5th ISCA/DEGA Workshop on Perceptual Quality of Systems (PQS 2016)
29-31 August 2016, Berlin, Germany

# Impact of Simplemux Traffic Optimisation on MMORPG QoE

*Mirko Suznjevic*[1], *Jose Saldana*[2], *Maja Matijasevic*[1], *Matko Vuga*[1]

[1]University of Zagreb Faculty of Electrical Engineering and Computing
[2]Aragon Inst. of Engineering Research (I3A), University of Zaragoza

mirko.suznjevic@fer.hr, jsaldana@unizar.es, maja.matijasevic@fer.hr, matko.vuga@fer.hr

## Abstract

This paper studies the impact of a traffic optimisation method called Simplemux, on Quality of Experience (QoE) of Massively Multiplayer Online Role Playing Games (MMORPG). Simplemux allows a number of header-compressed packets of different flows to travel together, sharing the overhead of a common end-to-end tunnel. As a result, significant reductions in terms of bandwidth usage and packet rate can be achieved, but with a trade-off of introducing additional delay and delay variation. In the presented subjective studies we combine artificially created network flows of an MMORPG with traffic of real users, and this mix is sent through an implementation of Simplemux. In this way we simulate a case scenario in which a number of flows share a common network path. Two parameters are modified: the number of active game flows, and the multiplexing period. We examine if the players notice any degradation in their game experience, by comparing two situations: when Simplemux is active or not. Our goal is to investigate whether users' QoE is degraded by using Simplemux.

**Index Terms**: traffic optimisation, Quality of Experience, massively multiplayer online role-playing games, user study

## 1. Introduction

From a network point of view, interactive applications, such as games, have to send updates at a very fast pace to maintain the perception of responsiveness. The content of each update is relatively small, so the traffic profile of these services consists of small packets. As a result, these applications produce traffic flows comprised of high rates of small packets. Similar traffic features can also be observed in non-real time applications such as instant messaging or Machine to Machine (M2M) services. Such *small-packet services*, as they are sometimes called [1], result in a reduction of the overall network efficiency, as their payload-to-header ratio is very low, so effective application-level bandwidth usage is reduced.

One solution to this problem is to have the packets buffered before being sent in a multiplexed bundle, with the counterpart of introducing an additional delay in the communication and also some variation of the delay (jitter). Multiplexing brings the possibility of reducing packet rate and bandwidth usage in exchange for a small increase of network delay and some jitter. Nevertheless, when a sufficient number of flows share a network path, significant savings can be achieved with addition of very minor values of delay and jitter. The present work specifically addresses the problem of the impact of this delay and jitter on players' Quality of Experience (QoE). We examine the effect of multiplexing on subjective quality for *World of Warcraft* (WoW) which is a TCP-based Massively Multiplayer Online Role Playing Game (MMORPG). We have selected Simplemux [2] as an example of a multiplexing solution for our tests, although the results are also applicable to any other aggregation technique.

In our use case we optimise traffic flows from one server of an MMORPG, and specifically look at the influence at the user level. In this study we keep the round trip time (RTT) under 100 ms at all times, which is considered as *good* even for more interactive games such as First Person Shooters [3]. For MMORPGs there are conflicting results regarding the impact of delay on player's QoE [4, 5], but the most conservative results suggest that for Mean Opinion Score (MOS) above 4, RTT should be kept under 120 ms [4]. As regards to variation of delay, the same work [4] stated that variation of delay of 10 ms has significant impact on QoE. Therefore, we want to make a subjective study evaluating Simplemux effects on QoE.

For that aim, we conduct a subjective users' study in which the players compare their QoE in the case in which Simplemux optimisation is run versus a case in which no optimisation occurs. Our goal is to evaluate whether the QoE of the players would be degraded when the Simplemux optimisation is used, but always maintaining the overall RTT low enough (below 100 ms, including the delay introduced by Simplemux). In other case, based on the scores of the evaluation parameters, multiplexing parameters can be adjusted to ensure high QoE.

