ISSN 2303-4521

PERIODICALS OF ENGINEERING AND NATURAL SCIENCES
Vol. 4 No. 2 (2016)
Available online at: http://pen.ius.edu.ba

Comparative analysis of AMF, JSON and XML technologies for data transfer between the server and the client

Munir Šabanović
University of "Džemal Bijedić"
Faculty of Information Technologies
Sjeverni Logor br.12 88000 Mostar, BiH
munir.sabanovic@uinp.edu.rs

Muzafer Saračević
Department of Computer sciences,
University of Novi Pazar
Dimitrija Tucovica bb, Novi Pazar, Serbia
muzafers@uninp.edu.rs

Emruš Azizović
Faculty of Computer sciences
University of Prizren
Rruga e Shkronjave, nr. 1, Prizren, Kosovo
emrus.azizovic@unhz.eu

Abstract

This paper presents a comparative analysis of data formatting technology in AMF, JSON and XML, during data transfer between client and server. Data is authoring application that allows comparative analysis of these three technologies. These include aspects of the data transfer speed, the size of the output file, the data transmission safety, the code complexity of both the client and the server.

Keywords: AMF; JSON; XML; client-server communication; data transfer;

1. Introduction

The paper presents an authoring application, which allows measuring the time of data transfer from the moment of sending a request to the server, up to the moment of presenting data to the user. The measurement is realized for three data packaging technologies AMF, JSON and XML. The number of data from the base is chosen by the user in combo box, ranging from 1 to 76000. Time required to transfer data from the server to the client can be divided into several stages. Each phase is analyzed in detail and it is determined which of them introduces a delay during the data transfer. On the difference in time of data transfer, in addition to transfer technology, also affects the size of the output file, the complexity of the code of both the client and the server. The paper analyzes the aspect of data transmission security. At the end are given recommendations when and which technology to use.

2. Used scientific methods and procedures

In order to achieve the objectives and tasks of the research the following scientific methods and procedures were used:

- By comparative methods were compared results of the existing methods and new proposed methods of problems solving.
- By experimental application we have got results that demonstrate the effectiveness of new methods (in speed or saving memory space).
- The method of generalization is applied in the analysis of a number of cases where there is a general statement that applies to all the cases. The method of specialization presents specific cases as a specific example.
- Methods of deduction and induction are used in the course of experimental tests, where after the obtained results a conclusion is formed about the new techniques and the possibilities of their application.

3. Technologies used for making authoring application

For creating software solutions Flex technology was used on the client side, PHP on the server side, while, for data transmission were used three technologies AMF, JSON and XML. JSON and XML are used in many areas [6,7,11,12].

Flex is a highly productive, free open source software environment for creating executable version of expressive mobile, web and computer applications. Flex enables creating executable Web version and mobile applications that share a common basic code, thus shortening the time and cost of creating applications and long-term maintenance [1].

While Flex applications can only be created using the free Flex SDK, the Adobe Flash Builder ™ software can accelerate development with features such as intelligent code editing, gradual debugging, programs for memory optimization and performance as well as visual design.

DOI: 10.21533/pen.v4i2.57

Flex in its environment contains two MXML languages for visualization and AS3 for functionality [2].

Server PHP language, is executed on the server side. PHP is called a scripting language because it is written in the form of scripts and was purpose built for use on the Web [7]. PHP can be written in separate files, and can be inserted within HTML.

The PHP processor on the Web server interprets the PHP code, and the Web server on exit emits HTML or other types of data that the client web browser can understand. 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 is before that translated into a form that the client computer will know how to interpret. HTML parts in the script are left as they are until PHP code interprets the PHP processor and executes, and the result of execution is sent to the client.

PHP code has great possibilities, of communicating with other computers, creating images, access to databases, work with graphics, creating desktop applications using PHP-GTK extensions all the way up to reading and writing files. PHP does not have its owner, it is a free language, a group of enthusiasts gathered together and made the PHP. For AMF technology, they took the Action Message Format and made serializators that correspond to action message format in PHP.

AMF or Action Message Format, is a binary format that is used for serialization of objects and sending messages between the client and the remote service. Action Script 3 language has classes for encoding and decoding of the AMF format. Adobe Systems has released AMF binary protocol specification on December 2007, and announced that it will support the developer community in making this protocol for all major server platforms. So today there is AMF support for platforms written in Java, PHP, .NET and other languages.

