Impact of Application Architecture on Performance

Kharazyan Hayk A.
lead developer Higher School of Economics
E-mail: haykking@gmail.com

Abstract

The article discusses the impact of application architecture on performance. It is noted that one of the important characteristics of any modern web application is its performance. The most popular web application implementation architectures are considered in terms of application performance and security.

Keywords: application architecture, web applications, performance, software product

In the modern IT-sphere, software implementation is carried out through various development tools, methodology, approaches, as well as software architecture. Software architecture is driven by a number of evaluation factors for any application, such as ease of implementation, ease of use of the application, and performance of the finished software product. The performance indicator of the program is associated with the architecture used, and within the framework of this article it is planned to consider the existence of such a relationship by considering several of the most common architectures for implementing web applications.

The architecture of any software product can be described in detail as a system structure, which has a set of components, the functional purpose of which is to perform a certain functionality. The purpose of an architecture is to organize the interaction between these components in order to provide the required level of functionality. This kind of organization of functionality is often referred to as grouping components based on functional areas.

The impact of an application's architecture can be assessed based on various criteria and from different perspectives. So, from the user's point of view, criteria such as responsiveness and convenience when updating data and navigating through pages, as well as the convenience of the application interface, are used. Also, this set of criteria should include the ability to quickly and conveniently work with sections of the site, and the ability of the application to work in the absence of a network connection. For the developer, the speed of application development, its performance, scalability and testability are more important, that is, the convenience of developing and finalizing the application, testing its functionality, etc. However, from the point of view of the customer, slightly different criteria are put forward. First of all, these include the ability to search for an implemented application through any search engine, the level of costs for ensuring the operability of the application, the ability to change the functional composition of the application, taking into account time and financial costs, as well as the security of the application [8].

To analyze the impact of the web application architecture used on its performance, it is necessary to consider the most common architectures and evaluate them based on the performance criterion [6].

One of the most common web application architectures at the current time is the "Server — Page" type architecture. The essence of the implementation of the application based on this architecture is quite simple — the server generates the necessary content and sends it to the client in the format of a full-fledged HTML page.

For this architecture is also often used the name "Web 1.0" as the name of the architecture for implementing web applications. This architecture is convenient for developers in terms of security, testability, and development speed, since all application logic is actually located on the server platform. The generated content is known in advance, hacking the application is impossible without gaining access to the

server platform, and to simplify development procedures, one of the common server languages or frameworks can be used. From the point of view of the client, this architecture is not very convenient, due to the fact that it requires the transfer of large amounts of data, and any action with data changes requires updating the entire page [7].

However, in terms of performance, this architecture is not the most perfect. Large amounts of data being transferred result in lower performance levels. Due to the need to completely update the page in the event of any data change, the time period required for updating them increases. As the load increases, sooner or later there will be a moment when you need to implement load balancing mechanisms. Passing data in HTML format means that it is not possible to change individual blocks in the page structure.

The next architecture under consideration is an architecture that can be conditionally called “Widgets generated using the JavaScript language”. Its essence lies in the fact that the structure of the displayed page of a web application is divided into independent blocks, which are called widgets. In fact, it is a modified version of the architecture of the first type [3].

The operation of the application based on this architecture occurs in the form of generated AJAX requests and generation based on this data request by the server, followed by the formation of a response. Depending on the requirements, this data can be presented either as a full-fledged HTML page or as a script that generates the required area of the page. This version of the web application architecture allows you to selectively load page blocks, eliminates the requirement to use separate frameworks on the client side. By reducing the level of interactivity, the speed of the implementation of the software product increases, and the functionality of the application becomes cheaper and more reliable.

The main advantage of this architecture is the fact that the client receives from the server the set of data that must be updated in the required part of the page (widget). At the same time, these widgets are functionally separated, and updating one of the widgets or a set of widgets in the page will not affect the entire page [1].

In this case, the amount of data transferred during the refresh of the web page is reduced. This has a positive effect on the responsiveness of the application in the process of working with it. But due to the structure of the page in the format of a set of widgets in the form of a UI template, the first load of such a page will take a little longer, compared to the implementation of this page entirely using HTML. It turns out that the first time the page content is loaded in an application or web service, this page will load a little longer, but subsequently it will work more quickly. Speaking about the development of an application based on this architecture, it requires the use of not only server-side technologies, but also client frameworks, as well as the formation of certain web services on the server.

The amount of work on the program code testing is increasing due to the need to test both server and client code. And the fact that part of the logic of the service or application is transferred to the client side in the form of scripts, the security level is reduced due to the possibility of measuring this code by attackers [5].

