IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 24, 2022, accepted March 10, 2022, date of publication March 22, 2022, date of current version March 31, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3161455

Introducing Cloud-Assisted Micro-Service-Based Software Development Framework for Healthcare Systems

JOHN ZAKI[1], S. M. RIAZUL ISLAM[2], (Member, IEEE), NORAH SALEH ALGHAMDI[3], M. ABDULLAH-AL-WADUD[4], (Member, IEEE), AND KYUNG-SUP KWAK[5], (Life Senior Member, IEEE)

[1]Department of Computer and Systems, Faculty of Engineering, Mansoura University, Mansoura 35516, Egypt
[2]Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea
[3]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh 11671, Saudi Arabia
[4]Department of Software Engineering, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia
[5]Department of Information and Communication Engineering, Inha University, Incheon 22212, South Korea

Corresponding authors: Norah Saleh Alghamdi (nosalghamdi@pnu.edu.sa) and Kyung-Sup Kwak (kskwak@inha.ac.kr)

This work was supported in part by National Research Foundation of Korea-Grant funded by the Korean Government (Ministry of Science and ICT-NRF-2020R1A2B5B02002478) and in part by Sejong university through its faculty research program (20212023). The work was also partially supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R40), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

John Zaki and S. M. Riazul Islam contributed equally to this work.

ABSTRACT In healthcare services, application development is considered the most complex and time-consuming phase. As it is difficult to plan and time-intense, it requires high maintenance. Healthcare applications need strict compliance and the scope of application is immense along with associates, classes in services, and classified system. Application designing in healthcare with the help of traditional approaches such as monolithic and service-oriented architecture (SOA) generate problems in different areas like service availability, remote access to services, service provisioning, scalability, healthcare systems integration with each other. That is why there is a need for less sophisticated and user-friendly healthcare systems, which are easy to plan and develop, inexpensive requirement maintenance, and agile testing. To overcome the aforesaid issues in the domain of healthcare application development, this paper develops a framework of micro services for the development of healthcare services using cloud computing infrastructure. Micro-service-based techniques provide lightly coupled and fine-grained methodology. With the use of micro services technique presented in this work, the efficiency, scalability, and performance are improved. In this research, an approach for development and deployment properly in the cloud for healthcare applications is developed. Thus, it contributes to the system design approach and system analysis. Quantitative and qualitative results are reported showing the advantages of micro services approach used.

INDEX TERMS Service oriented architecture, micro services, cloud computing, healthcare, application design, monolith approach, application software.

I. INTRODUCTION

E-Healthcare systems have recently developed a considerable attention for service providers of health care and their consumers. They are classified into different categories such as electronic healthcare records, and personal records. The healthcare records focus especially on information storage and exchange of information. Users of healthcare applications are facilitated by the access to health information systems such as E-records, test report of patient, personal health record banks created by patient, healthcare information broadcast in real time environment with help of monitoring devices, wearable sensors and systems based on the Internet of things technology (IoT) [1]. The usage of healthcare systems has increased greatly due to belief of patients in health data.

The healthcare applications offer a complex set of services including many participants [2]. Each stakeholder has

The associate editor coordinating the review of this manuscript and approving it for publication was Claudio Zunino.

his own objectives. Thus, the context, and domain of the healthcare application become overwhelming. This leads to more complexity in terms of development. Healthcare application development is complicated, hard to plan and administrate, inflated in terms of development and maintenance, and time-consuming compared to the standard development of non-healthcare applications. The service of technical and specialized approaches in architecture style, data processing, database storage formats, and technology are reasons for interoperability issues in healthcare applications. The rampant costs of healthcare services have led the healthcare solution providers to offer healthcare services that can be remotely accessed. The available healthcare systems in use are more costly both for patients in respect of use and providers in terms of use and maintenance. Design and development of healthcare applications in cloud environment is a demanding trend with more focus on scalability of services.

Aksakalli *et al.* [3] developed a microservice architecture for deploying a microservice application. The research stated that monolithic approach is easier to implement on the cloud with different instances deployed behind a load balancer. While microservices are often more complicated particularly with a large number of services which is typical in any practical application. They showed that such an implementation is prone to errors and cumbersome. Then they proposed a systematic approach and tools to enable deployment of microservices to resources with limited capacity. In their model they defined the space and model parameters to be used for automatic deployment configuration.

Khaleq & Ra [4] focused on developing a scaling architecture for microservice applications at container level where the response time is the main quality of service factor to be maintained. In their study, they stated that scaling applications imposes quality of service requirements which demand a customized scaling approach. They showed that the challenge is identifying the right values for the different autoscaling parameters to guarantee the quality of service in a changing dynamic environment. Their approach depended on reinforcement learning (RL) for intelligent autonomous autoscaling in cloud applications with no domain knowledge. The first module of their two module system identifies the microservice resource demand while the second module uses reinforcement learning agents to learn and identify the autoscaling threshold values based on the resource demand. The results showed 20% better response time compared to the default autoscaling in Google Kubernetes.

