GraphQL_and_RESTful_in_SIM_LP2M_of_the_Hasanuddin_University.pdf

by Armin Lawi

Submission date: 26-Apr-2020 08:36PM (UTC+0700)
Submission ID: 1308057729
File name: GraphQL_and_RESTful_in_SIM_LP2M_of_the_Hasanuddin_University.pdf (158.09K)
Word count: 2682
Character count: 14750

The 2nd East Indonesia Conference on Computer and Information Technology (EIConCIT) 2018

Performance Analysis of GraphQL and RESTful in SIM LP2M of the Hasanuddin University

1st Dewi Ayu Hartina
Department of Computer Science
Hasanuddin University
Makassar, Indonesia
dewiayuhartina@gmail.com

2nd Armin Lawi
Department of Computer Science
Hasanuddin University
Makassar, Indonesia
armin@unhas.ac.id

3rd Jenny Leonard Enrico Panggabean
Department of Computer Science
Hassanuddin University
Makassar, Indonesia
bendo01@gmail.com

***Abstract*—GraphQL is a new concept in building an API. GraphQL is a Query Language developed by Facebook and implemented on the server side. Although it is a query language, the GraphQL is not directly connected with the database. In other words, GraphQL is not limited to both SQL and NOSQL databases. GraphQL which uses single endpoints is more efficient than RESTful which uses many endpoints but GraphQL will also be a little slower in querying complex databases and have many relationships beside that REST is built on multiple endpoints for specifying the return data, oftentimes multiple endpoints be required to be called when it needed. It will increase the number of client-server calls for displaying the data to the user and this could possibly result in poorer performance of the service in a Web Application needs. This paper analyses the performance calculation of the GraphQL and RESTful technologies in the web information services system of the Institute for Research and Community Service (LP2M) of the Hasanuddin University. The performance parameters used are Response Time and Throughput. Our results showed that in terms of speed RESTful is still superior to the GraphQL since the speed of RESTful is consistently stable in terms of access time and data size. Whereas the GraphQL is dynamic since it can be change depend on demand fluctuation.**

Keywords—GraphQL, RESTful, API, throughput, response time

I. INTRODUCTION

Lately, the crowd is being talked about by a web protocol called GraphQL. GraphQL is a query language for building APIs. Although a query language GraphQL isn't tied to any specific database or storage engine and is instead backed by existing code and data. The position of this GraphQL is on the server side executing the query using the type system specified for the data [1]. This web protocol was developed by Facebook technology company in 2012 but only in 2017 was this technology patented yesterday and immediately boomed among website developers. But of course there are still doubts about this technology because this technology is still in doubt when compared to the web protocol that has been used by developers from 2000, namely RESTful. REST is built on multiple endpoints for specifying the return data, oftentimes multiple endpoints be required to be called when it needed. It will increase the number of client-server calls for displaying the data to the user and this could possibly result in poorer performance of the service in a Web Application needs [2] beside that GraphQL which uses single endpoints is less efficient than RESTful which uses many endpoints and GraphQL will also be a little slower in querying complex databases and have many relationships [2] but among developers on Facebook and some web developers claim that GraphQL is better than RESTful because its usage is simpler and claimed to reduce the burden on the server. Due to the absence of specific testing of this new technology, there are still many people who doubt it.

Research on Performance Analysis from GraphQL and RESTful has been done, but GraphQL and RESTful implemented on e-Commerce company [2-4] while in this research, GraphQL and RESTful are implemented on information systems.

According to the explanation of the above problems, a solution is obtained by building an information system that uses GraphQL Web Service and RESTful then perform performance testing. One object that can be used as a test sample is the Web Service for Monitoring and Evaluation System of the Institute for Research and Community Service (LP2M) of Hasanuddin University so that the results of this research are expected to be useful for developers in considering the Web Service API migration from previous technology to technology GraphQL.

