VHD++ Development Framework: Towards Extendible, Component Based VR/AR Simulation Engine Featuring Advanced Virtual Character Technologies

Michal Ponder [(*)], George Papagiannakis [(**)], Tom Molet [(**)],
Nadia Magnenat-Thalmann [(**)], Daniel Thalmann [(*)]

(*) Virtual Reality Lab (VRlab)
Swiss Federal Institute of Technology (EPFL)
e-mail: [name.surname]@epfl.ch

(**) MIRALab,
University of Geneva
e-mail: [name.surname]@miralab.unige.ch

Abstract

This paper presents the architecture of the VHD++ real-time development framework that after several years of intensive research, design, and development effort has been released and enters its validation phase. This paper discusses the key aspects involved in architectural structure, design and practical implementation of an efficient, flexible and extendible real-time software framework based on the modern 3D game-engine design principles. This framework supports researchers and application developers with rapid, component based development of VR/AR systems featuring advanced virtual character simulation technologies. The discussion covers motivation, main concepts, survey of related work, the main functional and design requirements, design principles and key architectural elements. It concludes with the initial validation results including overview of existing VHD++ based VR/AR virtual character simulation applications

1. Introduction: The Demand

The very recent revolutionary advancements in computer graphics and in real-time virtual character simulation technology put a completely new light on the VR/AR systems and in particular on their scaled down cousins: interactive video games. In the extremely competitive environment there is only one rule to follow: deliver always newer, always faster and always more in shorter time. Chasing that continuous tough demand results in system complexity rising exponentially with the number of components and semantically distant technologies being integrated under a single interactive, real-time, audio-visual application roof. This explains the current, rapidly growing interest of both research and industry in advanced, complexity curbing, framework oriented, middleware solutions that while well established in other IT domains are just coming to life in the interactive real-time audio-visual simulation field.

Significant investments combined with intensive research and development in the core fields made the technologies, once affordable only to few research laboratories and government based institutions, now widely available. We are reaching the point where majority of researchers and developers has nearly equal access to similar technologies providing comparable price/performance ratio.

It seems that in the near future the success of a particular interactive real-time audio-visual product or service will unlikely be a function of any certain quality or the number of individual qualities involved. Instead it will be based on the way or rather the art of combining multiple qualities, relating them and letting them interplay inside one consistent and extendible framework. The complexity and the technologies that will let handle it will distinguish the successful products and services in the coming new era of interactive audio-visual applications.

Quoting Bill Raduchel, former Chief Strategy Officer at Sun Microsystems, Inc.: "*The challenge over the next 20 years will not be speed or cost of performance; it will be the question of complexity*".

2. Motivation: Curbing Complexity

2.1. Common Experience: Facing Complexity

Carrying on proprietary research activities while being at the same time involved in demanding, tightly timed, development projects targeting concrete applications is a daily reality of many research groups. Overall complexity of the resulting applications reaches the levels that one can barely handle with the methodologies currently at hand.

At the certain moment one faces the following problems: a) how to allow many researchers and developers (~50 in authors' case) to carry on research within their respective domains b) so that the results can be made easily and quickly usable by the application builders c) who would rather reuse and compose applications instead of starting each time from scratch.

Experience shows that reuse of existing applications is very limited. Along the subsequent adaptations each application quickly reaches a maximum of its inherent architectural capacity that makes new extensions unfeasible. Moving of useful functional features between applications is far from easy and in most cases practically impossible. Integration of any new features requires intensive per-application effort in order to comply with diverse architectures. In effect continuous extension, integration and maintenance activities start to consume more time than it can be afforded.

2.2. Challenge: Building Real-Time Framework

Majority of the individual, heterogeneous though complementary VR/AR simulation technologies gain a lot of individual R&D interest and nowadays one may draw from a broad spectrum of the masterpieces. However the matter of integration of those functional components under the roof of one consistent framework is still hardly understood and there are barely any accepted guidelines available.

For this reason when searching for possible approaches and solutions we looked closer to frameworks and components being presently software engineering concepts of rapidly growing importance. Frameworks offer architectural complexity curbing through predefined, reusable system design, common class vocabulary and uniform conventions. They are "born for" extensions, customisations, and replacements. Components promote massive code reuse, and complexity hiding by provision of binary units that are assembled to quickly compose applications. The remaining question is: how to combine both and make sure that the resulting framework delivers real-time performance ?

