A model and software architecture for MMORPG traffic generation based on player behavior

Mirko Suznjevic · Ivana Stupar · Maja Matijasevic

Received: date / Accepted: date

Abstract In this paper we present a software architecture for traffic generation based on application level player behavior in Massively Multiplayer Online Role-Playing Games (MMORPGs). We have performed measurements of network traffic for each of the previously defined action categories for MMORPGs *(Trading*, *Questing*, *Dungeons*, *Raiding*, *Player versus Player Combat*, and *Uncategorized)*, as well as measurements of application level player behavior in terms of listed action categories. Based on the obtained datasets we have created network traffic models for each action category and player behavior models with focus on hourly and daily trends. Network traffic models are implemented in Distributed Internet Traffic Generator, and verified through comparison with the real traffic. Player behavior models explore number of active players, session duration, as well as lengths and probability of session segments (i.e., parts of the session consisting of only one category of player actions). We propose an architecture which enables scalable, behavior driven traffic generation and implement it in a laboratory testbed. In order to achieve scalability of the system, we use Linux Containers as a virtualization technique. The resulting implementation can generate hundreds of MMORPG streams on a single PC. As a case study we used *Activision Blizzard's World of Warcraft.*

Keywords MMORPG · Traffic generation · User behavior model · Network traffic model · Network · Game

Mirko Suznjevic
University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000, Zagreb
Tel.: +385-16129-755
Fax: +385-16129-832
E-mail: mirko.suznjevic@fer.hr

Ivana Stupar
E-mail: ivana.stupar@fer.hr

Maja Matijasevic
University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000, Zagreb
E-mail: maja.matijasevic@fer.hr

1 Introduction

Massively Multiplayer Online Role-Playing Games (MMORPGs) are a genre of online games in which the player typically controls a virtual character (avatar) which represents him/her in a persistent virtual world. MMORPGs are one of the most popular types of online games, and have a significant impact on the network. While they are, in general, low-bandwidth applications, there is a tendency of increase in bandwidth usage as the genre evolves [26]. Also, new MMORPGs, such as *Guild Wars 2* by *ArenaNet*, promise raising the scale in virtual battles (e.g., thousands of players participating in *World versus World* battles), which will also significantly increase the network load.

As the number of MMORPG titles in the market is growing [6], game providers are trying to keep the existing player base, but also to attract new players. Service providers need to ensure that players have a satisfying level of Quality of Service (QoS). Networking is one of the most important aspects in the overall performance of the virtual world. In order to be realistic and immersive, a virtual world needs to be responsive, which, on the network level, means that the latency values need to be low, and game data must be delivered in a timely manner. Without network delivering adequate QoS, all other aspects of an MMORPG become insignificant as the virtual world stops being responsive in real time.

Game server stability and performance are another important aspect of the QoS for MMORPGs.

Knowing the properties of network traffic generated by online games is crucial for network operation as well as for network planning and dimensioning. Software and hardware traffic generators are indispensable tools for simulation, emulation and testing in real and emulated environments. Tools used for server load testing, for example LoadRunner (www.loadrunner.com), are capable of emulating thousands of concurrent users. Due to the complexity of online games some specific tools are developed specifically for game testing [15]. Regardless of the service type, traffic generators are essential for such testing systems in order to generate representative background traffic [41].

The main motivation for this work is to develop a scalable solution for traffic generation based on user behavior. We claim that traffic generation based on user behavior enables realistic and adaptive testing procedures. Also, behavior based traffic generation is applicable in the testing period before the game release, but also during the game lifespan. MMORPGs are dynamic systems which change constantly in the terms of content addition and technical updates. As our behavior model is created to be meaningful in terms of game design, it is possible to anticipate how changes in the game will effect user behavior as defined within our model (e.g., if in a new content, e.g., additional dungeons, are released, it will result in more players going into dungeons).

In this paper we propose an approach for network traffic generation based on application layer user behavior. We develop a network traffic model based on player behavior. We also present an architecture and implementation of a distributed system for traffic generation, named User Behavior Based Network Traffic Generator (UrBBaN-Gen), based on this model.

This paper presents a converging point of our previous research efforts. In order to enable user behavior based traffic generation we had to perform several steps. Starting from the relationship of what the user does in the virtual environment and how it affects the communication characteristics [34], we focused our further research efforts to MMORPGs. Firstly, we have defined classes of application layer user behavior which we named "action categories" and validated them through measurements of network traffic [46,48]. Secondly, we have formed network traffic models for each of the defined action categories [50]. Thirdly, we have measured and modeled player behavior with respect to the time of the day, and day in the week [45,49]. Additionally, we have investigated how much users communicate through both chat and voice [45,47]. As a case study we have used *World of Warcraft* (WoW) by *Activision Blizzard* [3]. As of November 2011, WoW is still estimated to be the most popular subscription based MMORPG with 11.1 million active players [22].

Logically, the paper can be divided into three areas. First is describing the network traffic level, traffic measurements, modeling, and implementation of the models. As a basis for our traffic generator, we selected Distributed Internet Traffic Generator (D-ITG), developed at the University of Naples Federico II [1]. Traffic models for each action category are implemented in a new application protocol as an extension within D-ITG. With respect to our previous research [50], we have improved our models by adding Lognormal distribution, and solved problems regarding the use of TCP, and generation of packets smaller than 20 bytes. The previously low goodness of fit for traffic generated over TCP has been improved through disabling Nagle's algorithm and delayed acknowledgments in TCP.

The second area describes measurements and modeling of application level user behavior. With respect to our previous paper [49], we have improved the model in several aspects. We have added another technique for modeling the number of players in order to address the problem of extreme variation of player count (e.g., surge of players due to a release of new content). We have performed additional measurements in order to further explore action category *Uncategorized* (which indicates the time that could not be assigned to any of the previously defined categories). We have modified our existing player behavior model in order to include *Uncategorized.* Finally, we have added the models of voice communication usage depending on the action category. We have added two improvements to the implementation of behavior models in our behavior simulator: 1) option to replay previously performed experiments, and 2) adjustments of various simulation parameters through a Graphical User Interface (GUI).

The third area describes the architecture of the proposed traffic generation system. We present control mechanisms for the distributed traffic generation, the API interfacing the simulation of application level user behavior to traffic generation control, and virtualization scheme which enables high scalability of the traffic generation process. Our architecture enables generation of hundreds of MMORPG streams on a single commodity PC.

The main contributions of this work are a player behavior based network traffic model of an MMORPG and a software architecture of a traffic generator as a realization of the given model. Classification of user actions in our behavior model covers the majority of possible player behaviors as compared to the related work in [35], which lacks cooperative group based actions. We create a traffic generator truly driven by application level player behavior, whereas other similar traffic generators, e.g., [41], consider highly erratic characteristics of gaming traffic, and not what the user does in the in-game virtual world.

2 Related work

In this section we briefly survey related research efforts in the areas of traffic modeling for games, and modeling of player behavior in the MMORPG. One of the initial papers in the field of traffic modeling for network games was by Borella [9] who analyzed the traffic from the First Person Shooter (FPS) game *Quake*. Modeling methodology used by Borella was initially introduced by Paxon in [36]. We use the same modeling methodology in this paper, but we apply it only on a portion of overall game traffic belonging to specific action category. Initial research efforts in modeling traffic for networked games were focused on the FPS genre, and in particular to the following games: *Quake 3* [31], *Quake 4* [16], *Halo* [29], *Halo 2* [57], *Half-Life* [30], *Counter-Strike* [17,20].

As MMORPGs gained popularity in the market, research focus shifted from FPS games and Real-Time Strategies (RTS) (such as *Starcraft* [18]), to persistent virtual worlds of MMORPGs.

Chen et al. analyze the network traffic of *ShenZhou Online* in great detail and with respect to user behavior [12]. The network trace on which the analysis is performed is very large (55TB), and it is captured on the server side with the assistance of the game operator. The following properties of the MMORPG traffic are determined: tiny packets, periodicity, significant signaling overhead, temporal dependence of packet arrivals within connections and aggregate traffic. Specific characteristics of network traffic are related to certain user behavior (e.g., practice of team play, user behavior diversity, etc.). Also, it is shown that time of the day has significant impact on session inter-arrival times. Authors state that the problem of modeling user behavior and source traffic in MMORPGs is especially challenging due to the diversity of user behaviors and do not model it. This research group also performed further analysis on their trace in terms of interdependence between network characteristics and players' behavior. In [14] Chen and Lei study implications of player interactions on generated network traffic. Distribution of the players in the virtual world is claimed to be heavy-tailed, which implies that the static and fixed side partitioning mechanisms for the game world are not adequate. With respect to player behavior, the authors state that players with the higher degree of social interaction tend to have longer sessions. In [13] the relationship between network QoS and session times is examined using the survival analysis. Chen et al. determine that both the network delay and the network loss have significant impact on player behavior in terms of players' willingness to continue playing or to leave the game.

Svoboda et al. analyze the traffic trace of WoW captured within the core network of a mobile operator [51]. Results show that WoW is amongst the top 10 TCP based services in the monitored network and that WoW traffic represents 1% of all TCP based traffic. Additionally, they perform measurements on two groups consisting of five WoW clients connected to ADSL lines. Their analysis at the packet level shows that the packet size in the downlink direction (i.e., server traffic) follows Weibull distribution while the uplink packet sizes have large discrete steps. Packet Inter-Arrival Time (IAT) was modeled by a joint distribution of three variables.

Kim et al. analyze the traffic of the MMORPG *Lineage* by *NCsoft* [27]. They have captured over 281 GB of data over 8 days on the server side with the help of the game operator. Authors use the following traffic models: packet size is modeled by the Power Lognormal Distribution and packet IAT was modeled by Extreme Value Distribution, though authors state that the model is not a good match. They continue work in the area by analyzing the sequel – *Lineage II* and show that the major characteristic of the MMORPG traffic is asymmetry of upstream and downstream traffic [26]. Again, capture is performed on the server side, resulting in around 7.7 billion data packets. Comparison between Lineage I and II is performed and authors note a significant increase in the size of server packets (around 15 times larger). Authors state that there is a linear correlation between number of active users and client traffic (correlation coefficient 0.99), but that the server traffic has lower correlation coefficient (0.95). Average IAT of packets within session is around 200 ms due to the TCP delayed ACK system which should be changed in order to reduce the Round Trip Time (RTT). Significant heavy tailed characteristics are found in session durations (e.g., longest session duration measured was 80 hours). Session IATs show an average value of 401 milliseconds, while at peak times on average 2.5 new users arrive every second.

Wu et al. provide the traffic model of an MMORPG *World of Legend* from *Shanda corporation*. Their trace is gathered while accessing the game through mobile GPRS access network [56]. For packet IATs they use an Extreme Value distribution for the client side traffic and a sum of two Extreme Value distributions for the server side traffic. Packet size is modeled as a sum of discrete steps (on 66 bytes and 72 bytes) and an Extreme Value distribution for the server side and as a deterministic distribution for the client side (73 bytes).

A number of works in the area of traffic analysis and modeling acknowledge the influence of different situations in the virtual world on the traffic patterns [12, 26, 31]. The following works explore this relationship further.

Park et al. collect and analyze network traffic traces of FPS *Quake 3* (Q3) and MMORPG WoW [35]. They define user actions based on the number of players and player behavior. Actions defined for Q3 are: Shooting, Moving, Normal, No Play, while for WoW actions are defined as: Hunting the NPCs (Non-Player Characters), Battle with players, Moving, and No play. Authors perform modeling of every type of behavior; for WoW size of the packets is modeled by the Exponential distribution on the server side and Normal on the client side, while packet IATs are modeled with the normal distribution.

Analysis of the WoW traffic packet contents is done by Szabó et al. [52]. Authors claim that the player behavior has a high impact on traffic characteristics at both macroscopic level (e.g., traffic rate) and at microscopic (payload content) level. They measure and analyze the traffic of WoW and *Silk Road Online* by *Joymax*. Their definition of the virtual world states is based on two axes, the movement of the player (moving, stalling), and the number of surrounding players as a mean to determine the location (in or outside of the a densely populated area – "the city"). This approach results in four possible states: Moving in the city, Moving outside the city, Stalling in the city, and Stalling outside the city. Identification of the separate states was done through active measurements and wavelet analysis.

Antonello et al. [7] define traffic models for *Linden Lab* 3D virtual world *Second Life* (SL). SL is not an MMORPG, and (in general) has different traffic characteristics because of the different model of the virtual world. However, some common characteristics exist. Authors perform their modeling for combination of three categories of player movement – Standing, Walking, and Flying, and two categories describing density of avatars in a zone – Popular and Unpopular. Inter-departure times for different combinations of avatar density action are modeled with split distributions combining two or even three distributions among which are Beta, Gamma, Lognormal, Extreme, and Weibull distribution. Inter-departure times are modeled with a Beta distribution. Client packet size is modeled as either deterministic or Extreme distribution. Inbound packet size is modeled as a split distribution of four models combining: Uniform, Exponential, Deterministic, and Extreme distributions.

