A New Online Tool for Gamer Network Performance Analysis

Nick McDonald, David Leader, Cheng-Kao Chiang and Youry Khmelevsky*
Computer Science, Okanagan College, Kelowna, BC Canada
Emails: nick.mcdonald.94@gmail.com, solorak@gmail.com and fan119053@gmail.com, ykhmelevsky@okanagan.bc.ca

Rob Bartlett and Alex Needham
WTFast, Kelowna, BC Canada
Emails: {rob, alex}@wtfast.com

*Also Affiliated with Computer Science, UBC (Okanagan Campus), Kelowna, BC Canada

***Abstract*—This paper discusses a new online tool for online gamers to share networking performance results and to produce aggregate networking performance reports generated from the data reported by all users. It is necessary to improve the quality of service and make online games faster. This will help improve latency sensitive game data collection. It will help to improve gaming performance through the global network of servers, optimizing the game connection from end to end.**

A Web API was developed to receive data in the form of a HTTP post from WTFast clients while they are playing their games. A data generator was also built to simulate thousands of concurrent users posting data to the web API. The web API stores the data in a Elasticsearch database. A web interface was built for WTFast clients to view their historical network data and to share it on forums or social media. The web interface shows a graph of their network performance with and without using the GPN®. The complete system is anticipated to need to handle approximately 400,000 concurrent users sending network data every 2 seconds.

I. Introduction

The current "WTFast's Gaming Private Network® (GPN®)" [1] online tool "gathers data about the internet connection, such as the latency, route and the amount of data sent/received" [2], but gamers still can't see the comparison of the regular internet connection vs of the improved connection by the GPN®. Moreover, it would be nice for gamers and for the WTFast system administrators to see the historical information of the connection to improve the provided services.

Users should be able "to view statistical information about their network connections with and without using the WTFast's GPN®. It includes the collection and analysis of network data (pings, latency) and visualization of the analyzed data to provide meaningful information to the user, and allow the user to share his/her network statistics on forums and other social media. The WTFast's GPN® is a virtual private network that is configured specifically for gamers to provide an optimized online gaming experience. The data is collected by the WTFast's client software and sent to a back-end database through a web API. The data is analyzed and presented to the client via a web interface" [2].

Our research contributions are:

a) A new online tool for the online gamers, clients of the WTFast to share networking performance results and to produce aggregate networking performance reports generated from the data reported by the gamers.
b) A new data generator for the online tool and for the Elasticsearch database testing.
c) A proof of a concept for the networking metadata collection of the online gamers.

II. Related Works

The "large-scale network simulation is an important technique for studying the dynamic behaviour of networks, network protocols, and an emerging class of distributed applications, including Peer-to-Peer and Grid applications" [3]. The original GPN® infrastructure prototype, which was built in 2014 for initial tests we discussed in [4]. GPN® makes online games faster by increasing game speed, reducing game disconnections, deviations and lag caused by spikes in packet traffic [2].

A "*video game network*, which is a distributed set of apparatuses which are capable of exhibiting an interactive single identity game. Response times and network latencies are very important video game parameters, which can be a reasons for gamers frustration and dissatisfaction, especially in the multi-user environment" [5].

"The online service's computers themselves introduce latencies, which are bigger with the larger number of active gamers" [6].

Zhou et al. discovered that "latencies and delays are important in First-Person Shooter (FPS) games and it can give advantages to a gamer with less delays. The time the information reaches the server that matters, not the time the player actually pushes the button" [7].

"The latency in Internet is stronger for the First-person perspective online video games" [8]. Faerber shows, that "the client's traffic has almost constant packet and data rate in different First-person games" [9].

Lee investigated "the StarCraft II game and found network problems related to delays, jitters and packet losses" [10].

Claypool et al. (2012) investigated traffic, generated by a thin client of OnLive games as well as traffic characteristics such as the bitrates, inter-packet times and packet sizes' [11].

Zander et al. found that "gamers with more delays due to either a longer distance to servers, connection problems,

978-1-5090-6096-2/16/$31.00 ©2016 IEEE

or congestions have disadvantages to compare with the other gamers. The authors used bots instead of gamers to evaluate eliminating delay differences between players" [12].

