VRWork: Toolkit for Development of Web based Virtual Environments Supporting Multi Users

Junhyun Tak
Dept of Computer Science & Engineering, INHA University
Inchon, Korea
+82-32-868-9623
tak2023@hanmail.net

Seihoon Lee
Dept of Computer Science, INHA Technical College
Inchon, Korea
+82-32-860-7441
seihoon@true.inhatc.ac.kr

Changjong Wang
Dept of Computer Science & Engineering, INHA University
Inchon, Korea
+82-32-870-2332
cjwang@inha.ac.kr

ABSTRACT

Networked Virtual Environments allows multiple users to interact in real-time even though those users may be located around the world. The NVE developer must manage consistent distributed information, guarantee real-time interactivity with limited network bandwidth, processing and rendering resources. Moreover the majority of the functions for NVE applications are common. Therefore the NVE developer should tolerate the problem of duplication of development efforts for same functions.

So, we design a toolkit for supporting the development of NVE on Internet and name it VRWork. VRWork provides multi-user supporting functions and additional functions, which is needed for human oriented collaborative applications.

The multi-user supporting in VRWork lay emphasis on the scalability and availability of NVE applications. For this VRWork use multi-level event filtering. VRWork's additional functions for human oriented collaborative applications are as follows: Efficient access control for shared object, Easy sharing of legacy 2D application software in NVEs, and Dynamic notification of events.

Keywords

Networked virtual environments, toolkit, access control, application sharing, dynamic event notification

1. INTRODUCTION

Networked Virtual Environments (NVEs) allows multiple users to interact in real-time even though those users may be located around the world. These environments usually aim for an immersive experience by incorporating a sense of realism such as realistic 3D graphics and stereo sound, and so forth. In recent years, open standards for the delivery of NVEs over the Web are beginning to emerge [1,2]. Increasingly used for team training, collaborative design and engineering, and multi-player games, NVEs envisioned future commercial applications include virtual shopping malls and showrooms, on-line trade shows and conferences, remote customer support, and distance learning.

In many respects, NVEs form the foundation for a new generation of standard applications [3].

The NVE developer must manage consistent distributed information, guarantee real-time interactivity with limited network bandwidth, processing and rendering resources. Moreover the majority of the functions for NVE applications are common. So the NVE developer should tolerate the problem of duplication of development efforts for same functions. So, current focus in the NVE arena is on the development of toolkits that simplify the development of NVEs and provide a standard framework for NVE application development [3,4].

We design a toolkit for the development of NVEs on Internet, and call this VRWork. VRWork provides multi-user supporting functions and additional functions, which is needed for human oriented collaborative applications.

The multi-user supporting in VRWork lay emphasis on the scalability and availability of NVE applications. VRWork provides multi-level event filtering:

- Occlusion based filtering: restrict the event notification range into the zone
- Distance based filtering: differentiate event notification rate according to the observers DOI (degree of interest) on the observed
- Filtering or amplification for availability: filtering (or amplification) by dynamic group policy, although unlike the real world

VRWork's additional functions for human oriented collaborative applications are as follows:

- Efficient access control for shared object
- Easy sharing of legacy 2D application software in NVEs
- Dynamic notification of events

2. RELATED WORKS

Event filtering is the typical way that NVE developers use for the scalability and efficient awareness. Event filtering is based on spatial model that decides the event propagation region in the whole world.

There were several spatial models. NVE applications that on wide area terrain like as the DIS [5] based systems are use fixed-size static area. NVE applications that model the inside of buildings like as shopping malls or business offices are use variable-size static area [6]. Another approach is the way that uses the intersecting volume to model the interaction among users. This notion of dynamic spatial area is evolved form COMIC. This spatial model known as aura decide the level of awareness according to the focus of observer and the nimbus of observed [7].

3. SPATIAL MODEL for VRWork

In general, a logically single large virtual space on Internet is consisted of several virtual worlds each of that forms a downloadable VRML file. We define that the virtual space, VS is a power set of the individual VRML files.

