Developing VR applications: the TRES-D methodology

José P. MOLINA, Arturo S. GARCÍA, Víctor LÓPEZ-JAQUERO, Pascual GONZÁLEZ
Laboratory of User Interfaces and Software Engineering, LoUISE
Instituto de Investigación en Informática de Albacete, I3A
University of Castilla-La Mancha
Campus Universitario, Avda. España s/n, 02071 Albacete, Spain
{ *jpmolina* | *arturo* | *victor* | *pgonzalez* }*@info-ab.uclm.es*

Abstract. This paper presents the TRES-D methodology for the development of VR application, and identifies opportunities for creating tools that are non-VR expert friendly. This methodology is based on previous approaches in the creation of virtual worlds, both on observed practice and proposed methods, and also on methods and techniques form the software engineering and computer-human interaction domains. That previous work is also explored in detail.

1. Introduction

The development of a Virtual Reality (VR) application can became a very complex process and for that reason it should be carried out by experts and technicians with the knowledge and experience required. However, there is a growing interest in tools that allow non-expert people to develop or participate in the development of these applications. One reason for that is that the domain experts can better communicate the requirements of the application to the VR experts by using that kind of tools, reducing the number of cycles of development. Besides, the popularity that Virtual Reality gained thanks to the media produced a demand of tools that allowed average users to create their own virtual worlds too. In this sense, we can make a comparison with WIMP user interfaces.

Nowadays, for the development of desktop application interfaces, designers have visual tools that make their work much easier. This has not been always the case, at the beginning these interfaces were coded by programmers, but now these rapid prototyping tools also allow potential users to design the layout and appearance of their future application. Considering a VR application as the sum of the interface plus the logic behind, we could think of similar prototyping tools for VR. However, there are important differences. The tools used for prototyping desktop interfaces do not differ much from a 2D drawing tool, the designer just positions interface elements -from a given list- in the different windows of the application. This is so simple because the designer assumes that the user will use the keyboard and mouse to interact with the application, and the techniques used for such an interaction are well known. On the opposite, a VR application can involve input and presentation devices for which there are no standard interaction techniques. Anyway, drawing desktop interfaces with these tools is not enough as program logic must be added later, as well as in a VR application.

As for the creation of Web pages, the situation is quite similar. At the beginning, pages were made by manually adding HTML mark-ups in the text. Then, new tools emerged that hid the details of the HTML language from the author, who was able to focus just on the content, usually text and images. These tools allowed many non-expert people to create their own Web page. Soon after HTML, a new language was proposed for the creation of virtual worlds for the Web, VRML, which was standardized in 1997. VRML allowed anyone to create their own virtual world with a simple text editor, and then use the Web for its distribution. However, the number of VRML virtual worlds is extremely small if we compare it with the number of Web pages. This is partly because creating a Web page with current tools is not much different from creating a text document using a text processor. On the opposite, creating a 3D world, as it will be explained later, is much more difficult. Anyway, creating an e-commerce site is not as simple as creating a personal Web page, it is required to have technical knowledge, prepare a plan, give structure to the site, follow a method. The same happens with VR applications.

The creation of the VR application and its interface is quite different from the creation of a desktop application or a Web page. Even if some similarities can be found, Virtual Reality technology is more immature and the problems are worse, and so a structured approach together with guiding tools is very much required.

In this paper, we propose a methodology for the development of VR applications in an iterative and incremental way, which is named TRES-D and is composed of a set of sorted activities for which the involved roles are identified, the tools for carrying out these activities from different abstraction levels, and the principles and guidelines that should lead the realization of each activity. From that methodology, it will be identified the existing opportunities for creating tools that could be used by non-expert people.

This paper is structured as follows. It starts explaining why VR applications are so complex to create, and argues for iterative approaches for design. Then, a survey on previous methods is carried out, including methodologies for creating virtual worlds and VR interfaces, some based on software engineering methods, and others taken from the computer-human interaction domain. After that, the TRES-D methodology is described, and opportunities for creating tools for non-expert people are identified. The paper ends with a summary and guidelines for future research directions.

