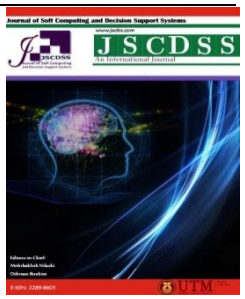
An International Journal
JSCDSS
http://www.jscdss.com
Vol.2 No.1 February 2015: 60-65
Article history:
Accepted 27 January 2015
Published online 28 January 2015

Journal of Soft Computing and Decision Support Systems



E-ISSN: 2289-8603

Self-Healing in Web Services Using Genetic Algorithm

Faezeh Yousefianarani [a], Eslam Nazemi [b,*]
[a] Department of Computer Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran
[b] Faculty of Science & Computer Engineering, Shahid Beheshti University, Tehran, Iran

*** Corresponding author email address**: nazemi@sbu.ac.ir

Abstract

In addition to monitoring, analysis, plan and execution phases in self-healing cycle, represents the knowledge base consumed and produced by all four previously mentioned tasks. In proposed approach, by using genetic algorithm, the required knowledge is prepared for healing operation. Healing operation takes place when the response time of the web service exceeds its threshold. In this case, using genetic algorithm, healing sequence is created to save response time and even to reach optimum state. Healing sequence causes the transition of service oriented systems from degraded state to healthy state as well as healing the error in web service and in this case lost time is recovered. To make healing sequence, healing approaches such as substitution, replication and skip is used which not only prevents process operation to be in no-response state but also results in optimization of response time. To provide healing sequence, execution of proposed plan benefits from consequent web services and is able to reduce response time and show the saved time.

Keywords: Self-healing, Web service, Response time, QoS

1. Introduction

A self-healing system should recover from the abnormal or unhealthy state and return to the normative healthy state, and function as it was prior to disruption (Harald and Schahram, 2010). The specification to which a system has been built is usually not fully known to those who maintain it. It is difficult to draw a discrete difference between healthy and unhealthy states of a system as the transition in between the two states is not abrupt. What generally obtains is a gradual transition from one state to another (Debanjan et al., 2006). Also, because web services are dynamic and unpredictable, one of their big challenges - quality of service- is in attention. Response time is one of the qualities of service parameters of which its increment is in inverse proportion to system performance. In degraded transition, response time increases continuously until it is not able to respond. There are several approaches that have been developed to achieve self-healing service-oriented systems. Some approaches implement self-healing after the detection of failures. When an error occurs, it stops currently running services and repairs or replaces the malfunctioned ones. A major issue with these reactive approaches is that they cause long disruptions of currently running service systems, which in most cases will incur high revenue cost or risk to lose a large number of customers (Hongbing et al., 2009).

In proposed approach, to diagnose the probable states of the system such as error and failure, an index is considered by which we can examine the quality of response time and can make healing sequence to continue their operation using proposed healing policies.

In Section 2, the related work is presented. In Section 3, we try to present proposed approach architecture in web services which named as Healing Sequence Creation Algorithm (HSCA). After that a brief description of implementation is shown and in Section 5 the results of execution of sample web services are given. In last section, we discuss about the results of Insurance Registration Example as a case study.

2. Related works

In (Yu et al., 2009), a self-healing approach is an integration of backing up in the selection and reselecting in the execution. In (Poonguzhali et al., 2011), a self healing approach which substitutes the alternative web service which provides the same service as that of failed service by considering the interrelationship between the component web services. In (Aziz et al., 2012), authors proposed a QoS-driven transactional service reselection model for reliable replacement. In (Ying et al., 2010), this paper proposed a T-QoS service selection model, a self-healing replacement model and designed related simulated environment. In (May and Judith, 2009), having a full

understanding of the interaction between different modules in a self-healing cycle provides the designer of a composition with the knowledge necessary to build more effective self-healing systems with minimum runtime overhead. Table 1 shows self-healing approaches in each work. Most of the works evolved so far for self-healing in web services for QoS based substitute the replica of original service. If the replicated one is not available then reselection of web services will go for execution without considering the interrelationship between the web services. Our approach makes healing sequence for web services. Table 1, shows the comparative study of existing and proposed works.

Table 1
Self-healing approach in various research works.

