MEST Journal
DOI 10.12709/mest.06.06.02.15

COMPARATIVE ANALYSIS OF DATA TRANSFER TECHNOLOGY BETWEEN SERVER AND CLIENT: *AMF* AND OBJECTS ON THE WEBSITE

Munir Sabanovic
University of Novi Pazar, Department of Computer Sciences, Novi Pazar, Serbia

Muzafer Saracevic
University of Novi Pazar, Department of Computer Sciences, Novi Pazar, Serbia

Kemal Dzemic
University of Novi Pazar, Department of Computer Sciences, Novi Pazar, Serbia

JEL Category: **L86**

Abstract

This paper presents a comparative analysis of data formatting technology in AMF when transferring data between client and server in the author application and on http://www.baguetteamf.com/. Given is an authoring application that allows timing and analysis of the AMF objects' transport, this technology. Included here are aspects of data speed rate, output file size, data transfer security, code complexity on the client's and the server's part. The time of data transition, packed in objects is divided onto the query preparation time (serializations), a connection establishing time, deserialization time on the server and a client side, and a query time in a database. By analyzing this time periods, it is established where the most of time is spent, from the moment of a data request to the moment of a data transportation. In a working application, the approximate time of a data loading is measured by fifty iterations for a certain number of objects, whereby the number of objects is selected in a combo box, within the work application. At the end of the process, the chart is displayed with a time of serialization, deserialization, as well as a connection establishing, with and without Baguette AMF addendum, on the AMF-PHP component, in the rage from 100 to 1000 subscribed objects on an authorized application, and for the objects on a specified website.

Address of the corresponding author:
Munir Šabanović
munir.sabanovic@uninp.edu.rs

Keywords: AMF, PHP, client-server communication, data transfer, objects on the website.

1 INTRODUCTION

The author's application is provided in the paper, which enables measuring the data transfer time from the moment the request is sent to the server until the moment the data is presented to the user. Measurement is achieved for the AMF (Action Message Format) data packing technology. The number of data from the database is selected by the user in the combo box, ranging from 1 to 76000. The time required to transfer the data from the server to the client can be divided into several stages. Each phase is analyzed in detail and determined which of them introduces a delay in data transfer. The difference in the data transfer time, in addition to the transfer technology, is influenced by the size of the output file, the complexity of the code on the client and the server's part. The paper analyzes the aspect of data transfer security. Finally, recommendations are given when to use AMF technology.

2 USED SCIENTIFIC METHODS AND PROCEDURES

To realize the objectives and the tasks of the research, the following scientific methods and procedures were used:

- The comparative method compares the results of the existing methods and newly proposed methods of problem-solving.
- Experimental application yields result that show the effectiveness of new methods (in speed or saved memory space).
- The generalization method has been applied in the analysis of a number of cases where a general assertion is made that applies to all cases. The specialization method presented certain cases in the form of a concrete example.
- Methods of deduction and induction were used during experimental testing, where, after the obtained results, a conclusion is formed about new techniques and possibilities of their application.

3 USED TECHNOLOGIES FOR CREATION OF THE AUTHOR'S APPLICATION

For the development of software solutions, on the client's part Flex technologies were used and PHP on the server's part, while the AMF technology was used for data transfer (Florescu, 2013; Pettit, 2013; Merelo-Guervos, 2008; Jun, 2008).

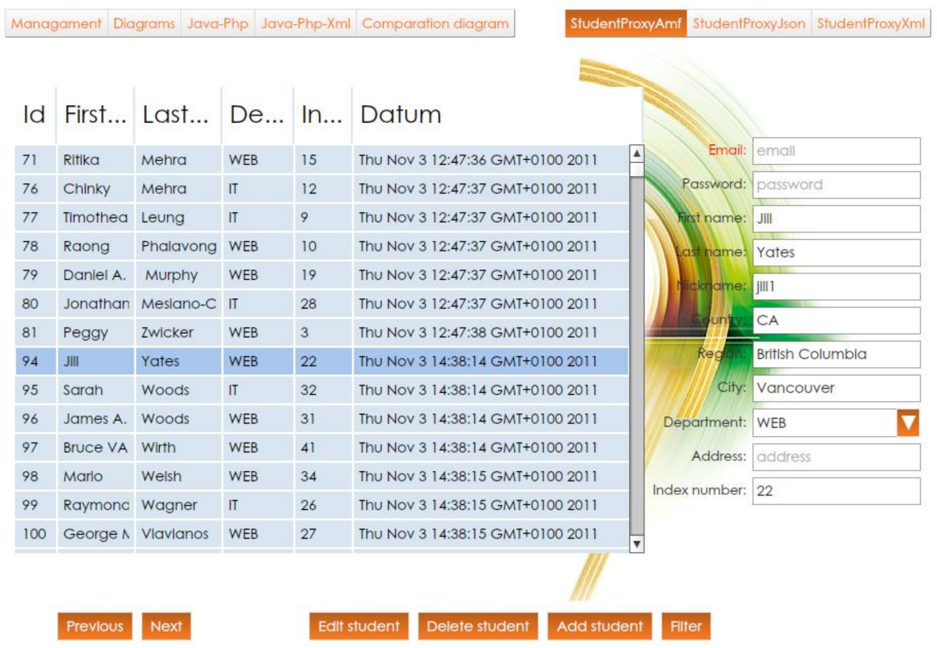
Flex is an extremely productive, free open source software environment for executing executable versions of expressive mobile, Web and computer applications. Flex enables the creation of executable versions of Web and mobile applications that share a common base code, which shortens time and reduces the costs of application creation and long-term maintenance While the Flex application can be created only with the free Flex SDK, the Adobe Flash Builder ™ software can accelerate the development thanks to features such as intelligent code editing, step-by-step troubleshooting, memory optimization and performance optimization programs and visual designing (Rodrigues, 2011). Flex in its environment contains two MXML languages for visualization and AS3 for functionality (Zimmermann, 2003).

