Journal of Computer Sciences and Applications, 2015, Vol. 3, No. 3A, 21-26
Available online at http://pubs.sciepub.com/jcsa/3/3A/3
© Science and Education Publishing
DOI:10.12691/jcsa-3-3A-3

Service Oriented Testing for Web Services

Paul Buck*, Qi Shi

Research Centre for Critical infrastructure Computer Technology and Protection School of Computing and Mathematical Sciences
Liverpool John Moores University Liverpool, L3 3AF, UK
*Corresponding author: P.Buck@2010.ljmu.ac.uk

Received May 07, 2015; Revised July 08, 2015; Accepted July 16, 2015

Abstract Web Services are an emerging facet of Service Oriented Architecture and have promised to address the challenging issue of application integration. Web Services, as a concept, have been developed in order to build and create distributed software applications. These services have many advantages such as, the interoperability between heterogeneous applications, and the ability to design and launch these services regardless of platform, programming language or operating system. For enterprises and academic establishments this is advantageous in a number of ways. It has become apparent however that Web Services face many diverse challenges which are preventing the technology from truly being adopted. Many industries such as defence and critical infrastructure have become increasingly interested in the use of Web Services but in order for them to play an integral part in today's society, an array of issues must first be addressed. We believe that as an issue, trust can be considered as one of the main technological barriers currently facing Web Services, in particular, trust challenges such as reliability and security. By implementing Web Services into systems that rely on having high standards of reliability and security, trust needs to be guaranteed between the provider and the consumer. One way to help provide assurance for this is to provide adequate means of testing. At its core, testing is the process of executing a program with the intent of finding errors to ensure whether a system is functioning as intended. This can involve activities such as specifying test cases, generating test data, monitoring test execution, measuring test coverage, validating test results and tracking system errors. In recent years, research on testing Web Services and Web Service compositions has been gaining much attention and is growing at a rapid pace. Testing is vital in any environment to help ensure a degree of trust. While there are many issues facing Service Oriented Architecture and in particular, Web Service technology, the trust challenge is a particularly critical issue which needs to be addressed. This paper outlines current research for the effective means of testing Web Services, online and in the cloud computing environment. We aim to provide means for Web Services to be composed and tested in real time, over the cloud, utilising test case generation methods and Oracle decision making. By testing Web Services and ensuring their functionality, we can provide a degree of trust to the service consumer, that the Service they are requesting is available and will function as intended. We have detailed the importance of Service testing and the reason for why it should be considered as an immediate issue to address.

Keywords: *web services, testing, cloud, trust, security*

Cite This Article: Paul Buck, and Qi Shi, "Service Oriented Testing for Web Services." *Journal of Computer Sciences and Applications*, vol. 3, no. 3A (2015): 21-26. doi: 10.12691/jcsa-3-3A-3.

1. Introduction

Web Services allow for the building of distributed software applications. This has presented many advantages such as, interoperability and heterogeneous design. Web Services are still yet to be widely adopted as a ubiquitous form of technology. This is predominantly due to the trustworthiness challenge that Web Services are faced with. Until there is a complete degree of confidence in Web Services, it is hard to imagine that this technology will be fully adopted, as guarantees cannot be made as to whether a Web Service or its subsequent composition will perform as expected and conform to a user's requirements. Testing Web Services is one way to help ensure trustworthiness. By providing testing, user or service consumer confidence can be improved.

Testing Web Services brings about its own set of challenges, which for testing to be truly effective, must be addressed. Unlike traditional software testing, Web Services testing occurs in a completely differently manner, as with the additional number of Web Services, the possibility of changes to a system increases. With every change to a service, there is a need for testing to occur. This presents challenges, such as when to test for changes and which operations are affected. New approaches and techniques must be applied to enable efficient means of testing, as existing methods are failing to be sufficient enough to provide the coverage that is to be expected.

This paper provides an overview of Web Services and their strengths and challenges. It also details the importance of providing testing to Web Services and outlines associated strengths and weaknesses. We propose our research direction, and how we can help in addressing the Web Service testing challenge. The layout of this paper is as follows: Section 2 provides background on Web Services, testing and their strengths and weaknesses. Section 3 outlines existing approaches currently attempting to address and resolve the testing problem.

