International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 4, August 2020, pp. 3883~3888
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i4.pp3883-3888

Formalization of SOA concepts with mathematical foundation

Asha H. V.[1], Shantharam Nayak[2], Annamma Abraham[3]
[1]VTU, Department of CSE, Nitte Meenakshi Institute of Technology, India
[2]Department of ISE, Rashtreeya Vidyalaya College of Engineering, India
[3]Department of Mathematics, BMSIT, India

Article Info

Article history:

Received Aug 1, 2019
Revised Des 1, 2019
Accepted Jan 7, 2020

Keywords:

Choreography
Service oriented architecture
Web service orchestration

ABSTRACT

With the development of the IT industry, agility and dynamicity are the two expected characteristics for any business enterprise. Service Oriented Architecture (SOA) an emerging and a promising advent aiming at address the current trends/needs of enterprise business applications. The current work aims at giving an overview of the conceptual model of SOA based on set theory approach. Also, it posits mathematical definitions for logical, environment and application service definition with the help of general systems theory and mathematical foundation. The mathematical model, definition and functionality of services enhance SOA principles which can be used efficiently in integrating heterogeneous components.

Corresponding Author:

Asha H.V,
Department of Computer Science and Engineering,
Nitte Meenakshi Institute of Technology,
Bangalore-560064, India.
Email: ashahebbandi@gmail.com

1. INTRODUCTION

With the advancement of time, the growing density of current complex software systems has continually resulted in the development of new programming technologies/paradigms. Every successive archetype has popularized a new abstraction level. One such paradigm is Service Oriented Architecture (SOA). Through formalization the basic concepts from object oriented to distributed computing to service oriented has evolved significantly.

As stated by OASIS, SOA is an “architectural paradigm for utilizing independent distributed capabilities that are/may be under the control of diverse ownership domains”. SOA aims at delivering standards based, self-describing, loosely coupled, platform and protocol independent functionality [1, 2] of an enterprise by providing a way of sharing the services via internet. The architecture aims to provide solutions to the existing bequest applications irrespective of their platform, technology, language or operation mode. According to the Open Group's definition, SOA is considered as one of the architectural pattern which support service orientation and treats each individual task as atomic services.

Many related SOA specifications and standards are developed by organizations and companies. At the conceptual level, these specifications and standards define the basic concepts and terms of SOA with the help of abstractions like Reference Models, Maturity Models, Reference Architectures, Concrete/Solution Architectures, Ontology’s, Modeling Languages, for different categories of users dealing with SOA based systems [3]. However, most of them focus on assembling applications through web services and enterprise information systems. A prevalent problem, however is that, there are no design principles that guide service development, service granularity or component development that implements them. Even though few of the basic foundational concepts of service orientation are highlighted in [4-6], they lack mathematical foundations and formalized principles in service orientation.

Formalization is essential for identifying appropriate software design methodologies and also to provide supporting tools to fulfill particular service oriented application challenges like platform independence, composability and adaptability. Automatic or semiautomatic synthesis of complicated web services from a collection of basic web services is one of SOA's basic values. This process is known as orchestration. Based on business objectives, it's vital to organize the collaborative web service in the required sequence of interaction through message transfer. This process is called choreography [7]. A lot of languages, tools and multiple approaches are available to accomplish these. Yet, it should be noted that to describe an SOA intuitive concepts, verbal descriptions are required. Despite all these, for a more in depth understanding and implementation of SOA and their properties, there is a need for proper theoretical /mathematical foundation. As an initial step, this paper discusses the theoretical framework and mathematical approach for SOA based on general system theory. Here, the possibility of set theory approach to model the structures of SOA based system is described.

An SOA-based service is responsible for maintaining its own state and hence termed as self-contained. These services limit the interface contract to platform-independent assertions and also SOA presumes to dynamically locate services, invoke and recombine the services. Shortly SOA can be termed as self-contained, dynamic and platform-independent.

In a broader sense, SOA is viewed as a defined combination of service implementation and an interface. The integrity of the services is defined by the service interface along with their invocation logistics. The service implementation is required to implement the work which is designated to the service. The platform independent feature of the interface helps the client to use any platform for computation, any operating system and use of any language for programming irrespective of which communication device is used by the client. Thus implementation and configuration of the services are described and managed as separate components, although they are strongly interrelated. Two key aspects service requestor and provider communicate through service request and emulates the type of participator in SOA [8, 9]. This interaction incorporates to find services, bind them and publish the services.