This paper will focus its attention on the PHP language, because it is the chosen server platform. JSON or JavaScript Object Notation is a very simple text format for data exchange between the server and the client [6, 10]. It is easy to parse and independent from any programming language. Parsing JSON format can be performed using the built-in JavaScript functions [11,12]. Compared to XML format, it is faster, shorter, there are no reserved words [8, 9].

XML, or Extensible Markup Language, is a language for creating electronic documents. One XML document is a hierarchy of XML elements. Each element represents part of the information contained in the document [3]. The content of the XML document includes: processing instructions, elements, attributes and comments.

4.Design and application functionality

Applications' operating environment contains a grid, with columns id, First name, Last name, department, index no. and Date. Figure 1. shows the applications' operating environment.

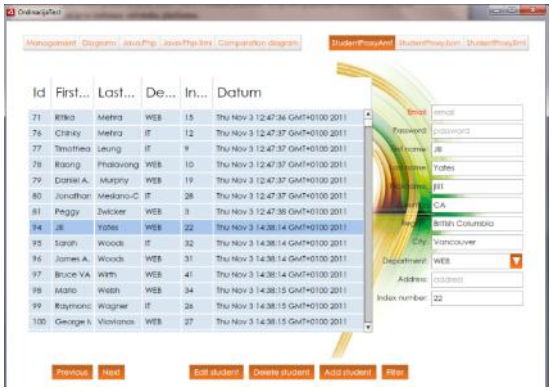

Figure 1. Applications' operating environment.

The columns are loaded data from the database, which is located on the server Wampserver2.4-x86. Number of loaded data is selected from the *ComboBox*, the range 1-76000 data.

Id	First Name	Last Name	Department	Index no.	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	Meslano-Crookston	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

Figure 2. DataGrid component on applications' workplace.

The application contains two Bar buttons. The first Bar button, *stateSelector* contains five states: *Managament, Diagrams, Java-PHP, Java-PHP-XML and Comparation* diagram. This control is defined by block code [3]:

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

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

```
<s:ArrayList id="statesProvider" >
<fx:Object label="Managament" state=
"managament"/>
<fx:Object label="Diagrams" state="diagrams"/>
<fx:Object label="Java-Php" state="javaPhp"/>
<fx:Object label="Java-Php-Xml" state=
"javaPhpXml"/>
<fx:Object label="Comparation diagram"
state="comparation"/>
</s:ArrayList>
```


DOI: 10.21533/pen.v4i2.57

The authoring application loads the data from two different servers, Wampserver2.4-x86, which works with PHP and Niti server, which works with Java. If the active state is Managament, then the work environment of the application is displayed. However, if the second state Diagrams is active, then are displayed graphs for three technologies of data transmission, AMF, JSON and XML. The state JavaPhp shows JSON charts, when there is client authoring application communication with the Wampserver2.4-x86 server and Niti server. If the fourth state JavaPhpXml is active, then graphs are displayed which measure the time of n data, when the data on the part of the server is formatted in XML. If the Comparation diagram button is active, then a comparison chart of AMF-JSON-XML query execution is displayed.

The second Bar button selectProxyBar, changes its appearance depending on the application's state. The state Management, displays three buttons `StudentProxyAmf`, `StudentProxyJson`and `StudentProxyXml`.

These three buttons offer choice of technology of data transfer from the database to the application in AMFU, JSON or XML.

Button controls, whose labels are Previous, Next, Edit student, Delete student, Add student and Filter allow, to be respectively displayed from the base the previous block of data in the grid, the next data block, edit the selected item in the grid and the database, delete rows in base, add new studentin the database, and filter data in the grid. Depending whether the state inclass `instancesStudentControlComponent` and `StudentManagerComponent` is normal or search, the filter control can take two values for label Filter and Back. TextInput fields and ComboBoxare used to show items from selected grid rows, changes to the content in the selected rows and adding new content.

In the state Normal, all the TextInput controls and ComboBox are shown, while in the state search are not displayed all TextInput fields.

