Performance Evaluation of a Dynamic Approach for Networked Servers Distributed Virtual Environments

Christos Bouras[1, 2], Eri Giannaka[1, 2], Thrasyvoulos Tsiatsos[3]
[1]Research Academic Computer Technology Institute, Greece
[2]Computer Engineering and Informatics Department, University of Patras, Greece
[3]Department of Informatics, Aristotle University of Thessaloniki, Greece
bouras@cti.gr, giannaka@cti.gr, tsiatsos@csd.auth.gr

Keywords: Distributed virtual environments, networked servers architecture, performance evaluation, resource management, load balancing

Abstract

Distributed Virtual Environment systems simulate the behaviour and activities of a great number of concurrent users interacting in a virtual world over a wide area network. The advances both in computer technology and networking infrastructures in combination to the advanced applications and services developed, expanded the popularity of DVEs and promoted their use in a wide range of areas, such as learning and training, collaborative work, military applications and multiplayer online games. The sizes of the virtual worlds simulated and the tremendous number of users that DVEs are called to support require additional bandwidth and computational resources. For handling these growing requirements a lot of work has been done both to the direction of alternative architectural solutions as well as to techniques and algorithms for handling and overcoming the limitations of these environments. However, recent research has shown that one of the major limitations of networked servers DVE systems is scalability. To this direction this paper presents on the one hand a dynamic approach for DVEs, which exploits the nature of these systems for the optimal resource management and extended scalability support and on the other hand, evaluates its performance through a series of experiments and under various settings of the virtual world, using Simul8 simulation tool.

1. INTRODUCTION

Distributed Virtual Environments (DVEs) simulate real or imaginary worlds by incorporating rich media and graphics for achieving high level of realism and for providing all required functionality and services to the end users, such as text and audio chat, streaming media support, application sharing, etc. Users are represented in the DVE system by an entity, called avatar, which is controlled by the user's computer (client) and has the ability to navigate within the virtual world and interact both with the other users of the system as well as with the system itself. A basic principle of all DVE systems is that all connected users, which could be scattered around the globe, share the same view of the virtual environment, as if they were physically located in the virtual space. Thus, it is of critical importance that consistency among users' view is constantly and efficiently maintained.

DVEs became notably popular during the last decade, which could be credited to the important and continuous advances of both hardware and software, but mainly to the wide expansion of high-speed internet access, which constitutes the basic medium of these systems. Thereby, many applications and platforms were developed for supporting large-scale DVEs, which in turn, were gradually adopted in a wide range of both research and industrial areas, such as learning and training, collaborative work, military applications and entertainment. It should be mentioned that the term *large-scale* is two-fold and refers both to the size of the virtual environment (in terms of rich graphics and media) as well as to the tremendous number of users it is called to support [1]. The characteristics of these complex systems introduce a number of issues that need to be addressed, such as resource management, control of network traffic, awareness, heterogeneity and reliability [2].

For handling these demanding applications, facing the issues that arise and overcoming various limitations, existing approaches fall usually into one of the following architectures: a) networked servers architectures and b) peer-to-peer architectures. To both directions they is lot of work done, including algorithms for the partitioning problem [3], [4], load balancing techniques [5], awareness methods [6] and other techniques [7], [8], while a great number of platforms has been designed and developed ([9][9], [10], [11]). Even though, many of the algorithms and techniques implemented can achieve effective results, current research has indicated that the main issue for networked servers DVEs is the scalability. In particular, as presented in [4], DVE systems reach saturation when any of the servers reaches 100% of CPU utilization. In these cases, the overall system performance greatly decreases.

To this direction, this paper presents a dynamic approach for networked servers DVEs for extending the scalability and for achieving an optimal resource management. In particular, the approach presented exploits the dynamic nature of these systems for selecting and assigning over time the resources necessary for the efficient operation of the system. Thus, the main objectives of the approach are, on the one hand, to ensure that the system will not reach the saturation point, for a longer period of time and, on the other hand, to optimally assign the necessary resources according to the specific needs of the DVE for a given period. Furthermore, the paper evaluates the performance of the proposed dynamic approach through a series of experiments and under different settings of the virtual world. At this point it should be noted, that the approach presented does not aim at guaranteeing that a saturation point will never be reached, but rather at ensuring that the system will be viable for a longer period of time.

