Asian Journal of Research in Computer Science

4(3): 1-15, 2019; Article no.AJRCOS.52337
ISSN: 2581-8260

SOADIWA: A Service-oriented Architecture for Data Interoperability in Web Applications

Yusuf Lateef Oladimejia[1*]

[1]Federal University of Agriculture, Abeokuta, Ogun State, Nigeria.

Author's contribution

The sole author designed, analysed, interpreted and prepared the manuscript.

Article Information

DOI: 10.9734/AJRCOS/2019/v4i330113
Editor(s):
(1) Dr. Young Lee, Associate Professor, Department of Electrical Engineering and Computer Science, Frank H. Dotterweich College of Engineering, Texas A&M University-Kingsville, USA.
Reviewers:
(1) I.T.S. Piyatilake, University of Moratuwa, Sri Lanka.
(2) Pasupuleti Venkata Siva Kumar, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering &Technology, India.
(3) Mohammed Farik, University of Fiji, Fiji.
(4) C. K. Gomathy, Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya University (SCSVMV), India.
Complete Peer review History: http://www.sdiarticle4.com/review-history/52337

Original Research Article

Received 08 September 2019
Accepted 12 November 2019
Published 22 November 2019

ABSTRACT

The realisation of Service-Oriented Architecture (SOA) to communicate data between systems running on different platforms lack an organised framework to capture the essential elements required for successful interoperability between web applications and their services. In this work, a SOA for Data Interoperability in Web Applications (SOADIWA) was designed. The architecture of SOADIWA was based on five layers, namely Web Application Layer (WAL), Quality of Service Assurance Certifier Layer (QoSACL), Web Service Layer (WSL), Visualization Input Layer (VIL) and Visualization Output Layer (VOL). In WAL, the Service Requester (SR) initiates a request for data from the Service Provider (SP) through the QoSACL to provide appropriate website via WSL for rendering of services which must be accepted, processed and returned for a particular need in VIL. The requested data is filtered in VIL for data exploration and analysis in VOL using context-sensitive visualization techniques. The purpose of QoSACL is to check and verify the claims made by the SP about its quality of service. This enabled the SR to choose the service that satisfied its needs. The implementation comprised of Java Script, Microsoft Visual Studio 2017 and NuGet packages; while the experiment was simulated on LoadUI pro application. Standard metrics such as Optimal Performance (OP) and Phased Effort Distribution (PED) were developed to test SOADIWA. These results conformed to basic web service interoperability. The work led to the integration of a host of techniques towards the creation of a novel tool that is useful in web domain using SOA approach.

**Corresponding author: E-mail: truevisionconsulting@hotmail.com;*

Keywords: Service-oriented architecture; visualization; service cost estimation; web service; web application.

1. INTRODUCTION

The increasing success of Information Technology Business on the Web and the advancement of software systems have led to the evolution of Service-Oriented Architecture (SOA) paradigm as an architectural style for creating large distributed applications [1]. SOA finds its origin in Object-Oriented and Component-Based software development, and aims at enabling developers to build networks of interoperable and collaborative applications through web services. Web services are self-contained, loosely coupled reusable components that are built with little or no knowledge about clients and other services involved in their operating environment [2]. Web services could be anything from, weather reports, credit checks, stock quotations, airline travel reservation processes, and travel advisories to a complex process such as Customer Relationship Management (CRM) or Enterprise Resource Planning (ERP). Each of these services may be self-contained (fine-grained) business services or can easily be integrated with other services from different organizations to create a complete new coarse-grained business process [3]. However, the current SOA effort lacks organized architectural framework with ability to collaborate with visualization platforms [4,5]. These challenges arose because there is no particular software suite that fulfils all requirements for an entire organization or case study; an end user must cope manually with a collection of tools and its exporting/importing capabilities to obtain the output needed for a particular visualization assignment [6]. The concept of visualization is not new, what is new is the ability of visualization environment to collaborate with web application data and services [7]. This paper follows this line of thought through the development of a collaborative five-layered architectural framework based upon web service and visualization interface that are compliant with SOA for Data Interoperability in Web Applications (SOADIWA). The layers are; Web Application Layer (WAL), Quality of Service Assurance Certifier Layer (QoSACL), Web Service Layer (WSL), Visualization Input Layer (VIL) and Visualization Output Layer (VOL). In WAL, the Service Requester (SR) initiates a request for data from the Service Provider (SP) through the QoSACL to provide appropriate website via WSL for rendering of services which must be accepted, processed and returned for a particular need in VIL. The requested data is filtered in VIL for data exploration and analysis in VOL using context-sensitive visualization techniques. Quality of Service (QoS) has been an important concern in selection, consumption and composition of services especially when many service providers offer similar services with common functionalities but different QoS and cost [8]. It is therefore, necessary to provide negotiation mechanisms between clients and service providers to reach a mutually-agreed QoS goals. To address this problem, a new service discovery model where QoS is taken as constraints when searching for web service was proposed to give confidence to SR about the quality of the service they invoked. Furthermore, cost estimation for SOA is an issue that has not been well addressed in the existing literatures [9]; attempt was made to address this subject by considering several cost factors relating to SOA. These cost factors were distributed among different SOA project phases; weights are assigned to each factor to arrive at the cost effort of wrapping, reengineering and replacement. This approach represents one possible way to estimate the cost of SOA project. A proof-of-concept prototype and its testing was presented to illustrate a SOADIWA web page that collect and process raw data from third-party SP and eventually rendered visualized images for simulation and testing. The objectives of this paper are to; design SOA-based web applications that collaborate with visualization environment; develop a cost estimation process for the system and evaluate the performance of the system.

2. LITERATURE REVIEW

This paper was motivated by the works of [5,10,11,12]. For instance, [5] and [10] proposed a four-layered SOA to orchestrate heterogeneous tools, their works failed to consider QoS which is core for the proper management of SOA projects; we therefore build upon the works by providing a novel five-layered architecture with consideration for QoS and interaction between the layers. [11] also proposed a three-layered architecture for the development of visualization as a service by using visualization as the first and last steps of a process, the work motivated us to introduce visualization of data within the process as a means of sieving raw data for further use. The

framework to cost SOA using divide and conquer method introduced by [12] failed to consider various cost factors that enable the proper cost of SOA systems. The work was built upon by extracting various cost factors from different literature. These cost factors were distributed among different phases to arrive at a proper cost.

