A Hierarchical Architecture for Improving Scalability and Consistency in CVE Systems

Journal:	*International Journal of Parallel, Emergent and Distributed Systems*
Manuscript ID:	GPAA-2009-0069.R2
Manuscript Type:	Original paper
Date Submitted by the Author:	14-Feb-2010
Complete List of Authors:	hu, xiaomei; Shanghai University Liu, Lilan; Shanghai University Yu, Tao; Shanghai University
Keywords:	collaborative virtual environment (CVE), dynamic clustering, partition, tailoring tree model, time-bounding-box

A Hierarchical Architecture for Improving Scalability and Consistency in CVE Systems

Xiaomei Hu Lilan Liu Tao Yu

Shanghai Key Laboratory of Mechanical Automation and Robotics, Shanghai University, Shanghai, China

Mailbox 232,Yanchang Road 149, Shanghai University, Shanghai, China,200072

(Received 28 July 2009; final version received)

Existing client/server architecture with multiple-server has the capacity constraint of users due to the fixed number of servers. A hierarchical architecture based on the tailoring tree model for improving scalability and consistency in the Collaborative Virtual Environment (CVE) is described. In our architecture, the main server, region servers, and clients maintain the state of CVE together. In order to improve the scalability and consistency, computing resources and communication resources should be effectively assigned. Two algorithms are proposed to solve the resource assignment problem in the hierarchical architecture: a dynamic clustering algorithm partitions the virtual environment according to the current load, which can effectively assign computing resources; a time-bounding-box-based filtering scheme is used to reduce the traffic, which can effectively assign communication resources. Experimental results show that the hierarchical architecture with our algorithms has a reasonable assignment of computing resources and communication resources at the same time, and indeed improves the scalability and consistency.

Keywords: collaborative virtual environment (CVE); dynamic clustering; partition; tailoring tree model; time-bounding-box

Introduction

Collaborative Virtual Environment (CVE) is the sharing virtual space including virtual entities and resources maintained by a group of computers which can support effective communication between the users to achieve better coordination tasks. It has been widely applied in distance education, online multi-user game, collaborative design, joint military training and other fields [1].

In a CVE system, users share the common virtual space, and each user is represented by an entity. When a user moves around or interacts with other entities, the CVE system not only consumes the computing resource to updates its own state, but consumes the communication resource to distribute the update of state to other users in time. With the expansion of the scale of applications and the increasing number of users, network and computing resources in the system are far from adequate to meet the requirement of growth of the scale. A scalable collaborative virtual environment system aims at keeping the operating efficiency and maintenance

of good consistency while the software structure do not need to do major modifications in views of different clients in support of a large-scale users participating in the virtual environment.

Peer-to-Peer architecture and client/server architecture with single server are used to design a distributed virtual environment [2,3,7], however their limitations are also obvious. To address the issue of scalability, the collaborative virtual environment usually adopts the multiple-server architecture so that more users can participate in the system. In this architecture, the management of the virtual environment relies on several interconnected servers. The virtual environment is partitioned into regions and each region is designated to a server. When a new user logs in the CVE system, it will select to connect with one of the servers in the system according to its position on the virtual environment. If a user modifies the state of system, it sends the update to its server, which in turn propagates this message to other servers or clients to maintain the consistency of the system. So there are two kinds of communication types in the CVE system: inter-server message and intra-server message. Inter-server message is transferred among the servers, and intra-server message is transferred between server and client, shown as Figure 1.

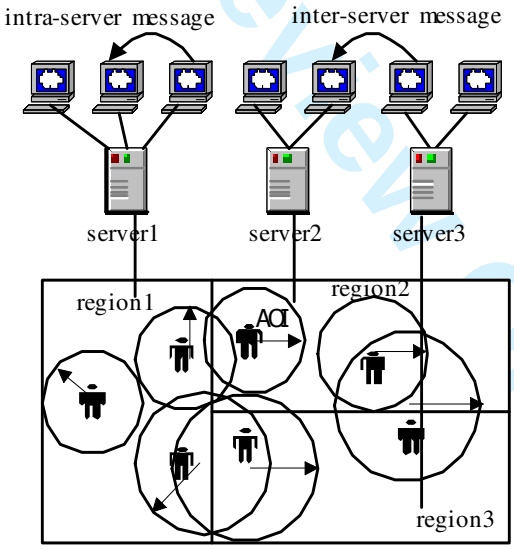

Figure 1. Two kinds of messages in a CVE system

From Figure1, it can be seen that when a new user enters into the CVE system, the server side will increase the communication workload with the clients and other servers. Though this architecture is popular during the last few years [4,5,6], the constraint of the number of servers will affect the scalability of system. With the

increasing number of participants, network and computing resources can still be run out.

The key issue of designing a large-scale CVE system is the conflict of consistency and scalability [8]. Consistency refers to the consistent state of various entities and users in the CVE system. Updates should be transmitted over the network so that other users can recognize the correct states, which will consume the communication resource. Scalability refers to the effective consistency control even if more users move into the virtual environment. However if a large number of users enter into the virtual environment simultaneously, consistency maintenance may saturate network bandwidth. In order to solve the conflict of scalability and consistency, a hierarchical architecture for improving scalability and consistency is given. Tailoring tree model is used to overcome the capacity limitation of system by dynamically deploying the region server. When the existing region servers can not afford the load of system, new region servers can be configured to share the excessive workload. Since the number of servers in the system is not fixed, the dynamic clustering algorithm is proposed to partition the virtual environment in order to make none of region servers overloaded and minimize the traffic among servers. Moreover, a time-bounding-box-based filtering scheme on the client side is used to reduce the traffic between servers and clients.

The rest of the paper is organized as follows: Section 2 discusses the related work in the field of the CVE communication architecture. Section 3 brings out a hierarchical architecture for improving scalability and consistency in collaborative virtual environment systems. It also details the hierarchical structures of three kinds of hosts in this architecture --the main server, the region server and the client. Section 4 proposes a dynamic clustering algorithm to partition the virtual environment. Section 5 describes the time-bounding-box-based filtering scheme which is applied in the client side. Section 6 presents some experiments and results about the effectiveness of the dynamic clustering partitioning and time-bounding-box-based filtering algorithm. Finally, section 7 summarizes the paper and makes some conclusions.

Related Work

There are several architectures for implementing a CVE system. The most basic ones are peer-to-peer (P2P) architecture, client/server architecture with single server, and client/server architecture with multiple-server [2].

In the P2P communication architecture, each client sends updates directly to other clients, shown in Figure 2. It has the advantages of low communication latency and fault-tolerance capability, for a single client's fault will not cause the whole system's crash. But it also has the disadvantage of communication complexity. Each client has to maintain many logical connections with others. In addition, this architecture usually adopts the filtering algorithm to reduce the consumption of network, which causes the inconsistency of the system.

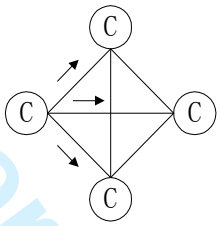

Figure 2. Peer-to-peer architecture

In the client/server architecture with single server, each client only sends updates to the server, and the server transmits them to other clients, shown in Figure 3. It reduces the communication complexity, for each client only establishes a logical connection. In addition, the server can do some administration tasks, such as message filtering, collision detection. But it also has the drawback that the server may become the system's bottleneck.

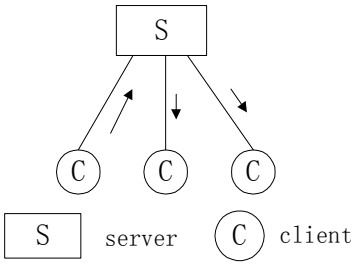

Figure 3. Client/server architecture with single server

In client/server architecture with multiple-server, the management of the virtual environment relies on several interconnected servers and each server handles a partition of the virtual world, shown in Figure 4.

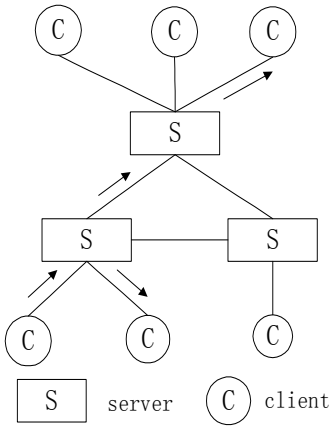

Figure 4. Client/server architecture with multiple-server

From Figure 4, a server must execute the following three functions to keep the consistency in the virtual environment:

1) Receiving the update messages from the clients.
2) Updating the whole virtual environment.
3) Transmitting updates of the virtual environment to other clients and servers.

And a client must execute the following four functions to keep the consistency in the virtual environment:

1) Receiving the user's input as the update message.
2) Transmitting the update message to the server.
3) Receiving the update messages from the server.