The remainder of the article is organized as follows: Section 2 describes the related work, Section 3 presents Simplemux in more detail. Section 4 contains the information about the methodology, namely the details regarding laboratory testbed and the approach to performing the user study. Section 5 describes the results, and we conclude the paper in Section 6.

## 2. Related Work

In order to improve the traffic efficiency, several optimisation techniques have been developed, based on aggregating a number of packets on different network layers. On layer 2, 802.11n [6] (and subsequent versions) include two frame aggregation policies: MAC Service Data Unit aggregation (A-MSDU) and MAC Protocol Data Unit aggregation (A-MPDU). The efficiency improvement is significant: A-MPDU can increase channel utilisation from 33% to 95% in some cases [7].

PPPMux [8] is an extension of PPP allowing the multiplexing of a number of packets into a single frame. Multiplexing at higher layers i.e., for VoIP optimisation was studied in [9], which focused on keeping the speech quality while performing multiplexing. Multiplexing can effectively be combined with tunneling and header compression [10] in order to amortize between a number of packets the overhead caused by the tunnel header [11]. Another aggregation proposal is Simplemux [2], a generic and lightweight protocol aimed to enable the multiplexing of a number of packets belonging to a protocol (the *multiplexed* packets), into another protocol (i.e. the *tunneling*

10.21437/PQS.2016-4

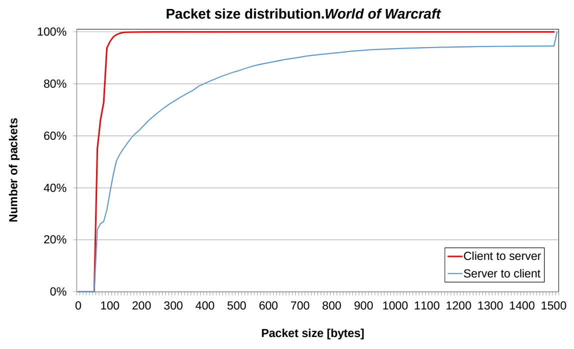

Figure 1: Packet size histogram of a traffic trace of *World of Warcraft*

protocol) [12]. PPPMux [8] and Simplemux [2], have two additional advantages: 1) aggregation can cover a shared path in the network instead of being run in a single hop; and 2) it can be combined with tunneling and header compression [10] in order to save bandwidth, as done in [11] for VoIP. The savings are especially significant in the case of small-packet services, where the size of the header and the payload are in the same order of magnitude.

Regarding subjective quality estimators, some of them have been proposed for different real-time services. They are based on tests with real users, who fill in a form reporting their subjective experience. Network conditions are modified (in terms of delay, jitter or packet loss) in order to capture the influence of each parameter on the subjective experience. Finally, a formula is devised based on the subjective data. The ITU E-Model [13] was proposed to estimate subjective quality of VoIP. Some models have also been proposed for games as e.g. [14] for a First Person Shooter or [4] for an MMORPG. Our work differs from those subjective studies as we do not directly try to map the impact of specific network parameters onto players' QoE, but we aim to investigate whether Simplemux traffic optimisation based on multiplexing, can be performed without QoE degradation. If the QoE is not degraded for a real-time service such as an MMORPG, this means that such optimisation can be easily applied to other services which do not have such tight delay requirements.

## 3. Simplemux

To illustrate the problem of small packets in the network, in Figure 1 we present the packet-size distribution of a traffic trace from WoW, an MMORPG based on TCP. It can be observed that the size distribution of client-to-server packets ends in 200 bytes, with a significant amount of pure TCP ACKs (40 bytes). In contrast, the size of server-to-client packets has a wider range, although a significant amount of them are also small.