The PHP server language is executed on the server's part. PHP is called a scripting language because it is written in the form of scripts and is intentionally made for use on the Web (Pettit, 2013). PHP can be written in special files and can be inserted into HTML. The PHP processor on the Web server interprets the PHP code, and the Web server outputs HTML or other types of data that can be understood in the client's Internet browser. A copy of the HTML page is sent directly from the server to the client's computer, while the PHP code is not sent directly, but it is before converted into a format that the client's computer will be able to interpret. HTML parts in the script are left as they are while the PHP code is interpreted by the PHP processor and executed, and the execution result is sent to the client. The PHP code has great capabilities, from communicating with other computers, creating images, accessing databases, working with graphics, creating desktop applications using PHP-GTK extensions up to reading and writing files. PHP does not have its owner, it's a free language, a group of enthusiasts has met and made the PHP. For AMF technology, they took Action Message Format and made the slitters to match the Action Message Format in PHP.

AMF or Action Message Format is a binary format used to serialize objects and sending messages between a client and a remote service. Action Script 3 language has classes for encoding and decoding the AMF formats. Adobe Systems issued the AMF binary protocol specification on December 13, 2007 and announced that it will support the programming community in making this protocol for the all large server platforms. Thus, today there is AMF support for platforms written in Java, PHP, .NET and other languages. This paper will direct its attention to PHP language because it is the chosen server platform (Jorstad, 2008).

4 DESIGN AND FUNCTIONALITY OF THE APPLICATION

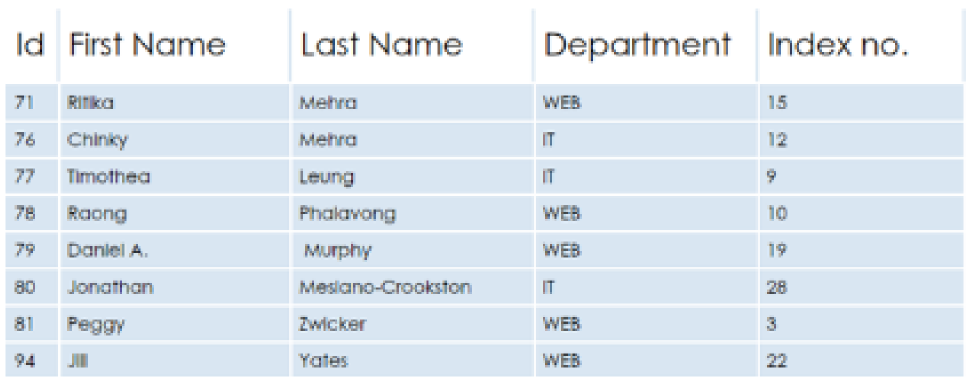
Figure 1. shows the working environment of the application. The application's working environment includes a grid, with columns id, First name, Last name, Department, Index no. and Date, Figure 2. Columns are loading data from the database which is located on the Wampserver2.4-x86 server. The number of data loaded is selected from ComboBox, ranging from 1 to 76 000 data.



