A FINE-GRAIN METHOD FOR SOLVING THE PARTITIONING PROBLEM IN DISTRIBUTED VIRTUAL ENVIRONMENT SYSTEMS *

P. Morillo, J. M. Orduña, M. Fernández
Departamento de Informática
Universidad de Valencia
Spain
Pedro.Morillo@uv.es

José Duato
DISCA
Univ. Politécnica de Valencia
Spain
jduato@gap.upv.es

ABSTRACT
Distributed Virtual Environment (DVE) systems have experienced a spectacular growth last years. The partitioning problem has been proven as the most critical issue in order to design scalable and efficient DVE systems. It consists of efficiently assigning clients (3-D avatars) to the servers in the system, and some methods have been proposed for solving it. However, only two of these methods take into account the non-linear behavior of DVE servers with the number of avatars attached to them. In this paper, we propose a fine-grain load balancing technique for solving the partitioning problem in DVE systems. Unlike a previously proposed technique, this proposal takes into account the estimated state of the target server before re-assigning avatars. The exceeding workload that causes the saturation of a given server is proportionally distributed among several servers, if necessary. This method avoids the cascading effect, and it allows to increase system throughput with few re-assignments of avatars. Evaluation results show that the proposed method can improve DVE system performance, regardless of both the movement pattern and also the initial distribution of avatars in the virtual world.

KEY WORDS
Distributed Virtual Environments, load balancing , cascading effect

1 Introduction

Professional high performance graphic cards currently offer a very good frame-rate for rendering complex 3D scenes in real time. On other hand, fast Internet connections have become worldwide available at a relatively low cost. These two factors have made possible the current growth of Distributed Virtual Environment (DVE) Systems. These systems allow multiple users, working on different client computers that are interconnected through different networks (and even through Internet) to interact in a shared virtual world. This is achieved by rendering images of the environment as the user would perceive them if he was located at that point in the virtual environment. Each user is represented in the shared virtual environment by an entity called *avatar*, whose state is controlled by the user through the client computer. Since DVE systems support visual interactions between multiple avatars, each client computer should be reported about any positional change of other avatars in the same virtual scenario. DVE systems are currently used in many different applications [1], such as civil and military distributed training [2], collaborative design [3], e-learning [4] or commercial multi-player game environments [5, 6].

Architectures based on networked servers are becoming a de-facto standard for DVE systems [1, 7]. In these architectures, the control of the simulation relies on several interconnected servers. Client computers are attached only to one of the servers in the system. In this architecture, when a client modifies an avatar, it also sends an updating message to its server, that in turn must propagate this message to other servers and clients. Servers must render different 3D models, perform positional updates of avatars and transfer control information among different clients. Thus, each new avatar represents an increasing in both the computational requirements of the application and also in the amount of network traffic. When the number of connected clients increases, the number of updating messages must be limited in order to avoid avoid a message outburst. In this sense, concepts like areas of influence (AOI) [1], locales [8] or auras [9] have been proposed for limiting the number of neighboring avatars that a given avatar must communicate with. All these concepts define a neighborhood area for avatars, in such a way that a given client computer c controlling avatar i must notify the movements of i (by sending an updating message) only to the client computers that control the avatars located in that neighborhood area of avatar i.

One of the key issues in the design of an efficient and scalable DVE system is the *partitioning problem* [10]. It consists of efficiently distributing the workload generated by avatars among the different servers in the system. The partitioning problem determines the overall performance of the DVE system, since it has an effect not only on the workload each server in the system is assigned to, but also on the inter-server communications (and therefore on the net-

*Supported by the Spanish MCYT, grant TIC2003-08154-C06-04

work traffic). Some methods for solving the partitioning problem have been already proposed [10, 1]. These methods provide partitioning solutions even for large scale DVE systems. However, they do not take into account the non-linear behavior of DVE systems with the number of avatars in the system, as shown in [11]. Therefore, they cannot avoid the saturation of the system due to the assignment of too many avatars to a given server.