A solution for the presented problem are traffic optimisation techniques. Several of them are available, mostly relying on compression of headers and multiplexing of packets on various network layers. One of such methods is Simplemux [2], a generic and lightweight multiplexing protocol aimed to enable the multiplexing of a number of packets belonging to a protocol, into another protocol [12]. As shown in Figure 2, Simplemux creates a tunnel covering a network path between an *ingress* and an *egress* machine. These machines may be the endpoints of the communication, but they may also be middleboxes able to multiplex packets belonging to different flows. This tunnel allows header-compressed packets to travel end-to-end, avoiding the need to compress/decompress headers at each node. Therefore, a number of compressed packets of different flows can travel together, sharing the tunnel overhead. As a result, significant reductions of bandwidth usage and packet rate can be achieved [2].

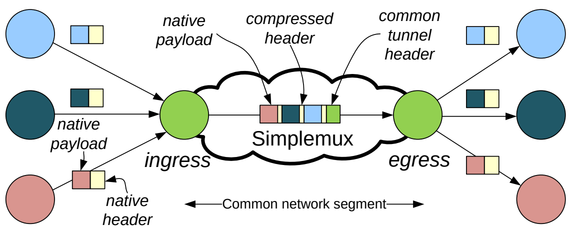

Figure 2: Scheme of Simplemux

Simplemux can work at Network or Transport layers, i.e. multiplexed packets can travel as IP or UDP payload respectively. It does not require a session setup, since the idea is that a specific protocol number (Network layer) or port (Transport layer) could be reserved to it. Thus, if the destination machine receives a packet with this *protocol* or *port number*, and if it implements Simplemux, it can demultiplex and extract the packets. There is no need to set up a session or to store any session parameter.

There are different policies which can be employed for aggregating small into large packets: 1) *fixed number* of packets - once a fixed number of packets has arrived, a multiplexed packet is created and sent; 2) *size limit* - once a size limit is reached (e.g., next to the MTU of the underlying network), a multiplexed packet is sent; and 3) *period* - a multiplexed packet is sent every time period. The combination of *size limit* and *period* policies enables control the additional delay introduced by multiplexing while maintaining high bandwidth usage savings: a multiplexed packet is sent at the end of each period but, if the size limit is reached, a multiplexed packet is sent immediately, and the period is reset. Thus, the added delay is for the worst case scenario equal to the defined period.

When multiplexing is based on a time period, a sawtooth-shaped delay is added to the packets [15], i.e., those arriving at the beginning experience a big delay, whereas those arriving at the end are sent almost immediately. The average delay is half the period, and some jitter is also added.

## 4. Methodology

### 4.1. Laboratory testbed

To enable testing we have first integrated Simplemux into Integrated Multiprotocol Network Emulator/Simulator (IMUNES) (http://imunes.net/) [16]. IMUNES was originally built on a FreeBSD platform, but as of recently a Linux-based implementation is also available. Using IMUNES, complex network architectures can be simulated on one machine, so there is no need for a physical testbed comprising multiple PCs and routers.

As we previously stated, as a use case MMORPG we use WoW. To generate realistic network flows associated with a WoW server, we use the User Behaviour Based Network Traffic Generator (UrBBaN-Gen) architecture [17], which provides player behaviour based network traffic generation. UrBBaN-Gen consists of three main components: User Behaviour Simulator (UBS), Distributed Traffic Generation Control System

