Journal of Southeast University (English Edition) Vol. 22, No. 2, pp. 180 – 184 June 2006 ISSN 1003—7985

A hybrid DVE architecture based on temp server

Wu Yanhua[1] Cai Yunze[2] Xu Xiaoming[3,4]

([1]Computer Science and Engineering Department, Shanghai Jiaotong University, Shanghai 200030, China)
([2]Automation Department, Shanghai Jiaotong University, Shanghai 200030, China)
([3]University of Shanghai for Science and Technology, Shanghai 200093, China)
([4]Shanghai Academy of Systems Science, Shanghai 200030, China)

Abstract: A hybrid distributed virtual environment (DVE) architecture is presented by importing a peer-to-peer (P2P) idea into the traditional client/server (C/S) architecture to improve the DVE system's scalability. The mathematical model of the overload of the center server was built and a series of simulation experiments were performed to validate the conclusions. When the client number increases to a certain value, the hybrid architecture can reduce server overload with some special clients (temp servers) selected with a certain heuristic strategy. With this architecture, the DVE system can support more system clients with the same server hardware than the C/S architecture can. The server overload "pulse" phenomena causing by the exiting of the temp server can be resolved by adopting a more optimized temp server selecting strategy and by reducing the child client capability of the temp server. By combining the advantages of the C/S architecture and the P2P architecture, the hybrid DVE architecture can effectively improve the scalability of the DVE system. This is validated by theoretical analysis and simulation experiments.

Key words: distributed virtual environment; temp server; user scale; peer-to-peer; client/server

With the development of network technology and 3D image accelerating technology, the distributed virtual environment (DVE) has attracted more and more attention in recent years. One possible definition of the DVE was given by Singhal and Zyda[1]. DVE application systems have been developed during the past decade in many fields such as military simulation[2], 3D on-line games[3], interactive e-learning[4], electronic business[5], etc.

There are mainly two kinds of architecture adopted by the DVE system: the client/server (C/S) architecture[6] and the peer-to-peer (P2P) architecture[7], as shown in Figs. 1(a) and (b).

The P2P architecture does not need an expensive central server any more, which can reduce the system cost effectively. However, for a P2P system with N users, there are $o(N^2)$ messages appearing on the network in each refresh cycle, which causes network transmission congestion and delay. It becomes more serious with the increase in the user scale.

It is probably true that the C/S architecture is more prevalent nowadays. Its center server gives commercial companies an absolute control power, which guarantees the company's commercial benefits. Utilizing this architecture, a client sends an update message to the server in each refresh cycle, which is then propagated to other clients by the server to keep a consistent state for all the clients. That all the messages go through the center server makes it a performance bottleneck and a single point of failure. To solve this problem, an amendment architecture is proposed in Ref. [8], as shown in Fig. 1(c). Its main idea is paralleling a group of servers to support larger scale clients. All the client loads are distributed averagely among all the servers. Based on this architecture several load balance algorithms were proposed[9–12]. This architecture can offer unlimited scalability for the DVE system. However an obvious shortcoming is that it requires a large number of expensive servers, which leads to high commercial cost. At the same time, the center workload distributed algorithm is expensive for the server's computing resources.

A dynamic hybrid architecture (DHA) is presented in this paper. It is shown in Fig. 1(d). The DHA borrows the distributed idea from P2P architecture and merges it into the C/S architecture. It can be described as follows: The system is equal to a C/S system at initial time. When its user scale gets to a predefined

Received 2006-02-23.
Foundation items: The National Basic Research Program of China (973 Program) (No. 2002CB312200), the National Natural Science Foundation of China (No. 60575036).
Biographies: Wu Yanhua (1976—), male, graduate, wyhross@vip.sina.com; Xu Xiaoming (1957—), male, professor, xmxu@mail.usst.edu.cn.

threshold, a few users are selected as temp servers with some predefined rules. At the same time, a certain number of users will be migrated to them. To the migrated users the temp server acts as a new server. The migrating users do not communicate directly with the center server any more and they just connect with temp servers. Once the temp server gets all its direct children's update messages it will encapsulate them into a bigger message packet and send it to the center server. The temp server technology can reduce bandwidth and computing load of the center server effectively, which can reduce system costs. Because the temp server is selected from clients, no additional server cost is required.

