Exercise for the Lecture

Cloud Computing

Summer Semester 2023

Handout 1

Please send the answers to: **mauladi.mauladi@gwdg.de** .

Note: A submission of your results is not necessary for this exercise sheet. However it is recommended to prepare the questions for the discussion during the exercise.

1 Introduction to Service Computing

Task 1.1 Virtualization and Cloud Computing

Provide short answers to the following questions.

(**Helpful reference:** *https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf*)

(a) What is virtualization?

(b) How is the virtualization related to cloud computing?

(c) Describe four different deployment models in cloud computing?

(d) Describe three different service models in cloud computing?

(e) In the following scenarios, Which companies should use cloud computing, and which service models should meet their needs?

- A startup company initiates their business by creating a mobile application. They require an adaptive database resource that can accommodate the increasing number of users and also consider a very low cost when the user base is still small.
- A university will require various operating systems and diverse resource needs such as memory and CPU to support research needs and a variety of applications and databases.
- A company has 5 TB archived files in a local storage system. They need an application to analyze the files and extract some required information.

Task 1.2 Service-Oriented Computing and Cloud Computing

Service-oriented computing and cloud computing represent two different paradigms that can be combined for mutual benefits.

Paper [1] Wei et al.

Wei Y.; Blake, M.B., "Service-Oriented Computing and Cloud Computing: Challenges and Opportunities," Internet Computing, IEEE , vol.14, no.6, pp.72,75, Nov.-Dec. 2010 doi: 10.1109/MIC.2010.147

By reading the paper please answer following questions:

(a) How a service is different from traditional software (two arguments)?

(b) What are the compound challenges between service-oriented computing and cloud computing (three challenges)?

Task 1.3 Service-Oriented Computing (SOC): The Research Challenges

Service-oriented computing promotes the idea of assembling application components into a network of services that can be loosely coupled to create flexible, dynamic business processes and agile applications that span organizations and computing platforms. This manifesto identifies future of the special issue topics in Service-Oriented Computing.

Paper [2] Yangui et al.

Sami Yangui, Andrzej Goscinski, Khalil Drira, Zahir Tari, Djamal Benslimane, Future generation of service-oriented computing systems, Future Generation Computer Systems, Volume 118, 2021, Pages 252-256, ISSN 0167-739X, https://doi.org/10.1016/j.future.2021.01.019. (https://www.sciencedirect.com/science/article/pii/S0167739X21000297)

By reading the paper please provide a brief answer with at least five examples of new issue topics in SOC

Service-Oriented Computing and Cloud Computing

Challenges and Opportunities

Yi Wei and M. Brian Blake • *University of Notre Dame*

Service-oriented computing and cloud computing have a reciprocal relationship — one provides the computing of services, and the other provides the services of computing. Although service-oriented computing in cloud computing environments presents a new set of research challenges, the authors believe the combination also provides potentially transformative opportunities.

Today, many organizations strive to cope with rapid market changes, such as evolving customer requirements and new business processes. One of the latest challenges is how to work with service-oriented computing (SOC) in a cloud computing environment. Here, we consider how to combine these two paradigms in mutually beneficial ways.

Service-Oriented and Cloud Computing

Competitiveness requires that companies continually modify their IT systems by adding new features or deleting old ones in a relatively short period of time. Traditional software lifecycle models haven't explicitly addressed this requirement for continuous integration of new capabilities.[1] SOC aims to use services as basic blocks to construct rapid, low-cost – yet secure and reliable – applications.[2] It reduces the need to develop new software components each time a new business process arises.

A service is different from a traditional software artifact in that it's autonomous, self-described, reusable, and highly portable. Services range from doing simple arithmetic calculations to executing complicated programs in distributed environments. By using standard description languages, such as the Web Service Description Language (WSDL), a service can expose its interface to the outside world for service discovery and, either by SOAP or Representational State Transfer (REST) protocols, be invoked separately or as a composition of multiple services. The advantages of this new computing paradigm are visible: companies and organizations can develop massively distributed software systems by assembling basic services dynamically.[3] These services may come from different service providers and use markup language techniques, such as XML, to exchange program information and data.

Five decades ago, in 1961, computing pioneer John McCarthy predicted that "computation may someday be organized as a public utility."[4] Cloud computing is that realization, as the paradigm facilitates the delivery of computing-on-demand much like other public utilities, such as electricity and gas. However, cloud computing isn't a new concept. Other computing paradigms – utility computing, grid computing, and on-demand computing – precede cloud computing by addressing the problems of organizing computational power as a publicly available and easily accessible resource.[4,5] Effectively, cloud computing wraps traditional distributed computing or grid computing paradigms with a dynamically scaled business model that mitigates the risk of service over- or underprovisioning for service providers by offering shared computational power that can be accessed as needed.

In contrast to grid computing, where traditionally a user needed to first share some resources before he or she could be granted access to a larger pool of shared resources, a cloud computing user need only pay for the

 1089-7801/10/$26.00

computing services. With cloud computing, new Internet services can be developed and deployed without capital acquisitions of hardware or large human integration expenses. Amazon launched its cloud offering back in 2006, known as the Elastic Computing Cloud or EC2 (see http://aws.amazon.com/ec2). Other companies, such as Google's App Engine (http://code.google.com/appengine) and Microsoft's Azure Platform (www.microsoft.com/windowsazure), released their cloud platforms later in 2008.

