# COMPARISON OF FRONT-END FRAMEWORKS FOR WEB APPLICATIONS DEVELOPMENT

## ABSTRACT

Modern web applications, due to the functionalities they provide in their user interfaces, have a complex program structure. Manually writing a program code, due to the complexity of the entire application, can result in uneven quality and content of individual application parts. Maintaining such developed applications is more difficult. Because of this, web applications are often developed by using different frameworks. A framework allows structuring, simpler and more uniform program script writing, and thus easier web application maintenance. There are various frameworks that can be used in the development of web applications, for different parts of the application. Those analyzed in this paper are used in the development of front end parts of web applications. According to their design, a web application can be developed as the Multi Page (MPA) or the Single Page (SPA). This paper explains the difference between MPA and SPA web applications. The advantages and disadvantages of MPA are demonstrated in relation to SPA web applications. Required characteristics that the framework should have in order to be optimized for creating MPA and SPA web applications are set. The hypothesis has been tested: There is a framework that is optimized for the development of both MPA and SPA applications. Possibilities, architecture and development techniques of a web application using front end frameworks, as well as the suitability of such frameworks for the development of MPA and SPA web applications have been analysed. Choosing a framework for the hypothesis testing has been performed based on the popularity of available frameworks. The required characteristics have been analyzed on the three most popular frameworks: Angular, Vue.js and React-js. It has been shown that the Vue.js framework is the most optimized framework for the development of both MPA and SPA applications.

Key words: SPA, MPA, framework

1. INTRODUCTION

In recent years, there has been a great demand for highly sophisticated and complex web applications, in order to replace the old desktop application in all areas. There is also the need to create desktop applications for mobile devices. The complex web applications can be the so called Multi Page applications (MPA) and Single Page applications (SPA).

Web application development is often based on the use of a certain framework (FW). Are frequently used FWs for creating web applications in line with the development of both MPA and SPA applications? Which FW is easier to use in developing web applications? Are FWs suitable for managing front end parts of a web application?

The purpose of the paper is to demonstrate the suitability of the available FWs for the creation of MPA and SPA web applications. The aim is to define key features that enable a more customized development of MPA and SPA web applications.

Hypothetically, it can be said that there is a front end FW which is optimized for creating MPA and SPA applications.

This paper will explain the meanings of MPA and SPA, and the need to develop such web applications. The issues affecting the development of MPA and SPA applications will be defined. Frequently used FWs will be analysed by qualitative assessments of each issue affecting the development of MPA and SPA applications.

1. RELATED WORK

To create a good user experience, it is important to choose a suitable web application architecture. A more detailed analysis of architectural features and user needs is often required when selecting between a Single Page Application (SPA) and a Multi Page Application (MPA). MPA applications are usually intended for larger systems with a large number of different types of services, which prefer more types of interactions with their visitors. SPA is a newer approach to web application development and is often used in the development of simpler applications with less content (Dimi, 2017).

Multi-page applications work in a “traditional” way. Any change in the browser (e.g.. display or transmission of data) involves retrieving new pages from the server (Neoteric, 2016). Routes are registered on the server, and each HTTP request from the client to server involves retrieving a new HTML page. This means that the request sent to the server will always retrieve the page showing the results of the request (so called view), or an error. Most of the application logic is on the server, and the client is only the recipient of the retrieved page.

This web application development approach is suitable for developing smaller applications and when, for example, Javascript5 is used for simpler activities performed over the loaded page (user interface) or for animation development. However, if there is a need to create a more demanding user interface, the page may become very complex and it can be loaded with lots of data and program elements (Javascript is often used) that load the browser and the client computer. Since the creation of complex pages on the server and their transfer and presentation to the client requires a lot of time and degrades user experience, at the beginning of 2000s the MPA was improved by introducing AJAX6, which enabled refreshing only parts of a page rather than entire pages. This technique has helped to improve user experience, but it has also enabled the development of more complex web pages, making the management of the source code more complex. This is one of the reasons why FWs appear (Shimanovsky, 2016).

SPA is a complete web application with only one page that serves as a “shell” for all parts of the user interface. Most resources (Javascript, HTML57, CSS8) are loaded only once throughout the application work cycle, and data is transmitted from previous to new state. The initial HTML document uploaded at the first opening of an application represents the starting point for the rest of the application. Each subsequent part is loaded dynamically and independently of the “shell”, without re-loading the whole page, giving the user the perception that the page has changed. The shell is mostly minimal in structure and often contains a single, empty tag (<div>) that will “host” the rest of the application content (Emmit, Scott, 2016).

