ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

Research on Application of Layered Technology Development Software in Cloud Computing Environment

Yu Ye[a], Hao Liu[b*]

School of Information, Hunan University of Humanities, Science and Technology, Loudi, China
[a]yuree2008@126.com, [b]LHKD0407@126.COM
**corresponding author*

Abstract: *With the continuous updating of computer software technology, more and more attention has been paid to the development and development of computer software. According to the characteristics of the application of layered technology in computer software development and the good application of layered technology in computer software development, it can meet the growing requirements of computer software applications under the new situation. The rapid development of the national economy has promoted the continuous progress of computer technology in China. With the advent of the information age, more and more people are paying attention to the development technology of computer software. Among the technologies used in computer software development, layered technology is a widely used technology. Computer software development and application promote the business of computer software processing gradually from a single item to a multi-level structure. This paper studies the application of layered technology in computer software development. This article uses a computer program software of a general scale as the experimental object for data statistics. Through the data analysis, the number of components in the architecture restored by 100 versions of 6 experimental projects is analyzed. Through the following figure, we can see the number of components of 90% of the tested projects. Both are lower than 20, so it is considered that the recovered architecture can be easily understood.*

Key words: *Layered Technology, Computer Software, Software Development, Hierarchical Structure*

1. Introduction

With the development of science and technology, the development of computer software applications has been promoted. The development of computer software has gradually changed from the previous two-tier structure to a multi-level structure, and has become a key content of the work of related software developers. Especially in today's increasingly complex computer environment, the use of layered technology in the development of computer software can continuously improve the completeness and clarity of software systems, and improve the flexibility of related software use in computers, promote the development of computer software. Computer software is an important factor in promoting computer popularity. Computer software has changed people's lifestyles, such as the application of WeChat and email, changed the way people communicate, and shortened the distance between people. In addition, the development of computer software also improves people's work efficiency. When encountering problems at work, people can use computers to communicate and communicate anytime, anywhere, which is convenient for people to solve problems in the shortest practice. But in terms of current software development, there is still a large gap between the level of computer software development in China and developed countries. In order to reduce the gap, related software developers should actively innovate development methods. At present, layered technology is the most effective way to promote the efficiency of software development. With the rapid development of China's information society, more and more attention is paid to the research and development of computer software, and the application of computer software is becoming more and more widespread. In the early days, the development model of computer software was mainly a single-layer structure. However, with the increasing complexity of network structures, users have put forward higher performance requirements for software. Computer software development models such as single-layer structure mode or two-layer structure mode have been unable to meet the increasing performance requirements of computing software applications. Reforms are needed to improve computer software

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

development technology. Computers have been widely used in people's lives. To a certain extent, computers have changed people's daily lives and production methods. They have even greatly improved production efficiency to a certain extent and have created great value. However, there are many security risks in the actual application process of the computer. If there are security risks in the process of use, it will easily lead to the leakage of user information, or the personal computer will be maliciously attacked, which will seriously affects people's normal application.

In recent years, today's computer security technology has also received widespread attention in the society, and in order to provide users with a good computer network application environment, security technology should be fully used in the entire process of computer software development in order to maximize ensure that the computer can have good stability and security during normal operation, thereby promoting the development of society [1]. Due to the impact of comprehensive social needs, the development consciousness of many traditional industries has also changed greatly. Computer software technology is the leader in advanced science and technology in the new century. It has been widely used in many industries in China and has achieved the ideal effect [2]. As the direction of computer software development becomes more and more diversified and functionalized, this also makes people's expectations for computer software technology have greatly improved [3]. After a period of practical research, it was found that the structural model of computer software development is not limited to the model of a single layer structure, but uses layered technology as the core of computer software, which also makes computer software applications to a large extent. The scope has been extended [4]. Today's electronic science continues to develop, people's demand for computers continues to increase, computer software functions have become more and more diverse, and the number of existing functions has continued to increase [5-6]. By using multi-level layering technology, software operations can be made more secure and flexible [7]. Layered technology has good performance, and can make the development of software systems more abstract, and the complex parts of software systems will become more and more simple. In this case, it is necessary to fully analyze the use of layered technology in computer software development [8]. Hierarchical technology is to set the software to different conceptual layers, and then develop a corresponding design scheme for each conceptual layer [9]. Of course, as far as specific design is concerned, although computer software can be divided into several conceptual layers, the different conceptual layers are on an equal footing and have a close relationship with each other. The first method can reasonably optimize the computer software development technology, and the second method can help to enhance the actual performance of the software, and then provide users with more high-quality and convenient services [10-11]. Computer is currently a very advanced information technology, which is used in various fields and has greatly promoted the development and progress of society. Computer software technology is developing and progressing in an all-round way, and research and development and application are all concerns of people [12]. The layered technology is the key technology of the current computer software, and it is also a new technology developed and applied in recent years. It can effectively promote the improvement of the level of computer software and can meet people's various engineering needs. Hierarchical technology was applied to the computer software development stage, which provided strong technical support for it and achieved significant results [13].

