ✗ not different)

	Navigation	Manipulation	Selection
Understanding	✗	✗	✗
Ease of use	✓	✓	✓
	3D perceiving	Responsive interaction	VE stimulated
Immersion	✗	✗	✗
	Visual tiredness	Physical tiredness	Headache
Satisfaction	✓	✓	✓

The results of the study from Table 5.15 show that ease of use, comfortable and immersion are not relevant to learning, which makes H3.1, H3.2 and H3.3 are not significant. Even the amount of interactions is not related to learning scores following the results in Fig 5.7.

From the results of the non-guided story experience, we found that the interaction time of the CAVE system is more than the 2D full screen system and the 2D single screen system. All interaction times are different with highly significance, which means different systems are affected to holding power in order to use the system. Moreover, in the Table 5.22, we found that immersion is related to navigation time and manipulation time, which confirms our hypotheses (H4.1 and H4.2).

5.8.2 Conclusions

From the results, users spent the same interaction time on average when they feel that the system is easy to use as same as the system they used to (2D system). If we use appropriate input device and good interaction techniques, each interaction system will take approximately the same interaction time. For example, the navigation time from the experience in Section 5.5 shown that the three systems were not significantly different because good design of navigation technique. Good performance of interaction system will take less interaction time, unless the system is capable of attracting users for longer than usual. When users feel that the system is easy to use and has a high immersion level, they will have more interactions with the system, whether they are comfortable or not.

For the development of VMs that need to interact with users, it should be designed to be easy to use and high level of immersion in the first priority. For the design of the VM content, it will affect the user perception when used on different interaction systems. Therefore, we should design the content consistent with the selected system. We will summarize the interaction systems used in the experiment.

The 2D Powerwall system has a large display that is ideal for multi-user applications. Content should focus on interacting with the system by selection. Because of its huge screen and wide range of view that is the immersion feature. Despite the huge wide screen, having a two-dimensional display makes it easy to interact with the selection. However, with two-dimensional display, users lack perception of the depth of field, resulting in loss of immersion in this issue.

Table 5.25: Results of user perspective on usability in the non-guided story experience (✓ different, ✗ not different)

	Navigation	Manipulation	Selection
Understanding	✗	✗	✗
Ease of use	✓	✗	✓
	3D perceiving	Responsive interaction	VE stimulated
Immersion	✓	✗	✓
	Visual tiredness	Physical tiredness	Headache
Satisfaction	✓	✗	✓

The 3D stereoscopic system has a large three-dimensional projection that can be viewed by multiple users at the same time with shutter glass, but the visibility of each user may be slightly difference by position. The advantage is the VE can be visualized in 3D and user perceives depth of field, which increases immersion in usage. Hence, the content should focus on 3D presentation where user can interact from distance. However, when the content is presented in 3D, it makes selection and manipulation more difficult than two dimensions because users are more experienced and used to interactivity in two-dimensional systems. It depends on the device and the interaction techniques used. In Section 5.5, we found that users spend more time in selection and manipulation in 3D system because of the depth of field is added. It is not easy to move the selector in a 3D scene by a joystick controller, while tracking device should be better to move a selector in 3D for selection and manipulation. In contrast, navigation time with joystick controller is not different from the 2D system, which means the interaction design is appropriate.

The CAVE system uses multiple immersive three-dimensional displays. Wider volume and viewing make it possible to perceive fully immersive VE, but it cannot be used by multiple users simultaneously due to individual tracking system. The content should be a three-dimensional content that users can access close to objects, such as first person mode navigation. The results show that user perception of immersion is not different when presenting VE from a far distance (from Table 5.13). By the way, using VE close to a user, immersion is significantly different (from Table 5.21). Therefore, the CAVE system provides the better immersion at the condition to be in first person mode in scale very close to objects without these conditions a 2D system provides a better immersion. Another advantage of the CAVE system is user attraction or holding power. In Section 5.7, we found that users spend most of their time interacting with the CAVE system. In this experience, we improved the interaction techniques and the new design was easier to use (from Table 5.20, usability of the CAVE system is better). This is consistent with the results of Section 5.5 (Figure 5.3) that users spend more time interacting than normal, which is not due to poor performance, but the system is attractive to users for longer using than usual. This is good for content that requires user attention for a long time. However, high immersion values are associated with lower satisfaction because users are uncomfortable when using the CAVE system too long.

Chapter 6

Conclusions and Perspectives

In this chapter, we will summarize the results of our research starting from the research question at Section 6.1. Then in the Section 6.2, we remind the development of the Storytelling platform, a tool to support investigation of the research question. Storytelling platforms to create a VM museum on-site installation was used with various interaction systems at G-SCOP laboratory. We used the 2D Powerwall, the 3D stereoscopic display and the MIHRIAD miniCAVE to test the differences of interaction between each system. In Section 6.3, we summarize the results of the experience by applying the VM application to all three systems and we conclude about our results. Section 6.4 discusses faced problems, including the limitations of this research. At last but not least, Section 6.5 discusses our perspectives and future works.

6.1 Proposed research question

Nowadays, VR technologies are developed and updated rapidly, bringing in high-quality, low-cost devices and powerful interactions that are more responsive for better user experience. These devices are applied into many VR applications. Virtual Museum is one of the research field using VR technologies to enhance learning experience of users. Hence, various devices are available and we want to know how to choose the right devices to create an experience for VM. We focus on the development of on-site installation VMs where user interaction is needed and requires devices to encourage user engagement. Here, the main research question is "How different interaction systems affect user engagement in the application of the on-site installation VM and what are the key factors?"

We need a tool that allows us to develop VM applications on different devices and to use different interaction techniques. This led us to the development of a Storytelling platform that creates a VM application and run it on a variety of devices without having to modify any part of the applications. It enables the usage comparison of the application that interacts with users on different devices.

6.2 Storytelling platform and implementation

A Storytelling platform was proposed to support a conceptual framework for developing an interactive on-site installation of VMs with the following objectives:

- To develop a complete platform for designing interactive VEs.
- To develop an interaction system for managing device connection.

In Chapter 3, Storytelling platform is designed and developed. It facilitates the creation of interactive content which is connected to various devices. Storytelling model and interaction model are applied to be a high- level abstraction for story description.

- The storytelling model is driven by the storyboard with three components: entity, event and action to create situation into a story. This structure allows many choices of actions to describe an event on an entity. Then different styles of stories are prepared to study user engagement on each style and also to investigate needs of interaction from the content we made.

- Interaction model defines a high-level description for connection of different devices. The same interaction tasks will be applied with the same result even using different devices. Model transformation allows device and interaction techniques change without modification of story structure or any programming effort.

In Chapter 4, we described the creation of an interactive VM on the Storytelling platform and implemented two case studies in the Chapter 5: the virtual tour and the semi-guided story experiences. Interaction systems and interaction techniques are designed based on the interaction model applied with different systems: the 2D Powerwall, the 3D stereoscopic display and the CAVE. Each system has different input device. Mouse and keyboard is the input device of the 2D Powerwall, a joystick is the input device of 3D stereoscopic display and a wand is the input device of the CAVE.

6.3 Results and contributions to knowledge

The research question is investigated by assessment of device performance, usability of interaction, user satisfaction and immersion. The result of this study will help us to design the content for interactive on-site installation VM application. This research achieved a number of contributions to understand the development of application to better immerse museum visitors into VM through the implementation and outcomes of the research. We highlight topics in following subsections.

6.3.1 A new platform for interactive content development

A new Storytelling platform to design interactive content with various devices is proposed. This platform allows us to develop applications for the on-site installation VM. It also enables us to develop applications with interactive content that runs on a variety of devices. This feature allows researchers to study user behavior with applications developed through devices using different interaction techniques. This is very useful for researches about VM development as the development of interactive media. In our research, we have set up devices for our experience with only three systems: the 2D system, the 3D system and the CAVE. There may be new device coming in the future, with the addition of the device to the platform, it will be possible to study user behavior of new device and to compare them with older device or system.

6.3.2 Design of interaction techniques based on user's behavior

We added functionality for reporting user interaction to the platform, allowing us to capture user data during interaction with the system. When users finished, the platform will report elapsed times of each user the navigation time, the manipulation time and the selection time. It supports developers to investigate mistakes in the actual usage to have better interaction techniques in a new development.

6.3.3 Design of interactive content based on user's behavior

Designing of VE on user demand is an important issue that makes VM development more effective. This platform is designed to monitor not only the time of interaction, but also the amount of interactions on each object in the scene. The report shows how many times users interact with each object in the scene, which allows developers know user behavior on their design. This information can be used to improve the interactive content of VM development to meet the needs of users. We reach a kind of user centered VM design.

6.3.4 Interaction performance, interaction need and learning behaviors

About interaction time (H1), we found that although the time spent on navigation is not different from the actual usage, but the user perspective of navigation is different. About interaction needs (H2), we found that correlation of the ease of use and immersion of using different interaction system contributed to have more number of interactions. However, on user perspective, they felt that the immersion of each interaction system is not different. That means users will have more interaction when they immerse to VE whatever device is used. At last about content learning (H3), we found that the ease of use, comfort and immersion do not affect user learning.

