FRONT-END FRAMEWORK SELECTION FOR A HIGH-LOAD DISTRIBUTED SOFTWARE

Prepared as a Homework in the Course "Applied System Analysis"

M.V. Shmakov

Abstract—This paper outlines the choice of the front-end framework for a distributed potentially high-load software product as it is a wicked problem that includes stakeholders, plenty of criteria. Multiple frameworks as React.js, Vue.js, Ember.js, and Angular are presented on the market with their pros and cons. The main criteria to select the best of them were chosen by stakeholders: scalability, cost of maintenance, speed of development, and availability of specialists on the labor market. To cope with the process of the best framework selection PQR and CATWOE analysis are provided as soft methodologies. An Analytical Hierarchy Process (AHP) is presented as a hard methodology. The overall result is selected based on the combination of the results gained from the mentioned methodologies.

Index Terms—front-end, framework, scalability, labor availability, cost of maintenance, CATWOE, PRQ, AHP, stakeholders.

1 INTRODUCTION

The right choice of the suitable framework is one of the main steps in developing a successful high-load distributed software product that is easy to scale, develop, and maintain [4]. There is no framework suitable for all the needs exists and every one of them has its pros and cons. It is a wicked problem also since the front-end is a significant part of the product, so there are stakeholders with their different interests. The following paper is dedicated to providing the example of such framework selection considering different points of view and criteria.

2 BACKGROUND

The most usable front-end frameworks are considered to be React.js, Angular, Ember.js, and Vue.js [11]. Let us observe each of them from different angles and reveal their advantages and disadvantages [3, 4].

React.js is a library that allows us to create a complex widescale project which requires a high level of load. The library is well documented, light-weighted, and provides high performance. Such technologies as rendering "lazy" components (React.Lazy) inside the components React.Suspense makes React.js based applications more responsive by rendering without blocking the main thread. This allows React.js to work without delays while processing the high-priority tasks, such as forming an app's response to users' interactions. However, React.js does not provide the big number of integral features, like, for example, Angular does. Moreover, it may seem challenging for beginners as it requires to study many additional instruments, like, for example, Redux or Flux [7].

Angular also has very detailed documentation, it is one of the most popular frameworks, moreover, it supports two-way data binding. Nevertheless, having a big variety of functions causes the disability to solve a simple task easily. Moreover, like React.js, Angular may seem complicated for junior developers as they have learned a lot of things and how they work together, like, for example, the components, dependency injections, modules, etc [8].

Unlike React.js or Angular, Vue.js is easy to learn: it has the smallest API surface area so it is

• *M.V. Shmakov is with the School of Software Engineering, Faculty of Computer Science, The National Research University Higher School of Economics, 109028, 11 Pokrovsky Bulvar, Pokrovka Complex, Moscow, Russia. E-mail: mvshmakov@edu.hse.ru.*

quite simple to start using in a short amount of time. Moreover, it provides to create flexible components that are simple to integrate and reuse. Unfortunately, sometimes excessive flexibility may cause the code irregularities as many front-end developers may contribute. Another significant disadvantage is not the big developers' community, Vue.js is quite new and unpopular yet [9].

Ember.js is less popular than Vue.js, however, it is suitable for complex and large projects, like React.js, and supports two-way data binding, like Angular. Nevertheless, it is not as flexible as Vue.js as there is no reuse of components at the controller level [10].

3 PROBLEM DISCOVERY

As it comes to frontend frameworks, it becomes necessary to consider several criteria to choose the most suitable of them. To do it we need to consider stakeholders' interests and to analyze which of the criteria are the most important for all of the participants. Methods that are existing to solve those issues are baseв on the 'soft' approach. Examples of them are the PQR-formula [13] and CATWOE analysis which is intended to figure out main customers, agents, transformation, world, owners, environment [12].

3.1 PQR-Formula

To assess the selection of the best front-end framework for the high-load software, there is a need to identify what exactly we want to to, how we want to do it and why do we want to solve this problem at all. To answer this question, it is a good idea to construct PQR-formula.

Element	Meaning	Interpretation
P	What?	Selection of the front-end framework for a high-load project
Q	How?	Compliance with requirements and stakeholders
R	Why?	Variety of front-end frameworks

Table 1. PQR-formula for a given problem

3.2 CATWOE analysis

The second step is to identify the main aspects of the given problem. We will construct the CATWOE analysis table to identify and include all of the stakeholders in the process of making a decision.

Element	Interpretation
Customers	End users
Agents	Team of developers, product owner
Transformation	Product quality and success depends on the chosen framework to some extend
World	Internet, browsers
Owners	Product owner
Environment	Resources, project limits, stakeholders, standards

Table 2. CATWOE analysis for a given problem

This analysis shows that we need to consider the opinions of the end-users, the team of developers, product owners and we need to select the framework that will suit the limitations of the project and will lay in the standards of the company.