A strategy that takes into account the non-linear behavior of DVE systems has been recently proposed for solving the partitioning problem [12]. However, the scope of this load balancing technique is local (the migration of avatars is always performed from the server managing a region of the virtual world to a server managing a neighboring region). Due to this feature, this method cannot provide efficient partitions for non-uniform movement pattern of avatars.

In a previous work [13] we proposed an adaptive load balancing technique of global scope. The global scope allowed that method to balance the workload generated by all of the avatars.This method allowed to improve the performance of DVE systems. However, that technique entirely assigns the exceeding workload that causes the saturation of a given server to the least loaded server in the system. When the system is close to its saturation point all servers are heavily loaded, and this assignment can cause the cascading effect.

In this paper, we propose a fine-grain method for solving the partitioning problem in DVE systems. This method consists of a global load balancing strategy, involving all the servers in the system. However, the exceeding workload that causes the saturation of a given server is proportionally distributed among the least loaded servers in the system. On the one hand, this strategy avoids the cascading effect, since the exceeding workload is not entirely assigned to a single server. On the other hand, this fine-grain assignment scheme allows to increase system throughput with few re-assignments of avatars. Evaluation results show that the proposed method can improve DVE system performance, regardless of both the movement pattern of avatars and also the initial distribution of avatars in the virtual world.

The rest of the paper is organized as follows: Section 2 describes the partitioning problem and the existing proposals for solving it. Section 3 details the proposed method for performing an efficient adaptive load balancing in DVE systems. Next, Section 4 presents the performance evaluation of the proposed method. Finally, Section 5 presents some concluding remarks and future work to be done.

2 Background

Lui and Chan have shown the key role of finding a good assignment of clients (avatars) to servers, and they have proposed a partitioning method [10]. However, this and other existing partitioning methods do not take into account the non-linear behavior of DVE servers with the number of avatars they support. As shown in [11], the performance of a DVE system greatly decreases when any of the servers in the system reaches 100% of CPU utilization, showing a non linear behavior. Figure 1 shows several examples (different DVE configurations composed of 3,4 and 5 servers) of such behavior. The X-axis in this figure shows the number of avatars supported by the DVE systems, and the Y-axis shows the average response time measured in these DVE systems for several different simulations.

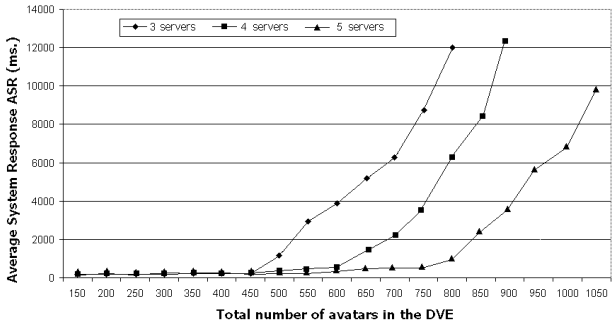

Figure 1. Average response times for different DVE configurations as the number of avatars in the system increases.

Figure 1 shows that the system response time remains constant as the number of avatar increases, until a certain number of avatars are connected to the system. From that point (that can vary from one to another DVE configuration) the response time of the system greatly increases as new avatars are added to the system. The saturation point is reached when *any* of the servers in the system reaches 100% of CPU utilization [11]. Therefore, the main purpose of any partitioning method should be keeping all servers in the system below the threshold of 100% of CPU utilization, regardless of the amount of inter-server messages. Since the method proposed in [10] and other existing partitioning methods do not take into account this fact, they cannot provide an actually efficient partitioning method for large DVE systems.

Recently, a new adaptive strategy that takes into account the non-linear behavior of DVE servers has been proposed for solving the partitioning problem [12] (this method is denoted as ARPS, for Adaptive Region Partitioning Strategy). In this proposal, all objects within each region of the virtual world are managed by a given server, and an adaptive load balancing scheme is provided. Nevertheless, this scheme provides only local load balancing among the servers managing adjacent regions. The local scope of the load balancing limits the performance of the partitioning method. When a group of servers managing adjacent regions are all of them heavily loaded, the exceeding workload proceeding from a saturated server can cause the target server to become saturated. If this situation occurs, the load balancing algorithm is started again. Since most of the

servers involved in the load balancing algorithm are close to saturation, the load balancing algorithm can be executed several times before it finds a target server that can accept the additional workload without entering saturation. This situation is known as the *load balancing cascading effect* [12], and it results in a poorer performance of the partitioning method when the DVE system is heavily loaded. As a result, this strategy only provides good performance if avatars move following a uniform pattern and they do not tend to concentrate in some regions of the virtual world.