After the initial request to the server, the browser loads the complete HTML page. Each subsequent client and server interaction is performed using AJAX technique, which means that the browser only updates a part of the page that changes (data) without refreshing the entire page (Saxena, 2014).

The main difference between MPA and SPA work cycles is in the nature of the request and the response, i.e. one MPA work cycle ends by receiving the response, and the SPA work cycle lasts and depends on the work of the user through the user interface. SPAs use AJAX when submitting a request to a server, and as a response they receive data (usually in JSON9) and receive small parts of HTML to view received data. Once the data arrives from the server, the client side will form the received content and display it at a specific location on the shell.

Some of the advantages of the SPA compared to the MPA are the following:

• Quick Response Time - Since SPA takes on the structure of the site in advance, there is no need for constant requests to obtain a new site from the server. When a user clicks on an area, changes are made “instantly”, which gives a sense of interaction with mobile or desktop applications. The user does not have to wait or, at least, does not have that feeling of waiting. This is a great advantage and the reason why SPAs are so popular today.

• Separating presentation from business logic – The code that manages the user interface behaviour is contained on the client side instead on the server. This allows the developer to focus on what is important to the user experience and to the critical business logic elements on the server. Thus, mixing presentation and business logic is more difficult, allowing you to maintain and update each side separately.

• Faster and easier data transfer (bandwidth) - Transactions with the server are easier and faster because, after initial delivery, only data is sent to or received from the server (Emmit, Scott, 2016, 13).

• Possibility of offline support - Since SPA uses Javascript running on the client side, the connection to the server is not necessary all the time. It is possible to ensure that the application still works when the user is not connected or temporarily does not use the Internet connection. In that case, local data on the client will be synchronized with those on the server (Apps Team, 2013).

There are certain disadvantages of the SPA in relation to the MPA:

• Search engine optimization (SEO) is difficult to implement. It refers to optimizing a website (application) to achieve the highest rank in ranking content on search engines (Fink, Flatow, 2014). Each search engine has three functions:

1. crawling – it refers to the disclosure of web page information. This includes location scanning and collecting details on each page: headlines, images, keywords, other related pages, etc.

2. indexing – processing and storing the data collected through crawling in the database.

3. serving – the purpose of this function is to retrieve the relevant content at the time of the user’s query in the search engine (Bruce, 2016).

When a user performs a specific action on the user interface in the SPA, and new data needs to be downloaded, Javascript will adjust only part of the page without refreshing and opening a new one. The problem arises because crawling involves scanning content on the basis of URLs and page changes, and the SPA has only one initial page. The crawler will not know that the page has changed (Mikowski, 2014). SEO for SPA has been improving every day. Today, there are tools and ways that enable the visualization of parts of an application on the server so that the crawler can see what the user sees. However, this is still in the development and more difficult to manage than in the other (MPA) approach (Zanon, 2015).

• Javascript must be enabled - If the user disables Javascript in the browser, the application will not be shown the way it should. Since such users account for 1.3%, this can be a problem (Hein, 2010).

• Security requires more work - this principle is not unsafe, but since this way of creating applications is relatively new, some security issues are partially resolved and more effort is needed to study ways and best practices to address specific security issues.

3. RESEARCH METHODOLOGY

3. 1 Analyzed platforms and frameworks

The following will explain the issues that will be analyzed for FW adaptation checking for MPAs and SPAs development. Given that there are differences between MPA and SPA applications, there are also different needs in the developmnet of such applications. For this reason, separate issues will be defined for the frameworks adaptation analysis for the development of MPAs and SPAs.

Table 1. Questions for analysis regarding the MPA

Question Explanation MPA

Is only js import sufficient to get started?

For MPAs, which often have a lot of pages on which they control smaller parts of DOM1, a convenient and easy startup can be just by adding scripts. This also refers to the creation of simpler components, and the FW that enables this is considered more manageable and easier to use. (Burgess, 2016)

Does the framework include too much overhead?

Since choosing a particular framework often entails a large number of already built-in plugins, this issue concerns the ability of the FW to provide the developer with a higher degree of control over the size of the application, i.e. by selecting only the necessary parts of the FW (Neuhaus, 2017). i.e. whether writing a lot of code is also necessary for creating smaller functionalities?