6.4 Limitations

There are some limitations on our research work that we wish summarize as follows:

- The Storytelling platform has an interactive model that can change the use of devices. However, we need to define interaction techniques for each device first. The interaction techniques should be consistent with other devices to allow the application to run without malfunction whatever system is enabled. Some functions are not so easy to reproduce on every device.
- In the CAVE, we need to use the wand as an interactor for selection and manipulation objects within the scene. Anyway, navigation requires a joystick because the wand does not have much control buttons and overlap to other functions on selection and manipulation. So, there are two interactors on the CAVE system that make it difficult for users to switch between them. New solutions could improve the CAVE perception.
- The whole stories comprise many scenes with a lot of models to create animation and interaction with the user. It affected the system performance which becomes very slow. Especially on the CAVE system, we have to separate the scenes into sub-files to prevent animation delay or interaction not responding. We cut the details of the model and cut some of the textures, so the realism of the story was reduced. The perception of the system is obviously impacted by some misbehavior of the overall rendering capacity.
- Initial experiences were very long and not compatible. We tried to reduce the experience time. So we cut off some parts of the experience, leaving only a main part, which led to some details missing in the experience.
- Based on the design of the experience, first, we designed four interaction systems: 2D system, 3D system, CAVE system and HMD system. Because of the time-consuming trial and testing of the HMD system was not complete, we removed the HMD system, so unfortunately we cannot study the user's interaction with this system.

Most of these limitations are due to some technological issues with a proof of concept system which is not perfect. Any how it did not avoid to validate the overall process we were

following.

6.5 Perspectives

This research focuses on presenting a methodology for studying factors affecting user interaction. There are also topics and details in this research that are interesting to complete with additional context, which are divided into the following issues:

6.5.1 Interactor design

In this case, the interactor is an input device that is used to interact with the system. The more complex the system is, the more capable the interactor is needed. A good interactor design will allow users to interact with the system more easily. For instance in the CAVE system, the wand is used as the main interactor for selection and manipulation, but when navigation is added, it is not enough to control all interaction. Thus, we have to design a new interactor or add another input device such as a joystick integrated with a wand. Users may have different interactions when using different interactor. It is interesting to study the appropriate interactor design for the application that will be used in the VM.

6.5.2 Study of interaction techniques

One interaction system can have many interaction techniques. On the 2D system, we use a mouse and keyboard as an input device and manipulate it by clicking and dragging on an object. We should change interaction technique to use buttons on keyboard instead, and study which interaction techniques are better for different contents. We can use the Storytelling platform to study the interaction techniques of each interaction system. With the features of the Storytelling platform, applications can use different interaction techniques on the same interaction system and content.

6.5.3 Full study of interaction systems

From the VE design (including the design of interactor and interaction techniques) we can propose a new conceptual framework to develop application for on-site installation VM. First, design the content and story to be presented in the application using various interactions. Next, install the interaction system and test the design of interactor and interaction techniques. This step will find the best solution of interactor and interaction technique of each system before it is compared. Then compare each interaction system, monitor and capture user interactions. Finally, collected data is analyzed to find the optimal interaction system and use interaction behavior of the users to improve the content designed in the first step. This approach will enable the creation of a VM application that responds to user needs with better learning. The prospective here is to offer a full suite of tools to follow this process rather than proving the concept platform.

6.5.4 Future works and VM in Thailand

For the development of the Storytelling platform, we create a wide range of application interfaces that enable us to choose the right device for our content. In the development of the Storytelling platform, we highlighted that when creating applications using different devices it has different performance. The user interacts with each device differently, but content learning of each device is not so different because the user learning mainly depends on the content. We can enhance the interesting of the content through the selection of appropriate devices and interaction techniques.

However, most VM applications have features adapted to multi-user applications. In the future, Storytelling platform should be developed to support multi-user and inter-user interaction. This will allow us to study the behavior between users to enhance the learning experience of the VM in the future.

What we learned from this research can be used to develop VM applications in Thailand. It still requires a design which is consistent with the immersive VR technologies that are increasingly involved enhancing the learning experience of the user. The Storytelling platform will support us to choose the appropriate devices and interactions for the VM's content about historical sites and culture in Thailand. This will enable interaction between users and the interaction system to enhance learning abilities through VM applications.

This research concept was communicated within two conferences:

- Khundam, Ch., and Noël, F. (2017). Digital Interactive Contents with Adaptive Interaction System for Developing Virtual Museum Applications. In Colloque AIP-PRIMECA 2017.
- Khundam, Ch., and Noël, F. (2017, July). Storytelling Platform for Virtual Museum Development: Lifecycle Management of an Exhibition. In IFIP International Conference on Product Lifecycle Management (pp. 416-426). Springer, Cham.

But we expect to publish in review papers almost about the VM platform on one side, and on a second side about the experiences and thus on the protocol to select the good system for a given VM.

Bibliography

Aliaga, D. G., Bertino, E., & Valtolina, S. (2011). Decho—a framework for the digital exploration of cultural heritage objects. *Journal on Computing and Cultural Heritage (JOCCH)*, *3*(3), 12.

Allard, J., Gouranton, V., Lecointre, L., Melin, E., & Raffin, B. (2002). Net juggler: Running vr juggler with multiple displays on a commodity component cluster. In *Virtual reality, 2002. proceedings. ieee* (pp. 273–274). IEEE.

Anfruns, J. (2014). International council of museums (icom). *Encyclopedia of Global Archaeology*, 3964–3965.

Anthes, C., & Volkert, J. (2006). Invrs–a framework for building interactive networked virtual reality systems. In *International conference on high performance computing and communications* (pp. 894–904). Springer.

Argelaguet, F., & Andujar, C. (2013). A survey of 3d object selection techniques for virtual environments. *Computers & Graphics*, *37*(3), 121–136.

Ayub, M. N., Venugopal, S. T., & Nor, N. F. M. (2005). Development of multimedia authoring tool for educational material disseminations. *Informatics in education*, *4*(1), 5.

Bacim, F., Bowman, D., & Pinho, M. (2009). Wayfinding techniques for multiscale virtual environments. In *3d user interfaces, 2009. 3dui 2009. ieee symposium on* (pp. 67–74). IEEE.

Basset, J., & Noël, F. (2018). Added value of a 3d cave within design activities. In *International conference on virtual reality and augmented reality* (pp. 230–239). Springer.

Bevan, N., Carter, J., & Harker, S. (2015). Iso 9241-11 revised: What have we learnt about usability since 1998? In *International conference on human-computer interaction* (pp. 143–151). Springer.

Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A., & Cruz-Neira, C. (2001). Vr juggler: A virtual platform for virtual reality application development. In *Virtual reality, 2001. proceedings. ieee* (pp. 89–96). IEEE.

Billen, R., Carré, C., Delfosse, V., Hervy, B., Laroche, F., Lefevre, D., ... Van Ruymbeke, M. (2012). 3d historical models: The case studies of liege and nantes. *3D issues in urban and environmental systems*.

Billinghurst, M., Kato, H., & Poupyrev, I. (2001). The magicbook: A transitional ar interface. *Computers & Graphics*, *25*(5), 745–753.

Boisvert, D. L., & Slez, B. J. (1995). The relationship between exhibit characteristics and learning-associated behaviors in a science museum discovery space. *Science education*, *79*(5), 503–518.

Bouhlel, Y. (2010). A nostalgic rummage through the history of flash. *Active Tuts Plus. December*, *31*.

Bowman, D. A., Kruijff, E., LaViola Jr, J. J., & Poupyrev, I. (2001a). An introduction to 3-d user interface design. *Presence: Teleoperators & Virtual Environments*, *10*(1), 96–108.

Bowman, D., Davis, E., Hodges, L., & Badre, A. (1999). Maintaining spatial orientation during travel in an immersive virtual environment. *Presence*, *8*(6), 618–631.

Bowman, D., Johnson, D., & Hodges, L. (2001b). Testbed evaluation of virtual environment interaction techniques. *Presence: Teleoperators & Virtual Environments*, *10*(1), 75–95.

Bowman, D., Kruijff, E., LaViola Jr, J. J., & Poupyrev, I. P. (2004). *3d user interfaces: Theory and practice, coursesmart etextbook*. Addison-Wesley.

Bowman, D. A. (1999). Interaction techniques for common tasks in immersive virtual environments. *Georgia Institute of Technology*.

Bowman, D., & Hodges, L. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on interactive 3d graphics* (35–ff). ACM.

Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., & Vanderdonckt, J. (2003). A unifying reference framework for multi-target user interfaces. *Interacting with computers*, *15*(3), 289–308.

Carlsson, C., & Hagsand, O. (1993). Dive a multi-user virtual reality system. In *Virtual reality annual international symposium, 1993., 1993 ieee* (pp. 394–400). IEEE.

Carrozzino, M., & Bergamasco, M. (2010). Beyond virtual museums: Experiencing immersive virtual reality in real museums. *Journal of Cultural Heritage*, *11*(4), 452–458.

Chen, J. C. (2016). The crossroads of english language learners, task-based instruction, and 3d multi-user virtual learning in second life. *Computers & Education*, *102*, 152–171.

Chittaro, L., & Ieronutti, L. (2004). A visual tool for tracing users' behavior in virtual environments. In *Proceedings of the working conference on advanced visual interfaces* (pp. 40–47). ACM.

Conway, M., Pausch, R., Gossweiler, R., & Burnette, T. (1994). Alice: A rapid prototyping system for building virtual environments. In *Conference companion on human factors in computing systems* (pp. 295–296). ACM.

