Push Notification System to Mobile Game Player Using Distributed Event-Based System Approach

Fiona Yunisa
Master in Computer Science, Binus Graduate Program, Bina Nusantara University, Jakarta, Indonesia
Jl KH Syahdan 9, Jakarta 11480, Indonesia,
fiona.yunisa@gmail.com

Suharijto
Master in Computer Science, Binus Graduate Program, Bina Nusantara University, Jakarta, Indonesia
Jl. Kebon Jeruk 27, Jakarta 11480, Indonesia
suharjito@binus.edu

***Abstract*— Push notification is a mechanism that allows individuals to receive updated information quickly. Push notification delivery needs to be effectived and has to be paid attention to time zone differences between countries in order to generate time-appropriate notifications. This push notification system was developed using distributed event-based system approach. The research methodology consists of four stages: initial research, data collection and processing, system development, and evaluation. The result of this research is a scalable web-based push notification system supported by the usage of scheduler and consumer that can help the company send notifications about game updates easier and quicker to millions of mobile devices.**

***Keywords*— Push Notification; Game player; Distributed event-based system**

I. Introduction

The existence of mobile devices nowadays is highly related to the increasing number of smartphone users and the development of the smartphone operating system technology. Based on statistical data, the number of worldwide smartphone users has increased by 25% in 2013-2014 [1]. The existence and usage growth of mobile devices have changed users' behavior from desktop-based application to mobile-based application, thus accelerating the need of mobile-based application development, such as mobile game applications. Statistical data between January and March 2014 indicate that while there are 86% amount of time used to access Android and iOS mobile device, 32% of that amount is used to access mobile game applications [2].

Referring to the facts related to the development of mobile game applications nowadays, the company has problem of how to make players do not miss the latest information about the games [3]. Thus, it is common to use push notification mechanism to alert the players with new game updates. Push notification tells the players that there is news or an update for the installed mobile application [4]. Statistic shows that the more often a developer sends push notifications to its users, the more frequent the mobile device users will open the mobile application [5].

XYZ PTE. LTD. is a company which focuses on the development of mobile game applications under Android and iOS operating systems. To deliver new information to application/game players, XYZ always uses push notification. In the existing application, XYZ still uses semi-manual push notification by implementing script for query push token. Then, notifications are sent sequentially one by one. XYZ has not implemented scheduler. This notification delivery process is still inefficient. Thus, there is a chance for improvement and automation of the process. In addition, the company cannot monitor the status of sent notifications, cannot do data segmentation, and the employees must be ready 24/7 to send push notifications across different parts of the world which have different time zones. Push notification delivery needs to be effective and has to pay attention to time zone differences between countries in order to generate time-appropriate notifications. Fig. 1 shows the distribution map of the targeted players that will reveice the notifications.

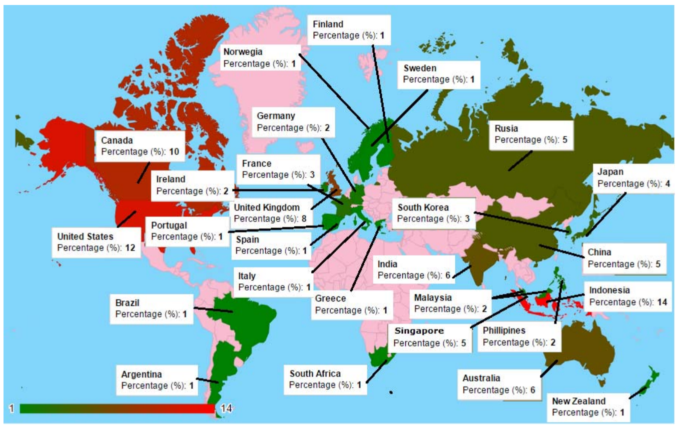


Fig. 1. Distribution Map of the Targeted Players

The limitations are the underlying reasons for developing a new method to help the company to find a solution for improving efficiency of notification delivery. In this research, a push notification system using distributed event-based system approach to facilitate the company in managing, optimizing, and monitoring the notification delivery process to millions of mobile devices is developed. The distributed event-based system contains components that interact by exchanging messages. This system allows adaptation and flexible system composition [6]. In addition, the communications also provide a scalable system composition [7]. The system composition that can be adaptable, flexible, and scalable is needed when developing the push notification system where the system can be adjusted as needed in order to make the notification delivery process become faster and more efficient.