2. RELATED WORK

Service oriented programming is stationed on the features of loosely coupled and protocol independent that are made available through the network [1, 10, 11]. In SOA, each process is treated as a service that includes business, scientific and technical processes. Each step in the business process is associated with a service reflecting specific functionality [12]. Service is considered in two different aspects with respect to SOA:

a. Service Interface: It identifies the service, defines the criterion and protocol for transmitting the outcomes of the service as a response to the customer.
b. Service Execution/ Implementation: Provides business logic as stateless computation.

Basically, SOA interacts with software agents through message exchange protocol by finding, binding and publishing services through a published interface involving service requesters and service providers. SOA identifies three forms of service [13]:

a. Infrastructure service: helps to manage and monitor including identification and security.
b. Business-neutral service: deals with service broker, notification detail and scheduling of service.
c. Business service: encompass business domain service.

SOA follows a find-bind execute paradigm consisting: service provider, consumer, and registry. The service provider makes the components available for consumers via the internet by registering and publishing them in service registers through service brokers. Each service has a separate contract and functionality interface. The service consumers will make use of the service published by service provider depending on their needs by accessing contract detail and endpoint address.

Inspired by the internet and web, SOAs' architectural style enables extensible interoperability. The splitting up of service interfaces from its implementation helps in better planning, development, and integration of enterprise applications. SOA based service is self contained and platform independent which are dynamically deployed, invoked, located, organized or combined. In order to attain this, SOA evolves as service-based multitier architecture with variegate technical implementation capability. The benefits of implementing SOA in enterprise architecture are discussed in [14]. A lot of researchers are contributing to the standardization of SOA interfaces and protocols using which guidelines of practical design and implementation of SOA based systems are developed. [15-17] discuss some well designed conceptual models for different levels of use of the SOA paradigm. These efforts have allowed a wide practical use of an SOA approach to business and design of complex software system. However, there is certain difficulty, consisting of the need for self-selection methodologies of analysis, designing and implementing SOA based systems [18]. Also, the specifications and standards are mainly expressed with the help of diagrams and verbal descriptions [7].

One of the major causes of the failure of the software projects is lack of discipline [19, 20]. As stated by [3] it is accepted that there is a need to develop mathematical frameworks to describe the main properties and relationship of SOA components. To create mathematical foundations of SOA based systems it is necessary to use the appropriate mathematical tools, which reflect the structural properties of SOA based systems to describe the behavior of web services. In [21] object-oriented paradigm based on a mathematical foundation is discussed and [22] highlights about software requirement approach. M.Broy [23] has provided a mathematical concept for component usage and in [24] refinement of systems in interactive model is discussed. G.D. Abowd et.al in [25] have highlighted the need for a semantic domain wherein they have explained the mathematical model for defining the style of architecture, axioms constraining the structure and mathematical proofs for the same. Alexander et.al [26] have contributed to a better understanding of architectural styles. [27] have presented an overview about the need of formulation for better results with respect to SOA.

3. MATHEMATICAL FORMULATION FOR SOA

According to [7], at a given moment T, the number of dynamically accessible service determines the capability of service orientation and the convention of software design. Thus, it can be stated that:

SOA = Principle + Platform + Technology

Each design will be having the requirements of its own which will be different from others. Even though providing a universal platform that fits all the tasks would be impossible [3] it ensures designing an assortment of products with potential integration viability.

According to [18], it is better to define the system in terms of observed features and relationships. These features can be used for the classification of organization and inter-relationships of components into a system. Accordingly, a Web Service can be described as:

Definition 1: A family of a set of web service properties is $V = \{Vi : i \in I\}$, where I is finite or infinite index set. A web service is a relation on these set of properties

$$S \subset x \{Vi : i \in I\} \quad (1)$$

Thus, a web service is a relation on the sets of its observed properties and appearances, abstracting from the domain where web service is used, and the technology with which it is implemented, but the representation of (1) is too general to explore the specific characteristics of web service as part of SOA. SOA based systems may have stakeholders who play the role of the owner, broker, delegate of any of the participants, provider of telecommunications etc as its participants. Hence, each of the said participants can see only their interested or in any way provided to it appearances of web services [15, 28].