Coutaz, J., & Calvary, G. (2012). Hci and software engineering for user interface plasticity. CRC Press.

Csapó, Á., Wersényi, G., Nagy, H., & Stockman, T. (2015). A survey of assistive technologies and applications for blind users on mobile platforms: A review and foundation for research. *Journal on Multimodal User Interfaces*, *9*(4), 275–286.

De Lucia, A., Francese, R., Passero, I., & Tortora, G. (2009). Development and evaluation of a virtual campus on second life: The case of seconddmi. *Computers & Education*, *52*(1), 220–233.

De Melo, G., Suchanek, F., & Pease, A. (2008). Integrating yago into the suggested upper merged ontology.

Derrick, B., & White, P. (2017). Comparing two samples from an individual likert question. *International Journal of Mathematics and Statistics*, *18*(3).

Dix, A., Finlay, J., & Abowd, G. (1998). R. beale: Human-computer interaction. Prentice Hall, Hillsdale.

Doerr, M. (2003). The cidoc conceptual reference module: An ontological approach to semantic interoperability of metadata. *AI magazine*, *24*(3), 75.

Dong, M. (2016). Towards understanding and developing virtual environments to increase accessibilities for people with visual impairments.

Dupont, F., Duval, T., Fleury, C., Forest, J., Gouranton, V., Lando, P., ... Schmutz, A. (2010). Collaborative scientific visualization: The collaviz framework. In *Jvrc 2010 (2010 joint virtual reality conference of eurovr-egve-vec)*.

Falk, J. H., & Dierking, L. D. (2016). *The museum experience revisited*. Routledge.

Forsberg, A., Herndon, K., & Zeleznik, R. (1996). Aperture based selection for immersive virtual environments. In *Proceedings of the 9th annual acm symposium on user interface software and technology* (pp. 95–96). ACM.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, *32*(200), 675–701.

Gaeta, M., Loia, V., Mangione, G. R., Orciuoli, F., Ritrovato, P., & Salerno, S. (2014). A methodology and an authoring tool for creating complex learning objects to support interactive storytelling. *Computers in Human Behavior*, *31*, 620–637.

Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., & Schneider, L. (2002). Sweetening ontologies with dolce. In *International conference on knowledge engineering and knowledge management* (pp. 166–181). Springer.

Gangemi, A., Guarino, N., & Oltramari, A. (2001). Conceptual analysis of lexical taxonomies: The case of wordnet top-level. In *Proceedings of the international conference on formal ontology in information systems-volume 2001* (pp. 285–296). ACM.

Göbel, S., & Mehm, F. (2013). Personalized, adaptive digital educational games using narrative game-based learning objects. In *Serious games and virtual worlds in education, professional development, and healthcare* (pp. 74–84). IGI Global.

Göbel, S., Salvatore, L., & Konrad, R. (2008). Storytec: A digital storytelling platform for the authoring and experiencing of interactive and non-linear stories. In *Automated solutions for cross media content and multi-channel distribution, 2008. axmedis'08. international conference on* (pp. 103–110). IEEE.

Guarino, N., & Welty, C. (2002). Evaluating ontological decisions with ontoclean. *Communications of the ACM*, *45*(2), 61–65.

Hart, S. G., & Staveland, L. E. (1988). Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology* (Vol. 52, pp. 139–183). Elsevier.

Hazan, S., Hermon, S., Turra, R., Pedrazzi, G., Franchi, M., & Wallergard, M. (2015). Deliverable report.

Hebbel-Seeger, A. (2013). Pedagogical and psychological impacts of teaching and learning in virtual realities. *Synthetic Worlds Integrated Series in Information Systems*, 233–249. doi:10.1007/978-1-4614-6286-6_9

Hergenhan, J., Rutschke, J., Uhl, M., Navarro, S. E., Hein, B., & Worn, H. (2015). A haptic display for tactile and kinesthetic feedback in a chai 3d palpation training scenario. In *Robotics and biomimetics (robio), 2015 ieee international conference on* (pp. 291–296). IEEE.

Huhtamo, E. (2010). On the origins of the virtual museum. *Museums in a digital age*, 121–135.

Hyvönen, E., Mäkelä, E., Kauppinen, T., Alm, O., Kurki, J., Ruotsalo, T., ... Kuittinen, H. et al. (2009). Culturesampo–finnish culture on the semantic web 2.0. thematic perspectives for the end-user. In *Proceedings, museums and the web* (pp. 15–18).

Irawati, S., Ahn, S., Kim, J., & Ko, H. (2008). Varu framework: Enabling rapid prototyping of vr, ar and ubiquitous applications. In *Ieee virtual reality 2008* (pp. 201–208). IEEE.

Johnson, L., Becker, S. A., Cummins, M., Estrada, V., Freeman, A., & Hall, C. (2016). *Nmc horizon report: 2016 higher education edition*. The New Media Consortium. Austin, Texas.

Kang, S. B., & Desikan, P. K. (1997). *Virtual navigation of complex scenes using clusters of cylindrical panoramic images*. Digital, Cambridge Research Laboratory.

Kaskalis, T. H., Tzidamis, T. D., & Margaritis, K. (2007). Multimedia authoring tools: The quest for an educational package. *Educational Technology & Society*, *10*(3), 135–162.

Keil, J., Pujol, L., Roussou, M., Engelke, T., Schmitt, M., Bockholt, U., & Eleftheratou, S. (2013). A digital look at physical museum exhibits: Designing personalized stories with handheld augmented reality in museums. In *Digital heritage international congress (digitalheritage), 2013* (Vol. 2, pp. 685–688). IEEE.

Kelleher, C., Pausch, R., Pausch, R., & Kiesler, S. (2007). Storytelling alice motivates middle school girls to learn computer programming. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 1455–1464). ACM.

Khundam, C., & Noël, F. (2017). Storytelling platform for virtual museum development: Lifecycle management of an exhibition. In *Ifip international conference on product lifecycle management* (pp. 416–426). Springer.

Kreylos, O. (2008). Environment-independent vr development. In *International symposium on visual computing* (pp. 901–912). Springer.

Krueger, A. B., Colletti, P., Bogner, H. R., Barg, F., & Stineman, M. (2017). Conducting focus groups in second life on health-related topics. *J Alternative Med Res*, *9*(2), 357–362.

Kuck, R., Wind, J., Riege, K., Bogen, M., & Birlinghoven, S. (2008). Improving the avango vr/ar framework: Lessons learned. In *Workshop virtuelle und erweiterte realität* (pp. 209–220).

Kybartas, B., & Bidarra, R. (2017). A survey on story generation techniques for authoring computational narratives. *IEEE Transactions on Computational Intelligence and AI in Games*, *9*(3), 239–253.

LaViola Jr, J. J., Kruijff, E., McMahan, R. P., Bowman, D., & Poupyrev, I. P. (2017). *3d user interfaces: Theory and practice*. Addison-Wesley Professional.

Liarokapis, F., & White, M. (2005). Augmented reality techniques for museum environments. *Mediterranean Journal of Computers and Networks*, *1*(2), 95–102.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of psychology*.

Lombardo, V., & Pizzo, A. (2013). Ontologies for the metadata annotation of stories. In *Digital heritage (2)* (pp. 153–160).

Luciano, C., Banerjee, P., Florea, L., & Dawe, G. (2005). Design of the immersivetouch: A high-performance haptic augmented virtual reality system. In *11th international conference on human-computer interaction, las vegas, nv*.

Marchiori, E. J., Manjón, B. F., & Moreno-Ger, P. (2010). *Weev: A multidisciplinary approach to educational game development* (Doctoral dissertation, Citeseer).

Marchiori, E. J., Torrente, J., del Blanco, Á., Moreno-Ger, P., Sancho, P., & Fernández-Manjón, B. (2012). A narrative metaphor to facilitate educational game authoring. *Computers & Education*, *58*(1), 590–599.

Martínez, V., Ríos, A. M., & Melero, F. J. (2016). A web 3d-scene editor of virtual reality exhibitions and 360-degree videos. In *Proceedings of the xxvi spanish computer graphics conference* (pp. 9–13). Eurographics Association.

Martz, P. (2007). Openscenegraph quick start guide. *PMARTZ Computer Graphics Systems*.

Masolo, C., Borgo, S., Gangemi, A., Guarino, N., & Oltramari, A. (2002). Wonderweb deliverable d17.

McCracken, D. D., & Wolfe, R. J. (2004). *User-centered website development: A human-computer interaction approach*. Prentice Hall Upper Saddle River.

McMullen, M. (2014). Direct3d 12 api preview. Retrieved from https://channel9.msdn.com/Events/Build/2014/3-564

Meditskos, G., Dasiopoulou, S., Efstathiou, V., & Kompatsiaris, I. (2013). Ontology patterns for complex activity modelling. In *International workshop on rules and rule markup languages for the semantic web* (pp. 144–157). Springer.

Merchant, Z., Goetz, E. T., Keeney-Kennicutt, W., Cifuentes, L., Kwok, O.-m., & Davis, T. J. (2013). Exploring 3-d virtual reality technology for spatial ability and chemistry achievement. *Journal of Computer Assisted Learning*, *29*(6), 579–590.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Mine, M. R., Brooks Jr, F. P., & Sequin, C. H. (1997). Moving objects in space: Exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th annual conference on computer graphics and interactive techniques* (pp. 19–26). ACM Press/Addison-Wesley Publishing Co.

