Future Generation Computer Systems 29 (2013) 1561–1572

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Flexible load distribution for hybrid distributed virtual environments

Emanuele Carlini[b,a,*], Laura Ricci[c,a], Massimo Coppola[a]

[a] Institute of Information Science and Technologies CNR-ISTI, Pisa, Italy
[b] Institute for Advanced Studies Lucca IMT, Lucca, Italy
[c] Department of Computer Science, University of Pisa, Pisa, Italy

ARTICLE INFO

Article history:
Received 6 February 2012
Received in revised form
12 August 2012
Accepted 3 September 2012
Available online 8 September 2012

Keywords:
Cloud computing
Peer-to-Peer
Virtual environment
Load balancing
Distributed hash table

ABSTRACT

This paper proposes an architecture for Distributed Virtual Environments (DVEs) integrating cloud and peer nodes. We define the overall structure of the architecture and propose a flexible strategy to distribute the load due to the management of the entities of the DVE. The proposed approach takes into account the utilisation of the nodes, the economical cost due to their use of bandwidth, and their probability of failure. A greedy heuristics is exploited to reduce the computational cost of the algorithm in order to maintain a fair interactivity level to the end user of the DVE. A mobility model generating realistic Second Life traces is exploited to evaluate our algorithm. The experimental results show the effectiveness of our approach.

1. Introduction

Distributed Virtual Environments (DVEs), like massively multiplayer games [1] or distributed simulations [2], have acquired lots of popularity in the last few years from both commercial enterprises and research communities. Currently, most DVEs rely on a centralised architecture which supports a straightforward management of the main functionalities of the DVE, such as user login, state management, synchronisation between players and billing. However, as the number of simultaneous users keeps growing, centralised architectures show their scalability limitations. To overcome these limitations, server clusters have to be bought and operated to withstand service peaks, also balancing computational and electrical power constraints. However, a cluster-based centralised architecture concentrates all communication bandwidth at one data centre, requiring the static provisioning of large bandwidth capability. Further, a large static provisioning leaves the DVE operators with unused resources when the load on the platform is not at its peak.

Cloud Computing [3] allows us to solve the aforementioned scalability and hardware ownership problems because of on-demand resource provisioning [4,5]. The possibility of renting machines lifts the DVE operators from the burden of buying and maintaining hardware, whereas it offers the illusion of infinite machines, with good effects on scalability. Also, the pay-per-use model adheres to the seasonal access pattern of the DVE (e.g. more users in weekends than in the middle of the week). However, Cloud Computing may still be costly for platform operators. Besides server time, bandwidth cost represents a major expense when operating a DVE [6]. Thus, even if an approach based exclusively on the cloud approach is feasible, its cost might be still too high for DVE operators.

On the other hand, many research efforts have been done to design pure Peer-to-Peer (P2P) infrastructures for DVEs [7–9]. These are inherently scalable, and may reduce the load on centralised servers by exploiting the capacity of the peers. Furthermore, if a peer fails, the network is able to self-repair and reorganise, so providing robustness to the DVE; network traffic is distributed among the users involved. Furthermore, these properties pair with little costs for the DVE operators. However, P2P infrastructures require mechanisms to ensure the persistence of the game state. When users leave the system they remove both their resources and the data they have managed so far, unduly stressing the self-repair features of the system. The lack of a central authority makes it complicated to enforce security and soundness of updates to the system state at any time. Moreover, user machines typically have strict and heterogeneous constraints on computational power and network capability, which makes them complex to be exploited.

These drawbacks are incentives to combine these two orthogonal approaches: Cloud (or on-demand) computing and P2P infrastructures. In [10] we have discussed the requirements and the

[*] Corresponding author at: Institute of Information Science and Technologies CNR-ISTI, Pisa, Italy.

E-mail addresses: emanuele.carlini@isti.cnr.it (E. Carlini), ricci@di.unipi.it (L. Ricci), massimo.coppola@isti.cnr.it (M. Coppola).

0167-739X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2012.09.004

issues for such a kind of architecture. We have also presented the initial design of a hybrid and flexible architecture combining the advantages of the two approaches. In this paper we continue the work initiated in [10] by completing and refining the proposed architecture. In particular, we separate the functionality of neighbours discovery (i.e. to find the entities of the DVE relevant for a player) from the management of their state. This has led us to the definition of two distinct distributed components of the architecture, the *localisation component*, which manages only the positions of the DVE entities, and the *shared access component*, which manages their state. Both components are distributed, and typical P2P techniques are used to define the overlay structure, i.e. Distributed Hash Tables [11].