4 HARD METHODOLOGY (AHP)

To actually make a decision we decide to use Analytical Hierarchy Process (AHP) which based on analysis and linear algebra equations [1-3]. The first step of this methodology is to construct the hierarchy of the given problem which consists of goals, criteria, and alternatives [2].

4.1 AHP hierarchy

The AHP hierarchy is shown in diagram 1. After the poll with stakeholders, it was decided to choose these criteria: scalability (S), cost of maintenance (M), speed of development (Sp), labor availability (L), framework size (F). And those main frameworks were selected: React.js, Vue.js, Ember.js, and Angular.

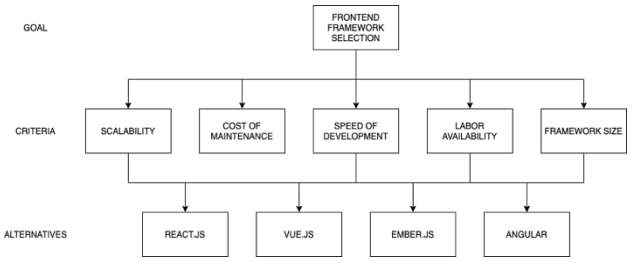

Diagram 1. Decision Hierarchy

4.2 Pairwise comparison

The next step is to conduct the pairwise comparison of the selected criteria. To do this first we need to construct the comparison table which can be referenced by all of the stakeholders to select the appropriate mark for each of the pairwise comparison items (table 5). Then the comparison table was created (Table 4).

	S	M	Sp	L	F
S	1	1/3	1/3	1/5	1/9
M	3	1	1	1/7	1/5
Sp	3	1	1	1/5	1/7
L	5	7	5	1	1/3
F	9	5	7	3	1

Table 4. Criteria with their importance values

According to AHP, we need to calculate the consistency index, a random index to check the consistency rate. For the consistency to be accepted, the ratio between CI and RI (consistency rate, CR) a given n, as suggested by T. Saaty [2], must be less than 0.1 (10%). Calculated statistics:

Consistency index: 0.07584961318282857+0j
Random index (RI): 1.12 (n = 5)
Consistency rate (CR): 0.068+0.000j < 0.1
Weights: [0.04006456, 0.08514782, 0.08126923, 0.29638624, 0.49713215]

So, we can accept this matrix with its weights and go further.

4.3 AHP alternatives

Following generally the same logic we then need to construct the respective matrices and analyze them in the same way (checking the consistency rate and gaining the weights of the frameworks with the respect to this alternative)

Alternative concerning Scalability

	Vue.js	Angular	React.js	Ember.js
Vue.js	1	3	5	5
Angular	1/3	1	5	7
React.js	1/5	1/5	1	1
Ember.js	1/5	1/7	1	1

Table 6. Criteria with their importance values

For the Scalability (table 6) the following values were calculated:

Consistency index: 0.07278826105365846+0j
Random index (RI): 0.9 (n = 4)
Consistency rate (CR): 0.081+0.000j < 0.1
Weights: [0.51038052, 0.33480938, 0.07904979, 0.07576031]

Since the consistency index is less than 0.1 and all of the stakeholders are satisfied with marks, we can move on with these weights.

For all of the criteria in Table 5, we need to calculate the respective matrices in the same way.

Alternative concerning Maintenance cost

	Vue.js	Angular	React.js	Ember.js
Vue.js	1	1/7	1/3	3
Angular	7	1	3	7
React.js	3	1/3	1	5
Ember.js	1/3	1/7	1/5	1

Table 7. Criteria with their importance values

	Vue.js	**Angular**	**React.js**	**Ember.js**
Scalability	Requires state managers (vuex)	Scalable out-of-the-box	Requires state managers (Redux)	Requires state managers (Redux)
Maintenance cost	Cheap due to simple API	High due to rich API	Cheap due to simple API	Middle
Speed of development	The high, steep learning curve	The slow, sharp learning curve, rich API	The high, steep learning curve	Middle
Labor availability	Middle	2nd in Russia	Top-1 in Russia	Middle
Framework size	33.5 KB (minified & gzipped)	563 kB (minified & compressed)	86 KB (minified & compressed)	95 kB (minified & gzipped)

Table 5. Features of the various alternatives

Consistency index: 0.04657703611996962+0j
Random index (RI): 0.9 (n = 4)
Consistency rate (CR): 0.052+0.000j < 0.1
Weights: [0.109375, 0.58363971, 0.25091912, 0.05606618]

Alternative concerning Speed of development

	Vue.js	Angular	React.js	Ember.js
Vue.js	1	1/3	1	1/5
Angular	3	1	3	1/7
React.js	1	1/3	1	1/7
Ember.js	5	7	7	1

Table 8. Criteria with their importance values

Consistency index: 0.07278826105366261+0j
Random index (RI): 0.9 (n = 4)
Consistency rate (CR): 0.081+0.000j < 0.1
Weights: [0.08910256, 0.19038462, 0.07948718, 0.64102564]