II. Related Works

At the early development of mobile devices, data transfer through Bluetooth among handheld devices are common and popular. Bluetooth technology is implemented and used for advertising Hypermarket [8]. Initially, hypermarket spread ads manually, by distributing flyers in various places. This method makes hypermarkets cannot convey the latest information in real time and need additional fee for printing the new advertising. With this weakness, hypermarkets build proactive advertising system that is capable of performing push advertising via Bluetooth to mobile phone customers in hypermarkets. By using the system, hypermarkets just need to keep the advertising on the server that is around the location, and push the advertisements to mobile phone customers.

In 2011, Lee designed a framework integrating push gateway to provide a push message store and forward functionality for mobile application. The integration gateway makes server side developers do not have to implement the gateway functionality and, therefore, can send notifications to many mobile operating systems [9].

According to Ji, Ganchev, O'Droma, and Zhao, real-time notification is a necessary part of a mobile application. XMPP (Extensible Messaging and Presence Protocol) is a protocol for sending messages in real-time between the client and the server. The design and implementation of push notification service use a client/ server design pattern with a distributed architecture. The client collects user behaviors, which are sent periodically to the cloud, and notifications will be pushed to users via XMPP [10].

Gusev, Ristov, Velkoski, Guseva, and Gushev made an alert notification system which is a scalable cloud solution for the implementation of the push service that aims to warn internet users when there are changes of texts or certain keywords on web sites [11].

Oliveira, Rodrigues, Elias, and Zarpelão proposed a push notification system solution in wireless sensor monitoring networks that is built to ensure scalability and reusability. Wireless sensor networks monitoring is built using REST interfaces and XML / JSON message that support the majority of mobile devices [12].

According to Ravindran, Qazi, Atre, Rohira, and Narkar, the distribution of mobile devices is so fast in campuses so that colleges need to provide a system of location and time based information to improve, manage, and send information automatically in the campuses. Campus Push using context aware architecture is designed to replace the old method that can help to provide information related to attendance, information status about library books, laboratory agenda automatic download and timetable via push notification based on location and time [13]. Several studies on the push method that have been done are summarized in Table I.

TABLE I. Push Notification Related Research

Researcher	Method	Device
[8]	Bluetooth Wireless Communication	Mobile
[9]	Integrated multimedia push framework	Mobile
[10]	Client/Server design pattern and XMPP protocol	Mobile
[11]	Scalable and elastic cloud architecture	Web and mobile
[12]	REST interfaces and XML/JSON messages.	Mobile
[13]	Context aware architecture	Mobile

1. Push Notification
   Push notification is defined as a short message sent to application or end user's smartphone. Push notification can show an alert or produce a sound to notify the end user about the latest updates. With the mobile technology advancement nowadays, it is easy to provide real time information delivery [4].
2. Distributed Event-Based System
   Distributed Event-Based System (DEBS) is a system which consists of separate components communicating by exchanging messages. This form of communication provides a flexible and scalable system composition [7]. Distributed event-based system components can be classified into 2 types: producers and consumers. The consumers' component can subscribe to an appealing message for the receiver. When the producer publishes the message, it will be filtered to help to reduce the bandwidth load by ensuring the message sent only to the intended customers [14].
   - 2.1 Publish/subscribe pattern
     Settle stated that the publish/subscribe pattern is commonly used in the push notification delivery process [15]. This model provides many advantages because it can replace system monitoring process, save time, increase customer number, and get information needed by the company. Publish/subscribe pattern consists of 2 main components: publisher and subscriber. If there is a new update from the publisher, the publisher will publish it to the right subscribers.
   - 2.2 Producer Consumer Problem
     In producer consumer problem, there are 2 involving processes: producer and consumer. The two are sharing a buffer. The producer's role is to place a certain item in the buffer in certain time, and the consumer's role is to get the item from the buffer [16].
   - 2.3 Advanced Message Queuing Protocol (AMQP)
     Advanced Message Queuing Protocol (AMQP) is one of standard protocols from Message Oriented Middleware (MOM) [17]. MOM provides a reliable