5.Experimental results and comparative analysis

The application measures the time of data transfer from the server to the client for three technologies of data formatting, AMF, JSON and XML. Execution time requires measurement from the moment of sending a request to the server from client's part, until the time the data is downloaded from the server in the Flex application. JSON, XML, AMF are protocols used for communication between the client and server. Between them there is a difference in data packing speed, the manner of data formatting, the complexity of creating applications, the size of the file that is being transported, as well as the complexity in terms of parsing data from one format to another.

AMF works with binary data, on the server's side it translates a language into binary data, while on the client's side the inverse process is performed, that translates zeros and ones into visual code that is displayed within the graphical interface.

In JSON technology, is carried out the translation of PHP objects into JSON objects, and eventually the JSON objects are sent as text file to the client through HTTP connections. The same is done with XML.

Working application uses AMFPHP package for the communication of client's ActionScript3 and server PHP language. In addition to the AMF, ActionScript3also supports XML and JSON technologies, performing data parsing to be transported in XML and JSON format.

During the data transfer between client and server, there are several phases, with each phase requiring certain time. On the client's side (Flex) there is a period for preparing the queries (serialization) and deserialization period, while on the server's side there is deserialzation period, the period query execution in the database, and serialization period. There is also a data transfer period. These periods are shown in Figure 3.

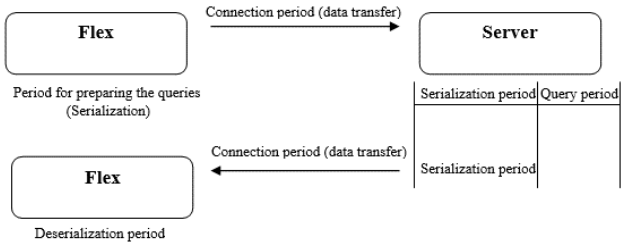

Figure 3. Period display, during data transfer from the server to the client.

Queries are extremely rapid for any technology, there are in microseconds and that period is not taken into account. Deserialization period on the server's side includes converting binary data into objects of some language. Query period is spent on reading data in the database, the data is then serialized, and then transferred to the client. And at the end, the data on the Flex side are deserialized for a certain time. Serialization has a global meaning, namely, in computer sciences, in the context of data storage, the serialization represents a process of translation data structure or objects in a format that can be saved (in the file or buffer memory, or can be sent through the network) and reconstructed, later in the same or different computer environment [5].

The production of AMF has, from the version 2.0, been undertaken by another company, and became considerably slower than JSON and XML. Slowness is due to the serialization, where a lot of time is spent on

DOI: 10.21533/pen.v4i2.57

converting objects of a language into binary data, which can be seen in Figure 4. This problem is solved by adding a plug-in, Baguette AMF, to the basic version of AMFPHP.

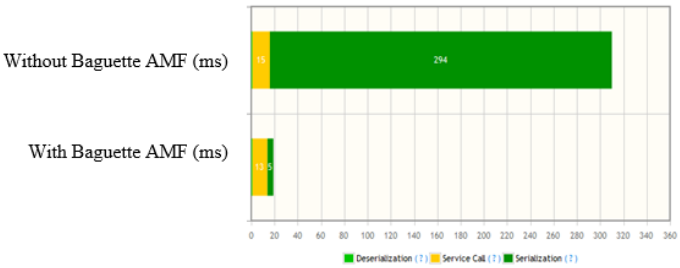

Figure 4. Period of serialization, deserialization and establishing a connection, when add-on Baguette AMF not in use and when it is used.

The application analyzes three important periods, the period of serialization, the period of deserialization on the client and server sides, the period of establishing a connection between the server and the client. The middle line defines the execution period of the service call.

The period of establishing a connection with the AMF technology is approximately the same with and without the Baguette AMF add-on. The deserialization periods are slightly different in both cases. Figure 4. shows that data serialization period drastically varies with and without Baguette AMF. The shorter serialization with Baguette AMF presents acceleration of the AMF.

Adding Baguette AMF plug-in to the basic version of AMF PHP, makes the AMF faster than JSON and XML when transporting larger amounts of data thus providing the Baguette AMF. Baguette AMF with AMFPHP is recommended for enterprise applications which share a large amount of data.

This paper translates PHP objects in AMF, JSON or XML format which is readable to the Flex client. Flex can serialize data in AMF, JSON or XML format and deserializate data from these formats. Figure 5. shows the communication between client and server where PHP is used as server language and serialization and deserialization of data is done with Baguette AMF.