As a result the new challenge becomes: elaboration of a real-time development framework that supports a) research and b) component based development VR/AR applications featuring advanced virtual character simulation technologies.

3. Survey of VR Frameworks

Framework based development methodologies are since some time a very active area of research and up to now number of books is available on the subject [1][2][3][4][5]. However, on the specific aspects of the real-time frameworks, the selection is much more limited [6]. Finally in case of methodologies involved in analysis, design and implementation of real-time frameworks supporting interactive audio-visual simulations, like VR/AR or digital entertainment systems, there is currently no widely referenced position available on the shelf.

Currently developers move away from constructing from scratch in favour of composing and integrating applications using frameworks and reusable components [7][8]. The natural reaction to this demand is the quickly growing number of powerful middleware solutions

On the interactive multimedia and entertainment side of the application spectrum the offer is impressive nowadays and one can choose from a wide variety of solution: Alchemy™ [18], LithTech™ [21], NetImmerse™ [22], Jack [23], RenderWare™ [24], Genesis3D™ [25], Unreal™ [26], Quake™ [27] VirTools™[28] with a price tags ranging from shared-source code GPL licenses [25] to hundreds of thousand USD. Recent flexible academic licensing policies allow as well researchers [19][20] to use UnReal™[26] for example. On the other side of the spectrum, in case of VR research and domain specific VR applications, researchers and developers may find as well quite big set of possible solutions like MRToolkit [9], WorldToolkit™ [10], dVise™ [11], Vega™ [12], OpenMASK [13], CAVELib™ [14], DIVERSE [15], VR Juggler [16][17].

When choosing the most optimal solution one needs to take into account that many of the existing tools are highly optimised for particular applications. Thus it may often occur that adapting them to continuously changing requirements, inherent to research projects, is a difficult task. That is why a new approach that combines the 3D game virtual human simulation technologies with a general applicability of research systems for rapid framework-based VR/AR prototyping, is needed and proposed by this paper. Instrumental to the ground up new research and development efforts, have been the principles explored in the earlier work of Sannier et al [30] where the fundamental ideas of VHD: "Virtual Human Director" were set.

4. VHD++ Framework Overview

Before going into more detailed discussion of the VHD++ architectural and functional aspects it is important to give a small snapshot of the framework size, development effort and use. VHD++ currently consists of ~500 C++ classes grouped in ~35 packages. The generic high performance operational *vhdRuntimeEngine* kernel (10 packages) hosts and power supplies extension plug-ins *vhdService* (currently 15 and rapidly growing). In the development effort, there are 3 kernel designers-lead programmers, 10 direct developers and a team of 40 contributing experts, researchers, developers and

designers. Due to the academic nature of VHD++ none of the above personnel is assigned full-time on these tasks. At the moment VHD++ is used in 5 EU projects that feature research and development in virtual character simulations. The active development has been going on for 2.5 years (commenced in March 2000) and its intensity is growing. These figures are not far from current state-of-the-art 3D games and engines: e.g. for the [41] 3D game engine (published by Microsoft™), 27 full time developers where assigned for its development for a time span of 3.8 years.

4.1. Design Principles

VHD++ has been designed as a strongly object oriented framework and developed entirely in C++. It features clear separation of concrete and. abstract class hierarchies providing implementation of multiple design patterns. Bearing in mind real-time performance classes and their mutual collaborations has been analysed from the point of view of the following operational domains and boundaries between them: a) time critical vs. non-critical, b) computationally heavy vs. light services, c) control vs. event driven, d) synchronous vs. asynchronous, e) containing vs. processing, f) data sharing vs. transmission, g) compile- vs. run-time configuration and resolution.