The rest of the paper is structured as follows: Section 2 presents the main characteristics and processes of dynamic DVE system and describes the main principles of the dynamic DVE approach in terms of the basic concepts and the parameters measured. Section 3 describes the simulation model in terms of its basic entities as well as their "translation" to Simul8 simulation tool objects. Section 4 presents the experiments conducted for testing the model's accuracy and efficiency under different settings of the DVE system and discusses the results. Finally, Section 6 concludes the paper and presents some planned next steps.

2. DYNAMIC APPROACH FOR DVE SYSTEMS

This section presents, in brief, the basic entities and characteristics of DVE systems and focuses on the description of the dynamic approach in terms of its basic concepts and primitives as well as of the workflow for the processes and events that take place.

2.1. DVEs' basic entities and characteristics

A virtual environment could be considered as a simulation generated by a computer, which can simulate either an imaginary or real world. In DVEs the simulated world runs not on one computer system but on several, which are connected over a network. Users that connect to these systems are able to interact in real time, sharing the same virtual world. DVEs aim at supporting a tremendous number of concurrent users, scattered around the globe. The participants constitute active parts of the DVE, usually represented by human-like entities, called avatars for enhancing the awareness [12]. The state and behaviour of each avatar is controlled by the user through the client computer. Connected users can view the virtual world on their computer (client), thus having their own local copy of the virtual world.

Apart from the users, DVEs, on their vast majority, are also comprised by non-autonomous entities, called objects, which constitute the graphical representation of entities that are placed within the virtual world for supporting the context of each simulated scenario (e.g. trees, books, chairs, weapons, etc). According to the nature of the DVE and the scenario it simulates, these objects could be static or moving, interactive or not.

In the majority of existing DVE systems users have the ability to navigate in the virtual world, thus changing their position coordinates, interact with the objects of the virtual environment, thus changing some of their attributes (such as location, shape, colour, etc), interact and communicate with other participating users. For achieving high-sense of realism and maintaining consistency, it is of critical importance that all connected users are always aware both of the presence of other users as well as of any actions performed.

In networked servers DVEs, when a user connects to the system, s/he is assigned to one of the available servers. This assignment is based on the algorithm and techniques that each DVE adopts for handling its resources and for achieving load balancing among the servers. Throughout the users' presence in the system, the responsible server accepts the messages produced by all avatars that it handles, processes these messages and updates the state of the virtual world accordingly. Then, it sends those changes to all avatars concerned, thus modifying and synchronizing their view of the virtual world. In addition, for maintaining consistency, synchronization messages are sent among the connected servers. These messages introduce an additional cost, usually called communication cost.

The majority of existing techniques uses all servers available for handling and supporting the DVE system. The term dynamic DVE, as used in this paper, is twofold. On the one hand it refers to virtual environments, which are initialized once and run always, with users joining and leaving the virtual space. At this point it should be mentioned that this definition does not take into account the cases of virtual environments, where users can dynamically add or create objects within the virtual space. On the other hand, the term "dynamic" is adopted for defining DVEs, where the number of servers running changes and adapts to the resources needed at a given period of time.

2.2. Basic Concepts

Taking into account the results of [4], already mentioned in the previous section, the dynamic scheme defines certain boundaries within which an application can operate efficiently. The efficiency of each application depends on the focus of each simulated scenario. For example, in a case of an educational DVE, the consistency of the world would not be importantly affected if a number of position messages were lost, while in a battlefield, where soldiers move and run, any loss would significantly affect the sense of realism and users' awareness. Thus, the boundaries defined for the DVE system are scenario driven.

Furthermore, as mentioned above, the communication cost among the connected server can importantly increase the load and performance of the system. In the cases where all available servers are used, regardless of the techniques adopted, there are cases, where the communication cost is importantly increased. In particular, in the cases of multiple servers available, and a low number of connected users to each server, the servers would still need to exchange communication messages thus overloading the system. Driven by this observation, the proposed approach defines the concept of *necessary* resources for maintaining the consistency of the system, while minimizing the communication cost. In particular, the resources of the system (which are the servers available) are relished according to the demands generated by connected users.

