A Comparison of the Efficiency of Using HTML over XML and JSON for the Asynchronous Communication in Rich Internet Applications

NR Dissanayake[1#], D De Silva[2] and GKA Dias[1]
[1]*University of Colombo School of Computing, Colombo 7, Sri Lanka*
[2]*Informatics Institute of Technology, Wellawatta, Sri Lanka*
[#]*nalakadmnr@gmail.com*

Abstract *— Rich Internet Applications use XML or JSON to format the data in asynchronous communication, which engaged a serialization process in the server and a de-serialization process in the client. If we use HTML to format the data it can get rid of the de-serialization process and make the development easier. In this paper we compare the efficiency of using HTML over XML and JSON, for asynchronous communication in Rich Internet Applications, by the means of time and the size of the data. Based on the results, we expect to introduce a set a facts to consider when selecting the technique / technology for the asynchronous communication in Rich Internet Applications.*

Keywords *— Rich Internet Applications, Asynchronous communication, HTML, XML, JSON*

I. INTRODUCTION

In this era of Web2, Rich Internet Applications (RIAs) have gained the demand of the users, with enhanced user experience via rich User Interfaces (UI), and faster responds (Lawton, 2008). Asynchronous communication in RIAs between the client and the server plays a major role in providing rich features, which respond faster (Busch & Koch, 2009). When the user initiates a process, the client-side RIA engine sends an asynchronous request to the server, and the server processes the request and sends only the results – instead of a complete web page – back to the client. The client-side app then processes and shows the results on the current page by updating only the necessary segments on the UI. This partial page rendering nature along with the asynchronous communication, enables developing rich UI components and rich features in web applications (Busch & Koch, 2009).

The data is sent from the client to the server as parameters along with the request, using GET or POST form methods. The respond from the server to the client may contain larger data set(s), where the client is supposed to understand and process the data; and display information on the UI. Extensible Markup Language (XML) (Bray, et al., 2006) or JavaScript Object Notation (JSON) (T. Bray, 2014) is used to ensure a good structure and semantic of the data exchanged in both request and respond of the asynchronous communication.

XML had originally been introduced to define structure(s) for data sets in storing and communication of data. It uses a nested arrangement of elements to provide structures for data; and uses Attributes to describe them further. The usage of XML spreads over a larger domain, but here we limit its scope to the use in the asynchronous communication of RIAs.

In web applications' data communication process, the time taken for the communication and the size of the data communicated are two main factors to be considered. Larger data sets introduce traffic in the network and may affect the speed of the communication too. Since the size of the XML data set is considered large, JSON has been used as a better solution for the data communication in web applications. JSON is a light weighted and text based format, with a small set of formatting rules to form a portable set of structured data. And JSON was proven that it is better than XML in data communication (Lin, et al., 2012).

Despite the technology used for the asynchronous communication, the process contains the following steps in the processing algorithm. The client sends the request; and the server processes the request and prepares the respond by serializing the data, which means preparing the XML or JSON structure of the data. Then the server sends the serialized data to the client and the client de T serialize/parse/extract data from the XML or JSON structure and re-formats them to be shown on the UI. Finally the client performs the partial update of the UI and renders the information to the user. Figure 1 shows the steps of the abstract algorithm of the asynchronous communication process.

In our ongoing research, we do experiment for better techniques to simplify the engineering of RIAs. Our main focus is to reduce the complexities by increasing the realization of the RIAs, using an abstract RIA architecture (Dissanayake & Dias, 2014). While conducting the experiments, we noted that using HTML structures for asynchronous communication can minimize some complexities in the asynchronous communication algorithms; hence it could make the development easier. Therefore we conducted this research – parallel to the main research – to compare the efficiency of using the HTML over XML and JSON by the meanings of communication time and the size of data communicated.

The methodology is presented in section II, and the discussion of the results is presented in section III with some criteria to be considered when selecting the communication technique. We conclude our findings in section IV mentioning the future work.