Self-healing approach	Self-healing Phases		
	Detection phase	**Diagnosis phase**	**Recovery phase**
Dai Y, Yang L, Zhang B, (Yu et al., 2009)	Monitor of the QoS-related context	QoS of certain service is predicted to be a large deviation	Integration of backing up in the selection and reselecting in the execution
S.Poonguzhali et al, (Poonguzhali et al., 2011)	QoS monitoring based on BPEL activity	Based average value of QoS parameters	Provides alternate service with the consideration of partner links
Aziz Nasridinov, Jeong-Yong Byun, Young-Ho Park, (Aziz et al., 2012)	Monitoring to extract information about the system health	Identify the QoS degradation	Reselect failed service
Ying Yin, Bin Zhang, Xizhe Zhang, (Ying et al., 2010)	Monitors the quality of component services	Violating current status of the execution from the execution plan	Replacement module
Proposed approach	Monitoring on the web services response time	Diagnosing errors in web services response time	Using Replication, Substitution, Skip

3. Architecture of the Proposed Approach

We named the proposed approach Healing Sequence Creation Algorithm (HSCA). HSCA consists of two phases, namely, monitoring phase and execution phase. The importance of monitoring phase is due to recognizing the response status of web services. Web services in execution environments and in heavy load undergo various states in their quality of response time. Similar requests at the same time are sent to a web service and some time leads to Heavy load. In this case, the web service may not be able to respond at the proper time. So, to prevent this state and probability of not responding, we will call HSCA which is shown in Fig. 1.

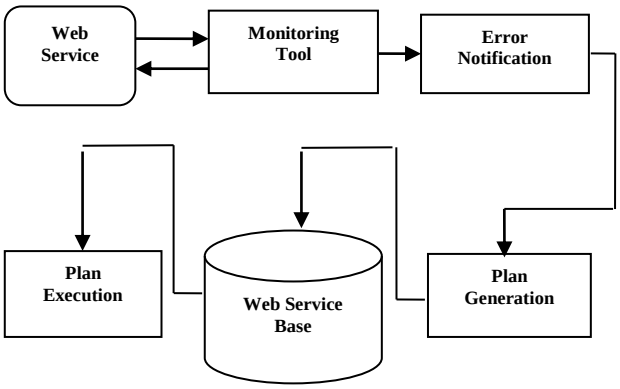


Fig. 1. Architecture of HSCA.

3.2 Monitoring Phase

In Fig. 1 monitoring tools frequently control the status of web services by the feature of response time. When response time of a web service exceeds the determined threshold time, we can use a proper healing approach for the web service which has lost normal time to its respond.

3.3 Executive Phase

During this phase, drop in the quality of response time of a web service, or in other words increment of response time in a web service which is too expected, can make error in the cycle of self-healing state. When response time of system exceeds the specified threshold time, system sets in error state. Fig. 2 represents the cycle of healthy, error and failure states in the system.

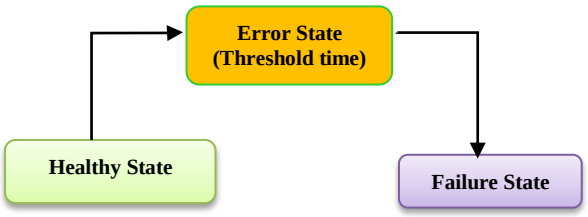


Fig. 2. Cycle of normal, error and failure states in web service.

As shown in Fig. 2, when response time of a web service exceeds the threshold time (error state), we have to find a solution for it; because some factors are slowing the system down and forcing it to failure state. In this state, web service is not able to respond anymore or it responds as too late as is not acceptable. After recognizing the error state in response time, HSCA tries to bring back the system from degraded state to healthy state. Table 2 shows the status of a web service from the viewpoint of response.

Table 2
Index of response quality.

QoS	System State
Responsible	Healthy
Exceeded from threshold time	Error

An International Journal JSCDSS E-ISSN: 2289-8603

No response	Failure

To overcome the issues mentioned in Section 2, we proposed a self healing approach which creates a healing sequence when each web service in error state. Depending on the type of web service operation and system administrators' opinion, healing sequence uses these healing policies:

i. Web service Substitution: with this operation web service is substituted with corresponding web service.
ii. Web service Replication: instead of using one sample of web service, several samples of parallel web services are activated and heavy load are divided between them.
iii. Skip from Web service: by this operation calling a web service is ignored and a web service which is located immediately next to it in workflow is called.