This architecture solves the system's bottleneck problem, which makes it suitable for designing a scalable CVE system. However because the virtual environment is divided into several regions, the unreasonable partitioning method will produce a lot of inter-server communication, which will take up overfull network resources. Partitioning problem is one of the key issues in the design of a scalable CVE system based on this architecture. An excellent partitioning method should reasonably assign the computation resources and communication resources among the servers. There are some methods proposed as follows:

Lui and Chan proposed a parallel incremental graph partitioning algorithm to divide the load among distributed servers [9]. Participants in the DVE will be migrated according to the result of the graph partitioning algorithm. In order to optimize the partitioning quality, the partitioning algorithm is periodically executed for adapting the partition to the current state of the DVE system, so the algorithm may

take much execution time for a large-scale DVE system, and it is not really suitable for real-time CVE applications.

Pedro Morillo et. al. proposed several heuristic search methods for finding the best assignment of clients to servers and presented a comparison study of several heuristics for solving the partitioning problem in CVE systems [10]. These methods improve the performance of the partitioning algorithm by providing better partitioning quality and requiring shorter execution time than LOT method for non-uniform distribution of users.

Kyungmin Lee et. al. proposed a scalable dynamic load distribution scheme for multi-server distribution virtual environment systems, where users are highly skewed rather than uniformly distributed over a virtual environment [11].

In [12], Pedro Morillo et. al. proposed a characterization study of a distributed virtual environment system. A distributed virtual environment system shows a non-linear behaviour with the increase of the number of users in the system. Average system response remains practically invariant with the increase of the number of users in the system until the system reaches a saturation point. In addition, the results show that load balancing mainly has an effect on system throughput, while the amount of inter-server messages mainly has an effect on system latency. Once the limit is reached, average system response greatly increases when new users are added to the system.

Therefore, in order to complete a large-scale cost-effective and high performance CVE system based on multiple-server architecture, new servers should be deployed according to the current load of system at any time. The partitioning algorithm should not only balance the workload among the servers in the system in such a way that none of the servers reaches their saturation point, but also repartition the virtual environment when servers join or exit the system. At the same time, the inter-server messages and migrated users should be minimized by the reasonable partition in order to decrease the system latency.

In order to reduce the traffic in the system, there are several methods to filter the user's updates. AOI management and grouping method can be easily used in the server side to filter the redundant messages. Usually, the system divides entities into groups [15, 16]. So the data will just be sent to the groups where interested entities exist, while other entities outside the groups will not receive the data.

Dead Reckon (DR) method can be used in the client side to filter the messages [17]. Each user not only has its own accurate model and DR model, but maintains the DR models of remote users. The user compares the difference of its accurate position and estimated position after it generates the update message. If the difference is greater than the pre-defined threshold, a state update packet will be sent to remote hosts. Otherwise, the estimated position of the entity is used in remote hosts through a local computation instead of transmitting state update packets. These message filtering methods can effectively reduce the traffic in the system, but they do not consider how to maintain the consistency.

Due to the network delay, the copies of entities' positions in the remote hosts always lag their real positions in the local hosts. In addition, the DR method adopts the prediction route to filter messages that leads to the short-term inconsistency of system. Therefore, Mauve proposed a local-lag and timewarp algorithm to guarantee the consistency through compensation scheme [18]. Instead of immediately executing an operation issued by a local user, the operation is delayed for a certain amount of time before it is executed, which is called local-lag. Local-lag can compensate the network delay and reduce the number of short-term inconsistencies. However, the timewarp algorithm increases the complexity of algorithm and consumes a large amount of network bandwidth with the increase of users.

A New Scalable Architecture Based on Tailoring Tree Model

In the collaborative virtual environment, users may log in or log out the system at any time, which makes the load of system uncertain. It may cause the existing servers overloaded when a large number of users log in the system simultaneously. Load balancing method can reduce the load of the overloaded server in some way, but when all the servers in the system are overloaded, the system response time will greatly increase. If a CVE system wants to have a good collaborative performance, the virtual environment should be re-partitioned and new servers should be deployed to share the excessive workload of overloaded servers in order to improve the response time of the system. A scalable communication architecture based on tailoring tree model is shown in Figure 5. In this architecture, there are three kinds of hosts according to their different functions in the CVE system: the main server, the region servers and the clients.

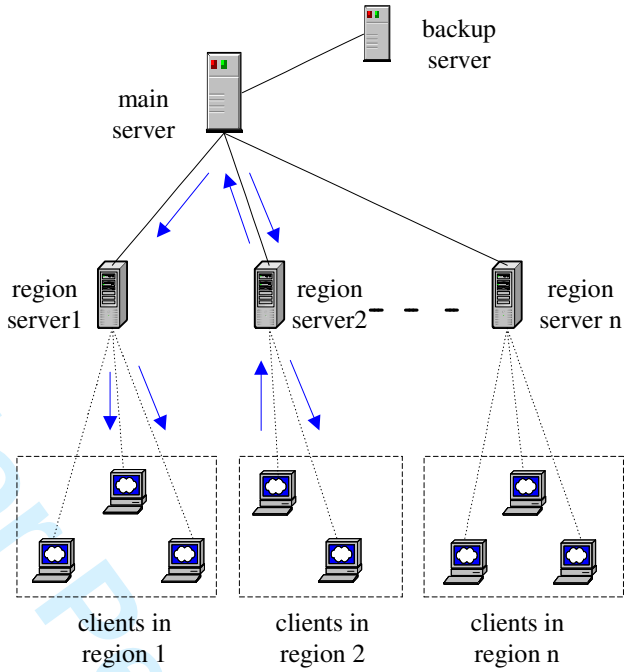

Figure 5. A Scalable Communication Architecture based on Tailoring Tree Model

Due to the uncertain number of the users and the region severs in the system, the main server monitors the current load to judge if the number of region servers is reasonable. When the main server finds the current load close to the saturation point, it starts partitioning the virtual environment by the partitioning algorithm and configuring the region servers by the communication program, some users are transferred and connected to new region servers. New region server receives the instruction from the main server by the communication program and starts the region management program to join in the CVE system, and initializes the partial virtual environment: the main server sends the transferring instructors and addresses to the clients. After the clients connect with the region server, the region server officially manages the partial virtual environment. When the region server is close to overload, load balancing scheme is adopted to transfer some clients to other region servers to make sure that each region server is not overloaded. The client connects with a region server to obtain the information about the virtual environment and exchange users' updates through the region server. The difference between the architecture based on tailoring tree model and the architecture based on multi-server is that the number of servers is not fixed. In our architecture, new region server can join the system and manage a part of virtual environment at any time only if there are backup servers. However, in the architecture based on multi-server, the number of servers is certain,

which assures the maximum load of a CVE system.

From Figure 5, the main server as the root node of tailoring tree is the father node of the backup server and region servers. Clients as the leaf node of tailoring tree can change their father node when the users move into a new region and manage by a new region server. If a region server becomes a leaf node, which means that no client connects with it, it can send a request to exit the system. When the main server receives the request, it calculates the current load of system at first. If the load of system is low, that is, other region servers are less-loaded, the main server will disconnect with the region server. Otherwise, the main server will balance the workload among the region servers, and some load will be transferred to the less-loaded region server. In this way, it can be avoided that the region server joins or exits the system frequently, which consumes overfull resources of the main server.

In the scalable architecture based on tailoring tree model, the function structures of the main server, the region server and the client are detailed as follows:

Hierarchical Structure of the Client

The hierarchical structure of the client is shown in Figure 6. User's application layer updates the local view and sends the update message to the data transmission layer when it receives the user's input through the devices such as mouse, keyboard, data glove and other trackers. Message filtering layer is used to reduce the traffic between servers and clients. View management layer receives 3D model, 2D texture, audio and video from the region server to build in a local view. In addition, it updates entity database and user database when it receives the updates from user's application or data transmission layer. Data transmission layer receives the update message from the user's application layer and sends it to the network layer after data compression, priority assignment and data packaging. Moreover, the data transmission layer receives the message from the network layer, and sends it to the receiver queue for data decompression. Network layer can use special protocol to guarantee their reliable and real-time transmission.