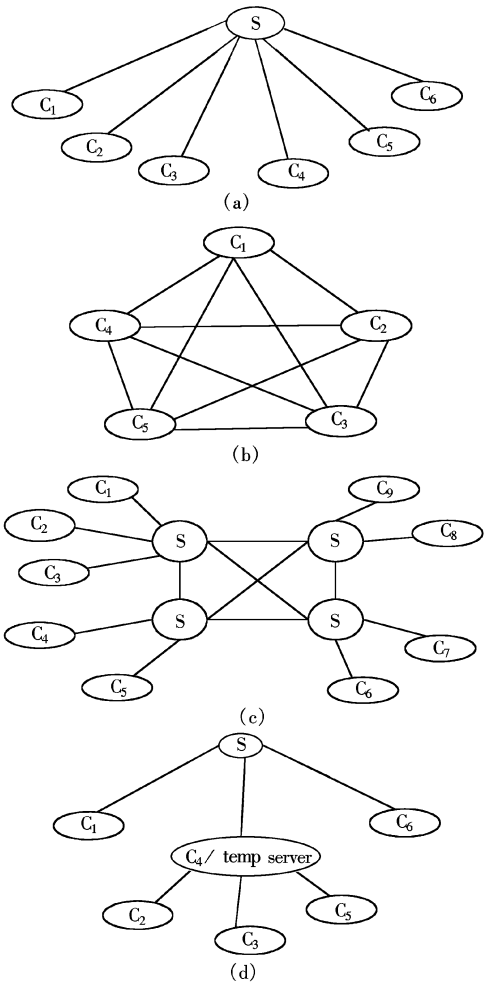

Fig. 1 Various DVE architectures. (a) Client/server; (b) Peer-to-peer; (c) Multi-server architecture; (d) Dynamic hybrid architecture

1 Dynamic Hybrid Architecture

1.1 Architecture analysis

The DHA is extended from the C/S architecture. The network topology in this system changes from time to time during running time. At the initial stage the user scale is small and the system equals the C/S architecture. When user scale increases, the architecture will change dynamically to adapt to it. Here a predefined threshold is used. Once the user scale exceeds the threshold, a client is selected as a temp server with a predefined rule. Then the center server migrates a few clients to it. Once this process finishes, the temp server and its "children" make up a group. The temp server manages all its children and substitutes all its children to communicate with the center server. The temp server receives its children's message packets in each system refresh cycle and aggregates them into a large packet and forwards it to the center server. When the center server's update message packet reaches the temp server, it decomposes it into different single packets and sends them to its destination client. From the viewpoint of server, the whole group is equal to a single user. So the user number directly connected with the server is reduced, which reduces the center server's computer and bandwidth resources. However, the temp server is a free client in nature so it can exit randomly. When a temp server exits the system, its children will have to reconnect to the center server again. This process may cause a pulse on the server's load to some degree. This problem will be discussed in detail in section 2.

1.2 DHA load model

In this paper we use server load to mark server bandwidth and CPU process resources. The main task of this section is to study how much the server load resaved for the DHA compared to the traditional client-server architecture.

The server's load is mainly associated with the number of clients the server directed to be linked, which is denoted as N_s. It includes two parts: common children clients and temp servers. Common children clients are the clients that are not chosen as temp servers. So we can easily obtain

$$N_s = N_1 + m \tag{1}$$

where N_1 is the number of common children and m is the number of temp servers. For example in Fig. 1 (d), $N_1 = 2$, $m = 1$ and $N_s = 3$. Then we define N as the total client number of the DVE system and γ_i as the children number of the i-th temp server. Then we can obtain

$$N - N_s = \sum_{i=N_1+1}^{N_1+m} \gamma_i \tag{2}$$

Because clients are free from the DVE system, they can exit randomly or not in each system refresh cycle. When a client exits, it causes different server