In the policy of web service substitution, substitute web service exists in the knowledgebase and makes over head. Using web service replication policy, several parallel web services do the processing operation of primary web service and compensate the lost time. The policy of skip from web service is applicable when halting a web service does not damage the sequence of execution of web services. Healing policies in HSCA will use search methods, by genetic algorithm. Genetic algorithm using healing policies suggests a healing sequence that tries to save response time of web services.

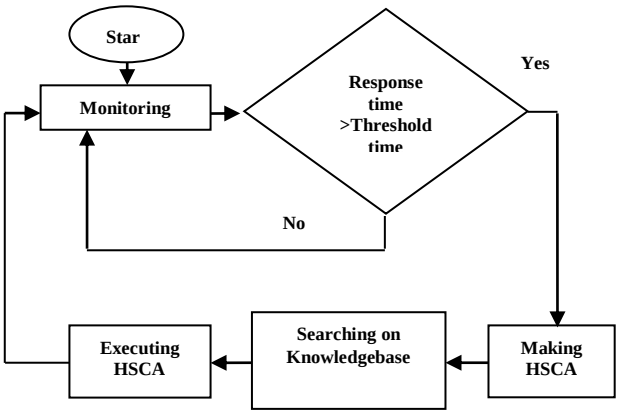
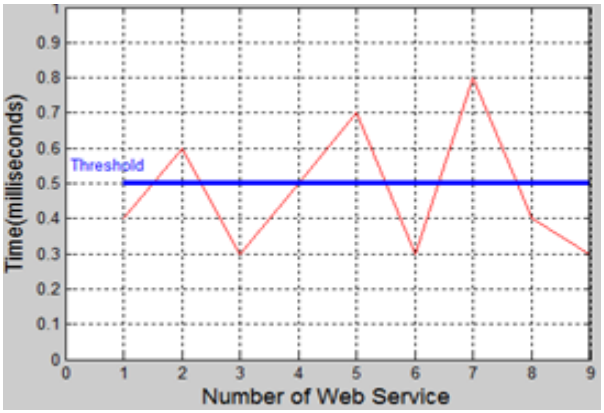


Fig. 3. Workflow HSCA in web services based on the quality of response time feature.

As shown if Fig. 3, when the response time of a web service exceeds its threshold time, the HSCA is called to make a healing sequence.

Fig. 4 shows an example of sequential execution of nine web services. The response time of second, fourth and seventh web services exceeds the threshold time. Therefore, for each mentioned web services the HSCA is called and healing sequence is made.



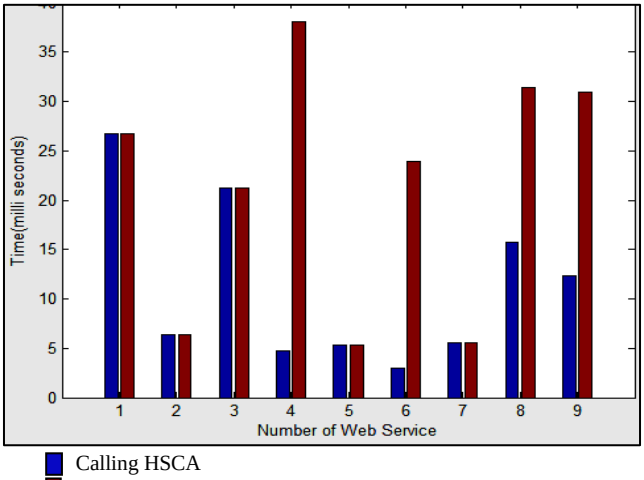
Error state
Response time for nine participating web services

Fig. 4. Execution time of HSCA.

4. Implementation

To evaluate HSCA, a tool designed in which the response time of the web service, threshold time, substitute web services and maximum of web services that can execute same as parallel stored in knowledge base. In this tool, when the response time of a web service exceeds the threshold time and web service sets in error state, the HSCA is called. In addition mentioned web service, HSCA suggests a healing sequence for the rest of web services. By using healing sequence, not only response time isn't lost but also shorter response time is obtained. If next web services will be in the error state, then HSCA is called again and will create healing sequence. HSCA cannot be left unanswered and process operation of sequential web services to be stopped.

Also, we can use the proposed algorithm same as case using. In this method, proposed algorithm is only used for the web service that it`s response time has exceeds the threshold time. Fig. 5 is an example of sequential execution of nine web services that shows a comparison between two calling states of proposed algorithm. HSCA compared with the case using obtains to shorter response time.