Docherty believed that models of complex systems have been widely used in the physical and social sciences, and the hierarchical concept based on graph theory structure is a common feature. We describe an intuitive substructure logic that gives a hierarchical description. Logic is a bundled system that combines the usual intuitive connectives with non-commutative, non-relevant connections (for capturing layers) and their related meanings. Regarding the Kripke semantics of graphs, we give the robustness and integrity theorems for labeled tableaux systems. Then, we give equivalent relational semantics, and prove that they are equivalent to algebraic semantics through the representation theorem. We use this result in two ways. First, prove the decidability of logic by proving the limited embeddability of algebraic semantics. Second, we proved the Stone-type duality theorem for logic. By introducing the concepts of ILGL's high doctrine and indexed hierarchical framework, we can extend this result to a logical predicate version [14]. Huysmans considers cavity angles to be one of the most common machining features in complex structural parts of aircraft, and plunge milling the cavity angle has the advantages of high metal removal rate, small machining distortion, and high machining accuracy, and is particularly suitable for cavity bodies. Large depth of corner processing. However, current corner plunge milling methods have some problems, such as excessive side wall cutting, invalid tool positioning points, and uncontrollable remaining height. Therefore, a general layering algorithm is proposed and established for plunge milling tool path generation. First, a two-dimensional angular model is defined and represented. Next, introduce the terminology of machining layer circle, tool circle and cusp point circle, and build a layered model. Then, according to the intersection point of the layer's

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

fillet model, a tool arc layout program is constructed. At the same time, the relationship between the residual height and the layered model was determined. Finally, a layered tool path is generated recursively according to the radial cut width. Two sets of comparative experiments effectively prove that the method is superior to the traditional method in ensuring the remaining height and avoiding invalid tool positioning points. In addition, this layered approach can maintain an effective machining tool path ratio of greater than 80%. In addition, the application of CAD solid models of aircraft panels has proved the feasibility and effectiveness of the layered approach [15]. Machen believed that Mobile Edge Cloud (MEC) brings the advantages of the cloud closer to users by installing small cloud infrastructures at the edge of the network. This enables new real-time applications that require extremely low latency, such as instant target recognition and security assistance in intelligent transportation systems. A key issue brought by proximity is how to ensure that users always get good performance when moving between different locations. Migrating services between MECs is seen as a means to achieve this. He proposed a layered framework for migrating active service applications encapsulated in virtual machines (VMs) or containers. This layered approach can greatly reduce service downtime. The framework is easy to implement using off-the-shelf technologies. One of its main advantages is that it supports containers, which is a promising emerging technology that has clear advantages over VMs. The migration performance of various practical applications was evaluated [16].

This article mainly works on designing and implementing a software architecture recovery technology based on hierarchical structure information extraction. Reverse engineering (architectural recovery) is carried out from the four levels of project source code, compilation and construction process, project directory structure, and architect's description of the architecture. Get architecture dependency graphs; provide automated architecture recovery services for large projects. The main research contents of this subject include: 1) Diversified sources of architecture information extraction. According to the previous discussion, it can be seen that the extraction of architecture information cannot rely on only one method, but uses multiple methods, and the information is complementary. In this case, this article intends to extract the necessary architecture information from four aspects: source code, project directory, compilation / construction project files, and architecture models designed or developed by the architect, so that the architecture recovery can be as accurate and complete as possible. 2) Architecture information processing automation. According to the existing technical methods and tools, it is found that if the information is obtained from multiple means, there may be problems such as diverse data types, variable formats, and even inconsistent information. In this case, this article needs to simplify the input conditions. For example, you only need to enter the package (design documents are selected by the user), and you can automatically complete the architecture recovery. 3) Improve recovery efficiency and apply to large projects. The complex file dependency graph needs to be pre-processed to obtain a high-level module graph, and appropriate pre-analysis and componentization are performed on this basis. Finally, the tool can be applied to large-scale projects such as linux projects with millions of scales. 4) Improve the comprehensibility of the architecture diagram. Realize multi-level architecture recovery, generate different levels of abstraction, and support the mapping relationship between different levels, so that relevant personnel can not only have a macroscopic understanding of the software architecture, but also can deepen layer by layer to understand the relationship between architecture and implementation. This article will implement multi-level architecture recovery for software, control the granularity of components, and take a layer-by-layer approach to avoid the architecture diagram being too complicated.

2. Proposed Method

2.1. Characteristics of Layering Technology

(1) Expand ability

Generally speaking, the layered technology can effectively promote the performance of computer software to a certain extent, and at the same time, it can also effectively promote the upgrade and optimization of computer software. The sharing technology will further improve each functional area inside the computer software after comprehensively decomposing the complex software system, which to a certain extent effectively guarantees that the computer software system can run effectively after it is perfected[17].

(2) Independence

In the actual development of computer software, if there is a technical problem in one layer, then

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

dividing into technologies will have a certain impact on the upper and lower layers of this layer, but it will not cause harm to other layers. Computer software can be provided with independent and stable interfaces for each different level in each step of actual development, which to a certain extent effectively facilitates the computer software to develop a comprehensive software system in the actual development process.

(3) Stability

Layered technology can fully improve the efficiency of the actual software development at the grassroots level, so as to ensure that the computer can develop in the direction of a comprehensive software system, and more capable of progressing towards abstraction. Because layering technology has many advantages in stability, it can not only effectively reduce computer software development, but also be more targeted and purposeful in practice. This has a significant effect on improving the entire software operation [18].