Open source cloud computing infrastructure systems have also been developed from university research groups, such as Eucalyptus.[6] These cloud computing offerings provide different levels of abstraction and services to cloud users. Cloud computing environments offer three major types of services: infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS).[7] For example, by leveraging IaaS, Amazon EC2 provides a computing unit that looks much like physical hardware. Users can control the entire software stack. On the other hand, while leveraging SaaS, Google App Engine requires strict restrictions on application architectures as it attempts to improve scalability and performance.

Editor's Note

Modularity and interoperability continue to emerge as the premises for reuse and efficiency. Packaging capabilities, whether human- or software-based, into modular software applications or services is the key to enhanced interorganizational or consumer-to-consumer interoperability.

In evolving their IT infrastructures, industry and government organizations strive to create this environment or service-oriented architecture. The goal of Web-scale workflow and the impetus for this column has been toward a large-scale, complex Web of shared services that, as a sum, creates new higher-level, perhaps unanticipated, capabilities. Researchers have responded to these challenges with the emergence of service-oriented computing in the past decade. Semantic processing of service-based interfaces, service discovery, service composition and consumption, and quality of service are the current leading research topics.

What happens when computing itself is the service? Essentially, this is what cloud computing provides: applications, platforms, network capabilities, and storage, all as services. Over time, it will be interesting to see how cloud-based services will enhance sharing and thus improve this web of possibilities. — *M. Brian Blake*

Compounded Challenges

SOC and cloud computing have many open issues. We wonder if the reciprocal definitions of these two paradigms also suggest that the challenges of one might serve as an opportunity for the other. Figure 1 illustrates the challenges unique to SOC and cloud computing in addition to challenges and opportunities gained when you combine the paradigms.

Maintaining High Service Availability

Often, SOC-based systems require that their underlying services maintain a high availability level. However, this requirement becomes more difficult to fulfill in cloud computing environments because services now reside on one cloud computing provider's infrastructure. Consequently, the availability and responsiveness of these services has a dependency not only on the service provider, but also on the cloud provider. Failures because of outages, network problems, and human error are compounded. An intuitive solution is to deploy services across multiple cloud computing providers to increase availability and provide redundancy. Unfortunately, current cloud computing interfaces tend to be proprietary and aren't explicitly designed for cross-platform interoperability. Consequently, consumers can't easily migrate their deployed services and programs from one cloud computing provider to another as a countermeasure against failures. In the near term, this might challenge widespread adoption of cloud computing.

Providing End-to-End Secure Solutions

Security concerns exist for both SOC and cloud computing. In SOC, services underlying a composite service may originate from different providers across multiple organizations. It's difficult to ensure an end-to-end security solution, as it would require all service providers to guarantee the same level of security guarantee. This is compounded when services reside in cloud computing environments. The underlying infrastructure of the service providers reside with other third-party providers, and as such, there are significant negotiations required between end users, cloud service consumers, and cloud providers to define a certain level of security.

A significant aspect of SOC is message passing. In realizing a workflow of services, messages are passed between services or between services and the service container. An issue related to security is the threat to the privacy of proprietary information. Not only is there a risk that messages could be intercepted, but there's also the threat that competitors might be able to infer business operations from message traffic. Cloud environments suffer the same problems with privacy.[8] In addition to message transfer, cloud consumers also have the concern that their stored information could be compromised or used inappropriately.

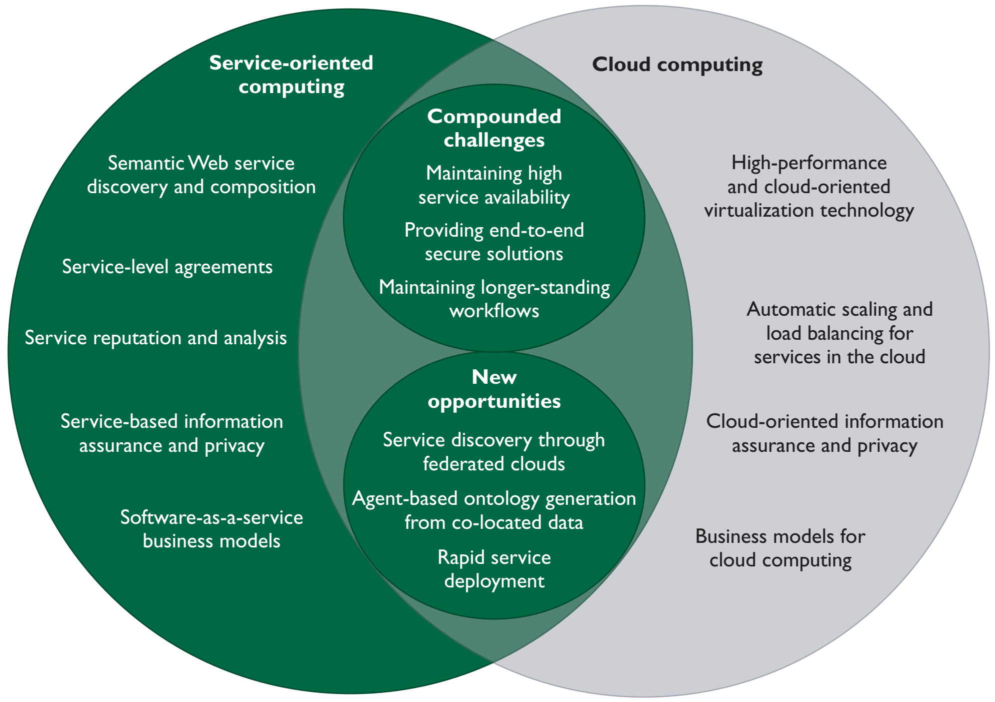

Figure 1. Overlapping service-oriented and cloud computing issues. The challenges of one paradigm might serve as an opportunity for the other.