The Client, within itself, has a built-in AMF format, which can be seen in Figure 5. AMF, XML and JSON technologies are widely used. They belong to cross technologies which means they do not depend on the operating system nor from the computer environment.

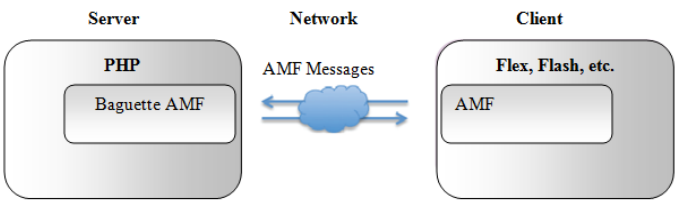

Figure 5. Flex – PHP communication

In the application, when the component Diagrams is active (see Figure 1), the charts for AMF, JSON and XML technology are shown. Comparison graph of data transport speed from the server to the client, which covers serialization period, deserialization, the period of establishing connection, is shown in Figure 6. In ComboBox is performed selection of data retrieved from the database on the server into the client application.

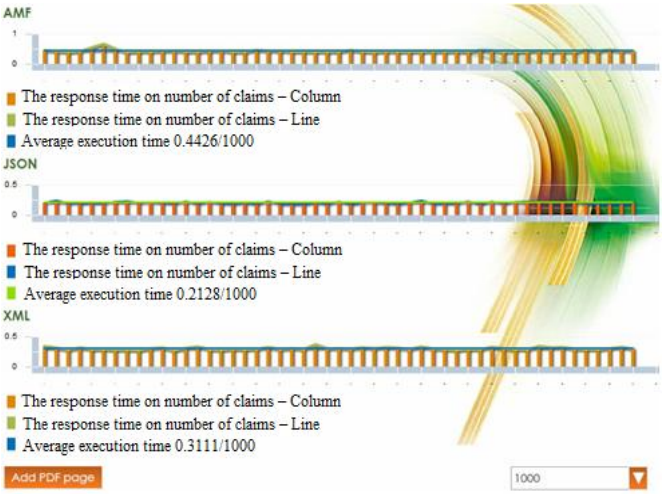

Figure 6. Graphs of time dependency from the number of loaded data for AMF, JSON and XML-technology

The choice of transfer technology is conducted by the selectProxyBar button bar. For each technology, fifty loading iterations are performed where in each iteration the same number of data is loaded, the time is measured, and at the end of the iteration the mean time is calculated.

Testing is performed on computer with the following performances: *CPU - Quad Core AMD Phenom X4 9550, 2200 MHz (11 x 200), L1 64 KB per core, 512KB L2 per core (On-Die, ECC, Full-Speed), 2 MB L3 (On-Die, ECC, NB-Speed), RAM Memory - 3 Gb, Graphic card - NVIDIA GeForce 9400 GT*. The environment of authoring application was used for the test.

On the graph each column presents the data loading period. Horizontal fault line connects the tops of columns and the right horizontal line in the chart describes the mean loading data period. The graphs represent time when 1000 data from the database are loaded.

On large number of data, AMF is faster than JSON. In Figure 7., on the basis of measurements in this model, is presented comparison diagram of dependency between the time and the number of data. Figure 7. shows, that up to 2000 data JSON is faster than AMF and above this number, AMF becomes faster. Therefore, it is recommended to use AMF technology above this number of loaded data. The diagram shows time dependency of the number of data downloaded, with JSON and XML it is much higher compared to the AMF. The Curve which describes the AMF technology, the third line, is of

DOI: 10.21533/pen.v4i2.57

considerably smaller angle, while with JSON, the first line is substantially steeper which means that the time is increasing faster with the number of loaded data. The middle line represents the XML format where we can see that the XML much more depends on JSON when the number of transmitted data increases.

