A New Business Model for Massively Multiplayer Online Games

Vlad Nae, Radu Prodan, Thomas Fahringer
University of Innsbruck
{vlad,radu,tf}@dps.uibk.ac.at

Alexandru Iosup
Delft University of Technology
A.Iosup@tudelft.nl

ABSTRACT

Today, highly successful Massively Multiplayer Online Games (MMOGs) have millions of registered users and hundreds of thousands of active concurrent players. To sustain their highly variable load, game operators over-provision a large static infrastructure capable of sustaining the game peak load, even though a large portion of the resources is unused most of the time. This inefficient resource utilisation has negative economic impacts by preventing any but the largest hosting centres from joining the market and dramatically increases prices.

In this paper, we propose a new business model of hosting and operating MMOGs based on Cloud computing principles involving four actors: resource provider, game operator, game provider, and client. Our model efficiently provisions on-demand virtualised resources to game sessions based on their dynamic client load, which dramatically decreases prices and gives small and medium enterprises the opportunity of joining the market through zero initial investment.

We validate our new model and its underlying business relationships through trace-based simulations utilising six months worth of monitoring data from a real-life MMOG using emulated resources from 16 of the largest Cloud resource providers currently on the market. We demonstrate that our model can operate state-of-the-art MMOGs with an average monthly gross profit of nearly $6 million excluding game purchase prices, overheads and taxation, while being able to maintain and control the QoS offered to all clients. Finally, we show how our approach is capable of operating next generation very highly interactive MMOGs with a small increase of 5.8% in the subscription price.

Categories and Subject Descriptors

K.6.2 [**Management of Computing and Information Systems**]: Installation Management—*pricing and resource allocation, performance and usage measurement*

ICPE'11, March 14–16, 2011, Karlsruhe, Germany.

General Terms

Economics

Keywords

MMOG, resource allocation, service level agreements

1. INTRODUCTION

Online entertainment including gaming is a strongly growing sector worldwide. *Massively Multiplayer Online Games (MMOG)* grew from ten thousand subscribers in 1997 to 6.7 million in 2003 and the rate is accelerating estimated to 60 million people by 2011. The release of World of Warcraft in 2005 saw a single game break the barrier of four million subscribers worldwide. The market size shows equally impressive numbers, estimated by the Entertainment Software Association (ESA) to seven billion US Dollars (USD) with an avid growth over 300% in the last 10 years. In comparison, the Motion Picture Association of America (MPAA) reports a size of 8.99 billion USD and the Recording Industry Association of America (RIAA) a size of 12.3 billion USD which has stagnated (and even decreased by 2%) in the last ten years. It is therefore expected that the game industry will soon grow larger than both movie and music market sizes. The Chinese market is growing faster than anywhere, top games being able to generate as much as 100 million USD a year each. The market is predicted to grow from 580 million USD in 2005 to 1.7 billion USD in 2010.

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. To comply with the variable computational and latency-aware resource demands of the players distributed worldwide, the MMOG operators maintain a multi-server distributed infrastructure with sufficient computational and network capabilities necessary to guarantee the *Quality of Service (QoS)* requirements and a smooth game play at all times. This statically provisioned infrastructure has several major drawbacks: is subject to over-provisioning, increases the operational costs of MMOGs, and is vulnerable capacity shortages in case of sudden increases in demand. The current industry approach to ensure that resources are available even when they are not needed is based on resource ownership and over-provisioning. For example, the operating infrastructures of leading MMOGs such as World of Warcraft and RuneScape comprise thousands of computers each in hundreds of physical locations, and resource ownership can take up to 40% of the game revenue

In previous work [13], we presented a new dynamic provisioning method for MMOGs based on Cloud computing technology that provisions virtualised Infrastructure as a Service (IaaS) resources to MMOGs on-demand based on the exhibited load, while still fulfilling the necessary QoS requirements. The new Cloud-based provisioning model of leasing resources only when and for how long they are needed at fixed cost established through *Service Level Agreements* (SLAs) presents great potential of eliminating the need of permanent over-provisioning of occasionally needed resources which frees companies such as MMOG operators from the large costs of buying and maintaining hardware and from the rapid deprecation of hardware investments. To support the Cloud-based MMOG hosting, we present in this paper a novel *MMOG business model* which opens the market, currently dominated by giants functioning simultaneously as publishers, providers, operators, and sometimes even as developers, to small and medium enterprises that no longer need to own large resource infrastructures in order to successfully operate MMOG sessions. Our new business model utilises the MMOG resource provisioning model described in [12] and involves four actors: (1) a *resource provider* which rents the (Cloud) infrastructure for running the MMOG servers, (2) a *game operator* in charge of renting (from resource providers) the appropriate resources such that the QoS requirements are fulfilled at all times; (3) a *game provider* which offers a selection of MMOGs to the clients and manages each distributed MMOG session, and (4) the clients who participate in the MMOG sessions managed by game providers by connecting to the MMOG servers. The interactions between the first three actors are regulated by bipartite SLAs, and by client accounts in case of the fourth game provider – client interaction. We validate our new model and its underlying business relationships through trace-based simulations utilising six months worth of monitoring data from a real-life MMOG called RuneScape. We demonstrate that our model can operate state-of-the-art MMOGs with an average monthly gross profit of nearly $6 million excluding purchase prices, overheads and taxation, while being able to maintain and control the QoS provided to the clients. Finally, we show how our approach is capable of operating next generation very highly interactive MMOGs with a small 5.8% increase in the subscription price.

The paper is structured as follows. In Section 2 we present the generic computational and resource models, involved business actors, MMOG types, and QoS requirements underneath our approach. In Section 3 we define the business relationships between the actors participating in our MMOG ecosystem. A real MMOG trace based evaluation of the operational and business models is presented in Section 4. We compare our work against the related work in Section 5 and conclude in Section 6.

2. MODEL

In this section, we introduce the abstract computational and resource models, the business actors, the MMOG types, and QoS requirements underneath our approach. An overall architecture is displayed in Figure 1.

2.1 Computational Model

Online games can be seen as a collection of networked *game servers* that are concurrently accessed by a number of *players* (also called *clients*). Clients connect directly to one

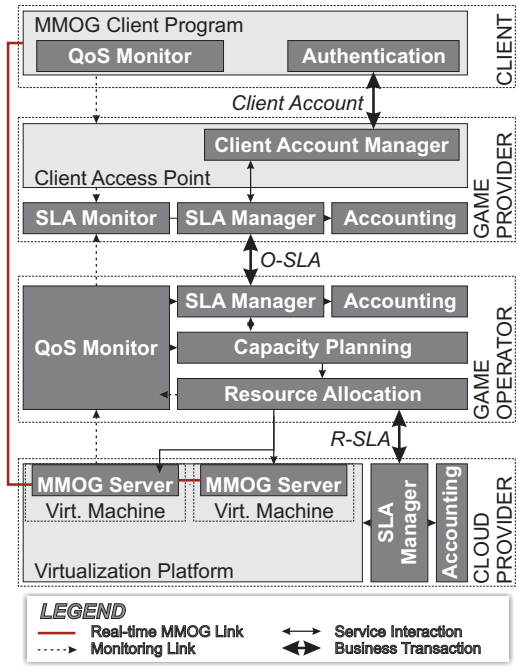

Figure 1: The MMOG ecosystem.

game server, send their play actions (e.g. movements, collection of items, shooting actions), and receive appropriate responses. Each player is usually mapped to one *avatar* lying at precise coordinates in the game world which. Based on the actions sent, the avatar dynamically interacts with others within a *game session*, influencing each others' state. The state update responses must be delivered promptly within a given time interval to ensure a smooth, responsive and fair experience for all players.

For the vast majority of games there is a similar computational model. The game server runs a large loop in which the state of all entities is first computed and then broadcast to the clients. All entities within a specific avatar's *area of interest* (usually a surrounding zone) are considered to be interacting with it and have an impact on its state. The more populated the avatars' areas of interest are and the more interactions between entities exist, the higher the load of the underlying game server is. An overloaded game server delivers state updates to its connected clients (i.e. movements and actions of teammates and opponents) at a lower frequency than the required *tick rate*, which makes the overall environment fragmented, unrealistic and unplayable.