"A traffic generation tool OpenAirInterface Traffic Generator (OTG), which can be used for online gaming testing and evaluation, as well as for modelling and simulation" was described in [13].

III. System Design

At the start of our project we gathered requirements for our system, decided what technologies we were going to use, and what features our software would have. The architecture we decided on consists of a data generator built on Java, web API built on C# .NET MVC, and Elasticsearch as our backend database. The web API allows the data generator (or WTFast client) to post data to our back-end Elasticsearch database and also hosts a website for viewing and sharing summaries of their data.

Fig. 1 shows a layered view of the system. Users interact with the system via web browser for viewing their data. The browser interprets the JavaScript, HTML and CSS that comes from our C# .NET MVC middleware. We use a web framework named Bootstrap to display our webpage in a nice, responsive way, and the D3.js library for graphing the network data. Our middleware is written using C# .NET MVC, which runs on our Nginx web server. Because .NET applications are ment to run on Windows, to run them on CentOS 7 we had to run the application on Mono, which is a framework that allows .NET applications to run on Linux. We used FastCGI to host the application on top of Nginx. We used Elasticsearch as our database. Elasticsearch is a NoSQL database, we chose to use it because it is good at handling the large amount of data we need to store, and it scales well so we could add more database instances easily to increase redundancy and performance of the system.

A. Overview of the database data

Fig. 2 shows a Kibana dashboard that an administrator could use to assess the state of the GPN®. The graph at the top of Fig. 2 shows the average pings sent to the database over the last five minutes. The User count graph (bottom left) shows the number of users for the top 5 games. The Data count (bottom middle) shows the amount of entries for the top 5 games. The count metrics (bottom right) show the total number of entries in the database, the total number of unique user ids, the total number of unique games and the total unique sessions. The Pie chart illustrates the count of data for the top 5 games. These are just a few of many metrics that would be useful to WTFast.

B. The web API of the system

The web API is a web application built on C# .NET MVC. It has 3 main functions, hosting a website where end users can view their sessions, and providing interfaces for posting data to the database, and retrieving data from the database, and handling the upload and retrieval of images generated by the web interface. The application works as a piece of middleware so that the user never has direct contact with the database, which would be a security risk.

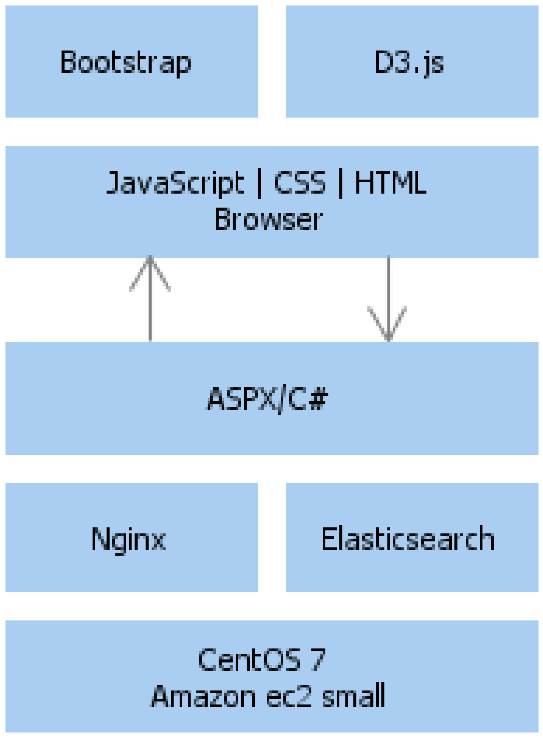


Fig. 1. System Architecture

IV. Development

We developed a data generator for testing our system. The generator is small Java application that generates data to test the web API and to populate our database. We made the data generator to simulate many users concurrently using the system. We used this generator for load testing the system to detect performance issues and insure that the system is capable of handling the loads that it would see in the real environment.

The Generator class diagram can be seen in Fig. 6. The GUI class provides a user interface that the user can use to configure and run the generator. When the generator is run the controller class creates a list of sender Threads. The sender Threads use the generator class to generate data to be sent to the web API via an HTTP request.