2.1 A Framework for Implementing Visualisation-as-a-service Approach

SOA is a collection of services with well-defined interfaces and shared communication model [13]. The underlying idea of SOA is that it would be cheaper and faster to build or modify applications by composing them out of limited-purpose components that can communicate with each other [14]. The advent of the Internet and World Wide Web (WWW) simplify the use of SOA in enterprise system [15]. For example, WWW facilitated an initial work on web software visualization applications that allows end-user to provide, create, view, save and share visualization data as presented in [16]. Also, study in [17] Introduced high-level guidelines for visual presentation of Model-Based System Engineering (MBSE) efforts with the insights drawn from best practices of information visualization (infovis) as applied to aerospace-based applications. The critical factors in the successful implementation of service-oriented architecture was also presented in [18] while a health care that supports personal health information management system was presented in [19] using SOA and web service technologies that provide remote medical care services that gives advantage of high reusability, flexibility and extensibility. However, the design was not yet implemented therefore not yet a full working solution. The study in [20] considered the Enterprise Service Architecture that represents the piece of software that lives between the business applications and enables communication among them, the failure to address security issue despite the fact that the ESB is a key enabler of a security-as-a-service model that enables a service-oriented approach to security infrastructure is a weakness to the study. The survey in [21] presents literature review solutions to SOA attacks on SOAP based web services which addressed various security measures available to SOA security standards for SOAP web services, the survey did not fully address preventing WSDL threats which is core to SOA. However, [22] provided a basic solution to control the messaging criteria between services by defining the process of message that fulfills the guaranteed delivery of message to its other end while communicating or asking for a service. Nevertheless, no solution for the better handling of guaranteed message delivery between the services in loosely coupled SOA was proffered. The systematic study reported in [1] presents an efficient automation model for the identification and evaluation of reusable software components to measure the reusability levels of procedure-oriented Java based (Object Oriented Software System), the model used a metric framework for the functional analysis of the object-oriented software components that target essential attributes of reusability analysis taking maintainability index to account for partial reuse. The development of Interoperable Intelligent Environmental Decision Support Systems (IEDSS) was presented in [23], the framework was based upon the cognitive-oriented approach for the development of IEDSS proposed in [24].

2.2 The Integration of QoS-based Service Selection Strategies within the Framework

Many authors have proposed quality of service as an integral part of SOA, for instance [25] suggested architecture for service-oriented pattern based on enterprise business solution which can play a role in the conceptual quality of service framework. The study in [26] also demonstrate how complex functionality in SOA can be composed and accomplished through many of the existing web services to form a coherent service flow for web service selection using BPEL. However, this work was demonstrated with only two services while our paper offers more services according to user request by using service discovery approach presented in [2]. A set of similarity measures aiming at addressing the problem of finding and selecting web service compositions that are similar with an initial composed service specification was proposed in [27] using a set of similarity measures that defined composition modeled as finite state machine. A global trust service composition approach based on random QoS and trust evaluation, considering the multi-criteria assessment of service quality was proposed in [28]. Also, a personalized context-aware recommendation approach for predicting the QoS of web services and designs of prediction framework was proposed in [29] by collecting QoS information from geographically distributed service consumers based on the QoS and context information they invoked. An enhanced composition model based on web

services was also proposed by [30]. A mathematical modelling of predominant QoS factors, availability and reliability of atomic services using Markov chain model and Weibull analysis respectively was also presented in [31] and based on test result, the combined effect of availability and reliability enhanced better estimation of QoS. A location-aware web service recommender system which helps users to select services with optimal QoS performance was proposed by [32], the web service QoS data set promoted future research and make experimental study reproducible. Furthermore, [33] recommended a web services monitoring framework supported by a software solution, while [34] suggested a web service recommender that uses collaborative filtering algorithm to deliver the most relevant data for a given queries by including investigative correlation between different QoS properties and detecting malicious users with inaccurate QoS information; their techniques were effective and efficient when compared to other approaches through experimental and simulation analysis. The systematic study in [35] also proposed an approach for updating reputation of the web service based on the trust factors of the consumers and two pre-defined thresholds, reputation threshold and agreement threshold. [36] proposed a novel collaborative filtering-based web service recommender system to help users select services with optimal QoS performance. In this work, we used a framework based upon web services and visualization tool-wrapping technique to advance the effectiveness of QoS in SOA.

3. METHODOLOGY

An approach to business process using web services to visualize information in a web application environment was presented. The architecture was factored into five layers that facilitate separation of concerns and assist the process of creating SOA with visualization capability. The layers operating in five stages are: Web Application Layer (WAL), Quality of Service Assurance Certifier Layer (QoSACL), Web Service Layer (WSL), Visualization Input Layer (VIL) and Visualization Output Layer (VOL).

In WAL, the Service Provider (SP) receives request from the Service Requester (SR) through the Stream Socket (SS) with a connection-oriented transmission control protocol that provide appropriate website for rendering of services in WSL. Consequently, SP provide data as requested through the QoSACL who broadcast various services data based on the requirement of the SR. The QoSACL main duty is to check and verify the claims made by the SP about its quality of service. The SR picks the data service that satisfy its needs based on the quality of service they invoke. The data is then sieved, processed in VIL for visualization activities in VOL using Hybridized Context-Sensitive Visualization Techniques (HCSVT). The system described here is depicted by the architecture in Fig. 1.

3.1 Web Application Layer (WAL)

The web application layer is where SR log-in to transact the visualization business. SR must be authenticated and authorized in order to transact business on SOADIWA site. The SP expose various Web Application site where various data can be retrieved for the need of SR using Remote Information Retrieval (RIR) approach. RIR enables the search for and retrieval of information located on remote computers (servers) using web browser. The two primary methods of RIR used for the purpose of this work are File Transfer Protocol (FTP) and World Wide Web (WWW). Traditional measures of information retrieval system performance are recognized in modified form by web users. The basic modeled as illustrated by [37] recognizes a three-way trade-off between the speed of information retrieval (response time), precision and recall. This trade-off becomes increasingly difficult to balance as the number of documents and users of a database escalate.

3.2 Visualization Input Layer (VIL)

Visualization Input Layer extracts raw data as input from third party website through the QoSACL to the SR. This layer also filters the raw data to the requirement of SR using Context-Sensitive Visualization (CSV) approach and Visualization Exploration Process (VEP). CSV is a user-centered knowledge representation reflecting user's context models for data exploration and context-sensitive information delivery [38] while VEP is a model that capture the salient aspect of the exploration [39]. The two models were combined to form a new model called HCSVT which captures the essential components of visualization exploration as shown in Fig. 2. During exploration, a user manipulates parameters which are applied to a transform to generate a result; the results are used by the user as feedback to continue the