Depending on the game, typical update frequencies to ensure fluent play must be between 60 and 120 Hertz for online *First Person Shooter* (FPS) action games and between 1 and 10 Hertz for *Massively Multiplayer Online Role-Playing Games* (MMORPG). A good game play experience is critical in keeping the players engaged, and has an immediate consequence on the income of the MMOG operators. Failing to deliver timely updates leads to a degraded game experience and triggers player departure and account closing.

To support at the same time millions of active concurrent players and many more other entities with guaranteed QoS, resource providers install and operate a large static infrastructure, with hundreds to thousands of computers hosting

a single distributed game session. The most common game session distribution technique is *zoning*, which is based on spatial partitioning of the game world into zones to be handled independently by separate machines, together with the contained avatars. Other techniques such as instancing and replication [6] are based on zone replication and entity distribution (as opposed to zone distribution).

2.2 Resource Model

We consider an *Infrastructure as a Service (IaaS)* Cloud resource model providing generic functionality for hosting and provisioning of access to raw computing infrastructure and its operating middleware. IaaS are typically provided by data centres such as Amazon EC2, RackSpace, FlexiScale, GoGrid, Voxel or NewServers that rent hardware facilities to customers which are freed from the burden of their maintenance and deprecation costs. IaaS is characterised by the concept of hardware *virtualisation* which allows a customer to deploy and run a guest operating system on top of the virtualisation software offered by the provider, encapsulated in a *virtual machine* (VM).

Virtualisation in IaaS is a key step towards distributed, automatic, and scalable deployment, installation, and maintenance of software, which inevitably adds overheads to the runtime performance of game servers that we studied in [13]. Our combined analytical and empirical analysis comprises two important classes of overheads: (1) *VM instantiation* is the overhead of starting a VM on a selected resource and consists of four components: *VM image preparation*, *VM transfer*, *VM start*, and *VM removal*; (2) *VM execution* is the overhead of executing a game server within a VM which has a *computation* and a *communication* component.

2.3 Actors

In this paper, we propose a new ecosystem for MMOG hosting, operating, and provisioning in distributed heterogeneous infrastructures consisting of the three actors, extending on our previous model introduced in [13]: resource providers, game operators, and game providers (see Figure 1). The communication between these business actors is negotiated and regulated through bipartite SLAs, representing wrappers around QoS parameter guarantees they mutually need to deliver to each other (e.g. the correct state update rate for a certain price in a certain period). These business relationships will be discussed in detail in Section 3.

2.3.1 Resource Providers

Resource providers (also known as hosters) are typically data centres that provision to game operators raw computing and storage infrastructure for running game servers with guaranteed QoS. In this paper, we consider hosters as IaaS Cloud computing providers that dynamically provision to game operators virtualised resources for scaling up/down a game session based on its dynamic load generated by connected clients. In previous work [13], we have studied the opportunity of employing IaaS-based Cloud infrastructures for MMOG hosting with respect to the performance penalties incurred by the virtualisation overheads. In this paper, we add a new dimension to this study by investigating business models and the cost of renting resources.

2.3.2 Game Operators

The game operators handle multiple simultaneous MMOG sessions of different types. They receive requests from game providers for operating certain MMOG zones with guaranteed QoS. Based on the total number of clients connected to a game zone, the game operators acquire from hosters the correct amount of resources for a certain duration to fulfill the game realtime QoS requirements. The game operator achieves this difficult task using five major services.

A *capacity planning* service is in charge of estimating the game session load which drives the allocation of the correct amount of resources. Based on the monitoring information collected from the game servers (i.e. from the QoS monitor – see Figure 1), a load prediction component [18] estimates the future distribution of avatars and other entities in the game world, distribution which has the highest impact on the session and underlying servers' load. On top of it, we devised accurate analytical models for translating the entity distribution and interactions into estimated game server load, including processor, memory, and network. Load models are typically devised by the game developer and supplied to the game provider together with the game software, and highly depend on the game type and its computational characteristics (see Section 2.4). The capacity planning also considers in the load modelling process the IaaS Cloud virtualisation overheads introduced in Section 2.2 and described in [13] for steering the resource allocation process.

Based on the estimated load, a *resource allocation* service [12] leases from the resource providers the correct amount of resources and starts/stops the corresponding servers to accommodate the new player number (through zoning, replication, or instancing parallelization techniques [6]). For example, by timely foreseeing critical hot-spots in the game world (i.e. excessively populated areas of interest generating a large number of interactions), one can dynamically provision additional servers on newly leased resources and take proactive load balancing actions that transparently redistribute the game load on the new zone instances before the old servers become overloaded.

The *QoS monitor* collects and analyses information about the state of the MMOG sessions and the QoS delivered by the running servers. The session state information like zone assignments and server-to-resource mappings is collected from the resource allocation service, while the actual QoS information like game loop tick rate, utilised memory and network bandwidth, and average client-server connection latency is gathered from the MMOG servers. At predefined time intervals, the QoS monitor analyses the collected information and aggregates it into monitoring reports which are utilised by the SLA manger, the capacity planning, and the game provider's SLA monitor.

Based on the resource utilisation plans provided by the capacity planning, the *SLA manager* constructs the SLA template offers, partakes in the negotiations with the game providers, and eventually instructs the capacity planning on what game zones to start, how many client connections to allow, and the target QoS parameters. The SLA manager is also responsible for *enforcing* the game operator's SLAs so that the underlying terms are fulfilled at all times, for example by negotiating new resources with the resource provider in case the capacity planning foresees an increase in the number of players and in the overall session load.

The signed SLAs are sent from the SLA manager to the *accounting* service which charges the game provider the final SLA price. When instructed by the SLA manager, the

accounting service may also charge game providers the appropriate *penalty* fees for any *SLA faults*, for example when assigning more clients than the SLA permits.

2.3.3 Game Providers

Game providers interact with clients and offer them a selection of MMOGs, usually by contracting new games from game development companies (this interaction is not modelled in this paper). Based on the requests received, the game providers assign clients to game zones which are delegated to game operators for execution with guaranteed QoS. These tasks are performed by a team of four services (see Figure 1).

The client-game provider interaction is managed by the *client account manager* which allows clients to create accounts and mediates their connections to the MMOG servers by providing the connection details and the server access tokens. The *SLA monitor* collects information from the game operators' and clients' QoS monitors and generates monitoring reports which are sent to the SLA manager. Based on these monitoring reports, the number of accounts, and the estimated client activity, the *SLA manager* signs with game operators the appropriate SLAs and distributes the clients and the game zones accordingly. Similar to the operator's service, the SLA manager is also enforcing the game provider's SLAs. The SLA manager also detects SLA faults, for example a state update rate below the minimum required threshold, and instructs the *accounting* service to charge game operators the corresponding penalty fees. The accounting service also charges the clients for access to MMOGs.

2.4 Load Complexity

In terms of the MMOG server computational complexity, we have identified three major types of games: (1) *low player interaction* like "trade" or "minigames" in which players interact usually in groups of two; (2) *average player interaction*, like the "mini-quest" and "quest" worlds, in which groups of players interact with each other and engage themselves in battle with non-player characters; and (3) *high player interaction*, like the player-versus-player and the "Clan Wars" worlds, in which players engage in battle freely or in groups against each other. The approximate computational complexities of these game types are $O(N)$ for low player interaction worlds, $O(N \cdot \log(N))$ for average player interaction, and $O\left(N^2\right)$ for high interaction games. Additionally, we envisage the next generation of FPS MMOGs characterised by a large number of players that interact in small game world areas requiring very frequent updates from the server. The estimated computational complexity of this *very high player interaction* MMOG type is $O\left(N^2 \cdot \log(N)\right)$.

2.5 QoS Parameters

The main challenge for the presented architecture is how to map SLAs as business contracts that must be honoured at all times to the realtime QoS requirements of MMOGs which can only be enforced through best-effort mechanisms using today's resource allocation mechanisms in Cloud and Internet-based infrastructures. We can distinguish between two major QoS parameters in MMOGs.