Managing Longer-Standing Service Workflows

SOC requires the management of loosely coupled services to maintain its working condition. A single service might be integrated into many complex applications and, as such, failure of one service can cause negative effects to numerous interdependent applications. Furthermore, each service within a workflow could reside with unique service providers. This is a challenge to service discovery because current service repositories are decentralized and not well advertised. Consequently, most operational service workflows either reside within one enterprise, or service providers are required to constantly monitor the health of their underlying services so that applications built from these services can maintain a certain level of performance.

In a cloud computing environment, the challenge of service management and monitoring is extended. Current cloud computing providers don't offer user-customized management and monitoring mechanisms built into their infrastructure. Hence, it's still the service developer's responsibility to provide programs and utilities to manage and monitor services.

Transformative Opportunities through Reciprocation

Now that we've looked at some of the challenges, what are some of the ways we could combine each paradigm's strengths to help neutralize the other's weaknesses? We can see at least three opportunities.

Service Discovery through Federated Clouds

In practice, the notion of automated service discovery and composition has been a major challenge. Techniques and technologies for openly discovering services on the Internet are limited. The UDDI standard hasn't been widely adopted or deployed. However, cloud computing might offer a significant opportunity. With next-generation services being deployed in cloud environments, these services effectively become more centralized. The database management community uses the term *federate* to connect multiple distributed databases. Federated databases[9] facilitate the query of information across multiple distributed data stores. Similarly, federated clouds may facilitate the ability to openly discover the services residing within them.

To realize this opportunity, the cloud community must incorporate service-based information similar to the type of information captured in UDDI directory services. Then researchers must investigate new approaches to cross-cloud connectivity. These technologies require specific protocols so that service information can be shared among cloud providers anonymously.

Rapid Service Deployment

SOC promotes the tenet that Web services can be accessed from anywhere and combined with other services to create higher-level workflows. However, merging and new organizations might need to relocate existing services to new IT environments. Portability, in these regards, is a significant challenge for SOC; cloud computing environments tend to mitigate these issues. In the future, standard cloud APIs will let service providers deploy their services seamlessly to multiple cloud computing providers. We believe that the demand for cross-cloud service deployment will increase as cloud environments are more widely adopted. Furthermore, cloud computing providers should add features to their cloud infrastructures to enable management and monitoring for deployed services.[10] These management and monitoring functionalities should not only consider the status of deployed services but also take into account the status of underlying cloud infrastructures. Service-level agreements in future integrated service and cloud systems will operate with more accuracy and confidence.

Agent-Mediated Ontology Generation from Co-Located Information

The management of service metadata is another challenge. Inference of machine-interpretable information about what the service can do and what it can provide remains an open issue. Syntactic interpretation of service-based information lacks the confidence to perform this function well because the meaning of underlying information is missing. Semantic approaches that allow meaningful definitions of information are limited to the users' willingness to construct effective ontological metadata. This issue is exacerbated, considering the fact that services are delivered by different providers across multiple organizations from different business domains. Cloud environments can offer solutions for many service providers who may reside within the same infrastructure. While it doesn't offer a strategic business advantage for organizations to agree on linked ontologies, third-party software agents operating within a cloud might be able to derive ontological information from the stored data and operations.

Service-oriented and cloud computing combined will indeed begin to challenge the way in which we think about enterprise computing. However, the potential for sharing could not only remove historical barriers but also encourage organizations to think more collaboratively.

References

1. M.B. Blake, "Decomposing Composition: Service-Oriented Software Engineers," *IEEE Software*, vol. 24, no. 6, 2007, pp. 68–77.
2. M.P. Papazoglou et al., "Service-Oriented Computing: A Research Roadmap," *Int'l J. Cooperative Information Systems*, vol. 17, no. 2, 2008, pp. 223–255.
3. X. Su and J. Rao, "A Survey of Automated Web Service Composition Methods," *Semantic Web Services and Web Process Composition*, LNCS 3387, Springer, 2005, pp. 43–54.
4. I. Foster et al., "Cloud Computing and Grid Computing 360-Degree Compared," *Proc. IEEE Grid Computing Environments Workshop*, IEEE Press, 2008, pp. 1–10.
5. M. Armburst et al., "Above the Clouds: A Berkeley View of Cloud Computing," tech. report UCB/EECS-2009-28, Electrical Eng. and Computer Science Dept., Univ. of California, Berkeley, 2009.
6. D. Nurmi et al., "The Eucalyptus Open-Source Cloud-Computing System," *Proc. 9th IEEE/ACM Int'l Symp. Cluster Computing and the Grid*, IEEE CS Press, 2009, pp. 124–131.
7. G. Rains, "Cloud Computing and SOA," MITRE, white paper, Oct. 2009; www.mitre.org/work/tech_papers/tech_papers_09/09_0743/09_0743.pdf.
8. S. Pearson, "Taking Account of Privacy When Designing Cloud Computing Services," *Proc. Int'l Conf. Software Eng.* (ICSE) *Workshop on Software Eng. Challenges of Cloud Computing*, IEEE CS Press, 2009, pp. 44–52.
9. A.P. Sheth and J.A. Larson, "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," *ACM Computing Surveys*, vol. 22, no. 3, 1990, pp. 183–236.
10. R. Nathuji, A. Kansal, and A. Ghaffarkhah, "Q-Clouds: Managing Performance Interference Effects for QoS-Aware Clouds," *Proc. 5th European Conf. Computer Systems*, ACM Press, 2010, pp. 237–250.