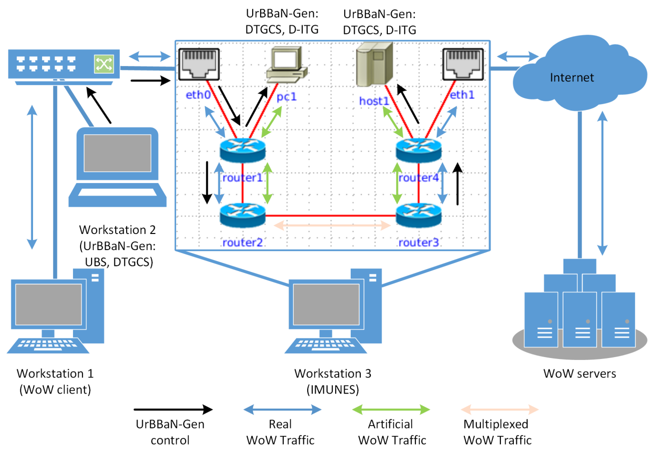

Figure 3: Laboratory testbed

(DTGCS), and Distributed Internet Traffic Generator (D-ITG) as a module for creating packets [18]. Both physical environment and simulated environment are presented Figure 3 which depicts the testbed. On workstation 1 WoW client is run and in the testing procedure the players play on this device. Workstation 2 hosts two parts of UrBBaN-Gen are run: UBS and server part of DTGCS. Both workstations are connected via Ethernet switch with workstation 3 which has 2 network interfaces (interfaces *eth0* and *eth1*) and runs IMUNES. IMUNES has the ability to include the interfaces from the workstation on which it is hosted to the simulation and through that include the traffic outside of the simulation and run it through the simulated network. Within IMUNES on simulated workstations *pc1* and *host1* other elements of UrBBaN-Gen are hosted: part of DTGCS and D-ITG network traffic senders (on *host1*, and part of DTGCS and D-ITG network traffic receivers (on *pc1*).

In this way the real WoW traffic flowing from the client to the server and back shares the network path with the artificially generated flows within the simulation. Based on the data from UBS, DTGCS controls the D-ITG network traffic senders and receivers which are run within the IMUNES simulation. In the simulation, server-to-client traffic of different action categories is generated, as defined in [19]. On the shared path the traffic is optimised using Simplemux located on *Router 2* and *Router 3*.

### 4.2. Subjective tests

Tests focus on users playing WoW, testing whether they notice the difference between a scenario in which Simplemux is run, compared to the scenario in which Simplemux is not run. We simulate different numbers of active artificial flows (i.e., 10, 20, 30, 50 and 600 flows), as well as different multiplexing periods (i.e., 10, 20, 30, 50 ms). We refer to each combination of numbers of artificially generated flows and the multiplexing period as a "scenario". The scenarios that have been tested are listed in Table 1. Each user has tested all the scenarios.

For comparison of gameplay between the case in which Simplemux is used and the one in which it is not, we used *paired comparison*, as defined in ITU-T P.800 [20] and ITU-R BT.500-13 [21]. In the Comparison Category Rating (CCR) method testers are presented with a pair of samples on each scenario. The order of the affected and unaffected sample is chosen at random for each scenario. In the CCR procedure, the 7pt. comparison scale, presented in Table 2, is used to judge the quality of the second sample relative to that of the first.

| Scenario | No. of generated flows | Period [ms] |
|---|---|---|
| 1 | 600 | 10 |
| 2 | 50 | 50 |
| 3 | 50 | 10 |
| 4 | 20 | 20 |
| 5 | 20 | 10 |
| 6 | 10 | 50 |
| 7 | 10 | 30 |

Table 1: Tested scenarios

| | |
|---|---|
| 3 | Much Better |
| 2 | Better |
| 1 | Slightly Better |
| 0 | About the Same |
| $-1$ | Slightly Worse |
| $-2$ | Worse |
| $-3$ | Much Worse |

Table 2: 7pt. comparison scale

The procedure for testing is as follows: first, the testing user would create a new character in WoW and play for a few minutes to get familiar with the commands. Then, the testing procedure would start. In each scenario, two different settings were tested (i.e., Simplemux on and off). Between both settings, the player has to be disconnected from the virtual world as the initialisation/shutdown process for Simplemux resets network interfaces within IMUNES so the connection with the WoW servers is severed. The duration of gameplay for each setting was one minute. After one scenario was completed (i.e., both settings in scenario were tested), the user would: 1) compare the QoE of the gameplay of the second setting with the first one on the 7pt. scale described above and 2) answer whether he/she would continue playing in these conditions. Additionally, RTT was noted for each setting (as reported in the WoW client).