From the above, it could be stated that a networked servers DVE system needs to locate the optimum solution for the following problem:

"Given a certain number of servers, with defined processing power, we need to find the optimal assignment of resources for serving as many users as possible with guaranteed efficiency, while minimizing the communication cost, based on each application's special characteristics".

This problem falls in the area of operational management and linear algebra, where the efficiency is defined by the boundaries set for the CPU utilization each of the available servers.

2.3. Dynamic Scheme Presentation

We consider a DVE system, which is comprised by a fixed number of servers (computers). Each computer has a certain amount of resources and can serve requests/messages from the users. We consider as resources the capacity of the Central Processing Unit (CPU). Each time that a user enters the DVE system a request is sent to a central server, which is denoted as ConMan Server. This server performs two main tasks: (a) accepts and authorizes the connection requests from the users and redirects them to the appropriate server, (b) monitors the performance of the working servers over a period of time and acts when it is identified that one or more servers need to be unloaded. For the monitoring task, the ConMan Server performs network management operations using the SNMP [13] protocol. At this point it should be mentioned that the ConMan Server communicates with the users only during the connection process and the assignment of the user to the appropriate server.

When users' avatars enter the virtual world, and from the moment they are assigned to a server, they start to navigate, interact and perform actions, thus sending messages/requests to the servers. Each of the servers of the DVE system is responsible for processing and serving these requests/messages initialized by the users, perform all the necessary updates to the virtual scene and notify all concerned users about the updates. The events that take place and their processing, based on their number and resources they require, affect the servers' performance. Therefore, the performance of each server is constantly checked (after a fixed period of time) by the ConMan Server. However, for ensuring higher reliability, each server has a self-monitoring mechanism. In particular, each server runs an SNMP agent, which monitors the CPU usage. When the defined threshold/boundary is reached, the SNMP agent sends a "trap" to the ConMan Server, which notifies it that the specific machine reaches the point of saturation. When the trap message is received by the ConMan Server, the unloading process of the saturated server is initialized.

In particular, when a "trap" is received, either within the standard checks of the system or by the active servers, the ConMan Server performs all the necessary actions for re-balancing existing workload among the servers of the system. It should be mentioned that the technique for workload re-balance is DVE specific.

```
//Checking Server Status
FOR (i=1, i<num_servers, i++)
   IF (cpu_i > cpu_max)
     Send SNMP trap
     Mark server_i->overloaded
   //First Check Among Idle Servers the candidate
   to activate
      FOR (j=i+1; j<num_servers; j++;)
        IF (server_j IS idle)
          SET server_j->active
          Move avatars & workload server_i->server_j
          MODIFY routing scheme
   //If all servers are active find among them a
   candidate for undertaking additional workload
      FOR (j=i+1; j<num_servers; j++;)
       IF (cpu_j <= cpu_capable)
         Move avatars & workload server_i->server_j
         MODIFY routing scheme
```

Table 1: Activation mechanism of idle servers

Upon the receipt of the trap message, ConMan Server first checks among the list of available servers to trace the idle ones. The first idle server traced is activated and workload is balanced among the two servers. If all available servers are active, ConMan performs a check for tracing candidate servers for undertaking part of the closely saturated server. An active server is considered to being able to handle additional workload when it has resources available. For defining this ability of an active server, the approach sets another threshold, denoted as CPU capable (cpu_capable). When the utilization of a server falls below this threshold and stays there for a defined period of time, the Server is considered as candidate for undertaking additional workload.

As mentioned above, in a DVE system, users enter and leave the environment. When the number of users is decreased, or in cases that the connected users are not very active, in terms of low request rate, then the CPU utilization on one or more of the connected servers could be importantly decreased. However, given the fact that this server is active increases the communication cost of the system, despite of the fact that it is being underused. This under-usage of a server is triggered by the proposed framework with the definition of another boundary, denoted as minimum CPU threshold (cpu_min). When the utilization of a server falls below this threshold and stays there for a defined period of time, the ConMan Server performs a process for deactivating this server.

