An Empirical Study of Web API Quality Formulation

Esi Adeborna[1] and Kenneth K. Fletcher[2(✉)]

[1] University of Massachusetts Lowell, Lowell, MA 01854, USA
esi_adeborna@student.uml.edu
[2] University of Massachusetts Boston, Boston, MA 02125, USA
kenneth.fletcher@umb.edu

Abstract. This paper presents an empirical study on one of the most popular web API repositories, www.programmableweb.com. The study is to ascertain the impact of the structure and formulation of external web API quality factors on the overall web API quality. The study is based on the hypothesis that, in such a multi-factor quality measurement, the structure and formulation of the quality factors can make a substantial difference in its quantification. Specifically, we employ statistical tools such as exploratory factor analysis, to determine the latent factors that contributes to web API quality. We subsequently determine the loading of each latent factors to propose a new quality model for web API quality computation.

Keywords: Web API · Web API quality · Web API quality factors · Factor analysis · Mashup development

1 Introduction

Web Application Programming Interfaces (APIs) have become increasingly prevalent in recent past as they provide a platform that allows other applications to interact and request for data or use their functionality. With this many web APIs, typically with similar functionality, it becomes challenging to select or recommend web APIs to meet users' needs. Therefore, in order to provide a distinction among functionally similar web APIs, quality factors are used [1,2].

Web APIs, unlike web services, hide their internal complexities and internal details. Therefore they depend on external factors to drive their suitability for integration into other applications [1,3,4]. According to the standard ISO/IEC 9126-1, external quality is based on a black box model and is related to the behavior of the software product in a given running environment [5]. Consequently, several API quality models, that depend on their external quality factors have been proposed [1,4,6]. In one of such quality models, proposed by Fletcher [1] which was an extension of Cappiello et al. [4], the quality of a web API depends on three quality dimensions, namely, *Functionality*, *Reliability* and *Usability*. This model was purely based on theoretical foundation and the behavior of web

Q. Wang et al. (Eds.): SCC 2020, LNCS 12409, pp. 145–153, 2020.
https://doi.org/10.1007/978-3-030-59592-0_10

APIs and as such has a couple limitations: (1) Their model assumes that all three quality dimensions contribute equally to the overall web API quality. This assumption equates the salience of each quality dimension in the web API Quality computation and therefore balances the Quality values of web APIs even if one dimension has increased salience than the others. (2) The formulation of the web API quality model does not take into consideration the correlation of the quality factors that make up the dimensions.

These limitations can lead to varied implications such as (1) inaccurate computation of low quality APIs as high quality, where low-quality web APIs could result in difficulty to integrate with other APIs and (2) cause developers to miss out on potentially quality web APIs because of inaccurate web API Quality values [1]. In this work, we propose a method to address the above limitations, by first performing an extensive empirical analysis of web API dataset using statistical parameters and exploratory factor analysis. We subsequently formulate a model for web API quality, based on results from our analysis.

2 Background on Web API Quality

Obtaining values of quality of service (QoS) parameters, such as availability, response time, etc. for web APIs is a challenging task because, web APIs typically hide their internal complexity and therefore external factors drive the evaluation of its quality computation [1,4]. For this reason, we adopt the quality model proposed by Fletcher [1] and Cappiello et al. [4], to define our black-box quality model for web APIs. This black-box quality model are organize along three main web API dimensions: (1) **Functionality:** considers the web API's *interoperability*, *compliance*, and *security* level [1]; (2) **Reliability:** measures the maturity of the web API by considering the available statistics of usage of the component together with the frequency of its changes and updates; and (3) **Usability:** A web API's usability is evaluated in terms of understandability by considering the available web API documentation by means of examples, API groups, blogs, sample source codes etc. [1].

3 Research Approach

This section first gives an overview of our approach and thereafter describes the main modules that drives our model. Our research study focuses on a series of statistical analysis to determine the content and distribution of the variables of interest to accurately compute the quality of a web API. We develop the conceptual and mathematical underpinnings of the proposed quality model and finally propose a web API quality model based on results of our statistical analysis.