Considering the three main participants in SOA: consumer, provider, and registry, the appearance that can be observed by each participant for the (1) is defined as:

Definition 2: Consumer's View (SC), Provider's View (SP) and Registry's View (SR) of web service as given in (1) have the following relations, respectively:

$$SC \subset x \{Vi : i \in IC\} \quad (2)$$

$$SP \subset x \{Vi : i \in IP\} \quad (3)$$

$$SR \subset x \{Vi : i \in IR\} \quad (4)$$

where, $IC \subset I$, $IP \subset I$, $IR \subset I$;

$$IC \cup IP \cup IR = I \text{ and } IC \cap IP \cap IR \neq \emptyset$$

Note, IC, IP and IR are not partition of I, because some elements of the set Vi can be common for the consumer, provider, and registry. Service orientations agility and dynamicity enable to design of the products with potential integration ability. Individual services, its working environments, and its applications (composed services) must be given equal importance to make the most of SOA capabilities.

3.1. Service oriented logical definition

An SOA is assumed to be consisting of n processes and services. A service i is deployed to provide the functionality fi symbolized as programmatic interface Ii . Service is capable of dynamically interacting with other entities in the given environment. As a consequence, logical service is the implementation of a set of synchronized and interacting processes:

$$Si = < P1\ i, P2i\ , \ldots\ldots . Pni, \wedge > \qquad (5)$$

where, S_i is logical service exemplar, $P_k{}^i$ - kth exemplar of implementing logical services functionality f_i via the programmatic interface I_i, and ^ - network communication function among discrete processes.

3.2. Service oriented environment definition

At any given instant of time T, environment services include finite set of accessible logical services

$$Env_T = < S_1; S_2; : : : ; S_n >; \qquad (6)$$

where, *n* - represents the logical service available.
By and large, service environments are classified based on:
a. Communication protocol used such as HTTP/S, SMTP.
b. Service Content.

3.3. Service oriented application definition

Logical services of the corresponding application at a given moment T determines the functionality F of service oriented application *A*.

$$F_A = < S_1{}^A, S_2{}^A, \ldots\ldots\ldots , S_n{}^A > Env_T \qquad (7)$$

Representation of Application Functionality in a directing graph is represented as:

$$V_A = (F_A, G) \qquad (8)$$

It represents vertices from F_A set and verge set *G* formalizing the synchronization among individual logical services of the F_A set. Lastly, Service Oriented application *A* is formalized as:

$$A = < F_A, V_A > \qquad (9)$$

Coordination ability in service oriented application is expected for a coherent implementation of service as a specific service will not be familiar with other application in which it participates [7]. V_A defined in (8) of application functionality directing graph represents such competence. In tangible application, V_A possibly will express implementation encapsulating service processes [29, 30].

4. DISCUSSION AND FUTURE WORK

Architecture places an important role in success of any application. Formalized concepts proved mathematically guarantees the result of the particular concept. The current work has focused on defining services through mathematical framework for capturing elements of service oriented paradigm. It is believed that it enhances the requirements of SOA application. Any service in SOA has to be implemented without any knowledge of the application it participates in [11] agreeing to specified Quality of Service Agreement [31]. The coordination capability ensures context invariant implementation of services as mentioned in (8). Implementation of SOA methodology follows the following steps:
a. Requirements of the application are collected.
b. Functionalities are defined.
c. Decompose the functionalities into individual services.
d. Create the functionality graph.
e. Realize the application.

In future it is planned to experiment with the following goals: Defining the services of an application based on the above discussion. Observing the behavior of the system while integrating different services of the application, Refinement of the model if requirement. Later, the finding will be extending on how to improve integration testing of services in SOA.

5. IMPLEMENTATION AND EXPECTED RESULTS

Proposed structure of service oriented methodology [27] follows the following steps: first specification criteria for the concept of functionalities are collected; then, the functionalities of the program are broken down into individual resources; thus the application features graph will be built; finally, the service functionality and feature graph is implemented for realizing the application. Based on the current discussion [32] has created an SOA related architectural design for defining services for an educational system. It consists of two views: Consumer and Provider view. The main objective of designing this rchitecture is to improve the accessibility, flexibility and extensibility of the application with the help of appropriate mechanism, approach and tools. It is possible to offer each service through single or multiple events. The expected results are reusable components which can be used to compose equipped services of the required system. For realizing the application, Gurunathan et.al considered two views during developed of the system: learning resources and research resources.