Most of the approaches for measuring player behavior rely on the polling mechanisms which gather data about the virtual world from one client using game mechanisms. Another approach is to obtain the proprietary data from the game provider on which detailed analysis can be performed.

Tarng et al. analyze WoW player's game hours, on a trace sample gathered for almost two years with the use of an add-on for WoW client which polled all active players on one server [53]. Parameters investigated are subscription time, daily and weekly patterns, and consecutive play days. Additionally, they try to predict player behavior for both short and long term.

Lee and Chen use the same strategy to gather data [32]. They study the hourly, daily, and weekly patterns of players number on one WoW server in Taiwan. They propose a server consolidation strategy for the reduction of hardware and energy usage requirements based on the number of players and spatial locality. In their following work they present the full dataset of measurements taken in a 1107-day period between Jan. 2006 and Jan. 2009 [33]. The dataset includes the avatars gameplay times and a number of attributes, such as avatars's in-game race, profession, current level, and in-game locations.

Zhuang et al. gather data with the help of a similar WoW add-on [58]. Based on five months of measurements, they calculate player count, availability, downtime, inter-arrival times, and geographical distribution of players in the virtual world. They note that the stay time of the player in certain area of the virtual world depends on the nature of the selected area, which they classify into Transit, City, and Quest.

Pittman and GauthierDickey model the virtual populations and behavior of WoW players [39]. Authors define four facets which are needed for a complete model: population changes over time, arrival rates and session duration, spatial distribution of players over the virtual world, and movements of players over time. They state that artificial workloads using uniform distribution of players differ significantly from the real load observed in their measurements. In their following work [38] they develop behavior models for WoW and *Warhammer Online* (WaR) by *Mythic Entertainment*. They prove that two MMORPGs with different play styles can be modeled by using a unified set of functions, through modeling session lengths (Weibull distribution), player distribution through zones (Weibull distribution), number of zones visited related to session lengths (linear dependence), time spent in zones (Weibull distribution), and player movement (Lognormal distribution).

While WoW is frequently examined due to its high popularity, other MMORPGs are also objects of studies. Feng et al. study the trace of MMORPG *EVE Online* which was provided to them by the game publisher *CCP* [21]. They focus on examining daily and weekly patterns of player load, growth of player population over time, and impact of adding new game content on play time. Also, a prediction of the overall workload and prediction of players disinterest in the game is presented.

Kawale et al. study the problem of player churn in *EverQuest 2* by *Sony Online Entertainment* [24]. They propose a modified diffusion model for the social influence on the player, which takes into account player engagement, based on activity patters. Their model increases churn prediction accuracy for their dataset when compared with conventional diffusion models for the player engagement.

To the best of our knowledge, the only behavior based traffic generation for games is created by Shin et al. [42]. While authors state that their traffic generation processes is based on player behavior, by that they refer to the highly erratic characteristics of gaming traffic, and not to what the user does in the virtual world in-game. They propose a novel method for modeling the network traffic of games based on a transformational scheme in order to simplify the shape of the traffic so it can be mapped to an analytical model. They prove that their approach yields better results in comparison to even mixture models consisting of several distributions for WoW and for the FPS game *Left 4 Dead* (L4D) by *Turtle Rock Studios*. Authors implement their method in an online game traffic generator [41] and show that the generated traffic is following the model in case of L4D traffic but not for WoW. They ascribe the observed discrepancies to the Nagle's algorithm and Delayed Acknowledgments of TCP. Disabling these options in the generator resulted in adequate goodness of fit.

Our approach to player behavior differs in that we determine what exactly players are doing in the virtual world and for how long, as opposed to examining the characteristics of overall session time. Our measurements have been taken on the client side of each participating player through our WoW add-on deployed on players' computers [44]. This approach has certain advantages compared to remote monitoring. First, our precision is the highest as compared to the works of others (1 second, as opposed to 5 minutes or more). Second, accuracy is improved since we can exactly determine the actions taken by a specific player regardless of the in-game virtual character. Regarding traffic modeling, we base it on action categories which are defined as sufficiently general, distinct, and measurable in terms of player behavior in MMORPGs. Such definition enables better estimation of traffic loads based on player behavior. Finally, through our architecture we create a traffic generator truly based on application level user behavior.

3 Action categories in WoW

During a game session, an MMORPG player may perform number of various actions, from, for example, picking flowers to slaying monsters. In our previous work [46, 48], we categorized user actions into six action categories: *Questing, Trading, Player versus Player (PvP) combat, Dungeons, Raiding.* WoW was used as a case study for defining the categories, but they are considered to be applicable to other MMORPGs in which the game involves player's character progression. Categories are determined based on seven factors common to most MMORPGs: 1) the number of actively participating players, 2) the number of Non-Player Characters (NPCs), 3) mobility of the character, 4) required level of cooperation between players, 5) combat between characters and/or NPCs, 6) the dynamics of player input, and 7) communication aspect. We briefly describe the properties of each category.
Trading: This category includes non-combat actions of creation and exchange of virtual goods. Virtual items can be exchanged directly between two players or through the auction system. The activities in this category are usually performed by one or two players.
Questing: Activities consisting of solving tasks which are assigned to players by NPCs. Players are performing the given task in order to get rewards such as experience or virtual items. Players usually participate in *Questing* alone or they collaborate with a small group of other players in case that the required task cannot be accomplished by a single player alone.
Dungeons: Primary activity for small groups which consists of combat between a small, limited group of players and hostile NPCs in specific instances. Instances are isolated portions of the virtual world which are replicated for each group of players so interruptions from players outside the group are not possible.
Raiding: Activities which involve battles between larger group of players and more difficult and complex NPCs. This category is similar to *Dungeons*, but it is more complex considering the group size, required level of communication, and higher rate of actions performed. Due to its overall higher complexity, *Raiding* results in better and much more valuable rewards than *Dungeons* and *Questing. Raiding* hence brings the best prizes to the players, both in the terms of virtual items, and publicly acknowledged achievements. This kind of action re-

quires a highly organized group of players so additional communication is often required for coordination purposes.
PvP Combat: Combat between players in an instanced or not instanced battlefields. It is one of the most complex action types in an MMORPG. The number of players participating in combat may vary significantly – from only a few players to tens of them. There are almost no NPCs; the action rate is very dynamic with the high player input rate and high mobility. Best players earn valuable prizes which makes activities in this category highly competitive.
Uncategorized: This special category designates the portions of the session which can not be assigned to neither of the above user behavior categories. During the time spent in this category players go Away From Keyboard (AFK), they chat with other players while not participating in any other action, or wait for some events to take place (e.g., waiting for the guild officers to pick the people for the raid). In order to validate our assumptions regarding actions that the players perform while in parts of the session marked as *Uncategorized*, we performed additional measurements.
Defined categories describe user behavior, but also the situation in the virtual world that surrounds the user. For example, if a player is raiding, all other players in the same instance are also raiding. In this way players actions also define the surroundings of the player as well. More detailed information regarding specification and validation of action categories can be found in [46].

4 Methodology

In this section we describe the performed measurements and methodologies applied in the modeling process. For fitting of empirical data to analytical distributions we used the tool Minitab 16 which can estimate the underlying data distribution from 15 available distributions with both least squares and maximum likelihood estimation (MLE) methods. We calculated the parameters using both of them in order to validate the results. Results presented here are obtained with maximum likelihood estimation.

4.1 Data gathering

The required data was obtained through several sets of measurements. We gathered the action specific network traffic traces in order to create the network traffic models for each of the defined action categories. Regarding player behavior, we performed three sets of measurements:

– *Set 1* - the initial set of measurements [46],
– *Set 2* - the largest set of measurements on which the modeling presented in this paper is based [45],
– *Set 3* - the measurements aimed to further explore the parts of the session previously assigned to *Uncategorized*.

Network traces for creating and validating of the action categories network traffic models were gathered by the six WoW players who volunteered to participate in our research. Traffic traces were captured on the client side, by using Wireshark tool for network protocol analysis (http://www.wireshark.org) as described next. Players would start the process of capturing the data when they were about to start participating in a specific activity. Some of the action categories (such as *Dungeons*, *Raiding* and *PvP combat*) are performed in so-called instances (specific areas of the virtual world). For those categories the players participating in the measurement process were instructed to start capturing the traffic as they enter the instance, and stop the capture as they leave it. As for the *Trading* category, players would capture the traffic of the session in which they participated in any activity that included selling or buying virtual items from another player or through an auction system, crafting virtual items or retrieving them from an in-game bank or mail. When the players received a quest in the game, they started capturing the traffic for the *Questing* category. Accordingly, they stopped it after they had completed all the tasks defined by the quest. We now describe the 3 data sets.

Set 1. The initial measurements considering the user behavior were taken over the six weeks from November to December 2008, following the release of the second expansion for WoW "Wrath of the Lich King" (WoTLK), on 13th November 2008. This measurement was done with the help of 11 volunteering WoW players. For the purpose of player in-game behavior we developed a WoW add-on named World of Warcraft Session Activity Logger (WSA-Logger) [44] using ACE3 framework [4] and Blizzard Entertainment's WoW Application Programming Interface [5].

Set 2. The second set of measurements was collected through a student project. To participate, students needed to find at least five WoW player volunteers who agreed to install and use the WSA-Logger and participate in this research. To find player volunteers students referred to their colleagues who play WoW, or their acquaintances from the game since some of the students were playing WoW themselves. Online communities such as WoW Internet forums were another source of volunteering players. No personal data about the participating players, other than their age, was gathered. The described method of obtaining the player sample resulted in somewhat younger player sample (average age 24) compared to the general population of MMORPG with the average age 33 (Williams et al. [55]). The monitoring period for the second set of measurements was from May 5, 2009, to June 21, 2009, and the measurements have been performed in the 3.x version of WoW with the total number of 104 participating players. While the first set of measurements has been performed right after the launch of new game content (i.e., WoTLK expansion) resulting in a non typical user behavior, the second set was performed in the time frame in which there was no recent release of additional content in order to observe typical player behavior. Three factors that affected the size of the player base were that players needed 1) to volunteer in order to participate, 2) to install the WSA-Logger and, 3) after collecting, submit the gathered data. However, the relatively small number of players involved was compensated with the high quality of information that we managed to gather during the process, since the traces enabled a much more detailed examination of each player's behavior. Information about the use of voice communication during the game session was also obtained through an additional questionnaire filled in by the volunteering players. The source of the add-on was made open so as to address players' concerns regarding privacy.

Set 3. The final set of measurements was performed under the same terms as the second set (i.e., as a student assignment with the same rules). The goal of the measurements was to further investigate *Uncategorized* by tracking players location and movement patterns. Monitoring period for this measurements was from May 15, 2011, to June 10, 2011, in 4.x version of WoW (The Cataclysm expansion) with 15 players participating in the measurements. WSA-Logger was renewed in order to add a GUI display of gathered data, and reduce the load on the system. Additionally WSA-Logger was renamed wJournal due to the fact that the previous name reminded the players of malicious software used to steal information regarding player accounts, i.e., the "key loggers". The latest version of wJournal can be obtained from [44].

The principle on which our add-on monitors player behavior is based on WoW API. After a certain action is performed in the virtual world, an event is fired by the WoW API (e.g., after player enters another zone in the virtual world, "ZONE_CHANGED_NEW_AREA" event is fired). Those events are recorded during the monitored period and are listed in the order of appearance in the log file produced as the output of the add-on. Each event in the log file has a time stamp that marks the date, time (hours, minutes, and seconds) and the day of the week. The name of the fired event and the corresponding type of player action are marked alongside the timestamps. We captured only those events for which it was possible to determine related user action category. Such structure of the log file enabled the extraction of the data related to the player behavior in the terms of specific categories. The precision of the data gathered is one second, which is currently the highest achieved event monitoring precision considering the related work in this area (as of November 2011). More detailed description of our add-on and the method of relating WoW API events to the player action types can be found in [45].

4.2 Data filtering

In the process of network traffic measurements after recording the traffic trace, players added a detailed description of their activities in the captured session. This annotation was needed so the captured data can be used to distinguish different action categories. Namely, during the capturing process players often forgot to start (or stop) the capture at the adequate moment due to high immersion in the game. This resulted in more than one action category being captured in one trace. A significant part of the measured data was discarded through the filtering process leaving the 83 context specific traffic traces and 1395940 packets to be taken in consideration.

