Comparison of Static and Dynamic Resource Allocations for Massively Multiplayer Online Games on Unreliable Resources

Radu Prodan[1], Alexandru Iosup[2], and Cristian Bologa[3]

[1] Institute of Computer Science, University of Innsbruck, Austria
[2] Parallel and Distributed Systems, Delft University of Technology, Netherlands
[3] Babeş-Bolyai University, Cluj-Napoca, Romania

Abstract. We investigate the use of a new Massively Multiplayer Online Gaming (MMOG) ecosystem consisting of end-users, game providers, game operators, and Cloud resource providers, for autonomous, self-adaptive hosting and operation of MMOGs on unreliable resources. For this purpose, we developed an MMOG simulator compliant with our MMOG ecosystem in which we inject traces collected from a real-world MMOG and resource characteristics from 16 Cloud providers. Using our simulator, we study the impact on the involved actors by considering different resource availability levels, and highlight the advantages of dynamic resource allocation over the static overprovisioning with respect to two types of metrics: QoS offered to the clients and financial profit of game providers and operators.

1 Introduction

Massively Multiplayer Online Games (MMOGs) are a new type of large-scale distributed applications characterised by a real-time virtual world entertaining millions of players spread across the globe. MMOG are designed as a collection of networked game servers, concurrently accessed by a large number of players (or end-users). The players connect directly to one game server and are mapped to one avatar in the game world to whom they send their play actions and receive appropriate responses. Based on the actions sent, the avatar dynamically interacts with other avatars within a game session, influencing each others' state. The state update responses must be delivered within a given time frequency to ensure an adequate *Quality of Service (QoS)* for a smooth and responsive experience. The load of the game server is proportional to the number of interactions between entities. An overloaded game server delivers state updates to its clients at a lower frequency than required which makes the overall environment fragmented and unplayable.

To concurrently support millions of active players and many other server-driven entities (non-playing characters and other game objects) generating variable computational and latency-sensitive resource demands, the MMOG operators purchase and overprovision a multi-server infrastructure with sufficient

L. Lopes et al. (Eds.): Euro-Par 2014 Workshops, Part I, LNCS 8805, pp. 299–310, 2014.

computational, network and storage capabilities for guaranteeing the QoS requirements and a smooth gameplay at all times. This statically provisioned infrastructure has two major drawbacks: it has high operational costs and is vulnerable to capacity shortages in case of unexpected increases in demand.

To address these gaps, we investigated in previous research [3,7,8] the use of Cloud computing technology for on-demand provisioning of virtualised resources to MMOGs based on their actual variable load. For this purpose, we designed a new ecosystem for MMOG operation and provisioning [8] consisting of four actors with smaller and better focused roles that facilitate their market penetration and chances of business success: clients, game providers, game operators, and resource (Cloud) providers. The interaction between them is negotiated and regulated through bipartite Service Level Agreements (SLA), representing wrappers around QoS parameters which they agree to deliver (see Figure 1).

End-users join MMOG sessions offered by game providers on the basis of MMOG subscriptions, representing contracts between the two parties comprising terms such as the price to be paid by the end-users and the QoS gameplay guarantees (i.e. state update rate) to be delivered by the provider.

Game providers offer a selection of MMOGs by contracting new games from development companies. Based on clients' requests, game providers assign clients to game zones delegated to game operators for QoS-based execution. The quality of gameplay is monitored and in case of *SLA faults* (e.g. state update rate below minimum threshold), the client is compensated.

Game operators receive requests from game providers for operating different MMOG session zones with guaranteed QoS. Based on resource utilisation estimations (covered in [7]), the game operators construct *operation SLA* offers negotiated with game providers [8], and allocate resources accordingly (i.e. start new zones, allow client connections). To fulfill these agreements, game operators acquire resources by establishing *resource SLAs* with Cloud providers [3]. At predefined *measurement timesteps* during the gameplay, the operators analyse the QoS information from the MMOG servers and, whenever SLA faults are detected, they compensate the game providers.