loads for the center server. We define L_i as the server load produced by common children i in a refresh cycle if it does not exit the system during this cycle and l_i if it exits. To simplify the system model, we suppose that all the temp servers lead to the same server load in a refresh cycle if it does not exit. If a temp server leaves the system in a refresh cycle, based on the DHA architecture, all the temp servers' children must be re-migrated to the center server, which may cause more server load. We define δ as the server's unit load caused by migrating a child from the temp server to the center server and P_i as the exiting probability of client i in one refresh cycle. Then the total server load defined as L_s of the DHA in a refresh cycle can be computed as

$$L_s = \sum_{i=1}^{N_s}(1-P_i)L_i + \sum_{i=1}^{N_1}P_i l_i + \sum_{i=N_1+1}^{N_1+m}P_i\gamma_i\delta \quad (3)$$

To compare to the C/S architecture, we give the load compute formula of the C/S architecture as

$$L_s = \sum_{i=1}^{N}(1-P_i)L_i + \sum_{i=1}^{N}P_i l_i \quad (4)$$

Eq. (4) can be rewritten as

$$L_s = \sum_{i=1}^{N_s}(1-P_i)L_i + \sum_{i=N_s+1}^{N}(1-P_i)L_i + \sum_{i=1}^{N_1}P_i l_i + \sum_{i=N_1+1}^{N}P_i l_i \quad (5)$$

We define θ as the server load difference between the DHA and the C/S architecture. From Eqs. (3) and (5), we can obtain

$$\theta = \sum_{i=N_s+1}^{N}(1-P_i)L_i + \sum_{i=N_1+1}^{N}P_i l_i - \sum_{i=N_1+1}^{N_1+m}P_i\gamma_i\delta \quad (6)$$

Not to lose generality, we assume that all the clients have the same leaving probability, defined as P. We also think all the clients lead to the same server load under the same conditions, which means L_i and l_i are constant values, defined as L and l. Then by Eqs. (2) and (6), we obtain

$$\theta = (N-N_s)[(1-P)L - P\delta] + (N-N_1)Pl \quad (7)$$

Eq. (7) is the expectation of the server load difference of the DHA and the C/S architecture in a refresh cycle with the clients exiting with probability P. In other words, the server load saved by the DHA can be computed with it. In a common case, the probability value of a user exiting the system in a short cycle time is very small. That means $P_i \ll 1$, even $P_i \approx 0$, which assures $\theta > 0$. Through the theory analysis above we can prove that the DHA architecture can reduce resource consumption compared to the C/S architecture with the equal client scale.

To study the determinant of θ, with the condition talked about above we rewrite Eq. (7) into

$$\theta \approx L(1-P)(N-N_s) \quad (8)$$

By Eqs. (2) and (8), we can obtain

$$\theta \approx L(1-P)\sum_{N_1+1}^{N_1+m}\gamma_i \quad (9)$$

We know that $\sum_{N_1+1}^{N_1+m}\gamma_i$ is the number of temp servers' children clients. Because $L(1-P)$ is a constant value, we can obtain

$$\theta \propto \sum_{N_1+1}^{N_1+m}\gamma_i \quad (10)$$

From formula (10), we can that the decreased value of center server's load of the DHA compared to the traditional C/S architecture is decided by the summation of temp servers' children clients.

2 Simulation Experiments

We have developed a simulation program tool in C++ language and done a group of simulation experiments. We logged the experimental results and studied them. A snapshot of our program prototype is shown in Fig. 2, where the big ball denotes the temp server and the small balls around it denote its children clients.

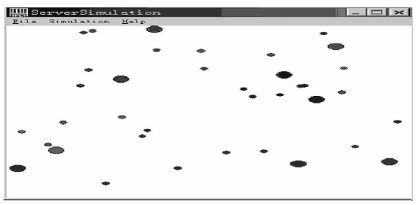

Fig. 2 A snapshot of simulation tool

2.1 Simulation process

Not to lose generality, we set up a special simulation process for client scale. In the process the system's user scale was added linearly from 0 to 50 and then reduced linearly back to 0, as shown in Fig. 3. Throughout the experimental process, the refresh cycle was set to 300 ms. The computer we used in the simulation experiment had a 2.2 GHz CPU process frequency and a 512 MB memory.