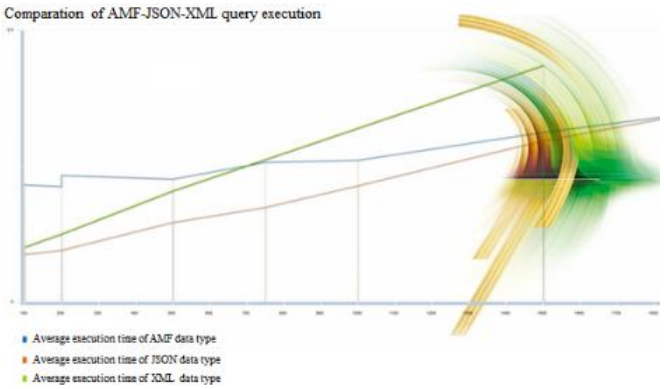

Figure 7. - Comparison diagram of the data loading speed for AMF, JSON and XML technologies, for range from 100 to 2000 data and the step 100.

AMF does not work with HTTP connection. The advantage of AMF is opening of a socket connection, through which the client is forwarded the binary data. In Flex, in AMF technology, 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 sending timing and receiving data.

In AMF, the period of data transfer does not depend on the amount of data, because it is binary transmission and is always practically the same.

Get SyntaxView | Transformer | Headers | TextView | ImageView | HexView | WebView

```
HTTP/1.1 200 OK
Date: Sun, 16 Nov 2014 14:18:35 GMT
Server: Apache/2.4.4 (Win32) PHP/5.4.16
X-Powered-By: PHP/5.4.16
Access-Control-Allow-Origin: *
Content-Length: 2802
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8

{"result":
[{"id":"71","email":"mca.ptu@gmail.com","password":"","first_name":"Riti
try":"CA","region":"Ontario","city":"Toronto","department":"1","address"
12:47:36","updated_at":"2013-01-27 07:49:59","index_num":"15"},{"id":"7
@gmail.com","password":"","first_name":"Chinky","last_name":"Mehra","nic
city":"Toronto","department":"2","address":"","created_at":"2011-11-03 1
18:16:39","index_num":"12"},
{"id":"77","email":"","password":"","first_name":"Timothea","last_name":
on":"Ontario","city":"Burlington","department":"2","address":"","created
25 08:50:19","index_num":"9"},
{"id":"78","email":"","password":"","first_name":"Raong","last_name":"Ph
":"Ontario","city":"Burlington","department":"1","address":"","created_a
05:18:45","index_num":"10"},{"id":"79","email":"","password":"","first_n
Murphy","nickname":"TLS","country":"CA","region":"","city":"Santa
Barbara","department":"1","address":"","created_at":"2011-11-03 12:47:37
10:39:39","index_num":"19"},{"id":"80","email":"","password":"","first_n
Crookston","nickname":"mar9usr","country":"CA","region":"Ontario","city"
at":"2011-11-03 12:47:37","updated_at":"2014-02-19 16:21:24","index_num"
{"id":"81","email":"","password":"","first_name":"Peggy","last_name":"Zw
"Nova Scotia","city":"Liverpool","department":"1","address":"","created_
06:05:44","index_num":"3"},
{"id":"94","email":"","password":"","first_name":"Jill","last_name":"Yat
itish Columbia","city":"Vancouver","department":"1","address":"","create
27 17:14:51","index_num":"22"},
{"id":"95","email":"","password":"","first_name":"Sarah","last_name":"Wo
Quebec","city":"Montreal","department":"2","address":"","created_at":"20
13:56:55","index_num":"32"},{"id":"96","email":"","password":"","first_n
A.","last_name":"Woods","nickname":"james
a.1","country":"CA","region":"Quebec","city":"Montreal","department":"1"
14:38:14","updated_at":"2014-02-08 08:19:43","index_num":"31"}],"status"
```

Figure 8. Structure of JSON file which is loaded into the client Flex application

In Figure 8. is displayed the contents of the JSON file when 10 data from the server are loaded in Flex application.

In JSON and XML the connection period depends on the amount of data, the data is packed in a file in text, and the time of the connection depends on the size of the file. Therefore, the more data, the connection period increases with JSON and XML except that the connection period with XML is bigger than with JSON for the same amount of data because of the structure of XML resulting in increasing the size of the output file and thus higher data transfer period.

Figure 9. displays the contents of an XML file when 10 data from the server are loaded to the client's application. Comparing the same content, in Figures 8 and 9, it can be concluded that XML file is more complex than JSON file, it has larger capacity and if both are legible.