[D1] Virtual Space

SW = {W|W is a world, which is a single VRML file}

VS = p(SW) , n(SW) >= 1, n(VS) >= 1

Each virtual world W is divided into several partial regions. We call this dividing unit zone.

[D2] Virtual World

W = {Z|Z is a zone}, n(W) >= 1

Zones contain SE and DO. SE is a set of static entities and DO is a set of dynamic objects.

[D3] Zone

Z = {SE, DO}

where, SE is a set of static entities

DO is a set of dynamic objects

Static entity is a VRML node, which construct the scene graph. Dynamic object contains a set of participants and a set of shared objects.

[D4] Dynamic Object

DO = {P, SO}

A participant p, element of P, has following attributes.

[D5] participant

p(id, position, v_direction, m_direction, roles, action)

where, id is a identification name

position is a presence position where he is

v_direction is a viewing direction at current position

m_direction is a moving direction at current position

roles are current stations which he has

action is a current action what he doing

Shared objects, which react to the participant's action, contain a set of general shared object and a set of boundary objects.

[D6] Shared Object

SO = {GO, BO}, n(BO) >= 1

GO = {go | go is general shared object}

BO = {bo | bo is a boundary object }

[D7] SO's attributes

go(id, position, proles, behaviors)

bo(id, position, proles, behavior)

where, id is a identification name

position is a presence position where it is

proles are current stations those can trigger so's behavior

behaviors are possible behaviors which it can do

Roles in [D5] and proles in [D7] are used to provide role based access control to the shared world. Proles of a shared object mean the permitted roles, which are allowed to trigger a behavior of the shared object. So, only the participants whose role is same as the shared object's prole can access the shared object.

Boundary object is a shared object that has special behaviors such as allowing or denying to the request of enter a spatial area. Boundary objects provide access control to spatial area. A real like virtual world would contains open areas such as corridor, park, etc. At the same time it may contains secret areas that should be protected from illegal access.

The primary role of NVE's spatial model is providing efficient awareness management. Our spatial model is based on zones that are used in occlusion based spatial division like as the Spline's locale. This occlusion based filtering is very appropriate to the case of human oriented collaborative applications in which a virtual world is consisted of rooms, which are divided by walls. But, when the zone is wide there can be a number of notifications of useless (or less useful) events. So distance based filtering which used in the MASSIVE's aura model is needed. In aura model, the focus of observer and the nimbus of observed determine the level of awareness. This determination goes with n(n-1)/2 times decision of distance with 3D points and angle difference from the observer's viewing orientation to the observed's position. This is a graceful way for event filtering which simulate the case of real world. But computational overhead grows a geometrically when the number of users is grow. For human oriented collaborative application, not accurate simulation, on Internet, which has low network bandwidth and low computing resources, it is appropriate to reduce the computational overhead even if it decrease the accuracy a little. For this, we use hexagonal cell in calculating the DOI (degree of interest) value and use it to handle the event transfer rate. DOI value is calculated as follow:

- A zone is mapped with several hexagonal cells.
- Decide distance and angle difference among users by comparison of cell ID rather than calculation of 3D floating point vectors.

For this, we define position attribute in [D5] and [D7] as [D8].

[D8] position of participant and shared object

position(Z, cellID, pos_coord)

where, Z is a zone in which the participant is locate

cellID is a name of hexagonal cell on which the participant is locate

pos_coord is a 3D coordinate value of the participant

4. VRWork BASED ON SPATIAL MODEL

VRWork is a toolkit for Internet NVE applications development. Fig.1 is the overall software architecture of VRWork.