On the packet level we also perform a filtering process in order to remove a large TCP signaling overhead in our traffic trace, as the TCP ACK packets carrying no payload are quite common. We filter out TCP ACK packets with empty payload, but we note the percentage of those for each category. The initial packet trace consisted of a high number of server side packets having the size corresponding to the Maximum Transmission Unit (MTU). We presume that this is the result of application trying to send datagrams larger than MTUs. In order to determine the Application Protocol Data Unit (APDU) size we have implemented an algorithm proposed in [51]. Their algorithm is based on the notion that some packets in their trace had the TCP PSH flag set, which is the case when application has data that it needs to have sent across the network immediately. According to the assumption, if APDU is larger than the MTU, the TCP service splits the APDU and sets the PSH flag for the last packet in the sequence. We follow this approach by processing our dataset in order to calculate the correct APDU size values and inter-arrival times between subsequent APDU, and modeling those values (not the packet size, or packet IAT).

For the application behavior data, data filtering was performed after the processing. After the behavior logs were parsed, the action specific data, comprising of start time, end time, player designation, type, and session number were stored into a MySQL database. For the purpose of visualization of the gathered data and the display of the data characteristics, an Ajax based web tool was developed [43]. This tool was used for visual data filtering and easier identification of anomalous behavior and outliers in the data.

4.3 Data analysis and processing

We perform traffic modeling through definition of analytic traffic models (i.e., mathematical description) which are easier to convey and to analyze compared to empirical models of traffic (e.g. tcplib [19]). We are following the approach for application traffic modeling introduced by Paxson [36], firstly used in the area of network games by Borella [9]. The algorithm is also described in detail in [28], and it is presented as follows:

1. The probability distribution of the data set is examined and an appropriate analytical distribution is chosen. This is usually done through the visual examination of the Probability Density Function (PDF) or Cumulative Distribution Function (CDF) of the data. As an example of the analytical distribution we present the PDF of Weibull distribution as it will be commonly used further in the paper:

$$f(x) = \frac{\gamma}{\alpha}(\frac{x-\mu}{\alpha})^{\gamma-1}e^{-(\frac{x-\mu}{\alpha})^{\gamma}} \tag{1}$$

   where γ is the shape parameter, α is the scale parameter, and μ is the location parameter.
   Quantile-Quantile plot (Q-Q plot) is a method for graphical comparison of two distributions by plotting their quantiles against each other. It is commonly used for visual evaluation of goodness of fit. First, the set of intervals for the quantiles are chosen. A point (x, y) on the plot corresponds to one of the quantiles of the second distribution (y-coordinate) plotted against the same quantile of the first distribution (x-coordinate). In this way, by plotting an empirical distribution, $F(x)$, against the chosen distribution $G(x)$ we can observe the goodness of fit. If the resulting points are in a straight line, it means that the $F(x) = G(x)$, but in practice there are usually deviations in the fit. By using Q-Q plot, it is easy to observe where deviations occur (e.g., lower tail, the main body, upper tail).
   In order to manage large data sets values are aggregated into "bins". Depending on the bin choice, the final results may become skewed. The algorithm for choosing the optimal bin size w is taken from [40]:

$$w = 3.49\sigma n^{(-1/3)} \tag{2}$$

   where σ is the estimate of the standard deviation and n is the number of observations.
2. Fitting the data set onto an analytical distribution using MLE to determine the parameters of the distribution.
3. If the fit is especially deviating from the part of the distribution (e.g., upper tail), it is possible to model the data with split distribution. Q-Q plot can be used to observe the deviations.
4. Calculating the λ^2 discrepancy measure. As the standard goodness of fit tests are biased for large and messy datasets, discrepancy measure is used [37]. In short we explain the discrepancy measure. If we have observed n instances of a random variable Y which we want to model using another distribution Z, N is the number of bins in which we partition the distribution Z. Each bin has a probability p_i associated with it, which is the proportion of the distribution Z falling into the i-th bin. Let Y_i be the number of observations of Y that actually fell into the i-th bin. Chi-Square goodness of test X^2 is defined as:

$$X^2 = \sum_{i=0}^{N} \frac{(Y_i - np_i)^2}{np_i} \tag{3}$$

   K parameter is defined as:

$$K = \sum_{i=0}^{N} \frac{(Y_i - np_i)}{np_i} \tag{4}$$

   Estimator for discrepancy for the grouped data is defined as:

$$\hat{\lambda}^2 = \frac{X^2 - K - df}{n-1} \tag{5}$$

   Where n is the number of observations in the data set and *df* is the number of degrees of freedom of the test. The value of *df* is calculated as the number of bins N minus the number of parameters that were used to estimate the analytical distribution. In the case of deterministic distributions, in which all observations are expected to have the same value, this equation causes a divide-by-zero ambiguity if the empirical data set contains values that vary from the expected value where the expected value is zero (i.e., np_i is zero). To avoid this problem, the following alternative versions of the equations are used:

$$\hat{X}^2 = \sum_{i=0}^{N} \frac{(np_i - Y_i)^2}{Y_i} \tag{6}$$

$$\hat{K} = \sum_{i=0}^{N} \frac{(np_i - Y_i)}{Y_i} \tag{7}$$

5. Examination of the tail, in search for deviations using the following expression:

   $$\xi = log_2 \frac{a}{b} \tag{8}$$

   Where a is the number of instances predicted to lie in a given tail, and b is the number of instances that actually lay in this tail. If the b equals zero it is replaced by 0.5. If the values of ξ are positive it suggests that the model overestimates the tail, and negative values indicate that the tail is underestimated.
6. Calculation of the autocorrelation function of the trace. Usually, short-term autocorrelation, or the autocorrelation at lag 1 is examined.

As a result of the measurement performed with all versions of our add-on, a log file is created. For the purpose of analyzing log files we developed a log file parser in Java. The parser was needed so that the add-on running on players computers could be kept as simple and lightweight as possible. We monitored several key session characteristics such as overall session duration and actions performed by a player within a session. We define session duration as the time between the point of player's logging in to the game and the point of the subsequent logout. "Session segment" is a part of the playing session in which the player is involved in actions from only one specific category. Session segments of instanced categories are labeled with their start and end time (there are specific events labeling entrance and exit from the instanced areas), and segments of non instanced categories are noted as a chain of events. The timeout period (period of time in which no events were fired) longer than 5 minutes is labeled as *Uncategorized.* In the measurement set 3 we also tracked players' movements in the virtual world in order to find out what exactly do players do during *Uncategorized* session segments.

5 Mathematical models of traffic, user and session characteristics

In this section we present mathematical models for network traffic, number of active users, session length, user behavior in terms of probabilities of switching between various action category segments, and session segment length. All models are based on the empirical datasets obtained in our measurements. The main addition in area of player behavior modeling in regard to our previous work is inclusion of the *Uncategorized* into the model which has been enabled by the new set of measurements that provided significant insight regarding what players actually do during these time periods. We investigated this through collecting players location in the virtual world. Over 73% of *Uncategorized* time, players proved to be located in the main cities (e.g., Stormwind, Ogrimmar, Dalaran). Also, some of the players participated in the study stated that sometimes they use the game just as a big chat application. This confirms our inital assumption that *Uncategorized* comprises periods in which players wait for some organized activity to start, or just to chat with other players. Main cities are also locations in which players are protected (they can not be attacked by hostile players) and are perfect spots for leaving the character unattended for a brief period (i.e., going AFK). Additionally, we present the probabilities of using VoIP communication in addition to, and per each action category.

5.1 Network traffic

First we present the traffic models for each of the categories. As stated in the methodology section, we model not packet, but the size of the APDU and APDU IATs, as we are trying to model the sources of the traffic. Due to the highly erratic traffic characteristics we mostly employ mixture models in which combination of distributions is used. For example, the lower part of the data (up to a certain value) is modeled with Weibull distribution, while upper part of the data (after the certain value) is modeled with Lognormal distribution. The best fits for our dataset proved to be Weibull, Lognormal, and Largest Extreme Value distribution. Also, we used deterministic values as client traffic showed significant discrete values. Parameters of the models for each action category and each parameter (i.e., client APDU size, client APDU IAT, server APDU size, and server APDU IAT) are presented in Tables 1-5. We do not model the traffic of *Uncategorized* as such, but use the traffic model of *Trading* instead, based on the results of the behavioral measurements. We chose this approach because *Trading* is an action category with the lowest player mobility and dynamics and *Uncategorized* comprises time periods in which users are commonly AFK. Also, regarding the server side traffic *Trading*, fits the best due to the virtual world collocation of the players in these two action types (i.e., both *Trading* and *Uncategorized* are usually performed in big cities).

Client side APDU IATs can be split in two sectors, below around 500 ms and above that value. For both areas, Weibull distribution showed as providing a good fit. Also, we found that significant "spikes" exist at values 0 and 500 ms. We assume that this phenomenon is due

Table 1: Network traffic model parameters for *Trading*

Data	Count	Model	Parameters	$\hat{\lambda}^2$	Tail	ACF(1)
Client APDU size	15612	Deter. p=5.25%	$a = 6$	0.0911	32(0.20)/0	0.24
		Deter. p=4.21%	$a = 10$			
		Deter. p=34.05%	$a = 14$			
		Deter. p=5.72%	$a = 15$			
		Deter. p=3.19%	$a = 18$			
		Deter. p=32.50%	$a = 35$			
		Deter. p=9.14%	$a = 39$			
		Deter. p=5.94%	$a = 51$			
Client IAT	15602	Weibull p=50.53%	$\gamma = 0.99, \alpha = 176.74$	0.0817	10(0.06%)/-	0.29
		Weibull p=28.53%	$\gamma = 0.66, \alpha = 1220.33, \mu = 500.95$			
		Deter. p=17.60%	$a = 0$			
		Deter. p=3.34%	$a = 500$			
Server APDU size	27082	Lognormal	$\mu = 4.16, \alpha = 1.15$	0.0888	145(0.54%)/-	0.05
Server IAT	27081	Lognormal p=82.68%	$\mu = 5.62, \alpha = 0.95$	0.1063	23(0.08%)/-	0.17
		Deter. p=9.62%	$a = 200$			
		Deter. p=7.7%	$a = 218$			

Table 2: Network traffic model parameters for *Questing*

Data	Count	Model	Parameters	$\hat{\lambda}^2$	Tail	ACF(1)
Client APDU size	63541	Deter. p=4.96%	a=6	0.0415	11(0.02%)/0	0.46
		Deter. p=7.34%	a=10			
		Deter. p=20.75%	a=14			
		Deter. p=2.82%	a=18			
		Deter. p=2.36%	a=21			
		Deter. p=50.18%	a=35			
		Deter. p=9.20%	a=39			
		Deter. p=2.39%	a=51			
Client IAT	63531	Weibull p=55.7%	$\gamma = 1.19, \alpha = 236.22$	0.1608	101(0.17%)/-	0
		Weibull p=12.6%	$\gamma = 0.84, \alpha = 1073.63, \mu = 525.95$			
		Deter. p=16.46%	$a = 0$			
		Deter. p=15.24%	$a = 500$			
Server APDU size	99163	Lognormal	$\alpha = 1.22, \mu = 4.55$	0.2304	163(0.16%)/-	0.05
Server IAT	99177	Normal p=71.51%	$\mu = 212.87, \sigma = 96.59$	0.1364	47(0.48%)/-	0.21
		Weibull p=7.49%	$\gamma = 0.91, \alpha = 451.55, \mu = 419.96$			
		Deter. p=2.15%	$a = 44$			
		Deter. p=12.27%	$a = 218$			
		Deter. p=6.58%	$a = 328$			

Table 3: Network traffic model parameters for *PvP combat*

Data	Count	Model	Parameters	$\hat{\lambda}^2$	Tail	ACF(1)
Client APDU size	66635	Deter. p=7.63%	a= 6	0.1307	0/0	0.39
		Deter. p=5.60%	a=10			
		Deter. p=13.12%	a=14			
		Deter. p=3.11%	a=19			
		Deter. p=59.50%	a=35			
		Deter. p=6.66%	a=51			
		Deter. 4.38%	a=58			
Client IAT	66631	Weibull p=78.4%	$\gamma = 0.79, \alpha = 208.50$	0.0681	155(0.23%)/-	0.24
		Deter. p=20.18%	$a = 0$			
		Deter. p=:1.42%	$a = 500$			
Server APDU size	71594	Weibull p=93.08%	$\gamma = 0.92, \alpha = 538.59$	0.0183	14(0.02%)/-	0
		Largest Extreme Value p=0.73%	$\mu = 7754.99, \alpha = 394.83$			
		Deter. p=6.19%	$a = 37$			
Server IAT	71594	Weibull p=83.32%	$\gamma = 1.71, \alpha = 193.26$	0.1799	97(0.14%)/-	0.22
		Deter. p=4.13%	$a = 44$			
		Deter. p=4.11%	$a = 200$			
		Deter. p=8.44%	$a = 328$			