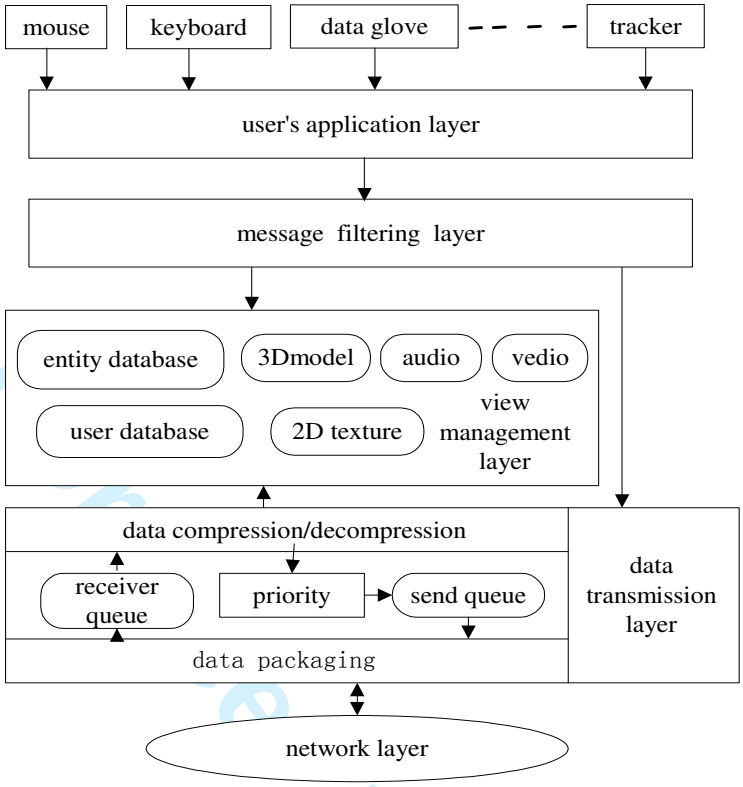

Figure 6. Hierarchical structure of the client

Hierarchical structure of Region server

The hierarchical structure of the region server is shown in Figure 7.

System maintenance layer maintains the logical connections with all the clients. When it receives partitioning information from the main server, it adjusts its control region by updating the entity simulation layer, database maintenance layer and area of interest (AOI) management layer.

Entity simulation layer keeps the movement models of the entities in the region. Also, it makes collision detection among entities and users to avoid the penetration phenomenon. When it receives new partitioning information from the system maintenance layer, it will add or delete some movement models of entities if the new entities join in or leave from the new region.

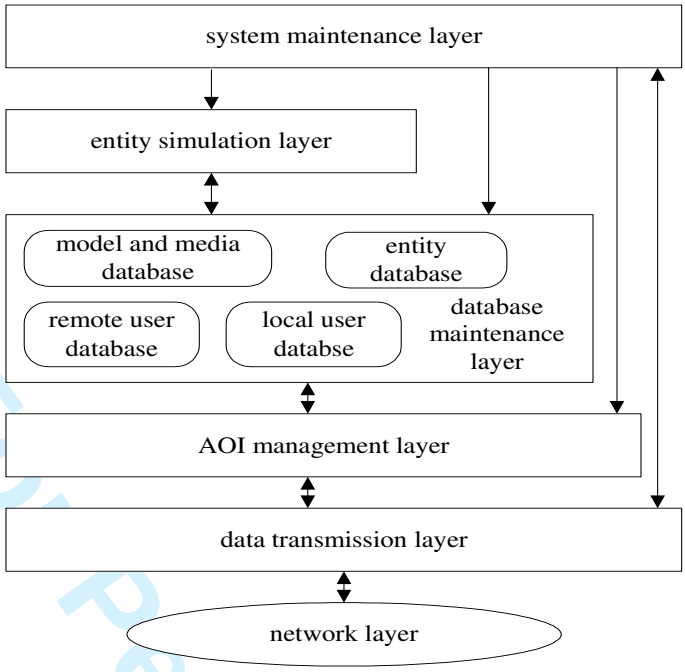

Figure 7. Hierarchical structure of region server

Database maintenance layer saves the models and media data, such as 3D model, 2D texture, audio and video, in the model and media database. Besides, it records the updates of entities and users into the entity database and user database. When it receives the partitioning information from the system maintenance layer, it will add or delete the corresponding records of entities or users. For example, new partitioning information enlarges the original region, which may make more users or entities managed by this region server, at this time, these updates from new users or entities are added into the database.

AOI management layer maintains the area of interest of each user. When it receives the update messages from the clients, it will find out the addresses of the clients where the messages are necessary to be sent in order to keep the consistent views from the different clients.

Hierarchical structure of main server

The hierarchical structure of the main server is shown in Figure 8.

The region maintenance layer partitions the virtual environment according to the messages from the region server about the load of system and traffic. When new region server joins or exits the system, new partitioning information is sent to the

region server. At this time, the region maintenance layer updates region server database and model and media database on database management layer. Region server database maintains the logical connections with all the region servers. Model and media database keeps all the models and media data, and a region mapping list. Backup server version database records the version information about the backup server.

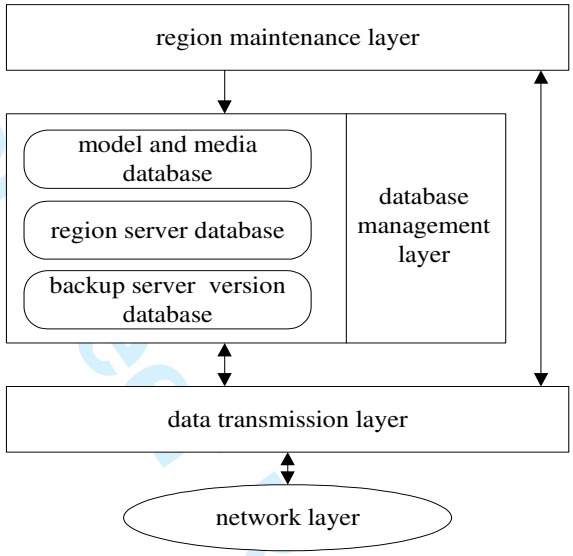

Figure 8. Hierarchical structure of main server

Backup server is the copy of the main server. It backups the main server at regular intervals and generates a new version number. This version number is put into the backup server version database in the main server.

According to the description above, the main functions, such as message filtering, entities simulation, view management and region maintenance, etc, are separated and realized in the different hosts. The new hierarchical architecture based on tailoring tree model simplifies the development complexity of a large-scale CVE system.

A Dynamic Clustering Algorithm for Partitioning the Virtual Environment

In the hierarchical architecture based on tailoring tree model, partitioning problem is one of the key issues. The existing partitioning methods have the precondition of fixed number of the servers. So these methods are based on the hypothesis: the number of servers is determinate. However, in our architecture based on tailoring tree model, the number of region servers is not determinate in advance. Therefore this

hypothesis makes these methods not suitable for this architecture. A dynamic clustering algorithm for solving the partitioning problem is proposed. This new algorithm calculates the most reasonable number of regions by tracking the number of users in the system.

Assuming that the virtual environment E is a two-dimensional space which is divided into N equal square cells $c_1, c_2 \ldots\ldots c_n$, users in the cells may only "see" a small area of the virtual environment due to the restriction of their field of vision. Consequently, users only interact with the entities in this AOI and communicate with others who reside in it. For simplicity, vision cells are used instead of the AOI. Vision cells mean the minimum set of cells that entirely cover the AOI of the user. The vision cells of a user consist of two parts: the cell where the user resides and the neighbor cells of the user, shown in Figure 9.

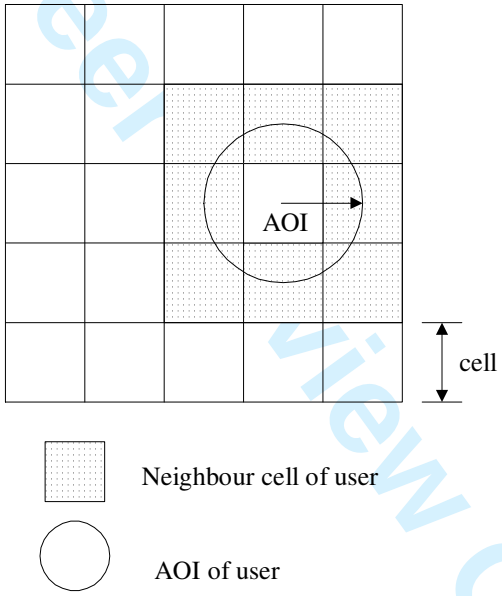

Figure 9. Vision cells of user

It can be seen that more users the cell has, more communication with its neighbor cells the cell produces. If the neighbor cells are assigned to other server, inter-server messages should be transferred to keep the consistent state. In order to reduce the inter-server traffic, the cells which have most communications with other cells are allocated as the centers of regions.