Id	First...	Last...	De...	In...	Datum
71	Ritika	Mehra	WEB	15	Thu Nov 3 12:47:36 GMT+0100 2011
76	Chinky	Mehra	IT	12	Thu Nov 3 12:47:37 GMT+0100 2011
77	Timothea	Leung	IT	9	Thu Nov 3 12:47:37 GMT+0100 2011
78	Raong	Phalavong	WEB	10	Thu Nov 3 12:47:37 GMT+0100 2011
79	Daniel A.	Murphy	WEB	19	Thu Nov 3 12:47:37 GMT+0100 2011
80	Jonathan	Mesiano-C	IT	28	Thu Nov 3 12:47:37 GMT+0100 2011
81	Peggy	Zwicker	WEB	3	Thu Nov 3 12:47:38 GMT+0100 2011
94	Jill	Yates	WEB	22	Thu Nov 3 14:38:14 GMT+0100 2011
95	Sarah	Woods	IT	32	Thu Nov 3 14:38:14 GMT+0100 2011
96	James A.	Woods	WEB	31	Thu Nov 3 14:38:14 GMT+0100 2011
97	Bruce VA	Wirth	WEB	41	Thu Nov 3 14:38:14 GMT+0100 2011
98	Mario	Welsh	WEB	34	Thu Nov 3 14:38:15 GMT+0100 2011
99	Raymonc	Wagner	IT	26	Thu Nov 3 14:38:15 GMT+0100 2011
100	George N	Vlavianos	WEB	27	Thu Nov 3 14:38:15 GMT+0100 2011

Email:	email
Password:	password
First name:	Jill
Last name:	Yates
Nickname:	jill1
Country:	CA
Region:	British Columbia
City:	Vancouver
Department:	WEB
Address:	address
Index number:	22

Figure 1. Application's working environment.



Id	First Name	Last Name	Department	Index no.
71	Ritika	Mehra	WEB	15
76	Chinky	Mehra	IT	12
77	Timothea	Leung	IT	9
78	Raong	Phalavong	WEB	10
79	Daniel A.	Murphy	WEB	19
80	Jonathan	Mesiano-Crookston	IT	28
81	Peggy	Zwicker	WEB	3
94	Jill	Yates	WEB	22

Figure 2. DataGrid component on the application's working environment.

The application contains two *Bar button* buttons. The first Bar button, the stateSelector button contains five states:

- *Management,*
- *Diagrams,*
- *Java-Php,*
- *Java-Php-XML,* and
- *Comparison diagram*.

This control is defined by the code block (Hall, 2011):

```
<s:ButtonBar
        top="25" left="30"
        dataProvider="{statesProvider}"
        labelField="label"
        id="stateSelector"
change="stateSelector_changeHandler(event)"
/>
```

The dataProvider is a series of statesProvider, which is defined by the following code block:

```
<s:ButtonBar
        top="25" left="30"
        dataProvider="{statesProvider}"
        labelField="label"
        id="stateSelector"
 change="stateSelector_changeHandler(event)"
        />
```

The authoring application loads data from two different servers, Wampserver2.4-x86, which works with PHP and NiTi server, which works with Java. If the Management state is active, then the operating environment of the application is displayed. However, if another Diagrams state is active, then the AMF data transfer graph is displayed.

The other Bar button, selectProxyBar, changes the appearance in regard to the status of the application. In the Management state, the StudentProxyAmf button is displayed. The StudentProxyAmf button offers a choice of data transfer technology from the base into the application in AMF.

Button controls, whose labels are Previous, Next, Edit student, Delete student, Add student and Filter allow, respectively, to display the previous data block from the database in the grid, the next data block, editing of the selected item in the grid and the database, deleting the order in the database, adding a new student in the database and filtering data in the grid.

Depending whether the state in the classes StudentControlComponent and StudentManagerComponent, normal or search, the filter control can take two values for the Filter and Back labels. TextInput fields and ComboBox are used to display items from the selected grid row, change content in the selected row and add new content. In the state normal, all TextInput controls and ComboBox are displayed, while all TextInput fields are not displayed in the search state.

5 EXPERIMENTAL RESULTS AND COMPARATIVE ANALYSIS

The application measures the data transfer time from the server to the client for data formatting technology, AMF. The execution time requirement is measured from the moment the request is sent to the server, from the client's part to the moment of loading the data from the server into the Flex application. AMF is a protocol used for communication between the client and the server. Analyzed is the aspect of the data packet rate, the manner of data formatting, the complexity of the application creation, the file size to be transported, and the complexity of parsing data from one format to another.

AMF works with binary data, it translates a language into binary data on the server's part, while on the client's part an inverse process is performed, the deserialization, which translates zeros and units into a visual code, which is displayed within the graphical interface.

The work application uses the AMF-PHP package for communication between Client ActionScript3 and the server PHP language.

When transferring data between the client and the server, there are several steps, each step requiring a certain amount of time. On the client's part (Flex) there is a query time (serialization) and deserialization time, while on the server's part there is a deserialization time, the execution time of the query in the database, and the time of serialization. There is also data transfer time. These times are shown in Figure 3.

