IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 6, Ver. V (Nov – Dec. 2015), PP 01-04
www.iosrjournals.org

Q-JSON - Reduced JSON schema with high Data Representation Efficiency

Pratik Tyagi[1]
[1](Bangalore,India)

Abstract: *Mobility has already become the mainstream interface requirement for applications and the end users; this is forcing all applications/companies to adopt Web services [1] based architecture. But supporting mobility and the growing demand for better user experience comes with technical constraints such as memory space and lower data transfer rate. This paper proposes a new reduced JSON schema (Q-JSON) that improves data representation efficiency and thus improving on both constraints without any need of extra encoding/decoding.*
Proposed Q-JSON schema format normalizes the redundant data key information without compromising the actual business relevant information.
Keywords: *JSON Schema, Data representation efficiency, mobility, response delay, web services*

I. Introduction

JSON [2] has achieved widespread adoption as the generic data representation format for fetching data from Web services due to its use as a JavaScript data structure and lightweight data representation over the legacy XML [3].However even with these advantages there are cases when generic schema of object representation using JSON can be further optimized to achieve better DRE(Data Representation Efficiency). Since data fetching operation owns major portion of user action response delay time, reducing the data transfer size in turn reduces the response delay. This paper proposes a reduced JSON (Q-JSON) schema that allows applications to send more business relevant information within the same data size.

This paper documents the proposed Q-JSON schema structure and comparative case study between generic JSON schema and proposed Q-JSON schema.

II. Observation

Applications that present large homogeneous data list in one view to the user such as list of users available in the system, uses server side APIs in Web services to convert data to the generic JSON schema format, e.g. JAX-B [4] to convert java objects to JSON response. These APIs use generic format (1-1 mapping) to convert business data to JSON format i.e. representing a homogeneous list of object as shown in section 2.1.

2.1 Generic JSON representation: Homogenous Single Level List

Two object representation with 4 attributes:-

```
[
        {
        AttributeName1 : AttributeValueA1,
        AttributeName2 : AttributeValueA2,
        AttributeName3 : AttributeValueA3,
        AttributeName4 : AttributeValueA4
        },
        {
        AttributeName1 : AttributeValueB1,
        AttributeName2 : AttributeValueB2,
        AttributeName3 : AttributeValueB3,
        AttributeName4 : AttributeValueB4
        }
]
```

This representation comes with major redundancy in form of repeated key information for each object; this information can be removed without compromising the actual business relevant information and thus improving the DRE in terms of data size.

III. Proposal: Q-JSON schema

Q-JSON uses Array as its base data structure for representing the data. It normalizes the redundant data key information and representing them as a single array once and the individual object values in separate array, this representation provides two advantages: - a) Reduced data size b) Faster iteration due to use of Array data structure.

Q-JSON Syntax:

```
{
   AttributeDefinationList :[<Attr Definition>, <Attr Definition>,…],
   AttributeDataList :[[<Attr Val>, <Attr Val>,…][<Attr Val>, <Attr Val>,…]]
}
```

Q-JSON schema can be used for representing homogenous single level data list as well as for complex homogeneous hierarchical data structures.

3.1 Q-JSON representation: Homogenous single level List

2 object representation with 4 attributes:-

```
{
   AttributeDefinationList: [AttributeName1, AttributeName2, AttributeName3, AttributeName4],
   AttributeDataList: [
                        [AttributeValueA1, AttributeValueA2, AttributeValueA3, AttributeValueA4],
                        [AttributeValueB1, AttributeValueB2, AttributeValueB3, AttributeValueB4]
                      ]
}
```