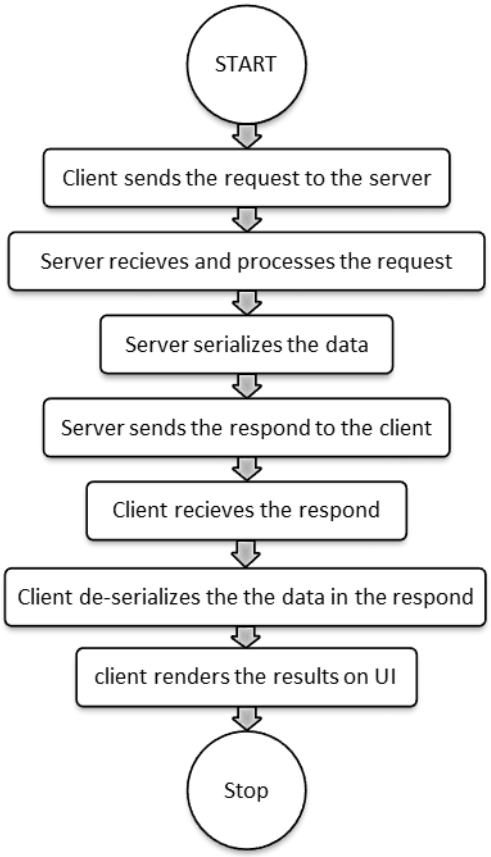


Figure 1: The steps of the abstract algorithm of Asynchronous communication process

II. METHODOLOGY

Here we discuss the methodology of the research under the title of this paper, excluding the details of the main on-going research.

For the comparison of the efficiency of the HTML over XML and JSON, we developed a small tool with following features. The tool sends a request to the server on a click of a button. There are 3 different buttons for HTML, XML and JSON and 3 dedicated methods in the server to use the specific technique to serialize the data. The tool reads the number of iterations and the number of entries as inputs. The request is sent as an AJAX request using the GET method, querying for a set of data. Server processes the request by reading a set of data from the database and serializing the data set using the demanded technique – according to the button used to initiate the communication – and sends the respond back to the client. Once the client receives the data set, it is processed and displayed in a table on the page. Figure 2 shows a section of the UI of the tool.

For the client-side development JavaScript and jQuery were used. The server-side was developed using PHP. MySQL was used as the database server and the Apache server was used to host the tool. For the asynchronous communication, AJAX and a RIA-Bus (Dissanayake, et al., 2015) were used.

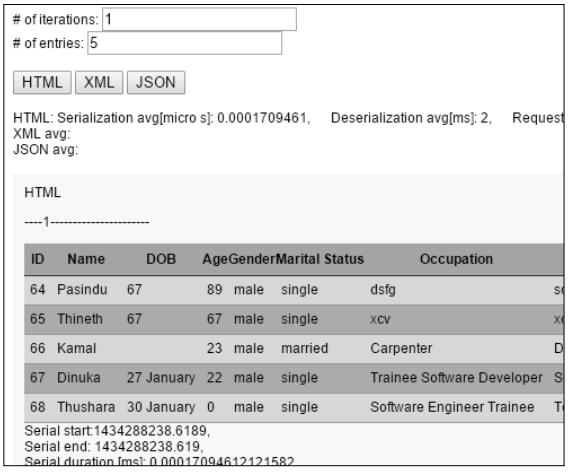


Figure 2: A section of the UI of the testing tool

First we conducted the experiments in the local host. Then to expand the scope to get a better view, we conducted the same set of experiments again using a remote server. The communication with the remote server was done in two different modes, 1) via a proxy and 2) using a direct connection. Tables 1, 2 and 3 include

the specifications of the technologies and platforms used in the three environments.

Table 1: Using localhost

Specs	Client	Server
Processor	Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 2.30GHz	
RAM	4GB	
Link	Local	
OS	Windows 8.1 (64)	
Platform	Chrome	Apache/MySQL [XAMPP]

We measured the time taken for the serialization (in micro-seconds) in the server, de-serialization (in milliseconds) in the client, and the total time duration (in milliseconds) for the complete process. Additionally we measured the size of the data set (in bytes) sent from the server to the client in each request.

