Research on the Evolutionary Development Method of Software Service Based on Trusted Agent

Tilei Gao, Ming Yang*, Xiaohui Jia and Tao Zhang
School of Information, Yunnan University of Finance and Economics, Kunming 650221, China
Email: 83300934@qq.com

Abstract. The factors that affect the trustworthiness of complex software system include not only the trustworthiness of subsystem services, but also the selection and construction process of subsystem services. According to the selection and construction process of sub services, this paper has proposed the concept of trusted agent. The trusted agent contains the key modules such as trustworthiness measurement and selection of software services, which can filter external software services and send the screened high-quality results to the system, so as to optimize the construction process of complex system. The method of software system construction based on trusted agent not only ensures the trustworthiness of the construction process, but also reduces the degree of manual participation, and improves the efficiency and automation of the software system construction process.

Keywords. Software service; trusted agent; evolutionary development method.

1. Introduction
The trustworthiness of complex software system is always one of the most important problems in software field. With the development of information and software technology, the principle of “high cohesion and low coupling” has been widely accepted and used. In the new computing modes of Internet of things, cloud computing and edge computing, most software service systems are developed by the combination, iteration and evolution development method which embodies the principle of “high cohesion and low coupling”. Complex software systems are composed of simple and relatively independent software modules or software services. Besides the dependability of the software module and the service itself, the dependability of the sub service is also affected by the composition of the sub service. Most of the existing researches in the field of software technology focus on the trustworthiness of the software module itself, while ignoring the research on the selection of trusted sub services and the combination of sub services. In essence, the combined and iterative development methods are used to develop the system by adding, deleting and replacing the software sub services. These developing methods can be regarded as the deformation of evolutionary development methods based on the maturity of software evolution technology. Aiming at the problem of software system trustworthiness development, this paper proposes a concept based on trusted agent. Through the construction of trusted agent and related evolutionary components, the selection of software services and evolutionary development method are realized. This method reduces the human participation and improves the efficiency and automation of software service evolution process. These are the support of the research of software system development framework in the new computing environment.
2. Related Work

2.1. Agent Theory and Research Status
Agent theory and technology was first used in the field of artificial intelligence. However, with the continuous expansion of the research content in the computer field, in the 1980s, agent theory, especially multi-agent theory, was used in the analysis and design of open systems, which was known as “another major breakthrough in software development” [1].

In the specific application process, agent system is often divided into single agent system and multi-agent system. Nowadays, agent theory has been applied to various emerging research fields. Canese et al. [2] summarized the common reinforcement learning algorithms using multi-agent theory. The research of Fernando [3] emphasizes the application possibility of multi-agent system in cloud service search, negotiation between cloud services and service composition. Cao et al. [4] proposed a model of using multi-agent system to manage cloud services in order to ensure the interaction between cloud services and the quality of cloud services. In the big data environment, the characteristics of agent have gradually become the bottleneck in the field of artificial intelligence, but in the field of software engineering, it can become an important method to solve problems.

2.2. Research Status of Software Evolution
The research of software evolution is still focused on the modification and maintenance of the system, while the research of applying evolution to the development process (evolutionary development method) is still rare.

As for the concept of software evolution, American scholar Evelyn defined software evolution as the change of software system with the advance of time [5]. Lehman et al. Holds that the system dynamics activities of continuous upgrading and maintenance of software system are carried out in corresponding life cycle [6]. Software evolution emphasizes the change of the essence of the software system, does not stick to specific forms, and focuses on describing the process of software from naivety to maturity [5]. With the continuous progress of software technology, the research of software evolution process itself is mainly carried out from the horizontal and vertical directions.

In the horizontal aspect, the research of software has gradually changed from traditional process-oriented system evolution to object-oriented software, component-based software, web services, SaaS services and so on. The research of evolution process also keeps pace with the development of software. There are object-oriented software evolution methods [7, 8], component-based software evolution methods [9], and Web Services evolution methods [10]. Zhong et al. [11] proposed a reverse engineering generation method of software evolution history, which constructs the system by evolving binary tree, and realizes the construction or optimization of new system by optimizing binary tree. The emergence of these research fields enriches the research content of software evolution field, and makes the research of software evolution process not only limited to the second half of the software life cycle, but extend to each period of the software lifetime.

In the vertical aspect, the theories and methods related to software evolution process are deepening, including requirements evolution [12], fault analysis and location [13], correlation analysis [14], and evolution process verification [15]. By introducing different research contents and methods, the theoretical system of software evolution process is enriched and improved. These research results provide a new perspective and method for the research of software evolution process in the new computing environment. The limitation of these studies is that most of them are to solve the local problems involved in evolution. In the process of specific evolution, the connection between different solutions is still weak.