Yi Wei is a doctoral student in the Department of Computer Science and Engineering at the University of Notre Dame. His research interests include service-oriented computing and cloud computing. Wei has a MS in computer science from the University of Delaware. Contact him at ywei1@nd.edu.

M. Brian Blake is a professor in the Department of Computer Science and the Associate Dean for Research in the College of Engineering at the University of Notre Dame. He conducts applied research in the areas of service-oriented computing, software engineering, and enterprise integration. Blake has a PhD in information and software engineering from George Mason University. He's a senior member of IEEE. Contact him at M.Brian.Blake@nd.edu.

Future Generation Computer Systems 118 (2021) 252–256

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Editorial

Future generation of service-oriented computing systems

Sami Yangui [a,*], Andrzej Goscinski [b], Khalil Drira [c], Zahir Tari [d], Djamal Benslimane [e]

[a] LAAS-CNRS, Université de Toulouse, INSA, 31400 Toulouse, France
[b] Deakin University, Geelong, Australia
[c] LAAS-CNRS, Université de Toulouse, 31400 Toulouse, France
[d] School of Computing Technologies, RMIT University, Sydney, Australia
[e] LIRIS, Université de Lyon 1, France

ARTICLE INFO

Article history:
Available online 19 January 2021

Keywords:
Service computing
Cloud computing
Fog computing
Business Process Management

ABSTRACT

Service-Oriented Computing (SOC) and SOC systems have been invented, studied, and developed in response to the problems generated by heterogeneity and poor latency, and a need for modularization and standardization. Wide application of SOC, practically in all areas of human endeavors, interconnectivity provided by wireless networks, huge volumes and diversity of data, increased frequency of security attacks, and reliability expectations have increased research, development, and deployment of SOC systems. These circumstances and requirements could only be dealt with by a future generation of SOC systems. This implies a need for fast and high-quality research and wide and efficient dissemination of achieved innovative results. Therefore, a Special Issue (SI) was proposed, and its relevant scope defined. This SI involves a set of high-quality and cross-community scientific papers from various disciplines (e.g., cloud computing, edge computing, business process management). The accepted papers focus on research leading toward future generation of service-oriented computing systems (e.g., dynamic quality of service management in fog computing, time-aware resources allocation in cloud computing, resilient composition in the Internet of Things) with emphasis on advanced results that solve open research problems and have significant impact on the field of service-oriented computing.

1. Introduction

Distributed Computing has arrived when two processes executing on two different platforms started to communicate using Message Passing or Remote Procedure Calls. A need for innovation followed the initial major step in Computing by problems of heterogeneity and low level of communication abstraction, that of the Internet. An initial response was provided by service-oriented architecture, which basic framework exploits the concept of a service, an abstraction that could be invoked to provide a requested computation, storage, and return or pass on a result to another service or even human being. Shortly after, Service-Oriented Computing (SOC) has started to be a subject of intensive research, development, and deployment. Since more and more private users, research institutions, businesses, hospitals, cities, and industry companies wish to benefit from interconnectivity provided by wireless networks, the interest in services study and development is exponentially growing. These days, SOC is in use in all areas of human endeavor, which are heterogeneous, deal with huge data volumes, and subject to security attacks. SOC systems must satisfy high-performance, reliability, and trustworthiness requirements of users. From the first concepts of services that are linked to the abstractions of infrastructure, platforms, and software, we have moved to services created for and exploited in networks, data storage and management, clouds, fogs, edges, huge data, security, trust, workflow, modeling, etc. This move implies a need for new knowledge and enhancement of the existing knowledge of services cloud, enhancing existing knowledge of in the area of services, exploiting new knowledge created for dealing with wireless networks, communication between clouds and edge devices, using Artificial Intelligence (AI) to deal with huge data, and generate new applications that would benefit other researchers, developers of service computing systems, and create a basis for other users to consider making their areas of interest better, faster, more secure, and trustworthy.

Thus, the question is: "what is Future Generation of SOC Systems?". To get even a partial answer to this question this special issue was seeking outstanding, original contributions, including theoretical and empirical evaluations, as well as, practical and industrial experiences, with emphasis on advanced results that solve open research problems and have significant impact in the field of SOC.

* Corresponding author.
E-mail address: yangui@laas.fr (S. Yangui).

https://doi.org/10.1016/j.future.2021.01.019
0167-739X/© 2021 Published by Elsevier B.V.