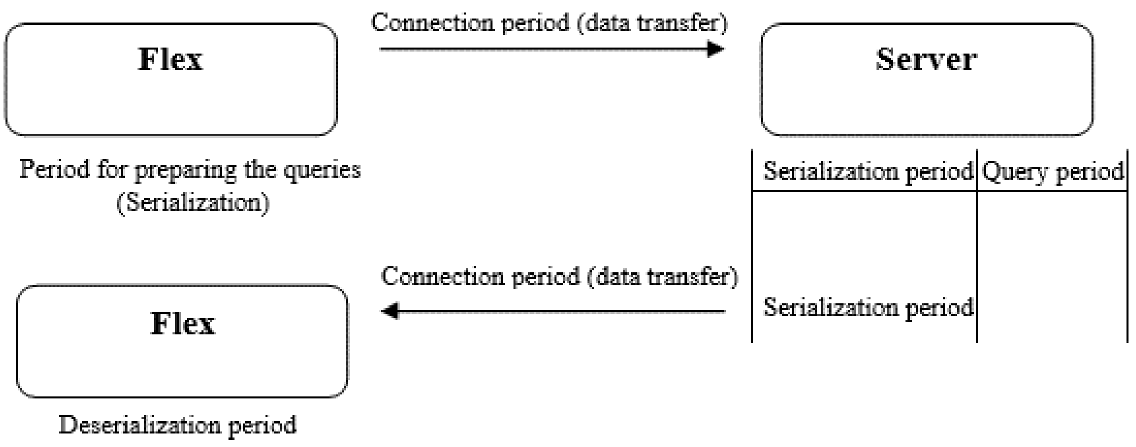

Flex
Connection period (data transfer)
Server
Period for preparing the queries (Serialization)
Serialization period
Query period
Connection period (data transfer)
Serialization period
Flex
Deserialization period

Figure 3. Time display, when transporting data from the server to the client.

Queries are extremely fast for the AMF technology; the order is microsecond and this time is not considered. The deserialization time on the server's part involves converting binary data into objects of a language. Query time is used to read data in the database, these data are afterward serialized and then transferred towards the client. In the end, the data on the flex side is deserialized for a certain period of time. Serialization has a global meaning, namely, in computer science, in the context of data storage, serialization is the process of translating data structure or objects into a format that can be saved (in a file or buffer memory or can be sent over the network) and reconstructed, later in the same or different computer environment.

From the 2.0 version, the AMF production has been taken over by another company and has become considerably slower than the previous version. The slowness is introduced by serialization, where it takes a lot of time to convert objects of a language into binary data, as shown in Figure 4. This problem is solved by adding plug-in, a Baguette AMF, to the basic version of AMFPHP.

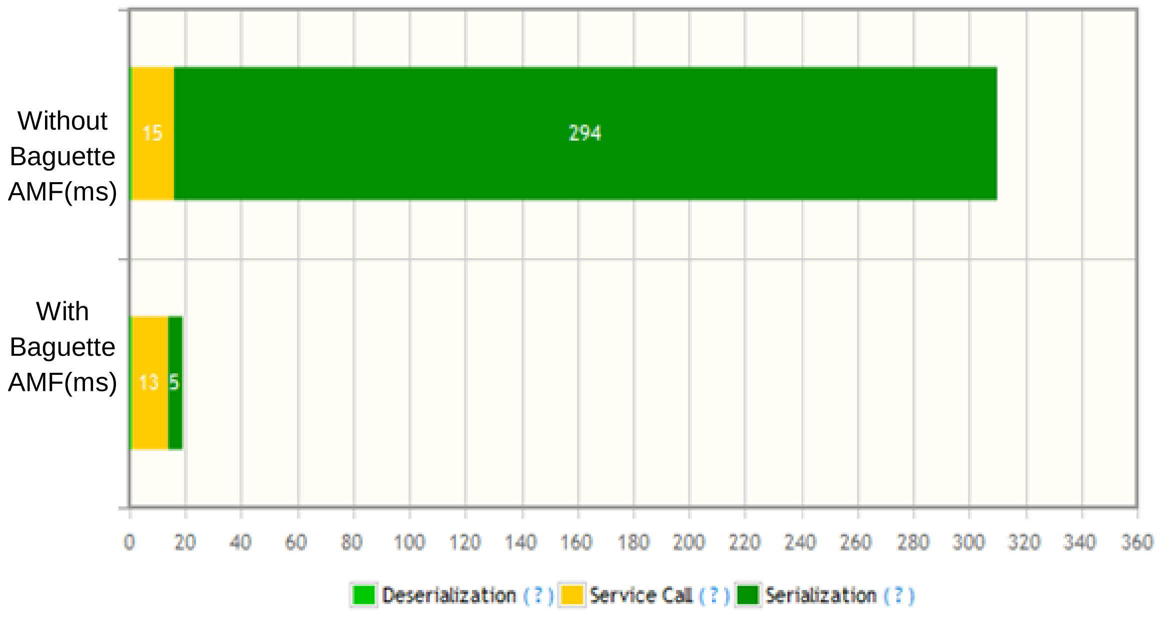


Figure 4. Time of serialization, deserialization and connection establishment, when the Baguette AMF is not used and when it is used.