Get SyntaxView | Transformer | Headers | TextView | ImageView | HexView | WebView | Auth

```
HTTP/1.1 200 OK
Date: Sun, 16 Nov 2014 14:29:46 GMT
Server: Apache/2.4.4 (Win32) PHP/5.4.16
X-Powered-By: PHP/5.4.16
Access-Control-Allow-Origin: *
Content-Length: 5151
Keep-Alive: timeout=5, max=99
Connection: Keep-Alive
Content-Type: text/xml; charset=utf-8

<root><result><student><id><![CDATA[71]]></id><email><![CDATA[mca.ptu@gmail.cor
</first_name><last_name><![CDATA[Mehra]]></last_name><nickname><![CDATA[hotski
</country><region><![CDATA[Ontario]]></region><city><![CDATA[Toronto]]></city>
<address><![CDATA[]]></address><created_at><![CDATA[Ontario]]></created_at><upd
</updated_at><index_num><![CDATA[15]]></index_num></student><student><id><![CDA
@gmail.com]]></email><first_name><![CDATA[Chinky]]></first_name><last_name><![C
[Barry]]></nickname><country><![CDATA[CA]]></country><region><![CDATA[Ontario]]
<department><![CDATA[2]]></department><address><![CDATA[]]></address><created_
<updated_at><![CDATA[2013-07-21 18:16:39]]></updated_at><index_num><![CDATA[12
[CDATA[77]]></id><email><![CDATA[]]></email><first_name><![CDATA[Timothea]]></f
</last_name><nickname><![CDATA[Timothea]]></nickname><country><![CDATA[CA]]></c
<city><![CDATA[Burlington]]></city><department><![CDATA[2]]></department><addre
[CDATA[Ontario]]></created_at><updated_at><![CDATA[2013-01-25 08:50:19]]></upda
</student><student><id><![CDATA[78]]></id><email><![CDATA[]]></email><first_nam
<![CDATA[Phalavong]]></last_name><nickname><![CDATA[Raong]]></nickname><country
[Ontario]]></region><city><![CDATA[Burlington]]></city><department><![CDATA[1]]
```

Figure 9. XML file structure which is loaded into the client Flex application

Therefore, for the transfer of XML files, due to higher capacity, more time is needed, and due to the complexity of the XML structure, on the server and client side are used more complex classes to handle such structure in relation to JSON and AMF content all of which increases during serialization and deserialization.Therefore, if it bears the same data amount, because of the very structure of XML, the XML file is bigger than JSON. In Table 1. are given data on the size of the output file depending on the number of loaded data from the database for JSON and XML file. Data were measured while loading the application.

Based on the sample from Table 1., from 10 to 100 loaded data, the difference in file size between JSON and XML file is approximately 84%. Therefore, the XML file is 84% larger than the JSON file and thus the transfer of the same amount of data is shorter in JSON format.

DOI: 10.21533/pen.v4i2.57

Table 1. List of values for the file size in JSON and XML format, for N loaded data

N	10	20	30	40	50
A	2,73	5,41	8,01	10,44	13,05
B	5,03	9,97	14,75	19,36	24,19
C	84	84	84	85,3	85,4
N	60	70	80	90	100
A	15,7	18,36	21,16	23,97	26,43
B	29,03	33,88	38,89	43,9	48,54
C	84,9	84,5	83,8	83,1	83,6

A-Size of JSON file(Kb)
B-Size of XML file(Kb)
C- JSON file is bigger than XML file for (%)

On the side of the server and the client, the structure of XML data is more complex and such a structure is difficult to treat because it has more nodes. So, if there is a complex structure of the object, such as XML, it is more difficult to treat in any language because it has to go through more nodes, through greater depth. Therefore, when it comes to serialization and deserialization, there is no difference between languages. Each language will process faster the less complex data structure. In this model, in PHP, it is easier to process JSON objects from XML objects. PHP functions for parsing the XML object are more complex than when processing simpler object.

The data speed transfer is affected by the time serialization and deserialization, for any technology, the more data there is, the serialization and deserialization period is greater. Serialization is for the loop, which passes through a huge number of objects turning them into technology. For all three technologies the procedure is the same. For JSON there is inquiry into the base, and then the response from the database is converted into JSON and sent to the client. The client reads JSON, converts it into objects and continues to execute the command. The same applies to XML and AMF.