In a previous work [13], we proposed a load balancing technique with global scope (ALB), where servers manage avatars instead of regions of the virtual world. Although this approach can increase the network traffic, this parameter does not have a significant effect on system performance while the system is kept below the saturation point [11]. Thus, this technique efficiently balances the workload generated by all avatars, regardless of both the movement pattern and also the initial distribution of avatars in the virtual world. It entirely assigns the exceeding workload that causes the saturation of a given server to the least loaded server in the system (coarse-grain assignment scheme). However, when the system works close to saturation, all servers are heavily loaded and the cascading effect can also arise, greatly decreasing the partitioning efficiency.

3 A Fine-Grain Partitioning Strategy

In order to improve the efficiency of the partitioning method proposed in [13], the load balancing cascading effect must be eliminated. However, this goal can only be achieved if a fine-grain assignment scheme is used. Although such strategy can increase the amount of inter-server messages (it can assign groups of avatars sharing the same AOI to several different servers), this parameter does not have a significant effect on system latency while the DVE system is kept below its saturation point, as shown in [11]. Therefore, a fine-grain load balancing technique can improve the DVE system throughput obtained with ALB technique without significantly increasing system latency.

We propose the *Fine-Grain Avatar Load Balancing (FGALB)* technique. As ALB technique does, FGALB technique starts with an initial assignment of avatars to servers. In order to obtain this initial assignment, any ad-hoc algorithm as well as an algorithm based on a heuristic search method can be used. As avatars freely move within the virtual world, both CPU utilization and also the number of avatars that each server supports are constantly monitored. From these measurements, the workload that each avatar adds to its server is statistically computed, taking into account the two factors shown in [14]: the movement rate and the *presence factor* of that avatar. Presence factor of avatar i is defined as the number of avatars in whose AOI avatar i appears. In this way, the estimated workload that each avatar represents to its server is periodically computed.

When a given server S_x in the system reaches 95% of CPU utilization, then FGALB algorithm is started. The purpose of FGALB algorithm is to select some avatars currently assigned to S_x and assign them to the servers in the system with the lowest CPU utilization, in such a way that the resulting estimated CPU utilization of S_x is reduced to 90%. Nevertheless, these threshold values (95% and 90%) can be different. They should be tuned according to the system latency, in order to avoid that any server in the system can reach 100% of CPU utilization (DVE saturation point, as shown in [11]) while FGALB technique is being computed and avatars are being re-assigned.

Unlike in ALB technique, in FGALB technique the exceeding avatars in S_x can be distributed among several servers (fine grain assignment scheme). Let S_{y_1}, S_{y_2},$\cdots S_{y_N}$ be a sorted list of the N servers in the system with the lowest CPU utilization (S_{y_1} is the server with the lowest CPU utilization, S_{y_2} is the server with the lowest CPU utilization if S_{y_1} is excepted, and so on). The first step of FGALB algorithm is to construct a list of candidate avatars in S_x to be transferred. Initially, this list contains all avatars currently assigned to S_x, sorted by their distance (in the virtual world) to the mass center of avatars currently assigned to S_{y_1}. We will denote this sorted list as *List of Candidate Avatars for server* S_{y_1} (LCA_1). This sorting criterion selects potential avatars to be re-assigned depending on the current state of avatars in the destination server. Thus, this criterion reduces the resulting amount of network traffic (inter-server messages).