Vural and Koyuncu [5] discussed the granularity level of microservice and the optimal point for granularity based on coupling and cohesion. They stated that high availability, high scalability and reduced recovery time are among cloud requirements for microservice. They used two case studies and applied domain driven design concept on both cases. They modified the granularity of each case to study the effect of different modularity levels. Then compared the modified cases with more and less granularity to the original examples. They showed that domain driven design delivered better end results for finding the optimal granularity of the microservices.

Calderón-Gómez *et al.* [6] developed and implemented a microservice architecture for assisting the diagnosis of elderly people infectious diseases via a telemonitoring system. Using a bio-sensor kit applied to elderly people, they proposed a design to continously update vital signs database based on a flexible microservice architecture. Their proposed work analyzed the usability of the equipment, the performance of the architecture, and the service concept feasibility for e-health system. The system was based on a five-tier architecture using microservices. The results showed that the system is suitable for distributed computing, big data and NoSQL structures. Moreover, their key innovation is utilizing the telemonitoring for obtaining medical information to build predictive e-health models.

Pandeya *et al.* [7] proposed a cloud computing-based healthcare system that gathers individual health information and distributes it in the cloud vessel for analysis purpose. For evaluation scheme, the developed prototype was used as a test bed for gathering of electrocardiogram (ECG) data use case. Essentially, the system remotely obtains real time ECG data from patients which is then analyzed and monitored by the specialists, doctors or patients themselves. The study focuses on the delivery of health monitoring system which includes tools for conducting pattern designed investigation of contemporary and historic ECG data of all users. The system is developed using a conventional way of software development approach. The system, ECG monitoring and analysis, presents a Software as a Service (SaaS) for public users, who pay for it on a per examination basis from anywhere and anytime. Thus, ensuring the economy of scale of service and cost. The authors used a robust PKI (public key infrastructure) based authentication and authorization technique. However, in spite of the immense efforts to develop such a profound structure, the system is not immune from data security, confidentiality and availability issues. Model quintessentially makes use of all 3 service models of cloud, i.e. Saas, Paas and Iaas.

Hanen *et al.* [8] developed healthcare application prototype. The web based application offers access to its users through mobile interface. The major functionality of the prototype was to enable users to manage their health information. The unification of SOA and mobile cloud computing makes it a kind of hybrid approach. The system used CloudSim tool for the simulation purpose. The system is widely accessible and available to the patients and doctors. However, it lacks a security evaluation procedure. The authors did not mention that how the service is accessible to other services, which clearly reflect the collaboration problem.

Abawajy and Hassan [9] presented a patient health monitoring and management system using cloud services. For effective and high-quality remote patient health status monitoring the authors presented a framework. To evaluate their work they implemented their framework on case study of real time monitoring of patient grief from congestive heart

failure using ECG. Evaluation showed that the proposed work is flexible and scalable. Data clustering and classification mechanism was used to improve optimization of the communication at system level. Patients experiencing heart failure using ECG were taken for the experiment as a case study for real time examining. The evaluation results showed that the system offers energy efficient monitoring with flexibility and scalability of access through thin devices. However, 37 the work is only academic based tested in the laboratory and not implemented in real environment. Besides being not implemented in real time the proposed framework also had data security and privacy issues.

Forkan and Khalil [10] proposed a clinical support system for vital signs that assists the doctors in diagnostic judgment. The aim was to develop an intelligent automatic system for monitoring and decision support system by primitive vital signs and forecasting of irregularities using inter-relation among compound indications of the patients. The authors made use of classification algorithms to forecast the quantities and related irregularities making the system interoperable. The proposed work made use of EC2 instance of Amazon AWS and IaaS model of the cloud. Amazon S3 service was also employed for storage objectives. The system utilized the SOA approach for development.

Bitsaki *et al.* [11] aimed to enhance communication between patients and physicians and overcome the cost of follow up controls based on mobile application technology and cloud. SOA and data analytics techniques were employed for the purpose of cross communication between different web services. The system is able to monitor patients and promptly identify impeding complications. This resulted in decrease of cost and time spent by the doctors and patients. The system effectively reduced the cost and time of follow up medical visits of the patients.

Ra *et al.* [12] proposed Health Node, which is a general-purpose framework for emerging healthcare applications on cloud platforms using Node.js. The principal goals of Health Node are to provide explicit software design, application programming interface (API), and guidelines to achieve quick and expandable cloud based healthcare application. This paper specifically tailor Health Node for healthcare applications due to a significant potential for pointing many of the challenges of offering accessible, cost-effective, and convenient healthcare. With improvement support systems such as Health Node, they envisioned that the development of mobile-connected application systems will certainly rise in the healthcare domain.

El-Sofany *et al.* [13] presented a proposed application as HeathCare-as-a-Service (HC-AAS) model. Identification of system inputs, outputs, and rules are handled by fuzzy logic operations. In this paper, the authors proposed a fuzzy model that will be used for developing a cloud-based health application for medical diagnostic. Furthermore to preserve the health record of patients, the proposed model used the diseases and their symptoms. The main goal was to present a novel approach to develop medical diagnostic model that can help to increase the quality of healthcare in Egypt and KSA.

