WebTalk04: a Declarative Approach to Generate 3D Collaborative Environments

Ugo BARCHETTI, Alberto BUCCIERO, Luca MAINETTI, Stefano SANTO SABATO

SetLab, Innovation Engineering Dept., University of Lecce, Italy
[ugo.barchetti, alberto.bucciero, luca.mainetti, stefano.santosabato]@unile.it

Abstract
Even if graphics hardware and 3D technologies are rapidly evolving and the increased internet connection speed allows to share amount of data and information between user geographically distributed, the development of networked three-dimensional applications is still complicated and demands expert knowledge. Though some collaborative 3D web technologies and applications had already been developed, most of them are concerned especially to offer an high level realistic representation of the virtual world as increasing the level of detail would mean to increase the "virtual presence" sense in the 3D world; at the same time they don't support from one hand an high level, non-expert authoring process, from the other the concept of programming flexibility and of component reuse is rarely taken into account.
We advocate the need of drastically simplify authoring and personalization phases through formal description of the interaction's sets as well as of the behavioral features and rules, that we call "collaborative metaphors", in a component oriented fashion to drive collaboration among users in the specific way a designer is intended to do. As result of previous considerations we present WebTalk04, a declarative 3D component system based on XML documents describing not only the environment formal structure of the virtual world were the action take place but also the complex interaction set of rules that control interactions between users and world objects used to stimulate certain kind of collaboration, thus effectively help fast prototyping and an easy building up of such collaborative applications. The WebTalk04 system also provide a runtime 3D rendering engine fully configurable through XML in order to easily modify virtual world settings as well as collaborative interaction rules thus allowing to control independently geometries, behaviors and contents, assigning to different developer (i.e. software developer , content developer , graphics developer , session designers) different task.

Categories and Subject Descriptors (according to ACM CCS): I.3 [Computer Graphics]: I.3.6 Methodology and Techniques – *Languages*. I.3.7 Three-Dimensional Graphics – *virtual reality*. I.3.8 Applications. D.2 [Software Engineering]: D.2.11 Software Architectures – *domain-specific architectures, declarative languages*

Keywords:

WebTalk, Collaborative Virtual Environments (CVE), Virtual Presence, Behavior, Collaborative   Metaphor, Extensible Markup Language (XML), XML Schema (XSD)

1. Introduction

The availability of 3D technologies on consumer platforms is continuously growing as result of the always improving of the 3D accelerated graphics hardware and the enhancements of the 3D software system. Moreover the widespread usage of internet as well as the improved average speed connection allows geographically distributed users to work together on specific tasks sharing large amount of data.

As result an always increasing number of web-based three dimensional applications has been developed, most of them working as virtual environments in which users are engaged in a common task sharing a virtual workspace (Collaborative Virtual Environments).

A CVE is a computer-based, distributed, 3D virtual space or set of places that support collaborative work and social play. In such places, people, provided with graphical embodiments called avatars that convey their identity, presence, location, can meet and interact with others, with agents, or with virtual objects.

Even thought CVE are increasingly becoming more and more widespread as well as the 3D technologies and design tool are, the development of collaborative three-dimensional applications seems to be deeply dependant on hard coding techniques yet too closely connected to specific web-3D formats often well suited only form particular application domains.

In our survey of the different implementations of existing commercial and academic CVE, we realized that only few of them addressed properly the major issues in developing and deploying their Collaborative VE platforms.

Our main considerations were:

- Almost all of them were applications, rather than flexible frameworks: they required extensive reprogramming for being used in different situations and for different purposes, being also inherently oriented towards code construction in a *code-centered* fashion. We wanted, instead, to create a "format" that could be used over and over, designing our application in a declarative way using simple authoring tools, thus using a *document-centered* approach since the environment can be automatically generated from this declarative descriptions.

- The authoring tools they provide are often limited to each particular web-3D format. So most of 3D scenes generated are mostly monolithic and restricted with regard to both content and programming code reuse. Furthermore poor or no support was given to the collaborative behavior design and the interaction control as mean to stimulate certain kind of cooperation among users.