The serialization period on the server's side depends on the number of objects and if there are more objects to be serialized, then more time is spent which is especially emphasized in JSON and XML because the complexity is greater. Also the more data there is then the file in JSON and XML are heavier and more time is needed to transport data. Hence, the time difference is mostly reflected in the time required to transport data in AMF, JSON and XML and increases with the size of the file.

With 100 data in AMF, the average load time is 0,21s and with 1000 data it is 0,31s. In JSON, with 100 data it is 0,08s and with 1000 it is 0,30s, the difference being 3.62 times. This means that data transport has a major role. Various resources which are used in the application, can affect the choice of packaging data mode between the server and the client. For example, if a working application is using more services which use XML, and it is necessary for a service to be written, then it is logical to choose XML in order to have everywhere the same method of preparation and processing and thus to use the same class in the entire application for faster code writing and code consistency. Therefore, if one is using multiple data sources then it is up on the developer to create a single source. If they all use single technology, such as XML, then it is the developer's decision to choose the language in the part he deals with and that is XML because it will thus adapt the service made with other services in order to use class parts from other services and enable to assemble the two objects. Since if one object is in XML and the other in JSON, then the two objects cannot be assembled. In this case, both objects would have to be converted into a readable part in order to connect, which would slow data load. But if both objects are in XML or JSON, then they can be assembled because they are naturally compatible, there are methods for them, and that is one of the advantages that can be used.

When it comes to data security, SSL or encrypted communication mode can be used. The HTTP protocol is not safe when technology transfer JSON, XML or web services are used. The HTTP protocol can be used where data security is not that important because the content of JSON and XML file is readable (Figure 8.). SSL is a protocol for encrypting traffic. Here the traffic between points A and B encrypts so the content that is sent is not seen. Another manner is authorization of the services which can be made or bought and used. Authorization is function specifying the access rights to some resources associated with data protection. For authorization, the standard OAuth can be used which belongs to the open standard for authorization. OAuth provides the client application with a secure access to the resources allocated to the server on behalf of the resources' owner. This means, the client sends the server a request whether he could see his data.

The server responds so that it gives an affirmative answer, in terms of number or security code which the user will use through each subsequent request. That number in each query addresses the server as the data owner and that the server has allowed access to the client data. And each time the client receives a different password when he logs in. In general, the client is registered on the server as someone who wants to access server data. The server usually gives the client three data. The first data is the ID number of the application.

Therefore, the authoring application is registered on the server, as the owner of the data, and requires from the user that the application accesses specific data and then the server assigns ID to the authoring application when in the database the application ID is written down and as

DOI: 10.21533/pen.v4i2.57

the owner of the application the client is issued two data, a username and password. Now the client has three data, one is the application ID, the second is a username and password is the third. These three data are not used to access data, but another service is called for, which need not be on the same server where the data is.

The client calls for the service, passing the three data whereby following data verification, whether the application ID for that username is correct and whether the code is correct. If it is, the service generates one random number which is returned to the client in response authentication. At the same time in the database with each data is entered that same client's authorization number or it is placed in an authorization table. Figure 10. shows the schematic display of the authorization.

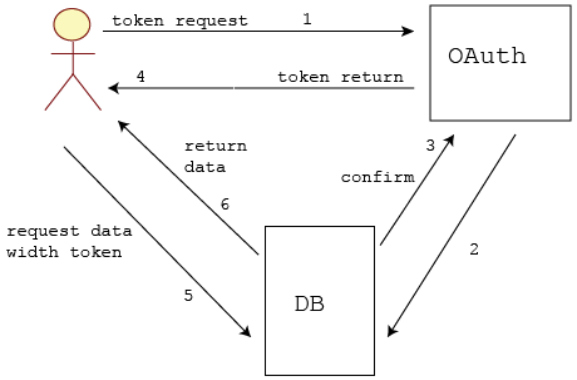

Figure 10. Access to the database with authorization

Following line 1, the user requires a key to the OAuth system. This system generates the key and following line 2 it is entered into the database DB. The DB basis following line 3, gives affirmative response that key was added. Following line 4, the key is sent to the client, that forwards it to the DB basis following line 5 while data requirement. Following line 6, the data is delivered to the client.