2.2. Specific Methods of Layered Technology in Computer Software Development

(1) Application of double-layer technology

The use of double-layer technology in the process of computer software development will greatly improve the efficiency and quality of computer software development, so it can further shorten the software development time. The so-called double-layer technology, specifically, is a method that divides the computer software development process into two parts, the server and the client, and then combines the two to form a complete whole. Of course, because the two parts can be developed simultaneously, software development time will be greatly saved. As for the two specific matters and tasks that are responsible respectively, the main functions of the server include querying information data and obtaining information feedback by collecting user interface operations. The main responsibility of the client is to meet the user's practical needs through a visual interface for the operator. Although the application of double-layer technology in software development does help improve the efficiency of software development, based on this complex computer environment, its deficiencies are increasingly apparent. For example, in a short period of time, once the amount of data, it is easy to cause the server to crash due to the impact of a large amount of data. However, the maintenance of computer software based on a two-tier architecture often requires higher costs. Therefore, the current software development process is based on three-layer, four-layer or even more-layer technology. Of course, as far as the above technology is concerned, it is originally based on the double-layer technology, so the performance will inevitably surpass the double-layer technology, and the double-layer technology will inevitably be replaced by these new technologies [19].

(2) Application of three-layer technology

The three-layer technology is a new technology based on the two-layer technology, which is rationally optimized and improved. As far as the three-layer technology is concerned, the biggest difference between it and the two-layer technology is that it adds a brand-new server between the client and the computer layer of the two-layer technology. In this way, not only can the computer software performance has been further optimized, and the operating efficiency of software and calculations will be effectively improved. Specifically, it refers to computer software designed based on the three-layer technology. While collecting information, it will pass through the data layer, the user processing layer, and the interface layer in order. In this way, it will be able to fundamentally solve the problems that cannot be taken into account by the two-layer technology, and then better meet the needs of contemporary users for computer software. As for the design concept on which the three-layer technology is based, its business layer is mainly responsible for analyzing the user's software application requirements, and then providing users with functions such as information extraction, going out, and making requests. The interface layer is based on the analysis of user requirements collected by the business layer to pass the final result to the business processing layer, so as to meet the user's software usage requirements. As for the data layer, it is responsible for processing various applications from users and improving related information. As for the above three, because it is responsible for different matters, and different matters can be launched at the same time, it can greatly improve the efficiency of software operation[20].

(3) Application of four-layer technology

At the beginning of the popularization of computers, the design concept of relying on a two-tier architecture became able to meet the needs of most computer software. Now, with the continuous

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

development of computer technology, the three-tier technology is gradually unable to meet the needs of contemporary users. The practical requirements of software, so practitioners in the software industry have also developed four-layer technology with more diversified and personalized features based on the three-layer technology. As for the four-layer technology, compared with the three-layer technology, a new storage layer is added to the data, business and service layers. As for the specific role of the storage layer, it is mainly to make up for the lack of the technology in the three layers. In short, the specific application of the three-layer technology in software design will also promote the independence of computer software at different levels. In this way, the operation process of personnel will be greatly simplified. Not only that, because the general operating idea of the four-layer technology is to upload the data that has been processed initially in the data layer to the business processing layer of the software for in-depth processing, and when the data layer data goes to the business processing layer and is further optimized, it once again passed to the web layer, and based on the powerful technology of the web itself, it can support the synchronous exchange of a large amount of data, which will further promote the performance of computer software[21].

(4) Application of middle layer technology

According to the design and development of computer software, it will also involve other aspects of technology and content, and in this process, the application of middle-layer technology is undoubtedly the most frequent. Based on the effective use of middle-tier technology, it will greatly enhance the sharing of resources in the computer system, thereby effectively improving the efficiency of resource utilization. At the same time, the application of middle-layer technology in the design and development of computer software will also help solve problems such as heterogeneity and distribution integration, and effectively reduce the complexity of software development and design work, so as to ensure excellent computer software performance and practical enhance the overall advantages of computer software [22].

(5) Application of five-layer technology

The current layered technology in the design and development of computer software has reached the stage of five layers. Among them, the most important feature of the five-layer software development technology is the integration of the resource layer. Of course, the purpose of integrating into the resource layer is to provide customers with better services on the one hand, and to improve the stability of the computer system on the other. Take the most popular shopping site right now. Regarding the specific design of the shopping website related system, J2EE is currently the most commonly used software development platform. Based on the built-in system framework of J2EE, on the one hand, it can support the reality in the user interface and the browser; on the other hand, it can quickly process and quickly search the relevant information after the user issues relevant instructions, which will greatly improve the system's operating efficiency, and then practically meet the needs of users[23].

2.3 Application Guarantee Methods of Layered Technology in Computer Software Development

(1) Promote the research and innovation of software development technology

Scientific advanced development technology is an important guarantee to ensure the quality and efficiency of computer software development. Therefore, enterprises should encourage developers to conduct research on software development technology, especially on layered technology, so that developers can understand the significance of layered technology for software development in the process of research, and then promote layered technology in software development[24].

(2) Strengthen the training of talents

Talent is the key to the smooth application of layered technology. Only the software developer's own professional skills and comprehensive quality are too high, can he realize the importance of layered technology to the software development industry, thereby increasing the utilization of layered technology enhance the efficiency of software development. Therefore, enterprises should increase the cultivation of pastels, guide software developers to learn advanced layered technology theoretical knowledge and skills, and improve the layered technology application capabilities of existing software developers [25].