Resource providers are Cloud data centres from which game operators lease computing resources for running game servers with guaranteed QoS. Whenever the resource SLA terms (mostly limited to resource availability) are breached, resource providers compensate the operators. We studied the opportunity of employing Cloud infrastructures for MMOG hosting in [3].

In this work, we investigate the use of our proposed Cloud-based middleware for autonomous, self-adaptive hosting and operation of MMOGs on unreliable resources. For this purpose, we developed an MMOG simulator compliant with our ecosystem capable of emulating the behaviour of the four actors using traces collected from a real-life MMOG and resource characteristics from real-world commercial Cloud providers. Using our simulator, we study the impact of MMOG operation and provisioning on the involved actors by considering different resource availability levels, and highlight the advantages of dynamic resource

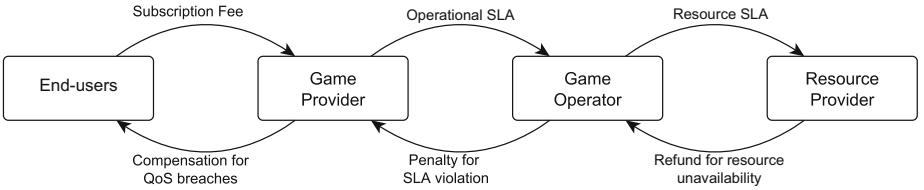

Fig. 1. Overview of MMOG ecosystem and SLA relationships

allocation over the static overprovisioning with respect to two metrics types: QoS offered to the clients and financial profit of the game provider and operator.

The paper is organised as follows. Section 2 introduces the failures that can occur during the MMOG operation considered in our model. Section 3 presents our MMOG simulation tool used for evaluation in Section 4 using traces collected from a real-life MMOG on commercial Cloud resources. Section 5 reviews the related work and Section 6 concludes the paper.

2 Failure Model

We identify in our MMOG operational model two principal types of failures that disturb the users' gameplay with a negative impact on the offered QoS, but from which the system is capable of recovering with no human intervention: *resource failures* and *management failures*.

Resource failure. In our scenario, the game operator runs the MMOG sessions on distributed heterogeneous resources provisioned from Cloud providers which are subject to a multitude of unexpected events that can lead to failures. If a machine crashes, hangs or becomes unreachable through the network, the running MMOG server is compromised disrupting the normal operation of the distributed MMOG session. In existing commercial deployments, such a severe unexpected event typically requires human intervention and can lead to hours of partial service unavailability, or even total unavailability in case of correlated failures. In our proposed architecture, the detection of this type of failure at the game operator level triggers a self-healing process consisting of two actions:

1. provisioning of a new resource or set of resources with the same (or better) characteristics as the failing one;
2. starting a new MMOG server part of the session and to which the clients are instructed to reconnect.

Thus, the MMOG session is salvaged but, regardless of these actions, the clients connected to the failing MMOG server will experience a *total interruption* in gameplay for a certain amount of time.