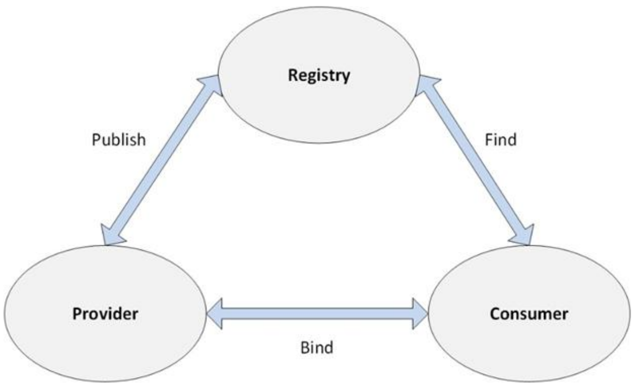

Figure 1. Typical Web Service architecture

Section 4 details our research direction within this area. Section 5 presents an overall summary.

2. Background

Web Service technology is currently an incredibly active area of research which has had a vast amount of interest from industry, academia and defence. Over the past few years, this area has grown exponentially. This can be attributed in part to the belief that Web Services can provide a solution to application integration and due to investment from many of the large technology companies [1,2]. In a report from Gartner, it was estimated that the Software as a Service (SaaS) and cloud based business application market was worth $13.4 billion by the end of 2011, The Gartner report also projected that by 2016 the market will be worth up to $32.2 billion [3].

There have been a myriad of proposed definitions for the term “Web Service” however, there is still yet to be a truly ubiquitous definition to define the concept. One accepted definition is presented by the World Wide Web Consortium (W3C) in which they refer to a Web Services as:

“A Web Service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web Service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialisation in conjunction with other Web-related standards.”– (W3.org. 2004). Figure 1 conveys a typical Web Service architecture, outlining the participants and operations.

Web Services are a paradigm in which software applications can be built using the Internet and a suite of open standards; they possess the ability to invoke other Web Services, in order to create more diverse systems. The open standards form an architectural style known as Service Oriented Architecture (SOA). These standards are clearly highlighted in Figure 2, showing the different application layers. Service Oriented Architecture aims to achieve loose coupling among interacting software components through the use of simple, well defined interfaces [4].

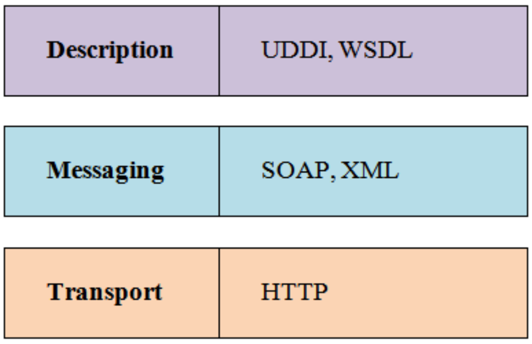

Figure 2. Web Service Standards

It is due to their ability to allow for the connectivity of applications that Web Services can be seen to have become such an attractive option, as regardless of the programming language the Web Service was created with or the owner of the Web Service itself, large systems can be formed. This is achieved via workflow languages such as the Business Process Execution Language (BPEL) and allows for Web Services to be composed into new composite services via the orchestration, coordination and composition of Web Services [5].

Web Service challenges and weaknesses are currently outweighing their benefits and thus currently, are preventing the mass adoption of Web Services into society as a ubiquitous form of technology. While plagued with challenges, the mass benefits of the strengths Web Services have are keeping the technology still very current within business, defence and academia. As the beliefs stand, once the challenges are addressed, Web Services will be the future of the Internet and software applications, providing unparalleled benefits to their benefactors [6].