2. Special issue topics of interest

Our study shows that there are some areas that define the research leading toward Future Generation of SOC Systems. They are as follows:

1. Service engineering and management, including
    - legacy systems migration and modernization
    - service design, specification, discovery, customization, composition, and deployment
    - service innovation, governance, and change and workload management
    - theoretical foundations of service engineering
    - service execution, monitoring and reconfiguration
    - service engineering and management, services and data, services in the cloud and on the edge, services for the internet of things, services for software network functions and software defined networks, services in organizations, business, and society
    - quality of service, security, privacy, and trust
    - architectures for multi-host container deployments
    - microservices deployment and management
2. Services and Data
    - services for big data and compute-intensive applications
    - data mining and analytics
    - AI services in data processing
    - data-provisioning services
    - services-related linked open data
    - automated knowledge graph creation
3. Services in the cloud and on the edge
    - migration to virtual infrastructures
    - service deployment and orchestration in the cloud
    - cloud and edge services and workflow management
    - cloud and edge brokers and coordination across multiple resource managers
    - workload transformation
    - edge AI and distributed intelligence services
    - AI, machine learning, neural network, and data analytic techniques in edge processing
    - analytics and knowledge generation services
    - lightweight service deployment and management
    - services and edge gateway architectures
4. Services for the Internet of Things (IoT)
    - embedded and real-time services
    - RFID, sensor data, and services related to IoT
    - services for IoT platforms and applications
    - AI services in IoT
    - secure intelligent IoT-edge systems
    - secure intelligent coordination and networking between edge and cloud
    - service-oriented protocols for IoT applications
    - REST APIs and services for IoT platforms and applications
    - trust and privacy management in intelligent IoT-edge systems
5. Services for softwareized network functions and software defined networks
    - service network function management and orchestration
    - AI services in routing
    - machine-learning algorithms for IoT applications
    - services for novel and emerging networking protocols
    - named data networking
    - virtualized network and transport mechanisms
    - virtualized network functions and services
    - virtualized service function chaining
6. Services in organizations, business, and society
    - services science
    - AI services in organizations, business, and society
    - social networks and services
    - cost and pricing of services
    - service marketplaces and ecosystems
    - service business models
    - enterprise architecture and services
    - quality of service and energy efficiency for intelligent application systems
    - service chatbots

3. Research outcomes of the special issue papers

All papers submitted to the special issue sit in the area defined by the listed topics. The main content and contributions of the selected papers are as follows.

Embedding reactive behavior into artifact-centric business process models:

Business processes are an important piece of service modeling and computing. A business process represents a set of activities and tasks that aims at accomplishing an organizational objective. The process must involve clearly defined inputs and a single output. These inputs are made up of all the factors that contribute (either directly or indirectly) to the added value of a service or product. These factors can be categorized into artifact-centric processes, management processes, operational processes and supporting business processes. Artifact-centric business processes are characterized by the changes and evolution of business data, or business entities. These changes are considered the main driver of the processes. Modeling artifact-centric processes focuses on describing how business data is changed/updated, by a particular action or task, throughout the process. This advocates somehow a sort of middle ground between a conceptual formalization of dynamic systems and their actual implementation.

In paper [1], the authors introduce novel approach to generate repairs for the activities in artifact-centric business process models. The proposed approach generates extensions to the activities to ensure that the integrity constraints in the data model are fulfilled considering a given a data model, a process model, and the specification of the activities in the process. The authors show that the approach supports the evolution of the data model independently from the business process model and the specification of the activities. The approach relies on service modeling languages such as Unified Modeling Languages and Business Process Management Notation (BPMN).

Topic-based crossing-workflow fragment discovery

Workflows enable making business processes more productive and efficient. Business processes allow modeling services compositions to deliver a result. Workflows provide technology or tool that can help to do achieve this result. The discovery of workflow fragments is significant to promote the reuse or repurposing of best practices evidenced in legacy workflows, when novel scientific experiments are to be conducted. The discovery of workflow fragments is closely related to service discovery, which exploits: Web Service Decription Language-based keyword search, semantic matching based on domain knowledge or ontologies, context awareness, or Quality of Service (QoS)-based discovery, and machine learning techniques adopted to examine service relevance. ZhangBing Zhou et al. proposed in paper [2] "a novel crossing workflow fragment discovery mechanism, where an activity knowledge graph is constructed to capture flat invocation relations between activities, and hierarchical parent–child relations specified upon sub-workflows and their corresponding activities". Semantic relevance of activities and sub-workflows is calculated based on their representative topics. Individual candidate activities or sub-workflows are discovered when considering their semantic relevance and short-document descriptions. The fragments are evaluated by balancing their structural and semantic similarities. The authors made four major contributions; (i) demonstrated that their approach is as accurate in discovering appropriate crossing-workflow fragments as the state of art's techniques; (ii) constructed an activity knowledge graph, whose edges capture (a) invocation relations between activities in scientific workflows, and (b) hierarchical parent–child relations specified upon sub-workflows and their corresponding activities; (iii) calculated the semantic relevance of activities and sub-workflows leveraging their representative topics, where these topics are generated by applying the biterm topic model based on the description in short texts for activities and sub-workflows, (iv) designed a novel and efficient crossing-workflow fragments discovery algorithm when the requirement is specified in terms of a workflow fragment.

Understanding complex process models by abstracting infrequent behavior

Process mining has become very popular in the last years to analyze the behavior of an organization by offering techniques to discover, monitor and enhance real processes. A key point in process mining is to discover understandable process models. To achieve this goal in complex processes, several simplification techniques have been proposed, from the structural simplification of the model to the simplification of the log to discover simpler process models. However, obtaining a comprehensible model explaining the behavior of unstructured large processes – for instance containing hundreds of activities – is still an open challenge.

