AN ADAPTIVE LOAD BALANCING TECHNIQUE FOR DISTRIBUTED VIRTUAL ENVIRONMENT SYSTEMS [*]

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

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

ABSTRACT
One of the key issues in the design of scalable and cost-effective Distributed Virtual Environment (DVE) systems is the partitioning problem. This problem consists of efficiently assigning clients (3-D avatars) to the servers in the system, and some methods have been already proposed for solving it. However, only one of these methods takes into account the non-linear behavior of DVE servers with the number of avatars they support, and this method uses a load balancing technique of local scope. As a result, it only provides good performance if the movement pattern of avatars is uniform. In this paper, we propose an adaptive load balancing technique of global scope for solving the partitioning problem in DVE systems. The global scope of the proposed technique allows to avoid DVE saturation as long as possible. Evaluation results show that the proposed strategy can improve DVE system performance, regardless of both the movement patterns of avatars and also the initial distribution of avatars in the virtual world.

KEY WORDS
Distributed Virtual Environments, load balancing , dynamic partitioning

1 Introduction

Distributed Virtual Environment (DVE) systems have experienced a spectacular growth last years. These systems allow multiple users, working on different 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 if they were perceived by the user. Each user is represented in the shared virtual environment by an entity called *avatar*, whose state is controlled by the user input. Since DVE systems support visual interactions between multiple avatars, every change in each avatar must be propagated to the rest of the avatars in the shared virtual environment. DVE systems are currently used in many different applications [1], such as collaborative design [2], civil and military distributed training [3], e-learning [4] or multi-player games [5].

One of the key issues in the design of a scalable DVE system is the *partitioning problem* [6]. It consists of efficiently assigning the workload (avatars) among 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 that each server in the system supports, but also on the inter-server communications (and therefore on the network traffic). Some methods for solving the partitioning problem have been already proposed [1, 6]. 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. Therefore, these partitioning strategies cannot avoid the degradation of system performance caused by the assignment of too avatars to a given server in the system.

Recently, an adaptive strategy that takes into account the non-linear behavior of DVE servers has been proposed for solving the partitioning problem [7]. However, the scope of the load balancing technique used by this strategy 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). Therefore, load balancing is only performed among some of the servers in the system, leaving unbalanced the rest of the servers. As a result, this strategy only provides good performance under uniform movement patterns of avatars. The reason is that, as shown in our previous work [8], DVE systems enter saturation if *any* of the servers reaches 100% of CPU utilization. When some of the servers must continuously support a heavy load, local load balancing cannot use the rest of the servers for load balancing.

In this paper, we propose a global adaptive load balancing technique for solving the partitioning problem in DVE systems. This adaptive technique considers information from all the servers in the system and it also provides a global response of the system. This global response allows the proposed method to balance the workload generated by all of the avatars as much as possible. Evaluation results show that the proposed technique can improve DVE system performance for both uniform and non-uniform movement patterns of avatars. The results also show that the proposed strategy is able to keep the DVE system under its saturation

[*]Supported by the Spanish MCYT, grant TIC2000-1151-C07-04

point if the total workload generated by all avatars remains under the total workload that the DVE system can support, regardless of both the movement pattern 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 strategy for performing an efficient adaptive load balancing in DVE systems. Next, Section 4 presents the performance evaluation of the proposed strategy. Finally, Section 5 presents some concluding remarks and future work to be done.

2 Background

Architectures based on networked servers are becoming a de-facto standard for DVE systems [1, 6]. In these architectures, the control of the simulation relies on several interconnected servers. Multi-platform client computers are connected to one of these servers. 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 a message outburst. In this sense, concepts like areas of influence (AOI) [1], locales [9] or auras [10] have been proposed for limiting the number of neighboring avatars that a given avatar must communicate with.

Depending on their origin and destination avatars, messages in a DVE system can be intra-server or inter-server messages. As an example, Figure 1 shows an example of a DVE system consisting of several servers interconnected through a network. In this figure avatars are uniformly distributed, and they are represented as dots. Each server manage a given region of the virtual world, and it also has attached a given number of clients (avatars). This figure also shows an example of both intra-server and inter-server avatar updating messages. Inter-server messages are those messages whose origin and destination avatars are attached to different servers. In order to design a scalable DVE systems, the number of intra-server messages must be maximized. Effectively, when clients send intra-server messages they only concern a single server. Therefore, they are minimizing the computing, storage and communication requirements for maintaining a consistent state of the avatars in a DVE system.

