IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 1 (Jul. - Aug. 2013), PP 17-21
www.iosrjournals.org

Scalability Enhancement of Push/Pull Server functions by converting Stateless process using AJAX Mechanism

J.Saravanesh[1], Dr.E.Ramaraj[2]
[1] *Assistant Professor, Department Of Computer Science, MKU College, Madurai, Tamil Nadu, India*
[2]*Professor, Department of Computer Science and Engg, Alagappa University, Karaikudi Tamil Nadu, India*

Abstract: *Understanding that the need for huge volume of data retrieved from the server requires a high performance system with versatile capability, the performance differs based on systems. Fundamentally there are many systems which will do the job of storing and retrieving the data from the server and one such system is Ajax. In this paper, we have worked on the server performance measures using AJAX mechanism. Though the performances of Ajax is low based on push and pull server functions, it has been overcome using state less push operation and pull operation. The result shown in this study provides a realistic approach for an efficient server mechanism.*

Keywords: *PUSH PULL architecture, AJAX framework, Website scalability, Architecture model, Server State notification*

I Introduction

The dynamic updates that are happening in the modern day makes the online system live. This is because of the architecture which balances both, the server and client mechanism. The versatility of an architecture lies on its mechanism. More efficient the mechanism, more versatile the update is. Never even imagine that a server updates the data just longer than a gist of time. To discuss more on the efficiency of the mechanism there must be an impact factor considered to be as the best. This is how this factor improves system efficiency.

Mostly all the data communication mechanism that happens between a server and client is done via push and pull architecture [2]. This architecture is widely used in many concepts; where, lays a scope for client-server mechanism. Some examples that use this concept are live new channel, international currency updates and so on. The updates that make these facilities possible are said to be efficient system and such system will never make a lag in potential data delivery.

The above said concept will vary depending on the size of data retrieval and accessing. All lies in the hands of the server to decide which techniques to be used to do so. The resultant of the discussion paves way for two models declared to be PUSH and PULL model. Both these models have an impact with the system architecture. Depending on the efficiency of these models the server performance is measured.
To explore further these models do the data transmission smoothly where as it will never let the process of data send and received to be noticed. Also it will never establish the size of data that is to be either retrieved or accessed from a server.

This paper proposed an AJAX based strategy for implementing PUSH and PULL model to be efficient. Since the AJAX works based on browser capability, it will make the functionality efficient in terms of client server mechanism. Though HTTP protocol has its own confines, it will be overcome by using AJAX concepts.

II. Related Work

Colossal amount of work were done before for providing efficient service using PUSH and PULL mechanism. All these work were done before identifying the area to be addressed and to improve. The work of Yen-Cheng chen[1] explains the concept of push service in WAP. In this work the author identified some key area for improving the efficiency of PUSH strategy. Further the author combined the aspects of WAP and WWW in his paper.

The key points of PUSH mechanism using a protocol to deal with the problem that arise by using PUSH mechanism are taken.

The work of Yang Zhao [2] addresses the PUSH model using the concept of Model of computation and the various other domains that uses PUSH model. Our previous work [3] deals with the scalable issues in the PULL architecture. Further work of Engin Bozdag [4] compares PUSH and PULLS mechanism along with the AJAX limitation in this work progress. The extension work of Engin Bozdag [5] addresses the data delivery and its transaction in AJAX application.

V. Trecordi and G. Verticale [6] use architecture model for providing effective web services. The work of A. Mesbah and A. Van Deursen address the migration concept using AJAX for multiple pages.We have structured our work with the idea of integrating PUSH and PULL mechanism using AJAX for improving the server performance by triggering the HTTP protocol to deal with scalable issues.

III. The Need For Efficient Push Pull Mechanism

The time gap between the clients sends requests and the server responses may differ. This difference that occurs between these tradeoff communications are referred to be as server lag. In this digital era almost all the communications that happen between two systems referred based on architecture deals with push and pull architecture. The concept and need for these mechanisms was discussed by Yen-Cheng chen [1]. In his paper he discussed the need for push mechanism and its working impact on the client and server system. In addition to this, he mentioned the role of www for retrieving a requested page from a server using wireless medium WAP. Irrespective of the mode of data transmission between client and server via www were majority of request was satisfied by http protocol. Now the discussion is how this http protocol is able to satisfy the request that comes from a client and displays the response from a server? To answer this, the server has to be efficient enough to satisfy both push and pull mechanism or else there will be a potential lag that will affect either of push or pull mechanism or both in particular.