Is the workflow necessary?

Does the framework require some workflow to be used when creating each page.

Server – side rendering

If the goal is to build a Multi-Page application, one of the reasons might be a better optimization the application in the search engines. Server-side rendering (SSR) refers to the execution of the application on the server, allowing the crawlers to see the content. Therefore, this will provide the answer on the question whether FW has the SSR option. (React Community, 2016)

SPA

The ability to write and maintain a large quantity of complex code

A cleaner, more organized JavaScript code and a well-modulated architecture, in which each part of the application forms a separate part, is a step forward in building a scalable and sustainable SPA (Takada, 2013) (Bruce, 2016). Using modules also helps to achieve data integrity, program code organization, and avoiding recurring names. Without such an approach to building SPA, the work with global variables and functions, above all, would soon become unmanageable (Emmit, Scott, 2016, 129). In order for the above to be possible, the following is necessary: a) workflow – which will enable the creation of applications by modules and which, by providing the user with the right tools and following the best practices, will properly bundle all the modules into one js document; b) the existence of the best practice - available in the documentation and describes the naming of files and folders since it is the only way for the application to be sustainable.

Question Explanation MPA

Should the developer rely on a large quantity of third party packages?

This issue refers to the question whether the FW is complete, i.e. whether all the packages needed to build a SPA are already embedded in it or is there an officially issued package, or is it necessary to receive community packages, which could be a problem. Can we rely on the maintenance (promptness of updates) of the package? This issue is very important when it comes to building a SPA where, since it is a larger and more complex Javascript application, all these packages will really be needed and there should be less worrying about finding other packages for missing features. This just means more dependency to worry about and more potential points of failure (Köhr, Schlaudraff, 2017).

What is the possibility of compressing the application, minimizing the bundle file?

One of the challenges of building SPA is the speed of initial opening of the site. In order for speed to be as fast as possible, the bundle file, or script that must be downloaded in order to launch the application, should be reduced to the smallest possible size. It refers to the process and simplicity of the transformation of the development program code into a concise production one (Dias, 2016).

Data state management

Development of the SPA is often complex because the application work cycle does not stop, and user activity on the application results in changes in the application state on the client. Some changes in the application state come through AJAX, but most of the changes do no come from the server, but from the local variables in the application. Various application state managers can be used for coherent DOM update. This question will answer whether there is a clear way to utilize the FW to ensure DOM state management. Is there an officially issued package or it is on the developer who, in such a case, must rely on the community (Botto, 2016)? Source: analysed by authors

The three most popular Javascript front end FWs will be compared, the popularity of which continues to grow. The figure below shows the ranking of all available Javascript FWs. Since this work refers to FWs for the front end side, the following are isolated and will be analysed: Angular, React and Vue.js.

React, with over 86,000 Github stars (https://github.com/facebook/react), is considered probably the most popular Javascript front end FW. Vue.js, with about 81,000 Github stars (https://github. com/vuejs/vue), is one of the fastest growing when it comes to popularity (developed only in 2016). On the other hand, Angular with its huge backdrop, Google (from whom it has been developed) and Microsoft, whose language it uses (Typescript), is one of the most stable. The analysis (Figure 1) was carried out among the six most well-known Javascript front end FWs, and it was established that only the three mentioned ones were currently gaining popularity while the others were stagnating.

Vue.js is today one of the fastest growing Javascript FWs when it comes to popularity. On its official pages it is described as accessible, comprehensive and high performing FW for development of interactive interfaces (Vue, 2018-1). Since FW is component-oriented and the component, after a certain internal behaviour / computation, returns the UI template in the form of output, all of these components are reusable within the page, or within other components. Vue.js uses virtual DOM which does not exist in the browser but in the memory, and the result is a much faster access than in the actual DOM. Such updated virtual DOM is in the end presented as actual DOM (Andersen, 2017). This FW is most popular in the Far East, and some of the companies that use it are Alibaba, Xiaomi, Baidu, Tencent etc. (Clockwise Software, 2017).

React is presented as a “Javascript library for building user interfaces”. It was released in March 2013 by Facebook, which uses the React components on several of its pages, instead of one SPA (Neuhaus, 2017). The same as Like Vue.js, React uses a component-based architecture, which makes it possible to write code in a simpler and more sustainable way (React, 2018-1). Although not mandatory, the components are usually written in JSX (Javascript XML), a React-specific extension of Javascript, which allows the use of HTML within Javascript (React, 2018-2). Since it uses virtual DOM, React creates the data structure in the memory, calculates differences between virtual and real DOM, and then updates the actual DOM in the browser in a very fast and efficient way (Kurian, 2017).