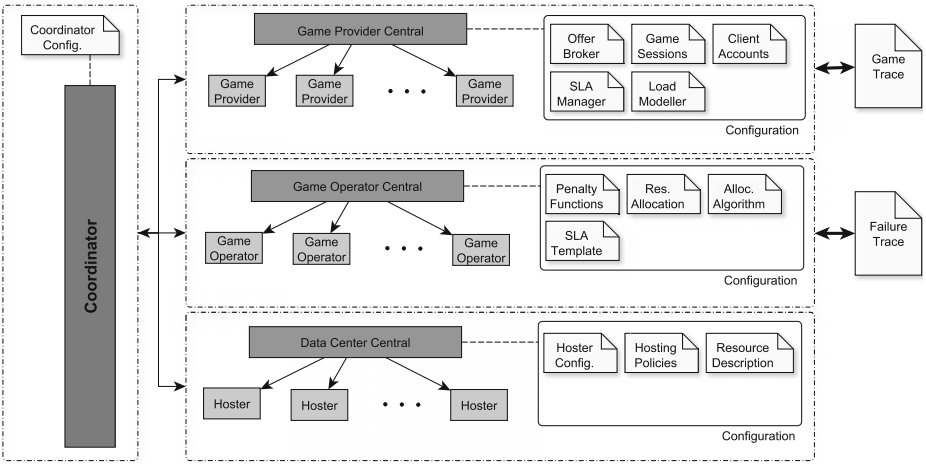

Fig. 2. Schematic MMOG simulation tool architecture

Management failure. The game operator automatically adapts the amount of provisioned resources in a dynamic fashion to achieve a proper MMOG session operation with reduced allocation costs while reaching the targeted QoS. In case of erroneous load estimations or sudden increases in the number of accessing users, the provisioned resources are not sufficient to handle the generated load leading to the degradation of the QoS. The players connected to the overloaded servers will experience in this case a fragmented, unrealistic gameplay. In this situation, the game operator compensates either by redistributing the users to other MMOG servers within the same session aiming a better load distribution and a more efficient utilisation of the already provisioned resources, or by provisioning more resources for the affected session. We call this type of disturbance, where the users are not disconnected from the MMOG session but their gameplay experience is degraded, as *partial interruption*.

3 MMOG Simulator

To ensure a flexible and complete validation of our MMOG research, we developed a trace-based MMOG simulation tool displayed in Figure 2 that implements our MMOG architecture and failure model.

Coordinator is the central part in charge of initialising the simulation according to the input configuration and managing the whole execution of the simulation through the following tasks: (1) calculation of the current game load and resources required; (2) checking of the current simulation state; (3) negotiation of new SLAs; (4) generation of resource requests and offers; (5) resource allocation; (6) creation output files needed for the evaluation of the experiment.

Game traces are the central element of every MMOG simulation, consisting of a data file for every active game zone, and containing the number of currently active clients and the corresponding timestamps. Based on these data, we simulate the same amount of game zones distributed over the world together with their related number of players using a configurable simulation time step. The overall duration of the simulation can be reduced by truncating the trace files.

User information is extracted from the trace files by a special service of the simulator based on the connected user accounts, necessary for the calculation of the subscription fees. To create a correct geographic mapping of the captured game, the simulator also uses the location of each individual client.

Game providers are in charge of managing the MMOG load models, connections to the user accounts, and provisioning of offers from the game operator to the customers. Based on the number of user access requests and their geographical origin, the game providers estimate the demand for each geographical area and construct operational SLA templates negotiated with the game operators. The SLA negotiation strategy and other fine-tuning provisioning variables can be altered in special configuration files. Additionally, the game providers manage the simulated game sessions by mapping the input traces to game zones.

Game operators are in charge of predicting and balancing of the load of the game sessions and zones, controlled by changing the properties for the respective services and algorithms. An essential part of the setup is the configuration of the *SLA templates* that manage the business transactions between the game operators and the game providers with variables such as the base price, the duration, and the number of serviced players. Based on historical data and the currently assigned users, the operator predicts the load for predetermined time intervals (shorter for dynamic MMOGs and longer otherwise) using the method presented in [7]. The prediction consists of the user distribution in the MMOG world and a load model for translating it into resource requirements. At each prediction interval, the operator evaluates the provisioned resources and makes adjustments by allocating or releasing resources and redistributing the load. Other configuration files allow controlling the penalties for QoS violations, resource unavailability, as well as sorting and ranking resource offers or the desired allocation time.

Data centre emulates a configurable network of Cloud providers with resources and services spread across the globe. To configure the resource providers, each setup configuration has a folder for every simulated data center containing information such as location, network bandwidth, and machines defined by their number of cores, processing speed and memory or disk size.