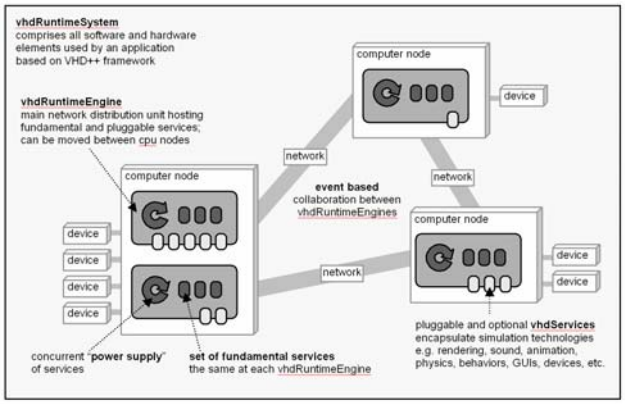

Figure 1. VHD++ architecture overview by example

4.2. Semantic Overview

The main conceptual division lines has been drawn across: a) a generic high performance operational kernel (*vhdRuntimeEngine*) and active plug-able components encapsulating heterogeneous simulation technologies (*vhdServices*[1]), b) hierarchy forming data objects representing both system and simulation state (*vhdPropertys*) and concurrent mechanisms operating on them.

vhdRuntimeSystem () represents the whole extent of an application including all software and hardware components. As such *vhdRuntimeSystem* can be regarded as a passive grouping element maintaining the highest level and the widest scope knowledge about an application state. In order to perform any job a *vhdRuntimeSystem* must feature at least one active *vhdRuntimeEngine*.

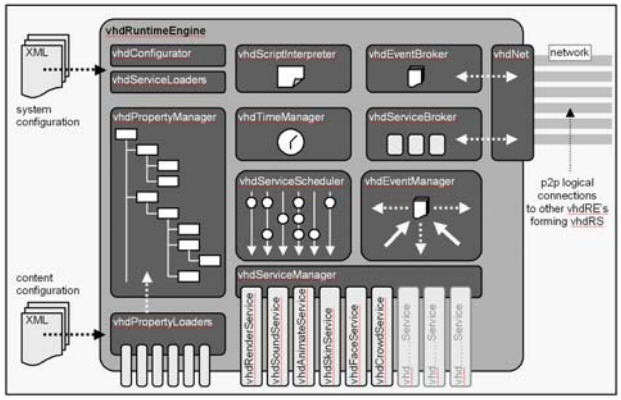

Figure 2. *vhdRuntimeEngine* **(vhdRE) key *fundamental components* and their roles**

vhdRuntimeEngine (Figure 2) constitutes a reusable architectural kernel of each VHD++ based application. It features highly optimised collaboration of a finite set of fundamental services inherent to all applications in the VR/AR simulation domain: on-demand runtime loading of simulation specific *vhdService*s (system runtime composition), configuration/data/content loading (XML, VRML, WAV, MP3, etc.), containment and serialization, concurrent data sharing, multi-threading and synchronization, time scheduling, event handling, networking, event and service brokering, object referencing, garbage collecting, etc. It hosts runtime collaboration and provides "power supply" to all *vhdService*s composing an application. The collaboration links between *vhdService*s are discovered, established and validated dynamically by the *vhdRuntimeEngine*. As a result *vhdService*s can be selected, loaded and activated at runtime, which allows for powerful runtime prototyping of application functional composition. Naturally, the price to pay for this kind of flexibility is certain runtime overhead especially in the periods of runtime (re)configuration. In case of real-time systems it is important to stop this overhead at a certain abstraction level. It must not be propagated deeper to the more fundamental levels that operate on much higher frequencies. In case of VHD++ this line is defined by *vhdService*s protecting their internal high performance workings by clear public interfaces visible to the

[1] For sake of clarity and conciseness we simply add 's' to class name as a shortcut for "multiple *vhdClass* entities".

vhdRuntimeEngine and other *vhdService*s. Finally *vhdRuntimeEngine* plays a "middleware role" by separating application abstraction level from the underlying operating system and hardware specifics.

***vhdService*s** are plug-able active software components encapsulating heterogeneous though complementary simulation technologies, for example: 3D rendering, 3D sound, collision detection, physics, skeleton animation, skinning, cloth animation, behavioural control, GUIs, camera tracking, in/out devices, etc. *vhdService*s form the main plug-in mechanism for defining application specific extensions of the framework. Applications are composed by drawing from the growing pool of existing *vhdService*s. Selected *vhdService*s are used as provided or they can be adapted to the particular requirements through derivation and overriding of virtual methods. If necessary, application builders can create new *vhdService*s that are then automatically added to the existing pool for future reuse.