Darwish *et al.* [14] offerings a new concept of integration of CC and IoT for healthcare applications, named as the CloudIoT-Health paradigm. Term CloudIoT-Health and some key integration issues were presented in this paper to offer a practical vision to incorporate current components of CC and the IoT in healthcare applications. Also, this paper aimed to present the state of art and gap analysis of different levels of integration components, analyzing different existing proposals in CloudIoT-Health systems.

Singh *et al.* [15] suggested a model of planning adaptable e-healthcare services administration framework supported on distributed computing. Recommended framework integrated diverse divisions to create healthcare services framework. Health Data Management System based on client side, simple healthcare cloud, and application side created a readily attainable network. Biometric based confirmation system was appropriate in this condition since it conquested the constraints of insignificant crime and forget passwords. This is applied to the regular nominal ID secret key instrument for providing security. Moreover, it has high accuracy rate for secure information access and recovery. At last, the proposed framework improved administration cost and time which were also important to Zhao *et al.* [16] who were interested to run delay-sensitive applications in fog-cloud Internet of Things (IoT). They developed a microservice container fog system to solve the cost-efficient task scheduling problem in the heterogeneous fog servers.

Moreover, it is interesting to discuss some disruptive technologies to the topic such as blockchain, internet of things (IoT), and bigdata. Ejaz *et al.* [17] discussed a blockchain based framework for improving the latency of healthcare applications. The framework focused on providing secure remote monitoring system for the elderly people by integrating the capabilities of edge computing with blockchain. Additionally, the authors compared the baseline framework without blockchain with the proposed framework which counts on distributed smart contracts. They evaluated the latency, power consumption, network utilization, and computational load of the proposed architecture. Their concept demonstrated that combining edge computing and blockchain provide decentralized trust and reliable real-time access among other benefits without compromising the performance.

Lakhan *et al.* [18] devised a secure cost effective internet of medical things (IoMT) system based on blockchain Ethereum smart contract and fog cloud. The paper studied task scheduling with an objective of minimizing the healthcare application cost during offloading. They tested and analyzed multiple key performance indicators such as execution cost, fault tolerance, security, and resource leakage. The results showed the proposed IoMT system improves the IoMT services, outperforms the baseline cost and security of the distributed healthcare system.

TABLE 1. Summary and comparison of technologies and techniques used in healthcare systems in descending order.

Study	Year	Focus	Issues
Aksakalli et. al. [3]	2021	Microservice systematic deployment on limited capacity resources	Large number of microservices is cumbersome in practice
Khaleq & Ra [4]	2021	Intelligent autoscaling system for microservices with QoS constraints with no domain knowledge	QoS and Scalability
Vural and Koyuncu [5]	2021	Use domain driven design for finding the optimal granularity of the Microservice based on coupling.	Cohesion and Granularity issues
Ejaz et. al. [17]	2021	A framework to integrate edge computing and blockchainto address smart remote healthcare requirements	They did not model the blockchain consortium
Lakhan et. al. [18]	2021	Developed a blockchain-enabled fog cloud framework	High failure impact of blockchain model
Aileni et. al. [19]	2020	Communication architecture of collaborative edge computing model	Security risks of low power computing
Calderón-Gómez et. al. [6]	2020	Design and implement a microservice approach to enable detection of infectious diseases.	Flexible microservice architecture and High performance application
El-Sofany et. al. [13]	2019	A cloud based model controlled by fuzzy logic for improving the quality of healthcare in underdeveloped countries.	No practical implementation of the model
Darwish et. al. [14]	2019	Using cloud computing and IOT for healthcare applications	Integration issues
Singh et. al. [15]	2019	Created a framework for health data management based on cloud and client side	High accuracy and security
Ra et. al. [12]	2018	framework for healthcare applications on cloud using Node.js	Model structure Deployment and implementation
Abawajy and Hassan [9]	2017	Presents pervasive patient health monitoring system infrastructure	The algorithm is tested in the laboratory and is not implemented in real environment
Forkan and Khalil [10]	2017	developed a monitoring and decision support system to monitor vital signs based on SOA	Classification Speed is an issue and Did not consider the correlation among vital signs
Bitsaki et. al. [11]	2017	Enhance communication between patients and caregivers based on SOA	Scalability issues
Hanen et. al. [8]	2016	Offering mobile cloud computing webservice through mobile application.	Collaboration problem to show how the service is accessible to other services Uses conventional software development
Pandeya et. al. [7]	2012	Gather ECG data and send them over a cloud vessel for analysis.	System vulnerability to confidentiality and availability issues

Aileni *et al.* [19] focuses on IoMT signal processing and communication technology. They discussed edge computing and annonymization for patients data. In their model, multiple edge devices can come together to create collaborative edge for healthcare. Different communication architectures were discussed such as low power technologies, sensor network architecture, and multi-agent architecture among many others. The chapter highlighted the IoT connected devices for healthcare services and their use cases for higher risk patients. Disruptive technologies such as 5G connected wearable devices and their security vulnerabilities were also discussed. In addition, the authors posed the challenges facing IoT for remote healthcare services.