asynchronous communication with message queuing techniques. Advanced Message Queuing Protocol (AMQP) consists of 3 main components: publisher(s), consumer(s), and broker/server(s). the publishers and consumers communicate each other via message queue that will be exchanged in brokers. AMQP provides reliability in message transmission.

2.4 Horizontal Scaling

System communication pattern in distributed event-based system makes the flexible and scalable system composition more likely [7]. Horizontal scaling is one of many ways to achieve scalability. Horizontal scaling is done by adding some nodes to the system, for instance adding new computers to make the workload of another computer lighter and the performance of a process become faster [18].

2.5 Exponential Back off Algorithm

By using exponential back off, a node will try to send an information package repeatedly. If there is a collision along the sending process, random value of delay in sending process will be doubled. Generally, after i collision occurs, there will be random number between 0 until 2^i-1 to be chosen as a resend waiting time for that package [19]. After a maximum number of package transmission is reached (such as 15 times), that package will be deleted and node reports a transmission problem[20].

2.6 Round Robin Scheduling

Round robin scheduling has a simple basic idea that is shown in Fig. 4. Round robin scheduling is not doing a single job continuously, but it does the job based on time slice (scheduling quantum) and changes to the next job in running queue. This is done repeatedly until the job is finished [21].

3. Apple Push Notification System (APNS)

APNS (Apple Push Notification Service) is a push notification service framework designed by Apple for devices using iOS platform.

4. Google Cloud Messaging (GCM)

Google Cloud Messaging (GCM) for Android is a free service that can help developer in data transmission from server to Android application devices that use Android platform and send messages from users' devices to cloud. GCM become a message sender to Android applications when there are new data in the server. This GCM service is responsible for all the message queue aspects and delivery to certain android applications that run in Android devices. GCM is free. So, it does not matter how big the message is sent, and there is no quota [22].

III. Research Method

There are 4 stages of this research: initial research, data collection and processing, system development, and conclusion and suggestions. The research begins with initial research steps until system implementation to do the performance evaluation. The performance evaluation of push notification system development will be done in 3 ways:

1. Setting and sending notification according to specified time zone by using system.
2. Comparing research variable value: throughput and average send time. In addition, the process of notification delivery will use XYZ system and system using DEBS approach. Systems using DEBS approach will be tested using 2 consumers, 5 consumers, and 10 consumers. Each system will be tested 30 times. System testing results will be compared to see the performance of each system.
3. Monitoring notification delivery status

IV. Result and Discussion

Fig. 2 shows an architecture design created to support the development of push notification system.