II. METHODOLOGY

A. Research Flowchart

This research starts from building information systems, in this case the Information System of the Institute for Research and Community Service (LP2M) Hasanuddin University. Then the web protocol was made with RESTful and GraphQL. Furthermore, entering data in an RDBMS, then later test and compare the performance between the two web protocols using the characteristics of QoS (Quality of Services). Research Flowchart as shown in Fig. 1.

B. RESTful

REST (Representational State Transfer) is an API (Application Programming Interface) that uses the HTTP protocol to client-server communications on a Web Application, making it easily acceptable since it is not bound to any particular transfer protocol [5-7]. The way it works, RESTful servers provide a path to access resources (data sources), while the RESTful client accesses resources and then displays or uses them. The resulting resource is actually in the form of text, but the format can vary depending on the wishes of the developer, generally JSON and XML. The main design principles of REST are addressability, uniform interface, and statelessness. REST overcomes acceptability by defining endpoints in directory structure through different URIs to extract data. The API works based on the principles of CRUD (Create, Read, Update, Delete), which appropriate the most popular functions INSERT, SELECT, UPDATE, and DELETE, in persistent data storage such as SQL [2],[7],[8].

978-1-5386-8050-6/18/$31.00 ©2018 IEEE

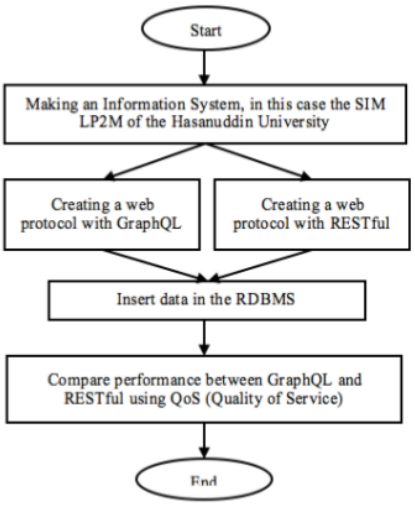

Fig. 1. Research Flowchart

C. GraphQL

GraphQL is a new concept in building an API an open source query language for developed by Facebook Inc. GraphQL executes server-side queries and returns only the data that is defined by a type system in the corresponding Web Service. Although it is a query language, GraphQL is not directly related to the database, in other words GraphQL is not limited to certain databases, either SQL or NOSQL. GraphQL is only a translator (query language) and runtime, so it doesn't depend on the programming language of the server side and any database. Available variables and fields for querying are defined in schemas that are located server-side. Specific queries can be constructed based on GraphQL have pre-defined services that are available for querying allowing for a single endpoint rather than multiple endpoints [2],[8-11].

D. QoS in Web Services

To build an information system that is small to large, it requires a standardization and mechanism, so that the system that is built has good quality, besides it is easy to maintenance. Web services have the ability to implement QoS on its services. There are huge numbers of frameworks can implement QoS on their web services and there are many aspects of QoS are important for web services. The need for QoS depends on Web services that refer to the quality of functional and non-functional characteristics of a web service. Each category must have a benchmark for web services. The QoS attribute of Web Service has the highest priority for various service providers because of the unpredictable and dynamic nature of web services, including web services Performance [3], [12-15].

Performance is one of the main issues that are very important in Web Services. Web services have proven that this technology is easier to implement and superior to the technology available at this time [3].

The performance of a Web service concerns how fast a Web service request can be processed and serviced. This requirement can be determined by measurements like throughput and response time. Throughput is the measure of the number of requests serviced in specific amount of time. And response time is the time when the Server replies after processing the request. Obviously, the response time and the throughput depend on the workload that the Web server is experiencing at that time [3],[14-16].

III. Results and Discussion

The Web Service concept appears to bridge existing information systems without disputing the difference in platforms used by each source. In building Web Services there are several technologies that can be used including SOAP, RESTful, etc. But in 2015 Facebook released a new innovation in query language which was later called GraphQL. According to the developer company, the GraphQL query language is easier to use and can reduce data calls that are not needed (over fetching).

A. Data Source

This activity is carried out by observing the overall system needs. Data was taken from the Institute of Research and Community Service (LP2M) Hasanuddin University.