The authors of paper [3] introduced the importance of a behavioral simplification in complex processes to understand what is happening in them. They have presented UBeA, a novel algorithm which, given an event log, a process model, and the events considered as core behavior, abstracts the remaining non-core behavior from the process model. They have also presented IBeA, a specific implementation of this algorithm to simplify process models by abstracting infrequent behavior. IBeA can detect the infrequent behavior which obfuscates a process, abstracting it to produce simpler process models with a trade-off between completeness and precision, allowing to obtain an overall view of the process. The proposal also simplifies the event log, allowing to analyze and enhance the process with other process mining techniques.

The contribution of this paper is related to Business Process Management research field which falls into service computing. Specifically, this work aims to simplify process models by detecting and abstracting infrequent behavior. Such processes are typically used when provisioning applications and services over mobile and highly dynamic environments like cloud and fog. They use business processes and workflow to describe the management procedures that we need to apply during the cloud/fog applications life cycle (migration, SLA management, service orchestration, load balancing and so on). They compared IBeA with other simplification approaches, using 11 logs from real scenarios, showing that IBeA can obtain, or allow to discover, a simpler and better model in complex processes.

Graph-based data caching optimization in edge computing

Currently, data collection, storage and processing are provided in many organizations, business, cities, hospitals, etc. by cloud computing systems. However, since they deal with huge volumes of data, their use is associated with high costs of transmission and major communication latency. Edge computing has emerged as a new computing paradigm that allows computation and storage resources and associated services in the cloud to be distributed to edge devices and servers. Those edge devices and servers are deployed at base stations close to users; they offer high-quality services. One of them is data caching because it ensures low latency for service delivery, minimizes the data caching cost, and maximizes the reduction in service latency. To achieve these outcomes, the authors formulated in paper [4] the Edge Data Caching (EDC) problem as a constrained optimization problem. They proved the NP-completeness of this EDC problem and provided an optimal solution named IPEDC to solve this problem based on Integer Programming. The authors carried out intensive experiments on a real-world data set and a synthesized data set to evaluate representative approaches. They demonstrated that IPEDC and AEDC significantly outperform the three representative baseline approaches. Major contributions of this paper are as follows: (i) the EDC problem from the app vendors' perspective was formulated, and then proved that it is NP-complete; (ii) an optimal approach named IPEDC was developed for finding optimal solutions to EDC problems with the Integer Programming technique; (iii) an approximation approach named AEDC was developed for finding near optimal solutions to EDC problems in large-scale scenarios efficiently, and analyzed its theoretical approximation ratio; (iv) carried out extensive experiments on both a real-world data set and a synthesized data set were carried out to evaluate the proposed approaches against four representative approaches. This research has established the foundation for the EDC problem and opened several research directions.

QoE-aware user allocation in edge computing systems with dynamic QoS

As online services and applications are moving toward a more human-centered design, many app vendors are taking the Quality of Experience (QoE) increasingly seriously. End-to-end latency is a key factor that determines the QoE experienced by users, especially for latency-sensitive applications such as online gaming, autonomous vehicles, critical warning systems and so on. Edge computing has then been introduced as an effort to reduce network latency. In a mobile edge computing system, edge servers are usually deployed at, or near cellular base stations, offering processing power and low network latency to users within their proximity. In paper [5], the authors tackle the Edge User Allocation (EUA) problem from the perspective of an application vendor, who needs to decide which edge servers to be used to serve which users in a specific area. Furthermore, the vendor must consider the various levels of QoS for its users. Each QoS level leads to a different QoE level. Thus, the application vendor also needs to decide the QoS level for each user so that the overall user experience is maximized. The authors first optimally solve this problem using Integer Linear Programming (ILP) technique. Being an NP-hard problem, it is intractable to solve it optimally in large-scale scenarios. Thus, the authors propose a heuristic approach that can effectively and efficiently find sub-optimal solutions to the QoE-aware EUA problem. They conduct a series of experiments on a real-world data set to evaluate the performance of their approach against several state-of-the-art and baseline approaches.

The authors made the following main contributions: (i) formally defined the QoE-aware EUA problem and show that it is an NP-hard problem; (ii) proposed an optimal approach based on ILP for solving this problem exactly; (iii) introduced QoEUA — a new heuristic that performs better than their previous heuristic under resource-scarce circumstances; and (iv) carried out experiments based on a real-world data set to demonstrate the effectiveness and efficiency of QoEUA against several baseline and state-of-the-art approaches.

Lightweight self-organizing distributed monitoring of Fog infrastructures

The authors of paper [6] stated "Monitoring will play an enabling role in the orchestration of next-gen Fog applications. Particularly, monitoring of Fog computing infrastructures should deal with platform heterogeneity, scarce resource availability at the edge, and high dynamicity all along the Cloud-IoT continuum". As a follow up, the authors presented extended, with an automated network overlay restructuring mechanisms, their lightweight, scalable, non-intrusive and robust methodology, and prototype for monitoring Fog computing infrastructures, namely FogMon. FogMon monitors hardware resources at different Fog nodes, end-to-end network QoS between such nodes, and connected IoT devices. Furthermore, FogMon features a self-organizing Peer-to-Peer (P2P) topology with self-restructuring mechanisms, and differential monitoring updates. These extensions ensure scalability, fault tolerance and low communication overhead.