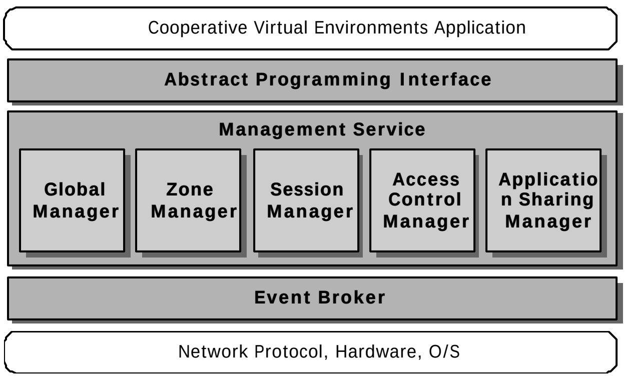

Figure 1. The architecture of VRWork Toolkit

VRWork's 3D virtual world is consists of a set of zones. Zone is a physical area unit, which decided by the division of the world while designing the virtual world.

Zone Manager (ZM) supports the interactions among the users in the zone. Session Manager supports dynamic group activity.

A session is a logical area in which inter-users interactions are. Group policy rather than locations of users control a session. For example, users in the same zone can be in separate sessions and users in different zone can be in a same session.

Global Manager (GM) manages the entrance and leaving of users and coordinates several Zone Managers. When a user is to enter a zone, GM sends the user information to the ZM and records the users' movement path.

Access Control Manager (ACM) manages the access right to the shared objects and zones based on the role of users. Virtual world can consists of open areas such as park, shopping mall, etc. At the same time it can have secret areas such as office, control center, etc. VRWork provides access control on spatial area.

Application Sharing Manager (ASM) enables the users share the legacy application software such as word processors, painting software, CAD software, etc. This can lead more available and efficient collaboration of users.

Event Broker supports the dynamic publishing and subscribing of events in order to provide the transparency between publishers and subscribers. Further, VRWork use self-describing XML representation for the events. Using XML, it is possible to filter the events by content. We developed several NVE applications on VRWork. The architecture of NVEs on VRWork is client/server architecture as fig.2.

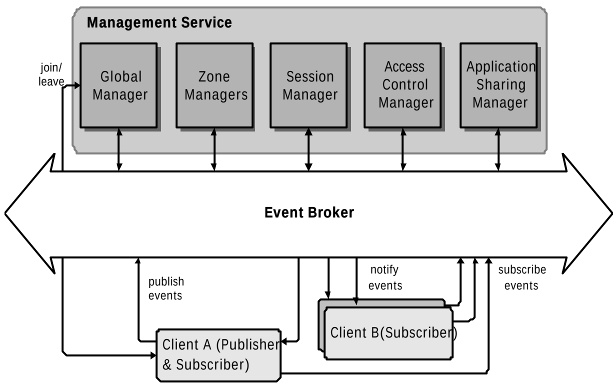

Figure 2. The architecture of NVEs on VRWork

4.1 Event Broker

The event-processing scheme in legacy VR environment is dependent to the implementing system, and the definition of events is dependent to applications of system. Therefore, we propose event broker(EB) as a generic event notification system to eliminate these dependencies.

The proposed EB designs Event Notifier Model [8, 9] to perform dynamic event filtering to reduce the amount of events. Additionally, we describe events based on XML, so it increases the efficiency of interaction with external services and makes searching convenient through event logs management.

Figure 3 shows the architecture of event broker.

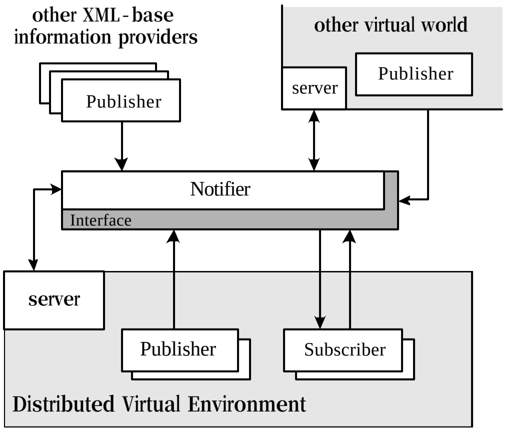

Figure 3. The architecture of event broker

The designed notifier has following features.