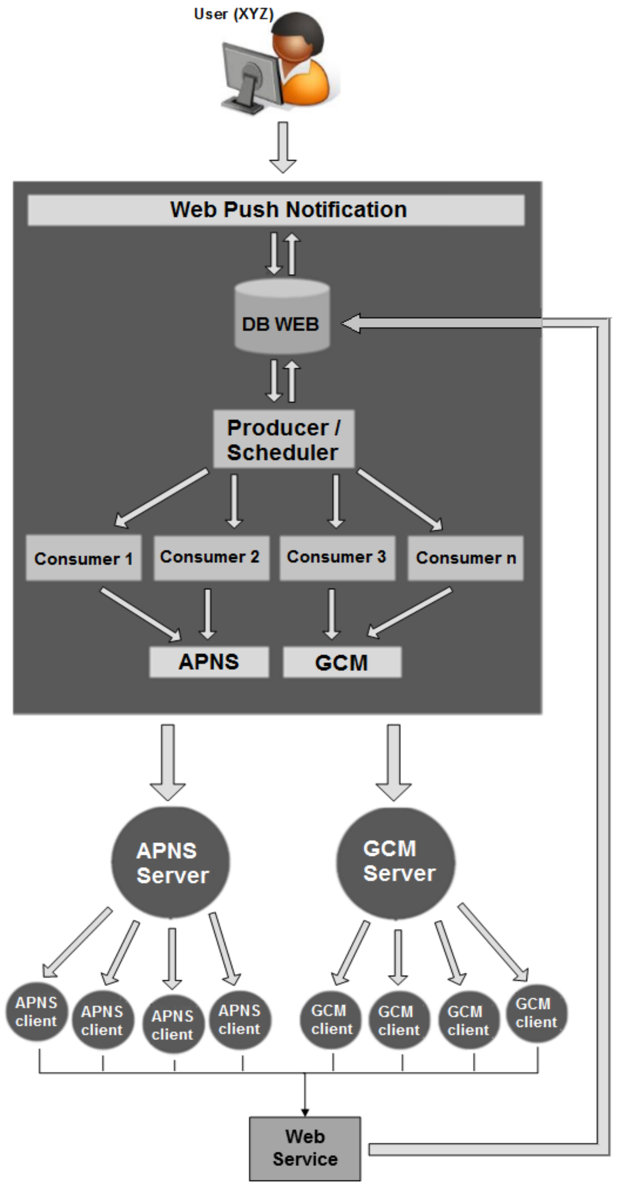


Fig. 2. An architecture of push notification system

Push notification system architecture consists of certain Components, including:

1. Web Push Notification
Web push notification is an interface that will be accessed by users for setting the game data, time zone, country, push client, notification, game player, and sending notifications to the game player.
2. Database
A web push notification is also supported by the use of the database that serves to store data, such as the game data, time zone, country, push client, notification, game player, and notification delivery data to the game player.
3. ***Producer***/scheduler
Producer/scheduler is used to access the database to find data notifications that will be pushed to the determined game players. In this case, the producer/scheduler will take the data to the database every five minutes to check if there are notification data that need to be sent. Data notification (messages) will in the ***queue*** and distributed to active consumers. At the producer/scheduler, the operation ***publish*** will be applied, where producer/ scheduler will only push notifications to consumers that subscribe to a specific channel.
4. ***Consumer***
Consumer is used to send notification data to the APNS (for iOS users) and GCM (for Android users). Consumer refers to a subscriber that will ***subscribe*** to a channel. On the consumer, the ***round robin*** will be applied method, where the consumer will be used interchangeably so that the load performance of the consumer becomes lighter when sending notification data to APNS or GCM. If the consumer fails to connect to APNS or GCM, it will run the ***exponential back off algorithm*** for resending the notification. The maximum limit for resending the notification is 15 times. For handling mores player effectively, we can add consumer at any time (***horizontal scaling***), so that notification delivery process becomes faster.
5. APNS and GCM
APNS is used to send the notification data to the APNS server and GCM is used to send the notification data to the GCM server.
6. APNS server and GCM server
GCM server and APNS server are used to send notifications to mobile game players with Android and iOS platform that have been determined.
7. Web service
When the player opens the notification, it will send data to the web service for storing hours of when the player opens the notification. It is intended that the company can identify the open rate and the active hours of players when opening the notification.

Notification delivery setting on web push notification system consists of:

1. Check active push client
Push client list as shown in Table II is used to check push clients with active status. Push clients are consumers that will send notifications to one specific target (APNS or GCM).

TABLE II. PUSH CLIENT LIST PAGE

Push Client Name	Push Client Username	Push Client Status	Busy Status
SubsSatu	SubsSatu	Active	Not Busy
SubsDua	SubsDua	Active	Not Busy
SubsTiga	SubsTiga	Not Active	Not Busy

2. Inserting target players and setting notification send time
Users can input target players by filtering player's time zone, player's country, and also setting notification delivery by setting the content of notification, date, hour, and time zone. For example, as shown in Fig. 3, notification will be sent to Indonesia's game players on February 19, 2016 07:20 AM (GMT+07) to notify that the game has new avatars. The notification data can be seen in the send notification list page shown in Table III.

