The Quality of Service Issue in Virtual Environments

Pedro Morillo
Universidad de Valencia, Spain

Juan Manuel Orduña
Universidad de Valencia, Spain

Marcos Fernandez
Universidad de Valencia, Spain

INTRODUCTION

Networked virtual environments (NVEs) have become a major trend in distributed computing, mainly due to the enormous popularity of multi-player online games in the entertainment industry. Nowadays, NVE systems are considered as the supporting technology for many networked and virtual organizations (NVO) (Singhal & Zyda, 1999), especially to those classified within the field of computer supported cooperative work (CSCW), where networked computer can be seen as a standard to provide the technological means to support the team design (Ott & Nastansky, 1997). These highly interactive systems simulate a virtual world where multiple users share the same scenario. The system renders the images of the virtual world that each user would see 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. Hundreds and even thousands of client computers can be simultaneously connected to the NVE system through different networks, and even through the Internet. NVE systems are currently used in many different applications (Singhal & Zyda, 1999) such as civil and military distributed training (Miller & Thorpe, 1995), collaborative design (Salles, Galli, Almeida et al., 1997) and e-learning (Bouras, Fotakis, & Philopoulos, 1998). Nevertheless, the most extended example of NVE systems are commercial multi-player online game (MOG) environments. These systems use the same simulation techniques that NVE systems do, and they are predicted to make up over 25 percent of local area network (LAN) traffic by 2010 (McCreary & Claffy, 2000).

Although centralized server (also client-server) architectures or peer-to-peer (P2P) architectures were also proposed for NVE systems (Singhal & Zyda, 1999), architectures based on multiple servers are becoming a de-facto standard for this type of distributed systems (Greenhalgh, Bullock, Frecon et al., 2001; Lui & Chan, 2002; Steed & Abou-Haidar, 2003). The reason for this trend is that on one hand, architectures based on a single, centralized server are not scalable with the number of connected clients (particularly for the most extended application of NVE systems, multi-player on-line games). On the other hand, NVE systems based on peer-to-peer architectures seems to be scalable enough, but they must still efficiently solve the awareness problem. This problem consists of ensuring that each *avatar* is aware of all the *avatar*s in its neighborhood (Smith, Hixon, & Horan, 2001).

In multiple server architectures, servers must contain the current status of different 3D models, perform positional updates of *avatars* and transfer control information among different clients. Thus, each new *avatar* represents an increase in both the computational requirements of the application and 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, different approaches have been proposed in order to limit the number of surrounding *avatars* that a given *avatar* must communicate with. Concepts like areas of influence (AOI) (Singhal & Zyda, 1999; Zou, Ammar, & Diot, 2001), locales (Anderson, Barrus, Howard et al., 1995) or auras (Greenhalgh et al., 2001) define a neighborhood area for *avatars*, in such a way that a given *avatar* must notify its movements (by sending a message) only to those *avatars* located in its neighborhood. Thus, the AOI of a given *avatar* determines the amount of network traffic generated by that *avatar*. Other approaches use three tiered architectures (Abrams, Watsen & Zyda, 1997; Lee & Lee, 2003) data filtering (Trefftz, Marsic, & Zyda, 2003) or distributed cache management (Capps, 2000) in order to minimize the impact of network traffic on the performance of the NVE systems.

Users in distributed systems usually perceive system performance either as the system *latency* (time interval required by the system in order to process each information unit) or as the system response time (Blommers, 1996; Duato, Yalamanchili, & Ni, 1997). Nevertheless, users in a NVE system perceive system performance not only as system *latency*, but also as system *throughput*. System *throughput* can be defined as the maximum number of users that the system can simultaneously support. For example, users in a multiplayer game environment want not only that the system quickly respond to their movements, but also they want that the system allow them to simultaneously play with the greatest number of players (users) as possible. In order to design actually efficient NVE systems, this special feature must be taken into account.

THE QOS PROBLEM IN NVE SYSTEMS

The term *QoS* can be applied to any system, meaning that the system not only supports a given client but also it fulfills some specific requirements (potentially in different dimensions) of that client, regardless of eventual system bottlenecks. For example, in network environments the term *QoS* means the ability of some networks to conform to some specific user requirements of *latency*, jitter delays, traffic peaks, etc, regardless of the current network traffic (Alfaro, Sanchez, Orozco, & Duato, 2003). System bottlenecks arise because the system is designed with a given capacity in its resources, and the workload that the system supports (in computers, the amount of information to be processed in a given period of time) eventually exceeds this capacity.

The problem of providing *QoS* to clients in a NVE system has been already described and some strategies have been proposed (Bernier, 2001; Choukair, Retailleau, & Hellstrom, 2000; Faisstnauer, Schmalstieg, & Purgathofer, 2000). One of these approaches (Bernier, 2001) uses *latency* compensating methods in order to repair the effects of high network jitter. A different approach consists of modifying the resolution of the 3D models depending on the connection speed of the client computer (Choukair et al., 2000). Both strategies try to provide *QoS* to *avatars* in exchange for reducing quality of graphics (either image resolution (Choukair et al., 2000) or quality of animation (Bernier, 2001). Bernier (2001) tries to provide good performance in a given dimension (sometimes at the cost of decreasing the performance in another dimension) in spite of potential system bottlenecks (connection speed, network jitter, number of clients connected to the system). However, none of these strategies takes into account the non-linear behavior of NVE systems with the number of *avatars* assigned to each server, as described in Morillo et al. (2005). Therefore, they do not care about guaranteeing a maximum *latency* to clients and they cannot guarantee that the system will not become saturated, greatly degrading the *latency* provided to all clients regardless of the connection speed or the network jitter.

The *QoS* problem can be expressed in NVE systems as *latency* constraints. In order to fulfill these constraints, and taking into account the non-linear behavior of these systems described in Henderson and Bhatti (2003), a trade-off among server saturation, clients' interactivity and system stability must be reached. A NVE system can only offer *QoS* to clients if it is working under its saturation point and at the same time the average round-trip delay for the messages sent by each *avatar* (denoted s ASR, for average system response) is lower than 250 ms. (Henderson & Bhatti, 2003). However, the ASR provided to a given *avatar* a_i depends on where *avatars* located in the AOI of a_i are assigned to. If *avatar* a_i is assigned to server S_X then the ASR for *avatar* a_i linearly decreases with the number of *avatars* in the AOI of a_i that are *migrated* from other servers to server S_X. Therefore, the problem of offering *QoS* to *avatars* can be expressed as a new partitioning problem. A partitioning method that provides *QoS* to *avatars* will have to maximize the number of neighbor *avatars* assigned to the same server and at the same time it will have to keep the system away from saturation. Additionally, since this strategy is a global load balancing scheme it must not migrate more than 30 percent of *avatars* in the system (Lee & Lee, 2003). Therefore, the partitioning problem will consists of finding a partition complying with all these three requirements. In order to solve this partitioning problem, a quality function that takes into account all these requirements is proposed.