- It simplifies complex event propagation routes by providing transparency between publisher and subscriber.
- It makes an easy to insert or delete the receiving events of shared objects or specific information dynamically regardless of system when it uses different interaction ranges according to avatar. The definition of method about these interaction ranges is dependent on the management of VR environments, but it is independent of notification system.
- It provides dynamic filtering facilities. Subscribers can personalize events propagated through the registration of filter about specific events to notifier. Events described in XML are also possible to filtering of event contents using searching and conversion of XML tags because XML based events are hierarchical and self-describing.

It can interact with legacy applications dependent on domain. And it enables to convert XML formed events with the registration of applied application interfaces to notifier.

4.1.1 Event Description

The basic event types for accepting events in VR environment are defined as follows.

We use XML to describe events because it is possible content based filtering and is convenient to build database of log information. In addition, these information are useful to retrieve events generated, and write working information or behavior patterns of participants in VR environment.

4.1.1.1 Action Event

Action events are generated when the object status is changed, or when active behaviors are happened. These events are related with persistency maintenance of VR environment. They are defined as follows according to the degree of synchronization.

- Independent interaction event: This event is generated independently regardless of participants such as movement of custom objects or changes of avatar location. Recent events can be substituted for previous events because it does not effect other participants immediately. However, this should be reflected upon other clients for consistency.
- Shared interaction event: This kind of event effects other clients immediately, so it has to be reflected firstly of all than independent events. For example, events generated in cooperative work should be reflected on participants firstly.

4.1.1.2 Information event

This event has no connection with the synchronization of virtual environment. It is not notified for consistency immediately, but should be transmitted. This event is generated when mail type data is propagated to internal/external virtual environment, or when the previous log data is requested from system. Figure 4 is a DTD for data structure of events, and Figure 5 is a sample event that describes the data structure of shared Interaction event using XML Syntax. This is event that avatar named “Ellio” makes a movement the target object of cooperative work, “CoWork”, according to the value of attribute <vec3fx>.

The sub-element of <Content> tag is used to describe the node name and value of VRML, which should be changed by this event. Events using XML syntax can be extended through adding event tag according to each application applied notification system, and enable event transmission between applications for interoperability of result in cooperative work. And building database of log data described in XML can be applied for analysis of the propensity to consume and the shopping route pattern in virtual shopping mall, analysis of study attitude of students in virtual class, the maintenance of a course of working in cooperative work, and so on.

The designed Notifier consists of Converter and Filter. Converter changes events, which are created in external services of other VR environment, to XML. Filter reduces the amount of events by filtering.

```
<?XML version=1.0/>
<!ELEMENT Event (EventType)>
<!ELEMENT EventType
(Independent | Shared | Control | Message)>
<!ELEMENT Independent
(Publisher, Executor?, CellID?, StateContent+, TimeStamp)>
<!ELEMENT Shared
(Publisher, Executor?, CellID?, StateContent+, TimeStamp)>
<!ELEMENT Control (Publisher , ControlContent, TimeStamp >
<!ELEMENT Message (Publisher, MessageContent, TimeStamp)>
<!ELEMENT StateContent ((NodeName, Value)+)>
<!ELEMENT NodeName (#PCDATA)>
<!ELEMENT Value (#PCDATA)>
<!ELEMENT MessageContent (#PCDATA)>
<!ELEMENT Publisher
(ObjectID | AvatarID | ManagerName, (ZoneID, CellID)?)>
<!ELEMENT Executor(ObjectID | AvatarID)>
<!ELEMENT ObjectID (#PCDATA)>
<!ELEMENT AvatarID (#PCDATA)>
<!ELEMENT ZoneID (#PCDATA)>
<!ELEMENT CellID (#PCDATA)>
<!ELEMENT ManagerName (#PCDATA)>
<!ELEMENT TimeStamp (#PCDATA)>
```