The primary advantages that Web Services have include:

- Increased reusability and cost reduction to develop Web based applications
- Open standards
- Platform independence
- Dynamic discovery and composition
- Interoperability

These strengths provide the foundation for what the technology is aimed to achieve, and once fully integrated, Web Services are set to become vast market places for businesses to build and develop potentially vast distributed systems based on Service Requester needs and requirements. While practical in concept, there still remain many challenges which need to be addressed.

The challenges to Web Services are vast but to name but few can include:

- Unauthorised access
- Parameter manipulation
- Denial of Service (DoS)
- Unintended software interaction
- Application security

Surveying Web Services, trust has become an apparent issue which has hindered the progress of the technology [7]. In particular, we have identified the trust challenges of reliability and security as primary challenges. As Web Service technology grows and is adopted by sectors such as banking, which requires high standards of reliability and security, trust has to be guaranteed between the provider and consumer. Zhang stated that the current methods are simply not sufficient enough to ensure Web

Service trustworthiness [8], for the technology to grow, researchers must act and not wait [9]. Trust is a multi-faceted challenge, with many sub issues from security to quality of service. We believe that online testing can be used to deliver a solution for establishing trust and ensuring functionality.

2.1. Web Service Testing

Testing is a method in which a program is analysed for the purpose of finding errors and to examine whether it is functioning as specified [10]. This can be achieved in a number of ways such as, specifying test cases and measuring test coverage [11]. The importance of testing Web Services has become more apparent due to the rapid acceleration of the technology [12]. Testing enables a degree of trust, and with that degree of trust, consumers and providers can feel secure in the use of the technology.

Testing faces many of its own challenges, especially when used in conjunction with Service-Centric-Systems (SCS). There exist numerous testing approaches which over many years have been refined and developed to provide traditional systems with a high degree of coverage and have been developed to help ensure many of the traditional systems can offer trust to the consumer. Many of these methods apply to Service Centric Systems however, due to the dynamic and adaptive nature that Service Oriented Architecture presents, most of the existing testing techniques fail to be directly applicable [13].

It is believed that the distributed nature of Web Services, multiple protocols and their limited information provided by the specifications play as the contributing factor to why service testing is a challenging area.

Some of the key challenges which are hindering Web Service testing include [14]:

- Lack of observability
- Dynamicity and adaptiveness
- Lack of control
- Cost of testing

As outlined above, testing is a diverse process with many approaches which can aid in the ability to test a software system. In order for Testing to be current and relevant in the use with Web Services, one of the fundamental requirements must be that the process can be automated in order for the testing process to be faster, cheaper, reliable and ultimately more desirable. Here however presents one of the main challenges of software testing, the Oracle problem [15]. An Oracle is a mechanism that is used for determining the expected output associated with a test input, a mechanism for determining the correct behaviour of a system. While a fair amount of study of recent has been concerning with automating the generation of test input, one problem that has alluded much attention is that of automating the test oracle [15]. Without the aid of an oracle the tester must manually check systems behaviour for all test cases generated. This is an inefficient method and thus requires automation. The primary challenge associated with a test oracle is the expected output generation of how software should behave.

3. Existing Testing Methods

Web Service testing is an ever growing area within the Service Oriented Architecture field, with many diverse approaches being developed and applied to try and tackle current challenges within the area. By tackling these challenges, researchers are hoping to alleviate the stigma currently attached to Web Services and allow for their adaptation into society. A number of researchers have identified that there is room for growth in Web Service testing. As such, a number of solutions have been proposed, and this section shall examine some of these approaches.

Bozkurt and Harman propose a solution to test data generation and automation stating that one of the primary challenges of testing online systems is the generation of realistic test data. The authors state that the existing solutions fail to provide adequate means. By using existing Web services as sources of realistic test data, the authors have developed a framework that can generate test data in an automated way. While their approach is effective, the cost of generating test data is still reported at being too high and their solution is still dependent on needing more reliable services to demonstrate the overall effectiveness of the approach [16].

In an approach by Zhong et al, the authors propose a test case generation method that combines information from WSDL specifications and user knowledge. Their approach allows the user to provide test generation rules for data types as well as providing the means for the user to customise data types. While interesting, the approach does not allow for any automation and is all user specific thus it becomes inefficient [17].