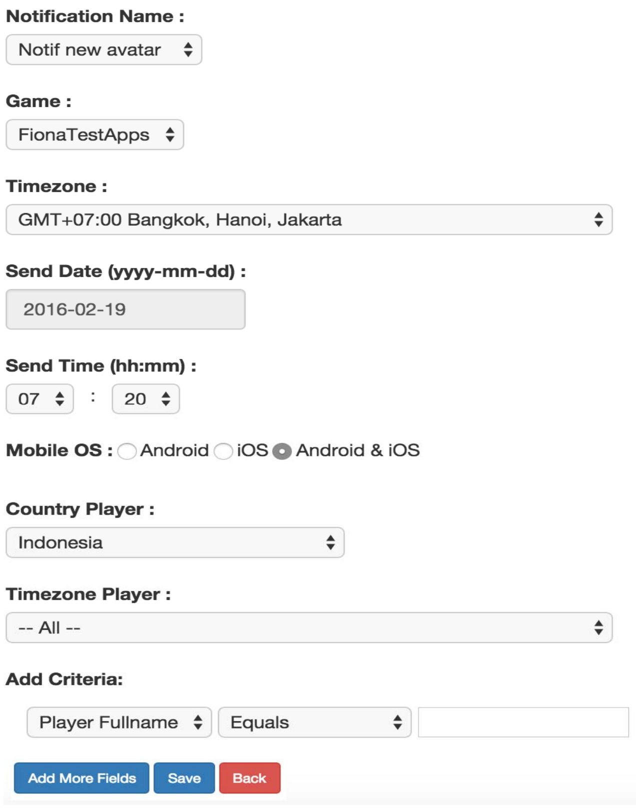
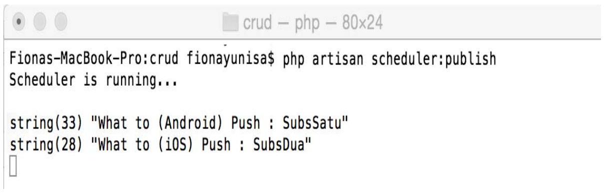

Notification Name : Notif new avatar
Game : FionaTestApps
Timezone : GMT+07:00 Bangkok, Hanoi, Jakarta
Send Date (yyyy-mm-dd) : 2016-02-19
Send Time (hh:mm) : 07 : 20
Mobile OS : Android iOS [✓ Android & iOS]
Country Player : Indonesia
Timezone Player : -- All --
Add Criteria: Player Fullname | Equals |

Fig. 3. Insert Target Player Page

TABLE III. SEND NOTIFICATION LIST PAGE

Notification Name	Timezone	Send DateTime	Status	Detail Player	Chart
Notif new avatar	GMT+07:00	2016-02-19 07:20	Not Send Yet	Player	View
Notif twitter	GMT+07:00	2016-02-14 21:00	Sent	Player	View

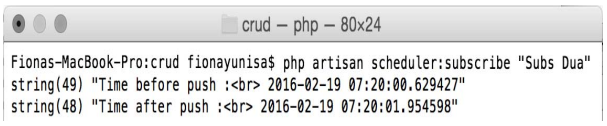
3. Producer/scheduler and consumer must be active
To send notification, producer/scheduler (Fig. 4) must be active for query to database to get players' data who will receive the notification. Producer will query to database every 5 minutes and "*publish*" the data notification to specific consumer (such as "Subs Dua", etc). Besides producer, consumer must also be active for sending notifications to one specific target (only APNS or GCM). In the consumer's display as shown in Fig. 5, the consumer does "*subscribe*" operation and the user can see the total send time of the notification delivery process.



```
Fionas-MacBook-Pro:crud fionayunisa$ php artisan scheduler:publish
Scheduler is running...

string(33) "What to (Android) Push : SubsSatu"
string(28) "What to (iOS) Push : SubsDua"
```