The participants of the study were 10 males, aged from 22 to 33. All the test participants were experienced gamers. The testing procedure lasted between 30 and 35 minutes on average.

## 5. Results

The order in which each player performed both tests was random. Therefore, for the presentation of results, we normalised the CCR scores so as to present all scores considering Simplemux as the second setting (i.e., we reversed the algebraic sign in the scenarios in which Simplemux enabled was the first setting within a scenario).

The results for scenarios 1-6 are depicted in Figure 4. For scenario 7 we do not display the data, as all CCR scores were 0 ("About the same"). It can be observed that in every scenario at least 7 test participants did not discriminate between both settings. No CCR score of 3, -3, and -2 was given, meaning that at the worst case, the setting in which Simplemux was active, was graded as "Slightly worse". What is interesting is that in two cases the scenario in which Simplemux was active was graded as "Better" (CCR score 2) than the scenario without Simplemux. The results indicate that for majority of time (81.43%) the users were not able to appreciate the difference. What is more important, all users in all scenarios stated that they would continue playing under these conditions. Therefore, we can conclude that the use of Simplemux in our scenarios did not introduce degradation at a scale that would impact players' willingness to play.

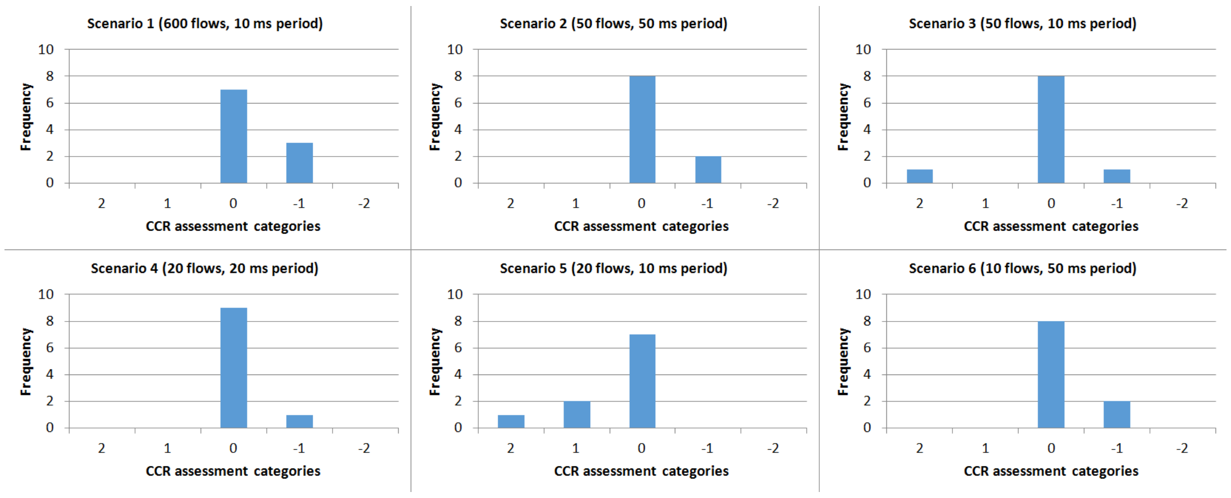

Figure 4: Subjective scores for scenarios 1-6

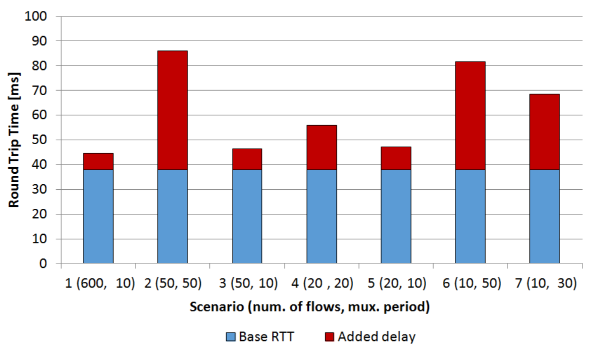

Figure 5: RTT for all scenarios

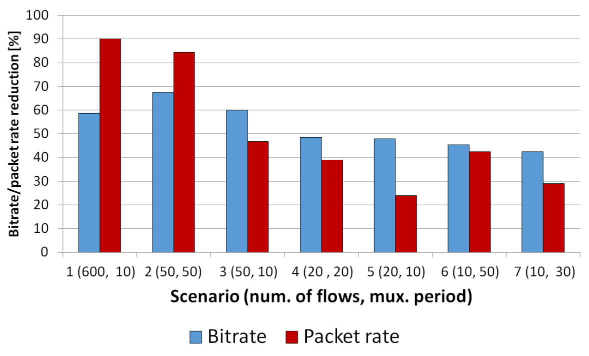

Figure 6: Reduction of the bitrate and packet rate per scenario

The RTT from our laboratory to the WoW server was 38 ms (as reported in the WoW client) when the Simplemux optimisation was not enabled (base RTT). In Figure 5 we present the average RTT measured in every scenario. It should be noted that RTT never reached 100 ms for any user. Half of the multiplexing period is added as delay in one direction, resulting in an average latency increase of one multiplexing period. Added latency also depends on the number of generated additional flows so it is lower in cases with higher number of users, where the MTU can be reached. When comparing CCR results with the introduced latency, no clear correlation is observed, meaning that the users do not notice the discrepancies in the latency. This could be expected, as one of our goals was to keep the latency below 100 ms. Another important finding coming from these results is that delay variation introduced by multiplexing is neither registered by the players in our scenarios.