3.1 Dataset Description

Our empirical study focuses on studying one of the popular online web APIs repository, www.programmableweb.com. This is by far the largest online web

Table 1. Top 5 Web API categories from programmableweb.com

Category	Number of Web APIs
Tools	787
Financial	583
Enterprise	486
eCommerce	434
Social	402

API repository that contains approximately 23,000 web APIs, with various functionalities [3,7]. We study a version of this data, API dataset [1], which was crawled from www.programmableweb.com in March 2018. This dataset contains 12,879 web API records with 383 categories. Table 1 shows a list of the top 5 categories in the dataset. Each web API in our dataset is described by 19 fields such as name, description, authentication model, request and response formats, etc.

3.2 Empirical Study

To explore the content and the distribution of our dataset, we performed a series of statistical analysis. In our analysis, we focus on the external variables that define the quality of a web API. Table 2 presents central tendency of web API dataset variables of interest and dispersion within the variables' distribution. Due to the great numeric range in the dataset on these different variables, we normalize the values for the variables of interest.

Noteworthy findings are that, most of the web APIs have almost no *SDKs* as revealed in its average and median for 0.15 and 0 respectively. In addition, the *SDKs* are highly skewed to the right of the distribution around the mean. *How-tos* and *Sample API Codes* have a similar and largely skewed distribution as *SDK*, with mean values almost zero. There is a relatively better distribution for *API Versions* than the previous variables with a mean and median almost

Table 2. Descriptive statistics for the external quality variables (N = 12,879)

	Mean	SD	Median	Range	Skew
SDKs	0.15	0.36	0.00	1.00	1.97
How-tos	0.01	0.09	0.00	1.00	11.45
Sample Codes	0.10	0.30	0.00	1.00	2.70
API Versions	0.70	0.14	0.74	0.88	−2.60
API Languages	0.42	0.06	0.42	0.50	0.82
Data Formats	0.80	0.40	1.00	1.00	−1.47
Security	0.36	0.20	0.40	0.80	1.29

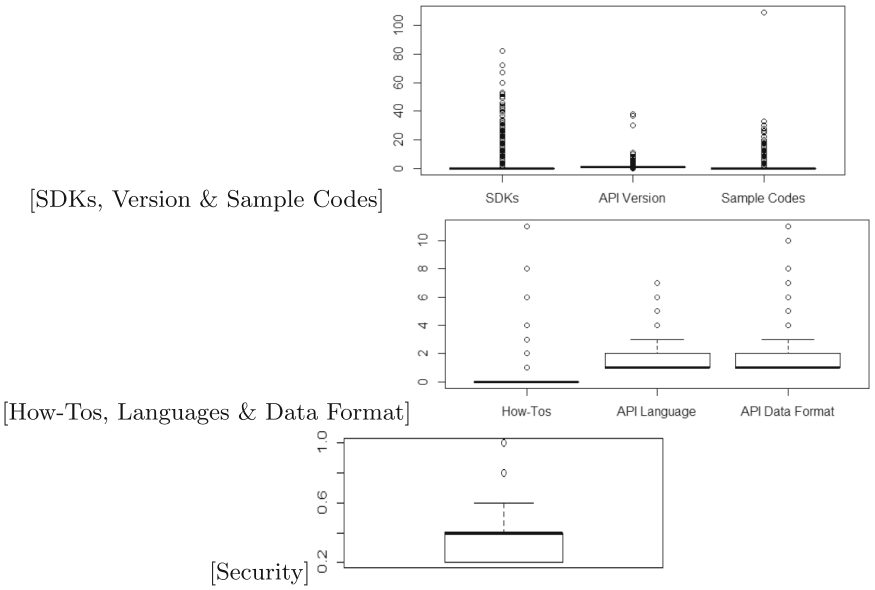

Fig. 1. Boxplots of the web API external variables

Table 3. Pearson correlation between web API external variables (N = 12,879)