Quality of the game play is usually achieved if the state update rate from game servers to clients is above the minimum game server rate specified by the game developer (ignoring the uncontrollable network bottlenecks at the client side). Maintaining the minimum tick rate can be challenging for game operators, especially in fast-paced FPS action games due to the highly dynamic and unpredictable actions which can occur in a relatively short time interval. We define the time interval during which the MMOG server delivers updates at a frequency lower than the required tick rate as *interruption event*, which generates penalties from the operators to the game providers, and from the game providers to the clients that suffer from a degraded game play experience. An effective method of avoiding QoS breaches while maintaining a good resource utilisation is through *controlled over-allocation* studied in [12], whose effect on the game operator's gross profit will be investigated in Section 4.6.

Service availability is the other important QoS parameter representing the time during which a client is able to connect to the desired MMOG server. There are three circumstances in which a *denial of service event* occurs, meaning that the client is refused connection to an MMOG server: (1) the server does not have enough memory to store the client's avatar and the associated data, which can be solved again through controlled over-allocation; (2) the game provider's SLA with the game operators are insufficient for the current client demand; and (3) the server has not yet been started or has not finalised the start-up procedure. We will study the second case in Section 4.3 by finding the best estimation for the number of concurrently active client accounts and the third case in Section 4.4 by triggering the resource allocating action ahead of time, to accommodate the Cloud provider-specific VM instantiation time and by adjusting the load prediction time step.

3. BUSINESS RELATIONSHIPS

In this section, we present in detail the business relationships among the different actors involved in our MMOG ecosystem outlined in the previous section.

3.1 Client and Game Provider

The interaction between the client and the game provider is in general automatic and requires human intervention only from the client side. The relationship is regulated by the client account, usually created through a Web portal or platform by agreeing upon a *contract* with the game provider. The contract should only include generic mutual obligations valid for all MMOGs and for all clients, while further refinement and details can be added in the form of annexes agreed upon for each particular MMOG. Typical client obligations comprise subscription costs, client community interaction rules, and costs for accessing MMOG sessions, while obligations of the service provider include guaranteed provider (Web platform, mediation of client connections to MMOGs) and session availability times, game world and game zone accesses, or compensations in case of contract violations. Client accounts can have unlimited duration, while MMOG contract annexes have well-defined validity periods of usually one month, but bimonthly, trimestrial, semestrial, or even yearly contracts are also possible.

The typical interaction between the client and the game provider takes place as follows. First, the client selects the desired game provider based on the selection of MMOGs and the account (contract) terms he offers. Then, he creates an account with the selected provider by accepting the contract one time, while future accesses will be done through the received account credentials, typically username and password. At this stage, the client has the possibility to create and cu-

stomise his own character (the avatar) which can be shared between different MMOGs. Furthermore, he also has access to MMOG statistics such as playtime, avatar abilities, owned items, etc. The client selects the MMOG he wants to play and, after agreeing upon the presented contract annex (one time only), he is allowed to connect to the MMOG session. On rare cases, the client might be refused the service either because of the game provider underestimated the demand for the requested MMOG, or because the game operator has not allocated enough resources for the respective session. In these situations, the game provider will compensate the client according to the terms of the contract.

3.2 Game Provider and Game Operator

The interaction between the game provider and game operator is automatic and requires no human intervention. Based on the total number of accounts for each MMOG and on the total estimated *concurrently active accounts* (representing clients connected to the MMOG session), the game provider computes his requirements in terms of the maximum number of clients for each game zone. Based on these estimated requirements, the game provider negotiates the most appropriate terms for hosting each zone by establishing *Operation SLAs* (O-SLA) with the game operators.

The O-SLA negotiation between the game operator and game provider consists of four steps. In the first step, the game operator checks the resources offered by different Cloud providers and, based on their policies, publishes an *O-SLA template* consisting of a set of terms which have either single or ranges value, as follows:

- *issuer* is the game operator that issued the O-SLA;
- *MMOG name* and *version*;
- *validity period* (range value) represents the SLA lifetime offered by the issuer with variable granularity from daily to semestrial;
- *client number* (range value) issuer is ready to service;
- *measurement timestep* is the time interval between consecutive QoS evaluations and, implicitly, between O-SLA fault checks;
- *instantaneous non-interruption ratio* (range value) represents the minimum percentage from the recommended state update frequency the game operator guarantees to maintain for all clients throughout the O-SLA validity period. For example if the optimal update frequency given by the game developer is 40 Hertz and the O-SLA specifies an instantaneous non-interruption ratio of 90%, the game operator should maintain the update frequency above 36 Hertz at all times (evaluated in every measurement step);
- *total non-interruption ratio* (range value) represents the percentage of QoS fulfilment (i.e. no interruption events) from the entire O-SLA validity time. As an example, considering that the total non-interruption ratio of an O-SLA with a 24 hour validity period would be 99.9%, then the game operator should provide at least 23.98 hours of service (i.e. game play) without any interruption events for all clients. This term is evaluated only at the end of the O-SLA validity period;
- *penalty clauses* are a set of functions defining the penalties the issuer has to pay in case any SLA faults (i.e. serviced client number, instantaneous non-interruption ratio, total non-interruption ratio);
- *base price* for accepting an SLA utilising the lowest values in the given ranges for all terms.

In the second step, the game provider selects the best matching O-SLA template for the game zones based on its own policy composed of four terms: (1) the amount of clients to service computed as an estimated concurrently active accounts ratio from the total MMOG subscriptions, (2) the minimum instantaneous non-interruption ratio, (3) the minimum total non-interruption ratio, and (4) an interval of acceptable O-SLA validity periods. Next, it instantiates the terms from the template specified as ranges, and returns the template to the game operator requesting a concrete offer. In the third step, the game operator rechecks the availability of the necessary resources, computes the final O-SLA price based on its internal policies defined as a set of *pricing functions* for each O-SLA term, and presents this final offer to the game provider. Finally, in the fourth step, if the game provider accepts the offer, the game operator activates the O-SLA and charges the game provider the final price.

When an O-SLA template is accepted with its default values, the final price of the O-SLA is the base price. Otherwise, the pricing functions $f\left(t^{\langle term\rangle}\right)$ taking the SLA term's value as parameter are used to compute the price increases $P^{\langle term\rangle}$ for the serviced client number, instantaneous non-interruption ratio, and the total non-interruption ratio (see Section 3.2), as follows:

$$P^{\langle term\rangle} = \frac{t^{\langle term\rangle} - t^{\langle term\rangle}_{\min}}{t^{\langle term\rangle}_{unit}} \cdot p_u^{\langle term\rangle} \cdot f\left(t^{\langle term\rangle}\right),$$

where $t^{\langle term\rangle}$ is the SLA term requested by the game provider, $t^{\langle term\rangle}_{\min}$ is the lower limit of the term's range, $t^{\langle term\rangle}_{unit}$ is the size of a term unit, and $p_u^{\langle term\rangle}$ is the price for one term unit. We define the *term units* as follows: one for the serviced client number, 0.01 for the instantaneous non-interruption ratio, and 0.001 for the total non-interruption ratio. The final price $P_{\text{O-SLA}}$ the game provider is charged when accepting the O-SLA is computed as follows:

$$P_{\text{O-SLA}} = P_{base} + [P_{cli} + (P_{ini} \cdot t_{cli}) + (P_{tni} \cdot t_{cli})] \cdot V_{coeff},$$

where P_{base} is the base price, P_{cli}, P_{ini} and P_{tni} represent the price increases for the serviced client number, instantaneous non-interruption ratio, respectively the total non-interruption ratio terms, t_{cli} represents the serviced client number requested by the game provider, and V_{coeff} is the *validity period coefficient* computed as follows:

$$V_{coeff} = \left\lceil \frac{v}{v_{\min}} \right\rceil \cdot f_{validity}(v),$$

where $\lceil\cdot\rceil$ denotes the ceiling function, v represents the validity period requested by the game operator, $v_{\min}$ represents the lower limit of the validity period range and also the validity period unit size (thus determining the time granularity of the given template), and $f_{validity}$ represents the validity price variation function.