Calling HSCA
Calling proposed algorithm for case using

Fig. 5. Proposed algorithm calling states.

An International Journal JSCDSS E-ISSN: 2289-8603

Fig. 5 shows that the success rate of HSCA is always better than of case using.

5. Discussion

This part will verify the effectiveness of the HSCA.

5.1 Evaluation factors

There are many factors related to the self-healing of web services concept. We consider three factors for the purpose of comparison:

Low.QOS% = $\frac{N}{N_T} \times 100 = \frac{3}{9} \times 100 = 0/33$

N is the number of web services that have exceeded from their threshold time (the number of callings of genetic algorithm) and N_T equals the total number of web services.

High.QOS% = $\frac{N}{N_T} \times 100 = \frac{6}{9} \times 100 = 0/66$

N is the number of web services that have processed in their response time and N_T equals the total number of web services.

Availability % $= 1 - \frac{\text{overload time}}{\text{consumed total time}} = 1 - \frac{0/38032}{90/1532} = 0.993127$

The overload time is caused by calling HSCA. In other words, represents the time in which no web service is available and at that time no processing operation is done.

5.2 Comparison

As it is shown in Table 3, response time was reduced by about 0.34. Also, shows the comparison between the consumed times without self-healing approach, using self-healing with case using of proposed algorithm and using self-healing with HSCA.

Table 3
Evaluation of proposal approach.

Situation	Consumed time (ms)
Without self-healing approach	291.3978
Using self-healing with case using of proposed algorithm	189.588
Using self-healing with HSCA	101.0756

Table 4
Comparison of existing and proposed works.

Self-healing approaches	Evaluated features	Proposed solution	Saved time
Yu. Dai	Response time, cost	Integration of backing up in the selection and reselecting in the execution	Not specified
S.Poonguzhali	Response time	Selecting the substitute web service	no
Proposed solution	Response time	Mixing the healing policy	yes

6. Case Study

The insurance registration example will present in this section to demonstrate the usage of the proposed self-healing cycle in solving the problem identified. Fig. 6 depicts the internal operations of insurance registration.

6.1 Insurance Registration Example

The process is initiated with a user logging into the system (Registration Form). The system then performs the necessary authentication on the account and delivers a primary tracking code (Account & Create Code). In order to next tracking, a tracking code is needed. But response time of this web service exceeds the threshold time. HSCA calls and suggests replication policy. Tracking code creates and sets the type of insurance for example fire, vehicle, age, travel and etc (Check type of request). Based on the results returned, an identity code for type of insurance invokes (Create identity). Also updating the company regulations is needed but web service suddenly gets problems. Therefore, the HSCA calls again and suggests replication policy (Update rules). In parallel to these two actions, results of request will also be checked (Accept rules). At this stage, the user can choose to Accepts or Not accepts.

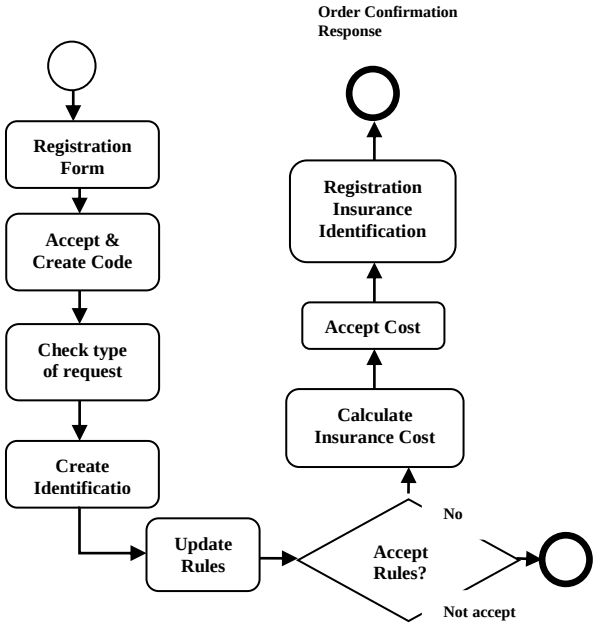


Fig.6. The insurance registration business process.

An International Journal JSCDSS E-ISSN: 2289-8603

If not accept selected, sends an email to the user. In case of accepting this inquiry, estimation of costs and payments is needed (Calculate insurance cost). Also web service is not able to respond in specified time. Therefore, HSCA is called and using replication policy.