The results regarding bandwidth usage savings and reduction of the packet rate were obtained by capturing network traffic on the *router 2*, before and after Simplemux multiplexing. Reduction of the bandwidth usage and packet rate is depicted in Figure 6. Very significant packet rate reduction (up to 88%) can be observed, and bandwidth usage reduction can be up to 68%. The higher the number of concurrent flows that are multiplexed, the better both the bandwidth usage savings and the packet rate reduction. All in all, the results indicate that bandwidth usage savings and especially significant packet rate reduction can be obtained without degrading MMORPG players' QoE, as long as the overall latency is kept under 100 ms.

## 6. Conclusions

In this study we performed a subjective evaluation of Simplemux method for traffic optimisation for a use case of a real time service - MMORPG. The main constraint of the subjective study is that the overall latency in all cases must be kept under 100 ms. The results of the study showed that even with a small number of flows (such as 10), and 50 ms multiplexing period, aggregation of flows using Simplemux results in a reduction of bandwidth usage and packet rate up to 40% without degrading the players' QoE. In cases in which the multiplexing period must be kept low to maintain overall latency below 10 ms, for already 20 players bandwidth saving of 45% and packet savings of 20% can still be achieved. Another important finding is that the introduced latency variation which can be up to half of the multiplexing period does not prove to be a problem for the players, who do not notice it.

## 7. Acknowledgements

This work has been partially financed by the EU H2020 Wi-5 project (G.A. no: 644262), and the Spanish Ministry of Economy and Competitiveness project TIN2015-64770-R, in cooperation with the European Regional Development Fund. Also, this work has been supported by Croatian Science Foundation under the project 8065 (HUTS) and project UIP-2014-09-5605 (Q- MANIC).

## 8. References

[1] Huawei, "Smartphone Solutions White Paper," July 2012. [Online]. Available: http://www.huawei.com/mediafiles/CBG/PDF/Files/hw_193034.pdf

[2] J. Saldana, "Simplemux. a generic multiplexing protocol," *draft-saldana-tsvwg-simplemux-02*, 2015.