Tsai et al propose an approach to address both service testing through test generation and behaviour checking from the Oracle. The approach proposed manages services on the cloud allowing users to compose services and test them. Users use a service interface to perform service composition and let the cloud environment map the interface to a suitable service implementation. The approach states that by using metadata or use scenarios, test scripts and cases can be generated and by using a voting mechanism, test oracles can be created with a confidence level. This is a very interesting approach; the primary issue with the method the authors propose is the additional complexity to the composition process [18].

One proposal to address online testing and trustworthiness is presented by Bertolino et al, in which the researchers suggest that one approach is to apply service federations and online testing. The researchers propose that the main purpose of online testing is to evaluate services in their real execution environment. Online testing primarily aims at verifying whether the behaviour a service manifests, while interacting with other services, abides by policies and complies with functional and non-functional specifications. The researchers state that a federation with online testing can help to assess whether a service that has just joined complies with the required specifications. It can also verify additional specifications that the service provides and its conformity to them.

The authors propose that by implementing an identity management component and testing components such as test robots, credentials and oracles, test cases can be created and implemented to provide different testing parameters and results and to provide a way to check aspects such as access control resilience. The primary challenges to this approach are the authors do not address the test selection process nor do they seem to address test cases automation. The authors also overlook the performance costs that such an approach could potentially have [6,19].

To address the choreography testing challenge, researchers have developed an approach in which they propose an online testing solution. The solution the researchers provide includes means for publishing choreographies, registering services to play roles within choreography, generating integrated test suites from the coordination model specified by the choreography, and launching the test suites online to evaluate how the coordinating services interact in their real execution environment. The framework also includes testing-based trust models to rank both choreographies and their linked services. The authors state to achieve this by implementing a suite of components which are implemented as Web Services which each performs activities for their framework such as their proposed test driver which generates test reports after executing a test suite. The authors outline an interesting framework, but seemingly fail to address the automation challenges of test case generation and rely on human interaction to provide the framework with a suitable set of tests [20].

Martin et al present an approach focussing on testing Web Services, developing a framework for robustness testing. The authors approach aims at automatically generating and invoking web services given a service providers WSDL. Their framework first generates necessary code to implement a client (service requestor) along with a wrapper class. The framework then leverages existing automated unit test generation tools to generate unit tests for the wrapper class and finally executes the generated unit tests, which in turn invoke the service under test. The results are collected and the responses from the web services are analysed in order to detect robustness problems [21].

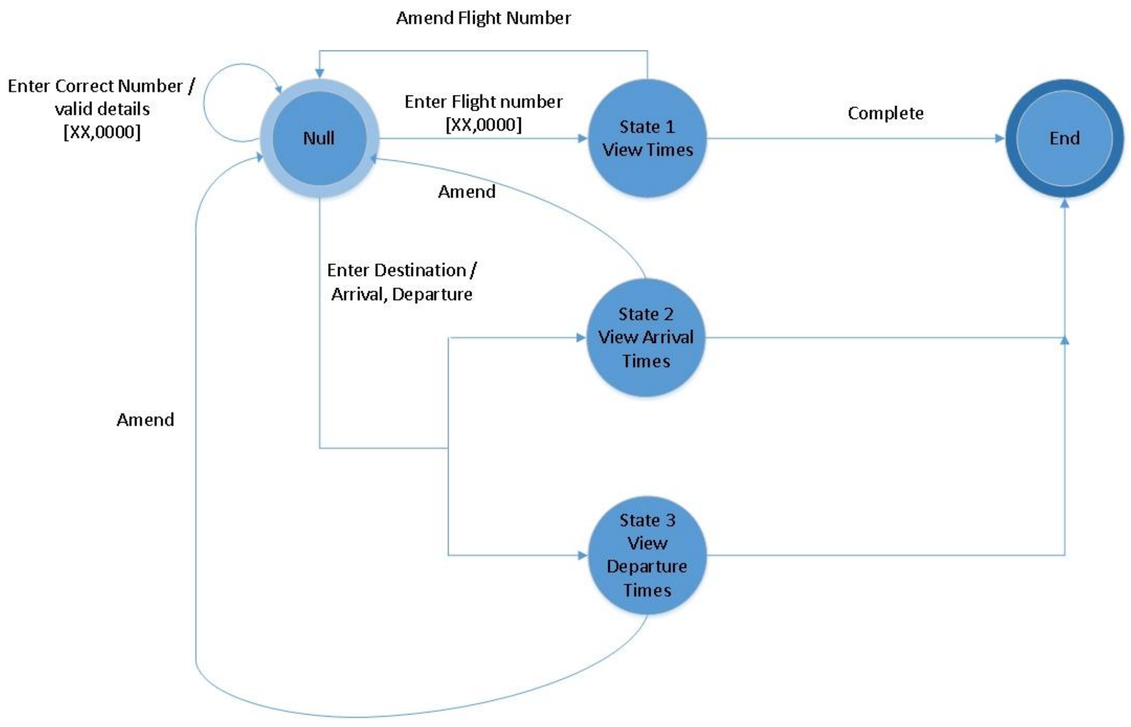
Amend Flight Number
Enter Correct Number / valid details [XX,0000]
Null
Enter Flight number [XX,0000]
State 1 View Times
Complete
End
Amend
Enter Destination / Arrival, Departure
State 2 View Arrival Times
Amend
State 3 View Departure Times