Failure traces are files containing the resource failures occurring during the simulated MMOG sessions. The format of the traces is inspired from the Failure Trace Archive (FTA), with a simplified format consisting of only three columns: the host name of the failing resource and the associated start and end (UNIX

epoch) timestamps. Our resource availability model is based on three important parameters: (1) *inter-arrival time (IAT)* describing time between two consecutive failures, (2) *duration* of each individual failure, and (3) *size* described by the number of affected resources. Other parameters required for the failure generation are the number of available resources in the MMOG environment, the total duration of one simulation period, and the percentage of resource availability.

4 Experiments

For evaluating the impact of the resource and of the management failures, we use our MMOG simulator and six months of trace data collected from a real-world MMOG called RuneScape (`http://www.runescape.com/`).

4.1 Setup

We collected traces from 150 RuneScape servers with a sampling rate of two minutes, containing the number of players over time for each server group, aggregating approximately 40 million metric samples per simulation. Based on the number of concurrently active accounts from the RuneScape traces, we approximate the actual number of connected clients using the concurrent active account ratio metric approximated at around 15% for RuneScape. We use a prediction interval and a simulation step of two minutes, which proves to be adequate for this type of games and the collected traces. Since we also study the financial impact of resource unavailability on the MMOG actors, we modelled the client accounts with different monthly, bimonthly, trimestrial and semestrial subscription types based on the real subscription and payment methods of Jagex Ltd. (the developer and publisher of RuneScape) as of August 2010 (see Table 1). We employ a setup with one game operator and one game provider since their number does not impact the QoS metrics for the clients targeted by our experiments.

Table 1. RuneScape subscription plans for monthly, bimonthly, trimestrial, and semestrial payments

Plan	*Payment*	*Subscription price [USD]*			
		Monthly	*Bimonthly*	*Trimestrial*	*Semestrial*
SCC	Credit card	5.95	11.9	17.85	35.7
SPP	PayPal	7.5	12.99	18.15	36.1
SBT	Bank transfer	7.99	14.59	19.99	37.99

The traces contain the maximum load of approximately 150 concurrent game zones spread all across the globe, where every zone needs one virtual machine (VM) instance to be properly executed. For this purpose, we model a distributed heterogeneous environment with 16 commercial Cloud providers aggregating 70 different VM types (see Table 2) and providing a large enough resource pool for the game operator of approximately 3500 concurrent VM instances for running all game zone instances. The parameters that model each VM type include the number of cores (1 to 12), CPU speed, RAM size (1 to 48 GB), and in- and outbound network bandwidth. While most of the parameters are clearly defined

Table 2. Summary of Cloud providers

Provider	*VM types*	*Location*	*Allocation time [hours]*
Amazon	6	4 (Asia, UK, USA (E, W))	1
CloudCentral	5	1 (AUS)	1
ElasticHosts	4	1 (UK)	1
FlexiScale	4	1 (UK)	1
GoGrid	4	1 (USA (E))	1
Linode	5	1 (USA (E))	24
NewServers	5	1 (USA (E))	1
OpSource	6	1 (USA (E))	1
RackSpace	4	2 (USA (E, C))	1
ReliaCloud	3	1 (USA (C))	1
SoftLayer	4	3 (USA (E, C, W))	1
SpeedyRails	3	1 (CAN W)	24
Storm	6	2 (USA (E, W))	1
Terremark	5	1 (USA E)	1
Voxel	4	3 (USA (E), NED, AUS)	1
Zerigo	2	1 (USA (C))	1

Table 3. Number of machines for a given utilisation

Utilisation [%]	*Number of machines*
5	2120
20	520
40	280
60	190
80	150
95	140

in the specifications of the commercial Cloud providers, the processing power of the VMs is not concretely quantified. Thus, we express it using an appropriate metric called *RS unit* representing the equivalent computational requirements of one RuneScape server servicing 2000 concurrent clients. We compute this metric based on existing benchmarks[1,2] and our previous performance investigations [4].