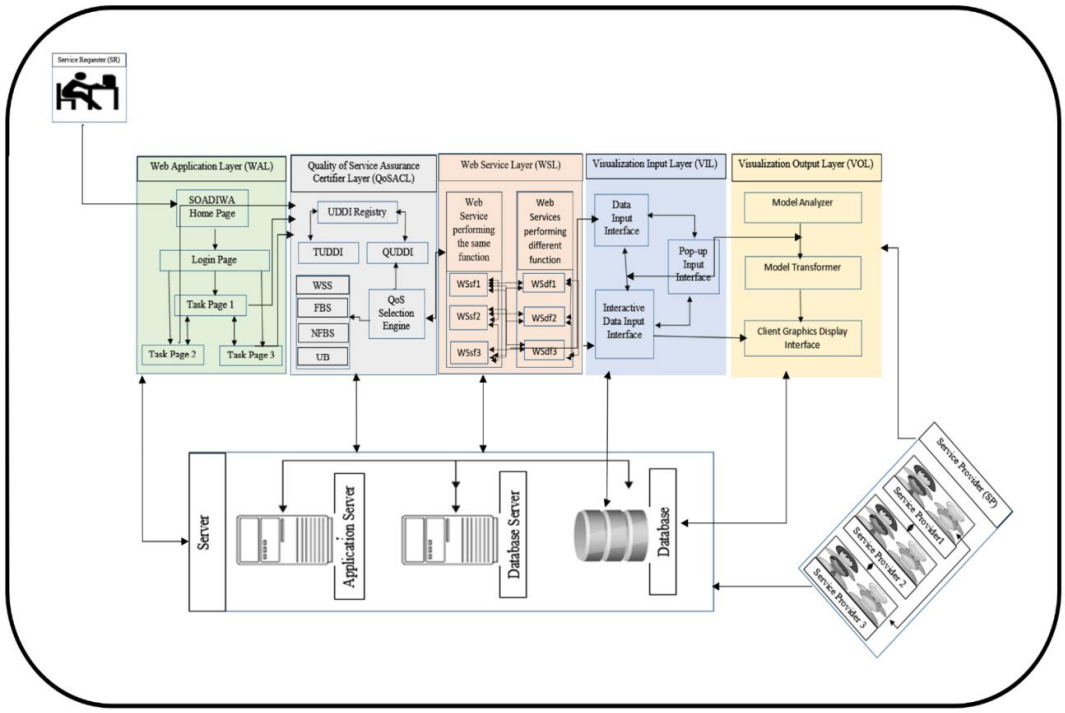

Fig. 1. Five layered architecture for SOADIWA

exploration. The key feature of the new model is that the visualization process follows an iterative sequence with user tracing a path through this space as they explore new result that serve as feedback for further transformation. Each parameter in the visualization transform corresponds to a dimension in the parameter space.

3.3 Visualization Output Layer (VOL)

Visualization output layer is the layer that renders graphics output display for interaction and exploration of graphics information using HCSVT model.

3.4 Quality of Service Assurance Certifier Layer (QoSACL)

The QoSACL is a mediator between the Service Requester (SR) and the Service Provider (SP). It checks and verify the claims made by SP about its quality of service to enable the SR choose the service that satisfy its needs. The VIL invoke WSL over the QoSACL through exchange of messages to collect the need data or information. The current Find-Bind model based on Universal Description and Discovery Integration (UDDI) registry was modified to include functional description of the web service and associated quality of service register in the repository using knowledge-based principles. Knowledge-based is an expert system that uses computer program to simulate the judgement and behavior of a human being or an organization that has expert knowledge and experience. The knowledge-based supplies specific facts and rules regarding the QoSAC domain, while knowledge-based service optimizer serves as an interface between the inference engine (which is needed to offer the reasoning ability that allows the expert system to make conclusion) and the service requester. The SP submit its QoS claims to the QoSAC through the database who check, verify and certify the claim for certification or rejection. The QoSAC register SP certificate and functional description of the service and its associated certified quality of service information back to the Database (Working memory) provided that its QoS claim is correct. SR queries the QoSAC-UDDI based on functional and non-functional (QoS) requirements that satisfy its needs through the Knowledge-Based Service Selector Optimizer (KBSSO). The approach thus described is summarized in Fig. 3.

3.5 Web Service Layer (WSL)

The WSL provides web service data to the VIL through QoSACL using stream socket. In stream

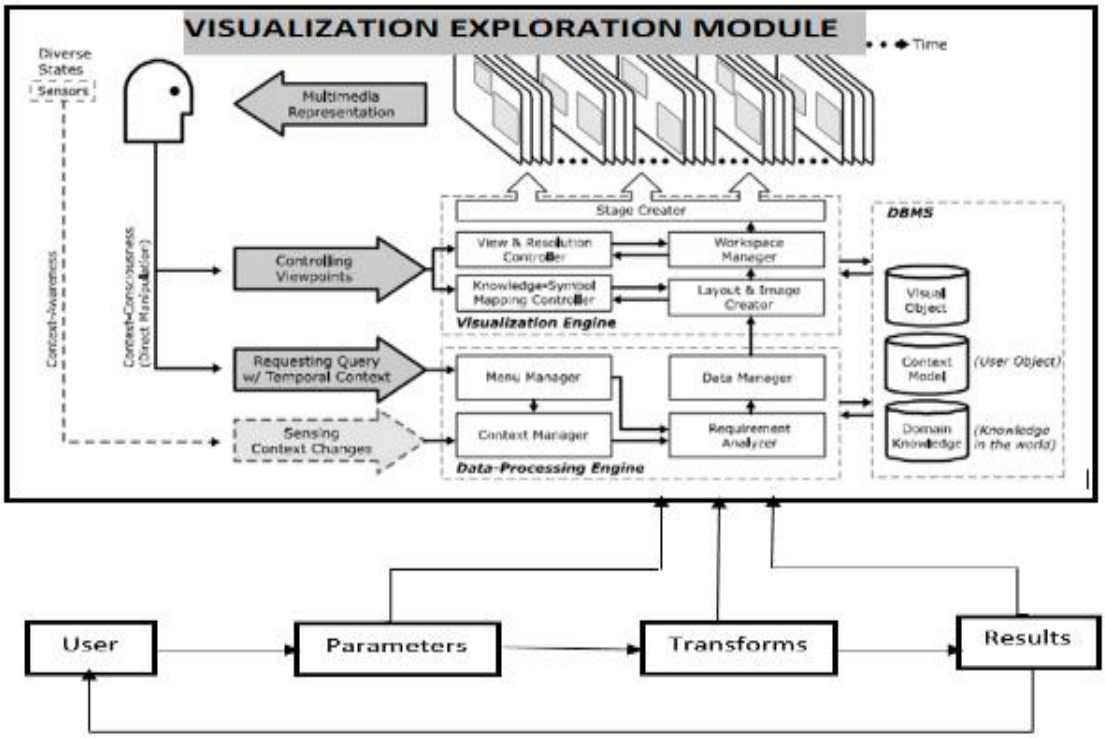

Fig. 2. Hybridized Exploration Component for Context-Sensitive Visualization Process Adapted from [25] and [24]

socket system, a client sends a request message to the sever asking for some web services. Web service is a software system designed to support interoperable machine-to-machine interaction over a network. Internet Protocol (IP) is the method by which data is transferred across the internet. To build web services, service identification must be accomplished to find the services that are already there and services that is needed but don't yet exist which can then be bought or developed from scratch (new service). Identification can also result in the need to modify existing services (reengineering), for example adding additional operations, or promoting existing IT assets into services (wrapping).