Moreira, R. M. F. d. C. et al. (2011). *Integrating a 3d application server with a cave* (Doctoral dissertation).

Mouton, C., Sons, K., & Grimstead, I. (2011). Collaborative visualization: Current systems and future trends. In *Proceedings of the 16th international conference on 3d web technology* (pp. 101–110). ACM.

Neelakantam, S., & Pant, T. (2017). *Learning web-based virtual reality: Build and deploy web-based virtual reality technology*. Apress.

NewYorkFilmAcademy. (2017). Best free game engines and development software. Retrieved from https://www.nyfa.edu/student-resources/best-free-game-engines-and-development-software/

Nielsen, J. (2003). Usability 101: Introduction to usability.

Padilla-Meléndez, A., & del Águila-Obra, A. R. (2013). Web and social media usage by museums: Online value creation. *International Journal of Information Management*, *33*(5), 892–898.

Parisi, T. (2012). *Webgl: Up and running.* " O'Reilly Media, Inc.".

Pease, A., Niles, I., & Li, J. (2002). The suggested upper merged ontology: A large ontology for the semantic web and its applications. In *Working notes of the aaai-2002 workshop on ontologies and the semantic web* (Vol. 28, pp. 7–10).

Pierce, J. S., Forsberg, A. S., Conway, M. J., Hong, S., Zeleznik, R. C., & Mine, M. R. (1997). Image plane interaction techniques in 3d immersive environments. In *Proceedings of the 1997 symposium on interactive 3d graphics* (39–ff). ACM.

Pinto, J., Dias, P., Eliseu, S., & Sousa Santos, B. (2015). Interactive configurable virtual environment with kinect navigation and interaction. In *Proceedings of the portuguese meeting of computer graphics and interaction-scitecin/epcgi*.

Polycarpou, C. (2018). The vimm definition of a virtual museum. Retrieved from https://www.vi-mm.eu/2018/01/10/the-vimm-definition-of-a-virtual-museum/

Poupyrev, I., Billinghurst, M., Weghorst, S., & Ichikawa, T. (1996). The go-go interaction technique: Non-linear mapping for direct manipulation in vr. In *Proceedings of the 9th annual acm symposium on user interface software and technology* (pp. 79–80). ACM.

QuickTime and the Rise of Multimedia. (n.d.). Retrieved from http://www.computerhistory.org/atchm/quicktime-and-the-rise-of-multimedia

Rantzau, D., Lang, U., Lang, R., Nebel, H., Wierse, A., & Ruehle, R. (1996). Collaborative and interactive visualization in a distributed high performance software environment. In *High performance computing for computer graphics and visualisation* (pp. 207–216). Springer.

Robertson, J., & Nicholson, K. (2007). Adventure author: A learning environment to support creative design. In *Proceedings of the 6th international conference on interaction design and children* (pp. 37–44). ACM.

Rohlf, J., & Helman, J. (1994). Iris performer: A high performance multiprocessing toolkit for real-time 3d graphics. In *Proceedings of the 21st annual conference on computer graphics and interactive techniques* (pp. 381–394). ACM.

Roussou, M. (2001). Immersive interactive virtual reality in the museum. *Proc. of TiLE (Trends in Leisure Entertainment)*.

Scherp, A., Franz, T., Saathoff, C., & Staab, S. (2009). F–a model of events based on the foundational ontology dolce+ dns ultralight. In *Proceedings of the fifth international conference on knowledge capture* (pp. 137–144). ACM.

Schultz, R. (2018). Second life infographic: Some statistics from 15 years of sl. Retrieved from https://ryanschultz.com/2018/04/23/second-life-infographic-some-statistics-from-15-years/

Schulze, J. P., Prudhomme, A., Weber, P., & DeFanti, T. A. (2013). Calvr: An advanced open source virtual reality software framework. In *The engineering reality of virtual reality 2013* (Vol. 8649, p. 864902). International Society for Optics and Photonics.

Seco, F., & Jiménez, A. R. (2018). Smartphone-based cooperative indoor localization with rfid technology. *Sensors*, *18*(1), 266.

Shapiro, S. S., & Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, *52*(3/4), 591–611.

Sharma, N., Uppal, S., Gupta, S., Patiala, V. B. P. R. D., & Panipat, S. D. (2011). Technology based on touch: Haptics technology. *IJCEM International Journal of Computational Engineering & Management*, *12*.

Sherman, W. R., Coming, D., & Su, S. (2013). Freevr: Honoring the past, looking to the future. In *The engineering reality of virtual reality 2013* (Vol. 8649, p. 864906). International Society for Optics and Photonics.

Siltanen, S. (2012). *Theory and applications of marker-based augmented reality*. VTT.

Stoakley, R., Conway, M. J., & Pausch, R. (1995). Virtual reality on a wim: Interactive worlds in miniature. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 265–272). ACM Press/Addison-Wesley Publishing Co.

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). Yago: A core of semantic knowledge. In *Proceedings of the 16th international conference on world wide web* (pp. 697–706). ACM.

Takala, T. M. (2014). Ruis: A toolkit for developing virtual reality applications with spatial interaction. In *Proceedings of the 2nd acm symposium on spatial user interaction* (pp. 94–103). ACM.

Taylor, R. M., Hudson, T. C., Seeger, A., Weber, H., Juliano, J., & Helser, A. T. (2001). Vrpn: A device-independent, network-transparent vr peripheral system. In *Proceedings of the acm symposium on virtual reality software and technology* (pp. 55–61). ACM.

Teyseyre, A. R., & Campo, M. R. (2009). An overview of 3d software visualization. *IEEE Trans. Vis. Comput. Graph. 15*(1), 87–105.

The Programming Languages Beacon. (2012). Retrieved from https://archive.is/20120530050106/ http://www.lextrait.com/Vincent/implementations.html

Torrente, J., Del Blanco, Á., Marchiori, E. J., Moreno-Ger, P., & Fernández-Manjón, B. (2010). ¡ e-adventure¿: Introducing educational games in the learning process. In *Education engineering (educon), 2010 ieee* (pp. 1121–1126). IEEE.

Tukey, J. W. (1949). Comparing individual means in the analysis of variance. *Biometrics*, 99–114.

Uijlings, J. (2006). Designing a virtual environment for story generation. *Unpublished master's thesis, University of Amsterdam.(to appear)*.

Urban, R. J. (2007). A second life for your museum: 3d multi-user virtual environments and museums.

Von Kapri, A., Rick, T., & Feiner, S. (2011). Comparing steering-based travel techniques for search tasks in a cave. In *Virtual reality conference (vr), 2011 ieee* (pp. 91–94). IEEE.

Walczak, K., Cellary, W., & White, M. (2006). Virtual museum exbibitions. *Computer*, *39*(3), 93–95.

Walker, G. (2012). A review of technologies for sensing contact location on the surface of a display. *Journal of the Society for Information Display*, *20*(8), 413–440.

Wang, R., & Qian, X. (2010). *Openscenegraph 3.0: Beginner's guide*. Packt Publishing Ltd.

Wernecke, J. (1994). *The inventor mentor: Programming object-oriented 3d graphics with open inventor, release 2*. Citeseer.

White, M., Petridis, P., Liarokapis, F., & Plecinckx, D. (2007). Multimodal mixed reality interfaces for visualizing digital heritage. *International Journal of Architectural Computing*, *5*(2), 321–337.

Wischgoll, T., Glines, M., Whitlock, T., Guthrie, B. R., Mowrey, C. M., Parikh, P. J., & Flach, J. (2018). Display infrastructure for virtual environments. *Electronic Imaging*, *2018*(1), 1–11.

Wits, W. W., Noël, F., & Masclet, C. (2011). Exploring the potential of 3d visualization techniques for usage in collaborative design. In *1st cirp design conference*.

Wright, T., & Madey, G. (2008). A survey of collaborative virtual environment technologies. *University of Notre Dame-USA, Tech. Rep*, 1–16.

Yahya, I. (1997). *Museum learning, the museum visitor, the museum visit* (Doctoral dissertation, Museum Studies).

Yáñez, C., Okada, A., & Palau, R. (2015). New learning scenarios for the 21 st century related to education, culture and technology. *International Journal of Educational Technology in Higher Education*, *12*(2), 87–102.

Zendjebil, I., Ababsa, F., Didier, J.-Y., & Mallem, M. (2008). Hybrid localization system for mobile outdoor augmented reality applications. In *Image processing theory, tools and applications, 2008. ipta 2008. first workshops on* (pp. 1–6). IEEE.

Zhai, S., & Milgram, P. (1993). Human performance evaluation of manipulation schemes in virtual environments.

Zhou, F., Duh, H. B.-L., & Billinghurst, M. (2008). Trends in augmented reality tracking, interaction and display: A review of ten years of ismar. In *Proceedings of the 7th ieee/acm international symposium on mixed and augmented reality* (pp. 193–202). IEEE Computer Society.

Zotkin, D. N., Duraiswami, R., & Davis, L. S. (2004). Rendering localized spatial audio in a virtual auditory space. *IEEE Transactions on multimedia*, *6*(4), 553–564.