Table 4: Network traffic model parameters for *Dungeons*

Data	Count	Model	Parameters	$\hat{\lambda}^2$	Tail	ACF(1)
Client APDU size	50460	Deter. p=4.57% Deter. p=8.00% Deter. p=16.28% Deter. p=4.04% Deter. p=8.28% Deter. p=55.70% Deter. p=3.13%	a=6 a=10 a=14 a=19 a=22 a=35 a=51	0.1048	55(0.04%)/0	0.27
Client IAT	50460	Weibull p=95.86% Deter. p=4.14%	$\gamma = 0.58, \alpha = 268.37$ $a = 500$	0.2038	221(0.44%)/-	0.33
Server APDU size	96035	Weibull p=99.15% Largest Extreme Value p=0.85%	$\gamma = 0.89, \alpha = 221.83$ $\mu = 7698.83, \alpha = 198.842$	0.0331	63(0.07%)/-	0.01
Server IAT	96056	Weibull p=78.35% Weibull p=2.58% Deter. p=3.06% Deter. p=9.55% Deter. p=6.46%	$\gamma = 2.28, \alpha = 231.3$ $\gamma = 0.79, \alpha = 344.14, \mu = 405.96$ $a = 44$ $a = 200$ $a = 328$	0.1443	32(0.03%)/-	0.16

Table 5: Network traffic model parameters for *Raiding*

Data	Count	Model	Parameters	$\hat{\lambda}^2$	Tail	ACF(1)
Client APDU size	19136	Deter. p=3.81% Deter. p=4.35% Deter. p=12.15% Deter. p=20.18% Deter. p=3.63% Deter. p=6.81% Deter. p=45.53% Deter. p=3.54%	a=6 a=10 a=14 a=19 a=20 a=29 a=35 a=51	0.1022	19(0.1%)/0	0.27
Client IAT	19135	Weibull p=85.73% Deter. p=14.27%	$\gamma = 0.76, \alpha = 299.52$ $a = 0$	0.0898	65(0.34%)/-	0.27
Server APDU size	37801	Weibull p=98.97% Weibull p=1.03%	$\gamma = 0.86, \alpha = 941.79$ $\gamma = 0.91, \alpha = 1183.28, \mu = 7298.20$	0.0342	16(0.04%)/+	0.16
Server IAT	37801	Weibull p=84.39% Deter. p=9.55% Deter. p=6.06%	$\gamma = 1.99, \alpha = 188.92$ $a = 44$ $a = 200$	0.0660	6(0.02%)/-	0.03

to dynamics of player activity coupled with the game timeout mechanisms. Subsequent packets (0 ms IAT) are sent while player is performing a highly dynamic action (e.g., highest percentage of 0 ms IATs is in *PvP combat*), while 500 ms IAT is probably due to some sort of keep alive mechanism (i.e., even when players are not performing any actions in deserted areas of the virtual world the server needs to maintain the connection).

We observe several discrete steps for server side IATs at 44 ms, 200 ms, 218 ms and 328 ms. While the step at 200 ms can be explained with the TCP delayed ACK mechanism, the rest of the steps are probably inherent to the WoW application protocol. Server IATs have, in general, been the most difficult to model, and in the end have worst fits compared to other parameters.

Client APDU sizes show several discrete steps. For client side, APDU sizes and packet sizes coincide as there is really low amount of packets which are larger than 1500 B which is the most common MTU size. The most frequent values of APDU sizes vary across different actions, but packets of size 35 B are the most frequent which is in contrast with the [51] who modeled WoW client traffic with packets of size 6 B, 19 B and 43 B. We assume that these packets carry information about character's movement as suggested in [52].

Server side APDU size has different behavior depending on the category. We noticed that action categories which include instances (i.e., *Dungeons, Raiding, PvP combat*) have significant spikes in the distribution around 7000 B, which is really high for MMORPG traffic. We decided to separately model these spikes as they carried a high amount of overall information (up to 20% of all data), although they appeared with relatively low frequency. Those parts were modeled with Weibull or Largest Extreeme Value distribution. We assume that this phenomenon is related to loading significant amount of data when entering instances. Lower section of the data was modeled with Weibull distribution with some discrete steps (usually at 37 B). For the non instanced categories (i.e., *Questing, Trading*) Lognormal distribution with some discrete steps proved to be a good fit for the whole dataset.

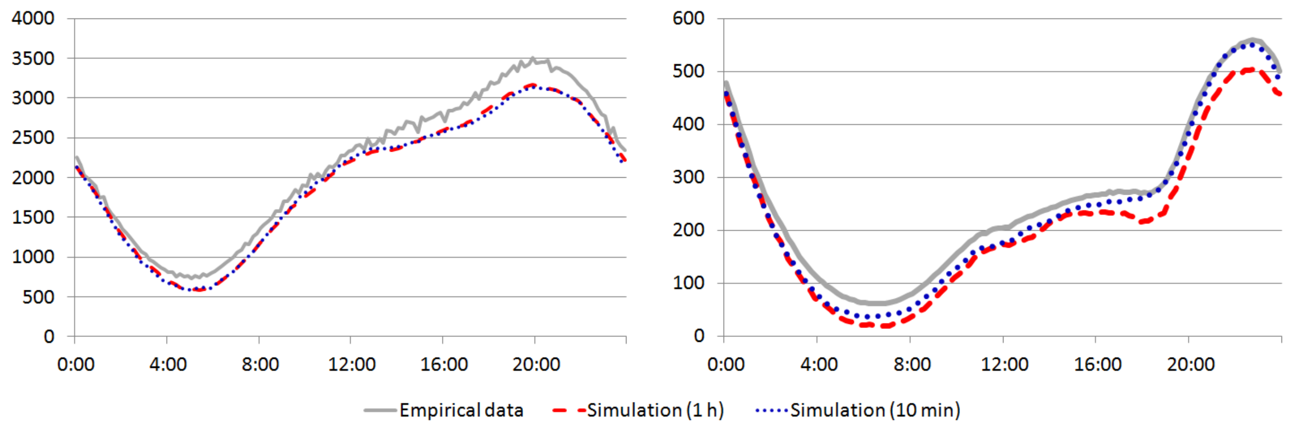


Fig. 1: Empirical and simulated (HPP) number of users over a 24-hour cycle (left – Pittman's dataset, right – Lee's dataset)

5.2 Number of players

In our modeling process we aim to model only the number of players on a single shard or a single WoW server. Number of active players on several different WoW servers has been studied in several works [32,33, 38,39,58] over a very long period of time (up to over 1000 days). Several datasets from listed measurements are publicly available, so we do not repeat the measurement process, but use the datasets described in [38] and [33]. The following differences between these datasets should be noted: 1) they were taken on two different servers of WoW, 2) measurements have been done in different time frames, and 3) while Pittman's datasets comprises data about both available player factions on the server, Lee's datasets contains information about only one. Inspection of the data from those works makes it evident that hourly and daily patterns exist in the number of active players. This conforms to results of Williams et al. [55], showing that the average age of the MMORPG player is 33, as opposed to the common misconception that mostly teenagers play MMORPGs. Thus, most of the players follow a hourly routine of work or school over day which results in their availability in the late afternoon/evening, and daily routine which enables them to be more available during the weekend.

We model the number of players through the initial value combined with arrival and departure processes for increasing and decreasing the current user number. We model the arrival of new players and departure of leaving players as an *Homogeneous Poisson Process* (HPP). This modeling is based on results of Chen et al. [12] who stated that arrival process can be modeled as HPP for each 1-hour interval. In other words, the same rate for the arrival/departure process is applied during 1 hour intervals leading to 24 rates for each process for one day and 168 rates for the whole week. While their results are obtained based on an analysis of a different game – ShenZhou Online, the authors stated that the results are applicable for other MMORPGs. Additionally, beside 1-hour intervals, we investigate arrival and departure processes in 10 minute and 15 minute intervals for Lee's and Pittman's dataset respectively, as their sampling frequencies enable this. We wanted to observe whether lowering the duration of the time interval, on which constant HPP rate is defined, would make the simulation of the number of players significantly better. Figure 1 shows Lee's and Pittman's results for number of users over a 24-hour cycle, as compared to our own simulation results. As it can be observed from Figure 1, both our simulations slightly underestimate the number of users. Also, the simulation results with 1 hour intervals are not significantly worse than those with 10 minute and 15 minute intervals. All parameters (i.e., starting number of users, and a 24 values for arrival/departure rates) are calculated for each day of the week in order to capture both hourly and daily patterns.

This modeling procedure is static, meaning that it is not adaptable to the variable number of users because of its dependence upon arrival and departure rates. In order to enable simulations with a significantly different number of players, we apply a normalization technique in an effort to obtain the "shape" of the user number characteristic for a WoW server over a 24-hour daily time cycle. We apply normalization methodology for prediction of number of players described in detail in [54]. The algorithm is described as follows. Our training set consists of the number of users over a period of D days and nights (i.e., around 1100). Let v_td be the number of active users at time t on cycle d. For each 24 hour cycle d, number of users at T timepoints is observed, which results in following training set:

$$V_d = [v_{1d}, ..., v_{Td}], d = 1, ..., D. \quad (9)$$

We use a slight modification as we normalize the number of players with the average number expected in the day in order to be able to start the simulation at any time in the day.

$$V_d^N = \frac{V_d}{avg(V_d)} = \frac{v_{1d}}{avg(V_d)}, ..., \frac{v_{Td}}{avg(V_d)} = [v_{1d}^N, v_{2d}^N, ..., v_{Td}^N]($$

For each timepoint t, we compute the 80 percent quantile.

$$q_t^{80} = 80\%quantile(v_{t1}^n, ..., v_{tD}^n), t = 1, ..., T \quad (11)$$

In this way we can obtain the predicted number of users at each time t no matter the average user number in the day. Based on the difference between current number of users and predicted number of users, in the next step we calculate parameters of HPP, an arrival rate (if an increase in the number of players is predicted), or a departure rate (if a decrease in number of players is predicted). This approach allows us to make simulations of a different number of users but still capture the characteristic curve which depicts hourly pattern in user number. We find this important, because the number of users can differ significantly between games, and as we can see in Figure 1, even between the different servers of the same game (i.e., depicted datasets are from two different WoW servers). It should be noted though, that Lee's dataset comprises only information about one (of two) more or less equally numbered factions in WoW so it represents approximately half of the total number of players.

5.3 Session length

Similarly to the number of users on a server, the session lengths in MMORPGs have been studied thoroughly. The issue with research regarding session lengths is that significantly different results are reported even in the case of the same game (e.g., WoW). These differences stem from two main causes: 1) lack of a firm session definition, and 2) different measuring techniques.

As a first step in measuring session length we want to answer a question: "What is a session?". We identify two types of sessions in MMORPGs. Firstly, we identify *character session* as a duration of presence of specific character in the virtual world, and secondly we define *player session* as the time period in which one player subsequently uses multiple characters. In short, one player session comprises multiple character sessions. This distinction is not always clear, nor emphasized in related work.

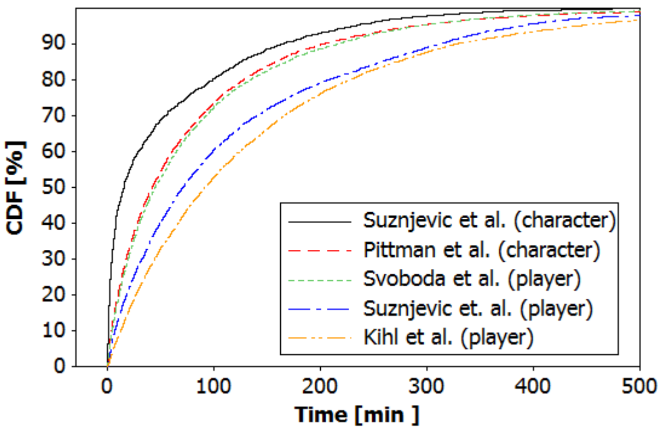


Fig. 2: Session length distributions in different measurements

The first, most frequent method of measuring session length is the polling technique. This technique is based on the ability of one character in the virtual world to obtain information about all other online characters. In WoW this is possible through the command "/who", which returns the list of active players. This enabled creation of automatic scripts which run in the client of one player and gather data about all other active players. This technique measures character sessions and is applied in a number of works [33,39,53]. Additionally, this type of measurement approach can introduce an error as sampling a player base each 10 minute interval results in the shortest session time of 10 minutes. The second approach is estimation of session duration from network traffic measurements [25,51]. Traffic measurements can extract the length of the whole TCP connection for the player resulting in player session lengths. We use the third approach in which the measurement process is taking place on the players' computers. This approach has an upper hand in measuring player behavior as it can be studied in great detail. The downside is that it is very difficult to find the players who are willing to participate in the research, as players need to perform some actions on their computers in order to participate which distracts them from playing the game. This results in a smaller number of players observed.