Angular is FW for creating client applications in HTML and JavaScript or in a language such as TypeScript compiled into JavaScript. It was developed by Google in 2010 as AngularJS, and in 2014 it was completely reworked and rewritten and since then it has been operating under the name Angular. FW consists of several libraries, some of which are basic and some optional. Angular applications are developed by writing HTML templates using “angularized” tagging, writing component classes to manage these templates, adding application logic to services, and registering components and service in modules. The application is launched by loading the root module. Angular takes over the presentation of the application content in the browser and responds to the user’s interaction according to the instructions you have entered (Angular, 2018-1). Angular 4 will be used in the analyses in this paper.

# Web应用程序开发的前端框架比较

## 摘要

由于现代网络应用程序在用户界面中提供功能，它们具有复杂的程序结构。由于整个应用程序的复杂性，手动编写程序代码可能会导致各个应用程序部分的质量和内容参差不齐。维护这样已经开发好的应用程序更加困难。正因如此，Web应用程序通常使用不同的框架来开发。框架允许结构化、更简单和更统一的程序脚本编写，从而更容易维护Web应用程序。对于应用程序的不同部分，可以在Web应用程序的开发中使用各种框架。本文所分析的内容在Web应用程序前端部分的开发中得到了应用。根据它们的设计不同，Web应用程序可以开发为多页(MPA)或单页(SPA)。本文解释了MPA和SPA Web应用程序之间的区别。结合SPA Web应用，展示了MPA的优势和劣势。设置了框架来创建MPA和SPA Web应用程序进行优化所需的特性。这一假设已经得到了检验：有一个针对MPA和SPA应用程序开发进行了优化的框架。分析了使用前端框架的web应用程序的可能性、体系结构和开发技术，以及这些框架对于开发MPA和SPA web应用程序的适用性。选择用于假设检验的框架是基于可用框架的受欢迎程度。在三个最流行的框架上对所需的特征进行了分析：Angular、Vue.js和React-js。事实证明，Vue.js框架对于MPA和SPA应用程序的开发都是最优的框架。  
关键词：SPA、MPA、框架

1. 介绍

近年来，为了在所有领域取代旧的桌面应用，人们对高度复杂的Web应用程序有了很大的需求。还需要为移动设备创建桌面应用程序。复杂的Web应用可以是所谓的多页应用(MPA)和单页应用(SPA)。

Web应用程序开发通常基于某个框架(FW)的使用。在创建Web应用程序时经常使用的框架们是否与MPA和SPA应用程序的开发保持一致？在开发Web应用程序时，哪个框架更容易使用？框架们是否适合管理Web应用程序的前端部分？

本文的目的是论证现有的框架们对于创建MPA和SPA web应用程序的适宜性。其目的是定义关键功能，使得MPA和SPA网络应用程序的开发更加定制化。

假设存在一个前端框架，该前端框架针对创建MPA和SPA应用程序进行了优化。

本文将解释MPA和SPA的含义，以及开发此类Web应用程序的必要性。将确定影响MPA和SPA应用开发的问题。对影响MPA和SPA应用开发的每个问题进行定性评估，对经常使用的框架进行分析。

1. 相关工作

为了创造良好的用户体验，那么选择一个合适的Web应用程序架构是非常重要。在单页应用程序(SPA)和多页应用程序(MPA)之间进行选择时，通常需要对体系结构功能和用户需求进行更详细的分析。MPA应用程序通常适用于那些具有大量不同类型服务的大型系统，这些系统喜欢与访问者进行更多类型的交互。SPA是一种较新的Web应用程序开发方法，通常用于开发内容较少的、更简单的应用程序(Dimi，2017)。

多页应用程序以一种“传统”方式工作。浏览器中的任何更改(例如，显示或传输数据)涉及从服务器检索新页面(Neoteric，2016)。路由在服务器上注册，从客户端到服务器的每个HTTP请求都涉及到检索一个新的HTML页面。这意味着发送到服务器的请求将始终检索显示请求结果(视图)或错误。大部分应用程序逻辑都在服务器上，而客户端只是检索到的页面的接收方。