- They try to reach a sense of *virtual presence* as fundamental requirement through a lifelike representation of the entire rendered scene, despite of really make the whole interactive session simply configurable in order to allow non expert users to easily define, besides the world and avatar's appearance, also collaboration rules, allowed interactions and, more generally, to delineate how the actions can take place through the composition of action and events to arouse the sense of virtual.

2. Related work

In the following section we will describe some relevant CVE research projects proposing alternative approaches to allow and control collaborative interactions and behavior composition.

DIVE project [FS98] proposes a framework to develop multi-user interactive virtual environments. In DIVE project the programming architecture is shared , distributed world database separating application and network interface and providing a scalable system to support different users on heterogeneous networks.

MASSIVE-3 System [GPS00], third edition of original HIVE project, is a CVE architecture based on a distributed database model making application behaviors explicitly visible within the Database. Each distributed database describes one well-defined portion of a virtual world. This system also support hierarchically structured virtual objects and heterogeneous computers and networks.

In literature there are also different approaches and solutions to combine 3D graphics and component technology. 3D component approaches [D01] can be divided into code-centered and document-centered solutions. The code-centered view uses component technologies oriented towards code construction using imperative programming languages. The document-centered view is based on automatically code generation from *declarative* descriptions. An example of code-

centered solution is the NSPNET-V architecture and NSPNET Bamboo component system [WZ98] where code modules operate in a cross-platform and cross-language manner. In this platform XML is used as a message interchange format and the components can be dynamically loaded at runtime. On the other hand CONTIGRA project [DA01] define a *declarative* component architecture based on X3D [X3DS] designed for the construction of Web-enable, desktop Virtual Reality applications and 3D scenes through declarative approach.

X3D supports basic 3D graphics primitives , animations, networking utilities and user interactions to define behaviors, script and communication mechanisms oriented to technical programmers not easily usable for designers and non-expert users.

The same research group proposes a concept for declaratively modeling 3D object behaviors based on X3D (Beahavior3D) [DR03].

3. History and motivations

The research projects for which, at first time, we were asked to build a networked virtual environment have a common feature: there is an overall educational "experience" (consisting in temporally separated cooperative sessions), within which 3D worlds play a well-defined role. In the virtual environment users (mostly undergraduate students) "meet" the other users, play with them, discuss with them (via chat), in a single word they interact. The whole experience is thought as real educational course, where, in each session, 8 students (from 4 different school classes) access the 3D world simultaneously and, with the help of a *guide,* walk through the different stage of the virtual world answering cultural questions, discovering clues about some specific topics and playing interactive games.

The idea to exploit the potentiality for collaboration at the maximum extent led, at the beginning, to the design and deployment of SEE, Shrine Educational Experience [DHP03], developed in partnership with the Israel Museum in Jerusalem. The whole WebTalkCube architecture (that was a previous and early version of the current WebTalk04 system), used in SEE project, was based on the 3D models generated by one of the commercial 3D design tool such as Discreet 3DStudio Max, Lightwave 3D and coded in Macromedia Flash, Shockwave and Director. The choice of the programming environment was mainly determined by the need of having a reliable web technology provider and a wide availability (with no additional cost for the users).

Main disadvantage of WebTalkCube project was that authoring and software development processes were too deeply coupled: once the 3D designer developed all geometries and models that should compose the 3D world and the objects (as a monolithic entity that was not possible to split into modules), he had to export them in Shockwave 3D (W3D), a file format, suitable to be understood by Macromedia applications. The programmer, then, had to import (at design time) everything into the Director's stage, and only then all behaviors (of the interactive objects), the events to which react and all the dynamic aspects could be hard coded. Even changing position or color to one object, forced the programmer to ask the

designer to make the changes then re-export the geometries that at the end should be re-imported inside Director's stage. Moreover the programmer, during the software development phase, had to refer to the geometric models with the name assigned in the 3D design tool, thus making necessary a continuous interaction between him and the 3D designer. Furthermore this development process made necessary, for the programmers, to learn 3D Studio Max, with waste of time, additional costs and a little confusion about the competences required to the different actors.