We illustrate how significant can the difference resulting from different measurement techniques and different definition of the session be in Figure 2. It should be taken into account that all of the measurements depicted refer to the same game (i.e., WoW). The shortest session duration is reported in our first work [46] as measurements had 1 minute precision and were char-

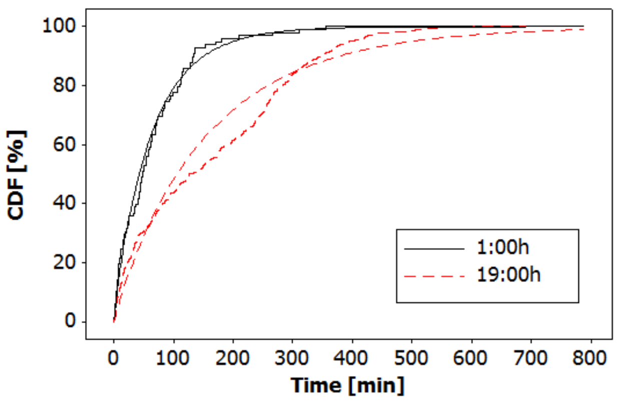


Fig. 3: CDFs with Weibull fits of the session length started at specific hour (19:00-20:00 and 1:00-2:00)

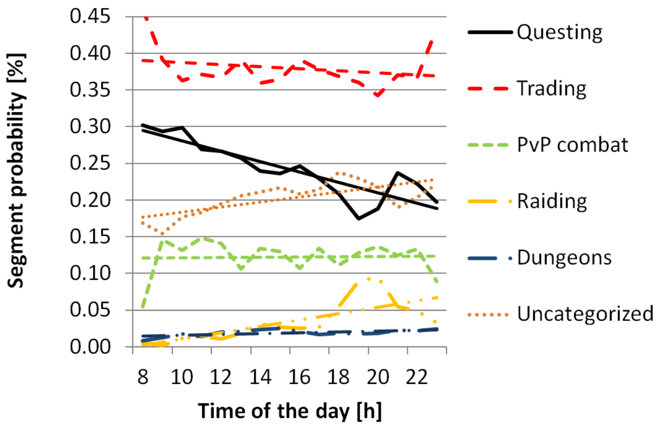


Fig. 4: Probability of specific session segments through the hours of the day

acter based (30% of the sessions reported were shorter than 5 minutes). It should be noted that the player sample on which measurement was performed was very small. Pitman measures character based sessions [38] with a 15 minute sample time, while Svoboda [51] models player sessions based on a traffic trace of a mobile network which have shorter session times compared to wired networks. Longer session times are reported in our previous work [45], as we measure player session with 1 second measurement precision and additional parsing to ignore disconnecting. The longest sessions are reported in the work by Kihl et al. [25], in which authors report player session length based on the results of traffic measurements in the wired accsess network. It is clear that character based estimates of session duration are significantly shorter than the real session duration.

We define the (player) session length as the time passed between player login and his logout from the game. If the time between a given player's logout and a new login is less than 5 minutes we treat those two subsequent sessions as one session. In this way using *Set 2* we identified 5872 player sessions, which comprised 11775 character based sessions. Additionally, we investigate whether session length also follows an hourly pattern. We assume that the sessions in the morning and during the day are shorter than the sessions in the evening. This assumption is based on the measurements performed in our previous work [45], in which we noted that *Raiding*, an action category with the longest average duration shows a strong daily pattern with high rise in the evening, corresponding with the increased availability of the average player to play at that time.

We confirm the results of previous works [38,51] by identifying that session lengths conform to Weibull distribution. Additionally, we prove that session lengths are heavily dependent on the time of the day. We do not present models for every hour, but only for two selected time frames between 1:00-2:00 and 19:00-20:00. In Figure 3, the Cumulative Distribution Functions (CDFs) of the duration of the session started between 1:00-2:00 a.m. and 19:00-20:00 are shown, together with their fits. Sessions started in the evening are on average a lot longer than sessions started late in the night. Also, sessions started in the evening do not conform so well to the Weibull distribution as those started later in the night due to the strong influence of *Raiding*.

We present the models for session length through inverse transformation function of Weibull distribution (time unit is second).

– Sessions started between 1:00 and 2:00:

$$f(x) = 61.8 * (-ln(x)^{(1/0.9327)}) \tag{12}$$

– Sessions started between 19:00 and 20:00:

$$f(x) = 155.7 * (-ln(x)^{(1/0.9414)}) \tag{13}$$

where x is uniformly distributed random variable $0 \leq x \leq 1$.

5.4 Session segment probability

Player behavior in terms of specific action categories shows a strong daily pattern due to the availability of an average player [45,46]. Probabilities of a specific action category segment appearing during the day are presented in Figure 4. The obvious trend is that group based activities (*Raiding* and *Dungeons*) have a raising trend from morning towards the evening, while *Questing* as a main single player activity has a strong decreasing trend. *Trading* and *PvP Combat* have more or less the same percentage of segments during the day.

Table 6: Session segment duration models

Action category	Fit	Shape	Location	Scale	AD	P-value
Trading	Weibull	0.69	-	190.98	14.70	<0.01
Questing	Weibull	0.66	-	440.01	59.93	<0.01
PvP Com.	Weibull	0.61	-	603.57	28.23	<0.01
Dungeons	Largest Extreme Value	-	1321.42	1116.00	4.80	<0.01
Uncategorized	3 - parameter Weibull	0.79	300.96	567.06	11.87	<0.05

While probability of the *Trading* segments is easily explicable, as *Trading* interleaves with most of the other action categories and is always and easily accessible to players, the question remains why does *PvP Combat*, as a group based action category have a stationary trend. The answer lies in the "battlegrounds" system of WoW. Players enlist for the battle and the game system automatically creates a player group consisting of random players which might not even be from the same server. In this way battles are always accessible, and players do not have to organize themselves as they have to do for *Raiding*. This shows how sensitive player behavior patterns are to game mechanics. It is expected of this patterns to change over time as the game designers expand the game with new mechanics. For example, a similar system to battlegrounds was introduced recently for the *Dungeons* category (after our measurements were performed). We expect that due to this change, the *Dungeons* will show the trend similar to that of *PvP Combat*. Another change has been implemented for *PvP Combat* – rated battlegrounds in which organized (i.e., not randomly assembled) groups of players compete in order to achieve higher ranks on the player scoreboards.

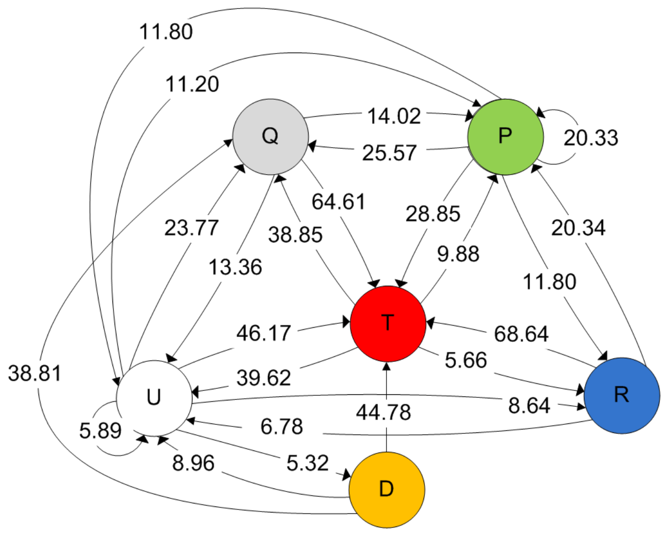


Fig. 5: Transition probabilities (percentage) between states in hour 20:00-21:00 (Q – *Questing*, P – *PvP combat*, R – *Raiding*, D – *Dungeons*, T – *Trading*, and U – *Uncategorized*)

We assume that this change will result in *PvP Combat* taking a trend similar to other group based categories. *Uncategorized* segments also have a raising trend towards the evening, which we assume is a result of more players being online and available for just chatting, as well as waiting for organized group activities.

We model the player behavior throughout the day through a first order Markov chains for each hour of the day as we assume that the next action is only dependent on the previous action (e.g., after a dungeon or a raid, it is typical for a player to go sell items he/she does not need, or upgrade the newly obtained ones). In Figure 5 transition probabilities between states for hour 20:00-21:00 are depicted. Each action category is modeled as a single state of the Markov chain and is marked with its starting letter. Transitions with probabilities lower than 5% are not shown in order to simplify the figure.

Based on the results of the questionnaire regarding the frequency of VoIP communication in parallel to gaming [45] we estimate probabilities of VoIP use per action category. Estimation is based on transferring the scores on the Likert scale as an interval-level data. Responses "Always" and "Never" were replaced with 1 and 0 respectively, while "Often", "Sometimes", "Rarely" are transformed to intervals $\langle 1-0.4]$, $\langle 0.4-0.2]$, and $\langle 0.2-0\rangle$ respectively. The probabilities were estimated as follows: *Trading* – 10.3%, *Questing* – 9.7%, *PvP combat* – 35.5% *Dungeons* – 33.2%, *Raiding* – 81.1%.

5.5 Session segment length

Markov chain models only the transition between states depending on the time of the day. In order to obtain a complete model we also need the duration in which the system stays in each state. In the next step, we investigate to which distributions the lengths of action specific segments conform, and whether the time of the day has an influence onto the length of the specific segment. First we determine whether the distributions of the session segment length vary through the day. We perform this through inspecting CDFs of segment lengths for each hour of the day. We conclude that *Questing*, *Trading* and *Uncategorized* segment du-

Table 7: Raiding segment duration models for different hours of the days

Hour	Data	Fit	Location	Shape	Scale	AD	P-value
18-19	42%	Weibull	-	1.175	2513	0.405	>0.250
	58%	Weibull	-	6.110	15306.5	1.722	<0.010
19-20	44%	Weibull	-	1.121	2583.76	0.663	0.085
	56%	Lognormal	9.4173	-	0.227	1.016	0.011
20-21	42%	Weibull	-	1.279	3303.08	1.18	<0.010
	58%	Lognormal	9.287	-	0.19762	0.768	0.045
21-22	100%	Weibull	-	1.368	6476.70	0.625	0.102
22-18	100%	Weibull	-	1.046	2786.5	5.209	<0.010

rations are not significantly dependent of the time of the day (i.e., differences between values of the CDF for different parts of the day were within 10%). *PvP Combat* has more differentiated CDFs, similar to *Dungeons*, but still the differences amongst hours of the day are rather small (i.e., different values of CDF within 20%). Based on this information we decided to model session segments of those four action categories as independent of the time of the day. The segment duration models are presented in Table 6, coupled with the values of goodness of fit tests (Anderson Darling statistic and the P-value). Most of the session segments conform best to the Weibull distribution, only *Dungeons* are modeled with Largest Extreme Value distribution. The P-Values are showing that even the best fits are not closely following the empirical distribution but it should be noted that these tests are biased for large messy datasets [23]. In order to graphically depict goodness of fit we plot CDFs of both empirical dataset and the models in Figure 6. As it can be observed, the fits are reasonably good.

On the other hand, *Raiding* shows significant differences in the CDF based on the time of the day. We note that the longest sessions occur in the evening, when players can afford long uninterrupted playing time spans (e.g., some *Raiding* sessions span few hours). We decided to model *Raiding* segment duration with five

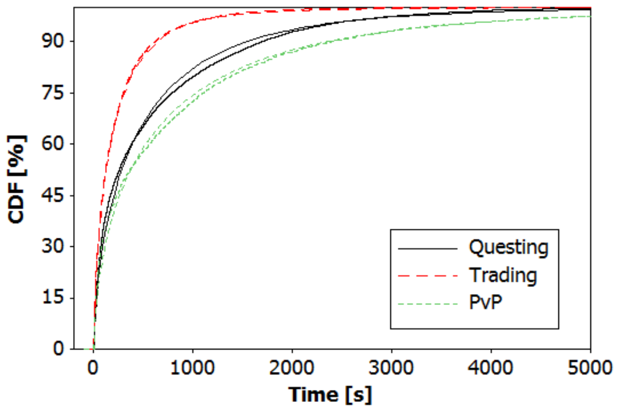


Fig. 6: Fits of Questing, Trading, and PvP Combat

different distributions based on the time, one for each of the hours 18:00-19:00, 19:00-20:00, 20:00-21:00, 21:00-22:00, and one distribution for the remainder of the day. Also, those segments proved to be harder to model, so mixture modeling is applied with a limit at 7500 seconds. Models for *Raiding* segment duration are depicted in Table 7 with portion of the data falling below and above 7500 seconds, and goodness of fit parameters.

6 UrBBaN-Gen architecture

The design goals of the traffic generation system architecture were the following:

– Good scalability of the system – existing hardware needs to be used to the maximum potential, new hardware can be added seamlessly if the simulation parameters require that;
– Independence of the service simulated;
– Low price.