2. Virtual Reality is not easy

As it has been introduced in the previous section, the creation of VR applications involves a number of difficulties that the developers of other kind of applications do not face. One of these difficulties is the creation of 3D content. Composing a Web page with HTML language only requires learning how to mark up the text with reserved words at the beginning and the end of each block of text, but creating a virtual world with VRML requires learning how to apply geometric transformations to objects, which deserves a great mental effort. 3D design tools may help, but they usually offer complex user interfaces, partly due to the use of 2D input and presentation devices -the mouse and the screen- for carrying out a 3D task, which means that the task have to be decomposed in multiple views and operations, making a more complex dialogue. Besides, many people have difficulties in mentally merging different projected views of an object and thinking how it is in three dimensions, as explained in [1].

One more difficulty is technology itself. The keyboard, mouse and screen are well-known devices. As part of the PC concept, they also benefit from its success, which has resulted in the progress of their technology and very low prices. At software level, the handle of these devices is facilitated by operating systems, giving support to interaction techniques such as the popular "point & click" or "drag & drop". On the opposite, VR devices -no matter how spectacular they are- do not have such a huge market, the technology has not progressed so much for years and prices are not so affordable. To select the right hardware and integrate it in the system requires knowledge and experience, and as it is not usually supported by the operating system, specific programming libraries for each device are needed. Even worse, there is no set of standard interaction techniques as it is still a hot research topic, which makes the developer search the whole bibliography to find the right answers. Most times, the designer decides to program its own solutions, and so little code is reused.

3. Iterative design for the development of VR applications

According to [2], design is inherently creative and unpredictable. It is the opinion of that author that interactive system designers must blend a thorough knowledge of technical feasibility with the mystical aesthetic sense of what attracts users. [3] characterized the design as a process, a non-hierarchical one -that is, neither top-down nor bottom-up-, which can lead to solutions that will not be part of the final design, or to the discovery of new goals.

Nevertheless, the maturity that the development of conventional applications has reached allows designers to carry out their work on the basis of the previous experience, saved in the form of templates or patterns, reducing the number of paths to explore until the final design is achieved. On the opposite, Virtual Reality is still an immature field, which means that designers must explore many different solutions, making of iterative design a must [4].

In spite of that, as Shneiderman explains, in every creative domain, there can also be discipline, refined techniques, right and wrong methods, and measures of success.

4. Main roles related to the development of VR applications

In the development of a conventional application, there are different people who play different roles. One of these roles is the client, which together with the end-users forms the audience of the application. There are also domain experts, who advise the development team regarding the domain of the application. In the development team it is possible to find designers, programmers and evaluators. During the design stage, one or several programmers usually create prototypes of the interfaces following the specifications of the designers, prototypes that are then evaluated in order to refine the design of the application.

The development of VR applications also needs to involve people that create the virtual world. These people are 3D model builders and animators. To complete the production of multimedia content of the virtual world, it can also involve other people, such as graphics designers, sound technicians and writers. Some of these people can collaborate with programmers to create the prototypes of the virtual world during the design stage, again led by the designers, who create the specifications of both the interface and the world model.

Another difference is that both the designers and the evaluators must be experts in computer-human interaction, as Virtual Reality is a technology where, more than in any conventional application, the human factor is extremely important. One of these expert people takes the role of the project leader, who -as an expert in Virtual Reality- coordinates the communication and work of the team.

In a similar way, the programmers must have special skills to produce the VR interface, whose non-conventional input and presentation devices require programming libraries that are not usual. They also have to program the application logic. Depending on the complexity of it, it could be necessary to involve software engineers who, working in parallel to the interaction designers, carry out the design of the system architecture of the VR application.

5. Methodologies for building virtual worlds

From the practice in building virtual worlds, several methods have been observed and several other methodologies have been proposed. [5] and [6] agree that current practice in building virtual worlds consist in, firstly, building the models using tools such as 3D Studio, then exporting them in a certain file format that allows them to be used in the development environment of the VR application and, finally, adding functions and behaviours using graphics libraries or simulation packages. However, the problem is that by applying this method the resulting virtual worlds can be visually compelling, but with a faulty interaction that leads to usability problems. This is because neither behaviour nor interaction is taken into account before defining the visual appearance and geometry of the objects of the world. For example, according to [7], building virtual worlds with EON Studio is as easy as performing these three steps, but our own experience with that tool -even with less demanding projects as virtual walkthroughs- confirms that is not enough.