Taking into account all this premises, Lui and Chan have shown the key role of finding a good assignment of clients to servers, and they have proposed a partitioning method [6]. However, this and other existing methods do not take into account the non-linear behavior of DVE servers with the number of avatars they support. As shown

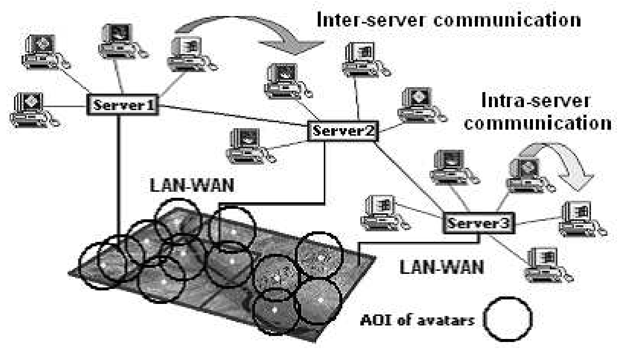

Figure 1. Multi-server DVE system

in [8], the performance of a DVE system greatly decreases when any of the servers in the system reaches 100% of CPU utilization. Therefore, the main purpose of any load balancing technique should be keeping all the servers in the system under 100% of CPU utilization, regardless of the amount of inter-server messages. Since these methods do not take into account this fact, they cannot provide an actually efficient partitioning method for large DVE systems. Recently, a new load balancing technique that takes into account the non-linear behavior of DVE servers has been proposed for solving the partitioning problem [7]. 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 reason for this locality is to reduce the number of inter-server messages (in this proposal, when an avatar scope crosses the boundary of two servers, both servers are responsible for transferring object models to that avatar). However, this local scope of the load balancing technique limits the performance of the partitioning method. Effectively, the performance evaluation of this proposal considers that all avatars in the virtual world move circularly, starting and ending at the same location. Under this simplifying assumption of a uniform movement pattern of avatars, local load balancing provides a good performance. However, is very unlikely that all avatars in a DVE system show a uniform movement pattern. DVE systems usually have certain “hot points” where avatars tend to head for [11]. In other cases, like 3-D networked games, these hot-points are game resources (energy, weapons, etc.) that dynamically appear and disappear, and only avatars located within a given radius of the hot-point tend to approach these points [12]. In these situations, all the neighboring regions around the hot points tend to get crowded with avatars, and therefore all the servers managing these regions tend to get saturated, while other servers support a low load. A global load balancing technique would use all the servers in the system, and it could provide better performance than a local load balancing technique for such cases.

3 A New Load Balancing Technique

As shown in [8], in order to provide a scalable and efficient adaptive partitioning method a load balancing technique should focus on keeping under 100% of CPU utilization all servers, rather than any other consideration. In order to achieve this goal as much as possible, the load balancing technique should provide a global scope, taking into account the workload assigned to all the servers in the system, and also considering any server in the system as a potential destination of avatars. Unlike in the load balancing technique propose in [7], we propose an object-oriented management of avatars. Servers manage the workload generated by objects (avatars), instead of regions of the virtual world. This approach allows to use any server in the DVE system for load balancing purposes when a given server reaches saturation. We propose the *Avatar Load Balancing (ALB)* technique. This proposal is based on a server-initiated load balancing technique developed for distributed operating systems [13]. ALB starts with an initial assignment of avatars to servers. In order to obtain this initial assignment, and ACS-based algorithm [14] or any other algorithm 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 [15]: the movement rate of that avatar and the *presence factor* (P_f) of that avatar. Presence factor of an avatar a_i is defined as the number of avatars in whose AOI avatar a_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 99% of CPU utilization, then ALB algorithm is started. The purpose of ALB algorithm is to select a number of avatars currently assigned to S_x and to assign them to the server with the lowest CPU utilization (denoted as S_y), in such a way that the resulting estimated CPU utilization of S_x is reduced to 90%. However, these threshold values should be tuned according to the network latency, in order to avoid that any server in the system can reach 100% of CPU utilization (DVE saturation point, as shown above) while ALB technique is being computed and avatars are being re-assigned.