The application analyzes three important times, the time of serialization, the deserialization time on the client and the server's part, the time of establishing a connection between the server and the client. The middle line defines the service call execution time.

The connection time for AMF technology is approximately the same with and without the Baguette AMF add-on. Deserialization times are slightly different in either case. In Figure 4., the time of data serialization is drastically different with and without Baguette AMF. The shorter serialization time with Baguette AMF is the acceleration of the AMF.

By adding the Baguette AMF plug-in to the AMFPHP basic version makes the AMF faster, when transporting a larger amount of data, thereby generating the Baguette AMF. Baguette AMF with AMFPHP is recommended for company applications that exchange large amounts of data.

The paper translates PHP objects into AMF, which is readable to the Flex client. Flex can serialize data in AMF format and deserialize the data from this format. In Figure 5., communication between the client and the server is displayed, where PHP is used as the server language, and serialization and deserialization of data are done by BaguetteAMF.

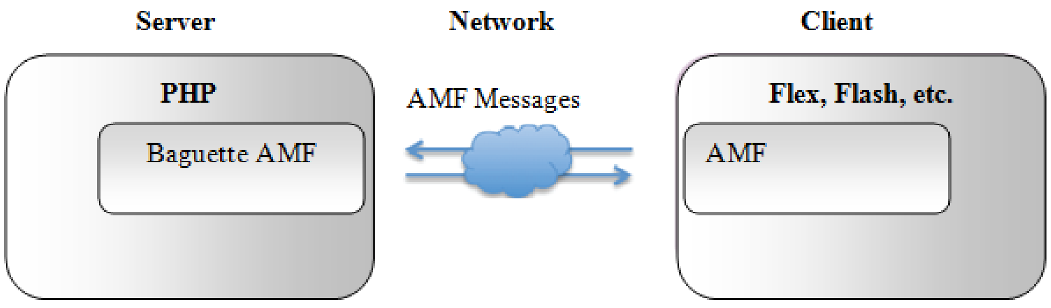


Figure 5. Flex - PHP communication

The client has a built-in AMF format, as seen in Figure 5. AMF technology has wide application. It belongs to cross technologies, which means they do not depend on the operating system, nor on the computer environment.

In the application, when the Diagrams component is active, Figure 1, the graph for the AMF technology is displayed. A comparative graph of the data transport speed, from the server to the client, which captures the time of serialization, deserialization, the time of the connection establishment, is given in Figure 6. In ComboBox, a selection is performed of a number of data that is loaded from the database on the server into the client application. The choice of AMF transmission technology is done using the button bar of the selectProxyBar. For AMF technology, fifty loading iterations are performed, with the same number of data being loaded in each iteration, the time is measured, and at the end of the iteration, the calculation of the meantime is done.

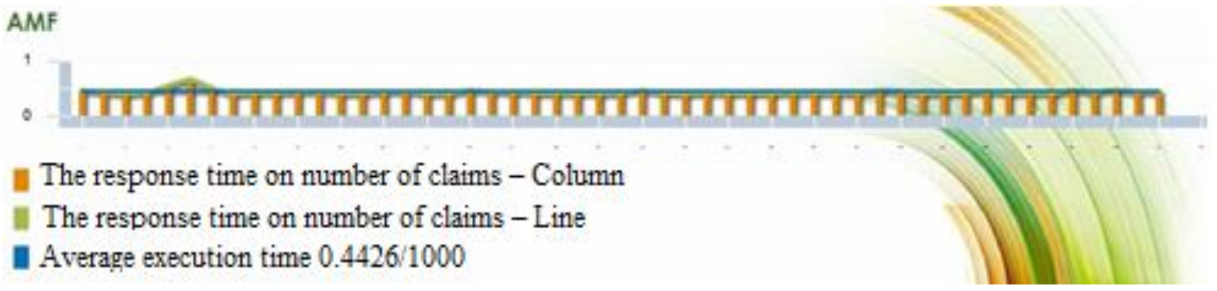


Figure 6. Graphs of time dependence on the number of data loaded for the AMF technology

The test was done on a computer with the following performances: CPU – QuadCore AMD Phenom X4 9550, 2200 MHz (11 x 200), L1 64 KB per core, L2 512 KB per core (L3, (On-Die, ECC, NB-Speed), RAM Memory - 3 Gb, Graphics card - nVIDIA GeForce 9400 GT. The authoring application environment was used for testing.