Figure 3. Example of state transition using contract based information

One approach by Atkinson et al is that of implementing a technique they refer to as test sheets to generate unit test cases and test oracles. The authors refer to test sheets as a language independent test definition language which allows for readable and understandable definitions of tests. The test sheets approach uses tables that define test cases, two types of test sheets are used in this approach; an input test sheet that contains specifications which are used for defining a set of test cases and a result test sheet that contains outputs from the service under test for the test cases in the test sheets. As further work for this approach the authors propose an extension to their method which extends the normal qualitative test results by the possibility to measure quantitative properties of the Service under test allowing for characteristics of web services such as timing deviations to be measured [22].

Chan et al propose a testing methodology to support both unit test and integration testing for service oriented computing through a metamorphic testing approach. By using metamorphic relations, Chan et al propose they can solve the test Oracle problem. Metamorphic relations refer to existing or expected relations over a set of distinct input and corresponding outputs. Chan et al, also propose encapsulating a service and imitating its functionality using metamorphic services. The authors accept that their approach has not been the subject to much evaluation and state that more detailed examinations are needed [23].

From the work examined, it has become relatively apparent that the existing methods in this area are struggling to provide a concise solution to the Web Service testing challenge. The current solutions provide theoretical and practical solutions but have all yet to truly provide the means to ensure trust to the consumer. Automation and verification has shown to be an area which much of the work fails to adequately address and leaves itself short in the literature. To overcome such challenges with automated testing we believe that the development of both automated test cases and test Oracles are in invaluable resource which must be established through research.

4. Research Direction

Our focus moving forward is analysing one approach which can be used for the accurate generation of test Oracles. This approach is referred to as Design by Contract. The approach generates Oracles by using pre-generated contracts. These contracts are able to carry information such as pre-conditions which specify conditions before an operation can execute, which include the system state and the arguments passed into the method. The post-conditions which specify conditions that must hold after an operation completes and invariants, these specify conditions anytime a user can invoke an object's method. This approach acts as an extension which the service provider must include.

By providing the contract based information we can establish whether the outcome of a test is accurate as we will be able to have a comparison of the expected responses of any given Web Service. Figure 3 is an example of a state transition diagram where contract information can be applied. The main advantages to this approach include helping to make the testing process autonomous as well as providing a degree of simplicity to the developer and tester. This approach can help ensure Oracle generation but generally can only provide support for single applications or "units" thus an extension to this approach would prove to be beneficial in order to provide support for the composition level of Web Services.