The first step of ALB algorithm consists of sorting the avatars currently assigned to S_x by their presence factor P_f. Then, the first avatars in this ranking are assigned to S_y, until the estimated workload of the re-assigned avatars represents at least 10% of CPU utilization in S_x. The reason for using the criterion of the presence factor is that usually the avatars with the highest presence factor are also the closest avatars. Therefore, most of the messages generated by these re-assigned avatars will be intra-server messages. In this way, the proposed technique not only avoids the saturation of S_x (the main goal of the load balancing technique in order to maximize system throughput, as shown in [8]) but also reduces the amount of inter-server messages as possible (this feature helps to decrease system latency, as shown in [8]).

4 Performance Evaluation

In this section, we present the performance evaluation of the load balancing strategy 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 modelling collaborative virtual environments, FIPA [16], DIS [17] and HLA [18]. We have developed a simulation tool that models the behavior of a generic DVE system with a network-server architecture, and we have performed experimental studies to evaluate the performance of the proposed technique. As it was described in [8] for other purposes, this tool is composed by a set of multi-threaded servers. Each thread in a server uses blocking sockets for communicating with a client. Each client simulates the behavior of a single avatar, and it is also implemented as a multi-threaded application. One of the threads of the client manages the communication with the server it is assigned to, and another thread manages user information (current position, network latency, etc.).

Our simulator model is composed of a group of S interconnected servers and n avatars. Following the approach specified in FIPA and HLA standards, one of the servers acts as the main server (called *Agent Name Service* [16] or *Federation Manager* [18]) and manages the whole system. The main server also maintains a partitioning file for assigning a given server to each new avatar. In this way, once the network address and the port number where the main server is listening, avatars can join the simulation through this main server, that assigns each new avatar to one of the servers in the system. At this point, the new avatar must connect with the assigned server in order to start the simulation.

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. For the purpose of evaluation of the proposed technique, each client only simulates a given rate of avatar movements through the virtual world, and assumes that no changes are produced in any element of the AOI. This simplifying assumption reduces the system workload, but does not change the behavior of the system. The message structure used for notifying avatar movements is the *Avatar Data Unit (ADU)* specified by DIS [17]. 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 notifies that movement to his server by sending a message with a timestamp. That server must then notify that movement to all the avatars in the same AOI of the sender avatar. When that notification arrives to these avatars, they return an ACK message to the sending avatar.

For comparison purposes, we have simulated the proposed load balancing strategy and also the *Adaptive Region*

Partitioning Technique (ARPS) proposed in [7], the only existing adaptive load balancing technique proposed until now. We have simulated DVE systems with three different movement patterns of avatars: Changing Circular Pattern (CCP) [7], Hot-Points-ALL (HP-ALL) and also Hot-Point-Near (HP-Near). CCP considers that all the avatars in the virtual world move circularly, starting and ending at the same location. HP-ALL considers that there exists certain “hot points” where all avatars tend to head for sooner or later, as shown in [11]. Finally, HP-Near also considers these hot-points, but only avatars located within a given radius of the hot-point tend to approach these points [12]. An iteration in a given movement pattern consists of all avatars in the system performing a movement. Additionally, different starting distributions of avatars in the virtual world (uniform, skewed and clustered) have been considered, as in other studies [6, 7]. As an example, Figure 2 shows how avatars are located in a 2-D virtual world in each initial distributions of avatars. In this figure, the virtual world is a square and each avatar is represented as a dot.

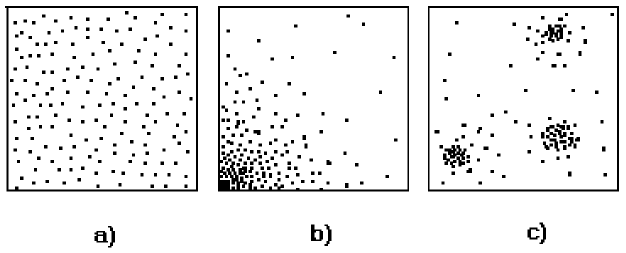

Figure 2. Different initial distributions of avatars in a DVE: (a) uniform, (b) skewed, and c) clustered

Figure 3 shows the final distributions of avatars in a 2-D virtual world when some movement patterns are applied and the initial distribution is a uniform distribution of avatars.