On the graph, each column represents the loading time of the data. The horizontal curve line links the tips of the columns, and the straight horizontal line on the graph describes the mean loading time of the data. The graphs represent the time when loading 1000 data from the base.

On many data, the AMF achieves a significant speed. In Figure 7., based on the measurements in this model, a time-dependent diagram of the data number is displayed. Figure 7. shows that AMF becomes much faster. Therefore, it is recommended that AMF technology is used above this number of data.

A curve that describes AMF technologies, the third line, is significantly less inclined, which means that time does not grow significantly with an increase in the number of data loaded.

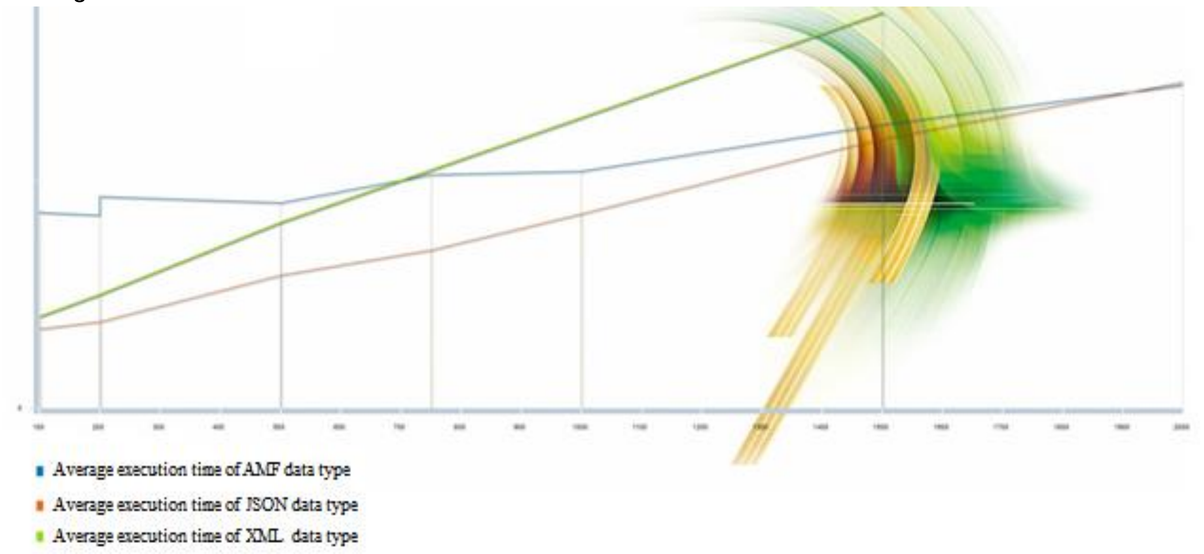


Figure 7. A Comparison Data Flow Rate for AMF Technology, for a range of 100 to 2000 data, with a step 100.

AMF does not work with HTTP connection. The advantage of the AMF is opening of a socket connection, through which the binary data is transmitted to the client. With Flex, in AMF technology, the RemoteObject class is used to establish a connection to the server, which inherits the Proxy class. AMF performs asynchronous communication, where data is sent and received without the agreed timing of sending and receiving data.

With AMF, the data transfer time does not depend on the amount of data, because it is a binary transmission, and it's always practically the same.

The speed of data transfer is affected by the time of serialization and deserialization, for the AMF technology, the more data there is, the time of serialization and deserialization is higher. Serialization is for the loop, which goes through a huge number of objects, transforming them into AMF technology.

For the AMF, a query is made into the database, then the answer from the database is converted into the AMF format and sent to the client (Adamanskiy, 2013). The client reads the AMF format, deserializes the procession of received units and zeros in a format that can be visualized.

The server-side serialization time depends on the number of objects, and if there are more objects to be serialized, then it takes more time. Also, having more data than the file in AMF is heavier and more time is needed for data transport.

On 100 data, within AMF, the average loading time is 0.21s, and on 1000 data it is 0.31s. This means that data transport plays a major role.

Different resources used in the application can influence the choice of how data is packaged between the server and the client.

Comparative data of the data transfer rate, from the server to the client, which incurs the time of serialization, deserialization, the connecting time of the connection is given in the following table. Namely, numerical data in the table were obtained because of a test in the client application and the test on the site http://www.baguetteamf.com/ for the same type of objects. The test was carried out on 1000 objects, where the problem is the time of serialization, thus, the packaging time of the objects.

The number of objects on the test, with and without Baguette AMF, can be changed from 1 to 3,000 objects. The measured values in the test, expressed in milliseconds (ms), are presented in a table with a step 100, in the range of 100 to 2000 objects. The test analyzes three periods, the period of serialization, the period of deserialization and the period of establishing a connection between the server and the client.