The microservices running on edge computing is one of the important disruptive technologies. Many researchers [16], [20]–[23] developed various frameworks for the deployment of microservices on edge computing. Islam *et al.* [24] were interested in the novel edge computing concept and applied it to a healthcare use case. They proposed virtual decentralized edge architecture platform based on lightweight microservices, named nanoservices [25]. They proposed a mechanism for efficient deployment of the nanoservices on local IoT nodes.

Microservice resource utilization, task scheduling [26], [27] for cloud or fog containers, autonomous microservice virtual server migration, container orchestration [28], and monolithic to microservice migration [29]–[32] are among the important topics for sensitive applications such as healthcare. Lin *et al.* [33] focused on resource scheduling for containers to reduce the overhead and imporve the cluster performance using an ant colony multi-objective optimization model for scheduling of microservice inside a container model. Many other researchers [34], [35] handled similar problems with various meta-heuristic techniques.

De Souza *et al.* [36] proposed BurstFlow tool for improving the communication of devices on the edge of the network and big data streaming applications hosted in the cloud. The tool adjusts the batch size dynamically depending on the required communication and computation time particularly for multi-cloud providers. The proposed tool overcomes the orchestration problem in cloud streaming. The authors experimented with real-time use case delivering up to 77% reduction on execution time compared to state of the art solutions for data partitioning based on Apache Flink.

Based on the above discussion, Table 1 shows the state-of-the-art reviewed microservice healthcare applications together with the disruptive technologies. The selection of the papers generally presenting a challenge of adopting microservice approach particularly in healthcare. It also discusses the advantages and shortcomings of microservices.

The comparison neither judge any of the pitfalls of the reviewed papers, nor offer any comment on the practicality of the discussed algorithms. In this paper, we have introduced a software development framework for developing healthcare applications. The contributions of this paper are as follows:

- We first provide a good background on software development approaches and discuss their suitability for healthcare.
- Then, we have proposed a microservice framework to develop healthcare applications exploiting cloud computing infrastructure along with the details of the phases, flow, and functional architecture.
- Based on the proposed framework, we developed and deployed a practical case-study of healthcare application in the cloud, taking the required design settings into account and demonstrate the effectiveness of our approach.
- The study reports a qualitative survey among experienced practitioners and developers to assess the preferences of the microservice-based framework for healthcare applications design.

The remainder of this paper is organized with the background in section II while the proposed framework is discussed in section III. Section IV shows the results and discussion and the paper is concluded with remarks in section V.

II. BACKGROUND

E-Healthcare systems have recently developed a considerable attention for service providers of health care and their consumers. They are classified into different categories such as electronic healthcare records [37], and personal records. The healthcare records focus especially on information storage and exchange of information. Users of healthcare applications are facilitated by the access to health information systems such as E-records, test report of patient, personal health record banks created by patient, healthcare information broadcast in real time environment with help of monitoring devices, wearable sensors and systems based on Internet of things technology (IOT). The usage of healthcare systems has increased greatly due to belief of patients in health data [38], [39].

A. MICRO SERVICES

The structure involved in the development of enterprise software systems is planned on the pattern to accommodate all the features of the business needs. The large business structures like enterprise resource planning (ERPs) and customer relationship management (CRMs) are constructed on this main concept of "single unit" for everything. However, for these systems countless post deployment problems occur e.g. horizontal scaling, single point of failure, maintenance cost, and technology lock-in and upgrading.

In SOA many services are aggregated based on the similarities of the contexts of functionalities. The SOA as compared to monolithic style is a more coarse-grained modular approach and the service entities communicate to each other through an enterprise service bus.

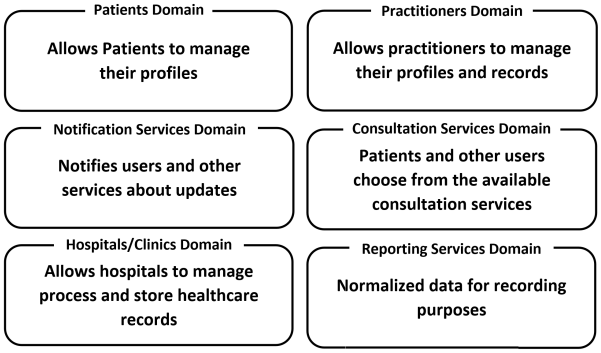

FIGURE 1. Domains and context segregation in micro-services.

The micro services are based on the concept of 'Single Responsibility Principle' assuming the fact that each micro service is responsible to do one and only one function. Each micro service is fine-grained autonomous structure with its own database and the service is independently developed and deployable. The independently developed services are considered independent because of the segregation of context and domains of the business requirements shown in Figure 1

The key concept is to isolate the functionalities based on the identified business capabilities. The implementation of micro-services leads the business competencies to be autonomous, fine grained and self-sustainable entities. Despite of the fact that the scope of a micro service is limited, it gives complete functionality without being dependent on other entities. The healthcare system can be realized with micro-services architecture in which business domains are clearly broken down into services contexts and micro services are structured from those service contexts.