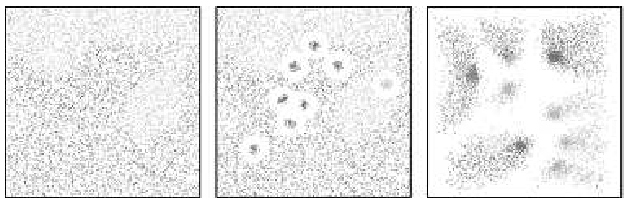

Figure 3. Final distribution of avatars for a) CCP, b) HP-Near, and c) HP-All movement patterns applied to an initial uniform distribution of avatars

However, we only present here the results for the experiments performed with HP-Near and HP-ALL patterns. The reasons are both space limitations and also that CCP movement pattern hardly starts the load balancing algorithm if the initial partition is well-balanced, since all avatars follow a uniform circular path. Additionally, the results for a clustered initial distribution are not presented either, since the results are very similar to the case of an initial skewed distribution of avatars.

The hardware platform used for simulating both clients and servers in the DVE system have been PC’s with processor Pentium IV at 1.7 GHz, with 256 Mbytes of RAM and with NVidia MX-400 graphic cards. Each server has been implemented in a single PC, while up to 50 clients have been allocated in the same PC. We have used a 10 Mbps Ethernet as the interconnection network. The simulation tool has been executed on a Windows 2000 Professional operating system.

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 large DVE configuration (the worst case).

Tables 1 and 2 shows the performance evaluation results obtained for ARPS and ALB load balancing techniques when a skewed distribution of avatars and a HP-ALL moving pattern are considered. This table shows the percentage of CPU utilization in each of the DVE servers after some of the iterations in the simulation. Each row show the CPU utilization for a different server, and each column show the CPU utilization after a given iteration for a given load balancing technique. In both strategies the load balancing technique is started in iteration 5. Additionally, last row shows the sum of the resulting standard deviations $\sum \delta_w$ for the average estimated workload w that DVE system servers support. This value measures how balanced the DVE system is after each iteration (the lower this value is, the better load balancing is achieved). In order to make a fair comparison, in both strategies we have obtained the initial assignment of avatars by applying the same initial algorithm (the one proposed in [7]). Therefore, the first columns in both tables show the same CPU utilization.

	ARPS (It. num.)			
	1	**4**	**5**	**6**
S0	30.1	28.1	31.7	30.7
S1	34.4	32.8	30.5	32.7
S2	41.6	46.6	53.6	81.6
S3	34.5	35.7	36.9	36.4
S4	32.5	32.7	34.4	54.1
S5	45.0	64.8	98.2	58.8
S6	38.5	42.0	45.1	58.4
S7	39.4	50.5	60.8	100.0
S8	24.9	23.7	21.7	20.6
$\sum \delta_w$	194	403	667	908

Table 1. Results for ARPS technique with an initial skewed distribution of avatars and HP-ALL pattern

Tables 1 and 2 show that ALB technique provides

	ALB (It. num.)					
	1	**4**	**5**	**6**	**7**	**8**
S0	30.1	28.1	33.7	66.4	99.6	100.0
S1	34.4	32.8	38.5	44.7	99.4	99.5
S2	41.6	46.6	55.9	58.2	98.4	98.7
S3	34.5	35.7	42.8	65.6	89.6	100.0
S4	32.5	32.7	39.3	40.3	94.6	100.0
S5	45.0	64.8	77.7	89.4	96.9	97.9
S6	38.5	42.0	50.4	55.6	99.8	94.5
S7	39.4	50.5	60.6	88.9	98.5	100.0
S8	24.9	23.7	28.4	30.6	91.5	100.0
$\sum \delta_w$	194	403	390	26	190	880