To have a better understanding on how the time efficiency vary with the size of the data, we conducted the experiment for 4 different sizes of data sets, with number of entries of the final table 10, 20, 50 and 100. To have accurate measurements, the tool was developed in a way to perform the same request-respond process for a given number of iterations and calculate the average. We did set the tool to perform the same request-respond cycle for 10 times and get the averages of the time durations.

Table 2: Using remote server via proxy server

Specs	Client	Server	Proxy
Processor	Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 2.30GHz	32 CPU cores	Intel Quod Core 2.16ghz
RAM	4GB	32GB	4GB
Link	LAN-Ethernet	Internet	Internet:- ADSL LAN:- Ethernet
OS	Windows 8.1 (64)	Linux - 3.12.35.1418868052, Architecture - x86_64	Endian 2.4.1
Platform	Chrome	Apache version - 2.2.29, PHP version 5.4.24, MySQL - 5.5.42-37.1-log	Endian firewall

Table 3: Using remote server directly

Specs	Client	Server	Dongle
Processor	Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 2.30GHz	32 CPU cores	
RAM	4GB	32GB	
Link	USB	Internet	3G - HSPDA
OS	Windows 8.1 (64)	Linux - 3.12.35.1418868052, Architecture - x86_64	Service provider – Mobitel Sri Lanka
Platform	Chrome	Apache version - 2.2.29, PHP version 5.4.24, MySQL - 5.5.42-37.1-log	

III. DISCUSSION

Here we discuss the efficiency of performance under two categories, the time taken for processing and the size of the data communicated.

A. Time taken for processing

In the usage of HTML for communication, the construction of the HTML table structure in the server is considered as the serialization process. The de-serialization process in the client is limited just to displaying the received HTML content on the UI, by inserting the complete HTML table structure – which is received as the data set from the server – in to a division element.

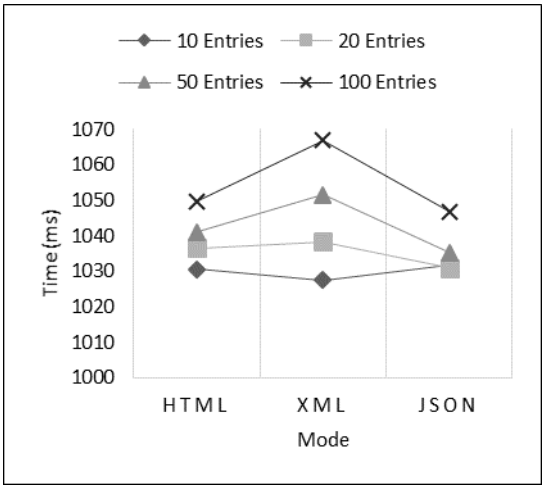


Figure 3: Complete process time comparison in Localhost

1).Localhost: For all serialization, de-serialization and complete process time, JSON performed best, and HTML is just behind JSON. XML exhibits a noticeable lack in performance. Figure 3 shows the complete process time comparison in localhost.

2) Remote server via proxy: Similar to the results in localhost, for the serialization, de-serialization and complete process time, JSON performed best, and HTML performs better than XML. Figure 4 shows the complete request time comparison when the remote server is used via proxy.