B. MICRO SERVICES SCHEMATIC STRATEGIES

The confined micro-service scope, about the business requirements context and domain, benefits organizations to construct service that only performs one function.

- In the design stage of the micro-services, the boundaries of the services are decided on the context, business requirements and each business context should support the associated competencies.
- As the services are small units of independent structures, the development, unit testing, deployment and maintenance become easy.
- The whole emphasis is to design micro services that address the context of the business capabilities and not nearby making service smaller in term of code or cost. Business capability should define the size of the micro-service.
- Each micro-service must have its own singular and dedicated database for persistence. This leads to a more autonomy in the governance of the micro-services.

In this healthcare application context, functionalities of the conventional column system are divided into numerous

TABLE 2. Micro services messaging.

Synchronous Messaging	REST	Thrift	-
Asynchronous Messaging	AMQP	MQTT	-
Message Format	JSON	HTTP	SOAP

TABLE 3. Micro services integration.

Pattern Style	Behavior
Point to Point	Invoking services directly
API Gateway	Gateway is involved
Message Broker	Publish/ Subscribe

different micro-services such as 'Patients', 'Practitioners', 'Consultations', 'Notifications', 'Clinics' and 'Reporting'. The segmentation of the services made the scope of the services limited but focused business competencies are established. So that each service is more loosely coupled from each other and ensures the agility in development and deployment.

C. MICRO SERVICES MESSAGE STRUCTURE

The function calls are used to implement the functionalities of business logic in the traditional monolithic development approach. While in SOA the implementation and business logic bond gets loosely-coupled by the means SOAP (Simple Object Access Protocol) oriented web services messaging protocol. In case of micro-services structure, REST (Representational State Transfer) is used for synchronous messaging while AMQP (Asynchronous Message Queuing Protocol) is used for asynchronous messaging. Finally, SOAP or JSON (JavaScript Object Notation) can be utilized for the messaging formats. Table 1 shows the different messaging protocols and formats.

D. MICRO SERVICES INTEGRATION

All the concept of micro services spin around structures that can be developed and deployed independently. In order for the whole application to work and to coordinate among different micro services, the micro services system are establish. The same is achieved in SOA with the implementation of enterprise service bus, managing the core of the communication between user and data hubs. However, in micro-services the business logic gets encapsulated in the services and client end points. As mentioned above that the standard protocols used are HTTP, JSON etc. there are numerous communication patterns surfaced in micro services architecture. Table 3 shows few of the emerged patterns.

E. DECENTRALIZED DATA MANAGEMENT

For the applications to work in a monolithic way the repositories and data stores are implemented in a centralized manner so that the data persistence is maintained. The SOA approach in terms of data management is also not considerably different from monolith style. Numerous services structures can share one single database. This is due to the fact that many services in SOA share the same context of certain business logic. In micro-service architecture the functionalities are distinct respective to certain context of a business logic and maintaining one extensive database for all the services will either make it behave like SOA or the services will remain dependent on each other. Thus breaking the very heart of micro-service approach. So, essentially every micro service needs to have its own database. Main feature of applying de-centralized data management for micro-services.

- Each micro-service has its own single database.
- Databases of other micro-service are not accessible directly to other micro-services.
- In case of updating transactions of other micro-services' databases, the APIs are the only solution to this activity.

The decentralized data management give micro-services the luxury of choosing its own distinct data management technique e.g. one micro-service may use SQL database while others use Redis database.

F. DECENTRALIZED GOVERNANCE

Micro-services are dispersed architectures and so these support dispersed governance or no governance at all, as each micro service is fully independent of other micro-services. In SOA perspective the governance defines the controlling mechanism and instructions that how services are designed, developed, deployed, accessed and maintained over time. It also determines the contract between various service providers and consumers of the services, explicitly revealing about service provisioning and expectations. The micro-services are made as fully independent and decoupled services with variation of skills and platforms. So there is no need of defining common standards for services designing and development. We can conclude decentralized governance abilities of micro-services as given below.

- In-micro-services architecture due to the autonomy of the service structure the centralized design-time governance loses its efficacy.
- Micro-services are not restricted to rely on other entities for its design and implementation.

Finally, healthcare applications can be applied in the cloud using any of the known service levels such as SaaS, PaaS, or IaaS. Also, the deployment can be done in any cloud computing environment whether public, private, or hybrid.

III. PROPOSED FRAMEWORK

The Micro Service based Healthcare Application Framework (MSbHA) outlines a holistic methodology to development lifecycle of healthcare applications. The high-level view of proposed framework for examination and design of healthcare applications is shown in Figure 2 The lifecycle of prospective development project is divided into two parts. The first part is the requirements elicitation and analysis phase. All the requirements and information is managed and stored in a repository. The second part of the lifecycle is about the design, development and deployment of the services. These individual services offer a holistic functionality on their own. However, they may work in collaboration with each other in an integrated application.