The authors proposed in their paper an extension of their previous methodology and open-source prototype for edge resources monitoring. To that end, they invented and presented a new mechanism for (re-)structuring the P2P overlay network. They also carried out experiments on a real testbed to demonstrate how the footprint of FogMon is limited and how its self-restructuring topology makes it resilient to infrastructure dynamicity.

Toward a correct and optimal time-aware cloud resource allocation to business processes

Cloud computing is an increasingly popular computing paradigm because it provides on-demand services to organizations for deploying their business processes over the Internet that reduces their needs to plan for provisioning resources. Despite providing competitive pricing strategies using them can lead to violating time constraints and exceeding budget constraints due to inappropriate decisions when allocating cloud resources to business processes. The authors of paper [7] present an approach to guarantee a correct (because time constraints on these processes are not violated) and optimal time-aware allocation (because the deployment cost of these processes is minimized) of cloud resources to business processes. The authors' approach uses timed automata to formally verify the matching between business processes' temporal constraints and cloud resources' time availabilities, and linear programming to optimize deployment costs.

The authors presented an approach for allocating cloud resources to the activities of business processes that are subject to time constraints. To achieve verification, they developed rules that transform 840 million BPMN-based business process models into a network of time-automata so that proper matching of activities' needs of resources to cloud resources is ensured despite the time constraints. To achieve optimization that the number of a business processes' activities could impact, the authors developed a linear programming model that considered the deployment cost of these activities over cloud resources. Experiments demonstrate the technical do-ability of the proposed approach. The contributions of this work are (i) the development of a set of rules to transform business processes into timed-automata as a step toward a correct time-aware cloud resource allocation in business processes; (ii) formalization of the optimization problem as a math-45-ematical model to minimize the deployment cost of time-constrained business processes, and (iii) evaluation of the technical do-ability of the developed approach for ensuring the correctness and optimization of time-aware cloud resource allocation to business processes.

Resilient composition of drone services for delivery

In recent years, drones and unmanned aerial vehicles, have proliferated rapidly around the globe in both military and civilian spheres. Drones are used in situations where manned flight is considered too risky, difficult or simply costly. They can be operated by either humans or programs. For instance, drones are increasingly used nowadays for package delivery.

SOC provides powerful mechanisms to abstract the functional and non-functional or QoS properties of a drone in the form of Drone-as-a-Service (DaaS). The functional property of a DaaS describes the delivery of a package from a given source to a destination following a skyway network. The non-functional properties of a DaaS are battery capacity, flight range, payload, and speed.

In paper [8], the authors propose a resilient drone service composition framework for delivery considering the recharging constraints and the dynamic weather conditions. An optimal set of candidate drone services is selected using the skyline approach at the source node in a skyway network. The authors introduced a formal model to represent constraint-aware drone services. They propose a deterministic look ahead algorithm to build an initial offline composition plan. The resilient service composition approach is heuristic-based. It incorporates the intrinsic and extrinsic factors affecting the drone services at runtime. The authors designed, implemented and evaluated through simulation a framework that enables the target compositions. The experimental results show that the proposed framework adapts to the runtime service failures and meets the users' expectations like minimum delivery time and cost. They also compared their obtained results versus Brute-Force and without look ahead approaches. The experiments show that the proposed approach guarantees the resilience of delivery services for the increasing number of failure rates.

Thread-level resource consumption control of tenant custom code in a shared JVM for multi-tenant SaaS

Software as-a-Service (SaaS) is a software distribution model in which a third-party provider hosts applications and makes them available to consumers in cloud setting. SaaS is one of three main categories of cloud computing, alongside Infrastructure as-a-Service and Platform as-a-Service. It is closely related to the application service provider. Prospective consumers can invoke and use SaaS according to the pay-as-you-go business model. SaaS providers commonly support customization of their services to allow them to attract larger tenant bases. The nature of these customizations in practice ranges from anticipated configuration options to sophisticated code extensions. From a SaaS provider viewpoint, the latter category is particularly challenging as it involves executing untrusted tenant custom code in the SaaS production environment. Proper isolation of custom code in turn requires the ability to control resource consumption of each tenant.

The authors of paper [9] present a practical solution for thread-level resource consumption control of tenant provided custom code. This solution provides control mechanism for four types of resources: CPU, memory, network, and storage. Usage data are gathered using the Java Resource Consumption Management (JSR-284) Application Programming Interface (API). In case of CPU and memory, where this API is not capable of imposing limits, both the Java runtime environment bytecode and tenant code are instrumented with usage control checkpoints which ensures that CPU and memory usage of tenants remain within their service-level agreements limits. The authors validate their finding through case studies and experiments. The results show that the proposed solution outperforms containers in terms of tenant accommodation capacity and performance overhead.

4. Research contributions

In response to the specified general needs, a set of high-quality papers from cross-community scientific papers have been submitted by experts from various disciplines, such as business-process management, distributed systems, computer networks, wireless and mobile computing, cloud computing, cyber–physical systems, networking, scientific workflows, services science, services computing, data science, management science, and software engineering. The main content and contributions of these papers are as follows. First, a set of topics and sub-topics that require research leading toward Future Generation of SOC Systems was specified. Second, the special issue accepted papers present outstanding, original contributions, including theoretical and empirical evaluations, as well as practical and industrial experiences, with emphasis on advanced results that solve open research problems and have significant impact on the field of service-oriented computing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The Guest Editors wish to thank all authors for sharing the outcomes of their research, which make this special issue possible. The Editors are grateful to the reviewers for their thoughtful and constructive comments and suggestions. The Guest Editors wish to express their gratitude to Professor Erik Cambria, the Special Issue Editor. The Guest Editors are also grateful to the members of the Journal Editorial Office, in particular Jacqueline Zhu and Mohammed Samiullah who provided excellent support and services.