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

3. Experiments

3.1. Experimental Object

For a certain scale computer program software in this experiment, the experimental environment is 64-bit Windows 10 operating system, 8G memory, Java runtime environment is JDK1.8, IDE uses EclipseNeon4.6.2, and the database uses MySql5.6.10. For large-scale programs, the experimental environment is a 64-bit Windows Server 2008 R2 Enterprise operating system, where the CPU is IntelXeonE7-4809v21.90GHz, 32G memory. The rest of the configuration is unchanged.

3.2. Experimental Design

(1) Design of information extraction module

The main technical line of architecture information extraction: recovering the architecture from source code, but in the architecture recovery process, it is necessary to use the necessary information obtained from the compilation and construction of project files, project directories, and the architect's description of the architecture to recover from the code. To complement, improve, and verify. Therefore, the information extraction is mainly divided into FileDG construction module, dependency extraction module, directory analysis module, compilation and construction analysis module, and architecture document analysis module.

(2) Modular design

The modular algorithm selects the most appropriate adjustment scheme to properly correct the directory dependency graph, and clusters, splits, and extracts some highly coupled external modules to obtain a more accurate module dependency graph. The modular input in this method is the file dependency graph extracted through multiple ways.

(3) Module rule design

The process of componentization rules refers to the process of obtaining a further abstract module graph by aggregating strong dependency types and strong dependency structures based on the module dependency graph. The component preprocessing module integrates the functions of strong dependency analysis, domain node judgment, single dependency analysis, tight coupling analysis, closed-loop dependency analysis, open-loop dependency analysis, and transitive-loop dependency judgment, and stores the final ModuleDG in the form of a database. Prepare for componentized calculations later.

(4) Modularized module design

K-means clustering is used to divide into appropriate clusters, and then these modules are clustered using hierarchical clustering. Finally, pruning is used to obtain appropriate clustering results. To put it simply, clustering is to combine the two closest data points or categories, and iterate this process repeatedly, and finally generate a clustering tree.

(5) Architecture optimization module design

The architecture optimization module is optimized based on the restored architecture diagram (especially in some special cases that the previous modules could not handle). It is mainly divided into component naming modules, similar structure optimization modules, architecture design and directory optimization modules.

3.3. Experimental Measures

(1) Verify the validity of the architecture diagram

The expectation of the reverse engineering prototype tool is to accurately recover the software architecture, so verifying whether the results of the architecture recovery are consistent with the actual situation of the software is also the core purpose of this article. It should be pointed out that, considering the possible complexity of the system and the lack of architecture-related documents, the goal of this process is not to achieve a completely "correct" software system architecture with the implemented system. Such characteristics are also difficult to verify. Instead, our goal is to build an architecture that is "meaningful" to the system in question. This comparative experiment is divided into two parts: subjective analysis comparison and objective measurement comparison.

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

(2) Verify the understand ability of the architecture diagram

The architecture diagram restored by the prototyping tool supports layer-by-layer abstract display of files-modules-components. Through several layers of topological structure, it is convenient for users to understand the software system to be analyzed. Ideally, the number of components in the architecture diagram is appropriate and the distribution should be relatively uniform, but the cluster size distribution of the clusters often shows extremes. In particular, the clustering algorithm should avoid the following two situations : (1) most files are grouped into one or several large clusters (also known as black holes), and (2) most entity sets are isolated. Therefore, this experiment is divided into two parts: (1) verifying whether the number of components in the architecture diagram is excessive; (2) verifying that the components of the architecture diagram are evenly distributed.

(3) Verify the effect of architecture evolution

In software development activities, code implementation and architecture design documents often have deviations, and design documents are often not updated synchronously, which makes it difficult to obtain the original software architecture. Only by recovering the architectural elements through the architecture can we provide scientific and effective data support for observing software evolution. This experiment is mainly to verify whether the results of architecture restoration can reflect the effects of software evolution, that is, whether changes at the code level can be manifested at the architecture level. According to different analysis perspectives, changes at the code level can be divided into two types: code structure changes and code function changes. So the experimental part is also divided into two parts. Expected effect: If the code size increases or decreases, the number of architecture components also increases or decreases to a certain extent; if the program function increases or decreases a certain number, the number of architecture components also increases or decreases a certain amount; if the program function changes, the corresponding architecture component scale will also change.

4. Discussion

4.1. Analysis of the Understand ability of the Architecture Diagram

(1) Number of components

The experiment first needs to verify whether the number of components in the architecture diagram is controllable, because too many components will greatly increase the difficulty of understanding. Through a certain range of questionnaire discussions, we simply grade the number of components and the number of elements. Different levels are considered to have different degrees of difficulty in understanding, as shown in Table 1 below.

Table 1. The relationship between the number of components, the number of elements and the difficulty of understanding

Number of components	Number of elements	Incomprehensible
1-3	1-20	Easy to understand
4-10	21-40	Easy to understand
11-21	41-100	Understandable
22-40	101-600	More difficult to understand
40 or more	600 or more	Difficult to understand

From the above table, we can consider that the architecture diagram with less than 20 components is acceptable. Figure 1 counts the number of components in the restored architecture of 100 versions of 6 experimental projects. Through the figure below, we can see that 90% of the tested projects have less than 20 components, so we think that the restored architecture can be very easy to understand.