2.2 Center server load

The center server load is denoted as the packet number, the server received and sent out in each refresh cycle is defined as φ. Variable η is defined as the client threshold of the center server. If $N_s > \eta$ we think that the server is overloaded and the redundant clients will be migrated to an unsaturated temp server.

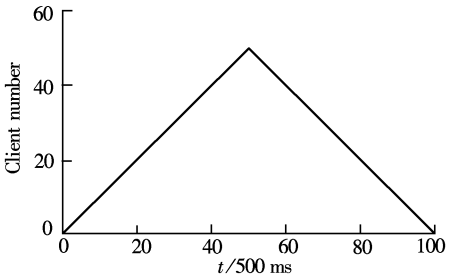

Fig. 3 User number change model

Term β is defined as the user scale a temp server can support. If $\gamma \geqslant \beta$, we think that the temp server is saturated, otherwise we think it is unsaturated. We define a triplet $V = [\eta, \beta, \varphi]$ to denote our simulation experiments. To simplify experiments, the temp server is selected randomly. With the definition of triplet, we can easily find that the C/S architecture is a special case of the DHA. It can be described as $V_1 = [\infty, 0, \varphi]$.

We chose two cases of the DHA architecture: $V_2 = [17, 3, \varphi]$ and $V_3 = [11, 5, \varphi]$. Based on the simulation process talked about above, we have done simulation experiments with the three cases (V_1, V_2, V_3) and logged φ. Simulation results are shown in Fig. 4.

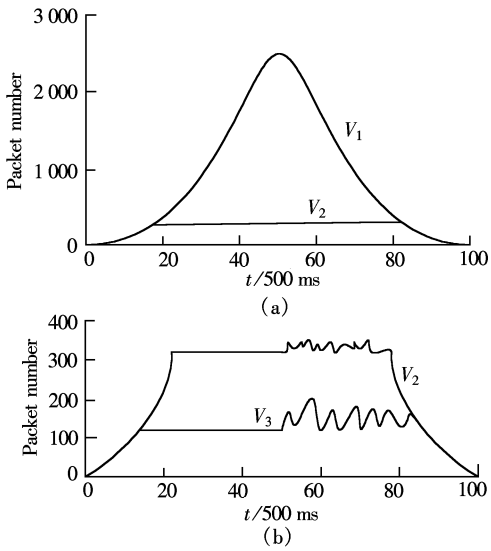

Fig. 4 Curves of server packet number. (a) V_1, V_2 curves; (b) V_2, V_3 curves

From Fig. 4(a), we can easily find that V_2 holds a lower server load than V_1, which means that even β is set to a small value. The DHA can reduce the center server's load remarkably compared to the C/S architecture.

2.3 Temp server exiting impact

From Fig. 4(b), we can find that there are a few pulses in the tails of the two curves. As we all know, a temp server is just a client in nature. So it can exit the system at any time. In the DHA, if a temp server exits the system, its children will be migrated to the center server. Some additional interactive message packets appear among the temp servers, clients and the center server, which will enhance the center server's load. It can give a reasonable explanation of the curves pulse in Fig. 4(b).

To study the effect of the temp server exiting from the server load, we define χ_i as the increasing load of the center server when the i-th temp server exits the system. From Eq. (3) we can obtain

$$\chi_i = P_i \gamma_i \delta \tag{11}$$

As we all know the exiting probability depends on the users' behavior habits. It is hard to discuss its influence, so we ignore it here. As for δ, it is determined by an objective factor-system interactive communication model. It cannot be changed at the programmers' will. So we ignore it here. We pay most our attention to the effect of γ_i-temp server's children number. From Eq. (11) we can obtain $x_i \propto \gamma_i$, which means that the addition of server load when a temp server exits the system is determined by its children number.

In Fig. 4(b) the pulse of curve V_3 is higher than curve V_2. The reason leading to this is the difference in the β of the two experiments. Because $\beta_{v_3} = 5$, there are a total of five children migrating to the center server when a saturated temp server leaves the system. However, when $\beta_{v_2} = 3$ the migrating client number is only three in V_2. So the pulse of V_3 is higher than that of V_2.