这种web应用程序开发方法适合较小的应用程序。例如，javascript用于在加载的页面(用户界面)上执行更简单的活动或用于动画开发。但是，如果需要创建要求更高的用户界面，页面可能会变得非常复杂，它可能会加载大量数据和程序元素(通常使用Javascript)来加载浏览器和客户端计算机。由于在服务器上创建复杂页面并将其传输和呈现给客户端需要大量时间，并且降低了用户体验，因此在本世纪初引入了AJAX，从而改进了MPA，它只允许刷新页面的一部分而不是整个页面。这项技术帮助改善了用户体验，也使得开发更复杂的网页成为可能，也使得源代码的管理变得更加复杂。这就是框架们出现的原因之一(Shimanovsky，2016)。

SPA是一个只有一个页面作为用户界面所有部分的“外壳”的完整Web应用程序。大多数资源(Javascript、HTML5、CSS)在整个应用程序工作周期中只加载一次，数据从以前的状态传输到新的状态。第一次打开应用程序时上传的初始HTML文档代表应用程序其余部分的起点。后续的每个部分都是独立于“外壳”动态加载的，无需重新加载整个页面，让用户感觉页面已经改变。Shell的结构大多很简单，通常只包含一个空标记(<div>)，它将“托管”其余的应用程序内容(Emmit，Scott，2016)。

在向服务器发出初始请求之后，浏览器加载完整的HTML页面。后续客户端和服务器的每个交互都是使用AJAX技术执行的，这意味着浏览器只更新(数据)的页面的一部分，而不刷新整个页面(Saxena，2014)。

MPA和SPA工作周期之间的主要区别在于请求和响应的性质，即一个MPA工作周期以接收响应结束，而SPA工作周期持续并依赖于用户通过用户界面所做的工作。SPA在向服务器提交请求时使用AJAX作为响应，它们接收数据(通常是JSON)并接收一小部分HTML以查看接收到的数据。一旦数据从服务器到达客户端，客户端将形成接收到的内容，并将其显示在shell的特定位置。

与MPA相比，SPA的一些优势如下:

* 快速响应时间——由于SPA预先承担了站点的结构，因此无需不断从服务器请求获取新站点。当用户点击某个区域时，会“立即”进行更改，这给人一种与移动或桌面应用程序交互的感觉。用户不必等待，或者至少没有等待的感觉。这是一个很大的优势，也是今天SPA如此受欢迎的原因。
* 将表示与业务逻辑分开——管理用户界面行为的代码包含在客户端，而不是服务器上。这使得开发人员将重点放在对用户体验和服务器上的关键业务逻辑元素重要的内容上。因此，混合表示和业务逻辑更加困难，允许您分别维护和更新每一方。
* 更快更方便的数据传输(带宽)——与服务器的交易变得越来越容易，因为在初始交付之后，只向服务器发送数据或从服务器接收数据(Emmit，Scott，2016，13)。
* 离线支持的可能性——由于SPA使用在客户端运行的Javascript，因此无需始终连接到服务器。可以确保在用户未连接或暂时不使用互联网连接时，应用程序仍可正常工作。在这种情况下，客户端上的本地数据将与服务器上的数据同步(Apps Team，2013)。

与MPA相比，SPA有一些缺点：

* 搜索引擎优化(SEO)很难实现——它指的是优化一个网站(应用程序)，使其在搜索引擎上的内容排名达到最高(Fink，Flatow，2014)。每个搜索引擎都有三个功能：

1. 爬行——指的是网页信息的泄露。这包括位置扫描和收集每个页面的详细信息：标题、图像、关键字、其他相关页面等。
2. 索引——在数据库中处理和存储通过爬行收集的数据。
3. 服务——此功能的目的是在搜索引擎中检索用户查询时的相关内容(Bruce，2016)

当用户在SPA中的用户界面上执行特定操作，并且需要下载新数据时，Javascript将只调整页面的一部分，而不刷新和打开新页面。之所以会出现这个问题，是因为爬行涉及根据URL和页面更改扫描内容，而SPA只有一个初始页面。爬虫不会知道页面已经改变(Mikowski，2014)。

SPA的搜索引擎优化每天都在改进。如今，有一些工具和方法可以使服务器上的应用程序部分可视化，以便爬虫程序可以看到用户看到的内容。然而，这仍然处于开发阶段，比其他(MPA)方法更难管理(Zanon，2015)。