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(1) SDKs	–						
(2) How-tos	0.12	–					
(3) Sample codes	0.49	0.09	–				
(4) Reliability	0.02	0.02	0.05	–			
(5) Interoperability	0.23	0.06	0.19	0.02	–		
(6) Compliance	0.01	0.01	0.03	0.64	0.11	–	
(7) Security	−0.12	0.01	−0.02	0.10	−0.45	0.03	–

identical, and a standard deviation of 0.14. The highly skewed *API Versions* imply that most of the web APIs are updated regularly. This is an indication of good **reliability**, which measures the maturity of the web API by considering the frequency of its changes and updates. The number of different *API languages* show a distribution that is moderately skewed to the right, skew = 0.82. Typically, a web API is **compliant** if it supports at least one standard web API language. The *Data formats* external variable is negatively skewed with a standard deviation of 0.4, showing a wide dispersion around the mean. *Data formats* together with *API Languages* measures the **interoperability** of a web API. *Security* is positively skewed and has a smaller dispersion around the mean with a SD of 0.20. This indicates most of the web APIs have some kind

of authentication system. Figure 1 shows the boxplots of the web API quality variables based on the API dataset.

3.3 Associations Between Web API External Variables

We employ Pearson Correlation (PC) matrix, to describe linear associations between the web API external variables. PC attempts to determine the amount of linear dependence between variables by describing their association as a straight line. Table 3 provides the results for the correlation between the variables. The values show a non-zero direct correlation between the variables with p-values less than 0.001. Our analysis of the correlation coefficients suggest the variables are weak to moderately correlated, mostly positive but negative for *Reliability* on *SDKs*, *Sample Code* and *Security*.

3.4 Exploratory Factor Analysis (EFA)

Exploratory Factor Analysis (EFA) is a statistical technique that is used to identify the latent relational structure among a set of variables [8]. Essentially, we use EFA to uncover the underlying structure of the relationship between the web API quality variables. First, we conduct parallel analysis scree plot to determine the acceptable number of factors. Figure 2 shows the scree plot of our parallel analysis. We locate the point of inflection (the point where the gap between simulated data and actual data tends to be minimum) to determine the minimum number of factors. In this case, it is 3.

Next we determined the factors to be extracted. We employ the oblique rotation based on the correlation between the variables. We used the "Ordinary Least Squared/Minres" factoring as it is known to provide results similar to "Maximum Likelihood" without assuming multivariate normal distribution and derives solutions through iterative eigen decomposition like principal axis [9,10].

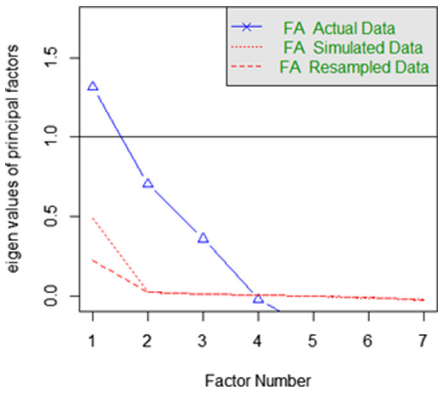

Fig. 2. EFA Parallel analysis scree plot

Using loadings of not less than absolute value of 0.3 [10], and not loading on more than one factor, Table 4 shows the results of our factor loading. With a factor of 3, our result produces a single-loading, also known as Simple Structure.

To validate the EFA for acceptable fit [9], we consider our Root Mean Square of Residuals (RMSR) from the result, which is 0. This is acceptable as the RMSR value for an acceptable fit should be closer to 0. Next we check the RMSEA (Root Mean Square Error of Approximation) index. Its value, 0.007 shows good model fit as it should be below 0.05. Finally, our Tucker-Lewis Index (TLI) is 0.99 - an acceptable value considering it's over 0.9. After establishing the adequacy of the factors, the EFA result was used as the basis for the proposed model, see Fig. 3.