3.6 SOADIWA Cost Estimation

The processes of estimating, planning and managing are crucial for web projects development [39]. This can be achieved by combining some existing agile techniques using work breakdown structure approach to produce a model that uses business values to guide the delivery of features. Phased Effort Distribution (PED) method was employed to cost SOADIWA through the identification of three migration strategies namely, wrapping, reengineering and replacement [40,41].

Wrapping: Provides a new SOA interface (for example WSDL) to a legacy component, making it easily accessible by other software components. It is a black-box modernization technique used when the legacy code is too expensive to re-write, relatively small, can be reused, and cost effective solution is needed.

Reengineering: Reengineering is the analysis and adjustment of an application in order to represent it in a new form. It includes activities such as reverse engineering, restructuring, redesigning, and re-implementing software.

Replacement: Replacement is the retirement of an application and replacing it with an of the-shelf package or a complete rewrite of the legacy system from scratch. An organization may choose replacement strategy if wrapping and reengineering impose too much cost.

Five phases were also recognised from Software Development Life Cycle (SDLC) principle [42]. Factors relating to cost of SOA were also

extracted from past literatures [43,44] to enhance each stages of the phases. The cost factors were distributed among the five phases and weighted for each identified migration strategy.

Assume Migration Strategy (MS) in phase P(n) is a matrix of rectangular array of a membership function defined in equation 6:

$$MS = ms(i, j) = \begin{pmatrix} ms_{11} & ms_{12} & ms_{13} & \dots & ms_{1j} \\ ms_{21} & ms_{22} & ms_{23} & \dots & ms_{2j} \\ \dots & \dots & \dots & \dots & \dots \\ ms_{i1} & ms_{i2} & ms_{i3} & \dots & ms_{ij} \end{pmatrix} \quad (6)$$

i = number of factors in each phase; j = number of strategies considered

$$P = \sum_{n=1}^{n}(P_n) \quad (7)$$

$$TNF = i * P \quad (8)$$

$$MTW = TNF * j \quad (9)$$

Weighted cost for each strategy is the sum of column matrix in (6) defined as:

$$WCW(n) = \sum_{i=1}^{i}(a_{i1}) \quad (10)$$

$$WCR(n) = \sum_{i=1}^{i}(a_{i2}) \quad (11)$$

$$WCP(n) = \sum_{i=1}^{i}(a_{i3}) \quad (12)$$

Where:

P = Total Number of Phases;

TNF = Total Number of Factors;

MTW = Maximum total weight in each strategy;

WCW(n) = weighted cost of wrapping in each phase;

WCR(n) = weighted cost of reengineering in each phase;

WCP(n) = weighted cost of replacement in each phase

$$TWCW = \sum_{n=1}^{n}(WCW_n) \quad (13)$$

$$TWCR = \sum_{n=1}^{n}(WCR_n) \quad (14)$$

$$TWCP = \sum_{n=1}^{n}(WCP_n) \quad (15)$$

$$\text{Phase Ratio (Wrapping)} = 100 * \frac{WCW(n)}{TWCW} \quad (16)$$

$$\text{Phase Ratio (Reengineering)} = 100 * \frac{WCR(n)}{TWCR} \quad (17)$$

$$\text{Phase Ratio (Replacement)} = 100 * \frac{WCP(n)}{TWCP} \quad (18)$$

MS = {TWCW, TWCR, TWCP}

MS = {TWCW | overall weighted cost of wrapping to wrap existing legacy systems}

MS = {TWCR | overall weighted cost of reengineering to adjust legacy applications by adding functionalities to form a service}

MS = {TWCP | overall weighted cost of replacement to replace an old application with a new one}

Phases = plrq ∪ desg ∪ devp ∪ test ∪ intg

Where:

plrq = {bua, ipc, buv, bur}
desg = {nor, olst, ern, nsc}
devp = {flex, cos, tsp, trm}
test = {fut, nft, inte, regt}
intg = {ste, mpm, sepl}

P ⊆ MS

Plrq = {plrq | plrq is a set of planning and requirement which define where major function of the service is defined i.e., business agility (bua), integration with partners' cost (ipc), business values (buv) and business risk (bur)}

desg = {desg | desg is a set of Design which define where the target service is described so that skilled developers can develop the service with minimal effort i.e., need for original requirements (nor), obsolete legacy sytem technology (olst), experience resources needed (ern) and Need for source code (nsc)}

devp = {devp | devp is a set of Development which involves writing the code that satisfies both requirements and design previously documented i.e., flexibility (flex), code size (cos), tool support (tsp) and time required for migration (trm)}

test = {test | test is a set of Testing which is where all test cases are run to validate and verify the service i.e., functional testing (fut), non-functional testing (nft), integration testing (inte) and regression testing (regt)}

intg = {intg | intg is a set of Integration which is where the services are integrated with the desired application and the gap between existing system and target developed system is identified and changes made i.e., stable environment (ste), maintainability post migration (mpm), solving existing problems in legacy systems (sepl)}

Weight assigned to strategies according to Esraa and Ramadan [16] varies from High, Medium to Low depending on the justification of the factor considered and its effect on the migration strategy hence, High → 3, Medium → 2 and Low → 1.