Actually, according to a study carried out by [8] with a group of developers of virtual worlds, the real practice is slightly more complex. In that study, Kaur identified a set of common activities, sorted in the following way: 1) Requirements specification; 2) Gathering of reference material from real world models; 3) Structuring of the graphical model and, sometimes, dividing it between designers; 4) Building objects and positioning them in the virtual environment (VE); and 5) Enhancing the environment with texture, lighting, sound and interaction, and optimising the environment. Kaur explained that the designers build and test iteratively, but the users rarely participate in these tests.

Similar to the methodology identified by Kaur in her study is the method explained in [9] for the creation of X3D worlds, consisting of four phases: design, model, assemble, and optimize. In this method, the design phase is meant to determine the performance characteristics and sketch the world. Also similar, but much more detailed, is the method explained in [10] as a tutorial about creating a 3D interactive product using VRML. For both methods, it is important to carry out cycles of optimization and test.

Nevertheless, as Kaur describes, the solely application of this method does not assure that the resulting virtual worlds are free of usability problems. Kaur was surprised that human factors were not among the issues that designers considered. For that reason, in addition to structuring the process in a set of activities, Kaur felt necessary to provide guidelines to help designers in the creation a usable virtual world. To obtain these guidelines, Kaur based her work on theories of interaction, and present the guidelines in the form of HTML pages to be easily accessed by designers. Besides, Kaur redefined the process of building virtual worlds in the following way: 1) Define requirements; 2) Specify components in VE; 3) Specify interactions; 4) Design components; 5) Design interactions; 6) Build environment; and 7) Evaluate environment. Kaur demonstrated in a new study the usefulness of her guidelines.

Structure the process and guide the designer are also the objectives of the VEDS (Virtual Environment Development Structure) methodology, described in [11; 12], but more in detail in [13]. This methodology is the result of many years of experience building virtual environments at the VIRART group. The number of activities that identifies is higher than the previous methods, due to the detail that this methodology uses to describe the process. It is remarkable the attention given to the specification of requirements, the distinction between overall design and detail design, and the proactive tools that are proposed to guide the designer -described in [13]-. Curiously, the authors make a distinction between building and implementation, considering the latter as an activity much closer to the deployment of a conventional application.

6. Use of software engineering for building virtual worlds

Software engineering can be used for the realization of some of the phases of the process, or even to structure and guide the whole process. According to [14], the design of a VE is a tension between engineering and aesthetic, between structure and perception. With this principle in mind, Fencott took the method identified by Kaur in her study, and proposed a new methodology with the following stages: 1) Requirements modelling; 2) Conceptual modelling; 3) Structural modelling; 4) Perceptual modelling; and 5) Building. For the perceptual modelling, Fencott proposed using his Model of Perceptual Opportunities, based on giving semantics to the objects of the world. For the structural modelling, he suggested to use UML diagrams, influenced by the work of [McIntosh, 2000], who uses the UML diagrams of use cases, objects, collaboration and sequence for describing the structure, behaviour and function of the virtual world, following the SBF model.

The distinction of form, behaviour and function is also the basis of the CLEVR methodology [15], following the work described in [5]. This methodology consists of three stages: 1) System specification; 2) Form, behaviour and function are refined by means of simulations and executions until the right performance is reached; and 3) Other aspects, such as presence and special effects, are taken into account. Message sequence diagrams complement class diagrams in the first stage. Flow diagrams and script languages are used to specify functions, whereas state diagrams serves for behaviour. Form is described using VOS (Visual Object Specification), an object-oriented representation.

One methodology that uses the software engineering as the core of the process of building VEs is SENDA [16]. The authors explain that, this way, the application of their methodology is not much different from previous experience of engineers. The specific techniques are provided by disciplines such as computer-human interaction and artificial intelligence. More precisely, one thing that does make SENDA different from the development of conventional software is the inclusion of activities such as the definition of specific requirements and 3D design, which are needed for the creation of VR application.