A. User's Interface

A prototype of the web interface can be seen in Fig. 3 and Fig. 4. The interface is very simple. When the use logs in to his/her account the user sees 2 dropdown menus. The first dropdown menu is a list of the games that the user has played. The second is a list of all of the sessions of the selected game. The session id that the user will see will be the start time of the session, though that has not been implemented yet. The most recent session is displayed by default when the user logs in. The graph of the session shows the ping time throughout the whole session for both the Internet (Red) and using the GPN® (Yellow). This shows the decrease in ping time as well as the frequency of ping spikes when using the GPN®. Below the graph there is some aggregate information that describes the session including highest and lowest ping, and the number of

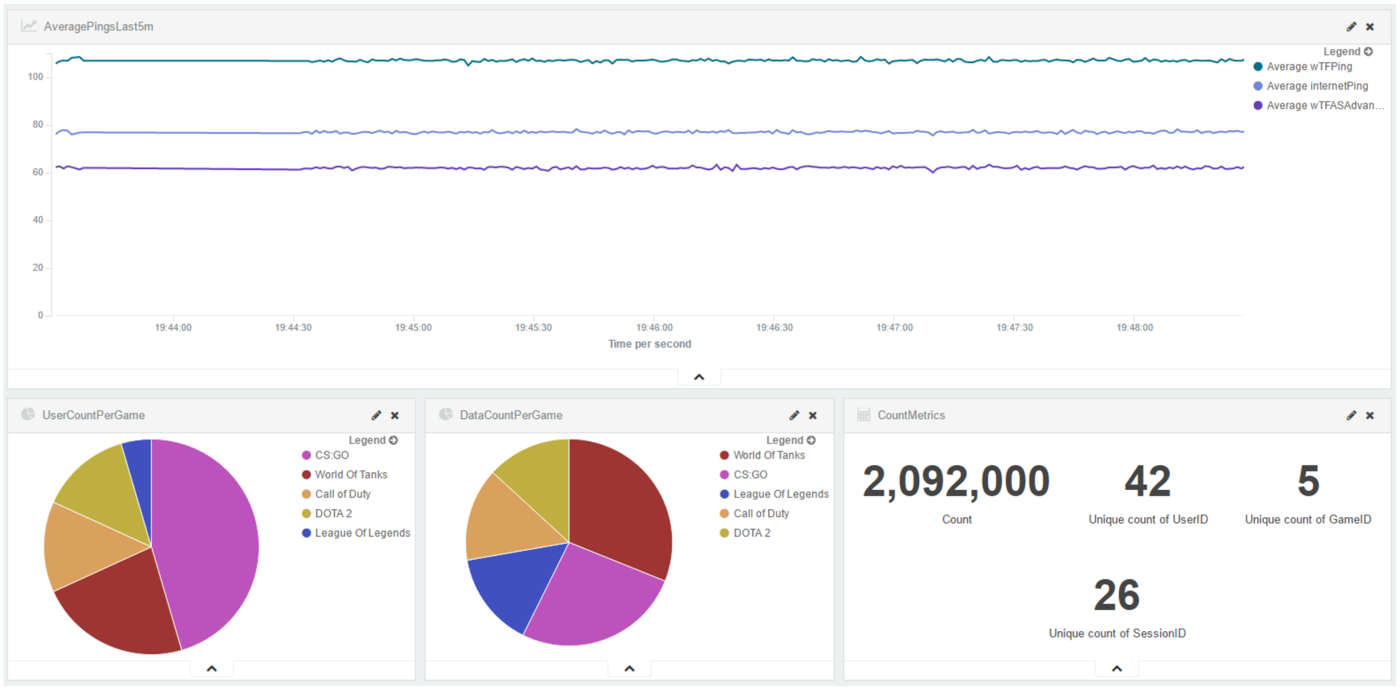
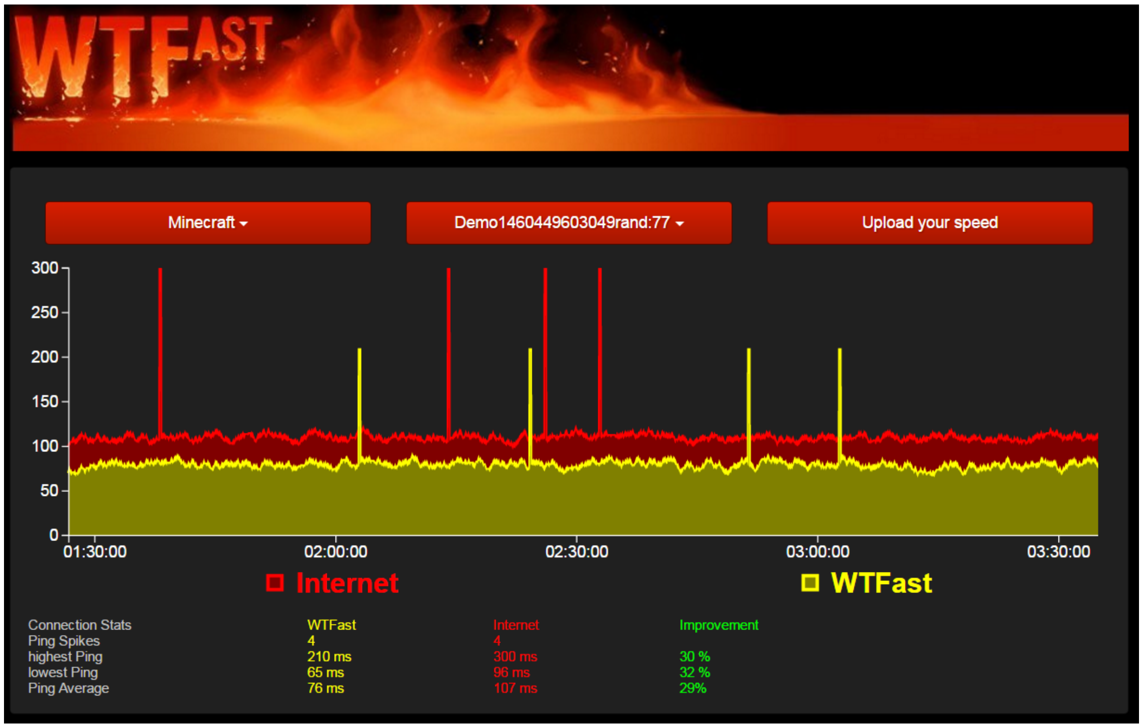