B. Architecture

System architecture, consists of 3 parts, namely database management system, web application, and web service. The system specifications used are as follows:

- The selected RDBMS is PostgreSQL 10
- The Web application used is a web application with the PHP 7 programming language using Laravel 5.6 framework that runs on port 443
- Nginx 1.6 as a web server

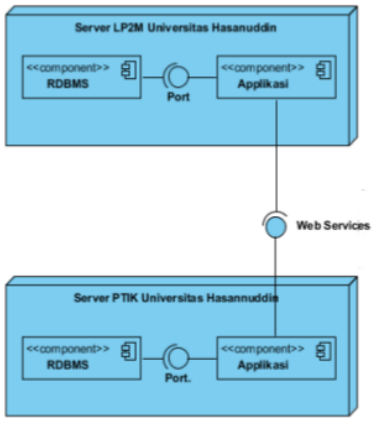

Fig. 2. System Design

C. Testing

Testing is done by a method that will represent the differences of the two technologies used to retrieve data on the server side that will be used on the client side. Which will produce conclusions after making comparisons to the two main factors that can distinguish these two technologies, namely calculating throughput and response time.

D. Query Data using GraphQL

Before querying data, all attributes or variables in the database must be declared in the model file as below:

```
return [
        'id' => [
        'name' => 'id',
        'type' => Type::string(),
        'description' => 'id of villages
searching with equals'
],
        'code' => [
        'name' => 'code',
        'type' => Type::string(),
        'description' => 'code of villages
searching with like'
],
        'name' => [
        'name' => 'name',
        'type' => Type::string(),
        'description' => 'code of villages
searching with like'
        ],
```

Fig. 3. Data Model

```
{LocationVillage(limit:300){
  data{
        id,
        code,
        name,
        slug,
        alt_slug,
        state_ministry_code,
        state_ministry_full_code,
        state_post_department_code,
        created_at,
        updated_at,
        created_by,
        dikti_name
        }
   }
}
```

Fig. 4. Query data on GraphQL

From the picture above it can be seen how to query using GraphQL data that is queryed as Village data and generate data with JSON format [15].

Where to make GraphQL require 3 important parts that must be defined first, namely: Mutation, Query, and Type.

- Mutation: as a place to update, input, and delete data.
- Query: for calling data.

Type: to describe the type of field / data.


```
{ "data": {
    "LocationVillage": {
       "data": [
            {
            "id": "5aa7dd98-25d9-49ad-b80c-
68f2d14ffe42",
            "code": "30117",
            "name": "1 Ilir",
            "slug": "sumatera-selatan-
palembang-ilir-timur-ii-1-ilir",
            "alt_slug": null,
            "state_ministry_code": null,
            "state_ministry_full_code": null,
            "state_post_department_code": null,
            "created_at": null,
            "updated_at": null,
            "created_by": null,
            "dikti_name": null        },
```


Fig. 5. GraphQL Query Result

E. *Query Data using RESTful*

To query the RESTful data call is done on the Controller

```
public function index()
{
 $models = Model::paginate(300);
 return response()->json ([$models],200);
}
```

Fig. 6. Query data on RESTful

And the data format is the same as if using GraphQL namely JSON format


```
[
 {
    "current_page": 1,
    "data": [
    {
        "id": "07e24b20-355b-4882-b5ee-
a22fc0a3cbdb",
        "code": "99354",
        "name": "Maruway",
        "sub_district_id": "907fba95-e8ff-
412a-8370-5c3037afc681",
        "created_by": null,
        "modified_by": null,
        "created_at": "2016-09-30 05:21:07",
        "updated_at": "2017-04-12 17:11:40",
        "deleted_at": null,
        "slug": "papua-jayapura-yokari-
maruway",
        "alt_slug": null,
        "state_ministry_code": null,
        "state_ministry_full_code": null,
        "state_post_department_code": null,
        "state_ministry_name": null,
        "dikti_name": null
    },
```


Fig. 7. RESTful Query Result

Where RESTful uses Data Manipulation Language (DML) in the SQL programming language. DML is a collection of query commands that are used to manipulate data in a database. Used to add data, change data, or delete data in the database [7].