The functional architecture is presented in Figure 7. The main components of the architecture are:

– *Control function and user interface* for controlling the parameters of the simulation;
– *Service repository*, containing input data which fully describes the simulated service;
– *Behavior process* in charge of simulating user behavior;
– *Traffic generation process*, which transforms the player behavior to network traffic.

The task of the *Control function and user interface* is to control other components of the functional architecture. Through the user interface, parameters of the simulation can be changed and parameters of the models can be adjusted.

The service repository contains behavioral definitions of services supported by the system. A service definition contains:

– An array of coefficients describing the "shape of the daily curve" for the purpose of simulation with different starting user numbers. These coefficients are

multiplied with the average estimated number of users in order to obtain the number of users at a certain point in the day. Values in the array are defined (at minimum) for every hour of the day. This array can be defined for each day of the week.
- Initial number of users and arrays with user arrival and departure rates on which the arrival and departure processes realized as HPPs are modeled. These rates are defined (at minimum) for every hour of the day for each day of the week. Rates can be defined in higher resolution as well (e.g., for every 10 minutes' interval).
- Markov chain parameters which define transition probabilities between specific player action categories depending on the hour of the day. These parameters could be defined for each day of the week as well. In our application for MMORPGs only hourly patterns are modeled.
- Probabilities of adding new media flows, depending on the session segment type (i.e., in our case, VoIP communication).
- Session and session segment length models are also defined for each hour of the day. As well as for the Markov chain parameters, there is an option to define parameters for each day of the week.
- User specific action coefficients contain user preferences towards certain action type (e.g., for our defined action categories, some users prefer *PvP combat* to *Raiding*).

Behavior process consists of *Session generator* and *User behavior function. Traffic generation process* consists of *Traffic resource manager*, *Traffic generation controller*, *Traffic sender* and *Traffic receiver. Session generator* handles the number of active users and the length of their presence in the system through arrival and departure processes. It takes input from the service definition, either in the form of arrival and departure rates and starting number of users, or, in the form of array of coefficients depending on the type of simulation chosen. It assigns *User behavior function* for each new (simulated) user, and communicates with the *Traffic resource manager* giving it input based on the user behavior. The output sent to *Traffic resource manager* consists of action category performed and the length of the action. Also, a notification about every user that leaves the system is sent.

User behavior function simulates the user behavior in terms of switching from one action category to another, and assigning a length to each session segment. The input for this function consists of Markov chain parameters, distributions of session and segment lengths, user specific coefficients, and probabilities of additional media flows. This function primarily communicates with the *Session generator*, which initiates new user generation and termination, while *User behavior function* returns the details about the session (i.e., notification about the type of each new session segment and its length).

Traffic resource manager manages and assigns the resources for virtualized nodes (i.e., range of IP addresses and ports). It takes into account for how long is each resource taken and when it should be released so it can be reassigned to another virtualized traffic sender. *Traffic resource manager* communicates with *Traffic generation controller.* It sends several types of requests: creating virtual nodes, restarting virtual nodes, and initiation of new flows. The request for creating and restarting new virtual nodes consists of the node type (sender or receiver) and virtualized node IP address. Request for initiation of new flows comprises IP addresses of the sender and the receiver, type of the session segment, duration of the session segment, as well as the next free port on which receiver will listen. Communication between *Traffic resource manager* and *Traffic generation controller* is reliable and uses TCP. Multiple *Traffic generation controllers* can be managed by one *Traffic resource manager* (depending on the scale of the simulation).

If more than one *Traffic generation controller* is used, then each one of them is replicated on a single PC used in the simulation. It directly initiates creation and restarting of virtualized traffic senders and receivers, as well as starting of new flows.

Traffic sender and *Traffic receiver* are the components used for sending and receiving network traffic respectively. Traffic models for each action category are built into these components. For each new service to be added and simulated by UrBBaN-Gen the traffic models need to be defined inside these components.

7 Implementation

The whole system is implemented as follows:

- EXtensible Markup Language (XML) files comprising definition of a single service
- Behavior simulator comprising *Control function and user interface* and *Behavior process*
- System for traffic generation control comprising *Traffic resource manager* and *Traffic generation controller*
- Multiple virtualized traffic generators consisting of *Traffic sender* and *Traffic receiver*

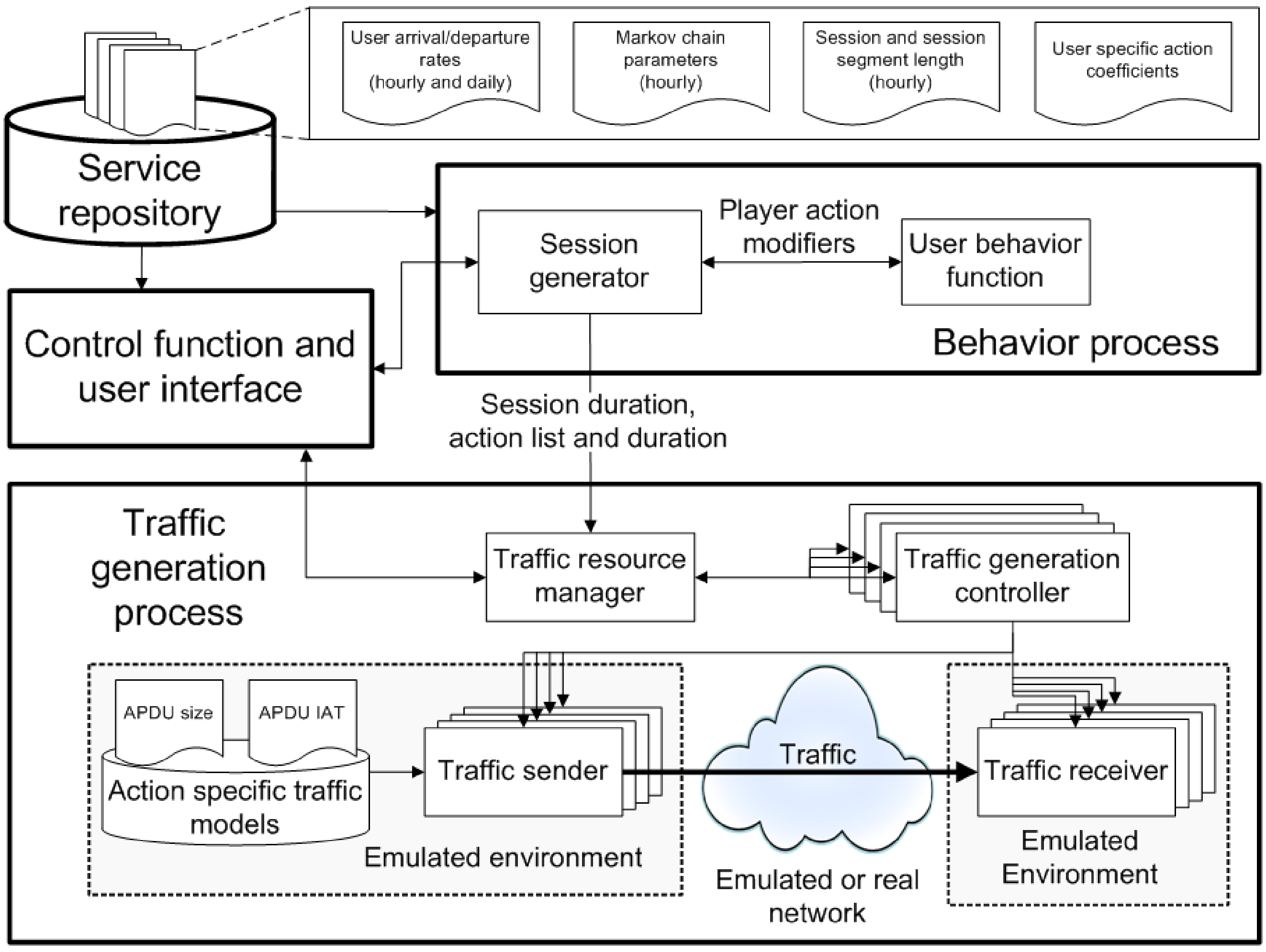

User arrival/departure rates (hourly and daily)
Markov chain parameters (hourly)
Session and session segment length (hourly)
User specific action coefficients
Service repository
Control function and user interface
Session generator
Player action modifiers
User behavior function
Behavior process
Session duration, action list and duration
Traffic generation process
Traffic resource manager
Traffic generation controller
APDU size
APDU IAT
Action specific traffic models
Traffic sender
Emulated environment
Traffic
Emulated or real network
Traffic receiver
Emulated Environment

Fig. 7: Functional architecture of UrBBaN-Gen

7.1 Behavior simulator

In this section we refer to a MMORPG user as “player” so as to avoid confusing him/her with the user of the simulation. To implement the player behavior model we develop a simulator of player behavior. The tool is developed in Java and it is applicable for simulation of different services.

The input data is provided through an XML file. Each service is completely defined with one file. Service definition XML file consists of parameters that define the player behavior model described in the previous section - initial player count, player arrival and departure processes, session segment length and session segment probabilities. Simulation can run on different time basis so the trends and behavior patterns related to certain period of time could be easily observed.

Based on the input data, a user can run a simulation of the whole week or just one day. Also, the simulation can run in real time, or it can be accelerated up to 1500 times. The arrival process defines the creation of new players and the departure process regulates the number of players leaving the simulation. Rates for both processes are, based on user decision, either read from input file or calculated based on the prediction of the number of players. Each player is represented by an individual thread, which simulates the sequence of his/her activities during a session according to the model of session segments probabilities (i.e., Markov chain). Each player thread also “decides” about the duration of each segment based on the distribution for specific action category.

The GUI enables the user to observe the progress of the simulation on two dynamically plotted graphs. One of them shows the number of active players and how the number of players changes through time. Each defined category of player behavior is labeled and marked with a different color so the share of user number participating in a specific action category can be observed. Users can choose to run simulation with additional media flows; in that case the second graph depicts the number of sessions in which there are media flows in addition to game traffic flows. Graphical user interface also enables the control over the parameters specified in the input file (i.e, users can modify the values of initial player count or the session segments duration of each action category).

Users may run the simulation without outputting network traffic to the network interface, or with the network traffic output. In the case that network traffic generation is enabled, simulator inputs data into the system for traffic generation.

Three log files are created at the end of the simulation - a log file with the duration of player sessions, a log file containing the number of active players in the system (taken every ten minutes in the simulated time), and a log file which records the simulation and contains the data about simulation progress including timestamps for actions of arriving and departing player or player's transition to a new action category. After the logs are created, information such as the average number of users in the simulation or the share of action categories are analyzed and compared to data from the model based on the original dataset, as described in the section with results.

7.2 System for traffic generation control

Traffic resource manager is realized as a Java module and is connected to behavior simulator through an API. It takes care about initialization of the virtualized node, initiates every new flow based on the arrival of new session segment, handles the distribution of free ports on receivers, as well as restarting the virtual nodes that reach their flow limit (around 180 flows for senders, and 2000 for receivers). *Traffic generation controller* is replicated on each workstation participating in the simulation and it comprises several Python and shell scripts for the following purposes: receiving the requests from traffic resource manager (over a TCP socket), starting and restarting instances of virtualized senders and receivers, and sending requests to D-ITG senders for initiation of new flows based on D-ITG API.

7.3 Virtualization and traffic generator

For the virtualization of the instances of D-ITG senders and receivers we use Linux Container (LXC) technology. In each LXC, only one instance of D-ITG sender or receiver is running. Senders are run in daemon mode so they can receive multiple flow requests. All LXCs are connected into a Linux bridge as well as the network card interface so the traffic from virtual bridge can be sent to the real network.

In order to validate our network traffic models and create the adequate testbed for their validation we needed a network traffic generator. We decided not to develop our own traffic generator as there is a number of existing solutions in the literature from which we chose the Distributed Internet Traffic Generator (D-ITG) developed at the University of Naples Federico II [1]. D-ITG sender and D-ITG receiver are used as an implementation of the components traffic sender and traffic receiver in our functional architecture. D-ITG is an open-source tool for network traffic generation which offers a choice of various transport and application layer protocols. Implemented application protocols are defined with the underlying transport protocol and the distribution of their payload and inter-departure times. D-ITG's distributed architecture includes sender, receiver, logger, and manager components. More details regarding overview of the architecture and capabilities of the implementation can be found in several publications related to D-ITG [8,10,11].

The traffic models of our action categories are implemented as application protocol models within D-ITG. The modifications needed for implementing traffic models have been made on the sender component. We have used the source code of D-ITG version 2.7.0 Beta2, and we have compiled it on Ubuntu 10.10. The version of the tool we have used to add our action categories already had several existing models for the network traffic with on-line game characteristics. Each action category has been defined with the distributions of APDUs and IATs and accordingly included as a new application protocol. After the models of action categories have been added in the network traffic generator we have generated traffic traces of each category so we could validate that the generated traffic has the required characteristics by comparing parameters of the generated traffic, analytical model, and the validation traffic. Detailed results of the traffic models validation are presented in the next section.