3.2 Q-JSON representation: Homogenous multi-level structure

Q-JSON schema can be extended to represent multilevel hierarchical data structure as shown below:-

```
{
   AttributeDefinationList: [AttributeName1, AttributeName2, AttributeName3, Children],
   AttributeDataList: [
                        [AttributeValueA1, AttributeValueA2, AttributeValueA3,
                                {
                                   AttributeDefinationList:[Sub-AttributeName1, Sub-AttributeName2],
                                   AttributeDataList:[[Sub-AttrValueA1, Sub-AttrValueA2],
                                                      [Sub-AttrValueB1, Sub-AttrValueB2]]
                                }
                        ],
                        [AttributeValueB1, AttributeValueB2, AttributeValueB3,
                                {
                                   AttributeDefinationList:[Sub-AttributeName1, Sub-AttributeName2],
                                   AttributeDataList:[[Sub-AttrValueC1, Sub-AttrValueC2],
                                                      [Sub-AttrValueD1, Sub-AttrValueD2]]
                                 }
                         ]
                      ]
}
```

IV. Comparative Case study

This case study uses a Web Application/client that request Web services for list of users existing in the application. Below comparative cases take account of resultant data size based comparison b/w the Generic JSON schema and Q-JSON schema.

4.1 Comparative Case study 1:

Sample Data Description
Object Type : USER
Sample object size : 1000, 5000, 15000 and 50000
Object Attribute Key to Value (chars) List:-
1. Name : 15 chars
2. Age : 2 chars
3. Gender : 4 chars

4. Address : 100 Chars

∑ Attribute keys char length: ∑Attribute value char length Ratio = (4+3+6+7) / (15+2+4+100) = 20/121 = 0.16

Table1.Data Representation Result

No. of Objects	Generic JSON format Size(KB)	Q-JSON format Size(KB)
1000	163	132
5000	811	655
15000	2432	1964
50000	8106	6544

Data size reduction = 19% approx.

No. of objects: Generic JSON format data size = 6.1 (min)
No of Objects represented in 2000KB = 2000*6.1 = 12200
No. of objects: Q-JSON format data size = 7.5 (min)
No of Objects represented in 2000KB = 2000*7.5 = 15000

Increase in No. of objects represented using Q-JSON schema (Data size 2000KB) = 2800 or 23%

4.2 Comparative Case study 2

Sample Data Description
Object Type : USER
Sample object size : 1000, 5000, 15000 and 50000
Object Attribute Key to Value (chars) List:-
1. Name : 15 chars
2. Age : 2 chars
3. Gender : 4 chars
4. Address : 100 chars
5. Phone No. : 10 chars
6. Company : 23 chars
7. Designation : 17 chars
8. E-ID : 3 chars
9. Team Name : 10 chars

∑ Attribute keys char length: ∑Attribute value char length Ratio = (4+3+6+7+9+7+11+7+9) / (15+2+4+100+10+23+17+3+10) = 63/184 = 0.34

Table2.Data Representation Result

No. of Objects	Generic JSON format Size(KB)	Q-JSON format Size(KB)
1000	296	208
5000	1475	1036
15000	4425	3106
50000	14747	10352

Data size reduction = 30% approx.

No. of objects: Generic JSON format data size = 3.37 (min)
No of Objects represented in 2000KB = 2000*3.37 = 6740
No. of objects: Q-JSON format data size = 4.80 (min)
No of Objects represented in 2000KB = 2000*4.80 = 9600

Increase in No of Objects represented using Q-JSON schema (Data size 2000KB) = 2860 or 42%

V. Conclusion

Q-JSON drastically improves the DRE (Data Representation Efficiency) for linear/hierarchical homogenous data without any need of encoding. This Increase in DRE directly correlates to ratio of represented object's attribute key char length to their value char length.

$$\text{Increase in DRE(Data Representation Efficiency)} \propto \frac{\sum \text{Object's attribute keys char length}}{\sum \text{Object's attribute values char length}}$$

Improving the DRE allow applications to transfer same business information in much lesser data size as compared to the generic JSON schema representation, thus in turn decreasing the user action response delay.

References

[1] Leonard Richardson, Sam Ruby, RESTful Web Services, Web services for the real world(O'Reilly Media,2007)
[2] JSON online: http://www.json.org/ - ECMA-262 (ISO/IEC 16262), ECMAScript® Language Specification, 3rd edition (December 1999)
[3] Peter Aiken, M. David Allen, XML in Data Management (Elsevier,2004)
[4] Java Architecture for XML Binding (JAXB), http://www.oracle.com/technetwork/articles/javase/index-140168.html