3. Trusted Agent
The evolution of software services should not only focus on the implementation process of evolution, but also consider the source of software services. Trusted agent is an effective mechanism for software service selection and evolution management.
On the one hand, trusted agent is used to obtain the trustworthiness information of the service to be evolved, which provides the criteria and basis for the selection of software services. On the other hand, the external software service collection is connected to obtain the available software service information. It is the key component of software service trustworthiness selection.

According to the definition of agent, trusted agent is an agent that manages the registration, trustworthiness calculation and selection of software services from the perspective of trust. Trusted agent can complete the self-service management and screening of software services by obtaining the information about the trustworthiness of software services. The trusted agent constructed in this paper belongs to reactive agent. According to the evolution requirements of trustworthiness provided by the service blocks to be evolved, reactive agents make corresponding responses to help screen out available and applicable software services to realize evolution. The use case diagram of trusted agent is shown in figure 1 (In the picture, case 1: software service information management, case 2: software service trustworthiness calculating, case 3: software service storage management, case 4: software service selection management, case 5: external software service registration management.).

![Figure 1. The use case diagram of trusted agent.](image)

### 4. Software Service Evolution Process Based on Trusted Agent

In essence, trusted agent is a container that encapsulates all functional modules. The container mainly includes information centre (for software service information management), Computing centre (for computing the reliability of software services), Selection centre (selection and implementation of object software service), storage centre (for storage management of software services) Several functional modules, such as Registration centre (for external SaaS registration) and output module (for output of software service selection results). According to the use case diagram, the internal structure of trusted agent is shown in figure 2, and the interaction relationship between components of trusted agent is shown in figure 3.

**Registration centre:** Obtain the registration request of external software services and the relevant information needed for external software service trustworthiness calculation. Returns the result of successful registration.

**Information centre:** Collect the relevant information from the service blocks to be evolved in the system to calculate the trustworthiness, and mark it according to the information source. Collect information about external software services from the registry for trustworthiness calculation. Obtain the relevant parameter information manually input for setting selection criteria and calculation. In the
information centre, the information table of software service is obtained, which includes the value results of trusted sub attributes, influencing factors of sub attributes and so on.

![Figure 2. The internal structure of trusted agent.](image_url)

![Figure 3. Trusted agent sequence diagram.](image_url)

Computing centre: According to the software service information provided by the information centre, calculate the weight of trusted sub attributes, calculate the independent trustworthiness of each software service, and calculate the combined trustworthiness of service blocks. According to the mark of software service source, the calculation results of internal software service information are sent to the selection centre as the basis of selection. The output of the computing centre includes: system trustworthiness preference (sub attribute weight), the independent trustworthiness of services and the combined trustworthiness of services.

Storage centre: Store the independent trustworthiness from external software services and the possible local trustworthiness results after evolution. Returns the flag of successful registration to the registry. The storage centre only stores the results of software service trustworthiness, not the software service itself.

Selection centre: According to the selection parameters provided by the information centre and the selection basis trusted value obtained by the computing centre. Filter the software services of the storage centre.
Output module: Obtain the calculation results of the selection centre and send them to the software service evolution candidate area directly operated by the evolution execution process.

Evolution execution centre: receiving alternative software service information provided by trusted agent. It is associated with the collection of external services in order to obtain the software service body for the evolution process.

The most important parts of trusted agents are computing centre and selection centre. The computing centre completes the trustworthiness of computing software services to facilitate the screening of services. The Selection Centre completes the selection of external software services according to the results of the computing centre. The specific selection algorithm is shown in algorithm 1.

**Algorithm 1.** Software service selection algorithm

Inputs: collection of software services in the storage centre $S = \{s_1, s_2, \ldots, s_n\}$, Service to be evolved $s_e$

Outputs: alternative services $s_c$

```
1 Begin:
2 \quad S' = \emptyset, S'' = \emptyset
3 \quad For each $s_i \in S$ do
4 \quad \quad If $s_i.t_s \geq s_e.t_s$ do
5 \quad \quad \quad $S' \leftarrow S' \cup s_i$
6 \quad \quad End if
7 \quad End for
8 \quad For each $s_j \in S'$ do
9 \quad \quad If $s_j.t_l \geq s_e.t_l$ do
10 \quad \quad \quad $S'' \leftarrow S'' \cup s_j$
11 \quad \quad End if
12 \quad End for
13 \quad If $S'' = \emptyset$ do
14 \quad \quad Return fail
15 \quad Else
16 \quad \quad For each $s_k \in S''$ do
17 \quad \quad \quad Find $s_c$ who has the maximal $t_l$ in $S''$
18 \quad \quad End for
19 \quad \quad End if
20 \quad Return $s_c$
21 End
```

The operation of lines 16-18 of the algorithm can be based on other selection criteria provided by the information centre, for example, price and other factors. The selection process can also select multiple alternative services based on the threshold set.