This time expensive and deeply involved development process forced us to reduce at the minimum all changes and fine tuning improvements that was need during the testing and even if a little bit of user personalization was made (avatar's name, html static links, etc.) it was always too tricky to be really effective.

Learning@Europe, the next academic research project in which we were involved, was sponsored by Accenture Foundation and was aimed at European high school students dealing with European history.

Because of the overall high number of participants expected (in the first test phase of the project more than 1000 students, from 6 different European countries, took part to the experience) and above all because of the need to personalize each session's contents (such as images, movies, questions, objects) and interaction rules (i.e. the ability to chat to each other or the move some object) depending on specific users participating and the particular topic given to the session we had to devise a different more flexible approach for the generation of collaborative 3D environments which could grant:

- a greater easiness of configuration both of the virtual environment settings and the collaboration issues
- a faster session prototype process

also taking into account that, thought a lot of session is very similar, sharing the main common features, there are however a lot differences due especially the different citizenship of the users thus causing: different skins of the avatars, different images of the place they are from to be loaded in their virtual dome, different clues hidden in the virtual world, different question to answer.

On the ground of previous considerations, then the main goals of WebTalk04 (WT04) architecture were:

- to provide a flexible and highly decoupled development environment to allow different developer teams to design and deploy specific 3D collaborative environments dealing with only the specific task (3D design, contents assembling, story board design, software develop) they are expert in, without regarding any other task.
- to allow to easily define and set up sessions' prototype,

meant as set of linked 3D stages that works as a mere place were the action take place sharing the same kind of virtual experience (be it educational, entertaining, informative..), main collaboration features and the same environmental containers, from which derive different, context specific instances of virtual world, well suited for different users and situations.

- to convey these virtual environment a high sense of virtual presence, leaving the full graphic driven paradigm to switch towards a new one that takes the overall application design into account and to provide a set of interaction features to drive the collaboration among users in the specific way the designer was intended to do.

In this paper then we propose a declarative format to reduce programming need for non expert designer of the collaborative experience, for this reason we'll examine the major issues concerned in the formal description of a CVE both from a static point of view, regarding on how the virtual word appears, when it is generated, and from a dynamic point of view regarding on how it can evolve during the collaborative session trough a declarative description of the *Event Conditioned Actions* (ECA) that can be performed in the virtual environment.

4. XML-based implementation

In this section we will show how the WebTalk04 declarative system is implemented using a markup language coded with XML Schema [XSD01]. Since a homogeneous encoding is provided on all levels, this is a consistent approach for all abstraction stages starting from scene graph level up the description of the complex user's interactions.

Moreover an XML representation is well suited to describe the 3D scene as structured data, that can be processed without paying attention to how the data should be presented. The conversion stylesheets in fact allow the switch from one format to another. Through different XSL files, the content of a WT04 XML scene graph file could be easily converted to VRML or X3D, to pretty-printed HTML, or to any number of other formats.

Expression of scenes in XML enables application of a wide range of existing and emerging XML-based tool for transformation, translation, and processing. XML provides numerous benefits for extensibility and componentization, as well as the ability to develop well-formed and validated scene graph, an extremely valuable constraint since "broken" 3D content would no longer be allowed to escape
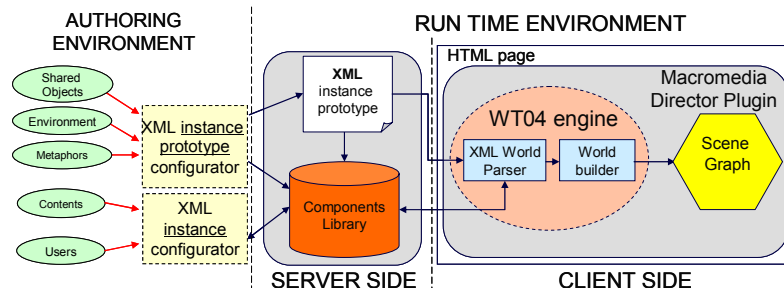

Figure 1: WebTalk04 technical architecture

onto the Web where if might cause larger scenes to fail.