8 Results

In this section we present experimental results through which we validate our models. We compare CDFs of the model, generated traffic, and the traffic from the validation traces to establish how good is the model fit vs validation traces, and how closely the generated traffic follows the parameters set by the model. It should be noted that, as the models were transfered from UDP to TCP, the delayed ACK on receiving machine and Nagle's algorithm in D-ITG sender needed to be disabled.

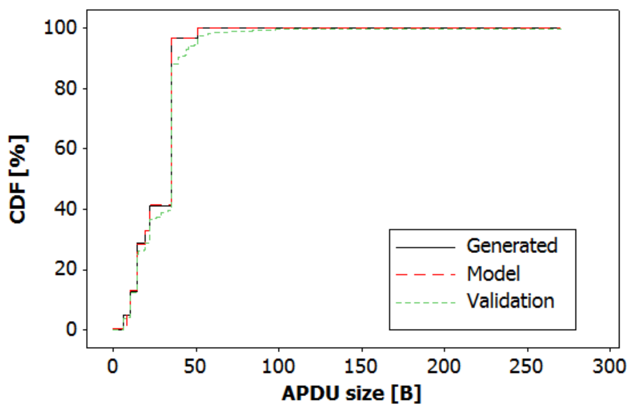


Fig. 8: CDF of *Dungeons* client APDU size

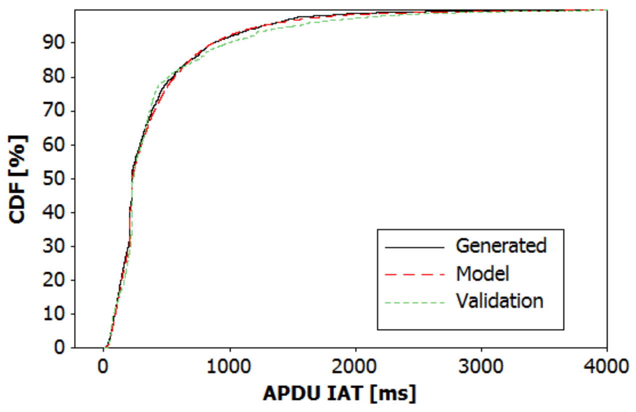


Fig. 10: CDF of *Trading* server APDU IAT

We do not present figures for all parameters through all categories, but select the figures for one of the each parameters for four different action categories as the most representative. Figure 8 shows APDU size of the client generated traffic of *Dungeons* which is closely following the model, and rising slightly faster than the validation data, due to the discrete steps. On all graphs we applied the algorithm for extraction of the whole APDU from the packets described in detail in the methodology section.

Client APDU IAT of the *PvP* category is shown in Figure 9, and the fit is very good, although slightly underestimating in respect to validation traces in the lower segment of the distribution.

The fit of the server APDU IAT of the *Trading* category, shown in Figure 10, is worse than other parameters, but still fairly good. Server APDU IATs were, in general, the most complex to model and to obtain a good fit.

Server APDU size of the *Raiding* category shows a very good fit with only a minor discrepancy, as shown in Figure 11.

We also wanted to confirm that generated traffic characteristics do not vary depending on the simulation scale. To find out how different number of virtualized nodes and flows imposed on each virtualized sender affect the characteristics of generated traffic, we performed a series of scalability experiments. The results are displayed in Figure 12 and Figure 13, with respect to APDU size and IAT, respectively. As it can be seen, there is no significant degradation in the characteristics of the traffic up to 400 flows on 100 virtual nodes in both packet size and packet IAT. After reaching approximately 400 flows, the system enters saturation and the parameters of generated network traffic deteriorate. The PC on which tests are performed uses a dual core i3 processor with 4GB of RAM. It should be noted that the PC under test runs both receivers

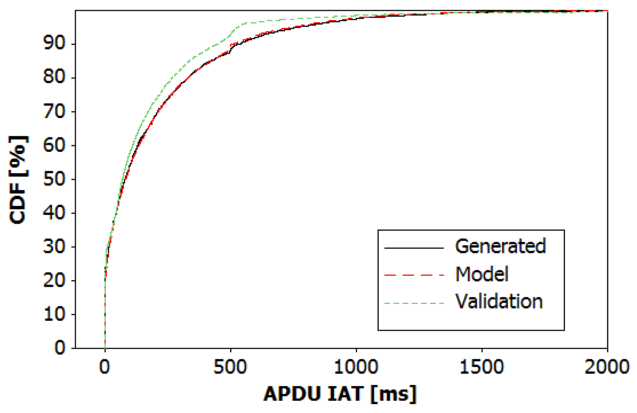


Fig. 9: CDF of *PvP combat* client APDU IAT

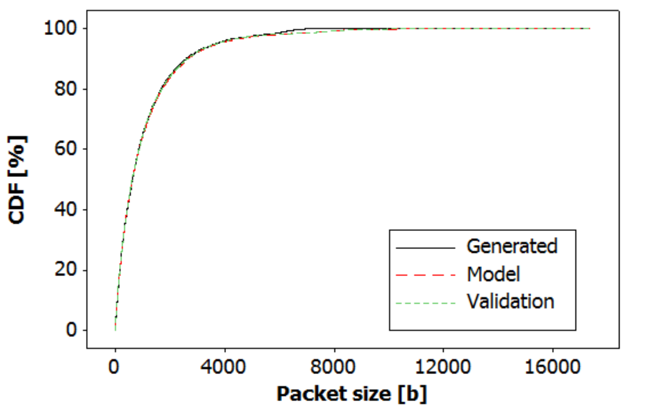
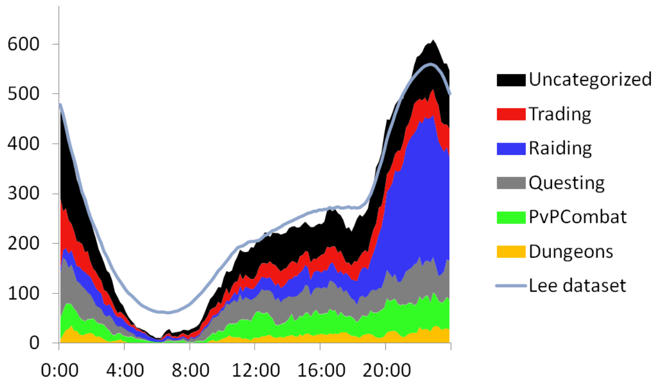
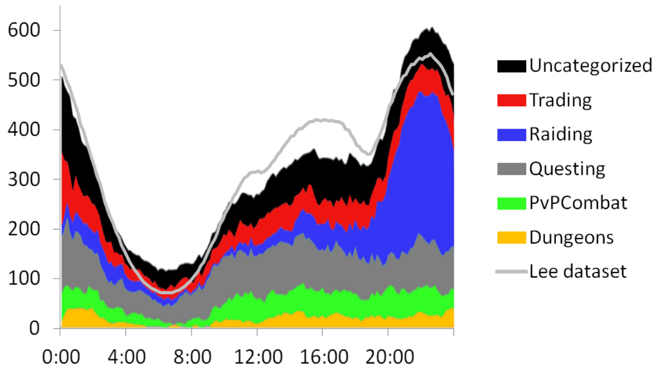


Fig. 11: CDF of *Raiding* server APDU size



(a) Average day



(b) Sunday

Fig. 14: Number of active players per action category (simulated) for a) average day of the week, and b) specific day of the week (Sunday)

and senders. Splitting the process to two PCs would probably yield better results.

In order to validate the results of the player behavior simulation we compare it with the dataset by Lee et al. [33] on which the model is based. We perform the comparison through plotting the average number of users in the dataset obtained through the measurements over the results of one simulation (in Figure 14). We compared a single day of the simulation with the average of all the Lee's data, and a specific simulated day – Sunday with the average of only Sundays from Lee's data. The simulation tends to slightly underestimate the number of users, but captures the main trends such as high decline in the late night, slow increase of players in the morning, and a steep increase in the evening. Also, a higher number of users in the middle of the day through weekends is captured. Figure 15 depicts the relationship of averages of five simulations to the measurements data in terms of percentages of players in each action category during the hours of the day. Most of the points form the diagonal, but slight discrepancies exist, especially in the *Questing* category. Simulation captures *Dungeons* patterns most closely, overestimates *Raiding* in early morning periods, and slightly disperses *Uncategorized*.

As we were unable to access the network traces of WoW from either network or server side, we perform a visual analysis of the graphs by Kihl et al. [25]. Authors of the paper present measurements of the WoW from an access network in Sweden. They depict numbers of players as well as the traffic load generated by those players. In their graphs it is evident that the ratio between load of the network and number of players is not constant. What is especially interesting is that, while there is a lower number of players on Friday and Saturday (around 75% of average number of players

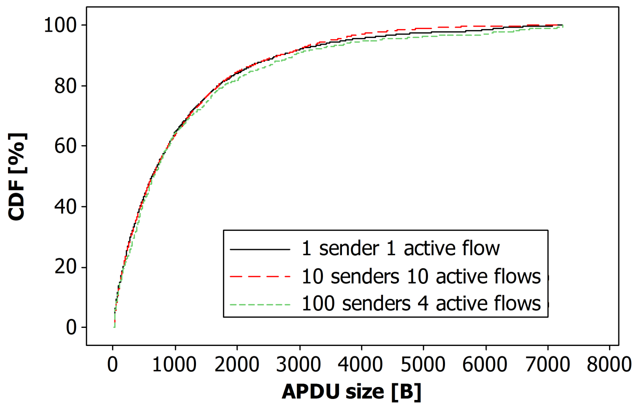


Fig. 12: Scalability test of APDU sizes

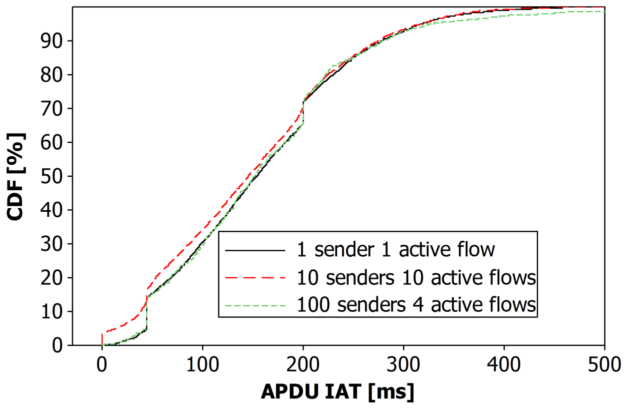


Fig. 13: Scalability test of APDU IATs

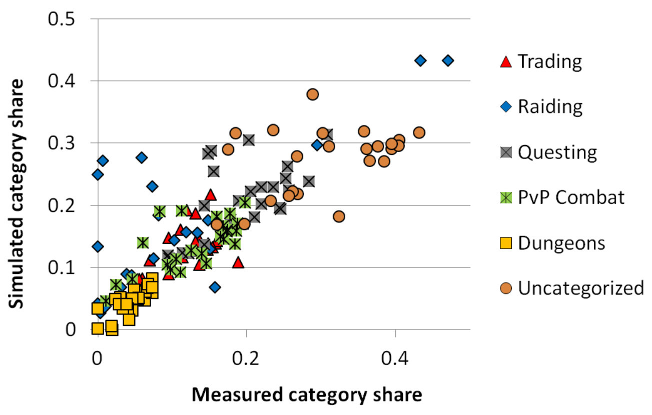


Fig. 15: Simulated vs measured share of action categories

on other days), on those days the traffic peaks are significantly lower (i.e., highest peaks are at 50% of the average value of other days). These results conform to our findings regarding the ratio of specific action types in trace during the days of the week [45], where it was shown that *Raiding* category is halved on Fridays and Saturdays.

Our testbed has enabled us to perform simulations in order to see to which extent player behaviors affect network load per player. Results indicate that the network load per player increases 3.1 times from the lowest to the highest value, with the peak value being 1.6 times higher than average value. When we included simulation of the media flows based on the characteristic of the codec used by WoW voice chat [2], the results showed 3.2 times higher peak value than the lowest value, and 1.7 times higher peak value than the average value. Also, the highest load per user overlaps with the highest number of active users, further emphasizing this phenomenon. This fact needs to be taken into account when performing efficient network capacity planning for MMORPGs. Currently most MMORPGs avoid these issues with heavy over-provisioning.