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

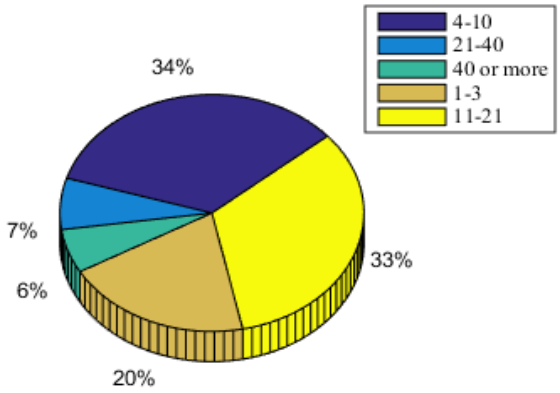

Figure 1: 6 Number of components in the experimental project architecture

(2) Component distribution

The non-extreme distribution (NED) measures of the test software were calculated experimentally, and the NED data of 18 versions of the three software were displayed. According to the table, it can be seen that the NED value fluctuates between 0.70-1.00. In the test process, the proportion of components with extreme distribution is relatively small. In order to show the distribution gap more clearly, the following uses a stacked column chart to show the comparison of the NED values of the 10 clustering methods. As shown in Figure 2, NEDMin represents the minimum value of NED, and NEDMax represents the maximum value of NED. Both are larger and better, and the overall is larger and better.

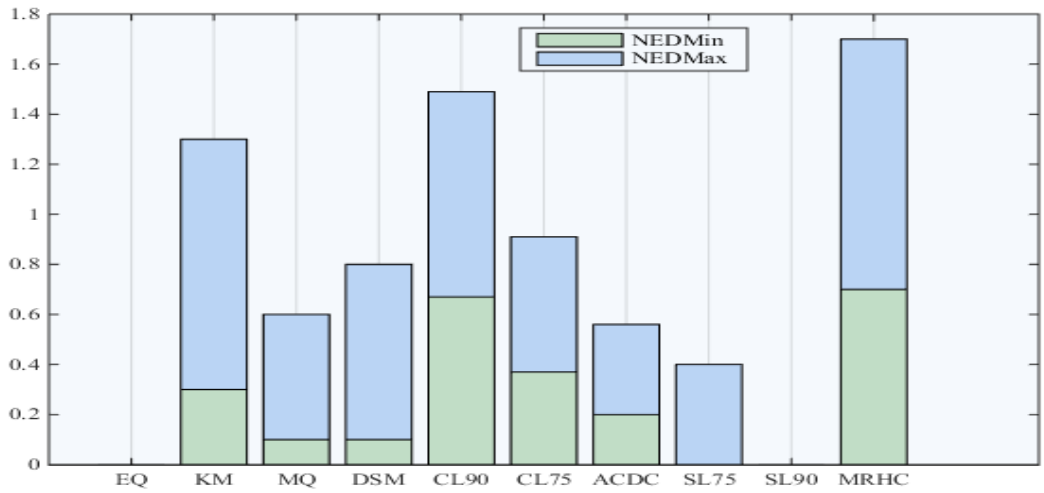

Figure 2: NED comparison of clustering method

4.2. Validity Analysis of Architecture Diagram

(1) Subjective analysis

The experiment requires the ground-truth architecture as the analysis material, and because either the document is lacking or the document cannot accurately reflect the software architecture, the authoritative architecture is usually difficult to find. After investigation, it was found that the team led by Professor Nenad Medvidovic of the University of Southern California has been doing research on architecture recovery in recent years, and has published many related articles in top conferences and journals in the field of software engineering (ICSE2013, ICSE2015, FSE2017, TSE2017, etc.). And the core that runs through its many achievements is the original architecture of the open source software

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

obtained by the team through years of cooperation with developers, including Bash, OODT, Arch Studio, Hadoop, ITK and Chromium. Bash-1.14 software was selected to compare and analyze whether the restored structure of the prototype tool matches the original structure. First, make a simple comparison. As shown in Figure 3 below, the original architecture of Bash_1.14 contains 25 components, and the recovery architecture contains 6 components (30 low-level modules that directly contain code files) .The components in the original architecture are the same as the recovery. The number of low-level modules in the architecture is equivalent. After manual analysis, the abstraction levels of the two are consistent.