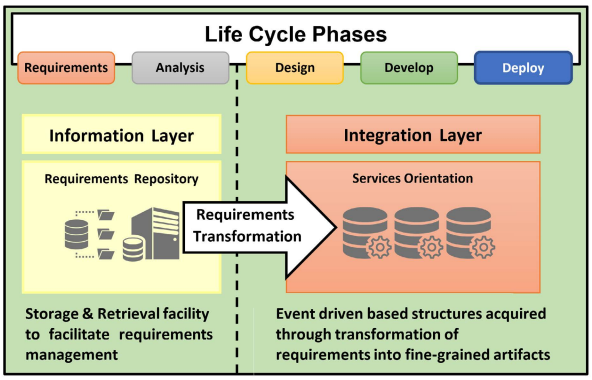

FIGURE 2. The proposed framework.

The micro services-based healthcare application framework consists of 4 major phases shown in Figure 3. In the predevelopment phase the requirements are gathered from different stakeholders. The regulations, standards, industry practices and compliances are taken into consideration for every stakeholder. In the analysis and design phase the requirements are classified into domain specific groups which are further broken down into contexts. Each context makes a distinct micro service. Different technologies and tools are identified by the system architects for the design of the micro services. Different teams are formatted for the design and development of tasks from the system. The system specifications, micro services development, coding and testing are the main tasks of development phase. The last phase of the framework is the services deployment phase. In the operational phase various micro services are deployed in the cloud infrastructure.

A. HOLISTIC FRAMEWORK IN ACTION

The holistic view of the proposed framework showing all the components and their interrelation is portrayed in Figure 4. The framework is organized in a stack view format. In the requirement elicitation phase the requirements of the healthcare system, to be built, are gathered from the stakeholder. The requirement engineers then separate the functional and non-functional requirements. The approved final requirements from the stakeholder are then forwarded to the systems domain experts. The main role of these experts is to identify the business domains of the required functionalities and also to define the contexts of the services. The identification of business domains helps the system developers in solving the complexity of the system into simpler entities, coarse-grained articles more specifically. Each business domain is then broken down into more fine-grained objects of micro services. Each micro service is responsible for delivering one task.

Once the micro services are designed then all the business logic and implementation details are forwarded to the application developers for coding. Here with the help of DevOps practice, development of microservice will take place. After the micro services are developed they are deployed in the

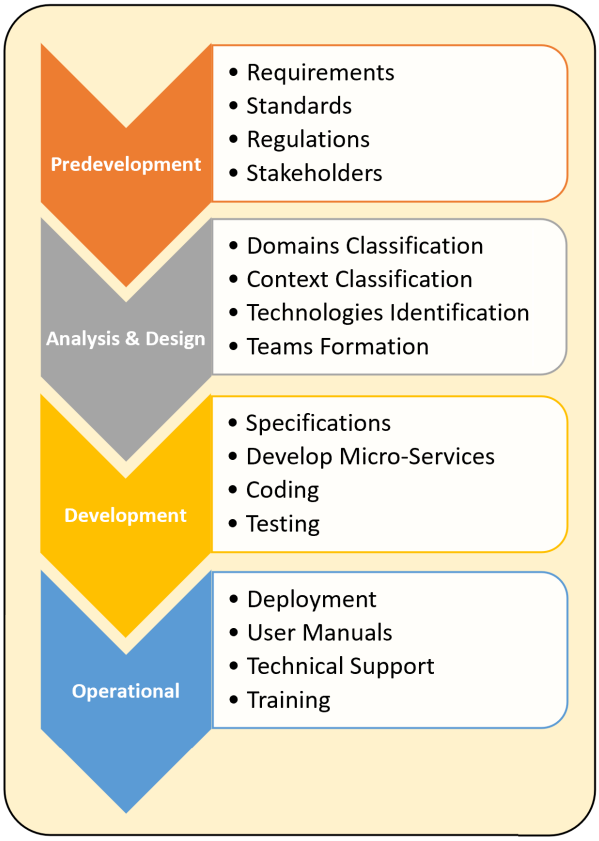

FIGURE 3. Phases of MSbHA framework.

cloud environment for users' utility. The users access the healthcare micro services through web browsers from their local devices. All the processing and storage of users' data are done in the cloud. The users access the healthcare micro services through web browsers from their local devices. All the processing and storage of user's data are done in the cloud. This paper defines system flow described by the specific situation and business space. Hence, the particular related micro services are characterized. In comparison to-SOA, micro services utilize basic light weight convention for information in this manner confining the overheads of service arrangement by enabling point to point administrations coordination. In service layer, patients, doctors, hospitals and clinic staff receive application services through browser. Each service communicates and accesses other services by the means of SOAP (Simple Object Access Protocol) oriented web services messaging protocol. In case of micro-services structure, REST (Representational State Transfer) is used for synchronous and AMQP for asynchronous messaging, AMQP (Asynchronous Message Queuing Protocol) is a simple and lightweight messaging procedure. While for the messaging formats SOAP or JSON (JavaScript Object Notation) can utilized. Data of patients, doctors consultation and notification is stored in database. The micro services functional view of healthcare application architecture is shown in Figure 5.