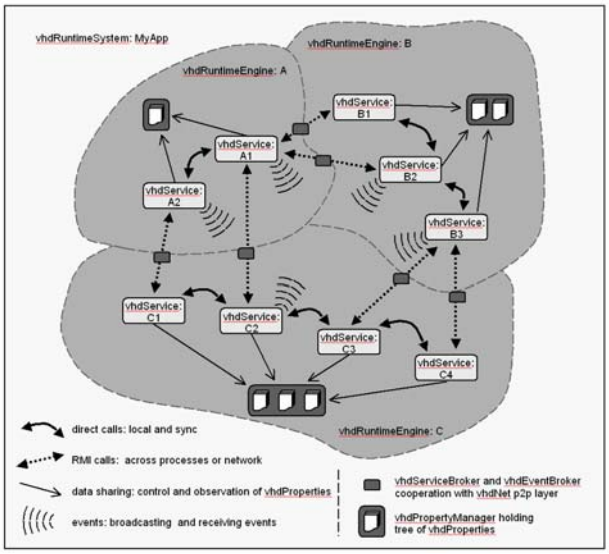

Figure 3. Types of mutual collaborations of *vhdService*s: discovered, established and validated on runtime.

vhdRuntimeEngine provides *vhdService*s with the ready-made selection of collaboration mechanisms of various type (Figure 3): asynchronous, synchronous, event reactive or/and data sharing character. Nature of collaboration predefines flexibility of a runtime deployment of a given *vhdService* over the set of *vhdRuntimeEngine*s. It may vary from unconstraint potential of execution on any of the *vhdRuntimeEngine*s to obligation of clustering with other *vhdService*s on a single, well-specified *vhdRuntimeEngine*. For example 3D rendering and collision detection services are always placed on a single *vhdRuntimeEngine* as they are of time-critical character and they need to share the scene-graph. On the other hand a vision based real camera tracking AR service and 3D rendering service despite their time-critical character can be placed on a single or separate *vhdRuntimeEngine*s since they only exchange low bandwidth camera matrix updates.

***vhdProperty*s** are data containing objects representing both system and simulation state. Hierarchical in nature, they form an "application graph" storing a single, thus consistent overall system state. For example *vhdProperty*s may represent a runtime engine or service configuration, scene-graph access points, virtual characters, behavioural states attached to those characters, sounds, scripts, animation data, etc. *vhdService*s composing an application operate concurrently on *vhdPropertie*s. Each *vhdProperty* class has a corresponding *vhdPropertyLoade*r singleton which interprets structured XML parameters to create *vhdProperty*s instances of a respective class. Developers are allowed to freely extend the existing set of *vhdProperty* and *vhdPropertyLoader* classes in order to introduce custom system state and simulation data types.

5. Initial Validation Results

When planning the first validation phase of the VHD++ framework it has been decided that instead of hypothetical test cases it is rather confronted right away with real-world requirements of the two ends of the Research and Development spectrum ("R&..", "..&D"). Numerous, multi-partner, tightly timed EU level R&D projects involving cooperation of many researchers offered a perfect though extremely demanding test environment.

5.1. Initial Validation on "R&.." Side

On this side of the validation spectrum the question is to see how far the proposed methodology meets the requirements of researchers who while working within their respective domains need to have precise and expert level (deep) access to the internals of their proprietary software modules enclosing VR/AR and virtual human simulation technologies? It is also interesting to assess if multiple researchers are able to work in parallel using a single framework and if as a result they get ready to use plug-able components (*vhdService*s) encapsulating respective heterogeneous technologies? Finally there is a clear need for the validation of the decentralized extensibility of the VHD++ framework.

In order to do so we gradually encouraged relevant researchers to migrate from their proprietary software tools toward the new framework. Each of the researchers receives usually one-day tutorial on how to enclose his/her technology in form of plug-able *vhdService* and

then how to quickly compose test applications using existing *vhdService*s and *vhdRuntimeEngine*.

Within the first 6 months more than 20 researchers moved to VHD++ and started to use it as the main software platform used to validate their respective research. The clear advantage for them is the ability to validate and show the proprietary research results in seamless combination with other simulation technologies virtually for free. No deep knowledge of other components (*vhdService*s) is needed in order to use them effectively. All researchers work independently (in parallel) and the need for the mutual communication is limited only to defining conventions and potentially sharable simulation data structures (plug-able *vhdProperty*s). In effect within the first 6 months more than 15 fully functional, plug-able *vhdService*s has been created and published for reuse. A brief overview of some of them is presented below together with the references to the papers describing encapsulated technologies (where applicable).