7. Beyond PC-based virtual worlds

The methodologies described so far assume that the virtual world will be rendered in a PC, and the user will interact with it using common PC devices, the keyboard, mouse and screen. The use of non-conventional devices is not considered in those methodologies or is almost overlooked. These devices force more attention to design and creation of the interface, specifically as accounts to interaction techniques because, as it has been mentioned before, there is not a standard set of them.

During the realization of the INQUISITIVE project, a great research effort was made in the domain of design and reuse of interaction techniques [6]. One result of that research effort was a hybrid model called Flownets that allow us to specify the interaction in terms of continuous and discrete processes by combining flow diagrams and Petri nets [17], supported by the MARIGOLD toolset [18]. Another result was a toolkit of interaction techniques created at the Rutherford Appleton Laboratory [19; 20].

Similar in concept to Flownets is PMIW, which combines flow diagrams with state diagrams [21]. Based on this previous work, [22] proposes a methodology named VRID (Virtual Reality Interface Design), focused on the design of the user interface of the VR application. Starting with the description of the system in natural language, the methodology proceeds as follows: 1) Identify incoming data flows to the interface; 2) Identify the objects; 3) High-level modelling of the objects; 4) Low-level modelling of the objects, where PMIW is proposed for formalizing both the behaviour and the interaction.

Another important reference is [23], which -based on the previous work of Kaur- describes a methodology consisting in the analysis of the requirements, domain and tasks, and the design and evaluation of the virtual environment. Through all the process, Sutcliffe remarks the role of technology, both in the design of interaction -according to the desired naturalness- and in the evaluation of it -usability must be achieved within the constraints of technology-.

8. Methodologies from computer-human interaction

In the previous section, it has been mentioned a methodology that is focused on a very important aspect of a VR application: the design of the user interface. Related to it, the field of computer-human interaction is plenty of methodologies that can add new points of view to the development of a VR application. For example, [1] explains a methodology for designing interfaces that considers the user interface from four different levels of abstraction: conceptual, functional (meaning), sequence (syntax) and hardware binding (lexicon). Thus, this methodology is composed of the following steps: 1) Definition of requirements; 2) Conceptual design; 3) Functional design; and 4) Sequencing and hardware binding. Similarly to the cycles of optimization and test that characterized the practice of virtual world builders, Foley also proposes to build prototypes and evaluate them in an iterative cycle.

Another methodology that separates the functionality of the interface from its presentation is OVID (Object View Interaction Design), based on the object model-view paradigm. The end-to-end approach of this methodology [24] is composed of four phases: 1) Discover; 2) Design; 3) Develop; and 4) Deploy. In the second phase, design, it is made a distinction between abstract design -independent from the platform- and presentation design -depends on the platform-. In the same way as the software engineering-based methodologies described before, OVID makes use of UML diagrams to describe the interface.

Another approach to be taken into account is commercial methodologies such as, for example, LUCID (Logical User-Centred Interaction Design) [2; 27]. This methodology consists of the following phases: 1) Envision; 2) Discovery; 3) Design foundation; 4) Design detail; 5) Build; and 6) Release. Although this phases can make this methodology similar to the previous ones, the difference in methodologies such as LUCID is the attention paid to management strategies that are used to keep to the schedule and budget, the specification of detailed deliverables for the various stages of design, and the incorporation of cost/benefit and return of investment (ROI) analysis to facilitate decision making.

Model-based design and automatic generation of user interfaces based on model compilation is another interesting approach. One example of this is the IDEAS methodology [25], which we used as a basis for a new methodology named IDEAS-3D, aimed at the creation of both 2D and 3D user interfaces [26], and that allowed as to gain experience to define the TRES-D methodology, which is described in the following section.