We shall also be exploring test selection and test case generation. Specifically we will be exploring the use of Genetic Algorithms. It has often been considered that testing should be considered as a search or optimisation problem one of which metaheuristic techniques can provide a solution for. One approach we have considered for our further work is through modelling BPEL (Business Process Execution Language) compositions through the use of a graph specification such as model or state. Once modelled, it has been considered that Genetic Algorithms can be applied to generate test cases based on coverage. The primary challenges associated with this approach are that the complete path or state coverage is difficult to obtain especially in larger constructs, state coverage can quickly become overbearing and lead to complex problems. We believe that this can potentially be addressed through different levels of state modelling and is going to be a focus for our future work.

Lastly, we shall also be exploring the use of federations to provide effective testing. By creating service federations, users would be able to use services from the domains of other providers, acting as though the services are from a single trusted provider. One of the primary benefits of using federations is that all services must comply with specified rules and policies which can help generate mutual trust. One challenge Web Services testing is presented with is the dynamic nature of Web Services. A deployed service which is updated without warning could create inconsistencies and failures for that specific configuration. Web Service federations could provide online validation which could ensure that any change is noticed and that relevant action could ensue. One of the primary issues with testing as a whole and with service federation testing is that of cost whether it is performance, human or other.

Our future work plan will be focusing specifically in the areas of hierarchical state modelling, design by contract, and optimisation which we feel will address the problem of accuracy, coverage and generation of autonomous test cases. The objectives for such include:

• Simulating the behaviour of a system. The behaviour can be analysed and represented in series of events, which could occur in one or more possible states.

• Design by Contract, using pre-generated contracts. Pre-conditions, Post-conditions which specify conditions before an operation can execute, which include the system state.

• Optimisation Algorithm through the use of genetic optimisation algorithms used to establish greatest path coverage and optimised solution for generating effective test cases.

5. Conclusion

In summary, Web Services are fast becoming a central part of today's Internet with services rapidly being developed [25], with many consumers ready to use the Services offered by providers and more critically to create Service Centric Systems based on a collection of different services composed together to effectively meet a consumers business needs. However, with the emergence of this technology, the question of how secure and trustworthy it is must be asked. Currently, the answer to that question can simply only be not enough. This has led to limitations in the true adoption of Web Services as business critical agencies such as banking or infrastructure critical agencies such as defence will only be able to see the benefits but be unable to use the technology for fear of insecurity which organisations cannot afford to have a lapse of.

While there are many challenges facing Service Oriented Architecture and in particular Web Services, the trust challenge is a particularly critical issue which needs to be addressed. One way of helping to address this is to provide effective means of testing Web Services. By testing Web Services and ensuring their functionality, we can provide a degree of trust to the service consumer that the Service they are requesting is actually available and will work as intended. We have detailed the importance of Service testing and the reason for why it should be considered as an immediate issue to address.

References

[1] Bichler, M. (2006). Service-Oriented Computing. Computer, 39(3), pp 99-101.

[2] El Ioini, N., Sillitti, A., & Succi, G. (2013). Using Rules for Web Service Client Side Testing. 2013 IEEE Ninth World Congress on Services, pp 158-165.

[3] Columbus, L. (2012). Cloud Computing and Enterprise Software Forecast Update, 2012. [online] Forbes. Available at: http://www.forbes.com/sites/louiscolumbus/2012/11/08/cloud-computing-and-enterprise-software-forecast-update-2012/ [Accessed 8 Nov. 2014].

[4] Inçki, k., Ari I., and Sozer, H. (2012). A Survey of Software Testing in the Cloud. Software Security and Reliability Companion (SERE-C), Proceedings of 2012 IEEE Sixth International Conference on Software Security and Reliability Companion, pp 18-23.

[5] Kovac, D. and Trcek, D. (2011). A Survey of Web services Orchestration and Choreography with Formal Models. [Online] Available at: http://www.softec.si/pdf/kovac-damjan.survey.pdf [Accessed 11 Nov 2014].

[6] Bertolino, A., Angelis, G. De, Kellomäki, S., & Polini, A. (2012). Enhancing Service Federation Trustworthiness through Online Testing. Computer, 45 (1), pp 66-72.