The connection period is approximately the same in either case. The deserialization time, as seen in Table 1. and Table 2., is slightly different in either case. From the table, it is noticeable that the period of data serialization is drastically different with and without Baguette AMF. The shorter serialization period with the Baguette AMF is an AMF acceleration.

Table 1. The period of serialization, deserialization and connection establishment, with and without the addition of the Baguette AMF on the Amfphp component, in the range of 100 to 1000 objects loaded on the author's application.

N	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**	**900**	**1000**
Serialization without Baguette AMF	9 ms	18 ms	40 ms	61 ms	79 ms	90 ms	151 ms	178 ms	208 ms	286 ms
Serialization with Baguette AMF	0 ms	0.8 ms	0.8 ms	1 ms	2 ms	2 ms	1 ms	3 ms	4 ms	4 ms
Deserialization without Baguette AMF	0ms	1ms	/	shorter	shorter	same	same	same	shorter	shorter
Deserialization with Baguette AMF	0ms	1ms	/	longer	longer	same	same	same	longer	longer
Connection period without Baguette AMF	10ms	8ms	10ms	11 ms	11 ms	8 ms	11 ms	10 ms	10 ms	11 ms
Connection period with Baguette AMF	6ms	10ms	6ms	10 ms	11 ms	7 ms	7 ms	11 ms	11 ms	11 ms

Table 2. The serialization, deserialization and connection period, with and without the Baguette AMF add-on to the AMF-PHP component, within the range of 100 to 1000 loaded objects. The test was made on the site http://www.baguetteamf.com/.

N	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**	**900**	**1000**
Serialization without Baguette AMF	10 ms	20 ms	41 ms	63 ms	82 ms	92 ms	154 ms	181 ms	210 ms	287 ms
Serialization with Baguette AMF	0 ms	1 ms	1 ms	2 ms	3 ms	3 ms	2 ms	4 ms	5 ms	5 ms
Deserialization without Baguette AMF	1ms	1ms	/	shorter	shorter	same	same	same	shorter	shorter
Deserialization with Baguette AMF	1ms	1ms	/	longer	longer	same	same	same	longer	longer
Connection period without Baguette AMF	11ms	10ms	11ms	12 ms	12 ms	10 ms	12 ms	11 ms	11 ms	13 ms
Connection period with Baguette AMF	8ms	11ms	7ms	11 ms	12 ms	9 ms	8 ms	13 ms	13 ms	12 ms

Table 3: The period of serialization, deserialization and connection establishment, with and without the addition of the Baguette AMF to the AMF-PHP component, in the range of 1100 to 2000 objects loaded on the author's application.

N	**1100**	**1200**	**1300**	**1400**	**1500**	**1600**	**1700**	**1800**	**1900**	**2000**
Serialization without Baguette AMF	340 ms	408 ms	455 ms	549 ms	630 ms	681 ms	760 ms	820 ms	890 ms	972 ms
Serialization with Baguette AMF	5 ms	5 ms	5 ms	6 ms	6 ms	7 ms	7 ms	9 ms	9 ms	9 ms
Deserialization without Baguette AMF	Longer	Approximately the same-short	Approximately the same-short	Approximately the same	Approximately the same	Approximately the same	/	/	/	/
Deserialization with Baguette AMF	Shorter	Approximately the same-short	Approximately the same-short	Approximately the same	Approximately the same	Approximately the same	/	/	/	/
Connection period without Baguette AMF	11 ms	Longer	Shorter slightly	Longer slightly	Longer slightly	Shorter slightly	Shorter Slightly	Shorter Slightly	Shorter slightly	Approximately the same
Connection period with Baguette AMF	Shorter	Shorter	Longer slightly	Shorter slightly	Shorter slightly	Longer slightly	Shorter slightly	Longer slightly	Longer slightly	Approximately the same

Table 4. The serialization, deserialization and connection period, with and without the Baguette AMF add-on to the AMF-PHP component, ranging from 1100 to 2000 loaded objects on the site http://www.baguetteamf.com/

N	**1100**	**1200**	**1300**	**1400**	**1500**	**1600**	**1700**	**1800**	**1900**	**2000**
Serialization without Baguette AMF	344 ms	411 ms	459 ms	555 ms	634 ms	687 ms	763 ms	824 ms	893 ms	976 ms
Serialization with Baguette AMF	6 ms	6 ms	6 ms	7 ms	7 ms	8 ms	9 ms	10 ms	10 ms	10 ms
Deserialization without Baguette AMF	Longer	Approximately the same-short	Approximately the same-short	Approximately the same	Approximately the same	Approximately the same	/	/	/	/
Deserialization with Baguette AMF	Shorter	Approximately the same-short	Approximately the same-short	Approximately the same	Approximately the same	Approximately the same	/	/	/	/
Connection period without Baguette AMF	13 ms	Longer	Shorter slightly	Longer slightly	Longer slightly	Shorter slightly	Longer Slightly	Shorter slightly	Shorter slightly	Approximately the same
Connection period with Baguette AMF	Shorter	Shorter	Longer slightly	Shorter slightly	Shorter slightly	Longer slightly	Shorter Slightly	Longer slightly	Longer Slightly	Approximately the same

6 CONCLUSION

Comparing the speeds of individual steps in transferring the same types of data in the authoring application, and at http://www.baguetteamf.com/, it can be concluded that serialization, deserialization, the period of reconfiguration of the connection takes place over a shorter time interval in the author's application.

