The Great RESTfulness test

Fredriksson, Henrik and Olaussen, Jonas
TDDE41, VT2019

Research Question

How to evaluate to what extent application programming interfaces are aligned with the REST architecture principles including the HATEOAS principles from the human developer perspective?

Background

The REST architecture principles

R. T. Fielding defines the REST architecture principles as follows (Fielding, 2000):

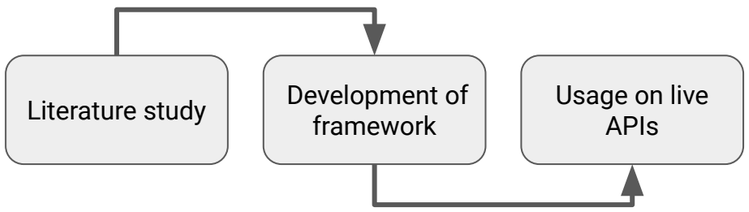
- **Client-server**: Separates the user interface from the data storage and/or processing power.
- **Stateless**: Implies that a request must contain all necessary information without utilizing any stored context on the server.
- **Cache**: Labeling API calls cacheable improves network efficiency.
- **Uniform Interface**: Simplifies and unifies API interfaces.
- **Layered System**: Hierarchical layers in the architecture improving scalability.
- **Code-On-Demand:** An optional constraint that allows code to be downloaded and executed directly.
- **HATEOAS**: The response to the client includes additional dynamic hypermedia, links, to navigate between functionality offered by the server. Thus, reducing the information needed to utilize the API (Fielding, 2008).

Defining a good API

F. Doglio discusses key features that define a good API and relates them to the REST architecture principles. Listing the following features as necessary for a well-implemented API: **developer-friendliness**, **extensibility**, **up-to-date documentation**, **proper error handling, providing multiple SDK/libraries**, **security, and scalability** (Doglio, 2018).

Method

In order to answer the research question, a literature study was conducted on the REST architecture principles and on literature surrounding what constitutes a well-implemented API in general. Based on the literature study, a framework for evaluating to what extent an API follows the REST architecture principles were developed. In addition, a sample web application incorporating **Google Maps, Youtube, NASA,** and **OpenWeatherMap** APIs was created in order to evaluate the viability of the proposed framework.



Result

The RESTfulness test framework

The table below is the proposed framework for RESTfulness testing incorporating different perspectives from the literature study. Question 1, 2 and 7 are aimed to capture more qualitative values aimed at the developer experience, while the remaining questions are of a quantitative nature. Each question consists of a value and weight which multiplied together generates the result. The maximum average score is 9,61.

Nr	Question	Value format	Value	Weight	Result
1	How up to date is the documentation?	Range 1 to 7		1,5	
2	How would you rate the developer experience?	Range 1 to 7		2	
3	How many endpoints have proper error handling?	%		10	
4	Does the API align with the principal of client and server?	1 or 0		10	
5	How many endpoints can be called without state?	%		10	
6	How many endpoints implies the cacheable nature of the data?	%		8	
7	How would you rate the uniformity of API calls and responses?	Range 1 to 7		2	
8	Does the API include the feature of Code-On-Demand?	1 or 0		3	
9	How many endpoints have incorporated the feature of HATEOAS?	%		7	
				Average	

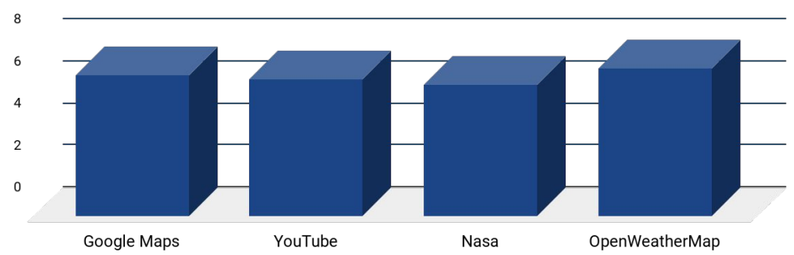
The sample web application

The developed sample web application contains a variety of functionality from the tested APIs, such as location search, weather info, and astronomy pictures. The web application can be found in this Git repo: bit.ly/2Wn2T30

The RESTfulness test framework applied on the chosen APIs

The testing conducted on the selected APIs resulted in the following results, that can be seen in the figure below.

Average score per API



Discussion

Methodology

The development of the sample web application resulted in, as expected, that experience was gained with the different APIs tested. It also resulted in insights regarding how important the different factors of the framework are in relation to each other, as well as in insights regarding weaknesses of the framework. A weakness with the methodology is arguable that four tested APIs are not enough to profoundly evaluate the framework.

The RESTfulness test framework

The proposed testing framework is considered a successful tool for evaluating different APIs in regards to RESTfulness. By examining question 1, 2 and 7 it is possible to evaluate the developer experience compared to the provided literature (Doglio, 2018). The remaining questions make it possible to analyse the factors considering the RESTfulness of the API (Fielding, 2000; Fielding, 2008).

A weakness with the framework is that some questions are to an extent subjective. It can for example be challenging to evaluate whether a certain API call has proper error handling or not, since API calls often have error handling to some extent but not fully for all variables provided. Another subjective question is regarding how many endpoints that implies the cacheable nature of the data. None of the four tested APIs provided explicit info regarding if the data response was cacheable or not.

It is also subjective to an extent how important the different factors are in the test and thus what weight each question should have. As a result, the weights could be adjusted by the tester to a preferred weight.

Furthermore, the resulting scores obtained when applying the framework on the tested APIs were similar, close to the maximum average. This could indicate flaws in the framework but are more likely due to the good development quality of the tested APIs.

Further Research

To examine the viability of the framework further, a larger set of APIs could be used. Automation possibilities, both fully and semi-automated testing, could be further explored since automated tests based on the framework would facilitate large scale API testing, as mentioned by (Hamza et.al., 2018). Due to the temporal nature of this study, this was not expanded on.

Conclusion

The conclusion is that the proposed framework can be used to successfully evaluate to what extent an API follows the REST architecture principles. It contains a clear connection between theory and practice but is not perfect and free from weaknesses. It is important to acknowledge that the questions in the framework in practice are somewhat subjective, which is derived from the fact that the implementation of the REST architecture principles themselves to an extent are subjective and depend on the circumstances and the experiences from the developer. This is what makes the automation of the test framework a challenge. This also emphasizes that the REST architecture principles should be seen as principles or as overall guidelines, rather than strict implementation rules. Something that in turn affects the evaluation of to what extent APIs align with them.