Assuming the threshold δ represents the maximum number of clients that a server can handle smoothly, let all the servers have the same threshold in order to simplify the calculation. Initially, the number of the users in the system is within the threshold δ, and there is only one region server managing all the cells in the virtual

world. The main server monitors the load of system periodically and when it finds that the number of clients reaches to the threshold $\delta \times 90\%$, which means that the region server will be overloaded soon, at this time, the dynamic clustering algorithm is executed to partition the virtual environment. The reasonable number of regions is calculated by the following formula:

$$P = \lfloor \frac{Num(E)}{\delta \times 90\%} \rfloor \quad (1)$$

P is defined as the number of regions and $Num(E)$ is the number of users in the virtual environment. From (1), it can guarantee that each region server is not overloaded if the virtual environment is partitioned into P regions. The average number of users managed by each region server is:

$$ANR = \frac{Num(E)}{P} \quad (2)$$

Assuming the virtual environment has been divided into $m(m \geq 1)$ regions, the dynamic clustering algorithm first finds the seed of each current region, that is, the cell which has the maximum users in the region. Then new seeds of regions are chosen and P regions are partitioned according to the following steps:

Step1: Calculate the aggregation degree of cells except the seeds of current regions according to the definition of aggregation degree shown as follows [13];

$$\begin{aligned} \varphi^{(m)}(c_i) &= \alpha \times \rho(c_i) + \beta \times f^{(m)}(c_i) \\ \rho(c_i) &= \frac{NUM(c_i)}{NUM(\sum_{j=1}^{n} c_j)} \\ f^{(m)}(c_i) &= \frac{\sum_{j=1}^{m} D(c_i, Seed_j)}{m \times n} \end{aligned} \quad (3)$$

In (3), $NUM(c_i)$ represents the number of users in the cell c_i, $D(c_i, Seed_j)$ represents the distance from the cell c_i to $Seed_j$. $\rho(c_i)$ represents the density function of the cell c_i. And $f^{(m)}(c_i)$ represents the distance penalty function of the cell c_i to $Seed_1, Seed_2, \cdots Seed_m$. This function can affect the choice of new seeds and avoid the seeds gathering together. In addition, $\alpha + \beta = 1$, α and β are two coefficients that denote the relative importance of the density and distance function.

Step2: Choose the cell which has the largest value of aggregation degree as the new

seed, the number of chosen seeds adds one;

Step3: If the number of chosen seeds equals P , goto Step4; Otherwise, goto Step1;

Step4: Designate each seed as the original region, and merge its neighbor cells into the region using the region growing algorithm [14] until the number of users in the region approximates or reaches to ANR .

Step5: Merge the cells c_i which has not yet been designated into a region R_j only if

$$D(c_i, Seed_j) = \min\{D(c_i, Seed_r)\} \quad r = 1, 2 \cdots P \tag{4}$$

In the dynamic clustering algorithm, the seeds of the current regions are selected in advance, and only $P - m$ new seeds are chosen by aggregation degree, which can avoid the obvious change of region center and reduce the number of migrated users. If all the seeds are chosen according to the step 1 and 2 above and the current centers of regions are not considered, this partitioning method is called static clustering algorithm. The static clustering algorithm maybe causes more number of migrated users because the center of the new region perhaps locates in the border of original region, and it makes half of the users have to migrate to another region.

When the main server receives the request to exit the system from the region server S_i , the dynamic clustering algorithm executes the following steps:

Step1: Calculate the current load $Num(E)$ in the virtual environment;

Step2: If $Num(E) < (P-1) \times \delta \times 90\%$, go to Step3; Otherwise, quit;

Step3: Find $Seed_1, Seed_2, \cdots Seed_{i-1}, Seed_{i+1}, \cdots Seed_P$;

Step4: Merge its neighbor cells of each seed into the same region using the region growing algorithm until all the cells are designated to regions.

Time-bounding-box-based Filtering Scheme

In a large scale CVE system, there is a conflict between maintaining the state consistency and supporting more users sharing the virtual environment. In order to meet the requirement of scalability, clients need to do more message filtering to decrease network traffic. However, in order to meet the requirement of consistency, clients need to do less message filtering to increase the accuracy of simulation. Time-bounding-box-based filtering scheme is proposed to make a tradeoff and solve the conflict [19].

Common message filtering methods in the client/server architecture with multiple server or single server set the AOI of each entity, and entities only receive the state change of other entities in their AOIs. However, messages in their AOIs are still not the same important to the users. Like in the real environment, people pay more attention to the things happened in the front sight. Therefore, time-bounding-box-based filtering scheme divides the virtual environment into two parts: the background space and entities' movement space. The time-bounding-box is defined to describe the entities' movement space. And time-bounding-box-based filtering scheme is used to reduce the traffic among the region servers and clients.

Let $S_i(t)$ and $V_i(t)$ represent respectively accurate position and velocity of the entity e_i in time t. T is the interval of the user's state update. The time-bounding-box of the entity e_i in position $S_i(t)$ is defined as

$$\begin{aligned} &TB(S_i(t)) = \{(x, y, z) \mid \\ &x_{\min,i}(t) + \min(V_{i,x}(t), 0) \times T \le x \le x_{\max,i}(t) + \max(V_{i,x}(t), 0) \times T, \\ &y_{\min,i}(t) + \min(V_{i,y}(t), 0) \times T \le y \le y_{\max,i}(t) + \max(V_{i,y}(t), 0) \times T, \\ &z_{\min,i}(t) + \min(V_{i,z}(t), 0) \times T \le z \le z_{\max,i}(t) + \max(V_{i,z}(t), 0) \times T\} \end{aligned} \tag{5}$$

$x_{\min,i}(t)$, $x_{\max,i}(t)$, $y_{\min,i}(t)$, $y_{\max,i}(t)$, $z_{\min,i}(t)$, $z_{\max,i}(t)$ represent respectively the minimum and the maximum of entity's projection on three coordinate axes when the entity is in the position $S_i(t)$. And $V_{i,x}(t), V_{i,y}(t), V_{i,z}(t)$ represent respectively velocity component of $V_i(t)$ in the direction of three coordinate axes. The time-bounding-box of the entity in t is shown in Figure 10.

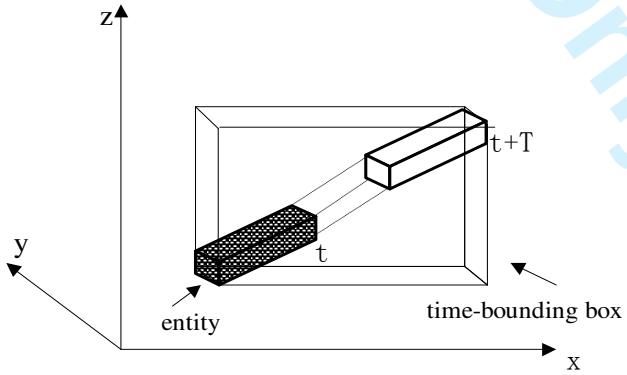

Figure 10. Time-bounding-box of the entity in t

In the time-bounding-box-based filtering scheme, each client not only maintains

accurate models of local entities and dead reckoning models of all entities, but maintains time-bounding boxes of all entities.

Let $S_i^{'}(t)$ and $V_i^{'}(t)$ represent respectively estimated position and estimated velocity of the entity e_i in time t. The time-bounding box of the entity e_i in position $S_i^{'}(t)$ is defined as

$$\begin{aligned} &TB(S_i^{'}(t)) = \{(x, y, z) \mid \\ &x_{\min,i}^{'}(t) + \min(V_{i,x}^{'}(t), 0) \times T \le x \le x_{\max,i}^{'}(t) + \max(V_{i,x}^{'}(t), 0) \times T, \\ &y_{\min,i}^{'}(t) + \min(V_{i,y}^{'}(t), 0) \times T \le y \le y_{\max,i}^{'}(t) + \max(V_{i,y}^{'}(t), 0) \times T, \\ &z_{\min,i}^{'}(t) + \min(V_{i,z}^{'}(t), 0) \times T \le z \le z_{\max,i}^{'}(t) + \max(V_{i,z}^{'}(t), 0) \times T\} \end{aligned} \tag{6}$$

Because estimated velocity of the entity in time t uses the accurate velocity of the entity in time $t-1$, that is

$$V_i^{'}(t) = V_i(t-1) \tag{7}$$

$S_i^{'}(t)$ is also represented as:

$$\begin{aligned} &TB(S_i^{'}(t)) = \{(x, y, z) \mid \\ &x_{\min,i}^{'}(t) + \min(V_{i,x}(t-1), 0) \times T \le x \le x_{\max,i}^{'}(t) + \max(V_{i,x}(t-1), 0) \times T, \\ &y_{\min,i}^{'}(t) + \min(V_{i,y}(t-1), 0) \times T \le y \le y_{\max,i}^{'}(t) + \max(V_{i,y}(t-1), 0) \times T, \\ &z_{\min,i}^{'}(t) + \min(V_{i,z}(t-1), 0) \times T \le z \le z_{\max,i}^{'}(t) + \max(V_{i,z}(t-1), 0) \times T\} \end{aligned} \tag{8}$$

The time-bounding-box is used to judge if the update should be sent to make a correction. If the time-bounding boxes of the local entity e_i in position $S_i(t)$ and $S_i^{'}(t)$ and the time-bounding box of the remote entity e_j in position $S_j^{'}(t)$ overlap, an interaction between e_j and e_i will be likely to happen soon. At this time, the update of the local entity e_i is sent immediately to make a correction. Otherwise, the updates are sent only when the estimated route derivates from the real route in some way.