Since one of our goals is to also show the impact of resource unavailability in a sparse resource environment, we derive six additional synthetic setups by gradually removing machines from the original VM instance pool to increase the resource utilisation, as illustrated in Table 3. In this table, the utilisation describes the percentage of machines to be allocated for a proper execution of all game zones at maximum load. In addition to the sparse resource environment, we introduce a controlled resource unavailability to our environment by associating to Cloud providers failure traces with tunable availability. We employ the resource availability model proposed in [2] and generate failure traces with average availabilities ranging from 99.5% to 99.9%. The traces are characterised through their failures' duration, size and IAT, each modelled through a statistical distribution. The distributions' parameters are presented in Table 4, along with the statistical properties of the resulting traces, where $Q1$, $Q2$ and $Q3$ represent the lower, the median, and the upper quartiles. For all traces we employ the same distribution for the failure duration and size, but we vary the resource availability by adjusting the failure IAT. Both independent and correlated failures are generated, the ratio between the two being $3:2$.

We evaluate the advantage of dynamic resource allocation over a static approach with respect to two metric: QoS and financial impacts. In case of static allocation, resource provisioning is performed once the zone is registered for

[1] `http://blog.cloudharmony.com/2010/05/what-is-ecu-cpu-benchmarking-in-cloud.html`

[2] `http://www.chadkeck.com/2010/05/cloud-hosting-provider-hardware-benchmarks/`

Table 4. Resource availability parameters and statistical characterisation

Failure metric	*Distribution*	*Scale*	*Shape*	*Statistical properties*						*Availability*
				Min	*Max*	*Average*	*Q1*	*Q2*	*Q3*	
Duration [min.]	Log-Normal	2.12	0.306	1	37	8	6	8	10	–
Size [machines]	Weibull	4	5	0	6	3	3	3	4	–
IAT [sec.]	Weibull	13600	7	2073	19668	12722	11381	12906	14254	99.5%
IAT [sec.]	Weibull	7000	7	888	10123	6548	5860	6645	7336	99.6%
IAT [sec.]	Weibull	4750	7	535	6988	4443	3976	4509	4977	99.7%
IAT [sec.]	Weibull	3550	7	476	5146	3320	2971	3370	3720	99.8%
IAT [sec.]	Weibull	2830	7	341	4119	2647	2369	2686	2965	99.9%

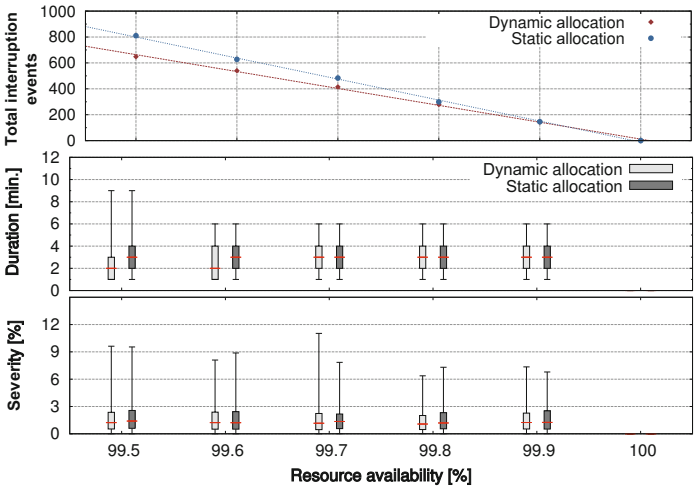

Fig. 3. Total interruption analysis for static and dynamic resource allocations for different resource availabilities

operation, and the session does not get additional resources after a failure event, but waits for the failure to end when the machine becomes operational again.