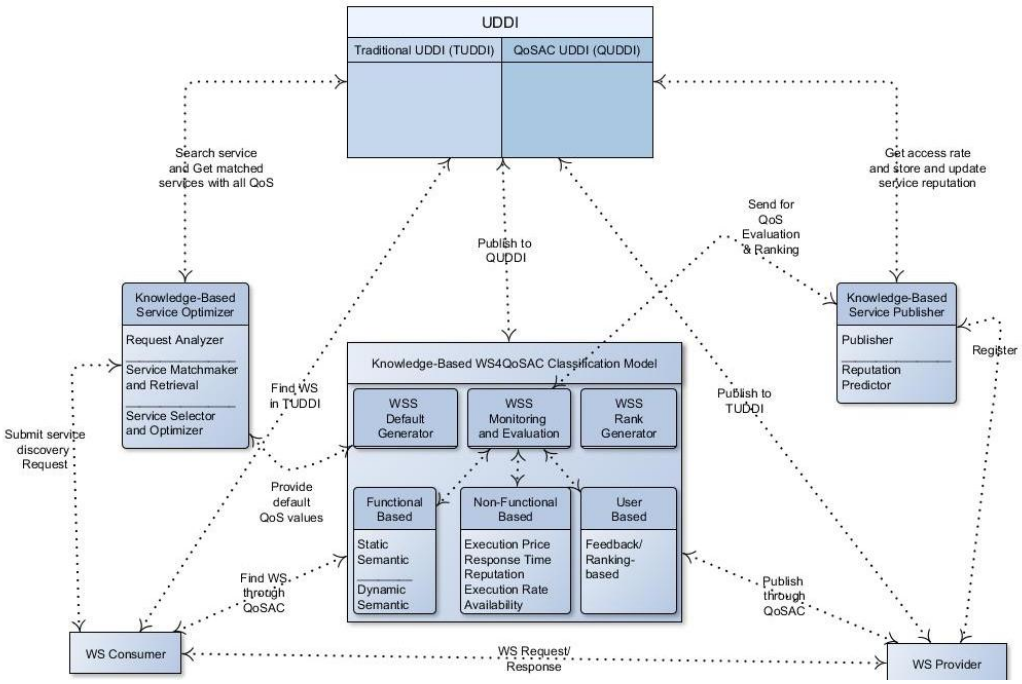

Fig. 3. Knowledge-based web service for quality of service assurance certifier

4. IMPLEMENTATION AND TESTING

4.1 Implementation

SOADIWA was implemented using Visual Studio .Net Express 2017 Community Edition on Window 10 Professional, 64-bit Operating System with Intel core duo CPU at 2.00 GHz, 4 GB memory. Java Scripts were used in implementing the core program while NuGet packages was employed to enhance code execution. A web page was created as SOADIWA page which was partitioned into five segments as shown in Fig. 4.