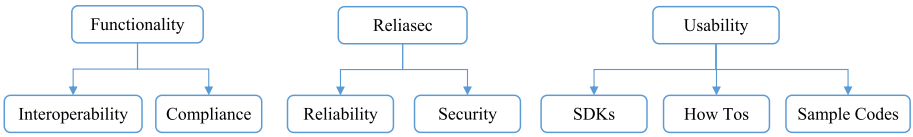

Fig. 3. Results of Factor Analysis of Web API variables

3.5 Web API Quality Formulation

Based on results from our analysis the following mathematical models have been proposed to define the web API quality dimensions. For each web API $w \in W$, there is a quality property $Q(w)$, that indicates the quality of the web API w given as:

$$Q(w) = 0.4 * Q_F(w) + 0.33 * Q_R(w) + 0.27 * Q_B(w) \quad (1)$$

where Q_F, Q_R and Q_B are the quality dimensions for *Functionality* (F), *Reliasec* (R), and *Usability* (B) respectively. 0.4, 0.33 and 0.27 are the weight each quality dimension, *Functionality* (F), *Reliasec* (R), and *Usability* (B) respectively, contribute to the calculation of the quality of a web API. A new dimension, *Reliasec*,

Table 4. Factor loading for web API external variables (N = 12,879)

	Functionality	Reliasec	Usability
SDKs	–	–	0.609
How-tos	–	–	0.440
Sample codes	–	–	0.553
Reliability	0.816	–	–
Interoperability	–	0.837	–
Compliance	0.789	–	–
Security	–	−0.572	–

has been introduced and explained in Eq. (3). The formalized descriptions of the proposed web API Quality dimensions are as follows:

$$Q_F = \frac{1}{2}[(1 + \frac{|lang|}{k} + \frac{|dformat|}{l}) + 3comp] \tag{2}$$

where *lang* and *dformat* are the languages and data formats supported by the web API, and *comp* is the compliance levels of the web API respectively.

$$Q_R = \frac{1}{2}[\frac{3}{5}sec + max\left(1 - \frac{cdate - ludate}{\frac{cdate - crdate}{|ver|}}, 0\right)] \tag{3}$$

where *cdate*, *ludate*, and *crdate* are the current date, last use date and creation date of the web API respectively, and *ver* is the set of version available for that web API. *sec* is the security level of the web API. Based on our correlation matrix Sect. 3.3, *Security* and *Compliance* are highly correlated 0.64 than any of the other variables, which suggests they are similar measures for *Functionality*. Also, the factor analysis suggests the grouping of *Security* and *Reliability* into one factor, see Fig. 3. The mathematical Eq. (3), considers *Reliability* and *Security* in one dimension, namely *Reliasec*.

$$Q_B = \frac{1}{3}(sdks + how\_to + sample\_codes) \tag{4}$$

where *sdks*, *how_tos*, and *sample_codes* are number of SDKs, How-to Documentations and Sample Codes that are available for web API users.

We use Cronbach's alpha to validate the internal consistency of our proposed quality model in comparison to the existing model. Cronbach's alpha is a measure of internal consistency that indicates how closely related a set of items are as a group. The alpha values for the revised dimensions in the proposed models are 0.78 and 0.63 for functionality and reliasec respectively. These values are higher than the existing model of 0.51 and 0.46 for functionality and reliability.

4 Related Work

Web API quality research is minimal and the articles that focus on API quality have no emphasis on the effect of splitting web API attributes by weight in the multi-attribute measurement of web API quality. Bermbarch and Wittern [6] proposed an approach and a toolkit for benchmarking the quality of web APIs considering geo-mobility of clients. Their quality model comprises of two interconnected attributes namely, availability and performance. Volatile latency and temporary unavailability were considered quality problems without quantification of the role that each attribute plays in the quality of the web API.