9. The TRES-D methodology

The TRES-D (ThREe dimensional uSer interface Development) methodology is introduced with the aim to put together the best of those previous approaches in an iterative and incremental development process that can be adapted to the variable complexity of different developments. The basis of this methodology is a set of activities for which we want to identify the roles that are involved, tools to carry out the activities with different degrees of detail, and principles and guidelines to help developers in the completion of the activities. These activities are distributed along an ordered set of stages, which are:

1. **Initial requirements**: In this stage, the client meets the project leader to set up the initial requirements of the application. The leader, as an expert in VR, makes use of their experience to discuss and clarify at this first moment some of the objectives, while the rest of them will need to be studied in depth. That person also prepares a first budget and time planning for the study of requirements and the production of a concept design.
2. **Understand requirements**: As stated in [1], before carrying out the design is necessary to understand what is meant to be accomplished, task that is performed by some of the members of the development team playing the role of analysts. At the end of this stage we want to know what kind of application is to be developed, which are the characteristics of

the people that will use it, and what tasks will be carried out by them. For that purpose, user profiles -which include information such as required skills, expected use, current attitude- are made. As regards the tasks, the analysts first become familiar with the domain of the application, and then analyze the tasks with the collaboration of the domain experts, who help the development team in the creation of a list of key scenarios. If there is a current solution for these tasks, that solution can be studied by means of ethnographic studies, or specific problems can be analyzed by the performance of tests and evaluations; in both cases, user collaboration is basic.

3. **Concept design**: With the data gathered by the analysts, a group of designers then start working on different solutions for the application. Brainstorming is a technique that can be useful at the first work meetings. As the ideas flow, they can be captured in the form of written scenarios, sketches or storyboards. Multimedia tools can be used to create early prototypes from these ideas. Furthermore, designers can produce videos to show with special effects what is not done yet. No matter the form, it must show who do what, with what and where. And through all this creative process, potential users and domain experts participate giving their own opinion. Same as VR programmers and 3D models builders, who discuss with the designers the feasibility of the solutions, taking into account both hardware and software. The main aim of this stage is to list the tasks that the user will be able to perform with the application, and what hardware and software will make it possible. At the end of the stage, a complete report will be written, describing the different solutions and detailing not only their benefits, but also the time and cost of development, and risk assessments, so that a proper decision can be made.
4. **Iterative design**: Once one of the solutions is selected for its realization, the design of the application is done from two different levels: abstract and presentation. Actually, the interaction designer works alternatively from these two points of view in iterative cycles until the final design is completed. At each cycle, interaction designers carry out the specification of each of the components of the application using tools of increasing detail, as the ones we describe in [28]. More precisely:

- ***Abstract design***: Independent of the platform. Plans of the virtual world are drawn, detailing the position of objects and navigation paths. The behaviours of objects are identified, as well as the way these objects react to user actions. The high-level dialogue between the user and the objects is obtained through decomposition of the tasks until individual operations are identified; these operations are functions that can be applied to individual objects, and their specification also includes their preconditions, parameters and outcomes. They are the abstract views of the objects. Through all this process, the designers collaborate with the domain expert and carries out tests and evaluations to make sure that the user's model is the right one.
- ***Presentation design***: Depends on the hardware and software selected at concept design. This time the low-level dialogue is detailed, which starts with the selection of interaction styles for each group of operations. The information required for each operation is then transformed into one or more basic interaction tasks, which are associated to concrete interaction techniques, whose behaviour is then specified in terms of input device actions. In parallel, the world structure is detailed, and so is the geometry and appearance of each object. As we already know the function of each object, then we know the visual requirements of each of them, and so we can proceed to carry out the gathering of reference material from real world models. The virtual world is integrated with the whole application interface, and some interaction and navigation aids are added to make the application more usable. Through all this process, the designers will count on programmers and 3D model builders in order to create prototypes that can be evaluated by experts or used in tests with potential users.

5. **Building and implementation**: The design and its successive refinement result in a set of specifications and prototypes that are the documentation used by 3D model builders to

create and animate the virtual world, and by programmers to create the interface and program the logic behind. Everything will be integrated in the hardware and software platform selected for the application, performing last minute changes.

6. **Deploy and maintenance**: Once the development of the VR application is finished, it can not be said the whole work is done. Due to the characteristics of the technology, much more fine tuning and maintenance is required than a conventional application.