The system proposed byYanchuk et.al. was analyzed against the service performance test against 10 users considering its bandwidth, load on the CPU, and traffic on the network. According to the results found, 16ms was the average click time for the users considered. The instance of time taken by protocol to instigate the URL is reflected in Figure 1. Similarly, Figure 2 demonstrated the server and the user bandwidth test. The detailed discussion about developing the service for educational system is found in [32].

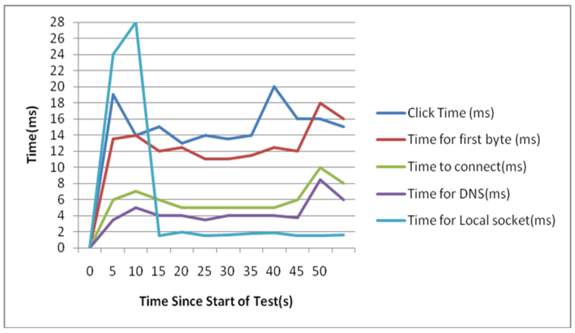

Figure 1. Protocol time for all Url's

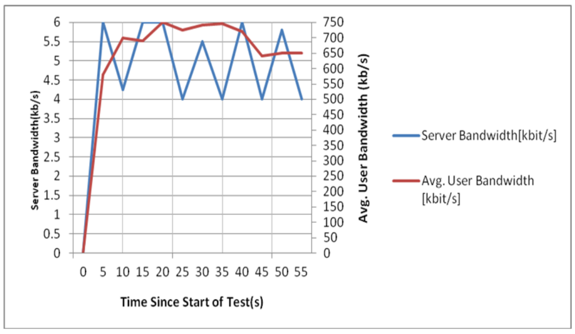

Figure 2. Server and user bandwidth analysis

6. CONCLUSION

SOA based on sound principles of loose coupling, agility, dynamicity, and platform-independent, standards-based, etc has emerged as a promising technology meeting the current industry needs. It aims at developing solutions for enterprise applications. It works on the basic principle of services. The current work stresses the need for a mathematical foundation for capturing SOA concepts. It extends the definition of SOA based on the mathematical framework and general theory systems as proposed by Yanchuk et.al and Ravi Kudermetov et. al. This formalization explored the SOA structure and properties of web services in SOA. It is believed that it will help in better designing and getting the best of SOA applications. Based on the set theory approach, SOA with the help of web services is discussed in three main perspectives: consumer, provider, and registry. In the future it is planned to experiment with the following goals: Defining the services of an application based on the above discussion; Observing the behavior of the system while integrating different services of the application; Refinement of the model if requirement. The findings of this work will be extended in further study to implement SOA based systems and look for improvements with respect to functionality and testing.

REFERENCES

[1] R. Heckel and M. Lohmann, "Towards contract based testing of web services," *Theoretical Computer Science*, vol, 116, pp. 145-156, 2005.
[2] H.K. Gustavo Alonso, Fabio Casati and V. Machiraju, "Web Services Concept, Architectures and Applications," *Springer*, pp. 123-149, 2004.
[3] Aliaksei Yanchuk, Alexander Ivanyu kovich, Maurizio Marchese, "Towards a Mathematical Foundation for Service Oriented Applications Design," *Journal of Software*, vol. 1, no. 1, pp. 32-38, Jul. 2006.
[4] M. P. Papazoglou and J. Yang, "Design methodology for web services and business processes," in *TES '02: Proceed-ings of the Third International Workshop on Technologies for E-Services*. London, UK: Springer Verlag, pp. 54-64, 2002.

[5] R. Dijkman and M. Dumas, “Service-oriented design: A multi-viewpoint approach,” *International Journal on Cooperative Information Systems*, vol. 13, no. 14, pp. 338-378, Dec. 2004.

[6] Quartel, R. Dijkman, and M. Sinderen, “Methodological support for service-oriented design with isdl,” in *ICSOC ’04: Proceedings of the 2nd international conference on Service oriented computing*. New York, NY, USA: ACM Press, pp. 1-10, 2004.

[7] Ravil Kudermetov, Olga Polska, “Towards a Formalization of the Fundamental Concepts of SOA,” *TCSET’*, LvivSlavske, Ukraine, pp. 492-494, 2016.