The choice of technology depends on several factors, one of the factors is the amount of data. If a large amount of data is used, over 2000, then AMF technology is recommended, the blue variant from graphics. When it comes to the complexity of a code, the AMF technology is suitable.

WORKS CITED

Adamanskiy, A., & Denisov, A. (2013). *EJDB - Embedded JSON database engine*. Fourth World Congress on Software Engineering (WCSE), pp. 161-164, DOI: 10.1109/WCSE.2013.29.

Florescu, D., & Fourny, G. (2013). JSONiq: The History of a Query Language. *IEEE internet computing, 17*(5), 86-90.

Hall, C. (2011). *ActionScript Developer's Guide to PureMVC*. O'Reilly Media.

Jorstad, I., Bakken, E., & Johansen, T.A. (2008). *Performance evaluation of JSON and XML for data exchange in mobile services.* WINSYS 2008: proceedings of the international conference on wireless information networks and systems, pp. 237-240.

Jun Y., Zhishu L., & Yanyan M. (2008). *JSON Based Decentralized SSO Security Architecture in E-Commerce*. International Symposium on Electronic Commerce and Security Location - Proceedings of the International Symposium on Electronic Commerce and Security, pp. 471-475.

Merelo-Guervos, J., Castillo, J., & Laredo, J. L. (2008). *Asynchronous Distributed Genetic Algorithms with Javascript and JSON.* Conference: IEEE Congress on Evolutionary Computation Location: Hong Kong - Book Series IEEE Congress on Evolutionary Computation, Vols. 1-8, pp. 1372-1379.

Pettit, J.B, & Marioni, J.C. (2013). BioWeb3D: an online webGL 3D data visualisation tool. *BMC Bioinformatics*, 14, Article Number: 185, DOI: 10.1186/1471-2105-14-185.

Rodrigues, C., Afonso, J., & Tome, P. (2011). Mobile Application Webservice Performance Analysis: Restful Services with JSON and XML. *Enterprise Information Systems, Communications in Computer and Information Science*, 220, 162-169.

Zimmermann, O., Tomlinson, M., & Peuser, S. (2003). *Perspectives on web services: applying SOAP, WSDL, and UDDI to real-world*. Springer

Received for publication: 13.02.2018
Revision received: 30.05.2018
Accepted for publication: 07.06.2018

How to cite this article?

Style – **APA** *Sixth Edition:*

Sabanovic, M., Saracevic, M., & Dzemic, K. (2018, July 15). Comparative Analysis of Data Transfer Technology Between Server and Client: AMF and Objects on the Website. (Z. Cekerevac, Ed.) *MEST Journal, 6*(2), 117-127. doi:10.12709/mest.06.06.02.15

Style – **Chicago** *Sixteenth Edition:*

Sabanovic, Munir, Muzafer Saracevic, and Kemal Dzemic. 2018. "Comparative Analysis of Data Transfer Technology Between Server and Client: AMF and Objects on the Website." Edited by Zoran Cekerevac. *MEST Journal* (MESTE) 6 (2): 117-127. doi:10.12709/mest.06.06.02.15.

Style – **GOST** *Name Sort:*

Sabanovic Munir, Saracevic Muzafer and Dzemic Kemal Comparative Analysis of Data Transfer Technology Between Server and Client: AMF and Objects on the Website [Journal] // MEST Journal / ed. Cekerevac Zoran. - Toronto : MESTE, July 15, 2018. - 2 : Vol. 6. - pp. 117-127.

Style – **Harvard** *Anglia:*

Sabanovic, M., Saracevic, M. & Dzemic, K., 2018. Comparative Analysis of Data Transfer Technology Between Server and Client: AMF and Objects on the Website. *MEST Journal,* 15 July, 6(2), pp. 117-127.

Style – **ISO 690** *Numerical Reference:*

Comparative Analysis of Data Transfer Technology Between Server and Client: AMF and Objects on the Website. **Sabanovic, Munir, Saracevic, Muzafer and Dzemic, Kemal.** [ed.] Zoran Cekerevac. 2, Toronto : MESTE, July 15, 2018, MEST Journal, Vol. 6, pp. 117-127.