The definition of two distinct components allows the minimisation of the data transfers between nodes due to the dynamic movement of the DVE entities. The data transfer involves only the *localisation component*, while the whole state of each entity is permanently mapped to a node of the *shared access component*. This avoids a continuous transfer of the whole object state between different nodes, which can jeopardise the smoothness of the DVE. Furthermore, the definition of two different components allows us to achieve a better optimisation for each of them. The presence of hotspots in the DVE makes the load of the localisation component inherently unbalanced. Advanced spatial data retrieval techniques taking into account both data locality and load balancing in hotspots may be leveraged. On the other hand, even if uniform hashing techniques may be exploited to map the entities to the nodes of the *shared access component*, load unbalancing may still occur due to the heterogeneity of the entities of the DVE. For instance, popular objects that are accessed by a huge amount of players. In this case the node which manages the popular objects is heavily loaded because of both the rate of state updates and the resolutions of conflict updates.

In this paper we focus on massively multiplayer games applications, since these are currently the most widespread DVE applications. This work describes the overall P2P/Cloud architecture proposed and in particular it focuses on the shared access component. We propose a flexible load distribution algorithm for the *shared access component* taking into account the main characteristics of the hybrid architecture. The assignment of tasks to nodes takes into account the computational power of the nodes, i.e. heavy tasks are assigned to cloud nodes, while lighter ones to peer nodes. In order to minimise the economical effort for the game operator, the algorithm considers also the economical cost of the bandwidth consumed for access to the cloud nodes. Finally the failure probabilities of the nodes are taken into account as well, in order to avoid charging a user's peer with too much sensible information.

The algorithm has been evaluated through extensive simulations, by taking advantage of realistic DVE workloads. The results show that the proposed algorithm greatly outperforms a solution where no load balancing strategy is exploited. In particular, our results show that our load distribution policy becomes useful when the system approaches its peak load.

The rest of the paper is structured as follows. In Section 2 we provide an overview of the current hybrid DVE architectures and of the load balancing strategies for DVE architectures. Section 4 describes our architecture, whereas Section 5 presents the mechanisms and the policies for load distribution. Section 6 describes the workload used to evaluate our algorithms and Section 7 discusses the experimental results. Finally, Section 8 concludes the paper.

2. Related work

Several hybrid architectures and load distribution mechanisms for DVEs have been proposed during the last decade. In the rest of this section, we collect and compare the approaches that, to the best of our knowledge, are more relevant with respect to our proposal.

2.1. Hybrid architectures for DVE

Hybrid architectures aim to exploit and combine user resources (i.e. referred as *peers* in the rest of this section) and centralised servers. A widely-used method is to divide the *Virtual Environment* (VE) into *regions* or *cells*, whose dimensions can be either fixed or variable. These regions are in turn assigned to a peer or a server, that becomes the *manager* of the entities in that region. Hybrid architectures follow mostly two different approaches: (i) a region can be assigned to either a peer or to a server without any restriction, or (ii) only a subset of the cells can be assigned to peers.

The work proposed in [12] belongs to the first category. The authors consider square cells, that are initially managed by a central server. The first peer with enough computational and bandwidth capabilities to enter a cell becomes the cell manager. Afterwards, a fixed number of peers that enter the same cell act as backup managers in order to increase failure robustness. Similarly, [13] proposes an hybrid system, including a central server and a pool of peers. The central server runs the DVE and, as soon as it reaches the maximum of its capacity, it delegates part of the load to the peers.

The same authors of [13] propose in [14] an approach belonging to the second category. A central server executes the main game whereas peers run *auxiliary games* which are typical of certain game genres, such as Massively Multiplayer Online Role-Playing Games (MMORPGs). They are separated instances of the DVE, shared only by a fixed (and usually not high) number of players. In a similar way, [15] proposes a functional partition of the DVE tasks. Central servers operate user authentication, game persistence and manage regions characterised by high-density user interactions, whereas peers support only low-density interaction regions. The authors of [16] provide an interesting distinction between *positional* and *state-changing* actions. They propose a hybrid architecture where peers manage positional actions, that are more frequent and prone to be maintained locally. Central servers handle state-changing actions, that are not transitory and require a larger amount of computational power.

The idea of distinguishing positional and state-changing actions is in fact an interesting idea which we have exploited in the design of our architecture. However, rather than assigning different actions to different types of nodes, we define two different and independent distributed structures, i.e. two Distributed Hash Tables, that manage, respectively, positional and state-changing actions. The management of the nodes of each DHT can be assigned to a peer or to a cloud node.

In other words, we exploit an intermediate approach. On the one hand, some functionalities, like authentication, must be handled by centralised and full controllable servers. On the other hand, other functionalities may be mapped to central servers or to peers. This requires a complete dynamic strategy allowing for more flexibility in load distribution, which allows a fine-grained management of the resources by the DVE operator. Resources control is very important for our approach, since the seamless combination of cloud and P2P requires us to keep under control the cost and to effectively deal with the implicit uncertainty related to peers. Therefore, a basic issue for the exploitation of hybrid architectures is the definition of effective load distribution mechanisms. The next section reviews the main approaches in this area.

2.2. Load distribution in DVE architectures

The management of avatars and passive objects constitutes the typical computational and bandwidth load of a DVE. Avatars move across the DVE and interact with each other. This interaction can be *direct* or *indirect*. In the former case, an avatar directly modifies the state of another one, while the latter case is that