* 必须启用JavaScript——如果用户在浏览器中禁用Javascript，应用程序将不会以应有的方式显示。由于这类用户占1.3%，这可能是个问题(Hein，2010)。
* 安全性需要更多的工作——这一原则并不是不安全的，但由于这种创建应用程序的方式相对较新，一些安全问题已经部分解决，需要更多的努力来研究解决特定安全问题的方法和最佳实践。

1. 研究方法。

3.1分析的平台和框架

接下来将解释框架适应MPA和SPA发展所要分析的问题。鉴于MPA和SPA应用之间存在差异，这些应用的开发也有不同的需求。基于这个原因，将为MPA和SPA开发的框架适应性分析来定义单独的问题。

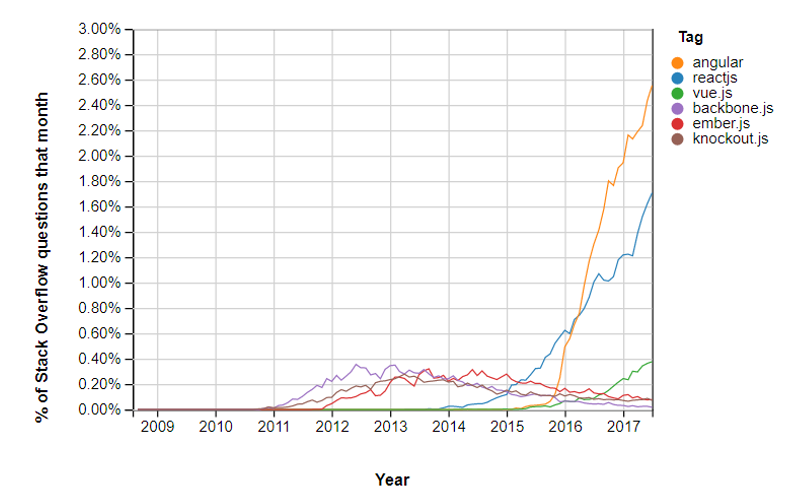
表1.关于MPA的分析问题

|  |  |
| --- | --- |
| 问题 | 解释 |
| MPA | |
| 只要导入js就足够开始了吗? | 对于通常有很多页面来控制DOM较小部分的MPA来说，只需添加脚本即可方便轻松地启动。这也指的是创建更简单的组件，实现这一点的框架被认为更易于管理和使用。(Burgess，2016) |
| 该框架是否包含太多开销？ | 由于选择一个特定的框架通常需要大量已经内置的插件，因此这个问题涉及到框架是否能够为开发人员提供对应用程序的大小有更高程度的控制，通过只选择框架的必要部分(Neuhaus，2017)。也就是说，编写大量代码是否也是创建较小功能所必需的？ |
| 工作流程是否必要？ | 框架在创建每个页面时是否需要使用一些工作流？ |
| 服务器端——渲染 | 如果目标是构建一个多页应用程序，其中一个原因可能是在搜索引擎中对应用程序进行了更好的优化。服务器端呈现(SSR)指的是在服务器上执行应用程序，允许爬行器查看内容。因此，这将为框架是否有SSR选项的问题提供答案。(Reaction Community，2016) |
| SPA | |
| 编写和维护大量复杂代码的能力 | 更干净、更有组织的JavaScript代码和调节良好的体系结构(其中应用程序的每个部分形成一个单独的部分)是构建可扩展和可持续的SPA的一步(Takada，2013)(Bruce，2016)。使用模块还有助于实现数据完整性、程序代码组织和避免重复名称。如果没有这样一种构建SPA的方法，处理全局变量和函数的工作，最重要的是，很快就会变得难以管理(Emmit，Scott，2016,129)。为了实现上述目标，必须具备以下条件：a)工作流程-这将使得通过模块创建应用程序成为可能，并通过向用户提供正确的工具并遵循最佳做法，将所有模块恰当地捆绑到一个JS文档中；b)存在最佳做法-文档中提供了最佳做法，并描述了文件和文件夹的命名，因为这是应用程序可持续的唯一途径。 |
| MPA | |
| 开发商是否应该依赖于大量的第三方包？ | 这个问题指的是防火墙是否完整的问题，即构建SPA所需的所有包是否已经嵌入其中，或者是否含有官方发布的包，或者是否需要接收共同包，这可能是一个问题。我们可以依赖包的维护(更新迅速)吗？当涉及到构建SPA时，这个问题非常重要，因为它是一个更大、更复杂的Javascript应用程序，所有这些包都是真正需要的，不应该那么担心为缺少的功能找到其他包。这只是意味着需要担心更多的依赖性和更多的潜在故障点(Köhr，Schlaudraff，2017)。 |
| 压缩应用程序、最小化捆绑包文件的可能性有多大？ | 建设SPA的挑战之一就是网站的初始构建速度。为了使速度尽可能快，应该将捆绑的包文件或是为了启动应用程序而必须下载的脚本减小到尽可能小。它指的是将开发程序代码转换为简明的生产代码的过程和简单性(Dias，2016)。 |
| 数据状态管理 | SPA的开发通常很复杂，因为应用程序的工作周期不会停止，而应用程序上的用户活动会导致客户端上的应用程序状态发生变化。应用程序状态的一些更改是通过AJAX进行的，但大部分更改不是来自服务器，而是来自应用程序中的局部变量。各种应用程序状态管理器可用于一致的DOM更新。这个问题将回答是否有一种明确的方法来利用框架来确保DOM状态管理。有没有官方发布的套餐，还是取决于开发者，在这种情况下，开发者必须依赖社区(Botto，2016)？ |