As it has been mentioned before, TRES-D aims to be a flexible methodology so that its stages and activities can be adapted to the specific requirements of each application. As an example, one of the recent projects where we used this methodology involved the creation a 3D virtual store, whose structured development process is detailed in [29].

10. Opportunities for introducing tools for non-experts

As it can be observed through all this paper, the process of creation of a VR application is made of several activities where different people are involved, possibly playing different roles, some of them as experts in the realization of the activity, some others as the ones who have the information needed for the completion of the activity. However, as it has also been mentioned before, if there is not a common vocabulary then misunderstandings can occur and, in the worst case, can lead to the creation of an application that is not what the client expects. This means that new cycles of application redesign have to be made.

To avoid this situation, one solution is to let the person who owns the information to directly participate in the realization of the activity, even though they are not experts in that domain. For that purpose, special tools are needed, more suitable to their skills, although some help and advice from the expert can be given.

From the TRES-D methodology, it is easy to identify different activities where this kind of tools can be useful, which are those activities where two or more different roles are meant to collaborate. One example is the presentation design, where the interaction designer collaborates with programmers and 3D model builders in the creation of prototypes of the virtual world. The interaction designer, although an expert in computer-human interaction, is not meant to be an expert in programming or 3D object modelling. However, if the interaction designer was given tools that did not require those skills, they could create the prototypes by themselves. This example, as it was mentioned in the introduction section, can be already found in the development of conventional applications. In this case, visual tools as the ones that form the MARIGOLD toolset -cited before- can help the interaction designer to build the interface, whereas they can build a simplified view of the virtual world using a tool such as [30] -which only requires drawing the floor of the world-. Furthermore, the expert in the domain could collaborate in the production of content if the appropriate tools were given to them, as explained in [31]. In any case, those tools do not substitute the work of the technician and the expert in the activity, who will have to work from the outcome in order to transform it into the final VR application.

However, it is our opinion that as important as creating tools for non-expert people is helping the VR designers and technicians to take advantage of the experience gained in the field through all these years of research, so they are not overcome with the huge existing bibliography. In this sense, we are working on the development of tools that help the designer in the selection of the best solution for each application, and one example of that is the tool that, in the form a decision tree, we use for choosing the appropriate selection/manipulation technique (fig. 1), based on the study that we describe in [32]. These tools can also serve as a basis for wizard-based application creation, which in the end could also be used by non-expert people.