After being negotiated, the provider tries to enforce the O-SLA terms for the entire interaction with the clients and

the game operator. To achieve this, the SLA monitor collects and aggregates data from two sources: (1) the game operator's QoS monitor providing monitoring data from MMOG servers and (2) the MMOG client software that regularly reports on the quality of game play on the client's desktop workstation (done in the background without user intervention). The SLA monitor eventually enforces the O-SLAs by compensating the clients according to the client contractual terms and by penalising the game operators in case any QoS value is below the values agreed upon. The correctness of this enforcement mechanism is ensured by the fact that the penalties are applied based on monitoring information cross-validated between the game operator and a large number of clients, ranging from hundreds to thousands.

3.3 Game Operator and Resource Provider

The business interaction between the game operator and the resource provider is again automatic and requires no human intervention. Based on the MMOG zones received from the game provider together with the associated O-SLAs, the game operator negotiates with different resource providers appropriate Cloud resources on which to host and run the zones using the model presented in Section 2.3.2. The result of this interaction is a *Resource SLA* (R-SLA) with the following structure:

- *issuer* or the resource (Cloud) provider;
- *geographical location* of the issuer's data centre;
- *resource bulk* representing the set of rented resources comprising the processor speed, memory size, internal and external network connection bandwidth;
- *validity period* representing the time for which the resources are available to the game operator from the time the R-SLA is accepted (usually hourly-grained and seldom weekly or monthly-grained);
- *compensation terms* in case of resource faults;
- *price* representing the requested non-negotiable price.

The R-SLA terms have fixed non-negotiable values and, therefore, there is no negotiation involved but a simple request-offer matching algorithm employed by the game operator.

A final aspect worthwhile mentioning is that Cloud providers often use fuzzy terms such as "the equivalent CPU capacity of a $1.0-1.2$ Gigahertz 2007 Opteron or Xeon processor" in case of Amazon EC2 or "vCPU" units in case of FlexiScale for describing processor performance. This makes the resource descriptions in the R-SLA resource bulk section fuzzy as well. Quantifying the quality of the offers becomes therefore cumbersome and defining finer-grained compensation terms other than for resource downtime (e.g. lower processor performance or network bandwidth) impossible. A solution for eliminating this fuzziness is to employ resource quality benchmarks before establishing R-SLAs with new Cloud providers, such as the RS unit benchmark employed by us in Section 4.1.2.

4. EVALUATION

In this section we present an evaluation of our MMOG operational and business model described in Section 3.

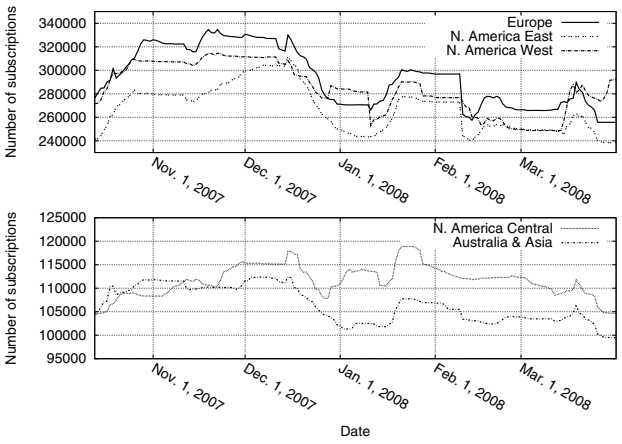

Figure 2: Number of RuneScape client accounts with daily resolution.

Cloud provider	*VM types*	*Locations*	*Price [$/ RSU/h]*	*GB/h]*	*Valid. [h]*	*VM inst. [seconds]*
Amazon	6	4	1.21	0.81	1	[65, 105]
CloudCentral	5	1	11.07	35.25	1	[50, 120]
ElasticHosts	4	1	1.22	2.73	1	[45, 120]
FlexiScale	4	1	0.72	1.46	1	[40, 50]
GoGrid	4	1	2.07	7.15	1	[60, 120]
Linode	5	1	0.67	2.37	24	[45, 120]
NewServers	5	1	0.38	0.71	1	[30, 120]
OpSource	6	1	0.09	0.15	1	[300, 540]
RackSpace	4	2	1.54	5.56	1	[100, 300]
ReliaCloud	3	1	0.96	1.04	1	[45, 60]
SoftLayer	4	3	0.70	1.75	1	[180, 300]
SpeedyRails	3	1	1.76	8.43	24	[80, 120]
Storm	6	2	0.99	1.54	1	[600, 900]
Terremark	5	1	1.40	6.14	1	[40, 60]
Voxel	4	3	0.83	0.94	1	[300, 600]
Zerigo	2	1	1.96	3.16	1	[60, 120]

Table 1: Summary of the modelled Cloud providers.

4.1 Experimental Setup

4.1.1 MMOG Traces

We use traces from RuneScape, a real MMOG ranked second after World of Warcraft by number of active paying customers in the US and European markets. We collected the traces for a period of six months from 150 servers spread across four continents. We run simulation experiments in which the game provider provisions O-SLAs according to the number of active client accounts given by the traces, as shown in Figure 2. The actual client numbers O-SLA requirements are computed utilising the *concurrent active account ratio*, representing the fraction of clients that are concurrently connected to the MMOG session.

4.1.2 Cloud Resources

The resource set used in our experiments is based on 16 of the major Cloud providers on the current market and comprises more than 100 different VM types, summarised in Table 1. The VM instantiation overhead intervals define the variation for different VM instances. The prices are hourly and are presented twice: first relative to the CPU power and second relative to the memory. The prices include the upstream and downstream network traffic, hence high network traffic prices have an important impact on the final R-SLA prices (as is the case for CloudCentral). While the geogra-

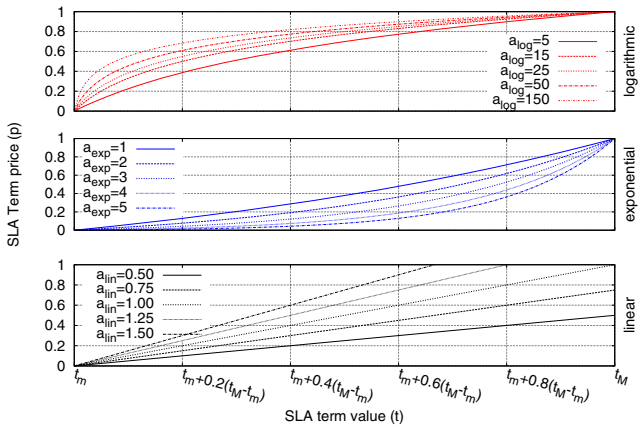

Figure 3: O-SLA template term pricing functions. $[t_m, t_M]$ represents the O-SLA term value range.

Plan name	*Payment method*	*Subscription price [$]* 1 m.	2 m.	3 m.	6 m.
SCC	Credit card	5.95	11.9	17.85	35.7
SPP	PayPal	7.5	12.99	18.15	36.10
SBT	Bank transfer	7.99	14.59	19.99	37.99

Table 2: RuneScape subscription plans for monthly, bimonthly, trimestrial, and semestrial payments.

phical location, the memory size, and the price are clearly defined by all providers, the processing power of the offered virtual processing units 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 computed this metric and the virtualisation overheads based on data resulted from our own investigations and other Cloud performance evaluations [3, 8, 16].

4.1.3 Business Contracts

For the clients-game provider relationship, we utilise the real prices of Jagex Ltd., the developer and publisher of the RuneScape, as of August 2010 for the different subscription plans and payment methods, summarised in Table 2. As described in Section 3.2, the game providers have different policies for acquiring O-SLAs, presented in Table 3. For game operators, we employ a set of O-SLA template offers presented in Table 4 and vary one or more of their terms. Each O-SLA template has associated three types of pricing functions displayed in Figure 3, which are used in the negotiation of the final price with the game provider: exponential, logarithmic, and linear, whose shapes are controlled by a coefficient a. Finally, we use R-SLAs corresponding to the Cloud resources described in Section 4.1.2 For these simulations, we consider all resources to have 100% of uptime, thus we did not include compensation terms in the R-SLAs.