Table 2: Deactivation mechanism of active servers

```
//Checking Server Status
FOR (i=1, i<num_servers, i++)
//If a server's utilization falls below the
minimum threshold then set this server as
underused
 IF ($cpu_i$ < cpu_min)
    Mark $server_i$->underused
//Check Among Active Servers to find a Candidate
    FOR (j=1; j<num_servers; j++; && j!=i)
     IF ($cpu_j$  <=  cpu_capable)
      Move all avatar and workload $server_i$->$server_j$
     Set $server_i$->idle
      Modify routing scheme
```

The deactivation of the server can be completed if and only if at least one of the rest active servers is able to process the additional workload, which will be assigned to it by the under-used one, which means that at least one of the active servers' CPU utilization must be below the cpu_capable threshold. The deactivation process is presented in the pseudo code of Table 2.

2.4. Parameters

This sub-section presents the main parameters used for setting up the performance and assessment results of the DVE system.

CPU_Utilization(t): the actual CPU usage of the server at time t. This parameter is used for indicating the state of a server. This utilization parameter represents the CPU usage of each server and is calculated as follows:

$$U_i = \frac{B_i}{T} \quad (1)$$

where B_i is the busy time of server *i* over a T time interval. The busy time B_i is defined as follows:

$$B_i = req\_served \times s_i\_process\_time \quad (2)$$

with req_served the number of requests served by server *i* and $s_i\_process\_time$ the time for processing each request.

Cpu_max: the maximum value used for indicating that a server tends to be overloaded and the SNMP "trap" is sent to the ConMan Server.

Cpu_capable: the value that indicates whether an already active server can accept additional workload from another either nearly saturated or underused server.

Cpu_min: the value which indicates that a server could be considered as under-used and its workload and tasks could be assigned to another already working server.

Routing technique: this parameter defines the way that workload will be balanced among the servers of the system. This parameter is strongly related to the partitioning and load balancing approach adopted by each type of application and should be carefully implemented for valid results. Some of the most common techniques implemented in DVEs are the following:

- Circular: avatars are forwarded in a circular way to the available servers of the system
- Equal Probability: avatars are forwarded with an equal probability profile to the servers of the system
- Spatial: in cases where each server manages certain part of the virtual world, avatars, according to their initial position would need to be forwarded to the appropriate server, which handled the corresponding partition.

System performance Check: this parameter defines the time interval that is used by the ConMan Server for checking the servers' status.

3. SIMULATION MODEL

As mentioned above, the approach presented faces the problem stated as an operational management one. To this direction, for testing the framework's efficiency and for studying the workflow process of the networked servers DVE, a discrete event simulation model is developed using SIMUL8 (version 12). SIMUL8 [14] is an integrated environment for working with simulation models and is one of the most popular and widely used tools, both in the industrial and academic area of operational management. It has a powerful language and visualization capabilities that allow the creation of accurate, flexible, and detailed simulations in a reasonable time. It also has several features (trials, warm-up period, random sampling, etc) allowing one to conduct statistical analysis of the simulation output.

The simulation model is developed to provide insights into the workflow process and to estimate the system performance measures. The simulation model is made of several interconnected simulation objects (input node, queues, and work centres). These objects as well as the simulation parameters are described in the sub-sections that follow.

3.1. Simul8 Objects

This section presents briefly some of the basic objects that Simul8 provides for designing and creating models whose behaviour can be simulated.

Work Centres: A Work Centre is a place where work takes place on Work Items. Work done at work centres usually takes up time and sometimes requires the availability of resources.

Storage Bins (Queues): A storage bin is a place where work to be done can wait until appropriate resources or work centres are available.

Work Entry Points: A work entry point is a place where work to be done appears in the model for the first time.

Work Exit Points: A Work Exit Point is a place where work that is complete (or otherwise "finished") leaves the model.

Work Items: A Work Item is the work which is done in the organization being simulated. Work Items flow through the simulation, being stored in Storage Areas, and acted upon by work Centres.

Components: Components consist of one or more existing objects (either the standards or other Components) that are tailored in some way then saved as a single new object for future use.

3.2. DVE Simulation Entities

This section presents the main entities of the simulation model. At this point it should be mentioned that the entities of the DVE system are mapped to the simulation objects provided by Simul8 tool, as presented in Table 3.