4.2 QoS Impact

Figure 3 compares the number, duration and severity of all total interruption events for varying resource availabilities for dynamic and static resource allocation approaches. In the top graph, it is visible that the number of total interruptions is increasing with higher resource unavailability for static allocation, leading to an average of 30% increase for a resource availability of 99.5%. We further observe stable, constant values for the severity and the duration of total interruptions (the two bottom graphs) across all resource availability values, which validates the automated process for recovery from resource failures of our MMOG architecture. The dynamic resource allocation reduces the medial duration of the total interruptions to two minutes (the duration of one simulation cycle) for the simulations with the lowest (99.5% − 99.6%) resource availability, and is below four minutes for more than 75% of the events. Despite the

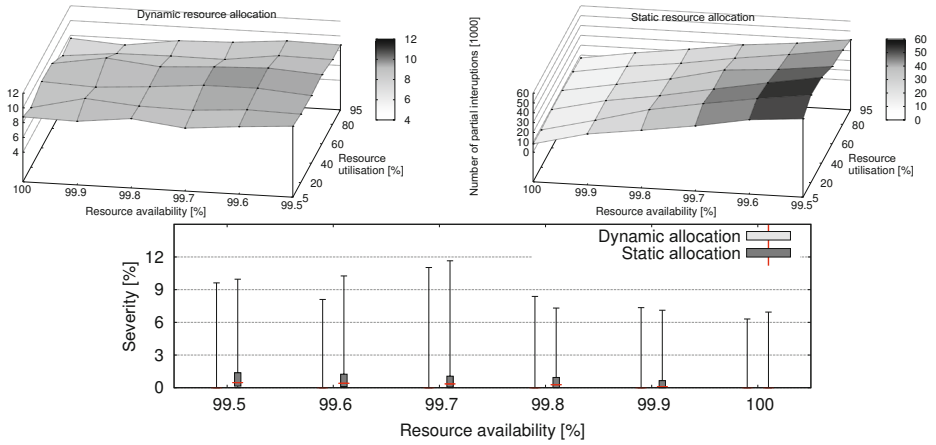

Fig. 4. Partial interruption analysis dynamic and static resource allocations for different resource availabilities

higher count of events and their prolonged duration, the median percentage of affected players (i.e. the severity) remains almost constant below 2% across all availability values, and is only slightly higher in case of the static allocations.

The impact of the different allocation strategies can be best noticed in the analysis of the partial interruption events displayed in Figure 4. While the number of partial interruptions increases with the decreasing resource availability for static allocations, they remain relatively constant for the dynamic allocation strategy for all availabilities and utilisations. On average, the number of partial interruptions for an availability of 99.5% is 4.5 times lower using dynamic allocations. With the dynamic allocation of resources, the partial interruption events have little to no impact on the QoS offered to the end-users (i.e. the severity) since our system can mitigates the effects by allocating additional resources or balancing the load. With static resource allocations, the partial interruption events affect on average about 1% of the connected players.

4.3 Financial Impact

Following the hierarchical architecture presented in Section 1, the first actors to analyse are the end-users who pay a subscription fee that allows them to enter gaming sessions provided by the game providers. Employing the static resource allocation approach not only leads to a degradation of the offered QoS, but also introduces a linear increase in the amount of QoS compensations paid to the users for lower availability and higher utilisation, as displayed in Figure 5. While the compensation payments lower the monthly costs for most of the clients, their overall play satisfaction will suffer.

The impact of the static allocation on the QoS can also be noticed in the financial situation of the game provider who earns money from penalties paid