Assuming Δt is the maximum of network delay in the CVE system and Δt_{ji} ($\Delta t_{ji} \le \Delta t$) is the network delay between host j and host i, when the host j receives the update position of the entity e_i in host i, the entity e_i has already changed its position in host i. The offset between its estimated position and its accurate position is represented as

$$\Delta S_{ji} = \int_0^{\Delta t_{ji}} V_i(t)dt \approx V_i(t) \times \Delta t_{ji}$$

(9)

In order to eliminate the inconsistency caused by the transmission delay, when the update of the entity e_i is sent to other hosts, the compensative position $CS_i(t)$ of e_i is sent instead of the accurate position:

$$CS_i(t) = S_i(t) + V_i(t) \times \Delta t$$

(10)

In addition, user's AOI is divided into interactive area and perceivable area, shown in Figure 11.

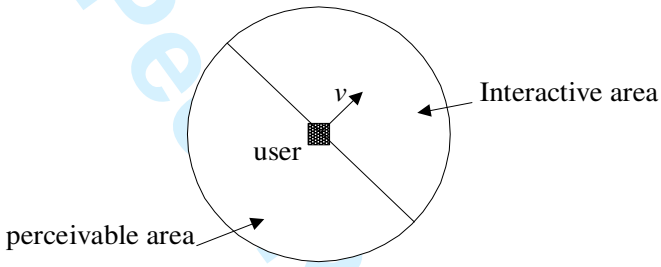

Figure 11. Interactive area and perceivable area of user's AOI

User's interactive area is a part of area in its AOI where the user can change the state of entities. User's perceivable area is the rest part of area in its AOI where the user can "perceive" the change of entities' states. In order to improve the consistency and filter more messages, multi-level DR algorithm is used [20]:

Level 1: If U_1's AOI and U_2's AOI do not overlap, only "heart-beat" messages are sent to each other, a maximum of threshold is used in U_1's extrapolation;

Level 2: If U_1's perceivable area and U_2' perceivable overlap, U_1 only can perceive U_2 but can not change U_2's state. At this time, a relatively large threshold is used in U_1's extrapolation;

Level 3: If U_1's Interactive area and U_2' AOI overlap, U_1 can change U_2's state. At this time, a relatively small threshold is used in U_1's extrapolation.

Level 4: If U_1's time-bounding box and U_2' time-bounding box overlap, U_1need to send messages immediately. The zero threshold is used at this time.

Experiments and Results

A CVE testing system was implemented to evaluate the effectiveness of this scalable architecture with the dynamic clustering algorithm for scalability and time-bounding-box-based filtering scheme for consistency in our laboratory. The hardware platform has been built by 13 PC with P8700 2.53GHz and 1Gbytes RAM. The software platform uses MAK VR-Link to realize the communication among servers and clients. The MAK VR-Link is an object-oriented library of C++ functions and definitions that implement the High Level Architecture (HLA) and the DIS protocol which realize the message transmission on the network. The main server first initiates the testing system: the virtual environment contains 5000×5000 m^2 ; the radius of AOI of each user is 250 m; the virtual environment is divided into 400 cells. Initially, the number of users in the system is 30. At this time, there are only a main server, a region server and six clients in the system. Each client maintains 5 users' movement states. Each user's movement is controlled by an application program. In order to join the CVE system, each user first connects the main server, then gets the network address of its region server according to the current location in the virtual environment. Finally, it connects with its region server to require the information about its visible virtual environment and the updates about other remote users. These users maintained in the clients move in the virtual environment and their update states are sent to the region server according to time-bounding-box-based filtering algorithm. Clients not only maintains real movement models of local users and dead reckoning models of all remote users in its visible virtual environment, but maintains time-bounding boxes of all users in this visible virtual environment. The threshold δ of each server is 100. The number of users in the CVE system increases by about 2 users per minute. Users' distribution in the virtual environment is uniformed, skewed or clustered respectively. Users move circularly in the virtual environment by Changing Circular Pattern (CCP) model [21] with radius of 250 m. The main server automatically generates the users' location and movement model by the test parameters above. New users continuously join the system and their states are maintained by the region servers. The region server periodically updates the region's information from the main server to require the new users' states. With the users increasing, a region server can not afford the users' update, the main server executes the partitioning algorithm and more region servers are connected with the main server. The region information including the region scope and environment feature, the users'

state and their movement models in this region is transferred to initiate the new region server. The new region server's dynamic configuration generally spends much time and many network resources since a large mount of data need to be transferred from one region into another new region, therefore a good partitioning algorithm should reduce the number of migrated users to improve the scalability when new region servers are deployed. In our CVE testing system, the partitioning results in different simulation time are shown in Figure 12, Figure 13 and Figure 14.