Picozzi et al. [11] also proposed a quality model for mashup services. The Mashup quality computation, proposed in their work, considers the different roles mashup components play (i.e. master, slave and filter) that affect the perception of the quality of the final integration [11]. These roles, however, are not

relevant to web APIs. Similarly, Fletcher [1] proposed a method that employs the black-box approach to analyze the quality of web APIs that match a mashup developer's requirement. Though his work recognized the need for web API quality computation based on its attributes, he computed web API quality as the normalized sum of its dimensions. This work intends to capitalize on this contribution but goes further to evaluate the multi-attribute measurement of web API quality.

In another research by Cappiello et al. [12], they address quality of mashups in the light of the activities that characterize their development process. They proposed evaluation techniques taking into account the constituent components of mashups. Their work supports our hypothesis but does not illustrate or validate the computation of the weights.

5 Conclusion

In this work, we proposed a reconfiguration of API quality computation model to promote increased accuracy in web API Quality calculations. This study uses Correlation to prove that, Security and Compliance are highly correlated, which suggests they are similar measures for Functionality. Exploratory Factor Analysis (EFA) suggests the grouping of Security and Reliability into one factor, in the Simple Structure. We have shown that our proposed model considers Reliability and Security in one dimension, which we name Reliasec. Also, this study uses EFA to examine how much weight each API quality dimensions contribute to API Quality. For future research, we would run more experiments to ascertain the performance of our model to increase accuracy in quality web API recommendations compared with other baseline methods.

References

1. Fletcher, K.K.: A quality-based web API selection for mashup development using affinity propagation. In: Ferreira, J.E., Spanoudakis, G., Ma, Y., Zhang, L.-J. (eds.) SCC 2018. LNCS, vol. 10969, pp. 153–165. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94376-3_10
2. Fletcher, K.K., Liu, X.F.: A collaborative filtering method for personalized preference-based service recommendation. In: 2015 IEEE International Conference on Web Services, pp. 400–407, June 2015
3. Fletcher, K.K.: A quality-aware web API recommender system for mashup development. In: Ferreira, J.E., Musaev, A., Zhang, L.-J. (eds.) SCC 2019. LNCS, vol. 11515, pp. 1–15. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23554-3_1
4. Cappiello, C., Daniel, F., Matera, M.: A quality model for mashup components. In: Gaedke, M., Grossniklaus, M., Díaz, O. (eds.) ICWE 2009. LNCS, vol. 5648, pp. 236–250. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02818-2_19
5. ISO/IEC: ISO/IEC 25010: 2011 systems and software engineering-systems and software quality requirements and evaluation (square)-system and software quality models (2011)

6. Bermbach, D., Wittern, E.: Benchmarking web API quality. In: Bozzon, A., Cudre-Maroux, P., Pautasso, C. (eds.) ICWE 2016. LNCS, vol. 9671, pp. 188–206. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38791-8_11
7. Xia, B., Fan, Y., Tan, W., Huang, K., Zhang, J., Wu, C.: Category-aware API clustering and distributed recommendation for automatic mashup creation. IEEE Trans. Serv. Comput. **8**(5), 674–687 (2015)
8. Child, D.: The Essentials of Factor Analysis. Cassell Educational, London (1990)
9. Hooper, D., Coughlan, J., Mullen, M.R.: Structural equation modelling: guidelines for determining model fit. Electron. J. Bus. Res. Methods **6**(1), 53–60 (2008)
10. Kline, R.B.: Principles and Practice of Structural Equation Modeling, vol. 2. Guilford Press, New York City (2004)
11. Picozzi, M., Rodolfi, M., Cappiello, C., Matera, M.: Quality-based recommendations for mashup composition. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 360–371. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16985-4_32
12. Cappiello, C., Matera, M., Picozzi, M., Daniel, F., Fernandez, A.: Quality-aware mashup composition: issues, techniques and tools. In: 2012 Eighth International Conference on the Quality of Information and Communications Technology, pp. 10–19. IEEE (2012)