Kostyantyn Tykhonov, Yurii Gordienko.

METHOD FOR OPTIMIZATION COMPUTING RESOURCES ON THE BASIS OF SPECIALIZED HYBRID SOLUTIONS

Abstract. The article discusses the main architectural approaches to building web applications, data for comparing different architectural approaches and the method of migration between different software architectures by distributing the functionality of the program between several applications.

Keywords: cloud, faas(function as a service, переклад укр. - функція як сервіс), serverless, microservice

Acute problem. The development of software applications and the need for constant scaling of these solutions complicates the process of optimization and improvement of implemented software solutions. A set of methods and software solutions should improve and enhance the process of optimizing program code.

Target setting. Limited human resources and time in the optimization and expansion of software applications and systems, leads to the need for a number of methods and software solutions to optimize and improve program code. Such solutions remove and optimize duplicate software code, increase its performance, offer newer, more optimal architectural solutions.

Actual scientific researches and issues analysis. As a result of the analysis of literature sources devoted to the optimization of computing resources, several main approaches to solving this problem were identified - optimization of server equipment and analysis and performance of a software solution, which is implemented using a programming language.

Uninvestigated parts of general matters defining. This article focuses on methods and tools for analyzing and optimizing software solutions by separating part of the program code and building a separate application using a different architecture.

The research objective. The task is to find the optimal method of analysis of the software application to separate certain parts for further construction of a separate software application.

The statement of basic materials. In the offered article mechanisms and the basic algorithms of the analysis of a source code of the software application for allocation and construction of separate sub-applications which should increase productivity of work of the application are considered.

The method is based on a serverless approach to building applications. Applications based on micro-service or monolithic architecture will be used as a basis for optimization

Serverless approach, this is a new paradigm that promises to change the way programs are built and developed. In this approach, the program is divided into small fragments united by a common logic, which are called functions and are placed on cloud platforms [1].

The microservice approach is an architectural style that emphasizes small services that are built to perform a separate business logic and are an evolution of the traditional architectural-oriented style [2].

Performance analyze. Faas(function as a service) is a more cost-effective approach in a situation where you do not need to constantly have access to the functionality of the application. In the case where requests to the program are not a continuous task, you can see a significant gain in productivity and save on the use of machine resources [3].

Comparative analysis without server and micro service approach

Table 1

Comparative analysis of microservice and serverless approach

Aspect	**Microservice**	**Serverless**
Code and delivery	Based on services, delivery is aimed at service architecture	Focuses on tasks (functions).
Sizing	Based on the number of instances and resources required for the service. There are risks of overestimation or underestimation of the load.	Based on the number of resources to run each event.
Environment	Fully managed by the supplier with the ability to configure the customer.	Fully managed by the supplier with the ability to configure the customer.
Maintenance	Interfering with the code changes the entire service	Intervention affects only one function
Resources	Preliminary allocation of resources with the ability to allocate additional copies on demand.	No pre-allocated resources. Transparent distribution on demand
Execution	Constant standby, no limit on the duration of events	No waiting state. The function is performed as needed. Limit the maximum duration of the event
Billing	For leased resources	For performing the event

Analysis of program code. Most modern programs are written in high-level languages [4]. Very often the same problems can be solved in several ways.

The choice of one of the solutions affects a number of factors - the speed of code execution, ease of reading program code and others.

To improve the speed and quality of code analysis, there are specific instructions for writing programs. Modern systems use similar solutions in building applications, so the approach is almost always the same.

The analysis depends on the type of project (the technology used and the programming language with which it was implemented). But there are commonalities in the analysis of the project:

1. Download project files as a plain text file
2. Analysis of class content
3. Analysis of the classical structure of the project (search for controllers, repositories, services with business logic)
4. Analysis of individual dependencies.
5. Analysis of configuration files
6. Analysis of global dependencies between projects

Building dependencies. A modern program is a complex solution that is divided into several levels, in order to transfer a specific program block to a separate application and preserve its functionality, it is necessary to take into account all its dependencies, access to different classes, libraries or queries to other applications.

The selection of the software element to be separated must be performed by a person, because the frequency of use of the selected functionality must be evaluated in order to obtain the maximum performance of the separated code, and to optimize the use of computing resources.

Common stages for all modern systems [6]:

1. Implement configuration files
2. Setting up communication with other services and database
3. Create the necessary instances of classes to establish communication between different levels of the application

Building a software application and its integration. After analyzing the software application and selecting the necessary components - the system will create and integrate a new application on a server or cloud platform.

When rebuilding a monolithic application, human intervention is minimal, the distribution of all functionality between different services will reduce the continuous load on one server, and allows you to integrate each application on a separate platform.

When giving applications to the serverless architecture, it is necessary to take into account the frequency of use of the function to be separated. Due to the nature of the function, the gain in reducing the load will be only if the function is not used as often as other elements of the application [5]. It is at this stage that human analysis is very important and will affect the percentage of improved performance of the new system.

Writing application code completely repeats the process performed by the programmer. Building an application skeleton, creating classes, functions, or methods, allocating to different levels, and using files with dependencies and constants. This process is fully automated using software solutions that operate on the basic structures of the programming language used in construction.

Conclusions. The approach to optimization of applications by means of software decisions is considered. A methodological basis for optimizing applications by smooth and partial migration of the application to another architectural approach is proposed. A comparative analysis of different approaches to building software applications is given.

This approach is difficult to implement and requires individual solutions for different programming languages and libraries, as in each case the syntax of the programming language is different.

You need to consider the time to reinstall the initialized applications and modify the entire application ecosystem on the server, or add new dependencies to existing applications.

The distribution of application functionality does not always lead to increased performance of the entire system, it is necessary to take into account all the dependencies and loads on each application and its element, it is not always possible to separate a function that is used less often than others.

The presented optimization method requires improvement and reduction of human impact at the stage of analysis of the application functionality and frequency of its use.

References

1. Paweł Skrzypek, Kyriakos Kritikos. A Review of Serverless Frameworks // 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)17-20 Dec. 2018,

2. James Lewis, Martin Fowler Microservices. March 2014, [Електронний ресурс]. – Режим доступу: http://martinfowler.com/articles/microservices.html.

3. Lucas F. Albuquerque Jr. Felipe Silva Ferraz , Rodrigo F. A. P. Oliveira1 , and Sergio M. L. Galdino1 ICSEA 2017 : The Twelfth International Conference on Software Engineering Advances, Function-as-a-Service X Platform-as-a-Service: Towards a Comparative Study on FaaS and PaaS.

4. TIOBE Web-Site. URL https://www.tiobe.com/tiobe-index/

5. Istemi Ekin Akkus, Ruichuan Chen, Ivica Rimac, Manuel Stein, Klaus

Satzke, Andre Beck, Paarijaat Aditya, and Volker Hilt, Nokia Bell Labs SAND: Towards High-Performance Serverless Computing, URL: https://www.usenix.org/ conference/atc18/ presentation/akkus

6. Reda Bendraou. Spring Framework, 2018 - URL: laser.cs.umass.edu

AUTHORS

Yurii Gordienko (supervisor) – professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

Kostyantyn Tykhonov – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”