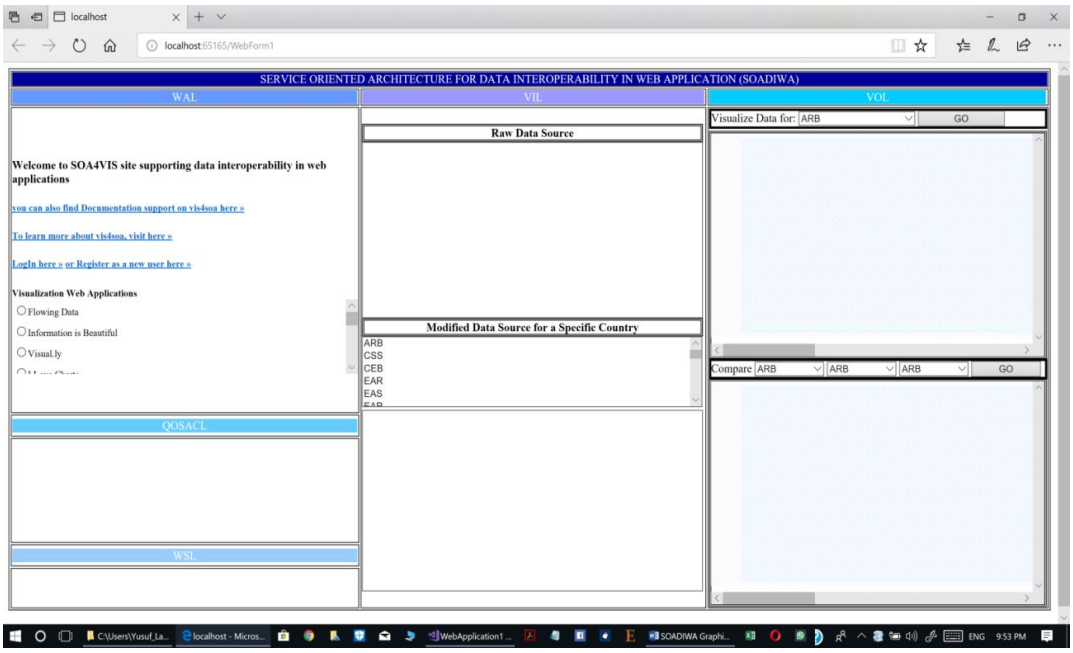

Fig. 4. SOADIWA web page

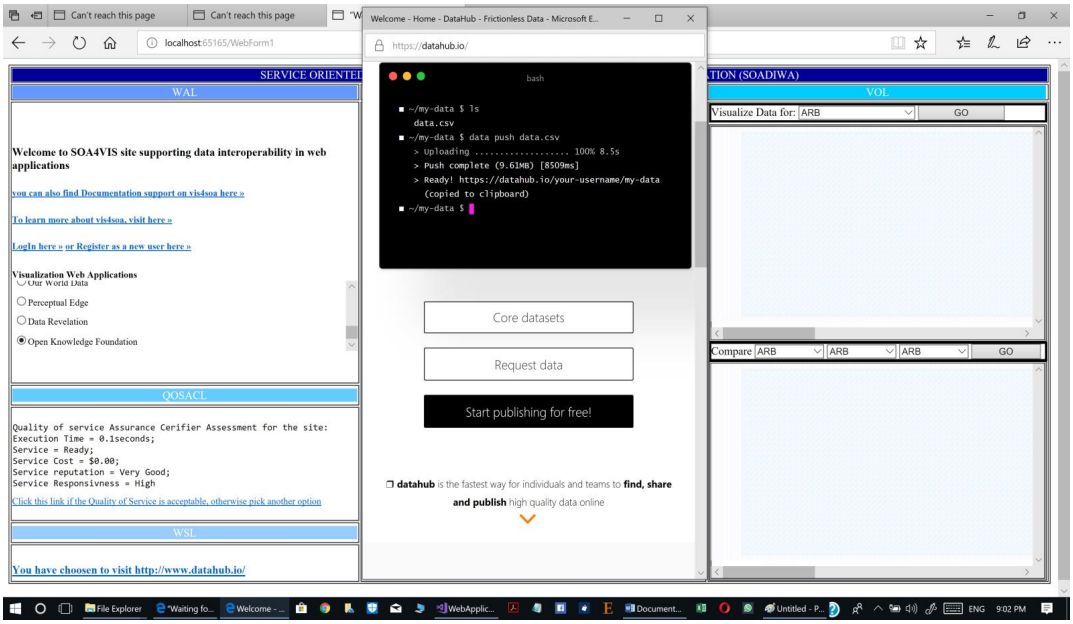

Fig. 5. Web service with data repository request popup

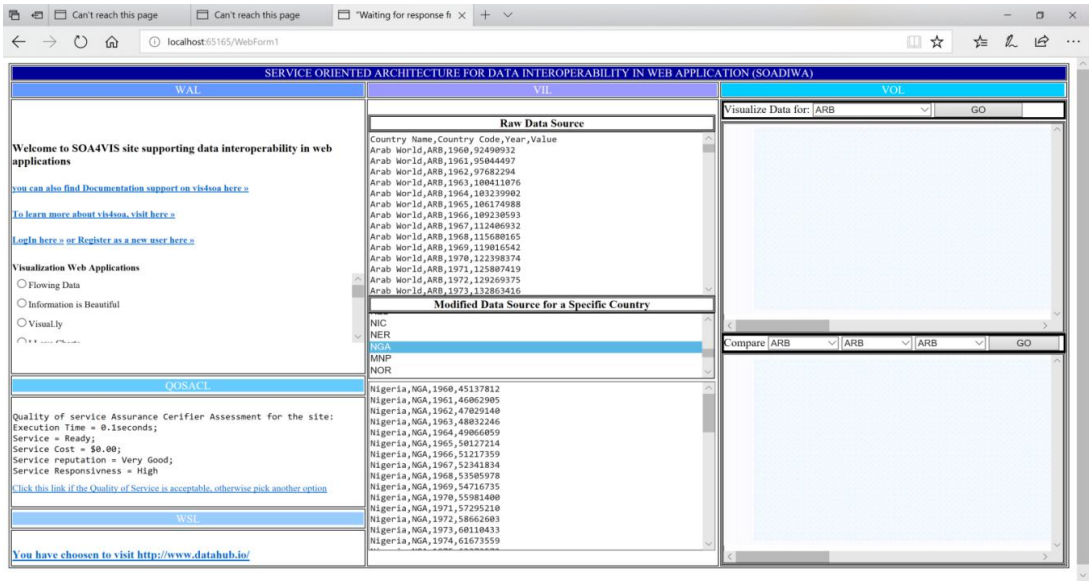

Fig. 6. Requested Raw and Modified Data

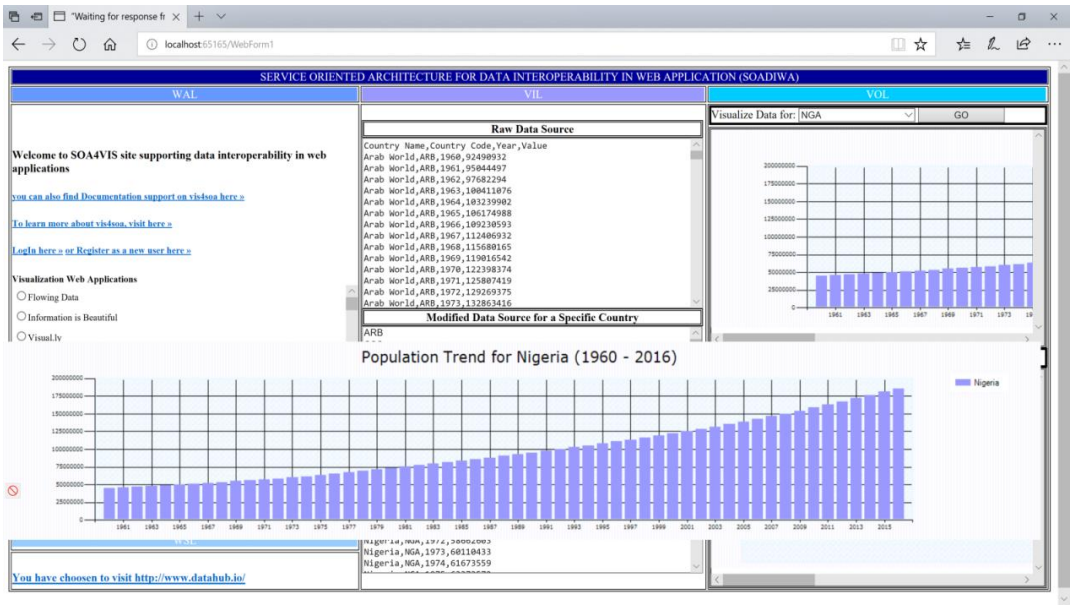

Fig. 7. Visualized Data showing population trend for Nigeria from 1960 - 2016

The WAL section is where the SR log-in or register in other to use the page. SR must be authenticated and authorised to access the "Visualization Web Applications" (VWA). The VWA is where SP expose web services that perform the same functions but with diverse quality of services. For example, the author intends to collect population data of different countries for analysis. Population data for various countries are already in the repository of many sites as web service which SR access based on the service agreement between the SP and the owners of these sites with the quality of service assurance certifier as a mediator. These web service sites were exposed using "Radio button list" as revealed in Fig. 5 when "Open Knowledge Foundation" site is checked.

The corresponding quality of service is exposed in the QOSACL segment which the SR is at liberty to accept or reject. The SR on acceptance of the site click the link button "Click this link if

