Future Generation Computer Systems 80 (2018) 211–214

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Editorial

Service-Oriented System Engineering

Nik Bessis [a], Xiaojun Zhai [b], Stelios Sotiriadis [c]

[a] Department of Computer Science, Edge Hill University, Ormskirk, Lancashire, UK
[b] Department of Electronics, Computing and Mathematics, University of Derby, Kedleston Road, Derby, DE22 1GB, UK
[c] Edward Rogers Sr.Department of Electrical and Computer Engineering, University of Toronto, Canada

ARTICLE INFO

Keywords:
Service-Oriented System Engineering (SOSE)
Service-Oriented Computing
Cloud computing

ABSTRACT

Service-Oriented System Engineering (SOSE) is one of the emerging research areas that involves a number of research challenges in engineering service-oriented systems, the architecture and computing paradigm as well as the development and management of service-oriented systems. Service-Oriented Computing (SOC) exploits services as the fundamental elements for developing computer-based systems. It has been applied to various areas and promotes fundamental changes to system architecture, especially changing the way software systems are being analyzed, architected, designed, implemented, tested, evaluated, delivered, consumed, maintained and evolved. The innovations of SOC also offer many interesting avenues of research for scientific and industrial communities. In this paper, we present the concepts of the SOSE from the related work. The motivation, opportunities and challenges of the SOSE is highlighted thereafter. In addition to this, a brief overview of accepted papers in our Special Issue on SOSE is presented. Finally we highlight and summarize this paper.

1. Introduction

In the past a few years, Service-Oriented Computing (SOC) has rapidly developed alone with the cloud computing, which is a new paradigm that splits the developers into three independent roles: application builders, the service brokers, and the service developers [1]. As result of this, the developer no longer needs to make the executable that meets the requirements translated from the task specification, and the application development is done via discovery and composition rather than traditional design and coding. In other words, the application is completed through a collaborative environment based on the three independent roles: application builders, service developers, and service brokers [2]. Currently, SOC has been applied to many areas, for example, electronic business, cloud computing, Internet of Things (IoT), Mobile-Edge Computing [3–5].

The remainder of the paper is organized as follows. We summarize the motivation, opportunities and challenges of Service-Oriented System Engineering (SOSE) in Section 2. Section 3 presents an overview of the solutions proposed in accepted papers of this Special Issue on SOSE. Finally, we provide the conclusion in Section 4.

2. Motivation, opportunities and challenges of SOSE

The SOC mainly involves developing Service-Oriented Architectures (SOAs) and corresponding middleware that enables to combine a number of interoperable services to virtually support any business process regardless of the psychical constraint or user context [6]. As result of this, SOA could offer an environment that could be used to smoothly run distributed applications simpler with reasonable cost. Particularly, the coarse-grained servicers approach allow the SOA provides the flexibility and capability for the specific business requirements.

Grid services and Web services are the most common service for implementing SOC, where Grid services provide the foundation of the distributed systems to support the processing of very large data sets. In contrast to Grid services, Web services provide relatively small service access over the network. Therefore, Web Services have become the preferred implementation technology for realizing SOAs.

3. A brief review of accepted articles of this special issue

In this special issue, a number of papers have been accepted in the domain of SOSE, which includes:

3.1. Software defined networks

H. Zhong et al. [7] proposed a load balancing scheme based on server response times by using the advantage of SDN flexibility,

E-mail addresses: nik.bessis@edgehill.ac.uk (N. Bessis), x.zhai@derby.ac.uk (X. Zhai), s.sotiriadis@utoronto.ca (S. Sotiriadis).

https://doi.org/10.1016/j.future.2017.11.025
0167-739X/

named LBBSRT (Load Balancing by Server Response Time). They process user requests by obtaining an evenly balanced server loads using the real-time response time of each server measured to balance the load. Simulation experiments show that their scheme exhibits a better load balancing effect and process requests with a minimum average server response times. In addition, their scheme is easy to implement, and exhibits good scalability and low cost characteristics. In comparison to the traditional load balancing schemes, the proposed scheme can be applied to existing types of servers for a better load balancing effect. The OpenFlow environment is used in the design, where the controller is used to obtain the real-time response times of each server in order to determine the minimum or most stable response time for the chosen server.

3.2. QoS

L. Barakat et al. [8] proposed a novel adaptive execution approach, which efficiently handles service changes occurring at execution time, for both repair and optimisation purposes. The adaptation is performed as soon as possible and in parallel with the execution process, thus reducing interruption time, increasing the chance of a successful recovery, and producing the most optimal solution according to the current environment state. The effectiveness of the proposed approach is demonstrated both analytically and empirically through a case study evaluation applied in the framework of learning object composition. In particular, the results show that, even with frequent changes (e.g. 20 changes per service execution), or in the cases where interference with execution is non preventable (e.g., when an executed service delivers unanticipated quality values), their approach manages to recover from the situation with minimal interruption.