In the next stage, user confirms estimated cost and pays (Accept cost). Identity code registers and user can see them for next references (Register Insurance Identification).

Table 5
Process of insurance request.

Web Service Name	Healing Policies	Output	Execution time(ms)
Registration Form	Nothing	Create Code	26.7564
Account & Create Code	Substitution	Check type of request	16.0547
Check type of request	Substitution	Create identity	16.4271
Create identity	Replication	Update rules	5.4333
Update rules	Replication	Accept rules	4.7577
Accept rules	Nothing	Calculate insurance cost	23.9283
Calculate insurance cost	Replication	Accept cost	4.9282
Accept cost	Replication	Register Insurance Identification	3.492
Register Insurance Identification	Replication	Finish	4.4214

6.2 Possible Violation Points

Because a Web service lives in a dynamic environment, any unexpected changes to a service could potentially leads to a fault. In Insurance Registration Example we can identify three Possible Violation Points, namely Account & Create Code, Update rules, Accept rules. These are explained in Table 5. Solving these violations will be demonstrated using the proposed self-healing composition cycle.

7. Conclusion

In this paper, response time values degradations are detected and repair action provided high availability. Also, in order to save the response time of web services, we proposed Healing Sequence Creation Algorithm (HSCA) using genetic algorithm, with replication, substitution and skip policies. Using this approach, we could save the response time of web services that their response time has exceeded the threshold time in runtime.

In the future work, we will add online planning to HSCA. In such a way, we will evaluate healing sequences by numerical values and use them for comparison and reuse.

References

Dai, Y., Yang, L., & Zhang, B. (2009). QoS-driven self-healing web service composition based on performance prediction. *Journal of Computer Science and Technology*, *24*(2), 250-261

Ghosh, D., Sharman, R., Raghav Rao, H., & Upadhyaya, S. (2006). Self-healing system survey and synthesis. *Elsevier*.

Chan, K., & Bishop, J. (2009). The design of a self-healing composition cycle for Web services. *IEEE*, 20-27.

Nasridinov, A., Byun, J., & Park, Y. (2012). A QoS-Aware Performance Prediction for Self-Healing Web Service Composition. *Second International Conference on Cloud and Green Computing*.

Poonguzhali, S., Sunitha, R., & Aghila, G. (2011). Self-Healing in Dynamic Web Service Composition. *International Journal on Computer Science and Engineering (IJCSE).*, *3*, 2054-2060.

Psaier, H., & Dustdar, S. (2010). A survey on self-healing systems: Approaches and systems. *Springer-Verlag*.

Wang, H., Wang, X., & Yu, Q. (2013). Optimal Self-Healing of Service-Oriented Systems with Incomplete Information. *IEEE International Congress on Big Data*.

Yin, Y., Zhang, B., & Zhang, X. (2010). QoS-Driven Transactional Web Service Reselection for Reliable Execution. *International Conference of Information Science and Management Engineering*.

Author Biographies

Faezeh Yousefian was born in Tehran, Iran. She received her BSc. degree in Islamic Azad University of Kashan in software engineering. She is MSc student in Islamic Azad University of Qazvin, in Computer Engineering.

Eslam Nazemi, PhD Assistant Professor, Faculty of Science & Computer engineering, Shahid behshti University. Eslam Nazemi was born in Sarab, Iran, in 1954. He got the BSc. degree in Applied Mathematics and Operational Research from School of Planning and Computer Application, Tehran, Iran in 1977, The MSc degree in Both System Engineering and Economics in 1987 and 1996, and PhD in Industrial engineering and Information technology in 2005, Iran. He was the faculty Member from 1978 in School of Planning and Computer application and then from 1986 to the present, he has been with the Electrical and Computer engineering Faculty and then Faculty of Science &

An International Journal JSCDSS E-ISSN: 2289-8603

Compute Eng. at Shahid Beheshti University (SBU), Tehran, Iran. He was deputy of graduate and education affairs and now is the manager of informatics development of education in SBU. He is an Assistant Professor of Computer Engineering Department. His main fields of research are Self-* Software Engineering, Large Scale Software Development, Web Mining, and Self- Adaptive Software quality. He has authored and co-authored more the 90 papers in Journals and Conferences and has 10 books on software engineering, Software Quality, game theory, mathematics and project management.

An International Journal
JSCDSS
E-ISSN: 2289-8603