It is best to use both methods for data protection, authentication and encoding, with the SSL service. There are two ways of authorization: a time-limited and indefinite authorization manner. With time-limited authorization, the authorization service, issued the client permission to access data on the server for a limited time. After the timeout, the license to the client is terminated. Either key is given to access the data for a period of time, which is calculated from the last access to the data on the server. This is typically done and on the web it is called session. This means, while the Client is active it gets unlimited time key. The time was measured from data requests, when a new request is sent it resets the time. When the time period expires, from the last request the system deletes the key from the authorization table. Facebook uses these things.

The things for authorization are linked to the http request, for JSON, XML, and Web services, since Web services are practically XML. The first call towards the server must be OAuth, the request for authorization.

6. Conclusions and Future Work

The choice of technology depends on several factors, one of them is the amount of data. By measuring data transfer period in the authoring application it is determined, for a small amount of data, up to 2000, JSON technology is recommended because it is faster, and it is closest to the object structure, orange variant on the graphics. When large amounts of data are used, up to 2000, AMF technology is recommended, blue variant on the graphics.

If using multiple sources that have been created in one technology, in XML, then it is recommended to the developer when creating the service to use the same technology, for the sake of simplicity, speed and consistency of code writing. When it comes to the complexity of the code, AMF technology is recommended, JSON and XML in the end. If it comes to the size of the output file, based on the measurements, JSON file is smaller for 84% of the XML file, and JSON file is recommended. If there is transfer of large data amounts, but segmentation is done, per segment is not transmitted more than 2000 data and then the recommendation is to use JSON technologies.

If it is a heterogeneous structure, which has support for XML, then it is advisable to use an XML format for communication between software and hardware components that communicate with each other. If data security is not important, then you can use the HTTP protocol for JSON, XML and Web services. If a more secure way of data transfer is important, then HTTPS connection is recommended.

7. References

[1] Using ADOBE FLEX 4.6. "Tutorial from official web site" help.adobe.com/ (visited 19.11.2014.).

[2] Zimmermann O., Tomlinson M., Peuser S. "Perspectives on web services: applying SOAP, WSDL and UDDI to real-world", *Springer*, 2003.

[3] Hall C., "ActionScript Developer's Guide toPureMVC", *O'Reilly Media*, 2011.

[4] C++ FAQ. "Kept by Marshall Cline", retrieved from www.parashift.com/c++-faq-lite/ (visited 19.11.2014.)

[5] Encryption Basics. "EFF Surveillance Self-Defense Project, Surveillance Self-Defense Project, n.d. Web". ssd.eff.org/tech/encryption (visited 19.11.2014.)

DOI: 10.21533/pen.v4i2.57

[6] Florescu D, Fourny G. “JSONiq: The History of a Query Language”, *IEEE internet computing*, vol. 17, no. 5, pp. 86-90, 2013.

[7] Pettit, Jb, Marioni, Jc. “BioWeb3D: an online webGL 3D data visualisation tool”, *BMC bioinformatics*, vol. 14, no. 1, pp. 185, 2013.

[8] Jorstad, I, Bakken, E, Johansen, Ta, “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, 2008.

[9] Rodrigues, C, Afonso, J, Tome, P, “Mobile Application Webservice Performance Analysis: Restful Services with JSON and XML, enterprise information systems”, *Communications in Computer and Information Science*, vol. 220, no. 1, pp. 162-169, 2011.

[10] Adamanskiy A, Denisov A. “Embedded JSON database engine”, *Fourth world congress on software engineering (WCSE)*, pp. 161-164, 2013.

[11] Merelo-Guervos J, Castillo J, Laredo A. et al.. “Asynchronous Distributed Genetic Algorithms with Javascript and JSON”, *Conference: IEEE Congress on Evolutionary Computation Location: Hong Kong*, pp. 1372-1379, 2008.

[12] Jun Y, Zhishu L, Yanyan M. JSON Based “Decentralized SSO Security Architecture in E-Commerce, International Symposium on Electronic Commerce and Security”, *Proceedings of the international symposium on electronic commerce and security*, pp. 471-475, 2008.

[13] http://www.tricedesigns.com/2011/11/07/AMF-vs-JSON-in-air-mobile-applications/ (visited 25.11.2014)

[14] http://www.jamesward.com/2007/04/30/ajax-and-flex-data-loading-benchmarks/ (visited 25.11.2014)

DOI: 10.21533/pen.v4i2.57