Every business space further has various individual micro service undertakings described and categorized as particular

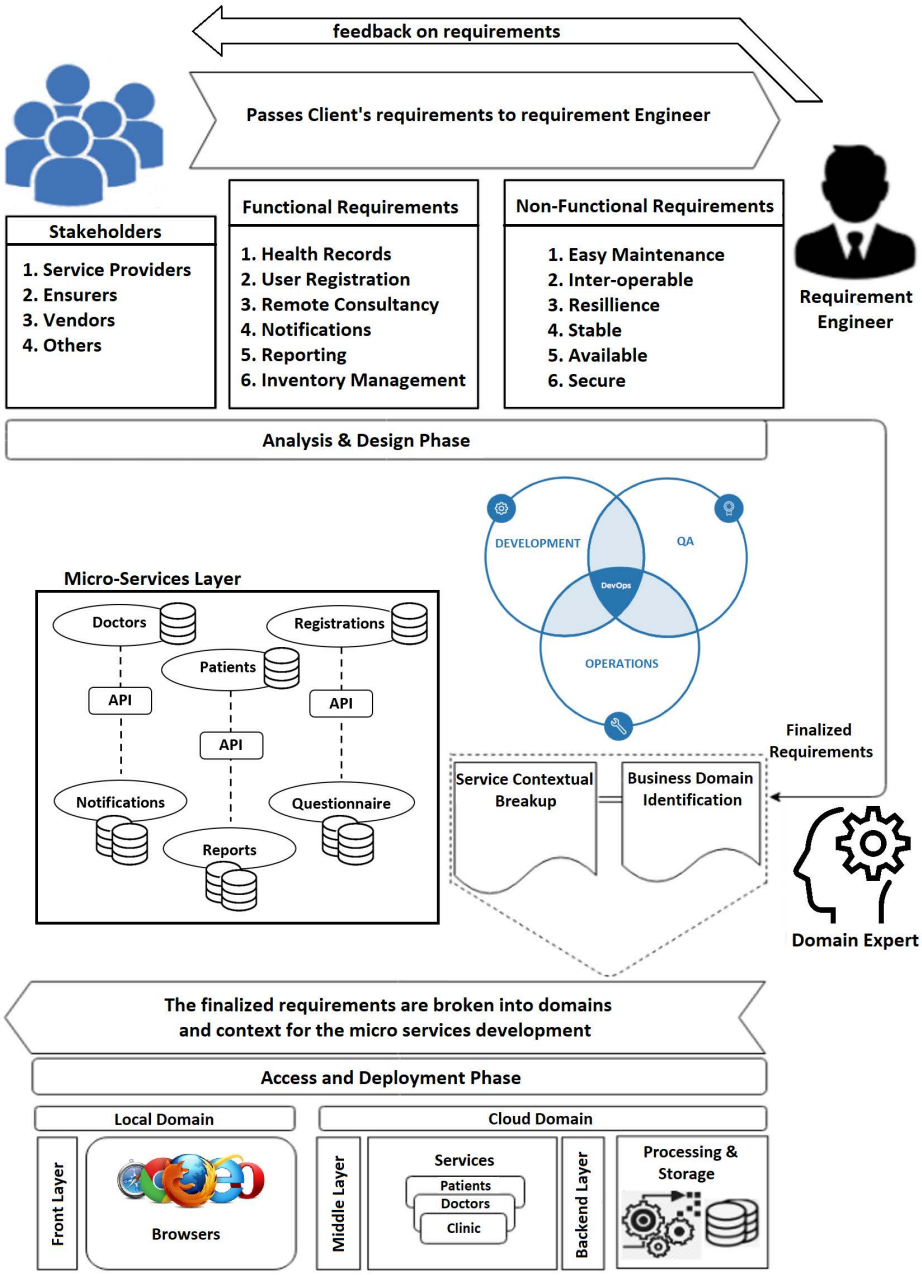

FIGURE 4. The flow of the proposed framework.

events. Representation of services' architectural flow is shown in Figure 6.

B. HEALTHCARE APPLICATION DESIGN

This investigation tries to build up an application to convey healthcare material facility with minimal effort, unrivaled medical worth and broad functioning importance. This examination suggests a particular SaaS-based-healthcare-service model to patients who have experienced a specific medical procedure and need additional follow-up from the specialists, specialists or doctors for post-surgery session. SaaS framework offers patients to associate respective primary care physicians for extra-consultancy by presenting an opinion poll. Figure 7 demonstrates suggested framework plan. At patient's side, SaaS-based system interface assembles and moves patients' data. This examination schematic plan gathers patient's basic data and administers this information in the cloud. The transmitted data is additionally broken down by the experts and physicians.