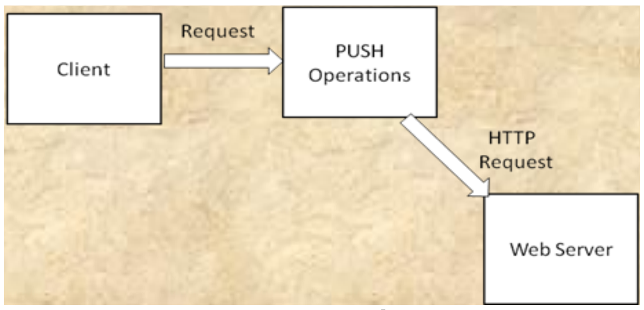


Fig 1: **PUSH** Mechanism

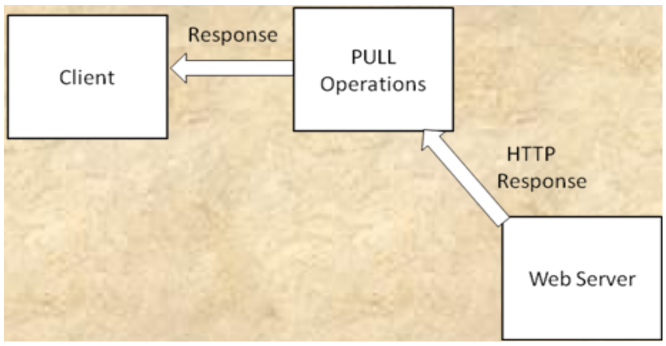


Fig 2: **PULL** Mechanism

III.A Problems with PUSH mechanism

In this mechanism there lies no notification from either end about the data transformation. In other words the system will not focus on acknowledgement to identify that the transformation process is complete or still in process. In extension, this problem grows rapidly along with data volume.

Here in this scenario the problem is directly proportional to data volume.

To identify the server lag with the data size of 100 MB, the server takes 1.5 seconds to retrieve and display or produce the data and as the data size grows to 250 MB the time taken for the server is 3 seconds.

$$(C+S)^n = \sum_{t=0}^{n} C^k S^{n-k} \qquad (1)$$

The lag or a legacy server mechanism of the data transmission will be either affected by the overall burden of the system and that comes when no request is satisfied yet or the data volume of each request has the maximum limit or with the server efficiency.

In the above formulae where t is the time bound between the limit 0 to n and k is the constant value that can be calculated based on server response.

III.B Eradicating PUSH mechanism problem

The solution to the above problem is AJAX mechanism. In this mechanism, the data will be identified by the server irrespective of the data size. If the data can he tackled without involving the server then the server elapse time will be reduced to half for data retrieval. Precisely if the data needs server role then the time will be summed up with the time spent for data retrieval and access.

This lag is due to HTTP protocol which provides access to the all the clients for accessing server data. Since the HTTP is not having a strong hold, data request comes from the various client and the entire client's request are directed to the server and the server starts its work based on the request received from the various clients.

As the client requests increases server performance decreases. If the server continues to serve on particular request of a client and due to the enormous data volume retrieval the lag prevails and the result would be; either, the current running request succeeds or else every other request result in failure.

The HTTP protocol provides stateless services [8] and hence the above problem will never be notified or acknowledged by the server service.

To overcome such issues arising in the name of data volume complexity and server response, the concept of AJAX is introduced to balance such case.

III.C Calculating Scalability

The time gap between the request and response time lag between the client and server were calculated using efficient prototype by the formula

$$f(x) = t_0 + \sum_{n=1}^{\infty} \left(c_n \quad \frac{n}{s} + s \quad \frac{r}{s} \right) \tag{2}$$

From the above formula, t_0 is the time for initiating the request and it was summed up with the client request c_n with the response received by s . All the requests and responses were calculated with the net request and response received from the number of clients and server. The number n corresponds to the number of clients

The time factor t was calculate by

$$t = c_n \times s \tag{3}$$

The combination factor of c_n and s leads to the overall time taken for request and response mechanism. This relation was derived further as

$$t/s \propto c_n \tag{4}$$

The overall time taken by the server depending on the number of requests made by the client and out of which how many requests was satisfied.