Audio-visual VR/AR immersion technologies:

vhdViewerService: high performance real-time multi-texture graphical 3D rendering supporting sequential stereo (e.g. shutter glasses) and two head stereo rendering (e.g. HMD), real camera image mixing (for AR applications) and "blue-boxing" occlusion effects [31] [35]. It encapsulates full support and OpenGL extensions for the high-performance Scene-graph Rendering API, OpenGL Optimizer [29].

vhdSoundService: multi-channel, real-time dynamic 3D environmental sound effects (using both emulation or EAX compliant hardware acceleration) supporting multiple speaker sets from simple stereo up to home cinema 5.1 surround sound systems.

vhdRealCameraTrackingService: robust real-time model-based real camera tracking using live video input or video sequence (for AR applications) [39].

Virtual character real-time simulation technologies:

vhdSkeletonAnimationService: multi-source animation playing, generation and mixing (HANIM [40] compliant) e.g. keyframing, procedural walking engine, real-time full body motion capture (support of Flock of Birds, MotionStar, CyberGlove), etc. [32].

vhdSkinDeformationService: skin (and skin-cloth) mesh deformation following automatically HANIM compliant skeleton animation.

vhdClothAnimationService: real-time cloth animation using highly optimised physical simulation and body to cloth collision detection [36].

vhdFaceAnimationService: MPEG-4 compliant virtual human face animation generation, playing, mixing [38] and deformation [37].

vhdVoiceService: equipping virtual humans with speech by synchronization of speech sound and respective facial animation.

vhdCrowdService: interactive virtual human crowd behaviours for population of virtual environments, VR training, etc. [33][34].

High level, multi-modal HCI technologies:

vhdSpeechRecognitionService: recognition of simple natural language commands facilitating interaction.

vhdVRNavigationService: based on head tracking (support for Flock of Birds, MotionStar systems) offering immersion dependent (wall projection, HMD) VE navigation paradigms.

vhdScenariorService: authoring and playing of interactive VR/AR scenarios.

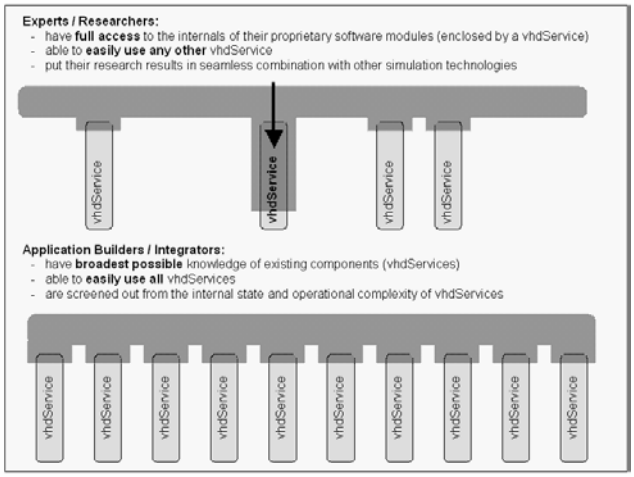

Figure 4 Needs of experts/researchers vs. needs of application builders/integrators.

Parallel development of *vhdServices* proved to be very efficient. Since all *vhdService*s conform to the common, framework wide conventions we do not experience any mutual operational clashes when plugging them into a single application. Concurrent manipulation of shared resources (*vhdProperty*s) is managed by the common kernel.

Thus it is clear that encapsulation of arbitrary complex technologies in form of a plug-able *vhdService*s pays off as non-experts can use them easily usually just by reading a flat API C++ header file while the experts still have full access to the *vhdService*s' internals through the well defined expert entry points (Figure 4). In effect each *vhdService* hides its state and operational complexity from non-experts while simultaneously not constraining the experts from advanced use required when carrying on research activities.

5.2. Initial Validation on "..&D" Side

On this side of the validation spectrum the first important question is to see whether a single framework architecture with its generic operational kernel (*vhdRuntimeEngine*) and plug-able components

(*vhdServices*) is able to support development of a broad range of VR/AR and virtual character simulation applications of diverse functional requirements? Secondly it is important to check how far the proposed methodology supports application builders who, while working on the relatively high abstraction level, must be able to grasp intuitively an overall application architecture and efficiently combine usually very broad range of simulation technologies?
The proposed framework has been confronted with the requirements of the concrete VR/AR projects featuring advanced virtual character simulation technologies. Currently VHD++ is being validated in frame of the following EU level projects: JUST, CAHRISMA, STAR, LIFEPLUS, E-TAILOR.