The next step of FGALB algorithm consists of assigning the first avatars in LCA_1 to server S_{y_1}, either until the estimated workload of the re-assigned avatars represents at least 10% of CPU utilization in S_x, or until the estimated workload of S_{y_1} is higher than the estimated workload of S_{y_2}. In the former case, FGALB finishes (all the exceeding workload in S_x has been re-assigned). In the latter case, FGALB algorithm then computes LCA_2. This list contains all avatars currently assigned to S_x, but now they are sorted by their distance to the mass center of avatars currently assigned to S_{y_2}. The first avatars in LCA_2 are then re-assigned to S_{y_2}, either until the total amount of avatars moved from S_x to S_{y_1} and S_{y_2} represents at least 10% of CPU utilization in S_x, or until the estimated workload of S_{y_2} is higher than the estimated workload of S_{y_3}. Following this process, FGALB continues assigning avatars from S_x to the servers with the next higher CPU utilization, until the estimated workload of the re-assigned avatars represents at least 10% of CPU utilization in S_x.

In this way, FGALB technique takes into account the status of the target server before re-assigning avatars. If necessary, the exceeding workload of S_x is proportionally distributed among several servers, according to their relative CPU utilization. As a result, the cascading effect is eliminated and system throughput is increased, as shown in the next section.

4 Performance Evaluation

In this section, we present the performance evaluation of the load balancing method proposed in the previous section. We propose the evaluation of generic DVE systems by simulation. The evaluation methodology used is based on the main standards for modeling collaborative virtual environments, such as FIPA [15], DIS [16] and HLA [17]. We have used the same simulation tool used in [13], and we have evaluated the performance of the proposed technique.

In each simulation, all avatars sharing the same AOI must communicate between them for notifying both their position in the 3D virtual world and also any change in the state of the elements in that AOI. The message structure used for notifying avatar movements is the *Avatar Data Unit (ADU)* specified by DIS [16]. A simulation consists of each avatar performing 100 movements, at a rate of one movement every 2 seconds. Each time an avatar performs a movement, he (the client computer controlling that avatar) notifies that movement to his server by sending a message with a timestamp. That server must then notify that movement to all the avatars (to the corresponding client computers) in the same AOI of the sender avatar. When that notification arrives to these avatars, they return an ACK message to the sending avatar. In this way, clock skewing is avoided when computing system latency (the same clock is used for both the initial and final timestamp).

For comparison purposes, we have simulated the proposed FGALB method, the *Adaptive Region Partitioning Technique (ARP)* proposed in [12] and also the ALB technique proposed in [13]. We have simulated DVE systems with three different movement patterns of avatars: Changing Circular Pattern (CCP) [12], Hot-Points-ALL (HPA) [9] and also Hot-Point-Near (HPN) [18]. CCP considers that all avatars in the virtual world move circularly, starting and ending at the same location. HPA considers that there exists certain "hot points" where all avatars approach sooner or later. Finally, HPN also considers these hot-points, but only avatars located within a given radius of the hot-point approach these points. An iteration in a given movement pattern consists of all avatars in the system performing a single movement. A simulation consists of 100 iterations. Additionally, different initial distributions of avatars in the virtual world (uniform, skewed and clustered) have been considered, as in other studies [10, 12].

Figure 2 shows the final distribution of avatars in a 2-D virtual world when each movement pattern is applied and the initial distribution of avatars is a uniform distribution. In this figure, each avatar is represented by a grey dot.

Due to space limitations, we only show here the results for the experiments performed with HPN pattern. CCP moving hardly starts the load balancing algorithm if the initial partition is well-balanced, since all avatars follow a uniform circular path. The results for HP-ALL movement pattern are very similar to those presented here, except that HP-ALL pattern leads to system saturation in fewer iterations. This is due to the fact that more avatars tend to concentrate around the hot points in HP-ALL pattern. Additionally, the results for a clustered initial distribution are not shown either, since the results are very similar to the case of an initial skewed distribution of avatars.

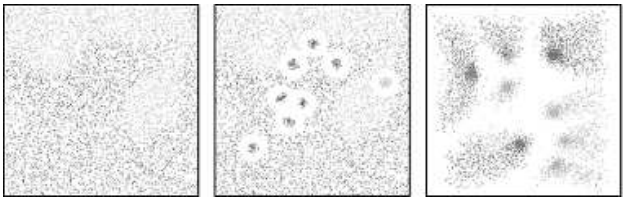

Figure 2. Final distribution of avatars for a) CCP, b) HPN, and c) HPA movement patterns applied to an uniform initial distribution of avatars.