Fig. 2. The average of all pings sent to the database within the last five minutes



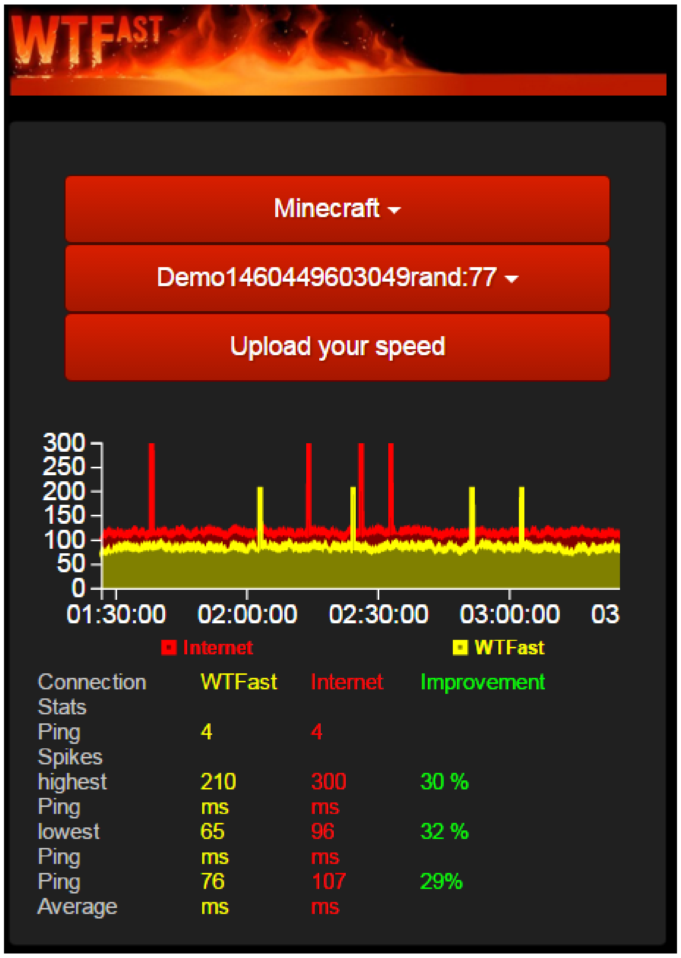
Connection Stats	WTFast	Internet	Improvement
Ping Spikes	4	4	
highest Ping	210 ms	300 ms	30 %
lowest Ping	65 ms	96 ms	32 %
Ping Average	76 ms	107 ms	29%