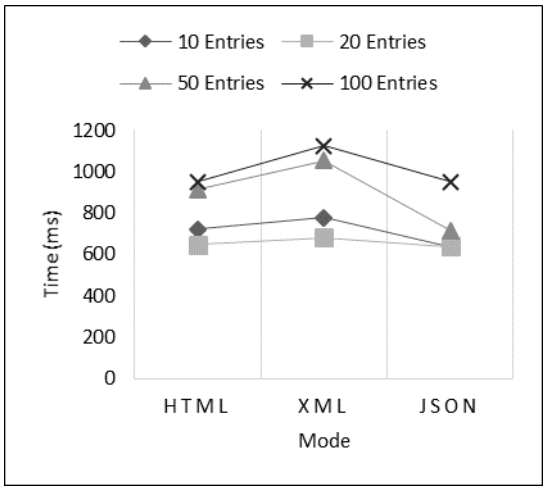
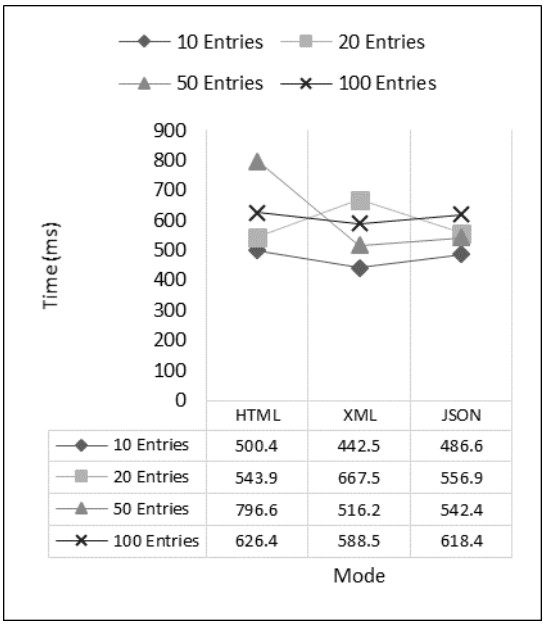


Figure 4: Complete process time for remote server via proxy

3) Accessing the remote server directly: In separate serialization and de-serialization time durations, similar to the other environments JSON has performed best, HTML is almost similar to and behind the JSON, and better than XML.

But for the complete process time in this environment, some cases show different behaviours than in other two environments. In the cases of 10, 50 and 100 entries, XML has performed best for the complete process time. In the case of 20 entries, HTML has performed best.
These results does not show any pattern as in other two environments, but all the time durations of the complete process are lower than the other two environments, which means the efficiency is higher.



	HTML	XML	JSON
10 Entries	500.4	442.5	486.6
20 Entries	543.9	667.5	556.9
50 Entries	796.6	516.2	542.4
100 Entries	626.4	588.5	618.4

Figure 5: Complete process time for remote server accessed directly

B. Size of the data communicated

When considering the experiment series using the localhost, since both the server and the client are in the same platform, we did not measure the size of the data communicated. In both the cases of accessing the remote server directly and via the proxy server, the results were identical. Figure 6 shows the results of the sizes of the communicated data sets, when the remote server is accessed directly.

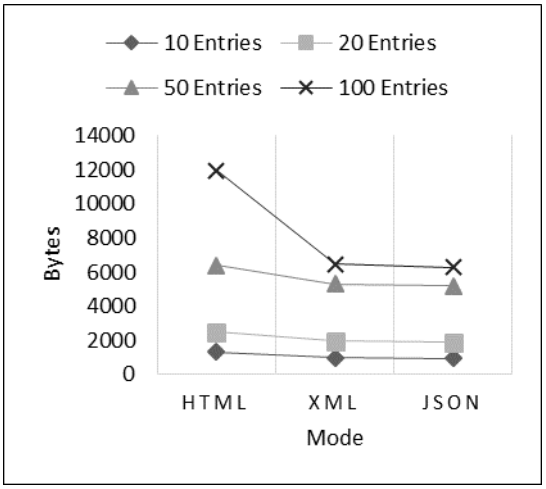


Figure 6: Data size when remote server accessed directly

As expected, JSON produced the data set with the lowest size, then XML and HTML produced the data set, which has the highest size. In the case of 100 entries, the size of

the communicated HTML data set is almost double the size of the JSON data set.