Table 3: Mapping of DVE entities to Simul8 objects

DVE Entity	Simul8 Objects
Avatar	Work Item
Virtual World Entry Point	Work Entry Point
Avatar Messages	Work Item
ConMan Server	Component
DVE Servers	Components
Inter-Server Messages	Work Items

Virtual World Entry Point: This entity corresponds to the Work Entry Point of Simul8. For the DVEs simulation, the Work Entry Point represents the point, where users' avatars enter the system. Each Entry Point is characterized by the distribution used for initializing and sending the messages as well as the inter-arrival times between messages.

Messages: In the DVE simulation model there are three types of messages taken into account and are presented as Work Items. The first type is called "avatar", the second one is called "request" and the third one "synchronization". The avatar processes represent the actual avatars that enter the system, which means that for each avatar there is an avatar message initialized. The request process represents the messages sent by each avatar and are labelled with their parent id (that is the avatar id which sent the message). For simplification purposes, we consider that all messages sent by the users' avatar are of the same type and require the same resources. This simplification is mainly related to the fact that in the simulation model we consider a general approach of a DVE system, where message types (e.g. position messages, object modification messages, chat messages) and attributes are not distinguished. However, it should be mentioned, that the simulation tool provides the necessary functionality for tailoring the messages exchanged by using additional labels attached to each message type. Finally, the synchronization messages represent the messages exchanged among the servers of the system for maintaining consistency and awareness.

ConMan Entity: The ConMan Server entity used in the DVE model is a combination of a Work Centre and a storage bin object (of Simul8 library). The ConMan Server is connected to the Virtual World Entry Point and the messages that arrive, which represent avatars first move to the ConMan Queue and then to the ConMan Work Centre, where they are processed. The processing implies that ConMan first labels the messages with a unique identifier and sets the life of each of these messages.

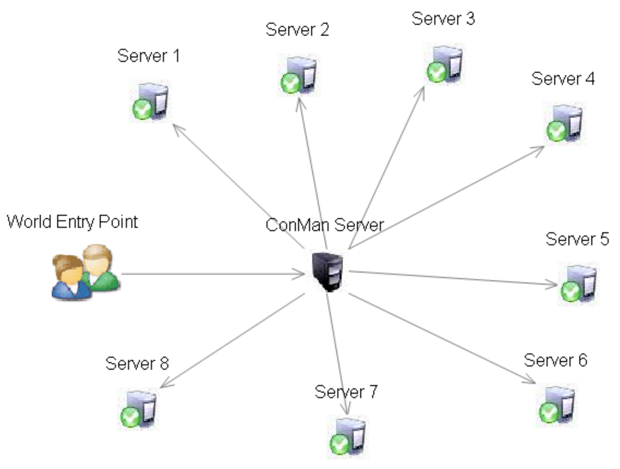

Figure 1: Dynamic DVE Simulation Model

Server Entity: The Server entity used in the DVE model is a Component of Simul8, and constitutes a combination of three Queues and Work Centres and two Work Exit Points. Each of these objects serves different tasks within the model. Work done at work centres usually takes up time and sometimes requires the availability of resources. For the DVE simulation, we consider that the time it takes for processing each message depends on the server's processing capabilities and this parameter could be adjusted by the system designers based on the infrastructure they plan to use.

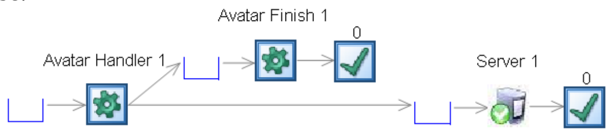

Figure 2: Simulation Model Server Entity

The simulation model for the DVE system is presented in Figure 1, while Figure 2 depicts the server entity, in terms of its functional blocks.

4. EXPERIMENTS

This section describes the experiments conducted for evaluating the behaviour and performance of the dynamic architectural DVE model under different setups of the virtual environments. In particular, for testing the proposed approach, a series of experiments were realized, where different values for critical parameters were taken into account.

Furthermore, the behaviour and performance of the proposed dynamic approach were also compared to the linear partitioning approach [5], which provides good results for large-scale DVEs. In particular, the linear approach first divides the virtual environment to the available servers of the system using a Divide and Conquer technique. The step that follows is that every defined period of time an algorithm checks for the workload on the servers and performs all the necessary re-assignments of entities so that a nearly balanced workload can be achieved. The last step of the approach encounters the exchange of some avatars among the servers so as to minimize the communication cost. For measuring the communication cost in the experiments conducted, some of the avatars that entered the virtual world were labelled as 'border" ones. For these avatars, the number of messages sent, should be communicated to the appropriate servers of the system so that consistency could be maintained.