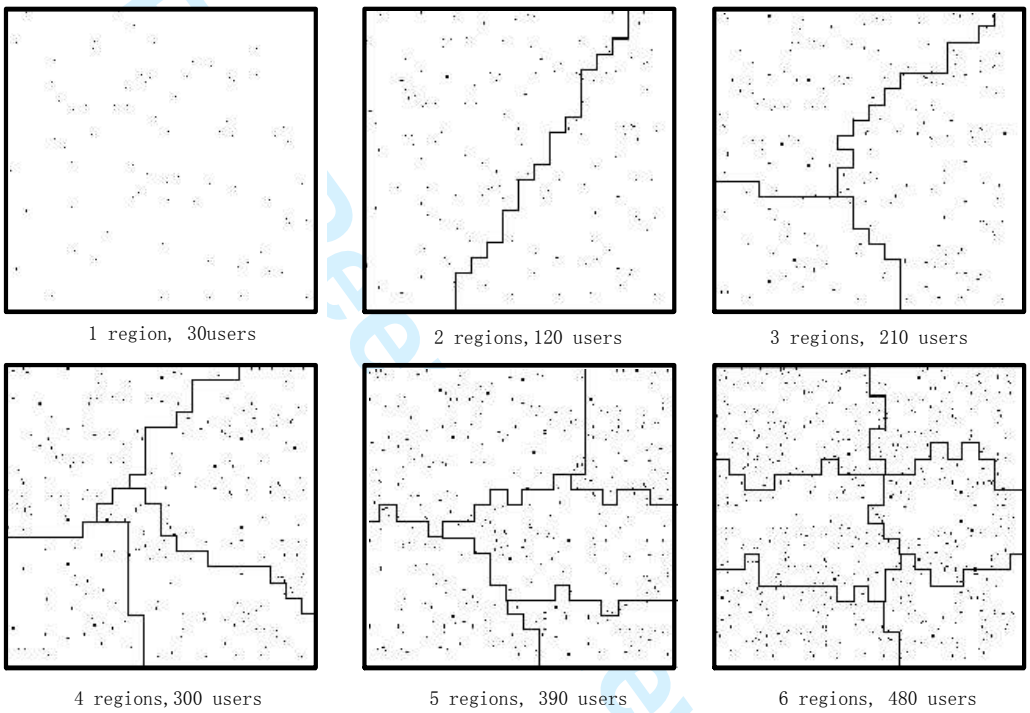

Figure 12.The partitioning result with uniformed distribution

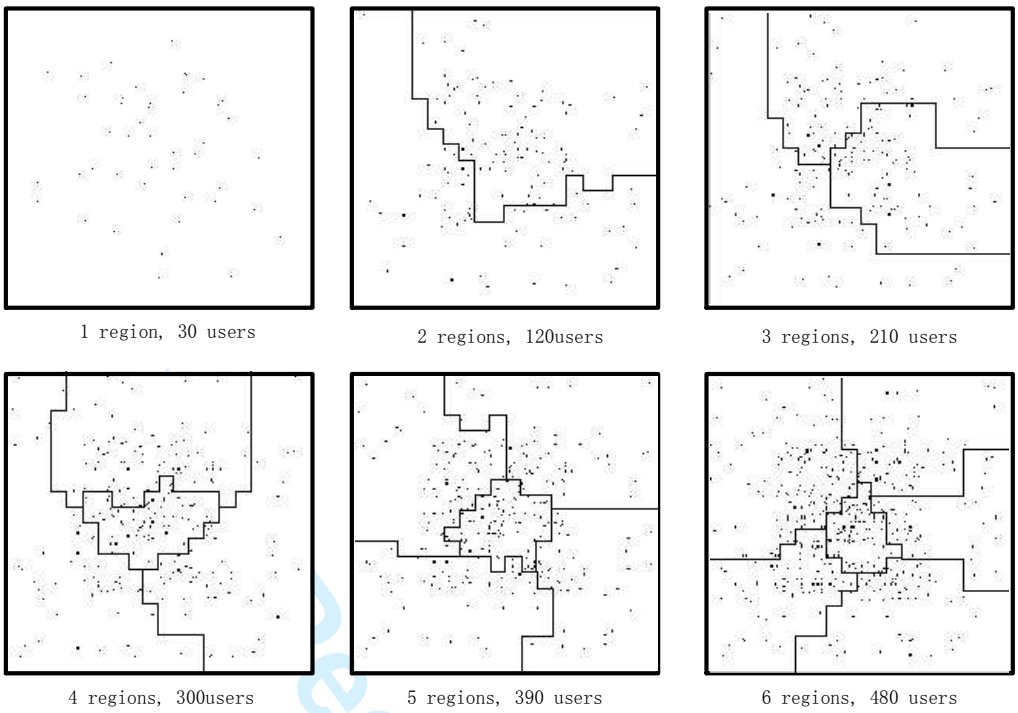

Figure 13. The partitioning result with skewed distribution

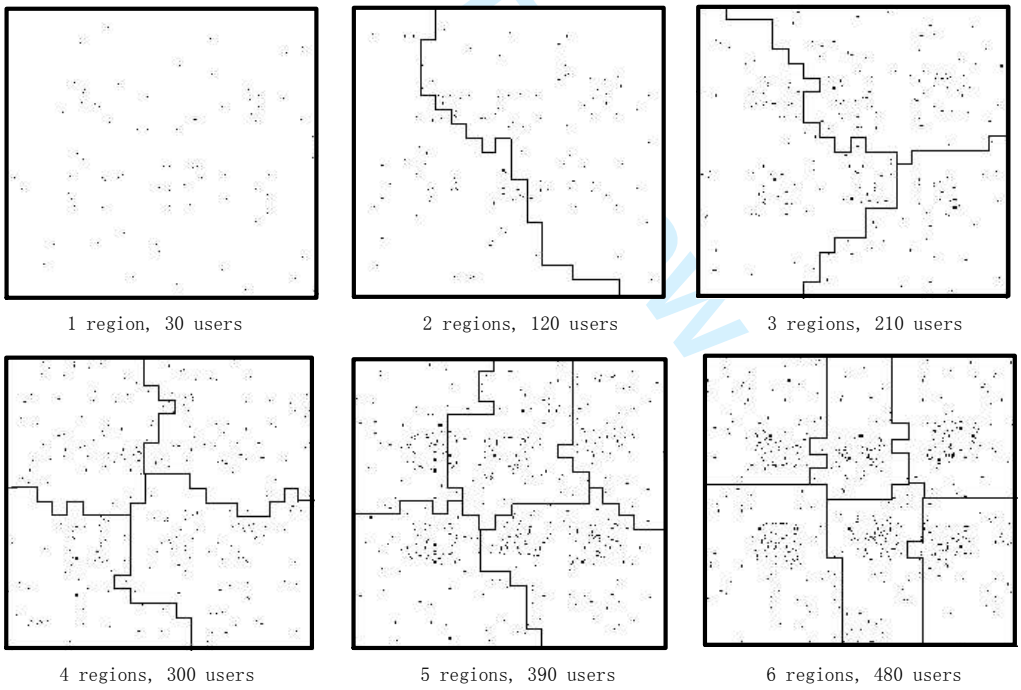

Figure 14. The partitioning result with clustered distribution

When the number of regions increases and new region servers are deployed, the number of migrated users in dynamic clustering algorithm is compared with that in static clustering algorithm in Figure 15, Figure 16 and Figure 17.

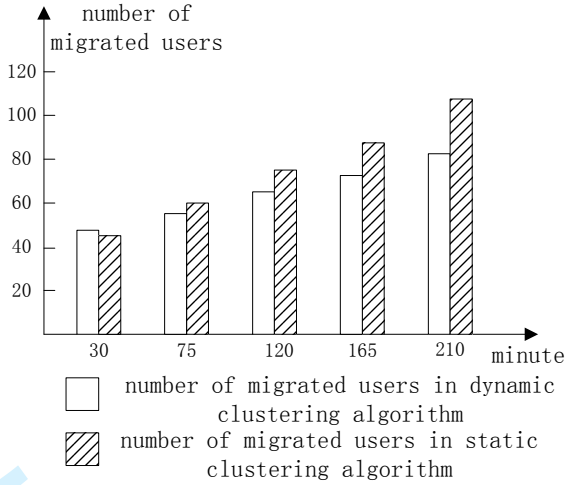

Figure 15. The number of migrated users with uniformed distribution

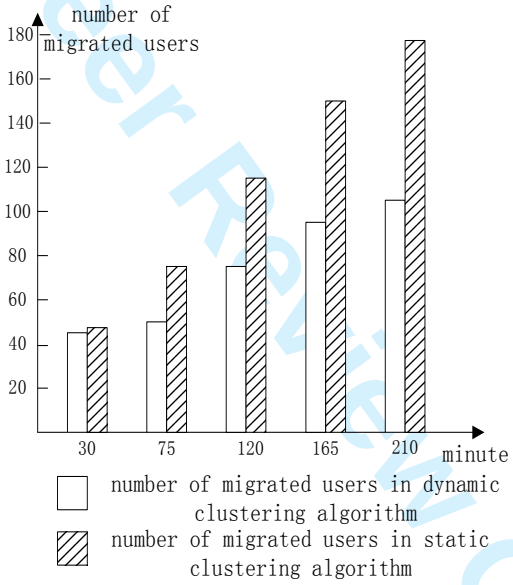

Figure 16. The number of migrated users with skewed distribution

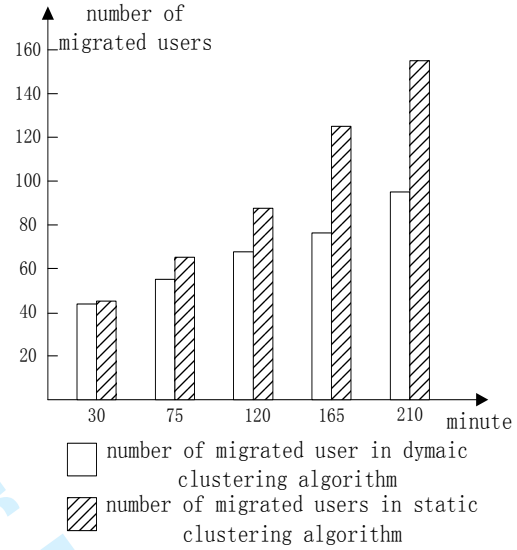

Figure 17. The number of migrated users with clustered distribution

According to the partitioning description above, most seeds are chosen not from the whole virtual environment but from the existing region in the dynamic clustering algorithm, which makes most of the cells in a region still managed by original region server. Therefore, the number of migrated users in dynamic clustering algorithm is less than that in static clustering algorithm.

To evaluate the quality of partition, the load rate of each region server and the traffic among the servers are used to test our partitioning algorithm:

Testing of Load rate of servers

The load rate of server i is defined as:

$$\tilde{l}_i = \frac{W_i}{\delta} \tag{11}$$

W_i represents the workload of the server i. In order to improve the response time of system, the reasonable partitioning algorithm should make none of the servers reach to the saturation according to the experience made by Morillo [12], that is $\tilde{l}_i$ <1.

The load rate of region servers is shown from Table1 to Table 3 when users are located in the virtual world following a uniformed, skewed, clustered distribution.

Table 1. CPU Utilization in Uniformed User Distribution

The	Load Rate of Servers

number of Users	Server1	Server2	Server3	Server4	Server5	Server6
90	0.46	0.44	0	0	0	0
180	0.63	0.66	0.51	0	0	0
270	0.71	0.73	0.69	0.57	0	0
360	0.75	0.76	0.72	0.70	0.67	0
450	0.72	0.78	0.74	0.81	0.74	0.71

Table 2. CPU Utilization in Skewed User Distribution

The number of Users	Load Rate of Servers					
	Server1	Server2	Server3	Server4	Server5	Server6
90	0.47	0.43	0	0	0	0
180	0.64	0.67	49	0	0	0
270	0.59	0.75	0.64	0.72	0	0
360	0.62	0.77	0.61	0.81	0.79	0
450	0.73	0.65	0.68	0.87	0.75	0.82

Table 3. CPU Utilization in Clustered User Distribution

The number of Users	Load Rate of Servers					
	Server1	Server2	Server3	Server4	Server5	Server6
90	0.48	0.42	0	0	0	0
180	0.67	0.58	0.55	0	0	0
270	0.72	0.67	0.69	0.62	0	0
360	0.68	0.75	0.65	0.79	0.73	0
450	0.73	0.77	0.71	0.80	0.71	0.78

From Table 1, Table 2 and Table 3, it can be seen that none of servers is overloaded in the simulation. This is because the number of region servers is calculated according to the formula (1). Our partitioning method always guarantees that the total load in the virtual environment is no more than 90% of the maximum load of existing region servers, or new region server should be deployed to reduce the workload of the existing region servers. The testing of load rate of servers proves that our dynamic partitioning algorithm can effectively assign the load to the computing resources.