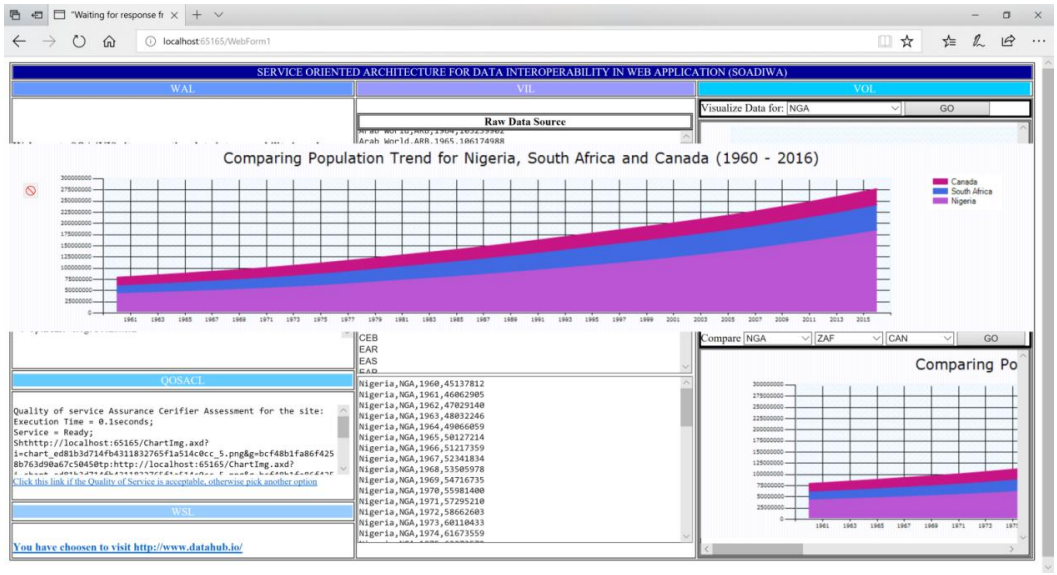

Fig. 8. Population trend for Nigeria, South Africa and Canada from 1960 to 2016

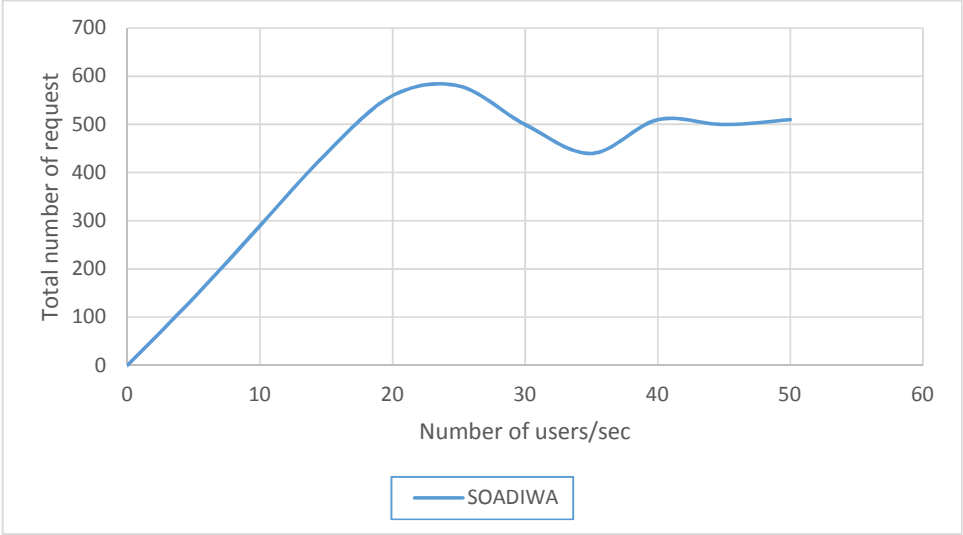

Fig. 9. Traffic performance trend

the Quality of service is acceptable, otherwise pick another option". The link thereafter exposed the requested website as "You have choosen to visit http://www.datahub.io/" link the WSL segment. On clicking the link, the website for the data repository popup (Fig. 5) while the requested data is automatically downloaded into VIL segment in Fig. 6.

The downloaded data which is for the whole countries in the world can be extracted for a specific country using the "Modify Data Source for a Specific Country" list box as experimented for Nigeria population from 1960 to 2016 in Fig. 6. The visualized data is display in VOL segment as shown in Fig. 7.

The web application was also used to compare population data between three countries namely Nigeria, South Africa and Canada to discover trends and anomalies. The visualized result is display in Fig. 8.

4.2 System Testing and Performance Evaluation

The system tool web service was tested for traffic trends using LoadUI automation tool. The trend

between currently running requests was measured against currently queued request for varying set of users per second. The result revealed that there was no queue when the user rate/sec was between 5users/sec and 15 users/sec. The queue starts building up at 20 users per second and it is at maximum in 25 users/sec at 20seconds. The result was employed to plot traffic performance trend which revealed an optimal performance at 25 users/sec of 580 requests as shown in Fig. 9.

5. CONCLUSION

This paper focused on the development of Service-Oriented Architecture (SOA) and its Design Strategies for a system-enabled visualization and exchange of messages with web services. SOADIWA is an internet-based user-initiated system based on five-layered architecture. In WAL, the Service Requester (SR) initiates a request for data from the Service Provider (SP) through the QoSACL to provide appropriate website via WSL for rendering of services which must be accepted, processed and returned for a particular need in VIL. The requested data is filtered in VIL for data exploration and analysis in VOL using context-sensitive visualization techniques. A negotiation mechanism is provided between clients and service providers to reach a mutually-agreed QoS goals. Phase Effort Distribution (PED) metric was developed to cost SOADIWA using three migration strategies namely; wrapping, reengineering and replacement. The PED metric allows organizations to make informed decision about whether to build a new service or migrate a legacy system as service. SOADIWA was also implemented using Visual Studio .Net Express 2017 Community Edition on Window 10 Professional, 64-bit Operating System with Intel core duo CPU at 2.00 GHz, 4 GB memory. With Java Scripts and NuGet packages to implement the core program. Moving to SOA involves trade-offs, for this reasons enterprise must be ready and willing to address the many issues facing distributed computing. Many of these challenges can be successfully addressed and overcome through a combination of open source tools, commercial tools and custom solutions as illustrated in this work. The interoperation of visualization-based systems with web applications and services for IT business process is a major contribution of this work. Despite the strong trend in SOA, some in the IT industry do not feel that the web services underpinning for an SOA is mature enough for their enterprise to consider migration to a service-oriented architecture. It is therefore advocated that software vendors should imbibe this paradigm shift in line with the industry movement or risk their own extinction. There are currently few available metrics for the detailed cost estimation for SOA-based software development. Future research will develop new metrics to resolve this issue. Also, in the future, user behaviour will be studied based on the services they invoked and try to get a trust factor based on the profile data of the consumers.

COMPETING INTERESTS

Author has declared that no competing interests exist.

REFERENCES

1. Surbhi M, Chetna G. A machine learning based efficient software reusability prediction model for java based object oriented software. I. J. Information Technology and Computer Science. 2014;02:1-13.
2. Soudeh P, Esmaeel K, Mehrdad J. A novel approach: A hybrid semantic matchmaker for service discovery in service oriented architecture. International Journal of Network Security & Its Applications (IJNSA). 2014;6(1)37-48.
3. Ismael N, Jose FA. BioSStore: A client interface for a repository of semantically annotated bioinformatics web services. Open Journal of Semantic Web (OJSW). 2014;1(1):19-29.
4. Bojan B, Zeljka M. Visualization services based on web services. Journal of Computing and Information Technology. 2007;15(4):339-345.
5. Christain E, Kim L, Karsten K, Stefan L, Christof M, Sebastian A. The SOA's layers. Cooperation & Management, Universitat Karlsruhe; 2012.
Available:http://cm.tm.kit.edu/CM-Web/05.Publikationen/2006/[EL+06]_The_SOAs_Layers.pdf
[Accessed online: 29-01-2015]
6. Stef E, Danny H, Jorik B, Jarke J. Dynamic network visualization with extended massive sequence views. IEEE Transactions on Visualization and Computer Graphics. 2014;20(8):1087-1099.
7. Riedl C, Bohmann T, Rosemann M, Krcmar H. Quality aspect in service

ecosystems: Areas for exploitation and exploration. In proceedings of international conference on electronic commerce (ICEC 08); 2008.
(15-01-2016)
Available:http://eprints.qut.edu.au/71353/

8. Esraa AF, Ramadan M. Phased effort estimation of legacy systems migration to service-oriented architecture. International Journal of Computer and Information Technology. 2014;03(3):661-670.
ISSN: 2239-0764.

9. O'Brien L. A framework for scope, cost and effort estimation for Service Oriented Architecture (SOA) Projects Proc. 20th Australian Software Engineering Conference (ASWEC'09), IEEE Press. 2009;101-110.
DOI: 10.1109/ASWEC.2009.35

10. Nino-Ruiz M, Stock C, Bishop I, Pettit C. Service oriented support for heterogeneous software tools in environmental modelling and visualization. 18th World IMACS/MODSIM Congress, Cairns, Australia. 2009;13-17.

11. Wood J, Brodlie K, Seo J, Duke D, Walton J. A web services architecture for visualization. In e-Science; 2008. IEEE Fourth International Conference on IEEE. 2008;1-7.

12. Zheng L, Jacky K. Software cost estimation framework for service-oriented architecture systems using divide-and-conquer approach. Proceeding of the Fifth IEEE International Symposium on Service Oriented System Engineering. 2010;47-54.