Finally a formal XML description of the scene graph and the user-to-user or user-to-object interactions seems the best "interface" between the different WT04 subsystems represented within the whole architecture in figure 1:

- *WT04 runtime environment*, in which this XML file works as a declarative description of how the virtual world have to be rendered as well as of in which way the action can take place and evolve.
- *WT04 authoring environment* where the designer defines and fine tunes the world structure and the specific behaviors attached to each user or interactive objects.

As shown in Figure 2, we devised our logical architecture as a 3 tier system detailed as follows:

1. *Scene graph and behavior schema*: coded through XML-Schema it represents the structure of a valid WT04 compliant XML-instance, where it has been defined the elements composing XML-instances, hierarchical rules between elements and supported data types.
2. *Scene graph and behavior instance prototype*: a XML document representing the skeleton of a collaborative session where designer had already fixed geometries forming the virtual environment as well as the main users' interaction rules.
3. *Scene graph and behavior instance*: the final XML instance completed with all information and contents depending on the specific context (specific users involved, specific target of the experience, particular topics given).

When an *XML instance* is generated, system is ready to start a collaborative session. The XML file is sent (at run time) to the clients' runtime environment (coded in Macromedia Director and Flash technology) that interprets the declarations and provides to instance the right components taken from a library (from one hand it instances the right 3D models in the world, on the other it sets up the collaboration's rule that will govern the shared experience).

5. Design considerations

One of the essential goals of CVE is to provide the capability of combining multi-participants and the information that they access and manipulate in a single place. We focus our designing considerations and research work on the following components of a CVE system:

1. Virtual world representation
2. Avatars
3. Shared objects

4. Virtual actions
5. Access control and behaviors
6. Collaborative metaphors
7. Network communications

To implement an effective CVE system, we have to take care of the previous components, each playing a basic and important role in the overall system, then in our declarative approach we have to formally describe all these issues.

5.1 Virtual world representation

First of all, a CVE system has to provide a shared environment for users to cooperate with each other that we call environmental container. Then we had to design a XML based language basically to describe the 3D scene. This language uses the scene graph paradigm: a hierarchical decomposition of the renderable components in a scene.

The WT04 scene graph is organized as a sequence of 3D environments called *parts* in which users can navigate moving form one to another simply colliding to special interactive objects, often in form of ports or gate, working as teleports thus causing the unloading of the current part rendered by the engine and the loading the next one.

5.2 Avatars

Avatars are graphical embodiments representing the participants in the collaborative virtual experience. WT04 schema describes avatars using a particular node structure describing all position and skinning properties.

5.3 Shared objects

WebTalk04 architecture make use of two different kind of objects: from one hand, we use object (typically embedded in the *part file*) that are static geometrical file used to populate the scene graph; on the other hand we use Shared Objects that are deeply dynamic objects that can interact with the other components in the collaborative session. Shared Objects are loaded and instanced at start time. WebTalk04 architecture can do mostly everything with these kind of objects because we can apply all the geometrical and textures transformations. We can configure two different Shared Objects Properties:

1. *Static Properties*
   In this XML section we set up the object's geometrical configuration, such us the world position (X, Y, Z), the rotation (rotX, rotY, rotZ) etc.
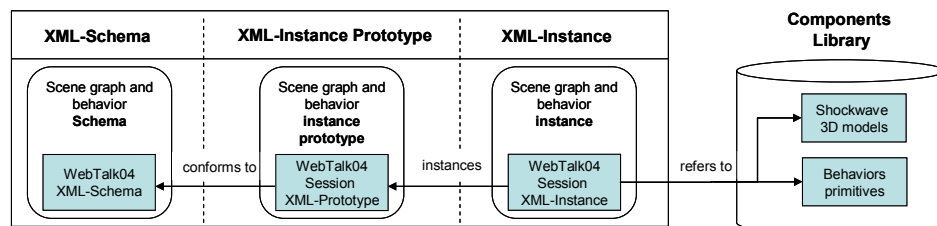2. *Dynamic Properties*

Figure 2: WebTalk04 declarative system levels

The dynamic properties involves with the interplay that the specific shared object can have. Each Shared Object, in fact, interact with other objects in the world. Configuring this XML node we describe how the shared object reacts to external solicitations. In few words, we are able to fine set up all the behaviors that the shared objects can have dynamically.