4.1. Scenarios' Setup

The experiments conducted consider a DVE system comprised by 8 available servers. All of the servers available have the same computing power and are dedicated in serving the DVE system requests (that is that no other programs or applications are running). The general parameters for the two approaches are presented in Table 4.

Table 4: General Parameters

Dynamic Approach Parameters	
CPU Max Threshold	80%
CPU Capable Threshold	40%
CPU Min Threshold	10%
CPU check time (SNMP)	1sec
Routing Technique	Spatial
Linear Approach Parameters	
Rebalancing Time	5 min
General Parameters	
Servers Number	8
Average avatar life	50 min
Experiments' Time	180 min
Avatar Enter Distribution	Exponential

Two of the defining parameters for the scalability and effectiveness of the system's behaviour is the rate that users join the virtual world as well as their activity profile. By activity profile, we refer to the rate of movements and interactions that users perform in a virtual environment. For evaluating the systems' behaviour and for assessing the impact of both avatars' incoming rate and activity profile, different virtual worlds' setups were selected and the corresponding scenarios were designed and tested. These scenarios are presented in Table 5.

Table 5: Scenarios' setups

User Inter-arrival Times	User Requests' Rate
9 sec	0.5 sec
	0.2 sec
	0.1 sec
6 sec	0.5 sec
	0.2 sec
	0.1 sec
3 sec	0.5 sec
	0.2 sec
	0.1 sec

At this point it should be mentioned that for each of the scenarios presented, multiple runs where executed and the results presented in the section that follows correspond to the average values obtained from these runs.

4.2. Results

From the experiments conducted it could be said that both the tested approaches behave well in all three scenarios examined. In particular, both of them achieve a relative balanced workload among participating servers, while none of them reaches the 100% of CPU usage, which would downgrade the overall system's performance.

4.2.1. Communication Cost

From all the experiments conducted, it became obvious that one of the major advantages of the dynamic approach, when compared to the linear one, was the tremendous reduction of the communication cost among the participating servers.

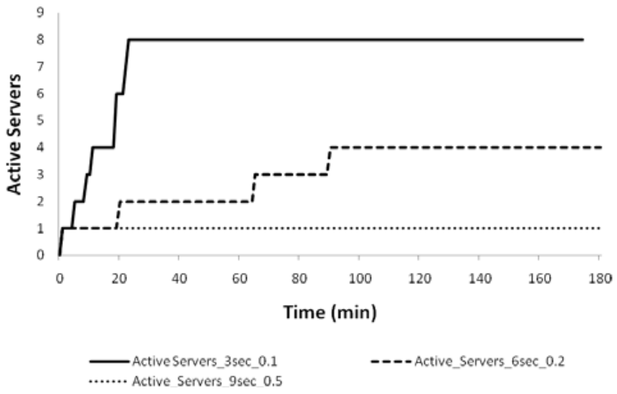

Figure 3: Number of active servers over time for the dynamic approach

In particular, in all cases examined the communication cost for the dynamic approach was significantly lower, which can be easily explained by the fact that in most of the cases the number of servers, which needed to be updated, was smaller, as it can be seen in Figure 3. Also, in the dynamic approach presented, when a new server needs to be activated, the communication cost that border avatars introduce is limited between the parent server (that is the server which reached the CPU max threshold) and the child server (that is the one that undertook workload from the parent server).

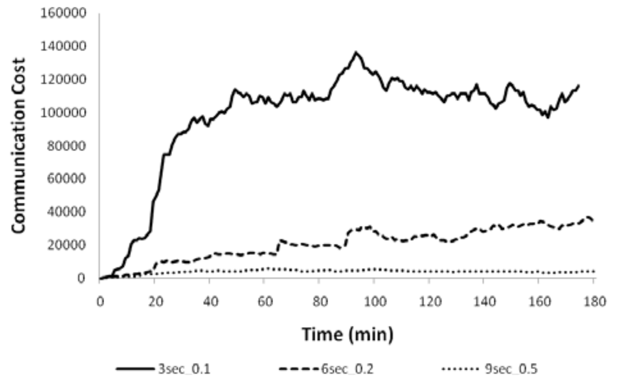

Figure 4: Communication Cost over time for the linear approach

Thus, even for the most demanding case, which is the one that avatars enter the virtual environment every 3 sec and send requests every 0.1 sec, the communication cost of the dynamic approach is lower than the one of the linear approach, even though the number of servers used is the same.