Performance analysis is carried out using the API Testing application, Insomnia and measured through Throughput and Response Time. Testing was carried out as many as 20 (twenty) times repeatedly against GraphQL and RESTful.

F. Response Time Test

For response time testing is done by calculating the response time given by RESTful and GraphQL when the user takes the data retrieval process. With the formula to calculate the response time:

$$\text{Response time} = \text{execution time} + \text{data transfer time} \quad (1)$$

According to our results in Figure 8, Response Time at GraphQL has an average time in the range 1810-2130 / ms while in RESTful it has a faster time with a time range of 891-1000 / ms. So it can be seen that RESTful has a better speed compared to GraphQL in terms of Response Time.

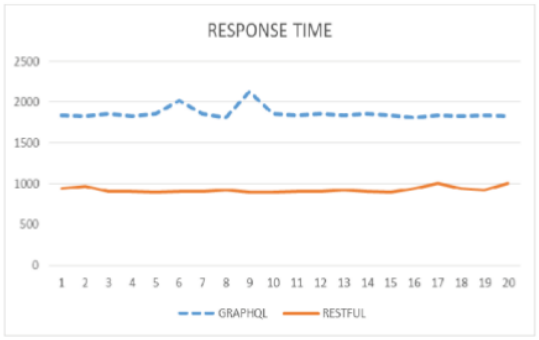

Fig. 8. Graph Response Time Test

G. Throughput Test

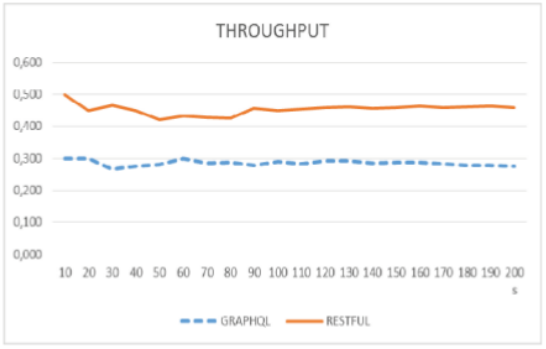

Fig. 9. Graph Throughput Test

For Throughput testing is done by testing how many times RESTful and GraphQL perform the process of retrieving data in a certain time unit. In this case, 10 (ten) seconds are given for each test. The addition of time from the first test and the next test is 10 (ten) seconds. Then the formula is given to calculate throughput as (2).

$$\frac{maximal\ number\ of\ request\ handeled}{time} \quad (2)$$

In accordance with Fig. 9, Throughput at GraphQL is able to make requests in 0.267-0.300 / s. Whereas RESTful is able to request as much as 0,420-0,500/s. Therefore it can be seen that RESTful is able to handle requests for more seconds union than GraphQL.

IV. Conclusion

Based on the research that has been done and described in the previous discussion, it can be concluded the difference in the use of GraphQL and RESTful web service technology in information systems in terms of its manufacture, where GraphQL requires 3 important parts that must be defined in advance: Mutation, Query, and Type before using Data Manipulation Language (DML). While RESTful directly uses DML in SQL programming languages. The comparison results obtained from the results chapter and discussion look at the picture that distinguishes between the two forms of service, the visible variables distinguish between the two services where RESTful speed looks superior because RESTful is close to constant based on time and data size, while GraphQL is dynamic because changes can occur on a request

Based on several points of the conclusions that have been described, the use of GraphQL and RESTful depends on the needs of the system or application designed by the developer, where GraphQL becomes the right choice if the need for data often changes.

References

[1] F. Inc, "Introduction to GraphQL," 2018. [Online].

[2] Helgason, Arnar Freyr. "Performance analysis of Web Services: Comparison between RESTful & GraphQL web services." (2017).

[3] Reddy, Ch Ram Mohan, and RV Raghavendra Rao. "QoS of Web service: Survey on performance and scalability." Computer Science and Information Technology 3.9 (2013): 65-73.