In terms of time complexity, the algorithm contains three loops and no nested loops. The algorithm is used to filter the services in the service set $S$ by three different standards. Therefore, in general, the execution times of three loops decrease. The maximum number of executions of each loop is the number of service sets $n$. The maximum number of executions of internal statements in each loop is assumed to be $k_1$, $k_2$ and $k_3$ respectively ($k_1$, $k_2$ and $k_3$ are constants). Then, in the worst case, the total time consumed is $t = (k_1 + k_2 + k_3) \times n$. Therefore, the time complexity of the algorithm is $T(n) = O(n)$.

5. Conclusion

In view of the reliability of software system, this paper proposes a framework of software system evolution development method based on trusted agent. By constructing a trusted agent, it encapsulates the measurement and selection process of software services, and realizes the connection between the system to be evolved and the set of external services. By constructing the evolution execution centre,
the software services can be added, replaced and deleted in the evolution execution centre. Through trusted agent and evolutionary execution centre, the service block to be evolved and the external service set are connected, so that the operation of the original system, the measurement and selection of service trustworthiness, and the implementation of evolution can be carried out simultaneously. Thus, while ensuring the system trustworthiness, the participation of human is reduced, and the efficiency and automation of software service evolution can be improved. And these are the support of the research of software system development framework in the new computing environment.

Acknowledgments
This research is supported by Yunnan Fundamental Research Projects (grand NO. 202101AT070211).

References
[1] Liu D Y, Yang K and Chen J Z 2000 Research status and development trend of agent Journal of Software (03) 315-321.
[2] Canese L, Cardarilli G C, Nunzio L D and Fazzolari R 2021 Multi-agent reinforcement learning: A review of challenges and applications Applied Science 11 (11) 4948.
[3] Prieta F D L, Rodriguez-Gonzalez S, Chamoso P, Corchado J M and Bajo J 2019 Survey of agent-based cloud computing applications Future Generation Computer Systems 100 (11) 223-236.
[4] Cao B Q, Li B and Xia Q M 2009 A Service-oriented QoS-assured and multi-agent cloud computing architecture IEEE International Conference on Cloud Computing 644-649.
[5] Barry E J, Kemerer C F and Slaughter S A 2007 How software process automation affects software evolution: A longitudinal empirical analysis Journal of Software Maintenance and Evolution: Research and Practice 19 (1) 1-31.
[6] Belady L A and Lehman M M 1976 A model of large program development IBM Systems Journal 15 (3) 225-252.
[7] Ma J, Pan J P and Liu F 2018 Object oriented software system evolution model Journal of Jilin University (Engineering Edition) 48 (02) 545-550.
[8] Cheng L, Liu Y and Li T 2017 Component importance measurement method based on node betweenness in software evolution environment Computer Application and Software 34 (10) 29-34.
[9] Zhong L H and Xie B 2014 Research on modeling and obtaining method of component based software evolution information Computer Application Research 31 (02) 401-403.
[10] Zhang Y M, Ni K and Lu J W 2017 Research on automatic evolution method of web service composition based on global dependency network Acta Electronica Sinica 45 (02) 267-277.
[11] Zhong L H, Fu L J, Ye H T, Qi J and Xu J 2020 Research on reverse engineering generation method of software evolution history Journal of Computer Science 47 (S2) 559-566.
[12] Stephan S, Jan W and Helmut K 2018 A case study investigating the role of context and experiences in the evolution of enterprise software requirements Journal of Information Technology 33 (2) 151-170.
[13] Hojat M, Mirja P and Hadi G 2019 A root cause analysis method for preventing erratic behavior in software development: PEBA Reliability Engineering and System Safety 191 106565.
[14] Cao X and Zhao K J 2019 Fault location method of test case execution trajectory analysis Journal of Chinese Mini-Micro Computer Systems 40 (01) 149-157.
[15] Dai F, Li T and Xie Z W 2012 Algebraic semantics of a software evolution process model Journal of Software 23 (04) 846-863.