[3] M. Claypool and K. Claypool, "Latency and player actions in online games," *Commun. ACM*, vol. 49, no. 11, pp. 40–45, Nov. 2006. [Online]. Available: http://doi.acm.org/10.1145/1167838.1167860

[4] M. Ries, P. Svoboda, and M. Rupp, "Empirical study of subjective quality for Massive Multiplayer Games," in *2008 15th International Conference on Systems, Signals and Image Processing*, Jun. 2008, pp. 181–184.

[5] M. Suznjevic, L. Skorin-Kapov, and M. Matijasevic, "Impact of User, System, and Context factors on Gaming QoE: a Case Study Involving MMORPGs," in *Proceedings of the 12th ACM SIGCOMM Workshop on Network and System Support for Games*, 2013, pp. 1–6.

[6] "IEEE 802.11n IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput," 2009.

[7] B. Ginzburg and A. Kesselman, "Performance analysis of A-MPDU and A-MSDU aggregation in IEEE 802.11n," in *2007 IEEE Sarnoff Symposium*, 2007, pp. 1–5.

[8] R. Pazhyannur, I. Ali, and C. Fox, "RFC3153," *PPP Multiplexing, Aug*, 2001.

[9] R. M. Pereira and L. M. R. Tarouco, "Adaptive multiplexing based on E-model for reducing network overhead in voice over IP security ensuring conversation quality," in *Fourth International Conference on Digital Telecommunications ICDT'09*. IEEE, 2009, pp. 53–58.

[10] G. Pelletier and K. Sandlund, "The RObust Header Compression (ROHC) Framework." [Online]. Available: https://tools.ietf.org/html/rfc5795

[11] B. Thompson, D. Wing, and T. Koren, "Tunneling Multiplexed Compressed RTP (TCRTP)." [Online]. Available: https://tools.ietf.org/html/rfc4170

[12] J. Saldana, I. Forcen, J. Fernandez-Navajas, and J. Ruiz-Mas, "Improving network efficiency with simplemux," in *IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM)*. IEEE, 2015, pp. 446–453.

[13] Recommendation, ITU-T, "G.107: The E-model: a computational model for use in transmission planning," 2015. [Online]. Available: http://www.itu.int/rec/T-REC-G.107

[14] A. F. Wattimena, R. E. Kooij, J. M. van Vugt, and O. K. Ahmed, "Predicting the Perceived Quality of a First Person Shooter: The Quake IV G-model," in *Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games*, 2006, pp. 1–4.

[15] J. Saldana, "The effect of multiplexing delay on MMORPG TCP traffic flows," in *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*, 2014, pp. 245–250.

[16] D. Salopek, V. Vasic, M. Zec, M. Mikuc, M. Vasarevic, and V. Koncar, "A network testbed for commercial telecommunications product testing," in *22nd Int. Conference on Software, Telecom. and Comp. Networks (SoftCOM)*. IEEE, 2014, pp. 372–377.

[17] M. Suznjevic, I. Stupar, and M. Matijasevic, "A model and software architecture for mmorpg traffic generation based on player behavior," *Multimedia systems*, vol. 19, no. 3, pp. 231–253, 2013.

[18] S. Avallone, S. Guadagno, D. Emma, A. Pescapé, and G. Venturi, "D-ITG Distributed Internet Traffic Generator," in *Proceeding of International Conference on Quantitative Evaluation of Systems*, 2004, pp. 316–317.

[19] M. Suznjevic, O. Dobrijevic, and M. Matijasevic, "MMORPG player actions: Network performance, session patterns and latency requirements analysis," *Multimedia Tools and Applications*, vol. 45, no. 1-3, pp. 191–241, 2009.

[20] Recommendation, ITU-T, "P.800," *Methods for subjective determination of transmission quality*, pp. 800–899, 1996. [Online]. Available: https://www.itu.int/rec/T-REC-P.800-199608-I/en

[21] Recommendation, ITU-R, "Bt. 500-13," *Methodology for the subjective assessment of the quality of television pictures*, 2012. [Online]. Available: https://www.itu.int/rec/R-REC-BT.500