4.1.4 Evaluation Metrics

We analyse the financial aspect of our MMOG operation model using three metrics:

- *gross profit*, representing the difference between the business actor's revenue and the cost of providing its services, excluding taxation and any other overheads;
- *penalty*, a fraction of the profit representing the total cost a business actor pays as compensation for any SLA faults for the entire simulation period;
- *penalty events*, a breakdown of the penalty metric which represents the penalties for all SLA term fault instances. The sum of all penalty events represents the previously defined total penalty.

Policy name	*Concurrently active accounts*	*Min. instant. non-interrupt.*	*Min. total. non-interrupt.*	*Validity period [h]*
PP1	19%	0.90	0.995	[24; 168]
PP[2:12]	[15%:25%]	0.90	0.990	[24; 168]
PP6	19%	0.90	0.990	[24; 168]
PP9	22%	0.90	0.990	[24; 168]
PP13	19%	0.85	0.990	[24; 168]

Table 3: Game provider O-SLA selection policies.

The QoS aspect is analysed through three metrics:

- *average non-serviced clients*, representing the average number of clients which are denied service within a measurement timestep because of improper O-SLA provisioning by the game provider or because of improper resource allocation by the game operator (e.g. in case the provisioned Cloud resource startup time is too long and the game world zone instantiation is delayed, making it unavailable to clients);
- *instantaneous interruption ratio*, representing the percentage from one measurement timestep in which a subset of clients (usually all clients connected to the affected game server) receive low QoS;
- the *total interruption time*, representing the total duration of the O-SLA validity for which a subset of clients receive low QoS.

4.2 Contribution of O-SLA Pricing to Gross Profit

This experiment investigates the relation between the price of each O-SLA term and the gross profit of the game providers and game operators. We set up a simulation with one game provider employing the PP1 O-SLA selection policy (see Table 3) and the SCC subscription plan, and one game operator which sequentially utilises the O-SLA range OSLA[1..24] from Table 4. The first O-SLA range OSLA[1..6] varies the base price term between $200 and $450 in $50 increments. For evaluating the impact of the serviced clients term price, the OSLA[7..12] range changes the price per client from $0.05 to $0.3 in $0.05 increments. Similarly, for the instantaneous and total non-interruption term pricing impact, the OSLA[13..18] and OSLA[19..24] ranges vary the price per instantaneous, respectively total non-interruption units between $0.005 and $0.03 with a $0.005 step. We run 24 experiments, each covering six months of MMOG operation with a different O-SLA from the described ranges.

Because the client subscription plans are constant throughout this set of experiments, the client expenditures are constant of around $38 million at the end of the simulation period. The different O-SLA term prices affect the distribution of this sum between the game provider and the game operator as shown in Figure 4. In Table 5 we present the quantification of the linear variation of the gross profits with the O-SLA prices. The *impact of term price on profit* shows the increase in the game operator's monthly income

O-SLA name	O-SLA terms: Base price[$]	Serviced clients	Inst. non-interrupt.	Tot. non-interrupt.	Validity [h]	Pricing functions: Serviced clients	Instant. non-interruption ratio	Total non-interruption ratio	Validity period
OSLA[1..6]	200..450	[2,000; 20,000]	[0.85; 0.95]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.15]	log[a=5,p_u=0.015]	log[a=5,p_u=0.01]	lin[a=1]
OSLA[7..12]	225	[2,000; 20,000]	[0.85; 0.95]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.05..0.3]	log[a=5,p_u=0.015]	log[a=5,p_u=0.01]	lin[a=1]
OSLA[13..18]	225	[2,000; 20,000]	[0.85; 0.95]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.15]	log[a=5,p_u=0.005..0.03]	log[a=5,p_u=0.01]	lin[a=1]
OSLA[19..24]	225	[2,000; 20,000]	[0.85; 0.95]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.15]	log[a=5,p_u=0.015]	log[a=5,p_u=0.005..0.03]	lin[a=1]
OSLA25	225	[2,000; 20,000]	[0.85; 0.99]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.07]	log[a=5,p_u=0.02]	log[a=5,p_u=0.01]	lin[a=1]
OSLA26	250	[2,000; 20,000]	[0.85; 0.99]	[0.99; 0.999]	[24, 168]	exp[a=1.5,p_u=0.1]	exp[a=1.2,p_u=0.015]	exp[a=1.2,p_u=0.04]	lin[a=1]
OSLA27	600	[2,000; 20,000]	[0.85; 0.95]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.025]	log[a=5,p_u=0.015]	exp[a=1.1,p_u=0.01]	lin[a=1]
OSLA28	600	[2,000; 20,000]	[0.85; 0.95]	[0.99; 0.999]	[24, 168]	exp[a=1.2,p_u=0.525]	log[a=5,p_u=0.015]	exp[a=1.1,p_u=0.01]	lin[a=1]

Table 4: O-SLA templates and associated pricing functions, where a represents the function shape coefficient and p_u the price per SLA term unit.

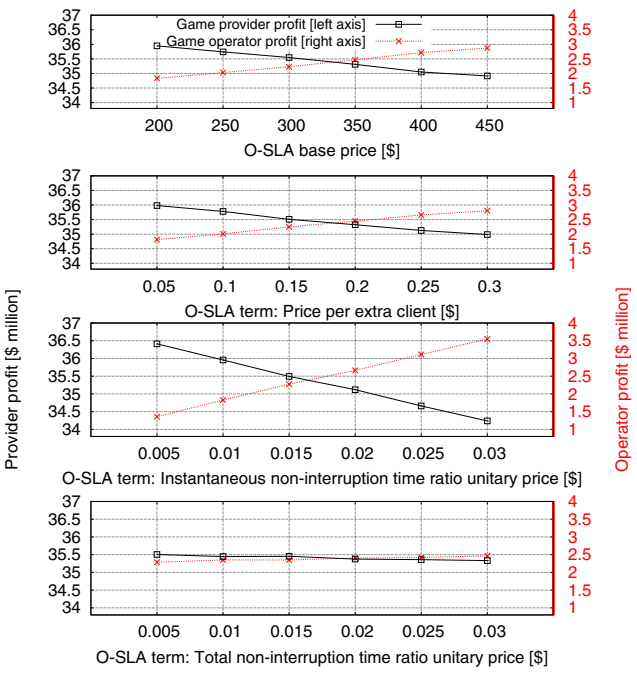

Figure 4: Impact of O-SLA term pricing on the game provider's and game operator's gross profits.

O-SLA term name	Impact of term price on profit	Impact of term price on profit per 1000 accounts
Base price	688.38 $/$	0.65 $/$/Kaccounts
Price per extra client	6630.90 $/$ct	6.23 $/$ct/Kaccounts
Inst. non-interruption	144938.07 $/$ct	136.09 $/$ct/Kaccounts
Total. non-interruption	11445.52 $/$ct	10.75 $/$ct/Kaccounts

Table 5: Variation of the game provider's monthly expenses and the game operator's monthly income with the O-SLA term pricing (a hypothetical impact of 20 $/$ signifies an increase of $20 in the game provider's monthly expenses and an equal increase in the game operator's monthly income for each $1 increase of the respective O-SLA term unitary price).

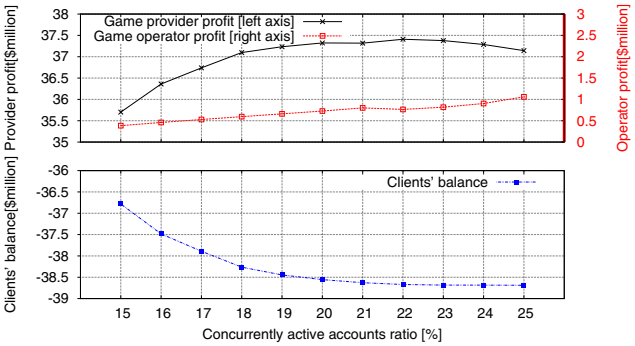

Figure 5: Variation of game provider's gross profit, game operator's gross profit, and clients' total expenses with the concurrently active account ratio.

and the corresponding increase in the game provider's expenses determined by the increase of an O-SLA term price. This metric is measured in [$/$]. Considering the fact that the resource usage is directly proportional to the number of clients connected to the MMOG servers and by extension, to the number of active accounts, we also show the values on a per-client basis. The metric's measurement unit becomes [$/$/active account]. These values can be utilised by the game operators for adjusting their O-SLA pricing policies for achieving their profit targets and by the game providers to estimate the impact different O-SLA pricings will have on their monthly expenses.