[8] Burbeck S., “The tao of e-business services: the evolution of Web applications into service-oriented components with Web services,” *IBM DeveloperWorks*, 2000.

[9] Krafzig, D., Banke, K., Slama D., “Enterprise SOA: Service Oriented Architecture Best Practices,” *PrenticeHall*, Englewood Cliffs, 2005.

[10] M.P. Papazoglou, “Service-oriented computing: Concepts, characteristics and directions,” in *WISE ’03: Proceedings of the Fourth International Conference on Web Information Systems Engineering*. Washing-ton, DC, USA: IEEE Computer Society, pp. 3-12, 2003.

[11] D.K. Barry, “Web Services and Service-Oriented Architecture: The Savvy Manager’s Guide,” *Morgan Kaufmann Publishers*, 2003.

[12] Groves, “Successfully planning for SOA,” *BEA Systems Worldwide*, Sep. 2005.

[13] Hohpe, “Stairway to Heaven,” *Software Development*, May 2002.

[14] Chindove, H., Seymour, L., and Merwe, F., “Service-oriented Architecture: Describing Benefits from an Organizational and Enterprise Architecture Perspective,” *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017)*, vol. 3, pp. 483-492, 2017.

[15] “Reference Model for Service Oriented Architecture 1.0,” *OASIS Standard*, Oct. 2006.

[16] “OMG: SOA Modeling Language Specification for the UML Profile and Meta model for Services,” Doc.No.ad/ 2008-11-01, OMG, Nov. 2008.

[17] “SOA Reference Architecture, Technical Standard,” *Open Group*, Nov. 2011.

[18] M.D. Mesarovic, Y. Takahara, “General systems theory: mathematical foundations,” New York: Academic, 1975.

[19] M. Broy, “Can practitioners neglect theory & theoreticians neglect practice?,” *IEEE Computer*, vol. 44, pp. 19-24, 2011.

[20] P. Johnson, M. Ekstedt, and I. Jacobson, “Where’s the theory for software engineering?” *IEEE software*, vol. 29, no. 5, pp. 94-96, 2012.

[21] B. Rumpe and C. Klein, “Automata describing object behavior,” In: *Specification of Behavioral Semantics in Object-Oriented Information Modeling,* H. Kilov W. Harvey (ed.), Kluwer Academic Publishers, pp. 265-286, 1996.

[22] Craigen, S. Gerhart, and R. T.J., “An international survey of industrial applications of formal methods,” *National Technical Information Service*, Springfield, VA, USA, Tech., pp. 1-5, 1993.

[23] M. Broy, “Towards a mathematical concept of a component and its use,” *Software Concepts and Tools*, vol 18, no. 3, pp. 1-23, 1997.

[24] M. Broy, “Compositional refinement of interactive system modeled by relation,” *Lecture Notes Computer Science*, vol. 1536, no. 3, pp. 130-149, 1997.

[25] D. Abowd, R. Allen, and D. Garlan, “Formalizing style to understand descriptions of software architecture,” *ACM Transactions on Software Engineering and Methodology*, vol. 4, no. 4, pp. 319-364, 1995.

[26] Malkis, Alexander and Marmsoler, “A Model of Service Oriented Architectures,” *SBCARS*, pp. 110-119, 2015.

[27] Yanchuk, Aliaksei, Alexander Ivanyukovich, and Maurizio Marchese. "A lightweight formal framework for service-oriented applications design,” *International Conference on Service-Oriented Computing. Springer*, Berlin, Heidelberg, pp. 545-551, 2005.

[28] Kenneth Laskey, *et al.*, “Reference Architecture Foundation for Service Oriented Architecture Version 1.0,” *OASIS Committee Specification 01*, Dec. 2012.

[29] “Business process execution language for web services,” *IBM*, [Online]. Available: http://www-106.ibm.com/developerworks/ library/wsbpel.

[30] “Ws-coordination specifications,” *IBM*, [Online]. Available: http:// www.ibm.com/developer works/library.

[31] J. W. P. Sandeep Chatterjee, “Developing Enterprise Web Services: An Architect Guide,” *Prentice Hall PTR*, 2003.

[32] Gurunathan, Pradeep, and Seethalakshmi Pandian, “Design and Composition of e-Learning and Research Resources using Service Oriented Architecture,” *International Journal of Computer Applications*, pp. 19-24, 2010.