Fig. 3. Web Client Interface

ping spikes as well as a percentage improvement when using the GPN®.

The last feature of the web interface is the ”Upload your Speed” button. Clicking this button will generate an image similar to what is shown in Fig. 5 and will upload the image to the server. In a pop up window users will be shown the image, and be given the URL of the image so that they can embed it in forums or share it on social media.

Using the Bootstrap framework allowed us to make a responsive design that would work on many different screen resolutions including mobile phones as seen in Fig. 4.



Connection Stats	WTFast	Internet	Improvement
Ping Spikes	4	4	
highest Ping	210 ms	300 ms	30 %
lowest Ping	65 ms	96 ms	32 %
Ping Average	76 ms	107 ms	29%

Fig. 4. Web Client Interface (Mobile Version)

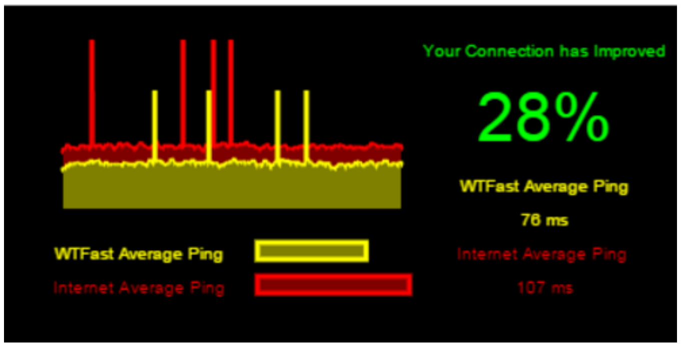


Fig. 5. A badge with summary of the connection data

V. Security

There are some security concerns when handling user data. The end users of the system should only have access to their own data, all other data should be secured and not be vulnerable to common attacks such as SQL of JavaScript injection. Due to the scope of our project we did not implement an authentication system. Because this system was designed to be integrated into an existing system at a later date, an authentication system would complicate that process. In the complete system authentication would ensure that an authenticated user could only view his/her data.

A non-functional requirement for the system was that all network communication should be secured using Secure Socket Layer (SSL). This includes both the communication between the Web Client and the web API, as well as between the web API and the database. We secured the database by using a reverse proxy through our Nginx web server [14] and enabling SSL and basic authentication.

VI. Future Work

This project has been passed on to a research team at Okanagan College for further research and development. Our project will be used as a prototype and will be rewritten using Python and Go programming languages. After being rewritten the research team will integrate it with WTFast's rich client to start gathering data from real users. They will analyze this data and look for patterns that could help improve the quality of service of the GPN®.

The future research will include the online tool prototype migration into production environment, security testing, and performance optimization. The collected data will be used for better service to the end users.

Another future research topic is related to the collected metadata analyzes and creating infographics to assess the vulnerabilities and network performance degradation during time, to predict traffic spikes or other performance degradation artefacts.

VII. Conclusion

We have created a system for collecting network data from thousands of clients all around the world. This system consists of a web API, an Elasticsearch database, a web interface for viewing data, and a data generator for testing the system. The data collected by this system will be used by WTFast's clients to view the network performance boost that WTFast gives them, as well as by developers at WTFast to help them improve their services.