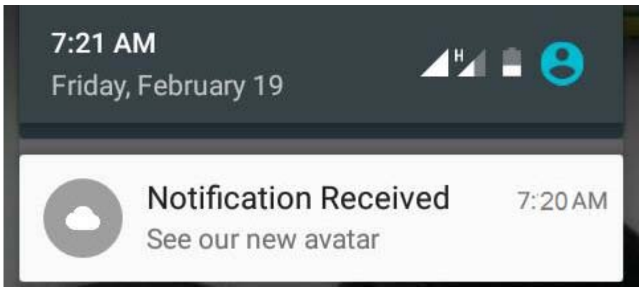
Fig. 4. Producer/Scheduler Display



```
Fionas-MacBook-Pro:crud fionayunisa$ php artisan scheduler:subscribe "Subs Dua"
string(49) "Time before push :<br> 2016-02-19 07:20:00.629427"
string(48) "Time after push :<br> 2016-02-19 07:20:01.954598"
```

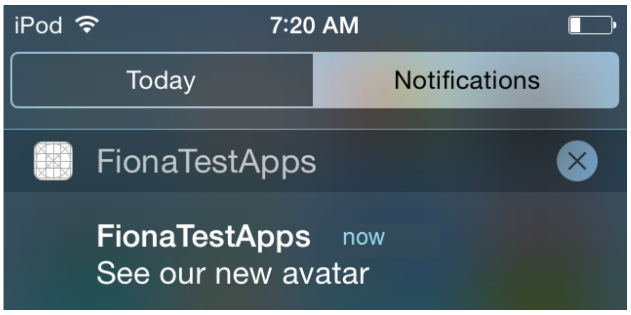
Fig. 5. Consumer Display

4. Check sent notification on mobile
Notification will be delivered according to the setting that has been made. The following figures show the results of sending notifications on Android (Fig. 6) and iOS (Fig. 7) mobile devices. After the notification has been sent successfully, the user can see which push client sent a notification to each player by clicking "Player" link in the send notification page as shown in Table IV.



7:21 AM
Friday, February 19
Notification Received 7:20 AM
See our new avatar

Fig. 6. Notification on Android Platform



iPod 7:20 AM
Today Notifications
FionaTestApps
FionaTestApps now
See our new avatar

Fig. 7. Notification on iOS Platform

TABLE IV. SEND NOTIFICATION – PLAYER LINK PAGE

Game Name	Player Full Name	Country	Push Client Name	Status Push
FionaTestApps	Fiona	Indonesia	SubsDua	Sent
FionaTestApps	Teddy	Indonesia	SubsSatu	Sent
FionaTestApps	Rifan W	Indonesia	SubsSatu	Sent

5. If players open the notification, users can monitor active hours of players and open rate of sent notifications by clicking "Chart" link in the send notification page.

System evaluation using XYZ system and DEBS (2 consumers, 5 consumers, and 10 consumers) aims to compare a throughput value and average send time from each system. The evaluation result using XYZ system and DEBS with 2, 5, 10 consumers based on throughput is shown in Fig. 8. Further, the evaluation result using XYZ system and DEBS with 2, 5, 10 consumers based on average send time is shown in Fig. 9.