Alternative concerning Labor availability

	Vue.js	Angular	React.js	Ember.js
Vue.js	1	3	5	3
Angular	1/3	1	1	1/3
React.js	1/5	1	1	1/5
Ember.js	1/3	3	5	1

Table 9. Criteria with their importance values

Consistency index: 0.06244887852833081+0j
Random index (RI): 0.9 (n = 4)
Consistency rate (CR): 0.069+0.000j < 0.1
Weights: [0.49728641, 0.11510854, 0.08989846, 0.29770658]

Alternative concerning Framework size

	Vue.js	Angular	React.js	Ember.js
Vue.js	1	1/3	1/5	3
Angular	3	1	1/3	5
React.js	5	3	1	5
Ember.js	1/3	1/5	1/5	1

Table 10. Criteria with their importance values

Consistency index: 0.06602264337157493+0j
Random index (RI): 0.9 (n = 4)
Consistency rate (CR): 0.073+0.000j < 0.1
Weights: [0.12758565, 0.27286684, 0.53288623, 0.06666128]

All of the calculated consistency indices are less than 0.1, so we say that we can take the weights, and these pairwise comparisons as something we can proceed.

4.4 Goal prioritization

The last step is to use the calculated weights ШТ order to calculate the resulting weights of each alternative and prioritize them. To make a decision, the various weights of each alternative is multiplied by the weight of the corresponding criterion and the result for an alternative are summed up. The results are shown in the table below:

	S	M	Sp	L	F	Weights
Criteria weight	0.040	0.085	0.081	0.296	0.497	
Vue.js	0.510	0.109	0.089	0.497	0.127	0.247
Angular	0.334	0.583	0.190	0.115	0.272	0.248
React.js	0.079	0.250	0.079	0.089	0.532	0.323
Ember.js	0.075	0.056	0.641	0.297	0.066	0.181

Table 11. Criteria with their importance values

From the results that are obtained from Table 11, it is obvious that **React.js** is the most preferred alternative from the list with a weight of **0.323**. This corresponds with the stakeholders' highest preference for cost since React.js is the most popular and flexible solution with a high speed of development and low size of a binary file. The next two possible solutions to choose are the **Angular** with the weight of **0.248** and **Vue.js** with a goal value of **0.247.** The least likely framework to choose is **Ember.js** with a weight of **0.181**.

5 CONCLUSIONS

The PQR-formula and CATWOE analysis are suitable to reveal main stakeholders, alternatives criteria to consider, and main reference points. Using the Analytical Hierarchy Process proposed by Thomas L. Saaty, we have conducted problem modeling, pairwise comparisons marking, verification, and final prioritization. According to it, React.js was chosen as the most preferred alternative considering all of the stakeholders' opinions. The other possible options are to choose Vue.js or Angular since they have nearly the same weights, but they are is much less appropriate than the main one.

REFERENCES

[1] Thomas L. Saaty (1990) "How to make a decision: The Analytical Hierarchy Process" Joseph M. Katz Graduate School of Business, University of Pittsburg, Pittsburg PA 15260, USA. pp. 15.

[2] Saaty, T.L. (2008) 'Decision making with the analytic hierarchy process', Int. J. Services Sciences, Vol. 1, No. 1, pp.83–98.

[3] Saaty, T.L. (1980) The Analytic Hierarchy Process, New York: McGraw Hill

[4] Nikulchev, E., Ilin, D., Kolyasnikov, P., Zakharov, I., & Malykh, S. (2019). Programming technologies for the development of a web-based platform for digital psychological tools. *arXiv preprint arXiv:1906.05276*.

[5] Pano, Amantia, Daniel Graziotin, and Pekka Abrahamsson. "What leads developers towards the choice of a JavaScript framework?." *arXiv preprint arXiv:1605.04303* (2016).

[6] bin Uzayr, S., Cloud, N., & Ambler, T. JavaScript Frameworks for Modern Web Development.

[7] React.js documentation - https://reactjs.org/docs/thinking-in-react.html [accessed - 23.05.2020].

[8] Angular documentation - https://angular.io/docs [accessed - 23.05.2020].

[9] Vue.js documentation - https://ru.vuejs.org/v2/guide/ [accessed - 23.05.2020].

[10] Ember documentation - https://api.emberjs.com/ [accessed - 23.05.2020].

[11] State of JavaScript research results https://2019.stateofjs.com/ [accessed - 23.05.2020].

[12] Bergvall-Kåreborn, B., Mirijamdotter, A., & Basden, A. (2004). Basic principles of SSM modeling: an examination of CATWOE from a soft perspective. *Systemic Practice and Action Research*, *17*(2), 55-73.

[13] Bjerke, O. L. (2008). Soft Systems Methodology in action: A case study at a purchasing department. *rapport nr.: Report/IT University of Göteborg 2008: 034*.

APPENDIX

N	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49

Table 12. RI (random index or consistency index of a random-like matrix) proposed by T. Saaty.