4.3 Maximising Game Provider's Profit

As described in Section 3.2, the game provider computes the requirements for game zone operation in terms of the number of clients to be handled by each game operator, based on which it provisions the necessary O-SLAs. These requirements are estimated as a ratio of concurrently active accounts from the total number of subscriptions for each MMOG. The goal of this experiment set is two fold: (1) to determine the best policy for estimating the concurrently active account ratio that minimises the denial of service QoS events, and (2) to determine the correlation between the game provider's profit and the accuracy of the estimated concurrently active account ratio. To investigate these aspects, we set up an experiment consisting of a game operator using the `OSLA25` template, a game provider employing the O-SLA policies `PP[2-12]` in sequence, and the same client accounts shown in Figure 2 as input, with a measured concurrently active accounts ratio between 19% and 22%.

The game operator's and the game provider's gross profits along with the clients' total expenses are plotted in Figure 5. The game provider's profit shows a consistent increase of 20.2% in the interval $[15\%, 22\%]$ of concurrently active accounts ratios, and then a slight decrease of 2.5% in the $[22\%, 25\%]$ range. The profit increase is explained by the fact that the provider gradually purchases better O-SLAs

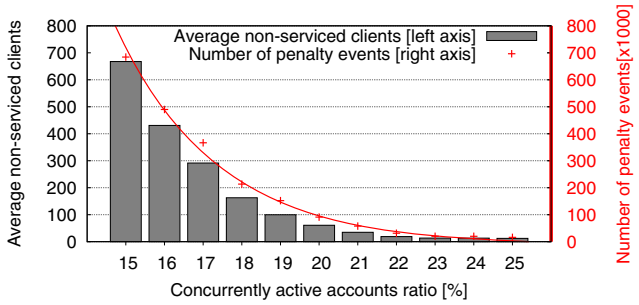

Figure 6: Game provider's denial of service penalty events (left axis: average number of clients with denied service; right axis: number of denial of service penalty events).

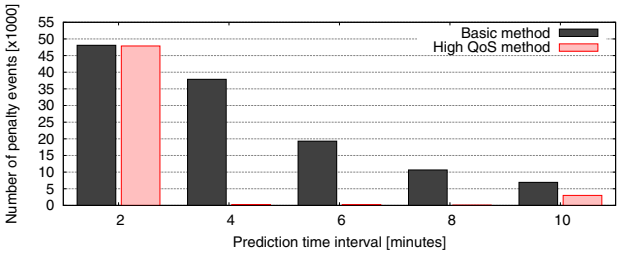

Figure 7: Variation of the game operator's denial of service penalty events with the prediction interval and resource provisioning method.

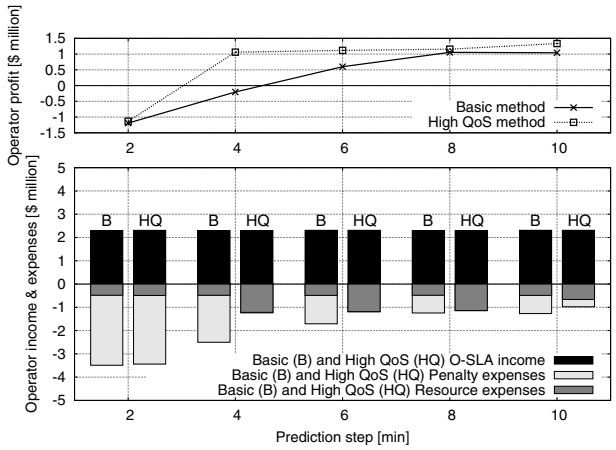

Figure 8: Variation of the the game operator's gross profit and its fractions with the prediction interval and resource provisioning method.

that drastically reduce the number of penalty events resulting from denying service to clients, as the estimated concurrently active account ratio gets closer to the real value (see Figure 6). Overestimating the concurrently active account ratio has moderate negative effects on the provider's budget because investing in larger O-SLAs does not result in significant decreases in penalty events and consequently, in non-significant cost reductions for the provider. Regarding the clients' accounting balance, the initial investment made by the clients is roughly identical for all runs, the variation observed in Figure 5 being a result of the game provider compensating the clients for denial of service events[1].

4.4 Maximising Game Operator's Profit

Uninitialised MMOG servers is the other cause for the denial of service QoS events which originates at the game operator. We identified two nonorthogonal techniques that can be used in combination and which the game operator can apply to prevent these low QoS events: (1) increasing the load prediction time interval (see Section 2.3.2) and keeping the allocation instant fixed, thus increasing the time the new resource has available for initialisation, and (2) employing a resource provisioning method called from here on *high QoS* which identifies and prioritises the allocation of resources with low initialisation overhead. In this experiment we evaluate the efficiency of these two methods by analysing the denial of service penalty events generated by the game operator to the clients and their impact on the operator's gross profit. The setup consists of a game provider employing the `PP9` policy (see Table 3), a game operator providing the `OSLA26` operation template (see Table 4), and the Cloud resources modelled in Table 1. We first set the game operator's resource allocation algorithm to the *basic* method that ignores all virtualisation overheads, and then to the high QoS method. For each method, we vary the prediction time interval from two to ten minutes.

Figure 7 shows the number of penalty events generated by denial of service faults originating at the game operator. We observe a steady decrease in the number of penalty events with the increase of the prediction interval when employing the basic provisioning method. The best result is an 85.6% reduction in the number of penalty events in case of the ten minute prediction interval compared to the original two minute interval. In the case of the high QoS provisioning method, the penalty events are almost completely eliminated for the eight minute prediction interval, representing a 99.9% decrease relative to the two minute prediction interval. The registered QoS improvement when employing the high QoS method as opposed to the basic one is of 71% on average. The second important aspect of this evaluation is the impact of the proposed techniques on the game operator's gross profit and its fractions in both cases, charted in Figure 8. We observe that the elimination of the penalty events for the high QoS method is achieved by using more expensive but better quality resources, which eventually leads to a higher gross profit.

We conclude that increasing the prediction time interval results in a solid and constant improvement both in the QoS provisioned, as well as in the game operator's gross profit. The best results are obtained when utilising the high QoS method which maximises the game operator's profit (starting with a relatively short prediction interval), while simultaneously providing an almost flawless QoS to the clients.

4.5 Different Game Type Operation

The goal of this experiment is to demonstrate that our proposed MMOG operation model can be successfully employed for the four important game types introduced in Section 2.4. We set up a simulation scenario with a game provider utilising the `PP13` policy (see Table 3) and a game

[1]The state-of-the-art client–provider contracts frequently offer free service to the clients instead of financial compensation, however, we employ here the latter method for an easier understanding of the money-flow between the actors.

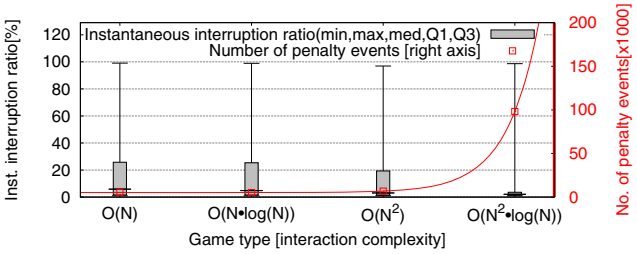

Figure 9: QoS evaluation for different game types.

