A New WEB Performance Testing Framework for Industrial Internet Platform

Siqi Zhao*, Yali Li, Rucai Liu, Ouhai Ye
China Academy of Industrial Internet, Beijing, China
**Corresponding Author.*

Abstract: Industrial Internet platforms are a core component of the development of industrial Internet. Testing of industrial Internet platforms is different from ordinary software systems. Industrial Internet platforms are generally platform systems based on the B/S architecture. Due to the special hierarchical architecture of industrial Internet platforms, their technical characteristics need to be comprehensively considered during testing. Most testers only focus on the implementation of product functions, and performance testing is often superficial, limited to basic tool operations. From the perspective of application scenarios, users can generally operate Industrial Internet platforms through browser based web pages, which poses new and higher requirements for web performance testing tools. With the development of testing technology, WEB performance testing tools have emerged one after another and each has its own characteristics. Many testing tools promote their advantages, but they do not consider the user experience of the tool. By studying the typical WEB performance testing tool Gatling, analyzing user usage issues, and designing an improvement plan for Gatling based on the performance testing execution process, Gatling has been improved. Based on retaining the high-quality features of the tool and fully considering the difficulties of users, a framework packaging was carried out to improve the usability of Gatling. This has had a profound impact on the high-quality development of web performance testing tools. We hope that web testing developers and tool researchers can think and practice more in this field.

Keywords: WEB Performance Test; Gatling; Encapsulation; Ease of Use

1. Introduction

The industrial internet platform is the core and center of the industrial internet, equivalent to the "operating system" of the industrial internet. Its essence is to build a data-driven commercial ecosystem through the comprehensive interconnection of people, machines, and objects, promote the integration and coordinated development of resources in various links of the industrial chain, and promote the formation of a new industrial production and manufacturing and digital service system [1]. The construction of industrial internet platforms is a complex and lengthy system engineering [2], so the evaluation of industrial internet is particularly important. Performance efficiency is one of the important technical indicators to measure software quality, but many application software have been launched without passing rigorous performance testing, and they simply cannot meet the needs of a large number of users [3]. Some enterprises pay more attention to functional testing for three main reasons: First, the functionality is more intuitive, and the performance efficiency is lower, which has not been paid enough attention; Second, enterprises do not fully realize the use scenarios and user volume of software products; Third, there are technical difficulties in performance testing, and most performance testers have insufficient technical capabilities. However, when the number of software user visits reaches the peak, once a performance problem occurs, it will seriously affect the normal use of software products, and even bring significant losses to enterprises and users.

There are more and more types of performance testing tools. Each tool has its own characteristics[4,5], but also has its own shortcomings, such as unfriendly display of test results, cumbersome configuration and installation, and proficient programming

language is required for script writing. Based on this background, this paper studies and analyzes the WEB performance testing tool, and improves the Gatling.

2. Introduction of Gatling

As a high-performance WEB performance testing tool, Gatling's underlying design is based on Scala language, which is mostly used to test the performance of HTTP servers and locate performance bottlenecks. It has many features, such as supporting Akka Actors and Async IO, which can achieve high performance; It can generate lightweight and rich visual test reports, and the test results are readable; It can support DSL programming language, which makes test scripts easier to develop and maintain; Supporting recording and generating test scripts, so that test scripts can be easily generated; Supporting multiple integrated development tools, such as IDEA, Maven, etc., to facilitate development; Supporting plug-ins to extend their functions, such as support for other protocols; With rich parameters, it can realize continuous pressure test, control speed pressure test, etc. Setting up and using Gatling mainly includes the following steps:

1) Configure and install the language environment for Java and Scala

2) Download and extract the Gatling installation package from the Gatling official website

3) Put the written script in the user-files directory

4) Run the galting executable file in the bin directory (galting. bat in Windows operating system; gatling. sh in Linux operating system), and select the corresponding number of the script name in 2)

5) View the results report in the results directory

It is not difficult to find that Gatling has outstanding advantages, and the construction process and use process are relatively simple. However, through research and practice, this paper found that the usability of Gatling still needs to be improved.

3. Gatling Improvement Analysis

Performance testing is conducted for WEB applications. At present, most of the WEB performance testing tools form test scripts by writing or recording scripts, and then run scripts at the terminal to simulate the concurrent request scenario of actual users. However, in practice, it is found that the recorded scripts need to be manually edited before they can be put into use. For example, the recorded scripts contain hard coded values, and the readability of the recorded scripts is poor. At this point, writing scripts directly becomes the best choice.

The premise of performance testing through Gatling is that you need to write performance test scripts using Scala language. However, Scala language is not at the top of the list of programming languages, and the design of Scala language does not meet the requirements of ease of use: the difference of index starting values between Tuple and Array, the basis of determining whether a literal is a variable or a constant in pattern matching, the inconvenience of using defined operators in the standard library, and the poor readability due to the large degree of freedom. For most people, Scala is a new programming language, and its design style is significantly different from other programming languages. Even for those who have a programming foundation, it is still difficult to form an executable and passed Gatling test script.