As seen, our modular approach based on the hierarchical representation of entities (i.e. user and objects) makes simple the process of populating the scene graph with shared objects and to assign the right behavior in a particular collaborative context.

5.4 Virtual actions

In WT04 architecture, all the collaboration issues are configurable by a set of action that take effect on a specific object or set of objects, or generically on the collaborative shared environment. From a WebTalk04 point of view, *actions* are single components that are invoked in a particular instant by someone (typically an avatar) or something (typically an object). So actions are modules that we can configure in order to obtain the dynamic reaction we want.

Below there are some important action modules that our architecture supports:

- *DragObject*: Allows to drag the shared object in the 3D world.
- *ChangeColor* or *ChangeTexture*: Changes the color or the texture of the specified shared object.
- *Showtooltip*: Shows a label near the object in the 3D world.
- *StarTrek*: Moves the avatar in a specific *part*.
- *goToUrl*: Opens a new browser window on a specific page.
- *StartAnimation*: This action is used to starts shared object embedded animation.

5.5 Access control and behaviors

In WebTalk04 the collaboration issues between user-to-user or user-to-object is intrinsically correlated to interactions' primitives above mentioned as *virtual actions* supported by the environment as well as its capability to control and drive collaboration in a specific and well defined direction.

In this sense issues as object ownership and resource management have a great importance on the whole architecture.

As [PM01] states, access control is a familiar concept in such field as operating systems and Computer Supported Collaborative Workgroups (CSCW). The term '*access model*' refers to a set of mechanisms used by a system to determine the operations that may be carried out on a given object by a given user.

In a CVE we need a set of rules or access patterns which determine whether a user, group of users, or user playing a particular role, can perform a specified action.

In our model, interaction is achieved and controlled through a mechanism based on three main concepts using the paradigm of *Event Conditioned Actions*:

- **Entity**: a general resource within the system. An entity may be a shared object with a graphical representation, or an abstract concept with no visible representation as, for example, a server side remote shared object mapping a certain property of the system. Such entities provide a set properties representing in some way the inner state of the entity and an amount of listener which the entity is registered to that allow to react events and perform one or more actions. In this view users and their own avatars can be considered a particular kind of entity.
- **Event**: is a trigger that can be raised by some users' interaction or by the system itself, notifying that some thing has happened in the shared environment.
- **Action**: see par. 5.4 Virtual actions.

According to the classical view the Event Conditioned Actions' rules are based on the following form:

on *event*
if *conditions*
do *actions*

Moreover in our declarative model the central concepts around which all the scene graph is described are the *objects* and *users*, or, in more general worlds, *entities*.

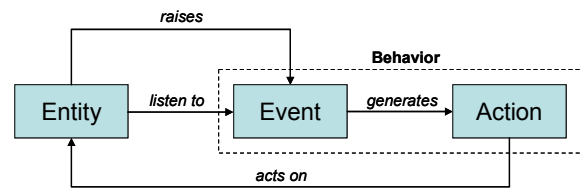Then we derived the following entity-oriented interaction control model:

Figure 3: Interaction control model

In this model the main component is entity that can be thought as an abstraction of both users, as active actors of the system allowed to raise events, and objects, as passive components of the system which listen to events propagated within the virtual environment and react to them performing one or more actions.

Action target can be indifferently the same entity which caused in some way the event, or another different one.

The group made of Event and Action take the name of *Behavior* as states the manner in which some entity evolve the system state when perturbed.

5.6 Collaborative metaphors

Collaborative Metaphors [BP01] are set of rules to support interaction and collaboration between users who want to explore complex content and information together. The rules determines how the "collaborative community" can be created and managed, how every member of the community can operate on his/her own or can cooperate with other members. Different types of situation, tasks and users roles determine different behaviors and therefore need different metaphors. WebTalk04 architecture offers a huge support to collaborative metaphors. We can set up in the XML world configuration file a set of specific behaviors that can be applied to a specific metaphor. So we can select what behavior is applied to a specific shared object in a particular moment. When the specific

collaborative metaphor is activated, the corresponding set of interaction rules take effect, and every object uses the right behavior to interact with the "shared world".