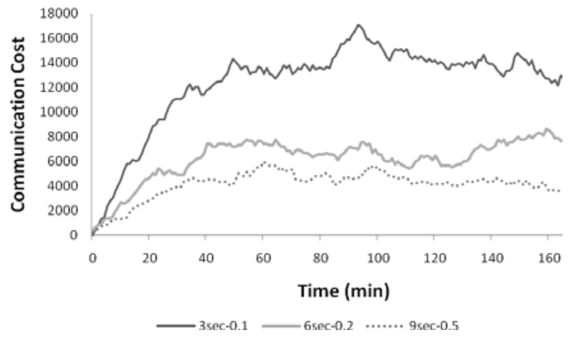

Figure 5: Communication cost over time for the dynamic approach

Figure 4 and Figure 5 present the communication cost for the linear and the dynamic approach respectively, for the most demanding (3sec avatar inter-arrival time and 0.1sec requests inter-arrival time), an average (6sec avatar inter-arrival time and 0.2sec requests inter-arrival time) and an optimistic (6sec avatar inter-arrival time and 0.5sec requests inter-arrival time) scenario.

4.2.2. CPU Usage

As mentioned above, in all experiments conducted, for both approaches, the CPU usage in all of the servers did not reach the saturation point of 100% utilization. The results of the CPU utilization of all connected servers for all cases examined are presented in Table 6. Columns 1 and 2 of the table present the users' inter-arrival time and requests' rate respectively, while the rest of the columns present the CPU utilization for the servers of the DVE system.

Table 6: Servers' CPU usage under different virtual world setups

Dynamic Approach									
Scenarios		**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
9 s	**0.5**	40							
	0.2	52	48						
	0.1	55	46	42	59				
6 s	**0.5**	60							
	0.2	38	39.5	36	36				
	0.1	41	39	39	35	31.5	30	44	40
3 s	**0.5**	62	54.8						
	0.2	42	36.5	35.5	36	30.5	38.5	37.5	35
	0.1	85	74	71	69	73	71	71	71
Linear Approach									
9 s	**0.5**	5.6	4.4	5.6	5	5.2	4.8	5	5.4
	0.2	14	11	14	12	12.5	12	12.5	13.5
	0.1	28	25	28	26	21	23	27	26
6 s	**0.5**	6.8	7.2	8	6.9	7.4	7.4	7.6	9
	0.2	17	18.5	21	18	17.5	19	18.5	21
	0.1	33	36	37	39	39	38	37	43
3 s	**0.5**	17	14.2	13.2	15	14	11.6	17	15.2
	0.2	41	36.5	33.5	36	37	29	43	37.5
	0.1	75	71	74	72	77	71	75	72

For the less demanding scenarios (9sec inter-arrival times) the linear approach presents very low CPU utilization for all connected servers.

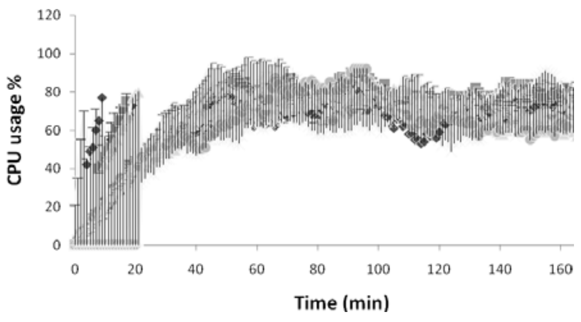

Figure 6: CPU usage zone

When compared to the results of the dynamic approach, where one to four servers are used, it could be observed that the activation of all servers available not only leads to their under-usage but also introduces unnecessary communication cost.