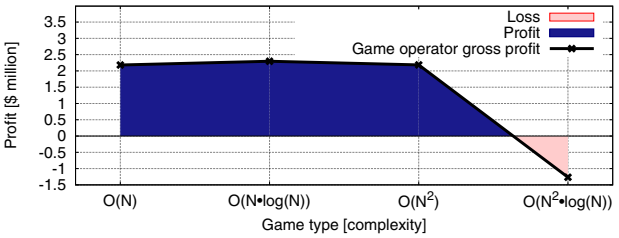

Figure 10: Variation of game operator's gross profit for different game types.

operator with the OSLA27 operation template (see Table 4). We run the simulation multiple times using the same traces from Figure 2, each time changing the type of simulated RuneScape game worlds and consequently, the load model the game provider exposes to the operators.

We utilise the instantaneous interruption ratio and the total number of penalty events to analyse the QoS of our MMOG operation model. Figure 9 shows a relatively constant QoS for the three RuneScape game types, with the median interruption ratio lower than 8% and the total number of penalty events below 7000. For very high interaction games, however, even though the instantaneous interruption ratio events are low in intensity, the total number of penalty events grows exponentially. As a result, there are significant financial losses for the game operator hosting very high interaction game types. This can be seen in the game operator's gross profit charted in Figure 10, which shows constant profit for the first three simulations and substantial losses in the case of $O\left(N^2 \cdot \log(N)\right)$ interaction complexity games.

We conclude that our model shows good behaviour for RuneScape game types with computational complexities between $O(N)$ and $O\left(N^2\right)$ in terms of QoS and game operator gross profit, while for the very high $O\left(N^2 \cdot \log(N)\right)$ complexity the QoS is substantially lower and the operator registers losses. In the next section, we present a method to significantly improve the efficiency of our model in this latter case.

4.6 Maximising QoS through Controlled Over-allocation

In the previous experiment, we identified an issue with hosting very high interaction MMOG servers, as the game operator registers financial losses due to significantly higher resource demand and high amount of penalties. As described in Section 2.5, the high number of QoS breaches can be diminished through a controlled over-allocation technique that we studied in [12]. To investigate its impact on the game operator's budget and QoS, we started an experiment similar to the one employed in the previous section: a game operator

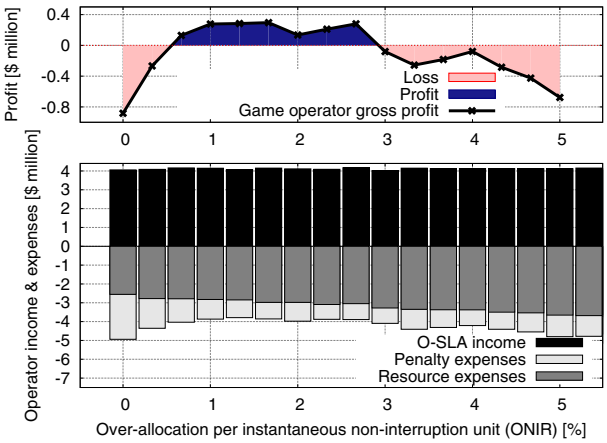

Figure 11: Variation of the game operator's gross profit and its fractions with the resource over-allocation per instantaneous non-interruption unit.

ONIR [%]	*Expenses [%]*		
	Penalties	*Resources*	***Total***
0.33	-33.91	+8.80	-11.83
0.67	-47.82	+9.19	-18.35
1.00	-56.20	+10.61	-21.67
1.33	-60.55	+11.51	**-23.30**
1.67	-63.17	+16.64	-21.91
2.00	-58.21	+16.72	-19.48
2.33	-67.00	+21.04	-21.49
2.67	-64.61	+19.37	-21.20
3.00	-65.76	+28.31	-17.13
3.33	-55.63	+31.10	-10.80
3.67	-60.74	+32.08	-12.76
4.00	-65.12	+32.27	-14.78
4.33	-61.90	+37.02	-10.77
4.67	-57.61	+38.48	-7.94
5.00	-52.17	+43.24	-2.85

Table 6: Game operator's expenses when employing controlled resource over-allocation.

with the OSLA27 template and a game provider simulating the very high interaction $\left(O\left(N^2 \cdot \log(N)\right)\right.$ game complexity. We associate degrees of resource over-allocation to the instantaneous non-interruption units (the unit's value being 0.01) starting from 0% as in the previous section, and gradually increase it until the 5%. For convenience, we abbreviate this newly introduced metric as *ONIR*. We utilise the PP6 O-SLA selection policy which differs from the previously employed PP13 by five instantaneous non-interruption units and thus, the effective resource over-allocation ranges from 0 to 25% throughout this experiment set.

Figure 11 (top) shows a significant increase in the game operator's gross profit for low ONIR values, followed by a relatively constant region and a descending trend. The detailed fractions of the operator's profit depicted in Figure 11 (bottom) clarify this behaviour. We observe a significant reduction in the amount of penalties for the first section of the chart until an ONIR value of 2.66%, from where the penalty expenses remain relatively constant which represents the low service availability penalties. The resource expenses exhibit a steady increase, proportional to the variation of the ONIR throughout the entire tested range.

We conclude that, as described in Section 2.5, controlled over-allocation can effectively improve the quality of game play. Regarding the game operator's profit, controlled over-

Game type	*Operator profit[$]*	*Provider profit[$]*	*Client expenses[$]*	*Income per client [$]*
$O(N)$	2,182,237	35,661,983	38,421,087	6.01
$O\left(N \cdot \log(N)\right)$	2,295,945	35,574,863	38,422,199	6.01
$O\left(N^2\right)$	2,184,772	35,653,816	38,405,335	6.01
$O\left(N^2 \cdot \log(N)\right)$	2,250,625	35,557,978	40,701,448	**6.36**

Table 7: Total game operator and provider gross profits, the clients' expenses for a six month period and the resulting monthly provider income per client.

Plan name	Payment method	Subscription price [$] 1 m.	2 m.	3 m.	6 m.
SCC1	Credit card	$6.3	$12.6	$18.9	$37.8
SPP1	PayPal	$7.92	$13.72	$19.17	$38.13
SBT1	Bank transfer	$8.41	$15.35	$21.03	$39.97

Table 8: Client subscription pricing plan for next generation very high interaction MMOGs.

allocation is a beneficent technique as long as the reduction in penalty expenses is lower than the increase of the resource provisioning costs. In our particular case, the quantification of the profit fraction variations presented in Table 6 demonstrated that the optimal ONIR value range is $[1\%; 2.67\%]$ for which the operator's gross profit is roughly around $\$300,000$. Although employing this method brings a 23% reduction in the operator's expenses, the final gross profit does not reach the $2 million obtained when hosting the other MMOG types. This is due to the very high computational complexity of this game type, reflected in the fourfold growth of the resource provisioning expenses. In the final experiment set presented in Section 4.7, we quantify the impact of operating very high computational intensity MMOG servers on the client subscription prices.

4.7 Very High Interaction MMOG Subscription Price

Building on the conclusions of the last experiment, we run a final simulation with the goal of evaluating the impact of operating very high interaction MMOGs on the client subscription prices. We set the gross profit targets for the game provider and operator to the values reached when operating the three RuneScape game types. We have previously shown that very high interaction games have a negative impact on the game operator's profit. Thus, for enforcing the operator's profit target, we compensate for this loss by employing `OSLA28` which has an O-SLA term pricing adjusted by utilising the O-SLA term pricing quantification determined in Section 4.1.3 (see Table 5). For enforcing the provider's gross profit target, the client subscription prices are dynamically modified by our simulator to maintain the provider's gross profit proportional to the initial subscription models.

The outcome of this final experiment is summarised in Table 7, which shows that the profit margins for both game operator and game provider can be maintained constant when operating the very high interaction MMOG type by increasing the average monthly income from the client subscriptions with $0.35, meaning a 5.8% increase relative to the original income. The new detailed subscription pricing plan is presented in Table 8.