The performance of applications built on the basis of this architecture will already be higher. This is justified by the reduction in time and resources that will be required to generate page content. However, at the same time, more time is spent on extracting data from databases and their subsequent preparation into templates. An extended type of architecture using data transmission with the JSON format reduces the amount of transmitted traffic, but leads to an additional abstraction layer within the application, which leads to some increase in the degree of protection of the application, but reduces the performance level.

The third architecture under consideration is Service Oriented Web Pages.

The essence of this version of the web application architecture is to generate a special HTML page by the service. It is a container that contains executable JavaScript code. This code, when accessing the

necessary data sources, receives all the required information, from which the content of the web application page will be subsequently generated. In fact, this is a development of the previous architecture, bringing it to a full-fledged level, when the application becomes independent, and at the same time quite complicated, since part of the executable code is already placed on the client side [2].

The amount of information transferred is reduced to almost a minimum, so the maximum responsiveness of the application is achieved, so the application works as quickly as possible compared to the previous two architectures. Using JavaScript allows you to generate any appearance of the application interface. When implementing an application, the issue of processing large amounts of necessary data is considered separately — these processes are most often implemented on the server side. From the point of view of development, this architecture is more complex, since it requires more time and labor costs, and also does not have developed and high-quality tools and approaches for implementing an application based on this architecture [8].

Testing applications against this architecture is similar to testing against the previous architecture. To do this, you need to check both the server code and the client code. From a security point of view, the system is not protected from intentional modifications of the client code, for this reason exchange protection mechanisms are used, for example, the use of third-party transmission channels for the exchange of encryption keys. And due to the fact that the logic of the application is on the client side, the server performs a minimum set of functionalities, and therefore the minimum load falls on it.

Applications based on this architecture can be called high-performance, but this assessment and characteristic will be such only for modern client devices. As noted earlier, the server is loaded to a minimum, the entire load falls on the client side. The server actually performs the formation of the structure of the JS application and passes it to the client browser. For this reason, the performance of the mobile application used by the client, as well as the type of browser used, will play the most important role [4].

The next type of architecture is the serverless architecture. This architecture allows you to simplify the work on the application, since developers no longer deal with server configuration issues, they rent computing power in the cloud from a cloud service provider. Most often, this architecture is used in situations where developers do not want to be distracted by maintaining servers, and also have no idea about the expected load. In this case, in the event of a significant increase in load, it will be possible to switch to another tariff plan without wasting time on server upgrades.

The next type of architecture is Progressive Web Applications. This architecture is characterized by a fairly extensive functionality, ease of deployment, as well as a fairly high level of reliability, allows you to implement an application that can function with various web browsers, as well as on various devices. Applications based on this architecture are implemented in the format of native applications that can provide pop-up notifications on mobile devices, offline access functions, as well as the ability to install the application directly on the device.

Based on the analysis, it should be noted that there is a dependence associated with the amount of data transferred between the server and the client of the web application, respectively, it is worth talking about the dependence of the application performance level on the type of architecture. At the same time, from the point of view of information security, any architecture has a whole set of tools that ensure the proper level of application security.

In conclusion, it should be noted that if there is a direct dependence of application performance on the type of architecture used, in the case of a real project, the decisive factor may be user experience, security, etc. A developer can optimize the architecture or use a hybrid architecture within his own project, which will undoubtedly affect not only performance, but also other important parameters of the application

being implemented. At the same time, it is important to put information security on a par with performance in the list of goals, since any high-performance application will quickly lose popularity in the event of a data leak.

References

1. Bogatyrev V. A. Reliability of information systems: textbook for secondary vocational education / V. A. Bogatyrev. M.: Yurait Publishing House, 2022. 318 p.

2. Dubovik E. V. Web in practice. CSS, HTML, JavaScript, MySQL, PHP for fullstack developers / E.V. Dubovik, A.P. Nikolsky. Moscow: Science and technology, 2021. 432 p.

3. Kazarin O. V., Shubinsky I. B. Reliability and security of software: a textbook for universities. M.: Yurait Publishing House, 2022. 342 p.

4. Lavrishcheva E. M. Software engineering and programming technologies for complex systems: a textbook for universities / E. M. Lavrishcheva. 2nd ed., rev. and additional M.: Yurait Publishing House, 2022. 432 p.

5. Laschevski T. Cloud architectures: development of sustainable and economical cloud applications / T. Laschevski, K. Arora, E. Farr, P. Zonuz. St. Petersburg: ID Piter, 2022. 320 p.

6. Lukyanov P.B. Development and implementation of portal solutions / P.B. Lukyanov. M.: Prometheus, 2020. 166 p.

7. Robert M. Pure architecture. The art of software development / M. Robert. St. Petersburg: ID Piter, 2022. 352 p.

8. Chernyshev S. A. Principles, patterns and methodologies of software development: a textbook for universities / S. A. Chernyshev. M.: Yurait Publishing House, 2022. 176 p.