Figure 4. DTD for event data structure

```
<?xml version="1.0" encoding="UTF-8"?>
  <Event>
     <EventType> Shared</ EventType>
     <Publisher>
       <ObjectID> Globe </ObjectID>
       <ZoneID> ClassRoom </ZoneID>
       <CellID> A35 </CellID>
     </ Publisher>
     <Executor>
       <AvatarID> Tak </AvatarID>
     </Executor>
     <StateContent>
       <NodeName> Translate </NodeName>
       <Value> 3.4 5.6 4.6 </Value>
     </StateContent>
     <TimeStamp> 05:12:04 </TimeStamp>
  </Event>
```

Figure 5. Sample shared interaction event

If Notifier receives events from publisher, it doesn't transmit an event to subscribers at once. Events are transmitted to subscribers after these events pass through Converter and Filter. Figure 5 shows the operation of XML Converter, the registration of filtering and the applying point of Converter and Filter during processing of event notification.

If events are created in other VR environment, Converter must change events to XML-form to interact with the system of other VR environment. Filter performs events filtering though examining the events that are registered by subscribers, and then transmits the filtered events to subscribers. At this time, Notifier knows the propagation range of event through accessing to VR environment server that events were generated in advance.

4.2 Collaborative Multi User Supporting Modules

4.2.1 Global Manager

Global manager consisted of zone controller, user manager, and position tracer. Zone controller coordinates several zone managers. User manager manages information about all users. This information contains log information. Position tracer records the users' movement path. User's movement path is important information that can be used to analysis the users behavior pattern.

4.2.2 Zone Manager

Zone manager consisted of state manager, participant manager, and request handler. State manager manages current state information of the zone in order to provide consistency to the participants. Participant manager manages the information about current participants in the zone. Request handler transfer request events from participants to event broker.

4.2.3 Session Manager

Session Manager consisted of group factory and group manager. Group factory manages the life cycle of a group. Group factory creates a group when users request it and manages group information such as group id, subject of the group, participants ids who are participate in the group.

Group manager manages the group context. Group context contains information about current participants, current chairman of the group, current participant who has the right to speak, and shared application.

4.2.4 Access Control Manager

Access control manager decides that whether permit or deny the request from participants. Participants' requests are contain the access to the shared objects and the request to change the system information such as proles of object. Access control manager uses traditional locking mechanism in order to provide concurrency control. Figure 6 shows the architecture of access control manager.

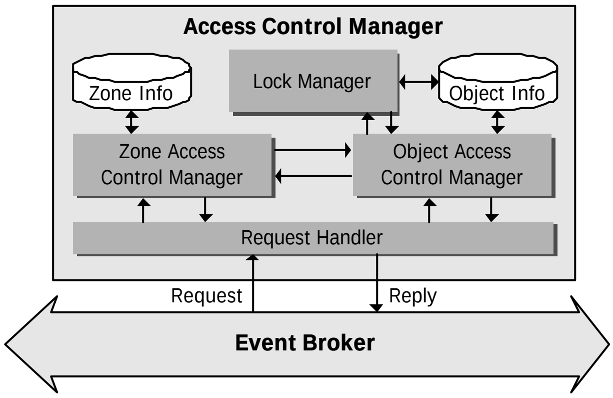

Figure 6. Role based access control manager

4.2.5 Application Sharing Manager

Application sharing manager applies the change event, which occurs on the application view from user, to the shared application. And it sends the applied view other users.

Figure 7 shows the architecture of application sharing manager. Application shell starts the application software at server when group manager sends the application launch request. And it applies the client's event on his shared view to the application software on the server. Scheduler translates the event on the shared view to the window event and schedules the participants' events in order to protect the atomic action such as mouse clicking or dragging.