In this research paper we discussed a new online tool for gamer network performance analysis. The developed online tool allows monitoring of client network performance metadata. A data generator was developed for the online tool and Elasticsearch database testing. This allows us to simulate a large data transmissions to the Elasticsearch database.

Acknowledgments

We would like thank Computer Science's students Jon Ohlhauser and Corey Frank, who participated in the initial project prototype development within COSC 470 — "Software Engineering" and within COSC 471 — "Software Engineering Project" courses in 2014–2015.

The project was related to Engage Grants for colleges "GPN-Perf: Investigating performance of game private networks" (number 465659-14), which was supported by NSERC in 2014 and to NSERC's "College and Community Innovation Program — Applied Research and Development Grant" Level 2 "GPN-Perf2: Game private networks and game servers performance optimization" (477506-14), which was started in January 2016.

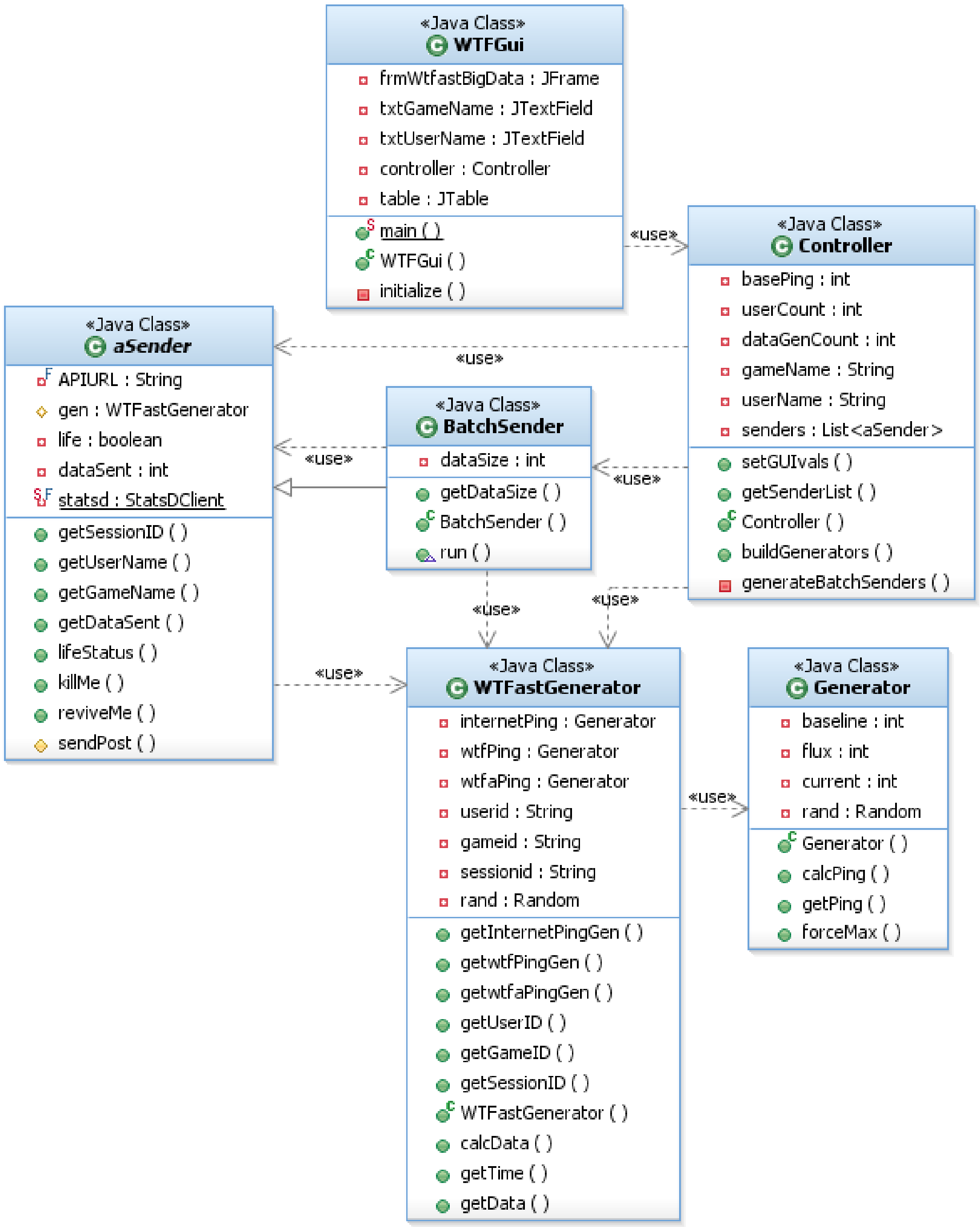


Fig. 6. Generator Class Diagram

We would like to thank to Amazon Web Services, Inc, Atlassian, IBM and all other companies, who supported our educational and research projects.

References

[1] A. I. P. I. WTFast., "http://www.wtfast.com."

[2] N. McDonald, C. Frank, Y. Khmelevsky, R. Bartlett, and A. Needham, "GPN game user performance data gathering and analysis by a custom-built tool," in *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 2016*, Vancouver, BC, Canada, May 15–18 2016, pp. 1099–1103.

[3] X. Liu and A. A. Chien, "Traffic-based load balance for scalable network emulation," in *Proceedings of the 2003 ACM/IEEE Conference on Supercomputing*, ser. SC '03. New York, NY, USA: ACM, 2003, pp. 40–. [Online]. Available: http://doi.acm.org.ezproxy.okanagan.bc.ca/10.1145/1048935.1050190

[4] T. Alstad, J. R. Dunkin, R. Bartlett, A. Needham, G. Hains, and Y. Khmelevsky, "Minecraft computer game simulation and network performance analysis," in *Second International Conferences on Computer Graphics, Visualization, Computer Vision, and Game Technology (VisioGame 2014)*, Bandung, Indonesia, November 2014, accepted for publication.