Figure 1. JUST immersive VR situation training of health emergency personnel: immersed trainee and execution of ordered Basic Life Support procedures.

The main objective of the JUST project is the development of an immersive VR tool for situation training of health emergency personnel. The trainee faces an interactive scenario based simulation (*vhdScenarioService*) standing in front of a large rear projection screen displaying sequential stereoscopic images (*vhdViewerService*). He/she is immersed in 5.1 surround 3D sound (*vhdSoundService*). As his/her head is being tracked by magnetic sensor he/she is able to navigate freely inside VE (*vhdVRNavigationService*). Many other *vhdServices* are plugged to the system to provide high quality virtual human simulation, HCI, etc. As opposed to usual analysis, design, development process the whole JUST application is composed of *vhdRuntimeEngine* and a selection of reusable *vhdServices*. In this way the main effort is shifted to the production of content (scenes, humans, animations, sounds, behaviours, scenarios, etc.). The system rendering 40k-polygons environment, 2x15k-polygons humans with deformable skins, faces, speech, runs on a PC (Win 2000), 1GB RAM, Pentium 2.0GHz, Quadro4 900 XGL yielding ~20fps performance. For captured in real-time video demo please refer to:
http://vrlab.epfl.ch/~ponder/VHDPP_MOVIES/

Figure 2. CAHRISMA VR reconstruction and preservation of cultural heritage: virtual human crowd simulation recreating a ceremonial liturgy.

The CAHRISMA project's objective is the realistic VR simulation of cultural heritage involving architecture, acoustics, cloths and overall reproduction of ceremonial liturgies [35] (in particular architectural and acoustic simulation of sacred edifices: Sinan's mosques and Byzantine churches in Istanbul, Turkey). In contrast to JUST here the focus is on real-time realistic simulation of very complex architectural edifices populated by crowds of virtual humans in ceremonial liturgies context. A challenging requirement is to achieve proper running of the system on a standard multimedia PC. The CAHRISMA application is also composed out of a selection of reusable *vhdServices*, but while some services are common to both projects the HCI here is completely different: Instead of VR devices it relies on the keyboard, mouse, and GUIs interaction. Therefore it does not use *vhdVRNavigationService*. On the other hand it employs *vhdCrowdService* that was not required in JUST. Furthermore, as the focus here is to build compelling, interactive and highly realistic virtual cultural reconstructions, their static lighting [29] is now based on radiosity pre-calculated illumination data. The resulting information is stored in 2-D textures (light maps), which according to the multi-pass and multi-texturing OpenGL 1.3 technique, are blended in real-time with the model's ordinary texture maps using the VHD++ Renderer (*vhdViewerService*). The system is rendering a 60k-polygons environment, 20x1.5k-polygons humans with deformable skins, while running on a PC (Win 2000),

1GB RAM, Pentium 1.5GHz, with NVIDIA Quadro 2 Pro graphics card, yielding 25fps performance.

STAR project targets AR training of hardware maintenance professionals. It will allow a trainee using a see-through HMD to walk around some real hardware devices and see one virtual human showing how to perform certain maintenance operations.

Figure 3. Early results from LIFEPLUS VR reconstruction, and life visualization in ancient Pompeii.

LIFEPLUS focuses on AR edutainment application allowing historical site visitors to see reconstruction of ancient life blended into the real site scenery. The project concerns cultural heritage reconstruction and preservation of the ancient Pompeii, Italy.

E-TAILOR focuses on the real-time 3D cloth simulation and possibility of virtual cloth try-on applications that can be accessible through the Web.

Those first concrete validation results demonstrate that the VHD++ framework architecture with its generic *vhdRuntimeEngine* and *vhdService*s is able to support diverse functional requirements of VR/AR and advanced virtual character simulations .

In each case the application builders strongly benefit from the reusable architecture of the VHD++ framework. As a result the time-consuming application design phase can be drastically reduced. In addition a risk of making common design mistakes is limited by reuse of ready-made and tested design solutions. With time application builders acquire certain intuition about the architectural elements and patterns. It helps a lot in grasping of an overall application map as well as in communicating complex design concepts more efficiently.