V. Gabrel et al. [9] provided a unified understanding of the wide variety of existing approaches and the theoretical complexity induced by each QoS criterion were analysed and compared. They stated that optimal solution for execution time or throughput QoS criteria can be determined in polynomial time but optimality is no more guaranteed in polynomial time for QoS criteria like cost or reliability. They also showed that the composition problem became NP-hard when optimizing such QoS criteria. Secondly, they proposed a novel approach for solving more efficiently polynomial cases. This approach is based on a scheduling formulation with AND/OR constraints, using a directed graph structure. Based on the experimental test using the Web Service Challenge-09 benchmark, the proposed algorithm outperforms the related work.

3.3. Modeling and automation

M. Geiger et al. [10] proposed a review of the state of support and implementation for Business Processing Model and Notation (BPMN) 2.0. The BPMN has been hailed as a major step in business process modelling and automation, and the vendors of business process management systems (BPMS) is going to switch to the new standard and support the execution in the process engines. Authors presented a detailed analysis of the current state and evolution of BPMN 2.0 support and implementation. They have found only three out of 47 BPMS considered support the execution format defined in the standard, although all of them claim to comply with the BPMN 2.0 standard. Furthermore, there are three process engines: BPM, jBPM and activiti have been evaluated and their degree of support over a period of more than three years has been assessed in the paper as well. The results of this evaluation have provided the first hand evidences that the areas of the standard considered by the implementers, which may conclude that the features that are not available by now will be implemented in the future.

3.4. Wireless Sensor Networks (WSNs)

Z. Zhou et al. [11] proposed a service-oriented wireless sensor networks (WSNs) framework that is used to work in a collective fashion for achieving complex application with the wide-adoption of the Internet of Things as well as heterogeneous smart things. In this work, the functional integration of neighbouring sensor nodes was achieved in a matter of the cooperation between sensor nodes, in which the sensor nodes were encapsulated and represented as energy-aware WSN services. WSN services are categorized into a number of service classes based on their functionalities. As results of this, the service class chains were generated with respect to the requirement of domain applications, and the composition of WSN services was constructed through discovering and selecting appropriate WSN services as the instantiation of service classes contained in chains. This WSN services composition was reduced to a multi-objective and multi-constrained optimization problem, where the particle swarm optimization (PSO) algorithm and genetic algorithm (GA) were adapted to resolve this problem. Experiment results show that PSO outperforms GA in finding approximately optimal WSN services compositions.

Y. Sahni et al. [12] proposed a middleware for WSN-based structural health monitoring (SHM) application using the SOA. The main reason for this proposal solution is that the SHM domain experts lack the knowledge of low-level network issues and cannot develop an application with high efficiency, and the proposed middleware will provide programming abstractions for an application developer that makes it easier to develop an application. In addition, the proposed middleware use SOA to increase the flexibility for developing different applications. The proposed middleware consists of a three-layered middleware with each of the layers containing various services that address issues such as in-network processing, fault tolerance, dynamicity, quality of service etc. They described operations of various services and use two application examples to illustrate the usability and flexibility of the proposed middleware for SHM, and a comparison with other SOA-based WSN middleware is also presented in this paper.

3.5. Cloud service

J. Li et al. [13] proposed an intelligent algorithm called Eagle+ to update the matching Automaton and Table to avoid re-computing the whole patterns after each update. In Eagle+, they only compute the latest update set of patterns to update the Automaton and Table. Moreover, Eagle+ achieves accurately local updating based on three atomic operations, adding, updating and deleting, each of which modifies values on classical Aho-Corasick (AC) automaton, Set Backward Oracle Matching (SBOM) automaton and Wu-Manber (WM) table with an incremental approach. Compared with existing pattern updating methods, Eagle+ reduces the computation complexity from $O(n^2)$ to $O(n)$. The experimental results show that Eagle+ can save nearly 72%–92% of the time consumption in updating automatons and perform 100 times faster in WM table.

3.6. Wireless rechargeable networks

F. Chen et al. [14] proposed a service based mobile charging network that could be used to reduce the charging completion time for wireless rechargeable networks. In order to reduce the charging completion time, the authors proposed that the charger charges network nodes while moving, which would significantly reduce overlooked while waiting for the charger to move to specific spot and then starts to charge the nodes nearby. However, the major challenge to exploit the charging opportunity is the setting of the moving speed of the charger. In order to balance the charging delay and the moving delay, they first formulated the problem of