[7] Bozkurt, M., Harman, M. and Hassoun, Y. (2012). Testing and Verification in Service-Oriented Architecture: a Survey. Software. Testing. Verification. Reliability 2012, 23 (4), pp. 261-313.

[8] Zhang, J. (2005). Trustworthy Web Services: Action for Now. IEEE, IT Pro, January February 2005, (Vol. 7, No. 1), pp. 32-36.

[9] Tsai, W.T., Chen, Y. and Paul, Y. (2005). Specification-Based Verification and Validation of Web Services and Service-Oriented Operating Systems. In Proceedings of 10th IEEE International Workshop on Object-oriented Real-time Dependable Systems (WORDS'05), pp 139-147

[10] Schieferdecker, I., and Stepien, B. (2003) Automated Testing of XML/SOAP Based Web Services. Kommunikation in Verteilten Systemen (KiVS) Informatik aktuell 2003, pp 43-54.

[11] Rusli, H., Ibrahim, S. and Puteh, M. (2011). Testing Web Services Composition: A Mapping Study. CIBIMA, pp. 1-12.

[12] Paradkar, A., Sinha, A., Williams, C., Johnson, R., Outterson, S., Shriver, C., and Liang, C. (2007) Automated Functional Conformance Test Generation for Semantic Web Services. In ICWS '07: Proceedings of the 2007 IEEE International Conference on Web Services, pp. 110-117.

[13] Canfora, G., and Di Penta, M. (2008) Service-Oriented Architectures Testing: A survey, Lecture Notes in Computer Science Volume 5413, 2009, pp 78-105.

[14] Canfora, G. and Di Penta, M. (2006). Testing Services and Service-Centric Systems: Challenges and Opportunities. IT Professional. 8 (2), pp. 10-17.

[15] Shukla, R., Carrington, D., & Strooper, P. (2005). A Passive Test Oracle Using a Components API. In Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSEC 2005), pp. 561-567.

[16] Bozkurt, M., Harman, M. (2011) Automatically Generating Realistic Test Input From Web Services. Service Oriented System Engineering (SOSE), 2011 IEEE 6th International Symposium, pp. 13-24.

[17] Zhong. J. Li, J. Zhu, L.-J. Zhang, and N. Mitsumori, (2009) Towards a Practical and Effective Method for Web Services Test Case Generation. in Proceedings of the ICSE Workshop on Automation of Software Test (AST'09), pp. 106-114.

[18] Wei-Tek Tsai, Peide Zhong, J. Balasooriya, Yinong Chen, Xiaoying Bai, and J. Elston. (2011) An Approach for Service Composition and Testing for Cloud Computing. Autonomous Decentralized Systems (ISADS), 2011 10th International Symposium on, pp 631-636.

[19] Ghezzi, C. and Guinea, S. (2007). Run-time Monitoring in Service-Oriented Architectures. Test and Analysis of Web Services, Springer, pp 237-264.

[20] Ali, M., De Angelis, F., Fani, D., Bertolino, A., De Angelis, G. and Polini, A. (2014). An Extensible Framework for Online Testing of Choreographed Services. Computer, 47 (2), pp. 23-29.

[21] Martin, E., Basu, S., and Xie, T. (2007) Automated Testing and Response Analysis of Web Services. in ICWS '07: Proceedings of the 2007 IEEE International Conference on Web Services, pp. 647-654.

[22] Atkinson, C., Barth, F., Brenner D., and Schumacher, M. (2010) Testing Web-Services Using Test Sheets. in ICSEA 2010 fifth International Conference on Software Engineering Advances. 2010, pp. 429-434.

[23] Zhang, J. (2005). Trustworthy Web Services: Action for Now. IEEE, IT Pro, January February 2005, (Vol. 7, No. 1), pp. 32-36.

[24] W3.org, (2004). Web Services Architecture. [online] Available at: http://www.w3.org/TR/ws-arch/#whatis [Accessed 8 Nov. 2014].

[25] Liu, Z., Jia, Z., Xue, X. and An, J. (2015). Reliable Web service composition based on QoS dynamic prediction. Soft Computing, pp. 1409-1425.