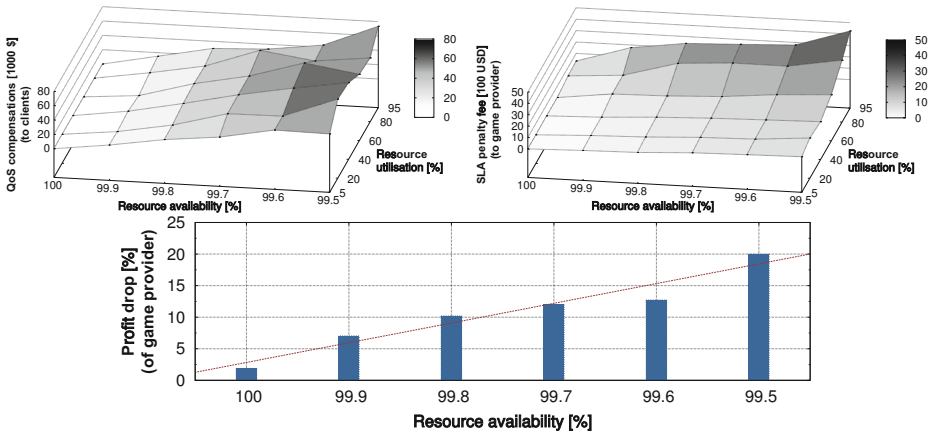

Fig. 5. Analysis of financial impact on involved actors for static resource allocations

by the game operator for SLA violations (part of which it transfers to the client as QoS compensations). The bottom chart Figure 5 shows the drop of the game providers' profit for using the static allocation approach in the different resource availabilities, averaged across all resource utilisation scenarios.

In the lowest tier, the game operator is directly affected by machine failures that occur in the data centres of the resource provider, where static allocation has a strong negative impact on the QoS. In case of a resource failure, the dynamic allocation mechanism provisions additional machines to ensure that no SLAs with the game provider are violated. Since it is not possible to allocate additional resources with the static allocation, almost every resource failure leads to a violation of an SLA, accompanied penalty fees. The top-right chart in Figure 5 shows increasing compensation fees to be paid by the game operator to the provider with increasing unavailability and utilisation of resources.

The financial situation of the resource providers is generally not affected by the static allocation since the negative impacts of this approach are mostly restricted to the three higher tiers of the architecture.

5 Related Work

Much recent work focuses on (soft) QoS guarantees for MMOG operation [5,6,15]. Wong [15] proposes a resource provisioning algorithm with QoS guarantees, but considers only networking aspects, whereas we focus on maintaining QoS even during total resource failures. Complementary to our study, Lee and Chen [5] investigate MMOG server consolidation techniques, focusing on energy consumption. There have been a number of research activities in assessing the performance of virtualised resources in Cloud computing environments [11] and in general [12], some also considering the availability of Cloud resources [9]. In contrast to these studies, ours targets realistic Cloud resources with limited availability for a new

application class (MMOG). Regarding the resource and MMOG deployment models, one study [13] comes close to our approach by proposing virtual machines for multi-player game operation. However, our work focuses on MMOGs which, in contrast to classic multi-player games, are distributed applications (multiple MMOG servers interconnected in a single session) serving a several orders of magnitude higher number of clients. Additionally, we also consider the virtualised resources as part of commercial Cloud computing platforms. In the area of reliability, there are studies which investigate the characteristics of resource and workload failures, but do not assess their effects on the underlying systems' performance [10,14]. Other efforts consider uncorrelated failures in distributed systems [1] and evaluate the resulting performance of the affected systems [2], but only for high-performance computing applications. In contrast, we employ the failure model introduced by [2], but apply it to Cloud resources and evaluate the consequences of utilising such resources on the QoS of MMOGs.

6 Conclusions

We presented a simulator that implements a new ecosystem for operating MMOGs on Cloud infrastructures which effectively splits the traditional monolithic MMOG companies into three smaller and better focused actors whose interaction is regulated through bipartite SLAs: game providers, game operators, and resource providers. In our model, game operators dynamically provision resources from Cloud resource providers based on the MMOG load so that the QoS to the end-users is guaranteed at all times. Game providers lease operation SLAs from the game operators to satisfy all client requests and manage multiple distributed MMOG sessions. These three self-standing, smaller, more agile service providers enable access to the MMOG market for the small and medium enterprises, and to the current commercial Cloud providers. We evaluated in this paper using traces collected from a real-life MMOG the impact of resource availability and utilisation on the QoS and financial situation of the involved actors by comparing our novel dynamic resource allocation method with the traditional static allocation strategy. We found out that our MMOG ecosystem successfully mitigates the performance degradation of running MMOGs on real commercial Cloud resources with limited availability to gameplay disruptions of less than four minutes, independently of the duration of the underlying resource failure. The majority of resource failures affect less than 2% of the users participating in autonomously operated MMOG sessions. A low resource availability increases the number of gameplay disruptions, while a high resource contention results in longer disruptions affecting more clients. Finally, static resource allocation has a negative impact on the financial situation of the involved actors due to high compensation and penalty fees for QoS and SLA breaches upon low resource availability and high utilisation.