Appendix A

Chapter 4

A.1 Historical model

Scene1: introduction
Wat Phrathat has been the center of the locals with a history of more than 1800 years and significant art and cultural aspects. Wat Phrathat is also considered the origin of people's faith towards Buddhism in the south of Thailand. It has influenced the form of architecture and communication and existence of Theravade doctrine of Lanka.
Historical facts: communication and existence of Theravade
Components: Wat Phrathat, landscape

Scene2: abandon
There have been various legends of Phrathat. One legend mentioned that the Buddha relics (the teeth of the Lord Buddha) have been kept inside. The relics were moved from Tontha Buri by Prince Thanakuman and Princess Hemchala. It has been assumed that the original form of Phrathat was Mondop, a structure with four arches and a pyramidal roof topped with five tiers as Srivijaya style.
Historical facts: Buddha relics, Srivijaya style
Components: Prince Thanakuman and Princess Hemchala and their vehicle, Mondop

Scene3: adjacent spreading
Then, in 1700BE, Theravade doctrine of Lanka has been prosperous and spread to Nakhon Si Thammarat, as a result Chedi of Theravade was constructed to cover the original, but existing one. This becomes a typical model of Theravade in the south of Thailand and influentially spreads to adjacent areas.
Historical facts: Theravade doctrine of Lanka
Components: Chedi Phrathat, pagoda in adjacent areas

Scene4: urban design
The value of Wat Phra Mahathat Woramahawihan also portrays creativity of human race in architecture via artistic Buddhism which, moreover, identified and clearly explained through the location of Wat Phrathat. Wat Phrathat is situated at the ideal position which reflects urban design, with mountain ranges and streams at both sides. In addition, Wat Phrathat is situated on the sand hill, hence this was recognized as the landmark by sailors.
Historical facts: urban design and landmark
Components: Wat Phrathat, urban components, mountain, river, sailors

Scene5: various viharas
The architecture, furthermore, reflects the integration between Buddhist philosophy and creativity of artisans who created components of the temples, as various viharas with their identity.
Historical facts: Buddhist philosophy in architecture
Components: various viharas

Scene6: golden top
Faith towards Buddhism, Buddhism has been playing roles which influence and reflect the locals' faith since the old days. In the past, the locals collected their valuable belongings and kept them at the top part of the Chedi. Hence the Chedi was called Phrathat Yod Thong, which means the golden top.
Historical facts: peoples' faith
Components: valuable belongings, Chedi Phrathat

Scene7: festival
Important activities reflecting faith and ways of life of the people are held twice a year, i.e. on Make Buja's Day and Visaka Buja's Day. The locals will buy white, red or yellow fabrics and sew them together as a long cloth to symbolize Phra Bot, the sacred cloth representing the Lord Buddha. There will be the procession, an activity that the locals come together to hold the cloth and put it around the Chedi. Wat Phra Mahathat Woramahawihan directly associates with the locals' ways of life. In other words, a religious center, along with cultural and architectural heritage of Buddhism, whose types of value has been retained until the present days.
Historical facts: Buddhist activities
Components: Chedi Phrathat, Phra Bot, fabrics, people