Table 2. Results for ALB technique with an initial skewed distribution of avatars and HP-ALL pattern

better load balancing than ARPS technique. ARPS technique (table 1) only keeps the system under saturation until the sixth iteration of the simulation, when server $S7$ reaches 100% of CPU utilization. However, at this point servers $S0$ and $S1$ are under 33% of CPU utilization. The column corresponding to the fifth iteration of ARPS also shows that server $S5$ reaches 98.2% of CPU utilization while servers $S0$ and $S1$ are under 33%. The reason for this behavior is the local scope of the load balancing technique. Under this movement pattern, all avatars tend to continuously concentrate in some points of the virtual world, and the servers managing these regions cannot share that workload with servers managing remote regions (with few avatars). Thus, this load balancing technique only achieves to keep the system under its saturation point until iteration 6, where server $S7$ reaches saturation. On the contrary, ALB technique (table 2) is able to share the workload among all the servers, and thus after iterations 5 and 6 it manages to decrease the value of $\sum \delta_w$. This technique achieves to keep the system away from saturation in iterations 6 and 7, although in iteration 7 all servers are around 90% of CPU utilization. The system only reaches saturation (iteration 8) when the overall workload reaches the maximum workload that the system can support (as avatar concentrate, the number of avatars in other avatar's AOI increases, and therefore the workload that each avatar adds also increases).

Tables 3 and 4 shows the results for a initial uniform distribution of avatars and a HP-near pattern. Since in this pattern only part of the avatars tend to concentrate around the hot-points, the total workload generated by all the avatars remains under the limit of the total workload that the DVE system can support. In this case, ARPS technique (Table 3) can only keep the system under saturation during 15 iterations. Again, servers managing the crowded regions reach 100% of CPU utilization, while servers managing distant regions are around 10% of CPU utilization. On the contrary, ALB technique (Table 3) balances workload among all the servers, showing a very much lower value of $\sum \delta_w$ for the first 15 iterations. As a result, this technique is able to keep all servers under 100% of CPU utilization during the whole simulation.

	ARPS (It. num.)					
	1	**11**	**12**	**13**	**14**	**15**
S0	9.0	66.2	93.9	54.2	74.3	65.1
S1	9.1	9.0	8.8	8.5	8.5	12.7
S2	8.8	26.9	33.3	44.4	55.8	79.8
S3	9.8	50.6	64.4	88.7	97.2	59.6
S4	9.0	15.4	16.0	76.3	94.6	100.0
S5	11.3	35.5	42.4	60.0	74.2	100.0
S6	9.5	14.0	18.0	15.8	26.9	28.5
S7	9.2	37.5	53.0	69.5	91.0	100.0
S8	8.6	54.7	65.1	78.1	98.7	100.0
$\sum \delta_w$	22	643	897	865	1030	1615

Table 3. Results for ARPS technique with an initial uniform distribution of avatars and HP-Near pattern

	ALB (It. num.)							
	1	**12**	**13**	**14**	**15**	**16**	**25**	**100**
S0	9.0	66.2	93.9	89.4	98.4	97.7	98.4	91.6
S1	9.1	9.0	8.8	38.9	89.5	96.5	87.5	92.6
S2	8.8	26.9	33.3	45.7	87.6	95.4	97.5	89.9
S3	9.8	50.6	64.4	81.1	74.1	89.6	89.6	98.6
S4	9.0	15.4	16.0	53.6	65.2	88.5	90.0	97.5
S5	11.3	35.5	42.4	52.9	89.6	94.6	90.4	90.2
S6	9.5	14.0	18.0	22.5	95.2	97.6	97.5	97.2
S7	9.2	37.5	53.0	65.9	96.6	96.9	94.1	94.3
S8	8.6	54.7	65.1	79.6	97.1	98.1	77.5	92.9
$\sum \delta_w$	22	643	897	214	74	89	61	40

Table 4. Results for ALB technique with an initial uniform distribution of avatars and HP-Near pattern

5 Conclusions and Future Work

In this paper, we have proposed a global adaptive load balancing technique for solving the partitioning problem in DVE systems. Unlike the only existing technique that takes into account the non-linear behavior of DVE servers with the number of avatars they support, in this approach servers manage the workload generated by avatars, not regions of the virtual world. The global scope of the proposed technique allows to balance the workload generated by all of the avatars as much as possible. We have evaluated the proposed technique by simulation. The results show that the proposed strategy can improve DVE system performance, particularly for non-uniform movement patterns of avatars. Moreover, the proposed strategy allows to keep the DVE system under the saturation point if the total workload generated by all of the avatars remains under the total workload that the DVE system can support, regardless of both the movement pattern and also the initial distribution of avatars in the virtual world.