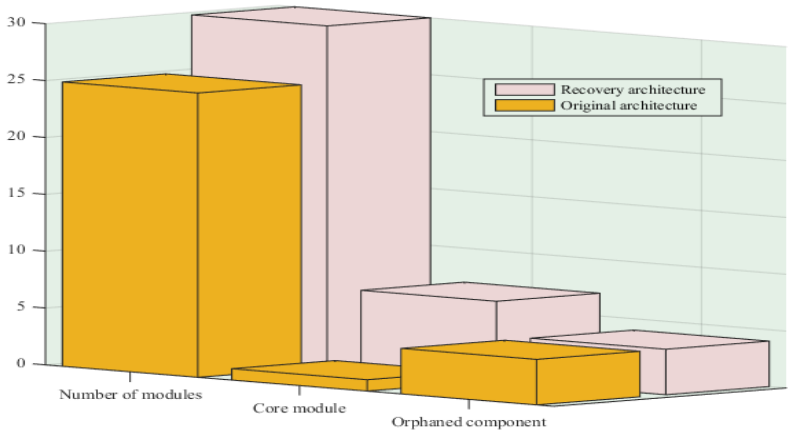

Figure 3: Comparison of original and restored architecture

(2) Objective measurement

The experiment selected three open source softwares: JEdit, OpenSSH, and JabRef for measurement. There are 73 versions, including 35 versions of JabRef (2.8-4.1), 18 versions of JEdit (4.3-5.5), and 20 versions of OpenSSH (3.1-5.3). For the sake of demonstration, the measurements of the 18 versions of each software selected in the measured results are shown in Table 2. As shown in the table, we can find that the overall value does not fluctuate much, and the MoJoSim value has a floating range of 0.55-0.75. Considering the subjectivity of the comparison architecture diagram, the overall authority is acceptable.

Table 2: MoJoSim measurement results

ID project	JabRef	JEdit	OpenSSH
V1	0.611	0.722	0.664
V2	0.627	0.733	0.665
V3	0.629	0.729	0.667
V4	0.659	0.728	0.655
V5	.0688	0.727	0.688
V6	0.710	0.739	0.666
V7	0.655	0.711	0.725
V8	0.644	0.726	0.722
V9	0.622	0.688	0.766
V10	0.633	0.666	0.669

4.3. Effect Analysis of Architecture Evolution

(1) Change of code structure

This tool is based on the program code. The entire process is implemented based on the code structure, and no code files are ignored during the recovery process. Therefore, the tool can theoretically be guaranteed to reflect the changes in the code structure during the software evolution process. In this experiment, five versions of the Apache-httped program are selected to verify whether the results of the architecture recovery can reflect the evolution of the architecture. It has an adjacent

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

historical version, and the evolution process must have relatively complete releasenotes. For the convenience of display, the first three versions are specifically shown here. Figure 4 shows the comparison between the evolution trend of code size and the evolution trend of component numbers. Through the following figure, we can find that： 1) the increase in code size is basically the same as the increase in the number of components. 2) The increase of components is much smaller than the increase of code. As analyzed in the previous section, the number of components should be controllable, and the rise of components should be more in line with the logarithmic function (a> 1), rather than infinite increase. Our tool also considers this requirement-the number of control components has grown, and experiments have shown that the number of controls has achieved its due effect.

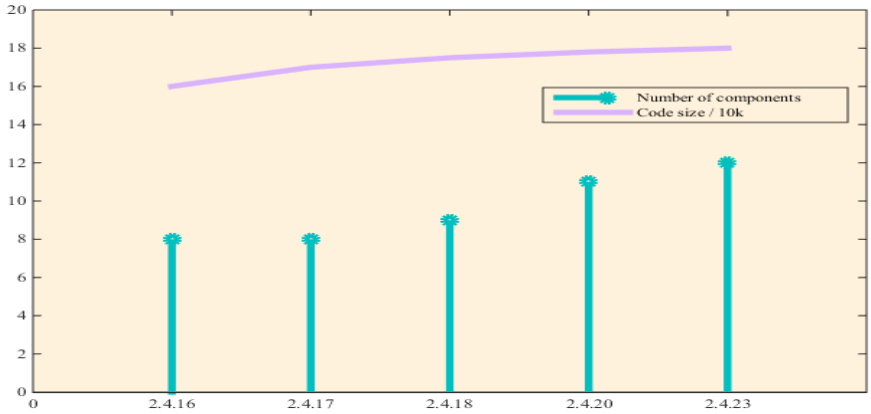

Figure 4: Evolution of code size and component count

(2) Change of code function

According to Apache-httped's releasenotes, the experimental statistics of important function change information and evolution information of related architectures. According to the information in Table 3, it can be seen that important function adjustments will have obvious evolution traces on the architecture diagram. It can be seen that it basically meets our expected effects, and changes in functions affect changes in the architecture. At the same time, it shows the module-module coupling relationship within the Modules in the Apache-httped architecture diagram. It can be clearly seen that during the evolution of version 2.4.17 to 2.4.18, due to the optimization of the core by the developer, the coupling relationship within the Modules of the component is large. The amplitude is reduced, and from 3 dependency center points to 1 dependency center point, cohesion is improved.

Table 3: Correspondence between feature changes and architectures of the Apache-httped evolution

Version 2.4.16 → 2.4.17		Version 2.4.17 → 2.4.18	
Release Notes	Architecture adjustment	Release Notes	Architecture adjustment
Began to support http2	New components modules \ http2	Optimize mod_http2	Component modules \ http2 increases in size
Optimize mod_proxy and mod_ssl	Component modules increase in size	Optimize mod_ssl and core	Reduced module size
Adjust mod_cache	Eliminate components modules \ cache	Adjust mod_cache	Restore components modules \ cache

4.4 Comparative Analysis of Architecture

Observing Figure 5 shows that the results obtained by MRHC on JabRef are higher than the MoJoSim values of the other four methods. It also validates our conclusions—the four methods that the MRHC method used is superior in terms of authority. In order to show the effectiveness gap more clearly, the following uses the stacked column chart to show the comparison of the MoJoSim range of the 10 clustering methods. MoJoMin indicates the minimum value of MoJoSim, and MoJoMax indicates the maximum value of MoJoSim. The larger both are, the better, so the larger the whole, the better. MRHC method is generally better than other methods, KM is second, SL75 and SL90 are the worst.