Testing of traffic among servers

Traffic among servers is defined as:

$$C_p^{\ L} = \sum_{j=1}^{P} \sum_{i=1}^{P} D_{ij} \tag{12}$$

D_{ij} represents the communication cost of region i and region j [22], $C_p^{\ L}$ represents the communication cost of the system. According to the description of architecture above, traffic among servers is recorded in the main server.

Table 4 and Table 5 respectively shows the traffic among servers and the executive time using the new dynamic clustering partitioning and time-bounding-box-based

filtering algorithm (DCPTF), ACS-based partitioning (ACS) [23] and Linear Optimization Technique (LOT) [9] when the users are uniformed, skewed, clustered distribution in virtual environment.

Table 4. Traffic Among Servers of Three Partitioning Results

Number of Users	DCPTF			ACS			LOT		
	Uniform	Skewed	Clustered	Uniform	Skewed	Clustered	Uniform	Skewed	Clustered
90	16	23	11	39	31	26	34	32	25
180	44	38	28	70	72	89	72	68	84
270	62	56	51	106	96	137	95	88	121
360	87	70	76	114	104	154	117	95	148
450	107	84	93	136	111	181	139	108	173

Table 5. Executive Time of Three Partitioning Results

Number of Users	DCPTF			ACS			LOT		
	Uniform	Skewed	Clustered	Uniform	Skewed	Clustered	Uniform	Skewed	Clustered
90	62	65	63	106	143	174	211	257	284
180	86	92	94	142	210	245	362	386	403
270	104	112	114	195	304	422	488	522	556
360	116	124	125	222	380	576	614	653	732
450	132	142	141	258	492	729	851	896	924

From Table 4, DCPTF algorithm has less traffic among the servers than ACS and LOT algorithms because the filtering scheme is adopted in the client sides, which proves that DCPTF algorithm can effectively assign communication resources and save the network bandwidth. From Table 5, DCPTF algorithm has the least executive time in three partitioning algorithms, which proves that DCPTF algorithm has a good performance in efficiency. Therefore it indeed optimizes both speed and quality in partitioning problem and improves the partitioning performance.

Testing of system response

In order to evaluate the scalability of several architectures, the parameters that model the virtual environment system are as follows:

The distribution of the users in the CVE is uniformed;

Six clients maintain the movement states of all the users;

Six new users join the system each three minutes and their states are specially maintained by 6 clients during the simulation experiment;

The simulation experiment ends whether system response value is more than one second or the simulation time is more than four hours.

In order to compare the system response in different architectures, different architectures are built. A CVE testing system based on our architecture with tailoring tree model is as follows: Initially, there are only a main server and a region server.

New region servers are deployed with the increase of users. System response value is tested in our architecture according to the following step:

(1) Region servers send test messages to the main server randomly during the simulation;

(2) The main server receives the test message and immediately sends a response message to the region server;

(3) Region servers receive the response message and record the intervals;

(4) The average value of intervals from all the region servers is calculated as the system response.

A CVE testing system based on P2P architecture is built and there is no server in the system. Each user sends the update to other clients according to the DR algorithm with the threshold 20. System response value is tested in P2P architecture according to the follow steps:

(1) Clients send test messages to other clients randomly during the simulation;

(2) Clients receive the test messages and immediately send response messages to the sponsors;

(3) Clients receive the response messages and record the intervals;

(4) The average value of intervals from all the clients is calculated as the system response.

A CVE testing system based on client/server architecture with single server is as follows: The main server is the only server in the system, and six clients all connect the server. Each user sends the update to the server and receives the update from the server. System response value is tested in client/server architecture with single server according to the follow steps:

(1) Clients send test messages to the server randomly during the simulation;

(2) The server receives the test messages and immediately sends response messages to the sponsors;

(3) Clients receive the response messages and record the intervals;

(4) The average value of intervals from all the clients is calculated as the system response.

A CVE testing system based on client/server architecture with three servers is as follows: The virtual environment is divided into three regions and each region is maintained by a server. There are three servers in the system, and six clients connect one of three servers. Each user sends the update to its server and receives the update

from the server. Each server sends the updates of users in the region border to other servers to update the remote users. System response value is tested in client/server architecture with three servers according to the follow steps:

(1) Servers send test messages to other servers randomly during the simulation;

(2) The servers receive the test messages and immediately send response messages to the sponsors;

(3) Servers receive the response messages and record the intervals;

(4) The average value of intervals from three servers is calculated as the system response.

Figure 18 shows the system response values in the different architectures with the increase of the number of users in the CVE system.

From Figure 18, when the users go beyond the limits of 100, the system response of the client/server architecture with single server rapidly increases because the server overloads. When the users go beyond the limits of 300, the system response of the client/server architecture with three servers rapidly increases because the load is beyond the threshold of three servers. Our architecture with tailoring tree model and P2P architecture with DR algorithm has the almost similar performance, and the system response is beneath the 200ms with the increase of the number of users. P2P architecture with DR algorithm also has the lower system response by filtering more messages among clients, which will reduce the consistency, shown in Table 6. Comparing the results above, our architecture and algorithms can support more users in the system, which proves that it has the better scalability.

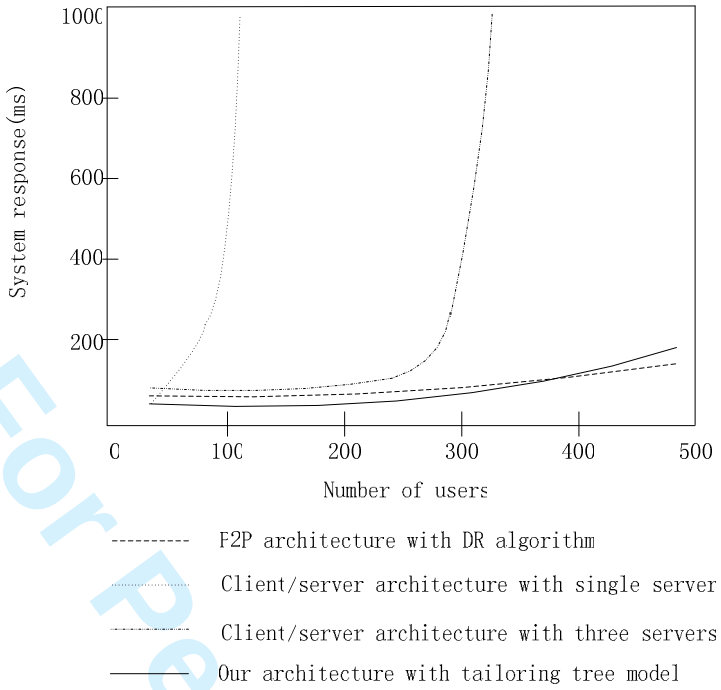

Figure 18. System Response in Different CVE Architectures

Testing of average error in interactive area and filtering rate

In order to evaluate the consistency performance, Average Error in Interactive Area (*AEIA*) is defined as:

$$AEIA = \frac{T}{n(t_n - t_0)} \sum_{i=1}^{n} \sum_{t=t_0}^{t_n} EIA_{it} \tag{13}$$

t_0 and t_n respectively refers to the beginning time and the ending time of simulation. T is the simulation step. n is the number of users in the system. EIA_{it} is the average deviation of users in U_i's interactive area at the time t. During the simulation, the clients record the accurate positions and their interactive areas of the local users at each simulation time. In the P2P architecture, the clients also record the estimated positions of the remote users at each simulation time. In our architecture, the region servers record the estimated positions of all the users at each simulation time. EIA_{it} is calculated according to the accurate position, the estimated position and the interactive area of U_i at the time t.

From the definition above, the lower is average error in interactive areas, the better is the state consistency in a CVE system.

Filtering rate of messages is defined as:

$$FR = \frac{1}{n}\sum_{i=1}^{n}\frac{NRev(e_i)}{NGen(e_i)} \quad (14)$$

$NRev(e_i)$ represents the number of update packets received from the entity e_i in the simulation duration, and $NGen(e_i)$ represents the number of update packets generated by the object e_i in the simulation duration. Therefore the lower filtering rate will improve the scalability of system by saving the network bandwidth and supporting more users sharing the virtual environment.

According to the definition above, the results of average error in interactive area and filtering rate of messages are shown in Table 6 when there are 30 users in the virtual environment and simulation duration is 30 minutes. In DR algorithm, the threshold θ is from 20 to 200. In time-bounding-box-based filtering algorithm, the threshold θ is respectively 200, 80, 20 and 0 in 4 levels.