Rapid application prototyping has been found robust and simple. Application builders can quickly validate their ideas by composing applications out of the existing *vhdService*s and using existing simulation data classes (*vhdProperty*s) together with their respective XML configurable loaders (e.g. VRML models, HANIM human models, sound samples, animation keyframes, etc.). Broad reuse of already tested components (*vhdService*s) increases in each case application robustness.

Finally, while building very complex simulation systems, application developers can rely on VHD++ kernel fundamental services providing numerous low level functionalities inherent to nowadays systems e.g. automatic garbage collection, loading of various simulation data, dynamic configuration and loading of *vhdService*s, multithreading, scheduling, thread safe data sharing, etc.

5.3. Conclusions

In this paper we have presented the VHD++ real-time framework that supports development of high performance interactive VR/AR applications featuring advanced virtual human simulation technologies.

Addressing the most common problems related to the development, extension and integration of heterogeneous simulation technologies under a single system roof the presented solution combines both framework (complexity curbing) and component (complexity hiding) based software engineering methodologies. As a result a large scale architecture and code reuse is being achieved which radically increases development efficiency and robustness of the final VR/AR and virtual character applications.

Although we are at the beginning of the validation phase it becomes visible that the proposed framework solution provides a very efficient research environment. It offers a full power to the researchers who can easily validate and practically deliver results in form of the ready to use, plug-able *vhdService*s. From the application builder and technology integrator perspective, existing and rapidly growing spectrum of concrete VR/AR applications demonstrates the architectural generality and operational effectiveness of the reusable kernel. Quickly expanding set of plug-able components, successfully used to meet broad range of functional system requirements, facilitates considerably the work of developers who rather compose application than develop them from the scratch. Currently our efforts are focusing on a) refining validation of the existing VHD++ elements, b) detailed validation of the growing number of applications, c) building new *vhdService*s to extend further the pool of plug-able simulation technologies, d) investigation of framework networking aspect that will extend application spectrum to include networked virtual environments.

6. Acknowledgements

Research and development of the VHD++ framework is supported by the Swiss Federal Office for Education and Science in frame of EU IST STAR project.

7. References

[1] G. Rogers, *Framework-Based Software Development in C++,* Upper Saddle River, N.J., Prentice Hall, 1997

[2] D.F.D'Souza, A.Cameron Wills, *Objects, Components and Frameworks with UML*, Addison-Wesley, 1998

[3] Mohamed E. Fayad (Editor), Douglas C. Schmidt (Editor), *Building Application Frameworks: Object-Oriented Foundations of Framework Design*, John Wiley & Sons, 1999

[4] Mohamed E. Fayad (Editor), Douglas C. Schmidt (Editor), Ralph E. Johnson, *Implementing Application Frameworks: Object Oriented Frameworks at Work*, John Wiley & Sons, 1999

[5] Mohamed E. Fayad (Editor), Ralph E. Johnson (Editor), *Domain-Specific Application Frameworks: Frameworks Experienced by Industry*, John Wiley & Sons, 1999

[6] B. P. Douglass, Doing Hard Time: *Developing Real-Time Systems with UML, Objects, Frameworks, and Patterns*, Addison-Wesley 1999

[7] R. Johnson, Frameworks = Patterns + Components, *Communications of the ACM*, Vol. 40, No.10, October 1997.

[8] Szyperski, C., *Component Software: Beyond Object-Oriented Programming*, Addison-Wesley, 1997.

[9] MRToolkit from Univertisy of Alberta, (www.cs.ualberta.ca/~graphics/MRToolkit.html)

[10] WorldToolkit™ from Sense8 (www.sense8.com)

[11] dVise™ from Division (www.division.com)

[12] Vega™ from MultiGen-Paradigm (http://www.paradigmsim.com/ products/ runtime/vega/)

[13] OpenMASK (www.irisa.fr/siames/OpenMASK)

[14] CAVELib™ toolkit from VRCO (http://www.vrco.org)

[15] J.Kelso, L.E.Arsenault, S.G.Satterfield, R.D.Kriz, *DIVERSE: A Framework for Building Extensible and Reconfigurable Device Independent Virtual Environments*, Proceedings of IEEE Virtual Reality 2002 Conference, (www.diverse.vt.edu)