我们将对三个最流行的Javascript前端框架进行比较，它们的受欢迎程度还在继续增长。下图显示了所有可用的Javascript框架的排名。由于这项工作涉及前端的框架，因此以下内容是孤立的，并将对其进行分析：Angular、React和Vue.js。

React，超过86,000个Github点赞(https://github.com/facebook/react)，被认为可能是最受欢迎的Javascript前端框架。Vue.js，拥有约81,000颗Github点赞(https://github.。Com/vuejs/vue)，是人气增长最快的网站之一(2016年才开发出来)。另一方面，Angular有着巨大的背景，谷歌(它就是从谷歌开发出来的)和微软，所使用的语言（Typescript)，是最稳定的之一。这项分析(图1)是在六个最知名的Javascript前端框架中进行的，结果表明，只有提到的三个框架目前正在流行，而其他的则停滞不前。

图1.6个最知名的Javascript框架的受欢迎程度



就受欢迎程度而言，Vue.js是目前增长最快的Javascript框架之一。在其官方页面上，它被描述为可访问、全面和高性能的框架，用于开发交互界面(Vue，2018-1)。由于框架是面向组件的，并且组件在经过一定的内部行为/计算后以输出的形式返回UI模板，因此所有这些组件都可以在页面中或其他组件中重用。Js使用的虚拟DOM不存在于浏览器中，而是存在于内存中，其结果是访问速度比实际的DOM快得多。这种更新的虚拟DOM最终被呈现为实际的DOM(Andersen，2017)。这款框架在华东地区最受欢迎，一些使用它的公司有阿里巴巴、小米、百度、腾讯等。(Clockwise Software，2017)

React被描述为“用于构建用户界面的Javascript库”。它是由Facebook于2013年3月发布的，它在几个页面上使用React组件，而不是一个SPA(Neuhaus，2017)。与Vue.js一样，React使用基于组件的架构，这使得以更简单、更可持续的方式编写代码成为可能(Reaction，2018-1)。虽然不是强制性的，但这些组件通常是用JSX(Javascript XML)编写的，JSX是Javascript的一个特定于React的扩展，它允许在Javascript中使用HTML(React，2018-2)。因为它使用虚拟DOM，所以Reaction在内存中创建数据结构，计算虚拟DOM和真实DOM之间的差异，然后以非常快速和高效的方式更新浏览器中的实际DOM(Kurian，2017)。

Angular是用于使用HTML和JavaScript创建客户端应用程序的框架，或使用编译成JavaScript的语言，比如TypeScript。它是由谷歌在2010年开发的，名为AngularJS。2014年，它被彻底修改和重写，从那时起，它就一直以AngularJS的名字运行。该框架由几个库组成，其中一些是基本的，另一些是可选的。Angular应用程序是通过使用“angularized标记编写HTML模板、编写组件类来管理这些模板、向服务添加应用程序逻辑以及在模块中注册组件和服务来开发的。该应用程序通过加载根模块来启动。Angular接管应用程序内容在浏览器中的呈现，并根据用户输入的指令响应用户交互(Angular，2018-1)。在本文的分析中将使用Angular 4。