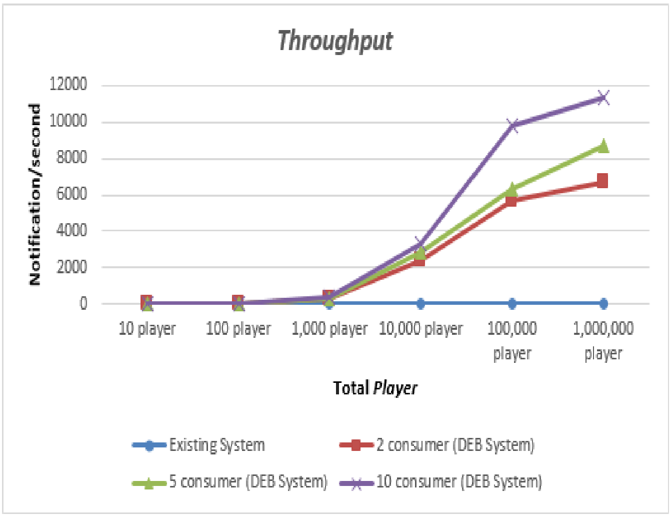


Fig. 8. Throughput using XYZ system and DEBS with 2, 5, 10 consumer

Based on Fig. 8, throughput using XYZ formed a straight line which means it has almost the same value, which is close to 2 notifications/second. When using DEBS with 2 consumers, the number of throughput increased to 6698.889 notifications/second. By using DEBS with 5 consumers, the number of throughput increased to 8725.237 notifications/second. Finally, by using DEBS with 10 consumers, the number of throughput increased to 11343.6 notifications/second. From these results, it can be seen that the addition of consumers at any time can produce higher throughput.

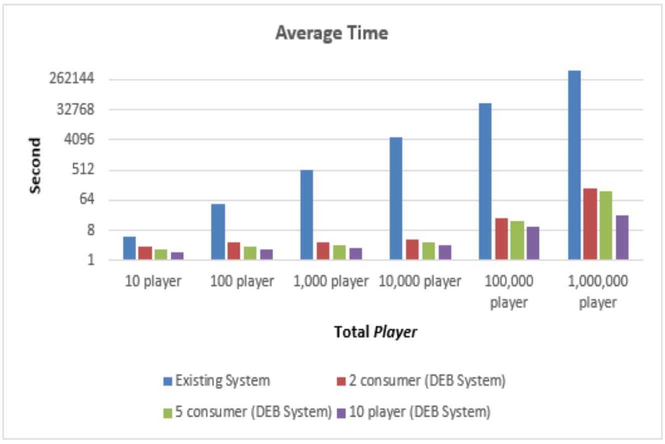


Fig. 9. Comparison of average send time for each number of players

Based on the evaluation results in Fig. 9, it can be seen that by using push notification system with distributed event-based system approach, the time required to send notifications becomes less. It can be seen from notification delivery time using XYZ system to 1 million players that reaches 5 day delivery time to less than 30 seconds by using the push notification system being developed. In addition, by adding the number of consumers gradually, it can be seen that the required notification delivery time will also become smaller so that the efficiency of the system is increasing.

V. Conclusion

Based on the discussion, the following conclusions can be proposed:

1. The notification system for mobile game players using distributed event-based system approach has been developed and implemented to help company to manage, monitor the process of notification delivery, and make the process become more efficient.
2. Push notification system using distributed event-based system approach makes notification delivery process become more efficient than XYZ system.

References

[1] eMarketer. (2014, December 11). 2 Billion Consumers Worldwide to Get Smart(phones) by 2016. Available: http://www.emarketer.com/Article/2-Billion-Consumers-Worldwide-Smartphones-by-2016/1011694

[2] Khalaf, S. (2014, April 1). Apps Solidify Leadership Six Years into the Mobile Revolution. Available: http://flurrymobile.tumblr.com/post/115191864580/apps-solidify-leadership-six-years-into-the-mobile

[3] A. Priyadarshi, R. Karamchandani, & A. Gundroo, Centralized Access of User Data Channel with Push, *International Journal of Innovative Science, Engineering & Technology (IJISET),* Vol. 2, n. 3, pp. 930-934, March, 2015.

[4] C. D. Wadate, P. T. Suvare, A. S. More, & R. Bora, A Survey of Automatic Wi-Fi based Push Notification in College Campus using Cloud, *IJCA Proceedings on International Conference on Advances in Science and Technology,* pp. 1-4, IJCA Journal, 2014.

[5] Gault, C. (2012, November 8). The Push Messaging Opportunity: Never Too Little, Too Late. Available: http://urbanairship.com/blog/2012/11/08/the-push-messaging-opportunity-never-too-little-too-late

[6] G. Mühl, L. Fiege, & P. Pietzuch, Distributed event-based systems (Springer Science & Business Media, 2006).