We have tested a great number of different DVE configurations, ranging from small virtual worlds (composed of 3 servers and 180 avatars) to large virtual worlds (composed of 6000 avatars and 9 servers). Due to space limitations, we only present here the results for a representative (large) experiment.

Table 1 shows the performance evaluation results obtained for ARPS, ALB and FGALB techniques when an initial uniform distribution of avatars and HPA movement pattern are considered. This table shows the percentage of CPU utilization in each server after some of the iterations in the simulation. Each row shows the percentage of CPU utilization for a different server, and each column shows the percentage of CPU utilization in all servers after a given iteration for a given technique. Thus, labels Rx, Lx and Fx means CPU utilization after applying ARPS technique, ALB technique and FGALB technique in iteration x, respectively. Additionally, next to last row shows the sum of all the resulting standard deviations $\sum \delta_w$ with regard to the average estimated workload w that servers support. This value measures how balanced the DVE system is after each iteration (since all PC's used as servers are identical, the lower this value is the better balanced the workload is). In order to make a fair comparison, in all techniques we have obtained the initial assignment of avatars by applying the same algorithm (the one proposed in [12]). Finally, last row (M. a., moved avatars) shows the amount of avatars that each technique has re-assigned in that iteration in order to achieve the value in the upper row. This value provides an idea of the cost for applying each load balancing technique (re-assigning an avatar implies both some computational and some communication cost, the *partitioning efficiency*). A great value in this row means that some cascading effect has appeared.

Table 1 shows that the DVE system is kept below the saturation point during 15 iterations. The avoidance of the cascading effect allows FGALB technique to improve DVE system throughput, keeping the system below the saturation point during one iteration more than ALB technique, and during two iterations more than ARPS tech-

nique. Effectively, in iteration 13 ARPS technique (column R13) is not capable to keep the system away from saturation (servers $S3$, $S7$ and $S8$ reaches 100% of CPU utilization), and in iteration 14 ALB technique (column L14) cannot avoid that server $S0$ reaches 100% of CPU utilization. Great differences appear in terms of both standard deviations and moved avatars. In iteration 13 the number of avatars migrated by ARPS technique is one order of magnitude higher than the number of avatars migrated by two other techniques (last row, first three columns). The value of $\sum \delta_w$ provided by ARPS technique is about twice the values provided by ALB and FGALB techniques. In turn, in iteration 14 the amount of avatars re-assigned by ALB technique is two orders of magnitude higher than the number of avatars re-assigned by FGALB technique, showing that ALB technique cannot avoid the cascading effect. In iteration 14 the value of $\sum \delta_w$ provided by ALB technique is three times higher than the value provided by FGALB technique. Finally, in iteration 15 ALB technique finishes without providing a partition that avoids system saturation. The values shown in iteration 15 are an example of how uniformly FGALB technique distributes the workload among the servers. In this column, where half of the servers reach the saturation point at the same iteration, the least loaded server shows a CPU utilization of more than 98%.

	Method - Iteration					
	R13	**L13**	**F13**	**L14**	**F14**	**F15**
S0	97.6	89.4	92.6	100	94.9	99.4
S1	15.3	60.1	74.3	91.6	94.5	98.2
S2	86.1	84.6	68.2	87.5	94.1	100
S3	100	91.6	89.9	94.8	95.4	100
S4	75.6	78.1	71.5	94.5	97.5	99.6
S5	81.9	79.3	84.2	96.6	94.7	98.4
S6	90.3	64.7	72.1	94.8	93.4	100
S7	100	89.4	84.1	99.9	94.3	98.6
S8	100	89.5	89.7	94.5	95.2	100
$\sum \delta_w$	609	326	295	94	28	23
M. a.	1070	196	13	991	16	125

Table 1. Results for an initial uniform distribution of avatars and HPA movement pattern of avatars

Table 2 shows the results for an initial skewed distribution of avatars and an HP-near movement pattern of avatars. For this initial distribution of avatars the initial workload is higher than for the initial uniform distribution (the AOI of most of the avatars are crowded with a lot of avatars). As a result, DVE system is kept below the saturation point during few iterations than in the case of HPA movement pattern. Again, the proposed technique is able to improve system throughput, keeping the DVE system under the saturation point one iteration more than ALB technique and two iterations more than ARPS technique. FGALB technique re-assigns only a few avatars even when the system is very close to saturation (column F13), showing that this technique avoids the cascading effect.