3 Conclusion

The DHA has more advantages than the traditional C/S architecture, which is proved by the theoretical analysis and simulation results. It can be regarded as an extension of the traditional C/S architecture. Compared to the traditional C/S architecture, the DHA can support a greater client scale with the same hardware condition while does not lose its advantages. Big group size can reduce the server load effectively but causes the big disturbance impulse. A beneficial tradeoff must be made between them.

References

[1] Singhal S, Zyda M. *Network virtual environments*: *design and implementation* [M]. Addison-Wesley, 1999.

[2] Zyda Michael, Hiles John, Mayberry Alex. Entertainment R & D for defense [J]. *Computer Graphics and Applications*, 2003, **23**(1): 28 − 36.

[3] Lewis Michael, Jacobson Jeffrey. Game engines in scientif-

ic research [J]. *Communications of ACM*, 2002, **45**(1): 27 - 31.

[4] Tohei Nitta, Kazuhiro Fujita, Sachio Cono. An application of distributed virtual environment to foreign language [A]. In: *The* 30*th Annual of Frontiers in Education Conference* [C]. Kansas City, Missouri, USA, 2000, **1**: 18 - 21.

[5] Zhang J, Li F S, Li H. Multi-user shared virtual reality in the exhibition of Chinese nationalities-virtual museum of Chinese nationalities [A]. In: *The Sixth International Conference on Computer Supported Cooperative Work in Design* [C]. Piscataway, NJ, USA: IEEE, 2001. 83 - 88.

[6] Larocque J. Client-server trends [J]. *Spectrum*, 1994, **31** (4): 48 - 50.

[7] Schollmeier R. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications [A]. In: *The First International Conference on Peer-to-Peer Computing* [C]. Linköping, Sweden, 2001. 101 - 102.

[8] Cai W, Xavier P, Turnr S, et al. A scalable architecture to support interactive games on the Internet [A]. In: *Proc of the* 16*th Workshop on Parallel and Distributed Simulation* [C]. Washington, DC, 2002. 60 - 67.

[9] Lui J C S, Chan M F. An efficient partitioning algorithm for distributed virtual environment systems [J]. *IEEE Transactions on Parallel and Distributed Systems*, 2002, **12**(3): 193 - 211.

[10] Nguyen T, Duong Binh, Zhou S P. A dynamic load sharing algorithm for massively multiplayer online games [A]. In: *The* 11*th IEEE International Conference on Networks* [C]. Atlanta, 2003. 131 - 36.

[11] Lau R W H, Ng B, Si Antonio, et al. Adaptive partitioning for multi-server distributed virtual environments [A]. In: *Proceedings of the* 10*th annual ACM International Multimedia Conference and Exhibition* [C]. Juan Les Pins, France, 2002. 271 - 274.

[12] Shivaratri G, Krueger P, Singhal M. Load distributing for locally distributed systems [J]. *IEEE Computer*, 1992, **25** (2): 33 - 44.

基于临时服务器的复合 DVE 结构

吴言华[1] 蔡云泽[2] 许晓鸣[3,4]

([1] 上海交通大学计算机科学与工程系,上海 200030)
([2] 上海交通大学自动化系,上海 200030)
([3] 上海理工大学,上海 200093)
([4] 上海系统科学研究院,上海 200030)

摘要: 为了增加分布式虚拟环境系统的规模可扩充性,在经典的客户/服务器(C/S)结构中引入 P2P 的概念,提出了复合分布式虚拟环境结构. 理论上对该结构下中心服务器的负载进行了建模分析,并利用仿真试验对该结构进行了试验验证. 该结构在系统客户规模增大时,动态地利用客户端资源减弱服务器端的负载,在相同的服务器硬件条件下拓展系统的客户可扩充性,同时中心服务器的存在还保持了 C/S 结构易于管理的特点. 而由于临时服务器退出导致的服务器负载"抖动"问题,可通过更优化的临时服务器选取策略以及控制临时服务器的客户规模来加以减弱. 理论分析和仿真试验表明,动态复合结构融合了 C/S 结构和 P2P 结构的优点,可有效改善 DVE 系统的规模扩充性.

关键词: 分布式虚拟环境;临时服务器;用户规模;点对点;客户/服务器

中图分类号: TP393