13. Brown A, Johnston S, Kelly K. Using service-oriented architecture and component-based development to build web service applications. Santa Clara, CA: Rational Software Corporation; 2002.
Available:http://dit.isuct.ru/Publish_RUP/soa.rup_soma/guidances/whitepapers/resources/using_soa_and_cbd_to_build_ws_applications.pdf

14. Lloyd B. Meaningful cost-benefit analysis for service-oriented architecture projects. Proceedings of the Seventh Annual Acquisition Research Symposium Thursday Sessions. 2010;2:471-475.

15. Huang GQ, Mak KL. WeBid: A web-based framework to support early supplier involvement in new product development. Robotics and Computer Integrated Manufacturing. 2000;16(2-3):169-179.

16. Laxmareddy A, Ganesan M, Kannan E, Dhilsath FM, Saravanan MS. A task oriented view of web software visualization and architecture. International Journal of Computer Science and Information Technologies. 2014;5(2):1219-1222.

17. Oleg S, Krystof L, Scott D, Frank D. Introduction to Information Visualization (InfoVis) techniques for model-based systems engineering. Procedia Computer Science. 2013;16:49-58.
Available:www.sciencedirect.com

18. Sima E, Raziye HH. Critical factors in the effective of service-oriented architecture. Advance in computer Scien30ce: An International Journal. 2013;2(3-4):26-30.

19. Babu RB, Lakshmi TC, Deepthi KP. Public oriented personalized health care platform based on web service. Global Journal of Computer Science and Technology (B). 2015;15(2).
Available:https://globaljournals.org/GJCST_Volume15/1-Public-Oriented-Personalized.pdf

20. Anibrika BSK. Implementing Enterprise sErvice Bus (ESB) architecture as a business model for electronic-commerce system. Africa Development and Resources Institute Journal, Ghana. 2014;8(2).
Available:http://journals.adrri.org/wp-content/uploads/2014/12/Implementing-Enterprise-Service-Bus-ESB-Architecture-as-a-Business-Model-.pdf

21. Mohamed IB, Mohamed SAR. Constructing solutions to SOA attacks on SOAP web services – A literature review. International Journal of Scientific Engineering and Technology. 2014;3(5): 564-569.

22. Jamal SM, Asif M. Grid approach with metadata of messages in service oriented architecture. I. J. Information Technology and Computer Science. 2014;02:64-71.

23. Miquel S. Interoperable intelligent environment decision support systems: A Framework Proposal. International Environmental Modelling and Software Society (iEMSs), 7th Intl. Congress on Env. Modelling and Software, San Diego, CA, USA; 2014.

Avaialble:http://www.iemss.org/society/index.php/Iemss-2014-proceedings

24. Bellocchi G, Foscarini F, Canonico M, Van den Eede G. Web components for development of computational methods: Example with Fuzzy. Proceedings of the iEMSs Fourth Biennial Meeting. 2008;2:1077-108.
25. Rajalakshmi S. A software ability network in service oriented architecture. International Journal of Science and Technology Education Research. 2014;5(2):7-14.
26. Susila B, Vadivel S. A novel approach to web service selection based on QoS through service oriented architecture. International Journal of Computer Science & Network Solutions. 2014;2(1):16-25.
27. Maricela B. Similarity measures for web service composition models. International Journal on Web Service Computing (IJWSC). 2014;5(1):1-16.
28. Weina L, Xiaohui H, Shangguang W, Xiaotao L. A multi-criteria QoS-aware trust service composition algorithm in cloud computing environments. International Journal of Grid and Distributed Computing. 2014;7(1):77-88.
29. Xue-Jie Z, Zhi-Jian W, Wei-Jiang Z. Personalized context-aware recommendation approach for web services. International Journal of Security and Its Applications. 2015;9(8):35-44.
30. Kaur J, Singh N. Quality of service based impact provider model for composition of web services. International Journal of Innovative Research in Computer and Communication Engineering. 2015;3(4): 3071-3079.
31. Kheradmand M, Motameni H. An approach to web service selection based on composite QoS parameters. Advances in Science and Technology Research Journal. 2015;9(26):129-136.
32. Gurjar NR, Rode SV. Personalized QoS-aware web service recommendation via exploiting location and collaborative filtering. International Journal of Advanced Research in Computer Science and Software Engineering. 2015a;5(1):695-698.
33. Miksa T, Mayer R. Ensuring sustainability of web services dependent processes. Int. J. Computational Science and Engineering. 2015;10(1/2):70-81.
34. Suria S, Palanivel K. An enhanced web service recommendation system with ranking QoS information. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS). 2015;4(1):116-121.
35. Elabd E. A dynamic reputation-based approach for web services discovery. I. J. Information Technology and Computer Science. 2015;08.31-36.

    Available:http://www.mecs-press.org/ijitcs/ijitcs-v7-n8/IJITCS-V7-N8-5.pdf
36. Gurjar NR, Rode SV. Web service recommendation based on usage history. International Journal of Advanced Research in Computer and Communication Engineering. 2015b;4(6): 173-177.
37. Mei K, Koichi T. Information retrieval on the web. ACM Computing Surveys. 2000;32(2):144-173.
38. Jung E, Sato K. A framework of context-sensitive visualization for user-centered interactive systems. Proceedings of 10th International Conference on User Modeling. 2005;423-427.
39. Jankun-Kelly TJ, Ma KL, Gertz M. A model and framework for visualization explorations. IEEE Transaction on Visualization and Computer Graphics. 2007;13(2):369-377.
40. Torrecilla-Salinas CJ, Sedeno J, Escalona MJ, Mejias M. Estimating, planning and managing agile web development projects under a value-based perspective. Information and Software Technology. 2015;61:124-144.
41. Almonaies AA, Cordy JR, Dean TR. Legacy system evolution towards service-oriented architecture; 2010.

    (12-11-2014)

    Available:http://research.cs.queensu.ca/home/cordy/papers/ACD_MigToSOA_SOAME10.pdf
42. Pelican E. The Software Development Life Cycle (SDLC). Document ID: REF-0-02, Version 2.0; 2014.

    (12-11-2014)

    Available:http://www.pelicaneng.com/DevDocs/sdlc.pdf
43. Erradi A, Anand S, Kulkami N. Evaluation of strategies for integrating legacy application of strategies for integrating legacy applications as services in a

service-oriented architecture. IEEE International Conference on Services Computing. 2006;257-260.

44. Zheng L, Liam O. A qualitative approach to effort judgment for web service composition-based SOA implementations. Proceedings of the 1 IEEE International Conference on Advanced Information Networking and Applications. 2011;586-593.

© 2019 Oladimejia; This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Peer-review history:
The peer review history for this paper can be accessed here:
http://www.sdiarticle4.com/review-history/52337