As seen above, thanks to the Collaborative Metaphor paradigm, we are able to manage complex interactions between entities in a virtual shared environment, specifying the current set of collaboration's rules governing every aspect of the virtual world.

Obviously, defining a huge number of metaphors and melting them together, we can govern the collaborative environment recreating the exact collaborative situation we wanted.

5.7 Network communications

This node of the WT04 XML Schema allows us to fine configure the network access. We use a simple declarative approach that permit to easily decide which communication server we have to use to share the experience state, and a set of properties that characterize the connection between the specific client and the communication server.

6. Conclusion and future work

The architecture we present has been deployed in 5 months for two applications, Learning@Europe (involving 48 schools in 6 different countries and more than 900 students) and Stori@Lombardia (involving 36 schools in Northern Italy and more than 700 students). The two applications had different settings (world and objects) and similar (not equal) activities. Overall we had to implement and run 84 collaborative sessions (48 and 36, respectively), with 12 students and 2 staff members acting as users for each session.

The proposed declarative approach provides a formal description both of the virtual environment scene graph and of the interaction settings. Thus it ensures a level of abstraction close enough to the way the session designer is used to think. In fact the description is centered on the same concepts used by the designer to mentally represents the whole collaborative situation, such as users, objects, behaviors, collaborative metaphors. This allows even users without any programming skills, such as content managers and storyboard designers, to set up their own virtual environment as a sequence of different parts, each populated by avatars and objects, interacting with each other in the ways wanted by the designer.

Benefits of this approach are also evident in the progressive reduction of the session setup time, as time spent to customize a new session decreases with the increasing number of available components (XML blocks, behavior primitives, geometry models).

7. Acknowledgments

8. Bibliography

[XSD01]   XML-Schema: www.w3.org/XML/Schema

[DHP03]   DI BLAS N., HAZAN S., PAOLINI P.: The SEE Experience: Edutainment in 3D Virtual Worlds. In Proceedings Museums and the Web '03, Charlotte (USA) 2004.

[D01]   DACHSELT, R.: CONTIGRA - Towards a Document-based Approach to 3D Components. In Workshop proceedings "Structured Design of Virtual Environments and 3DComponents" of the ACM Web3D 2001 Symposium, Paderborn, 2001.

[DA01]   DACHSELT, R.: CONTIGRA: A High-Level XML-Based Approach to Interactive 3D Components. In SIGGRAPH 2001 Conference Abstracts and Applications, Los Angeles, August 2001, 163.

[WZ98]   WATSEN K. , ZYDA M.: Bamboo – A Portable System for Dynamically Extensible, Real-time, Networked, Virtual Environments. In Proceedings of the IEEE Virtual Reality Annual International Symposium (VRAIS'98), Atlanta,Georgia,1998, pp. 252-259.

[PM01]   PETTIFER S., MARSH J. : A collaborative access model for shared virtual environments. In Proceedings of IEEE WETICE 01, pages 257-272. IEEE Computer Society, June 2001.

[FS98]   FRÉCON E., STENIUS M. : DIVE: A scalable network architecture for distributed virtual environments. Distributed System Engineering Journal,5, pp. 91-100, 1998.

[GPS00]   GREENHALGH C., PURBTICK J., SNOWDOWN D.: Inside MASSIVE-3: Flexible support for data consistency and world structuring. In Proceedings of Collaborative Virtual Environments 2000, pp. 119-127, San Francisco, September 2000.

[BP01]   BARBIERI T.,PAOLINI P.: Cooperation Metaphors for Virtual Museums . In Proceedings Museums & Web 2001, Seattle, USA, March 2001.

[X3DS]   X3D Specification: http://www.web3d.org/x3d/ specifications/

[DR03]   DACHSELT R., RUKZIO E.: BEHAVIOR3D : An XML-based framework for 3D graphics behavior. In Proceedings of the 8th International Symposium on Web3D Technologies. WEB3D Consortium, 2003.