[4] Rajendran, T., and P. Balasubramanie. "Analysis on the study of qos-aware web services discovery." arXiv preprint arXiv:0912.3965 (2009).

[5] Eizinger, Thomas. "API Design in Distributed Systems: A Comparison between GraphQL and Rest." (2017).

[6] Zur Muehlen, Michael, Jeffrey V. Nickerson, and Keith D. Swenson. "Developing web services choreography standards—the case of Rest vs. Soap." Decision Support Systems40.1 (2005): 9-29.

[7] Ghebremicael, Eyob Semere. Transformation of REST API to GraphQL for OpenTOSCA. MS thesis. 2017.

[8] Johansson, Petter. "Efficient Communication With Microservices." (2017).

[9] Dhore, Sunil R., and M. U. Kharat. "QoS based web services composition using ant colony optimization: mobile agent approach." International Journal of Advanced Research in Computer and Communication Engineering 1.7 (2012): 519-527.

[10] Cederlund, Mattias. "Performance of frameworks for declarative data fetching: an evaluation of Falcor and Relay+ GraphQL." (2016).

[11] Choi, Min. "A performance analysis of RESTful open API information system." International Conference on Future Generation Information Technology. Springer, Berlin, Heidelberg, 2012.

[12] Jamil, Hasan M. "Design of declarative graph query languages: On the choice between value, pattern and object based representations for graphs." Data Engineering Workshops (ICDEW), 2012 IEEE 28th International Conference on. IEEE, 2012.

[13] Hartig, Olaf, and Jorge Pérez. "An initial analysis of Facebook's GraphQL language." AMW 2017 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web, Montevideo, Uruguay, June 7-9, 2017.. Vol. 1912. Juan Reutter, Divesh Srivastava, 2017.

[14] Cederlund, Mattias. "Performance of frameworks for declarative data fetching." 2016.

[15] Gustavsson, Kit, and Erik Stenlund. "Efficient data communication between a webclient and a cloud environment." (2016).

[16] Medi Taruk, Edy Budiman, Haviluddin, Hario Jati Setyadi. 2018. Comparison of TCO Variants in Long Term Evolution (LTE). 10.1109/ICEEIE.2017.8328776.

GraphQL_and_RESTful_in_SIM_LP2M_of_the_Hasanuddin_Un..

ORIGINALITY REPORT

9%	4%	7%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	drezewski.eu5.net Internet Source	1%
2	Submitted to Pondicherry University Student Paper	1%
3	lab.wallarm.com Internet Source	1%
4	Syafruddin Syarif, Armin Lawi, Jeffry. "Proposed priority packet data dissemination scheduling mechanism", 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), 2017 Publication	1%
5	SungHeun Nam, YunHee Kang. "XML Schema Design for Web Service Quality Management", 2008 Second International Conference on Future Generation Communication and Networking, 2008 Publication	1%
6	ecollect.org Internet Source	1%

7	cs.sci.unhas.ac.id Internet Source	1%
8	Submitted to Pacific Union College Student Paper	<1%
9	silvermountainfinancial.com Internet Source	<1%
10	www.eduhk.hk Internet Source	<1%
11	Nurul Hardiyanti, Armin Lawi, Diaraya, Firman Aziz. "Classification of Human Activity based on Sensor Accelerometer and Gyroscope Using Ensemble SVM method", 2018 2nd East Indonesia Conference on Computer and Information Technology (EIConCIT), 2018 Publication	<1%
12	Jael Zela Ruiz, Cecília M. Rubira. "Quality of Service Conflict During Web Service Monitoring: A Case Study", Electronic Notes in Theoretical Computer Science, 2016 Publication	<1%
13	Submitted to Indian Institute of Technology, Bombay Student Paper	<1%
14	"Service-Oriented Computing – ICSOC 2017 Workshops", Springer Science and Business	<1%

Media LLC, 2018
Publication

15 kodepos.id
Internet Source

<1%

Exclude quotes	On	Exclude matches	< 5 words
Exclude bibliography	On		