A.2 Storytelling model and interaction model

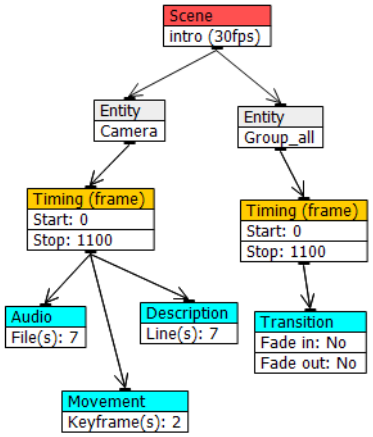

Figure A.1: Fully-guided story of introduction scene in the event editor

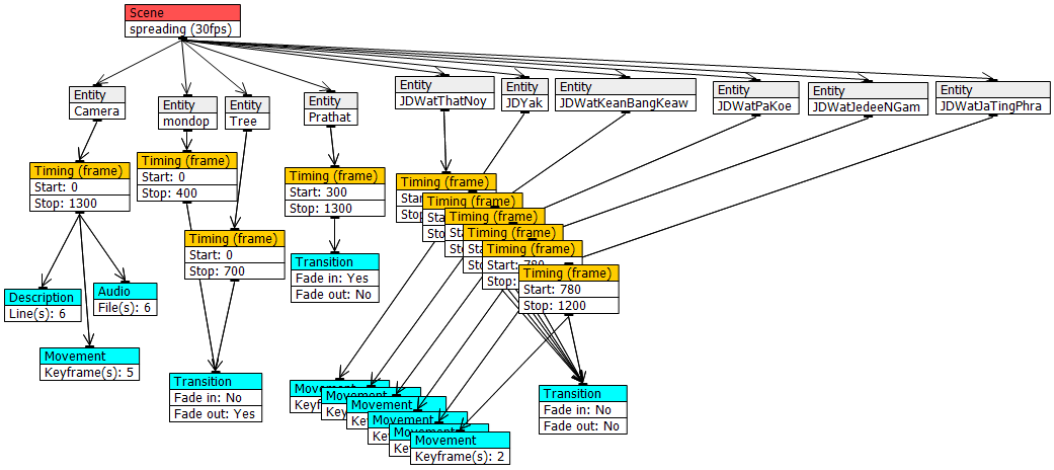

Figure A.2: Fully-guided story of abandon scene in the event editor

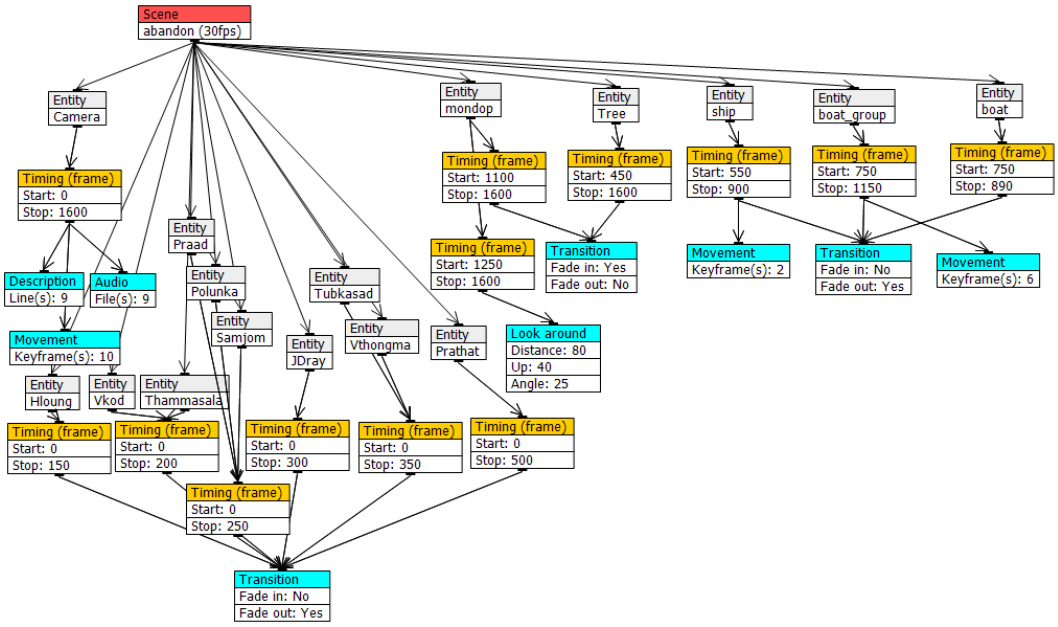

Figure A.3: Fully-guided story of spreading scene in the event editor

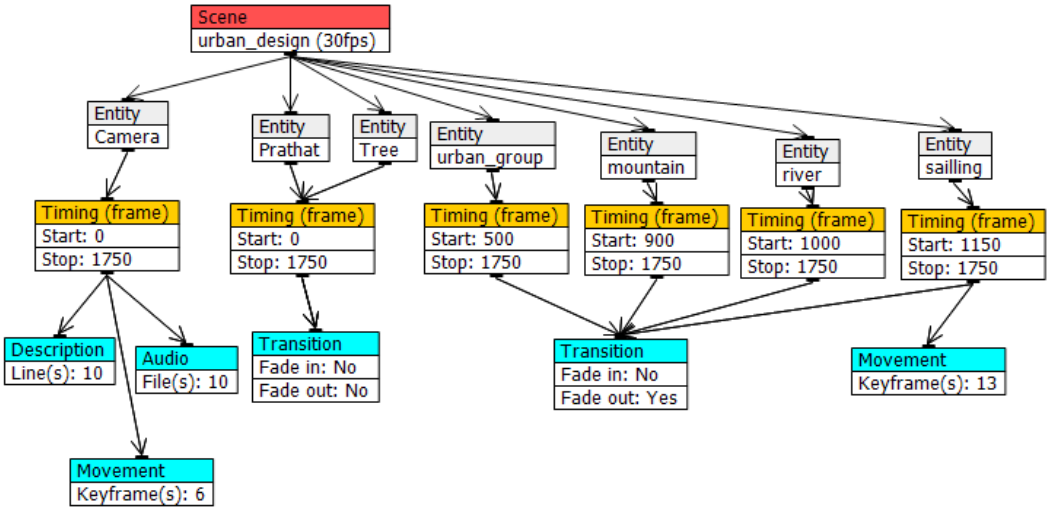

Figure A.4: Fully-guided story of urban scene in the event editor

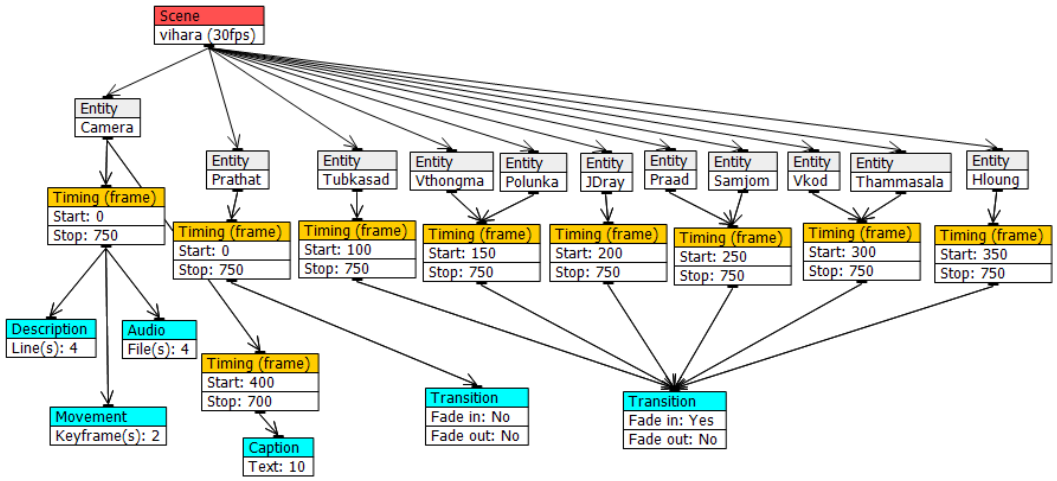

Figure A.5: Fully-guided story of vihara scene in the event editor

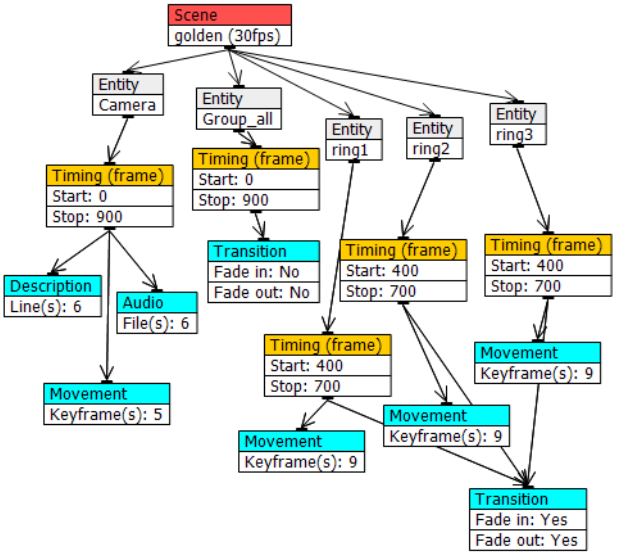

Figure A.6: Fully-guided story of golden scene in the event editor

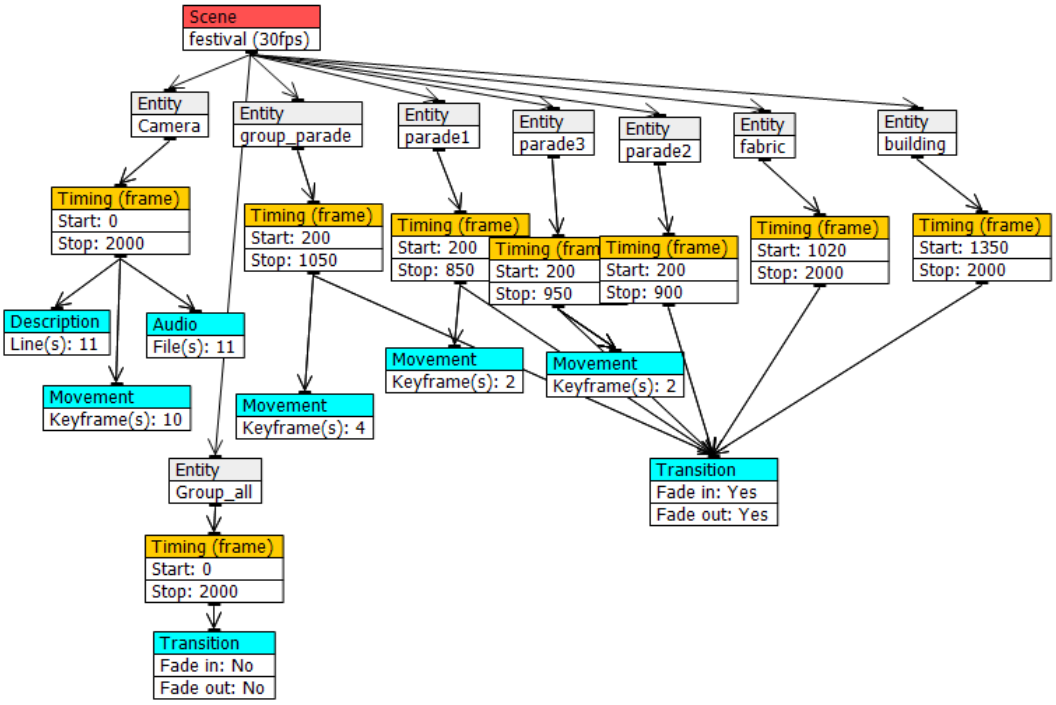

Figure A.7: Fully-guided story of festival scene in the event editor

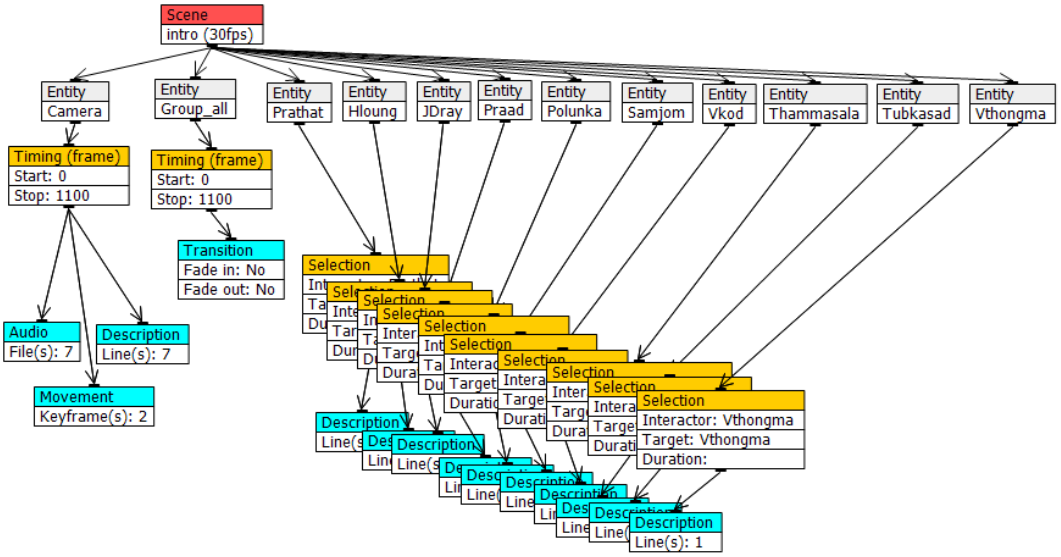

Figure A.8: Semi-guided story of introduction scene in the event editor

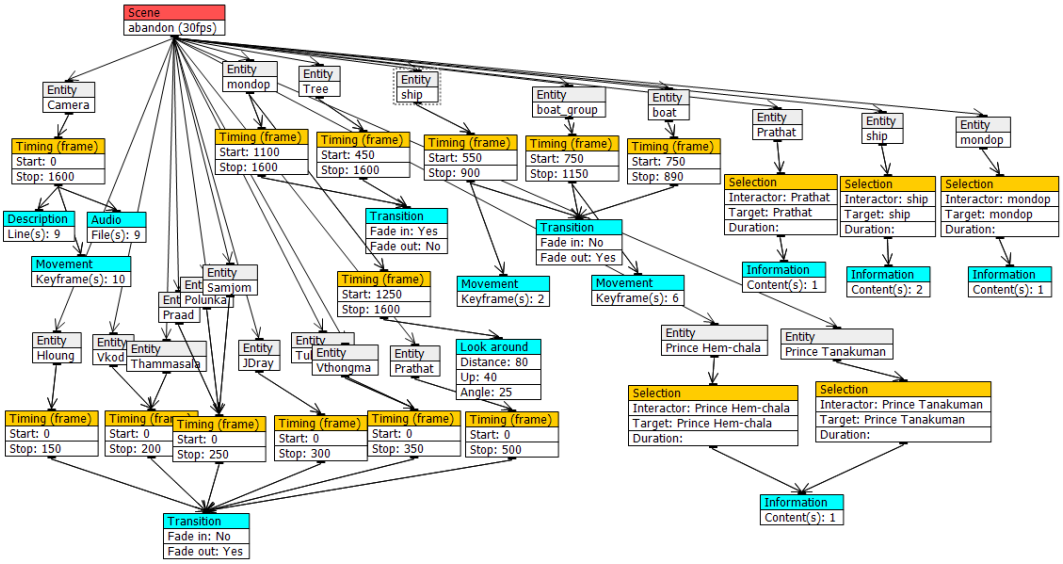

Figure A.9: Semi-guided story of abandon scene in the event editor

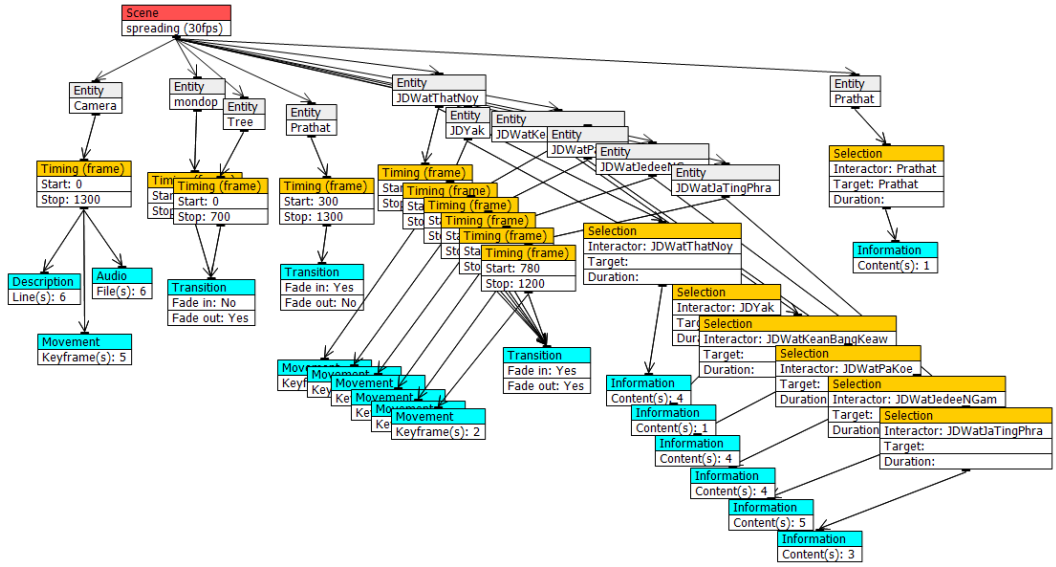

Figure A.10: Semi-guided story of spreading scene in the event editor

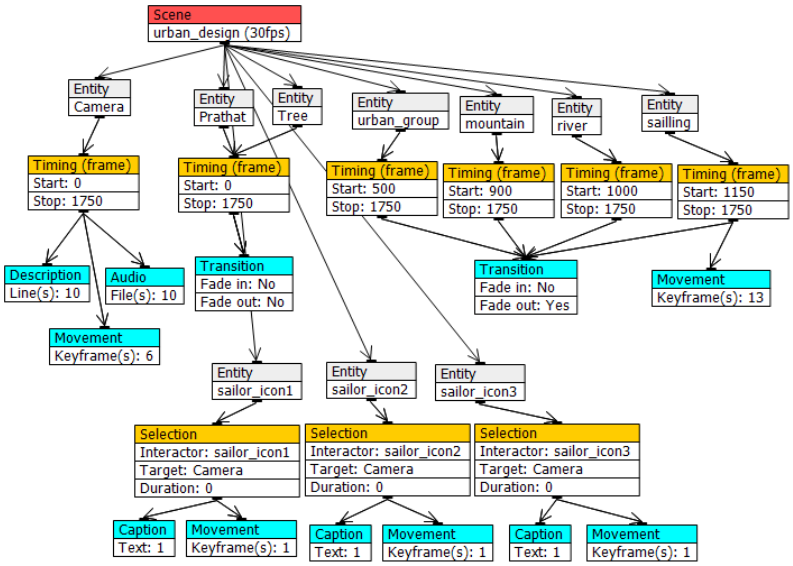

Figure A.11: Semi-guided story of urban scene in the event editor

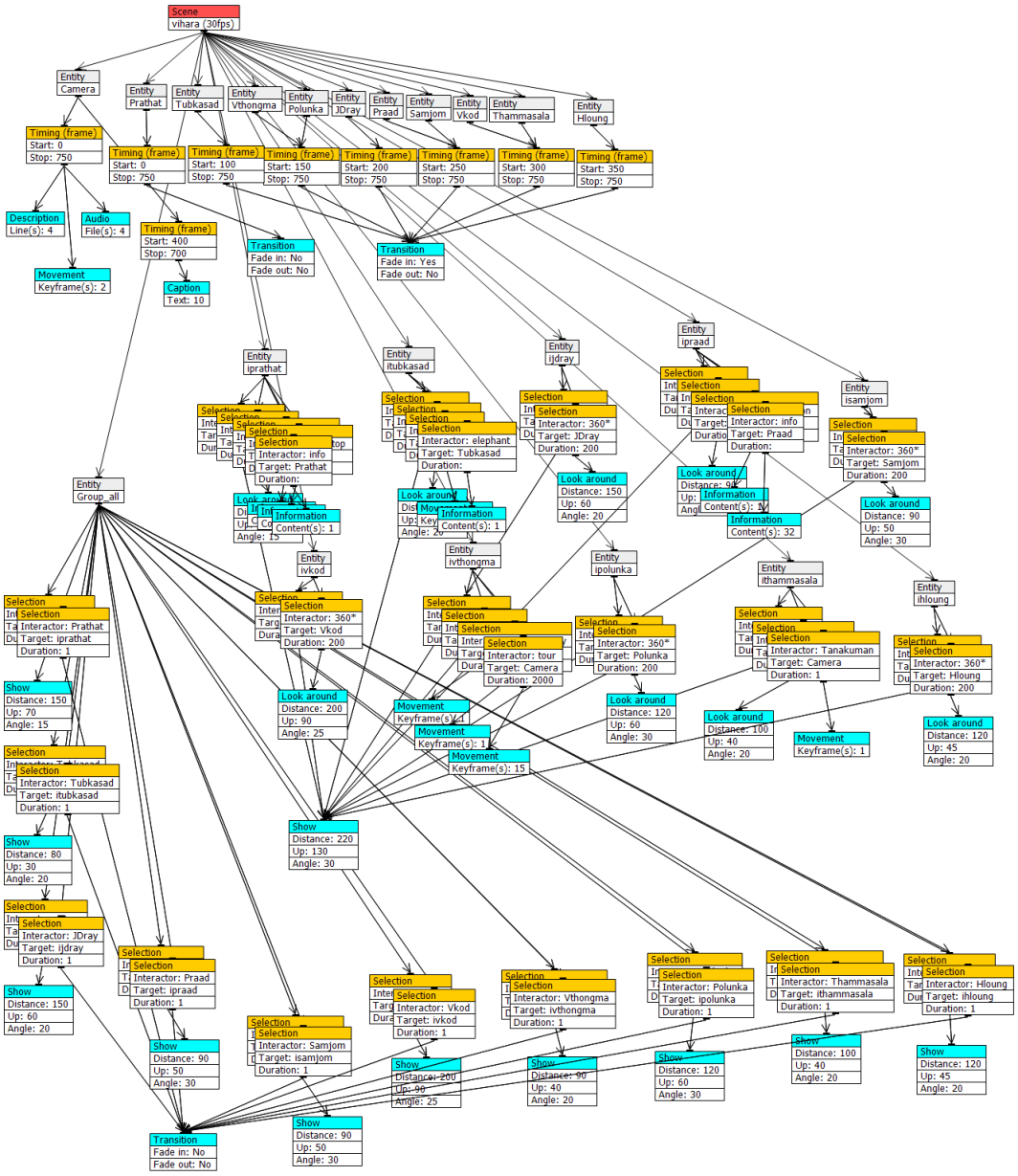

Figure A.12: Semi-guided story of vihara scene in the event editor

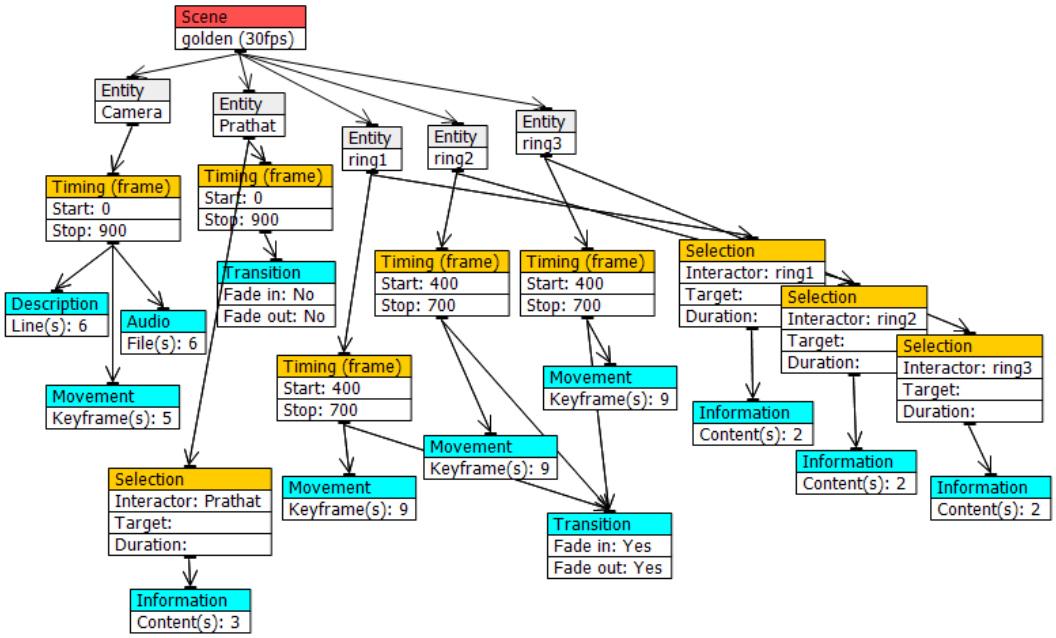

Figure A.13: Semi-guided story of golden scene in the event editor

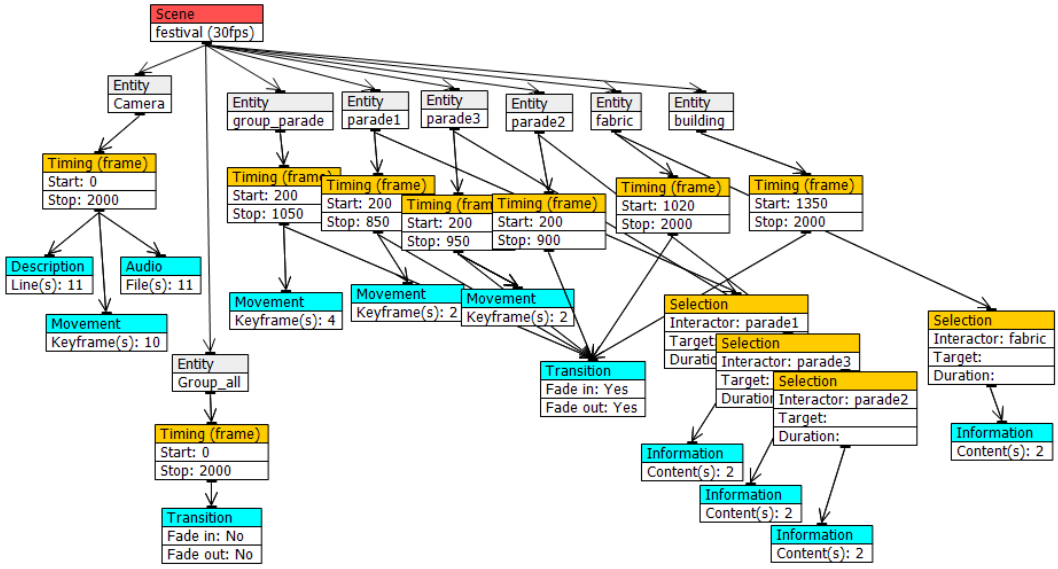

Figure A.14: Semi-guided story of festival scene in the event editor

Appendix B

Chapter 5

B.1 Tukey's test

Table B.1: Tukey's post-hoc analysis on manipulation time of the virtual tour experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D - 3D	1.120666667	15	15	9.851979013	0.113750411	3.44	reject
2D - CAVE	77.182	15	15	9.851979013	7.83416204	3.44	accept
3D - CAVE	78.30266667	15	15	9.851979013	7.947912451	3.44	accept

Table B.2: Tukey's post-hoc analysis on selection time of the virtual tour experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D - 3D	47.98533332	15	15	5.634546647	8.516272263	3.44	accept
2D - CAVE	2.570666667	15	15	5.634546647	0.456233097	3.44	reject
3D - CAVE	45.41466666	15	15	5.634546647	8.060039166	3.44	accept

Table B.3: Tukey's post-hoc analysis on amount of interactions of the semi-guided story experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D - 3D	11.2	15	15	1.944793619	5.75896583	3.44	accept