References

1. Bhagwan, R., Savage, S., Voelker, G.: Understanding availability. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 256–267. Springer, Heidelberg (2003)
2. Iosup, A., Mathieu, J., Sonmez, O., Epema, D.H.J.: On the dynamic resource availability in grids. In: 8th IEEE/ACM International Conference on Grid Computing. pp. 26–33. IEEE Computer Society (September 2007)
3. Iosup, A., Nae, V., Prodan, R.: The impact of virtualization on the performance and operational costs of massively multiplayer online games. International Journal of Advanced Media and Communication 4(4), 364–386 (2011)
4. Iosup, A., Ostermann, S., Yigitbasi, N., Prodan, R., Fahringer, T., Epema, D.: Performance analysis of Cloud computing services for many-tasks scientific computing. IEEE Transactions on Parallel and Distributed Systems 22(6), 931–945 (2011)
5. Lee, Y.-T., Chen, K.-T.: Is server consolidation beneficial to MMORPG? A case study of World of Warcraft. In: 3rd International Conference on Cloud Computing, pp. 435–442. IEEE Computer Society (July 2010)
6. Briceño, L.D., et al.: Robust resource allocation in a massive multiplayer online gaming environment. In: 4th International Conference on Foundations of Digital Games, pp. 232–239. ACM (2009)
7. Nae, V., Iosup, A., Prodan, R.: Dynamic resource provisioning in massively multiplayer online games. IEEE Transactions on Parallel and Distributed Systems 22(3), 380–395 (2011)
8. Nae, V., Prodan, R., Iosup, A.: SLA-based operation of massively multiplayer online games in competition-based environments. In: Proceedings of the International C* Conference on Computer Science & Software Engineering, pp. 104–112. ACM (July 2013)
9. Nagarajan, A.B., Mueller, F., Engelmann, C., Scott, S.L.: Proactive fault tolerance for hpc with xen virtualization. In: 21st Annual International Conference on Supercomputing, pp. 23–32. ACM (2007)
10. Nurmi, D., Brevik, J., Wolski, R.: Modeling machine availability in enterprise and wide-area distributed computing environments. In: Cunha, J.C., Medeiros, P.D. (eds.) Euro-Par 2005. LNCS, vol. 3648, pp. 432–441. Springer, Heidelberg (2005)
11. Palankar, M.R., Iamnitchi, A., Ripeanu, M., Garfinkel, S.: Amazon S3 for science grids: a viable solution? In: International Workshop on Data-aware Distributed Computing, pp. 55–64. ACM (2008)
12. Quétier, B., Neri, V., Cappello, F.: Scalability comparison of four host virtualization tools. Journal of Grid Computing 5, 83–98 (2007)
13. Reed, D., Pratt, I., Menage, P., Early, S., Stratford, N.: Xenoservers: Accountable execution of untrusted programs. In: Seventh Workshop on Hot Topics in Operating Systems, pp. 136–141 (1999)
14. Schroeder, B., Gibson, G.A.: A large-scale study of failures in high-performance computing systems. IEEE Transactions on Dependable and Secure Computing 7(4), 337 (2010)
15. Wong, K.W.: Resource allocation for massively multiplayer online games using fuzzy linear assignment technique. In: Consumer Communications and Networking Conference, pp. 1035–1039. IEEE (2008)