C. Analysis and the advantages of using HTML in asynchronous communication

The asynchronous communication in RIAs is limited only to the data sets necessary to display the required information, instead of loading complete page(s). In the requirements of displaying larger data sets in grids/tables, RIAs use pagination pattern (Anon., 2009), hence a smaller number of entries like 10 to 20 are displayed at a time. When a new data grid page is requested, only the data set need for the particular data grid page will be sent by the server to the client. Even though we conducted the experiments for 50 and 100 entries, it was only to identify the deviation(s) of the efficiency with the size of the data, and we presented the results just for the knowledge. For the conclusions we utilize the results of the 10 and 20 entries cases only, as our primary target of the research is comparing the efficiency of the techniques for the asynchronous communication in RIAs.

Analysing all the results of the series of the experiments, for the time efficiency we can highlight that the HTML is efficient than XML and performs very close to JSON. For the data size efficiency, HTML performs lower than both XML and JSON, but it is limited to less than 100 bytes. We do not think this as a drawback to be emphasized, according to the greater power of the available technologies and resources nowadays, such as faster communication technologies like 3G or 4G and higher bandwidth of the servers and networks.

When the HTML is used for the asynchronous communication, the de-serialization is limited just to displaying the data on the UI, without any parsing or further processing, hence it reduce the work load in the client-side development.

Furthermore, when the view's presentation of the data set needs to be modified, it needs only the modifications in the server-side code, since there is no parsing in the client-side. Therefore, the client's code does not need to be modified as the server code changes for the particular data set. This decreases the coupling between the client and the server, thus enhances the modifiability property of the RIA.

D. Facts to consider when selecting HTML in asynchronous communication

The time is a critical factor, especially when it comes to real time communication like in stock market related ecommerce systems. If the time is a critical factor and even few tens of milliseconds matter in the system, it is advised to select JSON over HTML. On the other hand, RIAs with AJAX based asynchronous communication use the request-respond model, hence not suitable for time critical systems, therefore before the data formatting techniques, had better consider better communication technique(s) like web socket and/or other critical resource factors like hardware, processing, bandwidth, etc...

When the system has a large client base and/or high amount of communication needs to be done, but the server bandwidth is limited, then it is not recommended to use the HTML for the asynchronous communication.
In critical situations where even less than 100 bytes and 100 milliseconds are substantial, the JSON might be used, but using JSON lacks in modifiability than HTML in the terms of RIAs

IV. CONCLUSION

We can recommend the use of HTML for the asynchronous communications in RIAs, as it increases the modifiability of the RIA. But some other time and bandwidth related facts also should be considered in critical situations.

In future, utilizing the advantage of absence of client-side parsing – when the HTML is used – we hope to derive some patterns in client-side asynchronous communication processing algorithms. And we expect to introduce more abstract and generic algorithms for forms and grid based Create, Read, Update, and Delete (CRUD) operations in RIA. Extending this idea – of using the abstract algorithms – we hope to introduce a JS library to minimize the development work load in the client-side.

IV. REFERENCES

Anon., 2009. Research and Implementation of Pagination Algorithm over Massive Data Based on Ajax Technology. Wuhan, IEEE, pp. 1-4.

Bray, T. et al., 2006. Extensible Markup Language (XML), s.l.: W3C.

Busch, M. & Koch, N., 2009. Rich Internet Applications - State-of-the-Art, Munchen: Ludwig-Maximilians-Universitat .

Dissanayake, N. R. & Dias, G. K. A., 2014. Essential Features a General AJAX Rich Internet Application Architecture Should Have in Order to Support Rapid Application Development. International Journal of Future Computer and Communication, 3(5), pp. 350-353.

Dissanayake, N. R., Dias, G. K. A. & Ranasinghe, C., 2015. RIA-Bus: A conceptual technique to facilitate the AJAX-based rich internet application development. Badulla, Sri Lanka, s.n.

Lawton, G., 2008. New Ways to Build Rich Internet Applications. Computer, August, 41(8), pp. 10 - 12.

Lin, B., Chen, X., Chen, Y. & Yu, Y., 2012. Comparison Between JSON and XML in Applications on AJAX. s.l., IEEE Computer Society, pp. 1174-1177.

T. Bray, E., 2014. The JavaScript Object Notation (JSON) Data Interchange Format, s.l.: Internet Engineering Task Force (IETF).