	Method - Iteration					
	R11	**L11**	**F11**	**L12**	**F12**	**F13**
S0	36.3	48.0	54.3	96.4	97.2	100
S1	54.3	89.4	89.4	98.5	97.1	99.9
S2	68.0	67.7	46.5	96.5	96.5	100
S3	100	79.7	73.9	95.8	95.8	98.8
S4	45.9	44.4	51.7	95.0	97.6	99.6
S5	42.7	53.6	60.6	99.2	98.2	100
S6	32.8	43.6	53.7	97.5	99.2	98.9
S7	46.5	48.2	47.2	93.9	97.9	99.1
S8	100	61.5	49.4	99.0	97.6	100
$\sum \delta_w$	740	480	387	81	21	16
M. a.	43	21	18	201	22	41

Table 2. Results for an initial skewed distribution of avatars and HPN movement pattern of avatars

Finally, Table 3 shows the results for an initial uniform distribution of avatars and an HP-near moving pattern. In this case the overall workload generated by avatars does not exceed the total workload that the system can support, and it is where the considered techniques show the greatest differences. In this case the system throughput provided by the FGALB technique can manage the workload generated by avatars. However, the system throughput provided by the two other techniques is lower than the total workload generated by avatars, and they cannot avoid system saturation. Concretely, Table 3 shows that ARPS technique can only keep the system under saturation during 15 iterations. Again, servers managing the crowded regions reach 100% of CPU utilization (servers $S4$,$S5$, and $S8$ in column R15) while a server managing a distant region (server $S1$) shows around 13% of CPU utilization. On the contrary, ALB and FGALB techniques balance workload among all the servers, showing a very much lower value of $\sum \delta_w$ for the 15th iteration. On other hand, the amount of moved (re-assigned) avatars shows that ARPS technique does not avoid the cascading effect, and it needs almost 1000 re-assignments in order to provide a partition that saturates several servers. The same occurs with ALB technique in iteration 25. Although ALB technique is able to provide a well balanced partition (it provides a very low value of $\sum \delta_w$), it cannot avoid system saturation (server $S2$ reaches saturation) after performing more than 1700 reassignments. This situation is due to the coarse-grain strategy that ALB follows. Finally, FGALB technique allows to keep all servers under 100% of CPU utilization during the whole simulation, providing a well balanced partition in all iterations and also re-assigning very few avatars.

5 Conclusions and Future Work

In this paper, we have proposed a fine-grain load balancing method (FGALB) for solving the partitioning problem in DVE systems. Like a previously proposed method, this

	Method - Iteration					
	R15	**L15**	**F15**	**L25**	**F25**	**F100**
S0	65.1	89.5	89.4	98.8	92.4	94.0
S1	12.7	58.2	71.0	94.3	91.6	96.2
S2	79.8	89.4	70.1	100	93.0	92.4
S3	59.6	89.4	89.5	96.3	89.6	91.5
S4	100	74.9	71.2	94.6	92.4	96.5
S5	100	62.3	76.9	96.5	90.0	93.9
S6	28.5	65.2	71.3	94.8	91.9	93.7
S7	100	78.0	81.5	99.7	88.5	93.1
S8	100	89.4	89.4	94.6	93.1	93.5
$\sum \delta_w$	969	391	273	69	49	41
M. a.	987	154	26	1721	11	18

Table 3. Results for an initial uniform distribution of avatars and HPN movement pattern of avatars

adaptive technique balances the workload among all the servers in the system. However, FGALB takes into account the estimated state of the target server before re-assigning avatars. Thus, in this technique the exceeding workload that causes the saturation of a given server is proportionally distributed among several servers, if necessary. On one hand, this strategy avoids the cascading effect, providing well balanced partitions in a single iteration of the load balancing algorithm. On the other hand, it allows to increase DVE system throughput with few re-assignments of avatars.