2D - CAVE	7.333333333	15	15	1.944793619	3.770751436	3.44	accept
3D - CAVE	3.866666667	15	15	1.944793619	1.988214394	3.44	reject

Table B.4: Tukey's post-hoc analysis on total navigation of the non-guided story experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D single - 2D full	36.27733333	15	15	27.86520847	1.301886306	3.44	reject
2D single - CAVE	182.7606667	15	15	27.86520847	6.558740332	3.44	accept
2D full - CAVE	146.4833333	15	15	27.86520847	5.256854026	3.44	accept

Table B.5: Tukey's post-hoc analysis on total manipulation of the non-guided story experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D single - 2D full	7.192	15	15	10.79317273	0.66634716	3.44	reject
2D single - CAVE	38.16666667	15	15	10.79317273	3.536186034	3.44	accept
2D full - CAVE	30.97466667	15	15	10.79317273	2.869838873	3.44	reject

Table B.6: Tukey's post-hoc analysis on navigation time of the semi-guided story experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D single - 2D full	7.192	15	15	10.79317273	0.66634716	3.44	reject
2D single - CAVE	38.16666667	15	15	10.79317273	3.536186034	3.44	accept
2D full - CAVE	30.97466667	15	15	10.79317273	2.869838873	3.44	reject

Table B.7: Tukey's post-hoc analysis on manipulation time of the non-guided story experience

Pair	Diff	n1	n2	SE	q	Cri.Value	Result
2D single - 2D full	13.93733333	15	15	6.666831469	2.090548321	3.44	reject
2D single - CAVE	49.762	15	15	6.666831469	7.464115484	3.44	accept
2D full - CAVE	35.82466667	15	15	6.666831469	5.373567163	3.44	accept

B.2 Wilcoxon's test

Table B.8: Wilcoxon's post-hoc analysis on navigation easiness of the virtual tour experience

Pair	Statistic	p-value	Result
2D - 3D	63.0	0.006773856	accept
2D - CAVE	63.5	0.006214871	accept
3D - CAVE	69.0	0.599891723	reject

Table B.9: Wilcoxon's post-hoc analysis on manipulation easiness of the virtual tour experience

Pair	Statistic	p-value	Result
2D - 3D	33.0	0.5705228180	reject
2D - CAVE	105.0	0.0008962423	accept
3D - CAVE	94.5	0.0076531065	accept

Table B.10: Wilcoxon's post-hoc analysis on selection easiness of the virtual tour experience

Pair	Statistic	p-value	Result
2D - 3D	60.0	0.01491205	accept
2D - CAVE	60.5	0.01376368	accept
3D - CAVE	29.0	0.87704743	reject

Table B.11: Wilcoxon's post-hoc analysis on visual tiredness scores of the semi-guided story experience

Pair	Statistic	p-value	Result
2D - 3D	78.0	0.002131667	accept