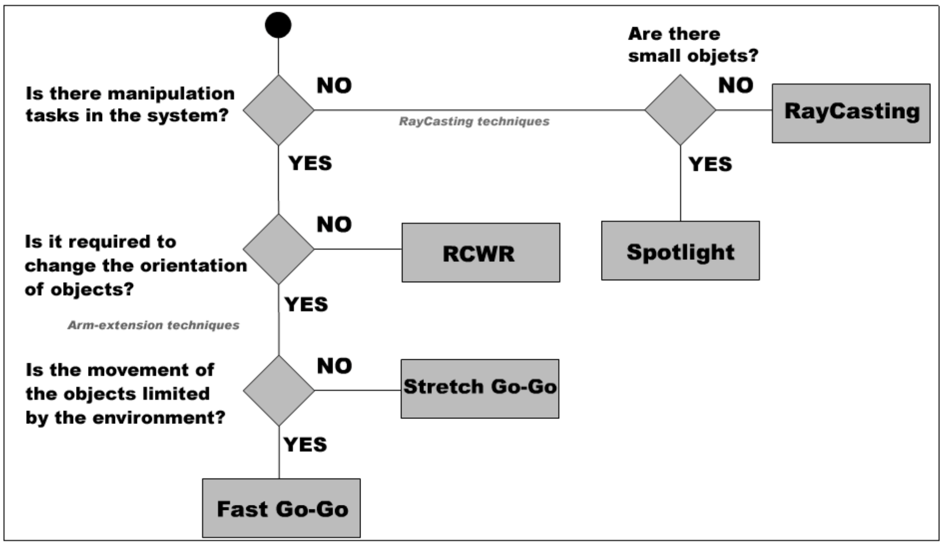

Figure 1. Decision tool for choosing a selection and manipulation technique

11. Conclusions

The process of developing VR applications is really more complex than building desktop interfaces or composing Web pages. A structured approach is part of the solution. Several methods have been observed or proposed in the past, and many of them have been reviewed in this paper. In order to put together the best of those approaches, the TRES-D methodology has been proposed as an iterative and incremental development process that can be adapted to different kinds of VR development. This is an ongoing work, so improvements are expected to be made. This framework has also been used in this paper to identify some opportunities for creating tools that are non-VR expert friendly, which have an increasing demand. However, we also argue that there is also an important necessity to create tools for guiding the designer, which in the end could also be used by non-expert people. In this paper, we have shown one example for selection techniques, but more tools like this are needed. Our research efforts are currently focused on interaction techniques, but in the future we expect to produce other tools for many other aspects of the development.

References

[1] J.D. Foley, A. van Dam, S.K. Feiner and J.F. Hughes, Computer Graphics Principles and Practice, 2nd ed. in C. ISBN: 0201848406. Addison Wesley, 1996.

[2] B. Shneiderman, Designing the User Interface: Strategies for Effective Human-Computer Interaction. ISBN: 0-201-69497-2. Pearson Education, 1998.

[3] J.M. Carroll and M.B. Rosson, Usability Specifications as a Tool in Iterative Development. In: H.R. Hartson (ed.), Advances in Human-Computer Interaction 1, 1985, pp. 1-28.

[4] R. Stuart, The Design of Virtual Environments. ISBN: 1-56980-207-6. Barricade Books, 2001.

[5] G.J. Kim, Kk.C. Kang, H. Kim and J. Lee, Software Engineering of Virtual Worlds. Proc. of the ACM Symposium on Virtual Reality Software and Technology, 1998, pp. 131-139.

[6] S. Smith and D. Duke, Design Support for Virtual Environments. Visualization and Virtual Environments Community Club (VVECC) Workshop: Design of Virtual Environments, 2001. URL: http://www-ais.itd.clrc.ac.uk/VVECC/proceed/vedesign/

[7] EON Reality. URL: http://www.eonreality.com

[8] K. Kaur, Designing virtual environments for usability. PhD thesis, Centre for HCI Design, City University, London, 1998.

[9] L. Daly, D. Brutzman and A. Hudson, Introducing X3D, Building X3D Worlds. Tutorial notes. SIGGRAPH 2002, San Antonio. URL: http://www.realism.com/x3d/presentations/s2002/