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

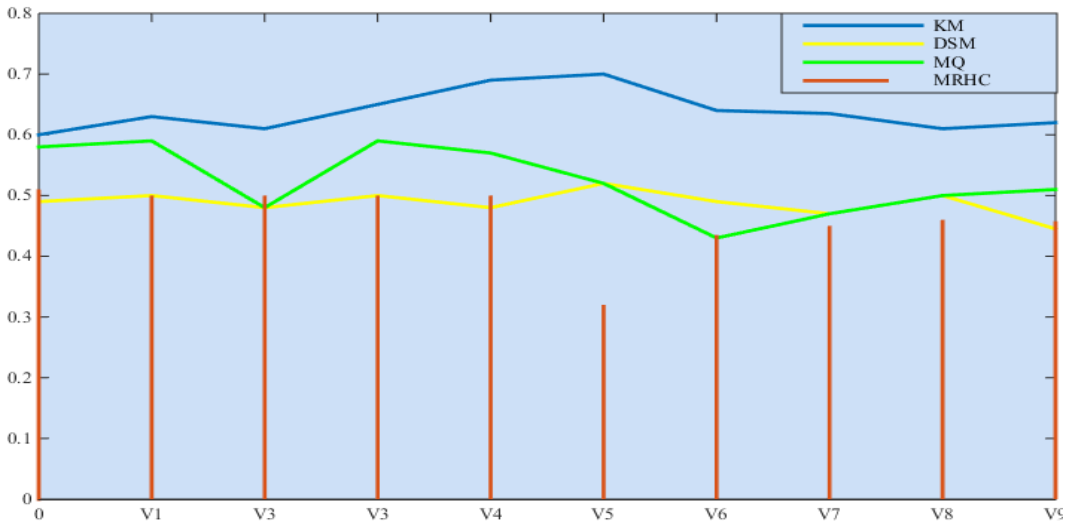

Figure 5: Mo JoSim sequences (JabRef) for four clustering methods

5. Conclusions

With the widespread use of Internet technology, people have higher and higher requirements for software. Computer is one of the information products. In order to meet people's needs, relevant staff should continuously improve the development process and technology of computer software so that the software developed can meet people's needs. The use of multi-layer technology is an important means to improve the efficiency and quality of software development. Software developers are extremely strong in the use of layered technology in the process of software development. The rapid development of the industry. With the advancement of modern information technology in China, it has been able to effectively promote the development of computer software development technology in China to a certain extent. Hierarchical technology is of great significance to the development of computer software in China. It can effectively improve the stability of the computer's actual operation to a certain extent, and it can also optimize the internal software system of the computer. Not only that, the layered technology applied in computer software development has also fully improved the development efficiency of computer software, and has fundamentally ensured that the computer software has a certain cycle of use.

With the advancement and development of information technology, China's computer software development technology has now developed rapidly. In the process of computer software development, the application of the hierarchical structure model has an important role. According to the characteristics of the application of the layered structure model, a good application can be guaranteed to meet the increasing performance requirements of computer software applications. In the computer environment, it is necessary to actively implement layered technology. With the increase of user needs, the difficulty of software development projects is also increasing. This requires technical personnel to conduct comprehensive analysis in combination with security and expansibility, and further shorten their operation and development cycle, improve the overall processing capacity of information, and on the basis of meeting the daily application needs, realize the comprehensive optimization of computer layering technology.

In short, layered technology is very important for the development of computer software. It can not only complete and detect the various layers and functions of computer software, but also provide new solutions for computer software development. Due to the increasing complexity of the computer environment, and the individual requirements of users for computer software development systems are more stringent, in order to meet user needs in software development, we have expanded the application of layered technology in computer software development. This article compares the advantages and disadvantages of different layer structures, selects multi-layer structures with high stability and high epitaxy as auxiliary means for computer software development, and focuses on the use of middleware technology between multi-layer structures to comprehensively improve computer software development technical level, fully meet the individual needs of different users.

We can see that at this stage, the layered technology has a lot of application space in China's

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

computer software development, and the long-term expansion of the application space has also been effectively reflected. Therefore, increasing the application of layered technology in computer software is of great significance to the overall development of computer software technology in China. In this era, people should be fully aware of the importance of the application and development of layered technology. The application of layered technology can effectively improve the quality and efficiency of computer software development, and this improvement is long-term, while shortening the computer software development cycle, it also makes the technology level of computer software development to a large extent to develop in a deeper and more intelligent direction. In the development and utilization of computer layered technology, it should be based on the actual situation of computer software development technology in China at this stage, and the serviceability and functionality of computer software development should be determined. When the layered technology is selected, In order to effectively ensure that the selected technology can effectively meet the actual needs of current computer software development.

References

[1] Ali Amiri. Application placement in computer clustering in software as a service (SaaS) networks[J]. Information Technology & Management, 2017, 18(2):161-173.

[2] Yang Li, Zhi-Ping Cai, Hong Xu. LLMP: Exploiting LLDP for Latency Measurement in Software-Defined Data Center Networks[J]. Journal of Computer Science & Technology, 2018, 33(2):277-285.