IV. Architecture

The Proposed architecture uses the concept of AJAX methodology for dealing with PUSH and PULL mechanism.

Ajax is used for the following reasons

1. HTTP is a stateless protocol
2. Time frame is not measured
3. Server efficiency in not measured
4. Overall time scale is not measured
5. Server scalability was not focused.

The answer to all the above mentioned cases is addressed by our proposed methods.

IV.A PUSH Model

The redefine PULL model [6] from V. Trecordi and G. Verticale was discussed using AJAX

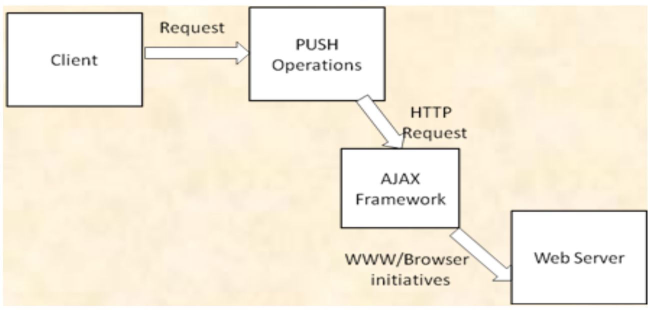


Fig 3: AJAX **PUSH** Mechanism

IV.B PULL Model

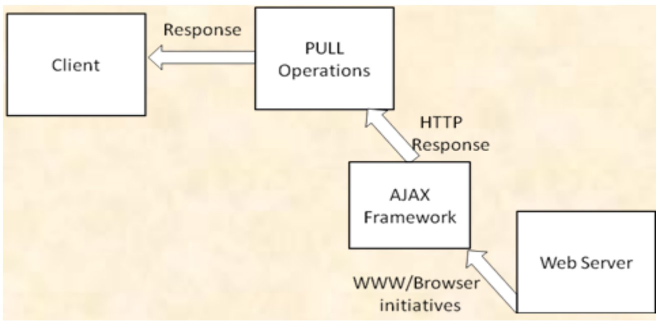


Fig 4: AJAX PULL Mechanism

V. Simulation Result

The study was implemented in AJAX DOJO framework. We have taken 100 users or clients and each client has a page URL as request. The entire requests were taken forward via AJAX DOJO framework. The time interval set were as follows
PUSH time: 5, 10, 15…… 120 were the maximum time is 120 seconds
The interval time considered being as response time or update time comes from the server.
PULL time: 2, 4, 6 …….120 seconds.

The PULL time set as low indexing range since the time taken for PUSH mechanism is directly proportional to time. Server Response time for PUSH and PULL: 1, 5, 10….120 seconds. This is the time taken for PUSH and PULL mode where as it will show the result via AJAX framework via WWW HTTP protocol to identify its state.
The result of this mechanism are shown below

S.NO	*Operation*	*FROM*	*TO*	*TIME*	*USER*
1	PUSH	CLIENT	SERVER	1	1
2	PUSH	CLIENT	SERVER	4	4
3	PUSH	CLIENT	SERVER	6	5
4	PULL	SERVER	CLIENT	1	4
5	PULL	SERVER	CLIENT	1	1
6	PULL	SERVER	CLIENT	4	3
7	PULL	SERVER	CLIENT	1	7
8	PULL	SERVER	CLIENT	10	4