[5] D. H. Sitrick, “Video game network. United States Patent number 4,572,509,” Feb. 25, 1986.

[6] S. G. Perlman, “Network architecture to support multiple site real-time video games. United States Patent number 5,586,257,” Dec. 17, 1996.

[7] Q. Zhou, C. Miller, and V. Bassilious, “First person shooter multiplayer game traffic analysis,” in *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*, May 2008, pp. 195–200.

[8] 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

[9] J. Färber, “Traffic modelling for fast action network games,” *Multimedia Tools and Applications*, vol. 23, no. 1, pp. 31–46, 2004.

[10] C.-S. Lee, “The revolution of starcraft network traffic,” in *Proceedings of the 11th Annual Workshop on Network and Systems Support for Games*, ser. NetGames ’12. Piscataway, NJ, USA: IEEE Press, 2012, pp. 18:1–18:2. [Online]. Available: http://dl.acm.org.ezproxy.okanagan.bc.ca/citation.cfm?id=2501560.2501583

[11] M. Claypool, D. Finkel, A. Grant, and M. Solano, “Thin to win?: Network performance analysis of the onlive thin client game system,” in *Proceedings of the 11th Annual Workshop on Network and Systems Support for Games*, ser. NetGames ’12. Piscataway, NJ, USA: IEEE Press, 2012, pp. 1:1–1:6. [Online]. Available: http://dl.acm.org.ezproxy.okanagan.bc.ca/citation.cfm?id=2501560.2501562

[12] S. Zander, I. Leeder, and G. Armitage, “Achieving fairness in multiplayer network games through automated latency balancing,” in *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, ser. ACE ’05. New York, NY, USA: ACM, 2005, pp. 117–124. [Online]. Available: http://doi.acm.org.ezproxy.okanagan.bc.ca/10.1145/1178477.1178493

[13] A. Hafsaoui, N. Nikaein, and C. Bonnet, “Analysis and experimentation with a realistic traffic generation tool for emerging application scenarios,” in *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, ser. SimuTools ’13. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013, pp. 268–273. [Online]. Available: http://dl.acm.org.ezproxy.okanagan.bc.ca/citation.cfm?id=2512734.2512771

[14] “NGINX Plus: Complete application delivery,” 2016. [Online]. Available: https://www.nginx.com/products/