[7] Lee, Y. K., Bang, J. Y., Garcia, J., & Medvidovic, N, ViVA: A Visualization and Analysis Tool for Distributed Event-Based Systems, *Proceedings of the 36th International Conference on Software Engineering (*Page: 580-583 Year of Publication: 2014).

[8] Chen, Y.-L., Chou, H.-J., Lin, C.-P., Lin, H.-T., & Yuan, S.-M., A System Implementation of Pushing Advertisement to Handheld Devices via Bluetooth, *In Networked Computing and Advanced Information Management, NCM'08, Fourth International Conference.* 1, pp. 133-136. IEEE, September, 2008.

[9] D. Lee, Designing the Multimedia Push Framework for Mobile Applications, *International Journal of Advanced Science and Technology,* Vol.32, pp. 117-124, July, 2011.

[10] Ji, Z., Ganchev, I., O'Droma, M., & Zhao, Q, A Push-Notification Service for Use in the UCWW, *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC),* pp. 318-322, Shanghai: IEEE, 2014.

[11] Gusev, M., Ristov, S., Velkoski, G., Guseva, A., & Gushev, P, Scalable Architecture of Alert Notification as a Service, *International Conference on Information Society (i-Society),* pp. 80-85, IEEE, 2014.

[12] L. M. Oliveira, J. J. Rodrigues, A. G. Elias, & B. B. Zarpelão, Ubiquitous monitoring solution for Wireless Sensor Networks with push notifications and end-to-end connectivity, *Mobile information systems*, Vol. 10, n. 1, pp. 19-35, 2014.

[13] R. Ravindran, N. Qazi, V. Atre, J. Rohira, & S. Narkar, Campus Push, Location, Context, Policy Driven Push Notification Application for Mobile Devices, *International Journal of Engineering and Technical Research,* Vol. 3, n. 1, pp. 143-147, January, 2015.

[14] Popescu, D., Garcia, J., Bierhoff, K., & Medvidovic, N, Impact Analysis for Distributed Event-Based Systems, *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, pp. 241-251, ACM, 2012.

[15] B. Stelte, A Real-Time-Enabled, Blackboard-Based, Publish/Subscribe Architecture for Wireless Sensor Nodes, *Wireless Sensor Network*, Vol. 2, n. 8, pp. 612-628, 2010.

[16] S. N. Mehmood, N. Haron, V. Akhtar, & Y. Javed, Implementation and Experimentation of Producer-Consumer Synchronization Problem, International Journal of Computer Applications, Vol. 14, n. 3, pp. 32-37, 2011.

[17] Rheddane, A. E., Palma, N. D., & Tchana, A, Scalable Store and Forward Messaging, The Fourth International Conference on Cloud Computing, *GRIDs, and Virtualization,* pp. 156-161, IARIA, 2013.

[18] N. A. Baig, V. K. Dindorkar, & P. D. Patil, A Survey on Scalability in Cloud Computing, *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 3, n. 4, pp. 545-548, March, 2014.

[19] M. Dashora & M. K. Porwal, Random Access Career Sense Protocol: A Comparative Analysis of Non Persistent and 1 Persistent CSMA, *International Journal on Recent and Innovation Trends in Computing and Communication*, Vol. 2, n. 12, pp. 4021-4025, December, 2014.

[20] I. S. Ahmad, A. Kalakech, & S. Kadry, Modified Binary Exponential Backoff Algorithm to Minimize Mobiles Communication Time, *International Journal of Information Technology and Computer Science (IJITCS),* Vol. 6, n. 3, pp. 20-29, 2014.

[21] R. H. Arpaci-Dusseau & A. C. *Arpaci-Dusseau, Scheduling: Introduction, Operating Systems*: Three Easy Pieces (Arpaci-Dusseau Books, 2014, 1-12).

[22] S. Muthukumarasamy, S., & Dr. Tamilarasi A., Design and Development of Mobile Assisted Language Learning (MALL) application for English Language using Android Push Notification Services, International *Journal of Research in Computer and Communication Technology,* Vol. 2, n. 6, pp. 329-338, June, 2013.