Table 1: Stateless service without AJAX framework

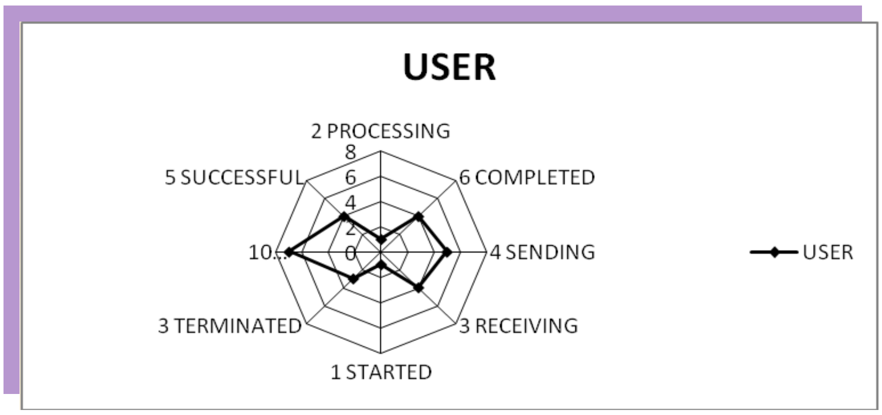
S.NO	*Operation*	*FROM*	*TO*	*STATUS/STATE*	*TIME*	*USER*
1	PUSH	CLIENT	SERVER	**PROCESSING**	1	1
2	PUSH	CLIENT	SERVER	**COMPLETED**	4	4
3	PUSH	CLIENT	SERVER	**SENDING**	6	5
4	PULL	SERVER	CLIENT	**RECEIVING**	1	4
5	PULL	SERVER	CLIENT	**STARTED**	1	1
6	PULL	SERVER	CLIENT	**TERMINATED**	4	3
7	PULL	SERVER	CLIENT	**UNSUCCESSFUL**	1	7
8	PULL	SERVER	CLIENT	**SUCCESSFUL**	10	4

Table 2: Simulation result with AJAX services producing STATUS information's

The stateless mechanism in the absence of AJAX framework was shown in the table 1. Generally, stateless mechanism without using AJAX is not possible without the dominance of HTTP protocol. The combination of the AJAX framework along with HTTP protocol does the job for server response time and our assumption is no such work will produce server response using protocol stack with AJAX framework.

In the above table client 4 successfully posted data into server and the server receives it successfully and provides the status of data transmission. Other users know the status of the data transmitted to the server with the status updates that comes from the server.

The graph below shows the time is evenly balance as the result will be published by the server depending on the transmission. Irrespective of increase in the number of user, server focus on the data send and receive and also with the acknowledgement, to be send as server initiative mechanism referred as server initiatives notifications. The graph depicts all the operations that come in the form of PUSH PULL mechanisms. It also gives the maximum deviated path of a server for providing effective notifications.



The graph depicts all the operations that come in the form of PUSH PULL mechanisms

VI. Conclusion and Future Work

In this paper we have imposed AJAX framework for converting HTTP protocol from stateless to stateful process for improving the time scalability of the server. This will make the server efficient in terms of time complexity and providing potential result of the process. Such model makes a server work efficiently by knowing its limitations. In this effort we haven't worked on database issues for storing the data once the server posted successful status info. The future work reflects on this case along with security breaches and how the server overcomes it.

References

[1] Yen-Cheng chen, "Enabling Uniform Push Services for WAP and WWW", *Proceedings of Workshop on the 21st Century Digital Life and Internet Technologies*, 2001.

[2] Yang Zhao, "A Model of Computation with Push and Pull processing", Research project, University of California at Berkeley, December 16, 2003

[3] J SARAVANESH, Dr.E.RAMARAJ, "Scalable Transaction Authorization Using Role Based Access Control for Time Based Content Access with Session management", International Journal of Engineering Research and Development eISSN : 2278-067X, pISSN : 2278-800X, www.ijerd.com, 2012

[4] Engin Bozdag Ali Mesbah Arie van Deursen, "A Comparison of Push and Pull Techniques for AJAX" TUD-SERG-2007-016a

[5] Engin Bozdag Ali Mesbah Arie van Deursen, "Performance Testing of Data Delivery Techniques for AJAX Applications", TUD-SERG-2008-009

[6] V. Trecordi and G. Verticale. An architecture for effective push/pull web surfing. In 2000 IEEE International Conference on Communications, volume 2, pages 1159–1163, 2000.

[7] A. Mesbah and A. van Deursen. Migrating multi-page web applications to single-page Ajax interfaces. In CSMR '07: Proceedings of the 11th European Conference on Software Maintenance and Reengineering, pages 181–190. IEEE Computer Society, 2007.

[8] Mikko Pohja , "Server Push for Web Applications via Instant Messaging", Journal of Web Engineering, Vol. 9, No. 3 (2010) 227–242