5. RELATED WORK

In the recent years, stimulated by the rapid market growth and the interesting challenges this topic brings for the scientific community, MMOG operation has received much attention. Wong [19] proposed a resource provisioning algorithm using fuzzy linear assignment with the main objective of ensuring QoS guarantees. In contrast to our work, it focuses on networking aspects, while we take into consideration a resource model which includes more parameters, like computation, memory, and virtualisation overheads present in Cloud resources today. Briceno [10] studies resource allocation methods for MMOGs focused on their computational requirements and approximates the computational complexity of MMOGs as quadratic relative to the number of connected clients assuming constant communication times. Our work studies different MMOG types assuming variable server computational complexities and variable communication time and volume. Another study of efficient resource utilisation is done by Lee [9] who proposes a zone-based MMOG server consolidation technique which, in contrast to our work, focuses on the energy consumption aspect.

Another approach to distributing MMOG load is designing them as peer-to-peer applications, as proposed by Douglas et al. [4] and Fan et al. [5]. Although the peer-to-peer model has some advantages, there are two downsides which make it difficult to adopt: (1) the need for a complete redesign of the networking platform and implicitly of the MMOG development frameworks and (2) the security risks (e.g. against cheating) of having parts of the MMOG session hosted on unsecured hardware. This latter issue is addressed by Picone et al. [17], but restricted to RTS games. Our MMOG operation model is minimally invasive in the MMOG design requiring only instrumentation for QoS monitoring which many existing games already offer [6].

Our proposed business model for MMOGs is closest in concept to the one introduced by Middleton et al. [11], which is also based on four business actors whose business interactions are regulated by bipartite SLAs and client accounts. In contrast to research, this work does not study the connection between the business and the hosting models, and the methods of controlling the provided QoS. Another focal issue regarding the MMOG business is the pricing model employed by the game providers. A study of the relationship between the pricing models and the clients' motivation for playing MMOGs is presented by Nojima [14], who evaluates the customer satisfaction relative to a large set of parameters characterising the clients and the pricing models. On the same topic, Oh [15] analyses the benefits of employing a different pricing model for clients who, instead of paying for time-based subscriptions, need to purchase in-game features (e.g. items, character abilities, access to special game worlds). These studies focus on the client – game provider business interaction, while our work extends it to the game provider – game operator and game operator – resource provider business relationships. More complex business models of Alves et. al [1] and Andersson et. al [2] analyse the higher-level business interactions and goals for MMOG operation, while we study the effects of a novel operation model on the gross profits of the involved actors.

The pervasiveness of the Cloud hosting model presents an incentive for the entertainment industry to migrate from the in-house model of hosting and operation towards the IaaS model. Such attempts are currently emerging from the industry side, the front-runner being the on-demand gaming platform OnLive(`http://www.onlive.com`) followed by competi-

tors like Gaikai(http://www.gaikai.com) and OTOY(http://www.otoy.com) which are still in the development and testing stages. These approaches, in contrast to our proposed model, are restricted to offloading of computation from the client graphical programs running on the clients' personal computers to privately owned Cloud resources. The game-play is carried on by compressing the game graphics into a light network stream which can be decompressed even on machines with minimal resources. There is also a similar approach from Geelix(http://www.geelix.info), an U.S. based company which published scientific aspects behind this remote game play technology in [7].

6. CONCLUSIONS AND FUTURE WORK

Computational Clouds remain highly specialised technologies that are only used by scientists and large commercial organisations. The proposed research is unusual compared with previous academic research projects by addressing a new class of applications that appeal to the public for leisure reasons. Online games have the potential to raise strong interest, providing societal benefits through increased technological awareness and engagement.

We proposed a new business model of hosting and operating MMOGs consisting of four main actors, clients, game providers, game operators, and resource providers, which efficiently provision on-demand virtualised Cloud resources to MMOG servers based on the dynamic client load. We have shown that this model dramatically decreases prices and gives small and medium enterprises the opportunity of joining the market with zero initial investment. Through simulations using trace data collected from one of the largest MMOGs on the market, we demonstrated that our model can operate state-of-the-art MMOGs with an average monthly gross profit of nearly $6 million excluding game purchase prices, overheads and taxation, while providing high QoS to all clients. Furthermore, we have shown that our approach is capable of operating next generation very highly interactive MMOGs with a small increase of only 5.8% in the subscription price. In future work we intend to study the effects of employing various penalty policies between all business actors.

7. REFERENCES

[1] T. Alves and L. Roque. Using value nets to map emerging business models in massively multiplayer online games. In *Proceedings of the Ninth Pacific Asia Conference on Information Systems (PACIS)*, 2005.

[2] B. Andersson and et. al. On the alignment of goal models and business models. In *REA 25:A Celebration of the REA Enterprise Ontology*, June 2007.

[3] CloudHarmony. What is an ECU? CPU benchmarking in the Cloud. http://blog.cloudharmony.com/2010/05/what-is-ecu-cpu-benchmarking-in-cloud.html, May 2010.

[4] S. Douglas et al. Enabling massively multi-player online gaming applications on a p2p architecture. In *In Proceedings of the IEEE International Conference on Information and Automation*, pages 7–12. IEEE, 2005.

[5] L. Fan, H. Taylor, and P. Trinder. Mediator: a design framework for p2p MMOGs. In *NetGames '07: 6th ACM SIGCOMM Workshop on Network and System Support for Games*, pages 43–48. ACM, 2007.

[6] F. Glinka et al. RTF: A real-time framework for developing scalable multiplayer online games. In *NetGames '07*, pages 81–86. ACM, 2007.

[7] O.-I. Holthe, O. Mogstad, and L. A. Ronningen. Geelix livegames: Remote playing of video games. In *6th IEEE Consumer Communications and Networking Conference (CCNC)*, 2009. ISBN 978-1-4244-2308-8.

[8] C. Keck and RackSpace. Cloud hosting provider hardware benchmarks. http://www.chadkeck.com/2010/05/cloud-hosting-provider-hardware-benchmarks/, May 2010.

[9] Y.-T. Lee and K.-T. Chen. Is server consolidation beneficial to MMORPG? A case study of World of Warcraft. *Cloud Computing, IEEE International Conference on*, 0:435–442, 2010.

[10] L. D. Briceño and et. al. Robust resource allocation in a massive multiplayer online gaming environment. In *4th International Conference on Foundations of Digital Games (FDG)*, pages 232–239. ACM, 2009.

[11] S. Middleton, M. Surridge, B. Nasser, and X. Yang. Bipartite electronic SLA as a business framework to support cross-organization load management of real-time online applications. In *Real Time Online Interactive Applications on the Grid (ROIA)*, 2009.

[12] V. Nae, A. Iosup, and R. Prodan. Dynamic resource provisioning in massively multiplayer online games. *IEEE Transactions on Parallel and Distributed Systems*, 99(82):–, 2010.

[13] V. Nae, A. Iosup, R. Prodan, and T. Fahringer. The impact of virtualization on the performance of massively multiplayer online games. In *NetGames'09*, pages 1 – 6, Paris, Nov. 2009.

[14] M. Nojima. Pricing models and motivations for MMO play. In B. Akira, editor, *Situated Play: Proceedings of the 2007 Digital Games Research Association Conference*, pages 672–681, Tokyo, September 2007.

[15] G. Oh and T. Ryu. Game design on item-selling based payment model in Korean online games. In B. Akira, editor, *Situated Play: Proceedings of the 2007 Digital Games Research Association Conference*, pages 650–657, Tokyo, September 2007.

[16] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema. An early performance analysis of Cloud computing services for scientific computing. ISSN 1387-2109 PDS-2008-006, Delft Univ. of Technology PDS Report Series, 2008.

[17] M. Picone, S. Sebastio, S. Cagnoni, and M. Amoretti. Peer-to-peer architecture for real-time strategy MMOGs with intelligent cheater detection. In *SIMUTools'10: Proc. of the 3rd Intl. ICST Conference on Simulation Tools and Techniques*, pages 1–8, 2010.

[18] R. Prodan and V. Nae. Prediction-based real-time resource provisioning for massively multiplayer online games. *Future Generation Computer Systems (FGCS)*, 25(7):785–793, 2009.

[19] K. Wong. Resource allocation for massively multiplayer online games using fuzzy linear assignment technique. In *5th IEEE Consumer Communications and Networking Conference (CCNC)*, pages 1035–1039. IEEE, 2008.