References

[1] X. Oriol, G.D. Giacomo, M. Estanol, E. Teniente, Embedding reactive behaviour into artifact-centric business process models, 2021, in press.
[2] Z. Zhou, J. Wen, Y. Wang, X. Xue, P. Hung, L. Nguyen, Topic-based crossing-workflow fragment discovery, 2021, in press.
[3] D. Chapela-Campa, M. Mucientes, M. Lama, Understanding complex process models by abstracting infrequent behavior, 2021, in press.
[4] X. Xia, F. Chen, Q. He, G. Cui, P. Lai, M. Razek, J. Grundy, H. Jin, Graph-based data caching optimization in edge computing, 2021, in press.
[5] P. Lai, Q. He, G. Cui, X. Xia, M. Razek, F. Chen, J. Hosking, J. Grundy, Y. Yang, QoE-aware user allocation in edge computing systems with dynamic QoS, 2021, in press.
[6] S. Forti, M. Gaglianese, A. Brogi, Lightweight self-organising distributed monitoring of fog infrastructures, 2021, in press.
[7] R.B. Halima, S. Kallel, W. Gaaloul, Z. Maamar, M. Jmaiel, Toward a correct and optimal time-aware cloud resource allocation to business processes, 2021, in press.
[8] B. Shahzaad, A. Bouguettaya, S. Mistry, A.G. Neiat, Resilient composition of drone services for delivery, 2021, in press.
[9] M. Makki, D.V. Landuyt, B. Lagaisse, W. Joosen, Thread-level resource consumption control of tenant custom code in a shared JVM for multi-tenant SaaS, 2021, in press.

Sami Yangui is an Associate Professor with Institut National des Sciences Appliquées (INSA), Toulouse, France. He is member of the CNRS LAAS research team. His research interests include distributed systems and architectures, service-oriented computing and Internet of Things. He is working on different aspects related to these topics, such as cloud/fog computing, network functions virtualization and content delivery networks. He is involved in different European and International projects, as well as, standardization efforts. He published several scientific papers in high ranked conferences and journals in his field of research. He is IEEE Member and he served on many program and organization committees of International conferences and workshops, as well as, guest editor in several journals such as Elsevier Future Generation Computer Systems journal and IEEE Access.

Andrzej Goscinski is Professor, Deakin University; and Adjunct Professor, RMIT University. He carries out research in cloud, edge, services computing, IoT, and distributed systems and applications. From January 1993 to December 2001, Dr. Goscinski completed tenure as Head of School, Deakin University. Now, he researches abstracting system components as services, publishing them to a broker, and executing them in SaaS clouds. His research results have been published in high quality journals and conference proceedings. Dr. Goscinski serves as Associate Editor of IEEE TSC; Inderscience's IJCC; Springer's FGCS, Guest Editor of Special Issues; and General, Program, and Honorary Chair of IEEE Services, Cloud, and Distributed and Parallel Systems and Applications conferences.

Khalil Drira received the Master degree in computer science from INP, Toulouse, in 1988, and the Ph.D and HDR degrees in computer science from Universite Paul Sabatier Toulouse in 1992 and 2005, respectively. Since 1992, he assumes a full-time research position in CNRS, France. His research interests include cooperative IoT network services, platforms and applications. His research activity addresses topics in this field focusing on Software architectures and communication services. He continues to be involved in national and international conferences and journals. He serves as a member of the program journals in the fields of software architecture as well as IoT and Internet networks.

Zahir Tari is a full professor in Distributed Systems at RMIT University (Australia). He received a bachelor degree in Mathematics from University of Algiers (USTHB, Algeria) in 1984, M.Sc. in Operational Research from University of Grenoble (France) in 1985 and Ph.D. degree in Computer Science from University of Grenoble (France) in 1989. Zahir's expertise is in the areas of system's performance (e.g. P2P, Cloud, IoT) as well as system's security (e.g. SCADA, SmartGrid, Cloud, IoT). He is the co-author of six books (John Wiley, Springer) and he has edited over 25 conference proceedings. Zahir is also a recipient of over 11M$ in funding from ARC (Australian Research Council) and lately part of a successful 7th Framework AU2EU (Australia to European) bid on 'Authorisation and Authentication for Entrusted Unions. Finally, Zahir is an associate editor of ACM Computing Surveys, IEEE Transactions on Computers (TC), IEEE Transactions on Parallel and Distributed Systems (TPDS) and IEEE Magazine on Cloud Computing.

Djamal Benslimane is currently a Full Professor at Lyon 1 University in France. His research interests lie in the areas of Services computing, database interoperability, ontologies, and recommender systems. Some of his works were published in IEEE Internet Computing, IEEE Transactions on Knowledge and Engineering, IEEE Transactions on Services Computing, IEEE Transactions on Systems, Man, and Cybernetics, Communications of the ACM, ACM Transactions on Internet Technology, ACM Transactions on Software Engineering and Methodology, SIGOD Record, and WWW Journal.