[3] Florian Dörfler, John W. Simpson-Porco, Francesco Bullo. Breaking the Hierarchy: Distributed Control and Economic Optimality in Microgrids[J]. IEEE Transactions on Control of Network Systems, 2017, 3(3):241-253.

[4] Suvendu Mandal, Markus Spanner-Denzer, Sebastian Leitmann. Complex dynamics induced by strong confinement – From tracer diffusion in strongly heterogeneous media to glassy relaxation of dense fluids in narrow slits[J]. European Physical Journal Special Topics, 2017, 226(14):3129-3156.

[5] Callum Bailey, Efrain Aguilera, David Espalin. Augmenting Computer-Aided Design Software with Multi-Functional Capabilities to Automate Multi-Process Additive Manufacturing[J]. IEEE Access, 2017, PP(99):1-1.

[6] Anderson Bergamini de Neira, Igor Steinmacher, Igor Scaliante Wiese. Characterizing the hyperspecialists in the context of crowdsourcing software development[J]. Journal of the Brazilian Computer Society, 2018, 24(1):17.

[7] Marco Kuhrmann, Jürgen Münch, Paolo Tell. Summary of the 1st International Workshop on Hybrid Development Approaches in Software Systems Development[J]. Acm Sigsoft Software Engineering Notes, 2018, 42(4):18-20.

[8] Robert Keyes. Development, Validation and Integration of the ATLAS Trigger System Software in Run 2[J]. Journal of Physics Conference Series, 2017, 898(3):032008.

[9] Jeff Gray, Bernhard Rumpe. The importance of flow in software development[J]. Software & Systems Modeling, 2017, 16(4):1-2.

[10] Guanfu Song, Ershun Zhong, Shaojun LI,Development of GISSoftware Technology in the Era of Big Data[J]. Journal of Geomatics, 2018, 43(1):1-2.

[11] Yu Ping, Zhang Qi, Huang Danian, Key Development Technology of Geoscience Software in Deep Earth Exploration Plan[J]. Journal of Jilin University, 2018, 48(2):501-506.

[12] Anderson Bergamini de Neira, Igor Steinmacher, Igor Scaliante Wiese. Characterizing the hyperspecialists in the context of crowdsourcing software development[J]. Journal of the Brazilian Computer Society, 2018, 24(1):17.

[13] Zhang H Z , Xie C , Nourian S . Are their designs iterative or fixated? Investigating design patterns from student digital footprints in computer-aided design software[J]. International Journal of Technology and Design Education, 2017, 28(2):1-23.

[14] Docherty S , Pym D . Intuitionistic Layered Graph Logic: Semantics and Proof Theory[J]. Logical Methods in Computer Science, 2017, 14(4):1-2.

[15]Huysmans M A , Eijckelhof B H W , Garza J L B , et al. Predicting Forearm Physical Exposures During Computer Work Using Self-Reports, Software-Recorded Computer Usage Patterns, and Anthropometric and Workstation Measurements[J]. Annals of Work Exposures & Health, 2017, 62(1):124.

[16]Machen A , Wang S , Leung K K , et al. Live Service Migration in Mobile Edge Clouds[J]. IEEE Wireless Communications, 2017, PP(99):2-9.

ISSN 2706-6827 Vol. 3, Issue 7: 100-112, DOI: 10.25236/IJFS.2021.030718

[17] Mandal S , Spanner-Denzer M , Leitmann S , et al. Complex dynamics induced by strong confinement – From tracer diffusion in strongly heterogeneous media to glassy relaxation of dense fluids in narrow slits[J]. The European Physical Journal Special Topics, 2017, 226(14):3129-3156.
[18] Bailey C , Aguilera E , Espalin D , et al. Augmenting Computer-Aided Design Software with Multi-Functional Capabilities to Automate Multi-Process Additive Manufacturing[J]. IEEE Access, 2017, PP(99):1-1.
[19] Prior D , Biscoe I , Rofe M , et al. Designing a system for Online Orchestra: Computer hardware and software[J]. Journal of Music Technology and Education, 2017, 10(2):185-196.
[20] Shi Y , Zheng G . The algorithm of layering calculation for corner plunge milling tool path[J]. International Journal of Advanced Manufacturing Technology, 2017, 91(5):1-17.
[21]Kern, Eva; Hilty, Lorenz; Guldner. Sustainable software products - towards assessment criteria for resource and energy efficiency[J]. Future Generation Computer Systems, 2018, 2018(86):199-210.
[22]Álvaro Rodríguez-Prieto, Ana María Camacho, David Merayo Fernández. An educational software to reinforce the comprehensive learning of materials selection[J]. Computer Applications in Engineering Education, 2018, 26(1):125-140.
[23]Munish Saini, Kuljit Kaur Chahal. Change profile analysis of open-source software systems to understand their evolutionary behavior[J]. Frontiers of Computer Science, 2018, 12(6):1105-1124.
[24]Kamel Barkaoui, Hanifa Boucheneb. Introduction to special issue on verification and evaluation of computer systems[J]. Innovations in Systems & Software Engineering, 2018, 14(2):1-2.
[25]Erik P. DeBenedictis. 3D Software: A New Research Imperative[J]. Computer, 2017, 50(10):74-77.