The results show that the proposed technique can improve DVE system performance, regardless of both the movement pattern and also the initial distribution of avatars in the virtual world. When the system works in deep saturation, then the proposed technique is able to improve system throughput while re-assigning less avatars. Additionally, the proposed technique can significantly improve DVE system performance when the system works close to the saturation point. Therefore, we can conclude that the proposed technique can be used as a valid adaptive method that provides efficient solutions for solving the partitioning problem in DVE systems.

References

[1] S. Singhal and M. Zyda. *Networked Virtual Environments*. ACM Press, 1999.

[2] D.C. Miller and J.A. Thorpe. Simnet: The advent of simulator networking. *IEEE TPDS*, 13, 2002.

[3] J.M. Salles Dias, Ricardo Galli, and A. C. Almeida et al. mworld: A multiuser 3d virtual environment. *IEEE Computer Graphics*, 17(2), 1997.

[4] C. Bouras, D. Fotakis, and A. Philopoulos. A distributed virtual learning centre in cyberspace. In *Proc. of Int. Conf. on Virtual Systems and Multimedia (VSMM'98)*, 1998.

[5] M. Abrash. Quake's game engine: The big picture. *Dr. Dobb's Journal*, 1997.

[6] J. Smed, T. Kaukoranta, and H. Hakonen. A review on networking and multiplayer computer games. Technical report, Turku Centre for Computer Science. Tech Report 454., 2002.

[7] John C.S. Lui, M.F. Chan, and K.Y. Oldfield. Dynamic partitioning for a distributed virtual environment. Technical report, Department of Computer Science. Chinese University of Hong Kong, 1998.

[8] D.B. Anderson, J.W. Barrus, and J.H. Howard. Building multi-user interactive multimedia environments at merl. *IEEE Multimedia*, 2(4), 1995.

[9] F. C. Greenhalgh. Analysing movement and world transitions in virtual reality tele-conferencing. In *Proceedings of 5th European Conference on Computer Supported Cooperative Work (ECSCW'97)*, 1997.

[10] John C.S. Lui and M.F. Chan. An efficient partitioning algorithm for distributed virtual environment systems. *IEEE Trans. Parallel and Distributed Systems*, 13:193–211, 2002.

[11] P. Morillo, J.M. Orduna, M. Fernández, and J. Duato. On the characterization of distributed virtual environment systems. In *Euro-Par' 2003- Lecture Notes in Computer Science 2790*, pages 1190–1198. ACM, Springer-Verlag, 2003.

[12] N. Beatrice, S. Antonio, L. Rynson, and L. Frederick. A multiserver architecture for distributed virtual walkthrough. In *Proceedings of ACM VRST'02*, 2002.

[13] P. Morillo, J.M. Orduna, M. Fernández, and J. Duato. An adaptive load balancing technique for distributed virtual environment systems. In *Proc. of Intl. Conf. on Parallel and Distributed Computing and Systems (PDCS'03)*, pages 256–261. ACTA Press, 2003.

[14] P. Morillo, J.M. Orduna, M. Fernández, and J. Duato. On the characterization of avatars in distributed virtual worlds. In *EUROGRAPHICS' 2003*, pages 215–220. Eurographics Association, 2003.

[15] FIPA. *FIPA Agent Management Specification*, 2000.

[16] IEEE. *1278.1 IEEE Standard for Distributed Interactive Simulation-Application Protocols (ANSI)*, 1997.

[17] F. Kuhl, R. Weatherly, and J. Dahmann. *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Prentice-Hall PTR, 1999.

[18] M. Matijasevic, K. P. Valavanis, D. Gracanin, and I. Lovrek. Application of a multi-user distributed virtual environment framework to mobile robot teleoperation over the internet. *Machine Intelligence & Robotic Control*, 1(1):11–26, 1999.