[10] B. Hay. Creating an Interactive 3D Product Using VRML. Online tutorial. URL: http://www.virtualrealms.com.au/vrml/tute01/tutorial.htm
[11] J.R. Wilson, R. Eastgate and M. D'Cruz, Structured Development of Virtual Environments. In: K. Stanney (ed.), Handbook of Virtual Environments. Lawrence Erlbaum Associates, 2001.
[12] M. D'Cruz, A.W. Stedmon, J. Wilson and P. Moderny, Building Virtual Environments using the Virtual Environment Development Structure: A Case of Study. 10th International Conference on Human-Computer Interaction, HCI International 2003, Crete, Greece, June 23-27.
[13] R. Eastgate, The Structured Development of Virtual Environments: Enhancing Functionality and Interactivity. PhD Thesis, University of York, 2001.
[14] C. Fencott, Towards a Design Methodology for Virtual Environments. User Centred Design and Implementation of Virtual Environments (UDCIVE) Workshop, University of York, 1999. URL: http://www.cs.york.ac.uk/hci/kings_manor_workshops/UCDIVE/.
[15] J. Seo, D.N. Kim and G.J. Kim, VR Object Reuse through Component Combination. International Workshop on Structured Design of Virtual Environments and 3D-Components, Paderborn, Germany, February 19th, 2001. URL: http://jerry.c-lab.de/web3d/VE-Workshop/
[16] M.I. Sánchez, A. de Amescua, J.J. Cuadrado and A. de Antonio, Software Engineering and HCI Techniques Joined to Develop Virtual Environments. International Conference on Software Engineering, ICSE 2003, Portland, Oregon, USA, May 3-4.
[17] S. Smith, D. Duke and M. Massink, The Hybrid World of Virtual Environments. *Computer Graphics Forum* **18**(3), September 1999, pp. 287-307.
[18] J.S. Willans, M.D. Harrison and S.P. Smith, Implementing Virtual Environment Object Behaviour from a Specification. Workshop on Guiding User through Interactive Experiences: Usability Centred Design and Evaluation of Virtual 3D Environments, Paderborn, Germany, April 13-14, 2000. URL: http://wwwcs.uni-paderborn.de/cs/ag-domik/workshop/Index.htm
[19] L. Sastry, M. Wilson and D. Boyd, Interacting in Task-oriented Virtual Worlds. Visualization and Virtual Environments Community Club (VVECC) Workshop: Design of Virtual Environments, 2001. URL: http://www-ais.itd.clrc.ac.uk/VVECC/proceed/vedesign/
[20] D. Boyd and L. Sastry, Development of the INQUISITIVE Interaction Toolkit-Concept and Realisation. User Centred Design and Implementation of Virtual Environments (UDCIVE) Workshop, University of York, 1999. URL: http://www.cs.york.ac.uk/hci/kings_manor_workshops/UCDIVE/
[21] R.J.K. Jacob, L. Deligiannidis and S. Morrison, A Software Model and Specification Language for Non-WIMP User Interfaces, *ACM Trans. on Computer-Human Interaction* **6** (1), March 1999, pp. 1-46.
[22] V. Tanriverdi and R.J.K. Jacob, VRID: A Design Model and Methodology for Developing Virtual Reality Interfaces. ACM Symposium on Virtual Reality Software and Technology, 2001, pp. 175-182.
[23] A. Sutcliffe, Multimedia and Virtual Reality: Designing Multisensory User Interfaces. ISBN: 0-8058-3950-X. Lawrence Erlbaum Associates, 2003.
[24] D. Bardon, D. Berry, C. Bjerke and D. Roberts, Crafting the Compelling User Experience Using a Methodical Software-Engineering Approach to Model Users and Design. Tutorial notes. URL: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/publish/1650
[25] M. Lozano, I. Ramos and P. González, User Interface Specification and Modeling in an Object Oriented Environment for Automatic Software Development. IEEE 34th International Conference on Technology of Object-Oriented Languages and Systems, TOOLS-USA 2000, pp. 373-381.
[26] J.P. Molina, P. González, M.D. Lozano, F. Montero and V. López, Bridging the gap: developing 2D and 3D user interfaces with the IDEAS methodology. In: J. Jorge, N.J. Nunes, J.F. Cunha (eds.), Proc. of the 10th Workshop on the Design, Specification and Verification of Interactive Systems. Springer-Verlag Lecture Notes on Computer Science, 2003, pp. 303-315.
[27] Cognetics Corp., The LUCID Framework. URL: http://www.cognetics.com/lucid/index.html
[28] J.P. Molina and P. González., Model-Based Design and New User Interfaces: Current Practices and Opportunities. In: H. Trætteberg, P.J. Molina, N.J. Nunes (eds), Proc. of the 1st International Workshop on Making Model-Based User Interface Design Practical: Usable and Open Methods and Tools. IUI/CADUI 2004, Madeira, Portugal. January 13. URL: http://CEUR-WS.org/Vol-103
[29] A. Robles, J.P. Molina, V. López and A.S. García, Even Better Than Reality: The Development of a 3-D Online Store that Adapts to Every User and Every Platform. Proceedings of HCI International 2005, Las Vegas, Nevada, USA, July 22-25.
[30] VR-Mog. URL: http://www.comp.lancs.ac.uk/computing/research/cseg/projects/vrmog/
[31] A. Celentano and F. Pittarello, A Content Centered Methodology for Authoring 3D Interactive Worlds for Cultural Heritage. In: D. Bearman and F. Garzotto (eds.), International Cultural Heritage Informatics Meeting, ICHIM, Milan, Italy, September 3-7, 2001, Vol. 2, pp 315-324.
[32] A.S. García, J.P. Molina and P. González, Exemplar VE design guidance tool for selection and manipulation interaction techniques. Proceedings of HCI International 2005, Las Vegas, Nevada, USA, July 22-25.