As a future work to be done, we plan to develop a partitioning strategy capable of providing certain quality of service. If the workload is high and the DVE system is close to saturation, then the purpose of the partitioning strategy may be maximizing the DVE throughput, in order to support the highest number of clients as possible. However, if the DVE system has a low workload, then the purpose of the partitioning strategy may be offering certain quality of service (for example, improving latency for those clients with the lowest communication bandwidth).

References

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

[2] J.M.Salles Dias, Ricardo Galli, A. C. Almeida et al, mWorld: A Multiuser 3D Virtual Environment, in *IEEE Computer Graphics*, Vol. 17, No. 2, March-April 1997.

[3] D.C.Miller, J.A. Thorpe, SIMNET: The advent of simulator networking , in *Proceedings of the IEEE*, 83(8), pp. 1114-1123. August, 1995.

[4] Tohei Nitta, Kazuhiro Fujita, Sachio Cono, An Application Of Distributed Virtual Environment To Foreign Language, *Proceedings of FIE'2000. IEEE Education Society.* Kansas City, Missouri, October 2000.

[5] Michael Lewis and Jeffrey Jacboson, Game Engines in Scientific Research , in *Communications of the ACM*, Vol 45. No.1, January 2002.

[6] Jonh C.S. Lui, M.F. Chan, An Efficient Partitioning Algorithm for Distributed Virtual Environment Systems, *IEEE Trans. Parallel and Distributed Systems*, Vol. 13, March 2002

[7] N. Beatrice, S. Antonio, L. Rynson, L. Frederick, A Multi-server Architecture for Distributed Virtual Walkthrough , in *Proceedings of ACM Symposium on Virtual Reality, Software and Technology 2002(ACM VRST'02)*. Hong-Kong, China, June 2002.

[8] P. Morillo, J.M. Orduña, J. Duato, On the Characterization of Distributed Virtual Environment Systems, *Proceedings of European Conference on Parallel Processing (Euro-Par' 2003)*, Klagenfurt, Austria. August, 2003.

[9] D.B.Anderson, J.W.Barrus, J.H.Howard, Building multi-user interactive multimedia environments at MERL, in *IEEE Multimedia*, 2(4), pp.77-82, Winter 1995.

[10] J.C.Hu, I.Pyarali, D.C.Schmidt, Measuring the Impact of Event Dispatching and Concurrency Models on Web Server Performance Over High-Speed Networks , *Proc. of the 2nd. IEEE Global Internet Conference*, November.1997.

[11] C. Greenhalgh, Analysing Movement and World Transitions in virtual Reality Tele-conferencing, in *5th European Conference on Computer Supported Cooperative Work (ECSCW'97)*, Lancaster, U.K., September 1997.

[12] M. Matijasevic, K. P. Valavanis, D. Gracanin, I. Lovrek, Application of a Multi-User Distributed Virtual Environment Framework to Mobile Robot Teleoperation over the Internet , in *Machine Intelligence & Robotic Control*, Vol. 1, No. 1, pp. 11-26 (1999).

[13] R.Chow and T.Jhonson, *Distributed Operating Systems & Algorithms* (Addison Wesley-Longman, 1997).

[14] P. Morillo, M. Fernńdez, J.M. Orduña An ACS-Based Partitioning Method for Distributed Virtual Environment Systems, *Proc. of 2003 Int. Parallel and Distributed Processing Symposium Workshops (IPDPS' 2003)*, Nice, France. April, 2003.

[15] P. Morillo, M. Fernńdez, J. M. Orduña, On the Characterization of Avatars in Distributed Virtual Worlds, in *Annual Conference of the European Association for Computer Graphics (EUROGRAPHICS' 2003)*, Granada, Spain. September, 2003. (See http://informatica.uv.es/grupos/ri/publicac.htm)

[16] FIPA Agent Management Specification. Foundation for Intelligent Physical Agents, 2000. Available at `http://www.fipa.org/specs/fipa00023/`

[17] DIS. 1278.1 IEEE Standard for Distributed Interactive Simulation-Application Protocols (ANSI). DMSO. DoD High Level Architecture. 1997.

[18] F. Kuhl, R. Weatherly, J. Dahmann, *Creating Computer Simulation Systems: An Introduction to the High Level Architecture* (Prentice-Hall PTR, Upper Saddle River, NJ, 1999).