2D - CAVE	60.0	0.015065306	accept
3D - CAVE	9.5	0.064077506	reject

Table B.12: Wilcoxon's post-hoc analysis on physical tiredness scores of the semi-guided story experience

Pair	Statistic	p-value	Result
2D - 3D	55.0	0.004402368	accept
2D - CAVE	66.0	0.002925020	accept
3D - CAVE	19.5	0.717245184	reject

Table B.13: Wilcoxon's post-hoc analysis on headache scores of the semi-guided story experience

Pair	Statistic	p-value	Result
2D - 3D	66.0	0.003101138	accept
2D - CAVE	52.5	0.008636217	accept
3D - CAVE	1.5	0.033645363	accept

Table B.14: Wilcoxon's post-hoc analysis on ease of use of all systems

Pair	Statistic	p-value	Result
2D - 3D	83.0	0.008671996	accept
2D - CAVE	91.0	0.001468850	accept
3D - CAVE	75.5	0.035748470	accept

Table B.15: Wilcoxon's post-hoc analysis on satisfaction of all systems

Pair	Statistic	p-value	Result
2D - 3D	105.0	0.0009787065	accept
2D - CAVE	85.5	0.0051558080	accept
3D - CAVE	19.0	0.0635432498	reject

Table B.16: Wilcoxon's post-hoc analysis on navigation easiness of the non-guided story experience

Pair	Statistic	p-value	Result
2D single - 2D full	12.0	0.70545699	reject
2D single - CAVE	52.0	0.01112251	accept
2D full - CAVE	78.5	0.01894101	accept

Table B.17: Wilcoxon's post-hoc analysis on selection easiness of the non-guided story experience

Pair	Statistic	p-value	Result
2D single - 2D full	6.0	0.31731051	reject
2D single - CAVE	54.0	0.05422100	reject
2D full - CAVE	57.5	0.02585225	accept

Table B.18: Wilcoxon's post-hoc analysis on 3D perceiving scores of the non-guided story experience

Pair	Statistic	p-value	Result
2D single - 2D full	4.5	0.01293409	accept
2D single - CAVE	22.0	0.02956471	accept
2D full - CAVE	27.0	0.18762002	reject

Table B.19: Wilcoxon's post-hoc analysis on VE stimulation scores of the non-guided story experience

Pair	Statistic	p-value	Result
2D single - 2D full	13.5	0.27332168	reject
2D single - CAVE	11.5	0.00928072	accept
2D full - CAVE	12.5	0.01931950	accept

Table B.20: Wilcoxon's post-hoc analysis on visual tiredness scores of the non-guided story experience

Pair	Statistic	p-value	Result
2D single - 2D full	12.0	0.179712495	reject
2D single - CAVE	106.5	0.007114947	accept
2D full - CAVE	96.0	0.005169365	accept

Table B.21: Wilcoxon's post-hoc analysis on headache scores of the non-guided story experience

Pair	Statistic	p-value	Result
2D single - 2D full	17.5	0.527089257	reject
2D single - CAVE	91.5	0.012784595	accept
2D full - CAVE	99.5	0.002059237	accept