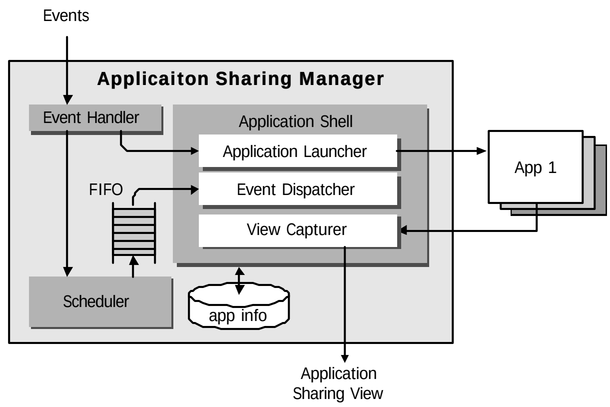

Figure 7. Application sharing manager

5. VIRTUAL SCHOOL

We developed VIRTUAL SCHOOL, a collaborative learning NVE system on Internet [10] using VRWork toolkit. Figure 8 shows a learning scene in which several students are participating.

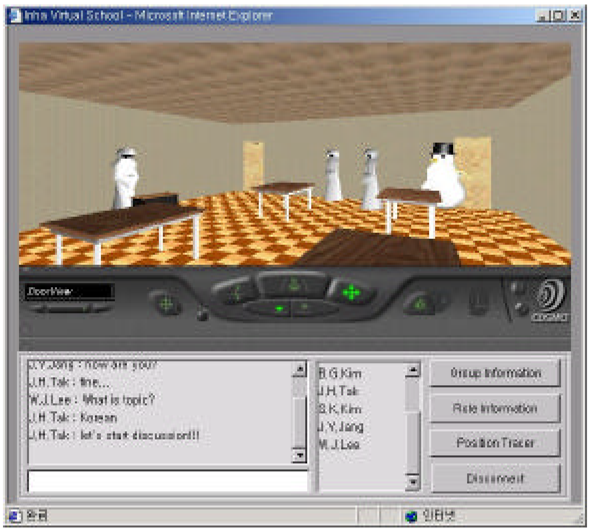

Figure 8. Collaborative learning scene in VIRTUAL SCHOOL

In VIRTUAL SCHOOL, students can realistic group discussion and teachers can coordinate the group policy at run time. So various learning models are easily applicable. System administrators can multi level event filtering according to the scale of the NVE application. In the case of VIRTUAL SCHOOL, which consisted of one corridor, two classrooms and one science lab, and 60 students and 3 teachers are participating in 30-minute learning, the amount of event transfer was reduced at 45%.

6. CONCLUSION

We designed and implemented VRWork-a toolkit for development of web based NVE applications supporting multi users. For the human oriented collaborative NVE applications on Internet, VRWork provides additional functionalities. VRWork is based on extended spatial model that support multi level filtering and role based access control and using event notifier model that provides dynamic event notification.

7. REFERENCES

[1] VRML '97 spec: http://www.vrml.org/technicalinfo/specifications/

[2] VRML Living Worlds Working Group Web Site: http://www.vrml.org/WorkingGroups/living-worlds/

[3] Bernie Roehle, "Channeling The Data Flood," IEEE SPECTRUM (1997).

[4] Sandeep Singhal and Michael Zyda, *Networked Virtual Environments*, Addison-Wesley, New York (1999).

[5] IEEE standard for information technology -protocols for distributed applications: entity information and interaction, IEEE standard 1278-1993. IEEE Computer Society, 1993

[6] J. Barrus, C. Waters, and D.B. Anderson, "Locales and Beacons: Efficient and precise support for large multi-user virtual environment," In Proceedings of the 1996 Virtual Reality Annual International Symposium, 204-213 (1996).

[7] C. Greenhalgh, and S. Benford, "Boundaries, Awareness and Interaction in Collaborative Virtual Environments," In Proceedings of 6th Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise(WETICE), 193-198, IEEE Computer Society (1997)

[8] Gupta et al., "Event Notifier: A Pattern for Event Notification," in Java Report Magazine, (1998).

[9] Gamma et al., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley (1997).

[10] YT Baek, JH Tak, SH Lee, CJ Wang, "Multi-users Collaborative Learning System Using Virtual Technology on Web", SchoolNet 2000, Busan, Korea (2000).