9 Conclusion

In this paper we presented models for player behavior based on action categories, as well as the network traffic models for each action category. Also, we have introduced an architecture for scalable player behavior based traffic generation. Traffic models have been implemented in an existing traffic generator D-ITG. We implemented the player behavior models through a simulation tool written in Java. We developed the system for traffic generation control using Java, Python, and shell scripts. The architecture has been deployed in the laboratory testbed, and validation and scalability experiments have been performed. Through our testbed we confirm the assumption that network load does not only depend on the number of players, but also and even more significantly, on player behavior (i.e., network load per player varies up to a factor of 3.2).

Our architecture enables generation of a large number of streams based on user behavior specified by model parameters. The traffic generation system is easily deployed, and yields a large number of simulated users on relatively cheap hardware. It should be noted that the user behavior simulator is realized as an separate module, and can be used as a tool for testing in other areas related to MMORPGs such as algorithms for virtual world partitioning, methods for churn analysis, server load optimization, etc. Also, different services which can be specified through the parameters of this model can be simulated, so the applications of both the whole system, and the behavior simulator alone are not limited to the area of MMORPGs.

In the future work we aim to perform additional scalability measurements, and prove general applicability of our architecture through implementation of other services.

Acknowledgment

This work was supported by the research project 036-0362027-1639 “Content Delivery and Mobility of Users and Services in New Generation Networks”, by the Ministry of Science, Education, and Sports of the Republic of Croatia, and Ericsson Nikola Tesla, Zagreb, Croatia. Also, the research leading to these results has received funding from the European Community’s Seventh Framework Programme under grant agreement no. 285939 (ACROSS).

References

1. Distributed Internet Traffic Generator (2010). URL `http://www.grid.unina.it/software/ITG/`
2. SPIRIT IP-MR - dedicated VoIP codec (2011). URL `http://www.spiritdsp.com/datasheets/SPIRIT-IP-MR-Datasheet.pdf`
3. World of Warcraft (2011). URL `http://www.battle.net/wow/`
4. World of Warcraft ace3 addon development framework (2011). URL `http://www.wowace.com/addons/ace3/`
5. World of Warcraft API. World of Warcraft wiki pages. Activision Blizzard (2011). URL `http://www.wowwiki.com/World\_of\_Warcraft\_API`
6. List of all MMO games (active/developed) (2012). URL `http://www.mmorpg.com/gamelist.cfm`

7. Antonello, R., Fernandes, S., Moreira, J., Cunha, P., Kamienski, C., Sadok, D.: Traffic analysis and synthetic models of Second Life. Multimedia Tools and Applications **15**(1), 33–47 (2009)
8. Avallone, S., Guadagno, S., Emma, D., Pescap?, A., Venturi, G.: D-ITG Distributed Internet Traffic Generator. In: Proceeding of International Conference on Quantitative Evaluation of Systems, pp. 316–317 (2004)
9. Borella, M.S.: Source models of network game traffic. Computer Communications **23**(4), 403–410 (2000)
10. Botta, A., Dainotti, A., Pescap?, A.: Do you trust your software-based traffic generator? Communications Magazine, IEEE **48**(9), 158 – 165 (2004)
11. Botta, A., Dainotti, A., Pescapè, A.: Multi-protocol and Multi-platform Traffic Generation and Measurement. In: INFOCOM 2007, 26th IEEE International Conference on Computer Communications, Demonstration Session (2007)
12. Chen, K.T., Huang, P., Lei, C.L.: Game Traffic Analysis: An MMORPG Perspective. Computer Networks **51**(3), 19–24 (2007)
13. Chen, K.T., Huang, P., Wang, G.S., Huang, C.Y., Lei, C.L.: On the sensitivity of online game playing time to network QoS. In: Proceedings of the IEEE International Conference on Computer Communications INFOCOM'06, pp. 1 – 12 (2006)
14. Chen, K.T., Lei, C.L.: Network game design: Hints and implications of player interaction. In: Proceedings of the 5th ACM SIGCOMM Workshop on Network and System Support for Games, pp. 1–9 (2006)
15. Choi, Y., Kim, H., Park, C., Jin, S.: A case study: Online game testing using massive virtual player. In: Proceedings of Control and Automation, and Energy System Engineering - International Conferences, CA and CES3 2011, Held as Part of the Future Generation Information Technology Conference, FGIT 2011, in Conjunction with GDC 2011, pp. 296–301 (2011)
16. Cricenti, A., Branch, P.: ARMA(1,1) modeling of Quake4 server to client game traffic. In: NetGames '07: Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games, pp. 70–74 (2007)
17. Dainotti, A., Botta, A., Pescapé, A., Ventre, G.: Searching for invariants in network games traffic. In: Proceedings of the 2006 ACM CoNEXT conference, pp. 43:1–43:2 (2006)
18. Dainotti, A., Pescape, A., Ventre, G.: A packet-level traffic model of Starcraft. In: HOT-P2P '05: Proceedings of the Second International Workshop on Hot Topics in Peer-to-Peer Systems, pp. 33–42 (2005)
19. Danzig, P., Jamin, S.: tcplib: A Library of TCP Internetwork Traffic Characteristics. Tech. Rep. CS-SYS-91-01, Computer Science Department, University of Southern California (1991)
20. Färber, J.: Network game traffic modelling. In: NetGames '02: Proceedings of the 1st workshop on Network and system support for games, pp. 53–57 (2002)
21. Feng, W.C., Brandt, D., Saha, D.: A long-term study of popular MMORPG. In: Proceedings of the 6th ACM SIGCOMM Workshop on Network and System Support for Games, pp. 19–24 (2007)
22. Geel, I.V.: Tracking of active MMOG subscriptions (2011)
23. Gleser, L.J., Moore, D.S.: The Effect of Dependence on Chi-Squared and Empiric Distribution Tests of Fit. The Annals of Statistics **11**(4), 1100–1108 (1983)
24. Kawale, J., Pal, A., Srivastava, J.: Churn prediction in MMORPGs: A social influence based approach. In: Proceedings of the International Conference on Computational Science and Engineering, pp. 423–428 (2009)
25. Kihl, M., Aurelius, A., Lagerstedt, C.: Analysis of World of Warcraft Traffic Patterns and User behavior. In: International Congress on Ultra Modern Telecommunications and Control Systems and Workshops, pp. 218–223 (2010)
26. Kim, J., Choi, J., Chang, D., Kwon, T., Choi, Y., Yuk, E.: Traffic characteristics of a Massively Multiplayer Online Role Playing Game. In: Proceedings of 4th ACM SIGCOMM Workshop on Network and System Support for Games, pp. 1–8 (2005)
27. Kim, J., Hong, E., Choi, J.: Measurement and Analysis of a Massively Multiplayer Online Role Playing Game Traffic. In: Proceedings of Advanced Network Conference, pp. 1–8 (2003)
28. Lakkakorpi, J., Heiner, A., Ruutuc, J.: Measurement and characterization of Internet gaming traffic. Tech. rep., Espoo (2002). Research Seminar on Networking
29. Lang, T., Armitage, G.: An ns2 model for the Xbox system link game Halo. Tech. Rep. 030613A, Swinburne University of Technology, Faculty of Information and Communication Technologies, Center for Advanced Internet Architectures (2003)
30. Lang, T., Armitage, G., Branch, P., Choo, H.Y.: A synthetic traffic model for Half-Life. In: in Australian Telecommunications, Networks and Applications Conference (ATNAC (2003)
31. Lang, T., Branch, P., Armitage, G.: A synthetic traffic model for Quake3. In: ACE '04: Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology, pp. 233–238 (2004)
32. Lee, Y.T., Chen, K.T.: Is server consolidation beneficial to MMORPG? A case study of World of Warcraft. In: IEEE International Conference on Cloud Computing, pp. 435–442 (2010)
33. Lee, Y.T., Chen, K.T., Cheng, Y.M., Lei, C.L.: World of Warcraft avatar history dataset. In: Proceedings of the second annual ACM conference on Multimedia systems, pp. 123–128 (2011)
34. Matijasevic, M., Gracanin, D., Valavanis, K.P., Lovrek, I.: A framework for multiuser distributed virtual environments. IEEE Transactions on Systems, Man, and Cybernetics, Part B pp. 416–429 (2002)
35. Park, H., Kim, T., Kim, S.: Network traffic analysis and modeling for games. In: Internet and Network Economics, Lecture Notes in Computer Science, pp. 1056–1065. Springer Berlin / Heidelberg (2005)
36. Paxson, V.: Empirically-Derived Analytic Models of Wide-Area TCP Connections. IEEE/ACM Transactions on Networking **2**(2), 316–336 (1994)
37. Pederson, S.P., Johnson, M.E.: Estimating model discrepancy. Technometrics **32**(3), 305–314 (1990)
38. Pittman, D., Dickey, C.G.: Characterizing Virtual Populations in Massively Multiplayer Online Role-Playing Games. In: Advances in Multimedia Modeling, *Lecture Notes in Computer Science*, vol. 5916, pp. 87–97. Springer Berlin / Heidelberg (2010)
39. Pittman, D., GauthierDickey, C.: A Measurement Study of Virtual Populations in Massively Multiplayer Online Games. In: Proceedings of the 6th ACM SIGCOMM Workshop on Network and System Support for Games, pp. 25–30 (2007)
40. Scott, D.W.: On optimal and data-based histograms. Biometrika **66**(3), 605–610 (1979)

41. Shin, K., Kim, J., Sohn, K., Park, C., Choi, S.: Online gaming traffic generator for reproducing gamer behavior. In: Proceedings of the 9th international conference on Entertainment computing, pp. 160–170 (2010)
42. Shin, K., Kim, J., Sohn, K., Park, C.J., Choi, S.: Transformation Approach to Model Online Gaming Traffic. ETRI Journal **33**(2), 219–229 (2011)
43. Simic, J.: Visualisation of information regarding player behavior in networked virtual envirnment (in Croatian). Diploma thesis, University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, Croatia (2011)
44. Suznjevic, M.: WoW add-on for measuring player behavior (2011). URL `http://www.fer.hr/tel/en/research/research\_groups/netmedia/software`
45. Suznjevic, M., Dobrijevic, O., Matijasevic, M.: Hack, slash, and chat: A study of players' behavior and communication in MMORPGs. In: Proceedings of the 8th Workshop on Network and System Support for Games, p. 6 (2009)
46. Suznjevic, M., Dobrijevic, O., Matijasevic, M.: MMORPG player actions: Network performance, session patterns and latency requirements analysis. Multimedia Tools and Applications **45**(1-3), 191–241 (2009)
47. Suznjevic, M., Matijasevic, M.: Why MMORPG players do what they do: relating motivations to action categories. International Journal of Advanced Media and Communication **4**(4), 405–424 (2009)
48. Suznjevic, M., Matijasevic, M., Dobrijevic, O.: Action specific Massive Multiplayer Online Role Playing Games traffic analysis: Case study of World of Warcraft. In: Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games, pp. 106–107 (2008)
49. Suznjevic, M., Stupar, I., Matijasevic, M.: MMORPG player behavior model based on player action categories. In: Proceedings of the 10th Workshop on Network and System Support for Games, p. 6 (2011)
50. Suznjevic, M., Stupar, I., Matijasevic, M.: Traffic modeling of player action categories in a MMORPG. In: Proceedings of the 2nd workshop on DIstributed SImulation and Online gaming (DISIO), p. 8 (2011)
51. Svoboda, P., Karner, W., Rupp, M.: Traffic analysis and modeling for World of Warcraft. In: Communications, 2007. ICC '07. IEEE International Conference on, pp. 1612–1617 (2007)
52. Szabó, G., Veres, A., Molnár, S.: On the impacts of human interactions in MMORPG traffic. Multimedia Tools and Applications **45**(1-3), 133–161 (2009)
53. Tarng, P.Y., Chen, K.T., Huang, P.: An analysis of WoW players' game hours. In: Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games, pp. 47–52 (2008)
54. Tyvand, J.E., Begnum, K., Hammer, H.: Deja vu - Predicting the number of players in online games through normalization of historical data. In: Proceedings of 10th workshop on Network and system support for games, NetGames '11, p. 13:2 (2011)
55. Williams, D., Yee, N., Caplan, S.E.: Who plays, how much, and why? Debunking the stereotypical gamer profile. Journal of Computer-Mediated communication **13**(1), 993–1018 (2008)
56. Wu, Y., Huang, H., Zhang, D.: Traffic Modeling for Massive Multiplayer On-line Role Playing Game (MMORPG) in GPRS Access Network. In: Proceedings of the International Conference on Communications, Circuits and Systems Proceedings, pp. 1811–1815 (2006)
57. Zander, S., Armitage, G.: A traffic model for the Xbox game Halo 2. In: Proceedings of the international workshop on Network and operating systems support for digital audio and video, pp. 13–18 (2005)
58. Zhuang, X., Bharambe, A., Pang, J., Seshan, S.: Player Dynamics in Massively Multiplayer Online Games. Tech. Rep. CMU-CS-07-158, School of Computer Science, Carnegie Mellon University, Pittsburgh (2007)