In addition, by understanding the use process of the WEB performance testing tool Gatling, using Gatling for performance testing requires a lot of manual operations by users: placing scripts in the user-files directory, running the Gatling executable file, selecting the script number, etc., which is also inconvenient for users.

In view of the above problems, based on the research on the use of Gatling, this paper proposes an improved scheme of Gatling.

4. Gatling Improvement Plan

In order to improve the usability of Gatling, the author packaged the Gatling framework. The improvement plan mainly considers the following aspects: First, there are many steps to manually execute test scripts, which do not meet the requirements of automated testing; The second is that Scala language script writing is difficult for testers. It is necessary to reduce the amount of code writing or replace it with another easier programming language.

Through investigation and analysis, we found that Python language is widely accepted in the software testing industry, and the language is

flexible and easy to get started, so we chose Python language for Gatling packaging, and designed a portable and lightweight Python code program file named GatlingPro. GatlingPro files include config directory, data directory, result directory, AutoRun.py file, YamlTest.py file, GatlingScriptCreate.py file, RunScript.py file, etc. The specific improvement scheme is as follows:

1) Store performance test data in the data directory, including but not limited to. csv file,. txt file, Excel file, etc;

2) Configure the yaml template file in the config directory. The yaml template file is used to configure the interface request information and test scenario parameters (the involved test data is read from the data directory, and the data file name can be configured in the yaml file);

3) Execute the AutoRun.py file to trigger the automatic generation and execution of the performance test script. The automatic operation during the process is as follows:

a) Automatically execute YamlTest.py file to automatically request information extraction and template filling.

b) Automatically execute the GatlingScriptCreate.py file to automatically encapsulate the pressure test scenario and concurrent configuration, automatically generate the script, and automatically store it in the result directory.

c) Automatically execute RunScript.py file to realize automatic performance test.

The improved GatlingPro file structure is described as follows:

1) Config directory: the configuration folder, which contains the relevant configuration files of gatling - interface request configuration files, script generation configuration files;

2) Data directory: data folder, which stores the test data files to be imported;

3) Result directory: used to store the generated Scala script files;

4) GatlingScriptCreate.py file: used to read the configuration file and generate the interface pressure test script;

5) RunScript.py file: used to execute Scala scripts stored in the result directory.

The flow chart of Gatling before and after improvement is compared as follows (Figure 1. Gatling operation before improvement and Figure 2. Improved Gatling operation):

Figure 1. Gatling Operation Before Improvement

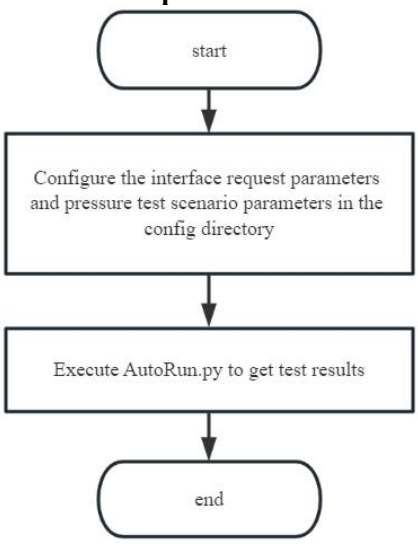

Figure 2. Improved Gatling Operation

5. Conclusion

Performance efficiency is an important

indicator of software product life cycle, especially for consumer-oriented software products. To measure whether the performance efficiency of software products meets the requirements, the selection of performance testing tools is crucial, which will directly affect the accuracy and effectiveness of the test results. The improvement of test tools is a long-term and complex work. When using tools, we should also pay attention to the technical principles, problems and shortcomings of the tools. We have encapsulated and improved the WEB performance testing tool Gatling to reduce the difficulty of using the tool. Further research and optimization work on Gatling is still ongoing.

References

[1] Chen Wu, Chen Jianan, Li Yanping, Industrial Internet Platforms: Connotation, Evolution, and Empowerment. Business and Management Journal, Issue 05, 2022, 189-208.

[2] Xu Liuyi, Ecological Development of Industrial Internet Platform under the Data-Driven Background. Decision & Information, Issue 01, 2024, 86-96.

[3] Wang Jingchao. Research on Application Software Performance Test based on JMeter. North China Electric Power University, 2023. (MA thesis) DOI:10.27140/d.cnki.ghbbu.2023.000820 .

[4] Qin Xiao. Design and implementation of local tax office system based on SSH. Xiamen University , 2013. (MA thesis)

[5] Li Zhining. Research on Application of Web-oriented Automated Testing. Huazhong University of Science and Technology, 2012. (MA thesis).