[16] A,Bierbaum, C.Just, P.Hartling.K.Meinert, A.Baker, C.Cruz-Neira, *VR Juggler: A Virtual Platform for Virtual Reality Application Development*, Proceedings of IEEE Virtual Reality 2001 Conference, (www.vrjuggler.org)

[17] C.Just, A,Bierbaum, P.Hartling.K.Meinert, C.Cruz-Neira, A.Baker, *VjControl: An Advanced Configuration Management Tool for VR Juggler Applications*, Proceedings of IEEE Virtual Reality 2001 Conference

[18] R.Arnaud, M.Jones, *Innovative Software Architecture for Real-Time Image Generation*, Interservice/Industry Training Systems and Equipment Conference (I/ITSC) Conference 1999, Ontario, Florida (www.intrinsic.com)

[19] V.DeLeon, R.Berry Jr, *Bringing VR to the Desktop: Are you Game?* IEEE Multimedia, April-June 2000, pp. 68-72.

[20] Virtual Story Telling (http://wheelie.tees.ac.uk/ users/ f.charles/ virtualstorytelling/)

[21] LithTech Engine (http://www.lithtech.com)

[22] NetImmerse Engine (http://www.ndl.com)

[23] Norman I. Badler, Cary B. Phillips, Bonnie L. Webber, *Simulating Humans: computer graphics, animation, and control*, Department of Computer and Information Science, University of Pennsylvania, Oxford University Press, March 1999

[24] Renderware Engine (http://www. renderware.com)

[25] Genesis3D Engine (http://www. Genesis3D.com)

[26] Unreal Engine (http://unreal.epicgames.com)

[27] QuakeIII Engine (http://www.quake.com)

[28] VirTools (http://www.virtools.com)

[29] SGI, *OpenGL Optimizer(TM) Programmer's Guide: An Open API for Large-Model Visualization* http://www.sgi.com/software/optimizer/

[30] G. Sannier, S. Balcisoy, N. Magnenat-Thalmann, D. Thalmann, *VHD: A System for Directing Real-Time Virtual Actors, The Visual Computer, Springer, Vol.15, No 7/8, 1999, pp.320-329.*

[31] S. Balcisoy, R. Torre, M. Ponder, P. Fua, D. Thalmann, *Augmented Reality for Real and Virtual Humans*, in Symposium on Virtual Reality SoftwareTechnologyProc. CGI 2000, IEEE Computer Society Press

[32] R.Boulic, P.Becheiraz, L.Emering, D.Thalmann, *Integration of Motion Control Techniques for Virtual Human and Avatar Real-Time Animation*, Proc. VRST97, ACM Press, 1997, pp.111-118

[33] Ulicny, B., Thalmann, D., Crowd simulation for interactive virtual environments and VR training systems, *Proc. Eurographics Workshop on Animation and Simulation '01, pp. 163-170, Springer-Verlag, 2001.*

[34] Ulicny, B., Thalmann, D., *Towards Interactive Real-Time Crowd Behavior Simulation*, Computer Graphics Forum, (to appear)

[35] George Papagiannakis, Gregoire L'Hoste, Alessandro Foni, Nadia Magnenat-Thalmann, *Real-Time Photo Realistic Simulation of Complex Heritage Edifices*, Proceedings of VSMM2001 (Virtual Systems and Multimedia), October 2001

[36] F.Cordier, N.Magnenat-Thalmann, *Real-time Animation of dressed virtual humans*, Eurographics 2002, Saarbrucken, Germany

[37] Stephane Garchery, Nadia-Magnenat Thalmann, *Designing MPEG-4 Facial Animation Tables for Web Applications*, Multimedia Modeling 2001, Amsterdam, pp 39-59.

[38] Sumedha Kshirsagar, Tom Molet, Nadia-Magnenat-Thalmann, *Principal Components of Expressive Speech Animation*, Proc. Computer Graphics International 2001, published by IEEE Computer Society pp 38-44.

[39] Ali Shahrokni, Luca Vacchetti, Vincent Lepetit, Pascal Fua. *Polyhedral Object Detection and Pose Estimation for Augmented Reality Applications*. Proceeding of Computer Animation 2002, Geneva, Switzerland

[40] The Humanoid Animation Group (H-Anim), http://www.h-anim.org/

[41] Bartosz Kijanka, *Gas powered Games, Dungeon Siege*, GameDeveloper, CMP Media, September 2002, pp 42-49