Table 6. Results of verage error in interactive area

	DR				Time-bounding-box-based filtering
	20	80	140	200	
AEIA	6	29	43	56	5
FR	22.5%	5.8%	3.5%	2.4%	3.1%

From Table 6, the *AEIA* increases as the threshold increases in DR algorithm. This is because that the deviation of the estimated position and the accurate position of each user increases as the threshold increases according to the description of DR algorithm. From Table 6 and Figure 18, although the P2P architecture with DR algorithm has the similar performance in the scalability of the CVE system, the time-bounding-box-based filter algorithm used in our architecture with tailoring tree model gets a relatively low filtering rate of messages with a small average error in interactive area. This is because that the threshold is zero when the time-bounding boxes overlap, which means no deviation between the estimated position and the accurate position at this time and reduces the *AEIA* value. Table 6 and Figure 18 prove that our filtering algorithm can effectively improve the consistency and reduce the traffic between clients and servers at the same time.

In summary, the results in testing of load rate of servers, traffic among servers and system response show that our algorithms in hierarchical architecture based on tailoring tree model can improve the scalability by the reasonable assignment of the

computing load among the region servers. The results in testing of average error in interactive area and filtering rate show that our algorithms in hierarchical architecture based on tailoring tree model can improve the consistency by the reasonable assignment of the communication load among the clients and servers. As a whole, our hierarchical architecture based on tailoring tree model and the DCPTF algorithm improves the scalability and consistency in the CVE system.

Conclusions

To solve the capacity constraint of users in existing multiple-server architecture, a hierarchical architecture for improving scalability and consistency is proposed to support more users in the CVE system. The function structures of the main server, the region server and the client are detailed. Compared with the traditional multiple-server architecture, the hierarchical architecture simplifies the complexity of the development of a large-scale CVE system. Moreover, the new architecture can deploy region servers at any time to share the excessive workload of overloaded servers in order to improve the scalability of the system. Since the number of region servers is uncertain, the virtual environment needs to be repartitioned. The existing partitioning methods have the precondition of fixed number of the regions and they are not suitable for this architecture. In our hierarchical architecture, the main server uses a dynamic clustering algorithm to partition according to the load of system; the clients use the time-bounding-box-based filtering scheme to reduce the intra-server communication to improve the consistency of the system. The experimental results show that our dynamic clustering partitioning algorithm and time-bounding-box-based filtering scheme can effectively utilize the computing resources and communication resources to improve the scalability and consistency.

Acknowledgment
This work was supported in part by the Grand Science & Technology Program Shanghai China (No. 09DZ1122900, 07DZ11310) and Industrial Innovation Grand Projects (No. 07CH-008)

Xiaomei Hu received the BS degree in Computer science and technology department from university of electronic science and technology of China in 2000. She received the MS degree in control theory and control engineering department from southwest university of science and technology in 2003. She received the PhD degree in computer science and engineering department from northwestern polytechnical university in 2007. Currently, she works in Shanghai Key Laboratory of Mechanical Automation and Robotics of shanghai university. Her research interests include distributed virtual environment, computer graphics, image processing.
Lilan Liu received the PhD degree in Shanghai university in 2004. Currently, she is an associate professor of school of mechatronics engineering and automation at shanghai university. Her research interests include virtual reality, Virtual Simulation, Manufacturing Grid, CIMS and mechatronics.

Tao Yu received the PhD degree in CIMS and Robert Center from shanghai university in 1993. Now he is a professor of school of mechatronics engineering and automation at shanghai university. Dr. Yu is a committee member in Technical Committee 5 of International Federation for Information Processing. His research interests include virtual reality, Manufacturing Grid, CIMS and mechatronics.

References

[1] M.R. Macedonia, M.J. Zyda, "Taxonomy for Networked Virtual Environments", IEEE Multimedia, 1997, pp. 48-56.

[2] M. Macedonia, M. Zyda, D. Pratt, P. Barham , and S. Zeswitz, "NPSNET: A Network Software Architecture for Large Scale Virtual Environments", Presence, 3(4), 1994: pp.265-287.

[3] E. Frecon, "DIVE: A Scalable Network Architecture for Distributed Virtual Environment", Distributed Systems Eng. J., Special issue on distributed virtual environments, 5(3), 1998: pp.91-100.

[4] Kyungmin Lee and Dongman Lee, "Scalable dynamic Load Distribution Scheme for Multi-server Distribution Virtual Environment Systems with Highly-Skewed User Distribution", In: Proceedings of VRST'03, Osaka JAPAN October 1-3, 2003: pp.160-168.

[5] K. Michael, and W. W. Johnny, "Scalability Analysis of the Hierarchical Architecture for Distributed Virtual Environments", IEEE transactions on parallel and distributed systems, 19(3), 2008, pp.408-417.

[6] P. Morillo, S. Rueda, J. M. Ordun, and J. Duato, "A Latency-Aware Partitioning Method for Distributed Virtual Environment Systems", IEEE Transactions on Parallel and Distributed Systems, 18(9), 2007, pp.1215-1226.

[7] C. Joslin, D.G. Thomas, M.T. Nadia "Collaborative Virtual Environments: From Birth to Standardization". IEEE Communications Magazine, 2004, 42(4), 28-33.

[8] L. Chen, G.C. Chen, and C.G. Ye, "Using Collaborative Knowledge Base to Realize Adaptive Message Filtering in Collaborative Virtual Environment", Proceeding of ICCT, 2003, pp.1655-1661.

[9] J.C.S. Lui, M.F.Chan, "An Efficient Partitioning Algorithm for Distributed Virtual Environment Systems", IEEE Transactions On Parallel and Distributed Systems, Vol.13, No.3, March 2002: pp.193-211.

[10] P. Morillo, M. Fernandez, J.M. Orduna, "A comparison study of modern Heuristics for Solving the Partitioning Problem in Distributed Virtual

Environment Systems", In: Proceedings of International Conference on Computational Science and its Applications, May 18-21, 2003, pp.458-467.

[11] Kyungmin Lee and Dongman Lee, "Scalable dynamic Load Distribution Scheme for Multi-server Distribution Virtual Environment Systems with Highly-Skewed User Distribution", In: Proceedings of VRST'03, Osaka JAPAN October 1-3, 2003: pp.160-168.

[12] P. Morillo, J.M. Orduna, M. Fernandez, J. Duato, "On the Characterization of Distributed Virtual Environment Systems", The 9th International Euro-Par Conference Klagenfurt, Austria, August 26-29, 2003: pp. 1190 – 1198.

[13] Hu XM, Zhai ZJ, Cai XB, "A task-based clustering method for the dynamic partitioning management in CVE systems", WIMOB'2005: IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, VOL, 4 Proceedings,2005 : 139-144

[14] Hu XM, Zhai ZJ, Cai XB, "A region growing algorithm for solving CVE partitioning problem", Knowledge Enterprise: Intelligent Strategies In Product Design, Manufacturing, And Management 207, 2006: 960-965

[15] E. Lety, T. Turletti, and F. Baccelli. "Cell-based Multicast Grouping in Large-Scale Virtual Environments". Tech. rep. INRIA, France, July 1999.

[16] Z. J. Zhai, X.M. Hu, and X.B. Cai, "An adaptive grouping Scheme in Collaborative Virtual Environment Systems", In Proceeding of 2005 IEEE conference on Cyberworld, 2005, pp. 311-315

[17] IEEE 1278. Standard for Informantion Technology - Protocols for Distributed Interaction Simulation Applications, 1993.

[18] M. Mauve, J. Vogel, V. Hilt and W. Effelsberg, "Local-Lag and Timewarp: Providing Consistency for Replicated Continuous Applications", IEEE transactions on Multimedia, Vol.6, No.1, 2004, pp.47-57.

[19] X.M. Hu, W.H. Zhu, T. Yu, "A New Consistency Maintenance Algorithm based on Prediction and Compensation Scheme in CVE Systems", In: Proceedings of 2008 IEEE International Conference on Networking, Sensing and Control, 2008: pp. 457-461

[20] W. Cai, F.B.S. Lee, and L. Chen, "An auto-adaptive Dead Reckoning Alogrithm for Distributed Interactive Simulation", In Proceedings of Workshop on Parallel and Distributed Simulation, 1999: pp.82-89

[21] N. Beatrice, S. Antonio, L. Rynson, and L. Frederick, “A Multi-server Architecture for Distributed Virtual Walkthrough”. VRST'02, June 2002:pp.163-170.

[22] Hu Xiaomei, Zhai Zhengjun, Cai Xiaobin, “A New Self-Adaptive Approach for Optimizing both Speed and Quality in Partitioning Algorithm in CVE Systems”, Journal of Northwestern Polytechnical University, Vol.24, No.5, 2006：619-623

[23] P. Morillo, M. Fernandez, J.M. Orduna, “An ACS-Based Partitioning Method for Distributed Virtual Environment Systems”. In: Proceeding of Bob Werner, Parallel and Distributed Processing Symposium, Nice, France, IEEE Computer Society , 2003.