Abstract

In the present world of ever-changing and dynamic technologies, web service is an entity that needs the most frequent changes in order to attract and retain customers, as well as to streamline the operations and to reduce the costs in doing business. Typically, the ability to change fast is enabled - even encouraged - by technology. These web services have to be updated in run-time that increases the complexity of the service maintainability process. The services provided most of the time do not meet the customers’ requirements and have to be changed periodically. The demands mainly focus on fine-tuning of the web services without affecting the core functionality and QoS parameters of the web services. The fine-tuned changes made by the developer can also lead to exponential problems as the solution of a particular service can lead to a need for another change. Thus, finding a solution for the change does not prove fruitful in long term but there is a need for ensuring successful service computation and better maintenance. The service computation is determined by the computability criteria which include computability, traceability, dependability, configurability and
exception handling. These computability criteria are the pillars of the web service, which
determine whether the web services do not spring up any side effects when any modification is
made as per the customers’ complaints. There is no standard model which carries out the
above given functions effectively and therefore the proposed business logic model (BLM)
cannels its functions to monitor and analyze the changes made towards the source code of
the web service computing environment by the developer. This impact analysis between the
existing and modified source code is done through the comparison of the cellular pattern which
is generated by the BLM. This cellular pattern gives us complete structure about the web
service and thus proves to be helpful in evaluating the computability criteria. We have carried
out an experiment to analyze a sample web service in the airline reservation domain and even
evaluate its computability criteria thereby providing the results that the computability criteria of
the modified source code are maintained. The model can be further extended in the future to
automate the generation of the cellular pattern which reduces the service developers overhead
and give a statistical historical background to the service change factor. The automation of the
cellular generation also gives confidence to the service developer that he is advancing in the
right path.

Reference

Sixth International IEEE Conference on Commercial-off-the-Shelf(COTS) Based Software
Systems(ICCBSS’07).


Aguilar, “RAWS: Reflective Engineering for Web Services”, Proceedings of the IEEE
International Conference on Web Services (ICWS’04).

Oriented Petri Net”, 2009 International Conference on Information Management and
Engineering.

Practice”, In Proc. of 19th Int. Conf. on Advanced Information Systems Engineering (CAiSE’07),

Proceedings of the 3rd international workshop on Traceability in emerging forms of software

Processes Modeling”, Eight International Symposium on Autonomous Decentralized Systems
(ISADS’07)

Maintenance of Web Services “, Proceedings of the 21st IEEE International Conference on
Software Maintenance (ICSM’05).

Index Terms
Computer Science Theory of Computation

Key words
Business Logic Model Source Control
Web service maintenance