Furthermore, for the most demanding cases of 3sec with high request rate, the dynamic approach achieves a relatively balanced workload among servers, which even if it exceeds in one of the cases the 80% threshold, it does not lead to the saturation of the system. From the results presented in Figure 6, it can be observed that the CPU usage zone is within the boundaries set for preventing the DVE system from reaching a saturation point, thus affecting the overall performance.

4.2.3. Additional Observations

From the results extracted by the experiments conducted, it could be noticed that even though avatars' incoming rate is of high importance for the system's behaviour, it is the avatars' activity profile, which is more indicative of how demanding an application could be. Thus, application designers and developers based on the scenario that each virtual world simulates, could select the techniques and algorithms that fit best to the predicted workload of their applications.

5. CONCLUSION AND FUTURE WORK

This paper presented a dynamic approach for handing DVE systems, which exploits the nature of these demanding applications for optimal resource management and extended scalability support and evaluated its performance under various different settings of the virtual world. The basic concept of the approach lies in finding an optimal resource assignment, which is driven from the application requirements, as they change over time.

For illustrating the effectiveness of the proposed approach experiments were carried out and were compared to a linear approach, which achieves good results for DVEs. The experiments were conducted under various settings of the virtual world using Simul8 simulation tool. The results showed that the proposed approach achieves a notable decrement to the communication cost among the servers of the system, while even in highly demanding cases achieves a balanced workload among the servers, without reaching the saturation point of 100% of CPU utilization for any of them.

Finally, some of the planned next steps include the simulation and evaluation of additional scenarios, the experimentation with other existing techniques, used widely for DVE systems as well as the assessment of the effect that different parameters and factors have on DVE systems.

6. REFERENCES

[1] C. Bouras, E. Giannaka, T. Tsiatsos, "Exploiting Virtual Objects Attributes and Avatars Behavior in DVEs Partitioning", The 17th International Conference on Artificial Reality and Telexistence - ICAT 2007, Esbjerg, Denmark, 28 - 30 November 2007, pp. 157 – 163

[2] N. Pryce: Group Management and Quality of Service Adaptation in Distributed Virtual Environments, 4th UK VR-SIG Conference, Brunel University, Uxbridge, UK, (November 1997).A.B. Smith, C.D. Jones, and E.F. Roberts, "Article Title", Journal, Publisher, Location, Date, pp. 1-10

[3] C. Bouras, E. Giannaka, T. Tsiatsos, "Partitioning of Distributed Virtual Environments Based on Objects' Attributes", 11th IEEE International Symposium on Distributed Simulation and Real Time Applications, Chania, Crete, Greece, , 22 - 24 October 2007, pp. 72 – 75

[4] P. Morillo, J.M. Orduña, M. Fernández and J. Duato, "Improving the performance of Distributed Virtual Environment Systems", in IEEE Transactions on Parallel and Distributed Systems (TPDS), volume 16, number 7, pp. 637-649, July 2005. IEEE Computer Society Press, 2005

[5] 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

[6] P. Morillo, S. Rueda, J.M. Orduna and J.Duato, "A Latency-Aware Partitioning Method for Distributed Virtual Environment Systems" , in IEEE Transactions on Parallel and Distributed Systems (TPDS), volume 18, number 9, pp. 1215-1226, September 2007. IEEE Computer Society Press, 2007

[7] T.A. Funkhouser, "Network Topologies for Scalable Multi-User Virtual Environments", IEEE VRAIS '96, San Jose, CA, April, 1996

[8] Macedonia, Michael R., Zyda, Michael J., Pratt, David R., Brutzman, Donald P., Barham P. T. "Exploiting Reality with Multicast Groups," IEEE Computer Graphics & Applications, September 1995, pp.38-45

[9] N. Beatrice, S. Antonio, L. Rynson, and L. Frederick. A multiserver architecture for distributed virtual walkthrough, In Proceedings of ACM VRST'02, pages 163–170, 2002

[10] Anarchy Online: http://www.anarchy-online.com

[11] Everquest: http://everquest.station.sony.com/

[12] Joslin, Pandzic & Thalmann, "Trends in networked collaborative virtual environments", Computer Communications, Volume 26, Number 5, 20 March 2003 , pp. 430-437

[13] SNMP v2: http://tools.ietf.org/html/rfc1908

[14] Simul8 Simulation Software: http:// www.simul8.com