1) PATIENT CARE ARRANGEMENTS

After operation consultancy with surgeon is generally required half month once patient has experienced through an operation. Specialist inspects patients to ensure they are

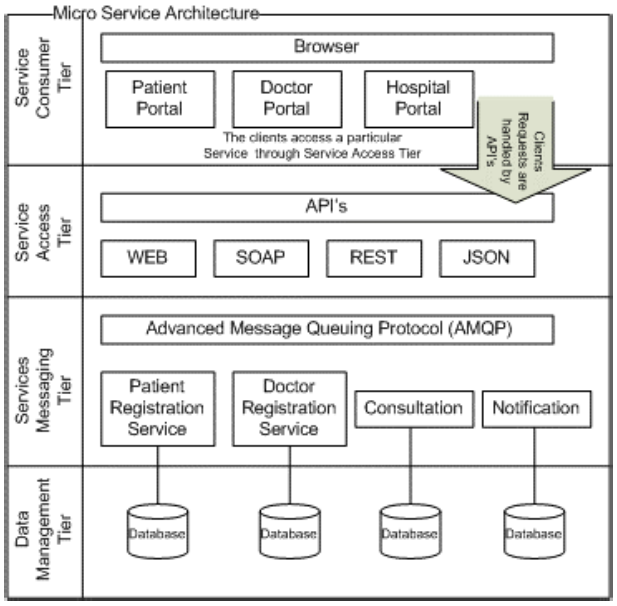

FIGURE 5. Functional view of healthcare micro services architecture.

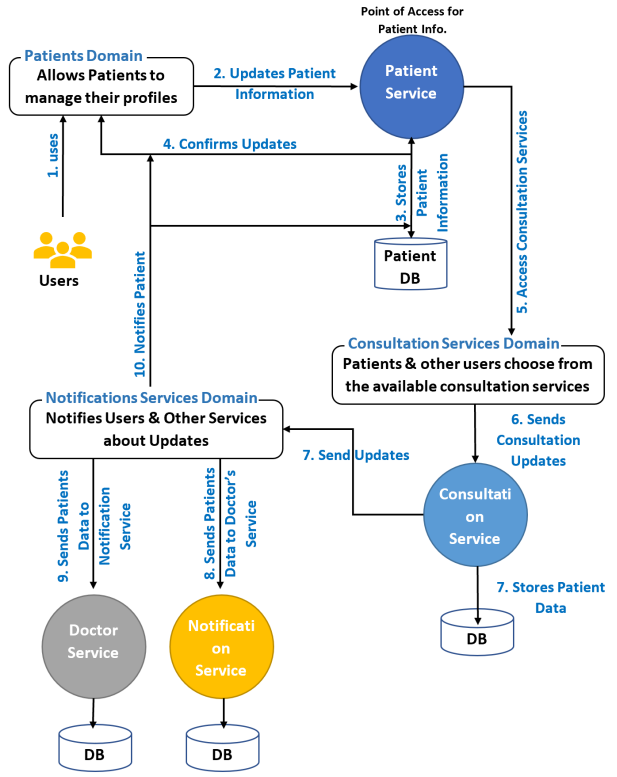

FIGURE 6. The micro services schematic flow.

recouping admirably. Procedure begins with specialists posing various inquiries about general treatment, some bothersome signs about patients' wellbeing, specific examining also explores results of operation. Surgeons may also give the patients advice and will explain if the patient needs further treatment. They can also give advice on any activity a patient must to be careful with. In case of open surgeries, the surgeons necessitate the patients to visit the clinic in person. However, in many post operation consultancies like laser treatment surgeries, for instance, the doctors' advice can be provided over phone or online, without requiring the patients to personally visit the clinic. Mind that, patients who require care also need a friend or relative to accompany them to the clinic. Follow up arrangements for the most part assume long time to take position. Patients would think that its significant to be furnished with a pre-visit survey that specialist would inquire. Consequently, a system would enormously profit the patients that request that they top off a consultancy shape and submit it online for more investigations preparing by the specialists to give advice to patients without visiting the facility.

2) USER SERVICES

Initial phase in process will be enrollment of clients, for example, patients, surgeons and specialists. Patient administration layer of business element conjures patient registration micro service. Patient management layer speaks to solitary business element space. It contains small scale administrations identified with patients, specialists and emergency clinics. register patient, remove patient, patient history is some of the micro services maintained in the patient management layer. The system offers patients, while staying at home, to fill up a questionnaire designed by the practitioners. In this questionnaire targets various inquiries concerning the post-medical procedure wellbeing and the state of patients. Patients cautiously record reactions to every one of inquiries. A few queries can be replied in just Yes or No arrangement; be that as it may, others may require a short. At the point when the entirety of the inquiries are addressed the patient just presents the structure.

3) CONSULTATION SERVICES

These are the services that direct and control all the interview correspondence among various clients for example patients, professionals, specialists and emergency clinics or facilities. They likewise deal with the surveys, their reactions either from the specialists or from different clients of system. Questionnaire service is expressly summoned and instated by the discussion administration at whatever point the patient requests an appointment from surgeons or doctors.

4) NOTIFICATION SERVICES

The surgeons are informed by notification micro service accommodation of structure. The specialist or doctor at his/her most punctual time looks at all the appropriate responses given by the patient and provides medical advice. Meeting structure will be spared in the distributed storage. The notice administration will be conjured once more, informing the patient about the conveyance of the specialist's restorative guidance. Additionally, the medical clinics can utilize the patients' record in the event that the patient is admitted to emergency clinic at some other event. The proposed system efficiently resolves the follow-up communication issue for the patients who live in remote areas and

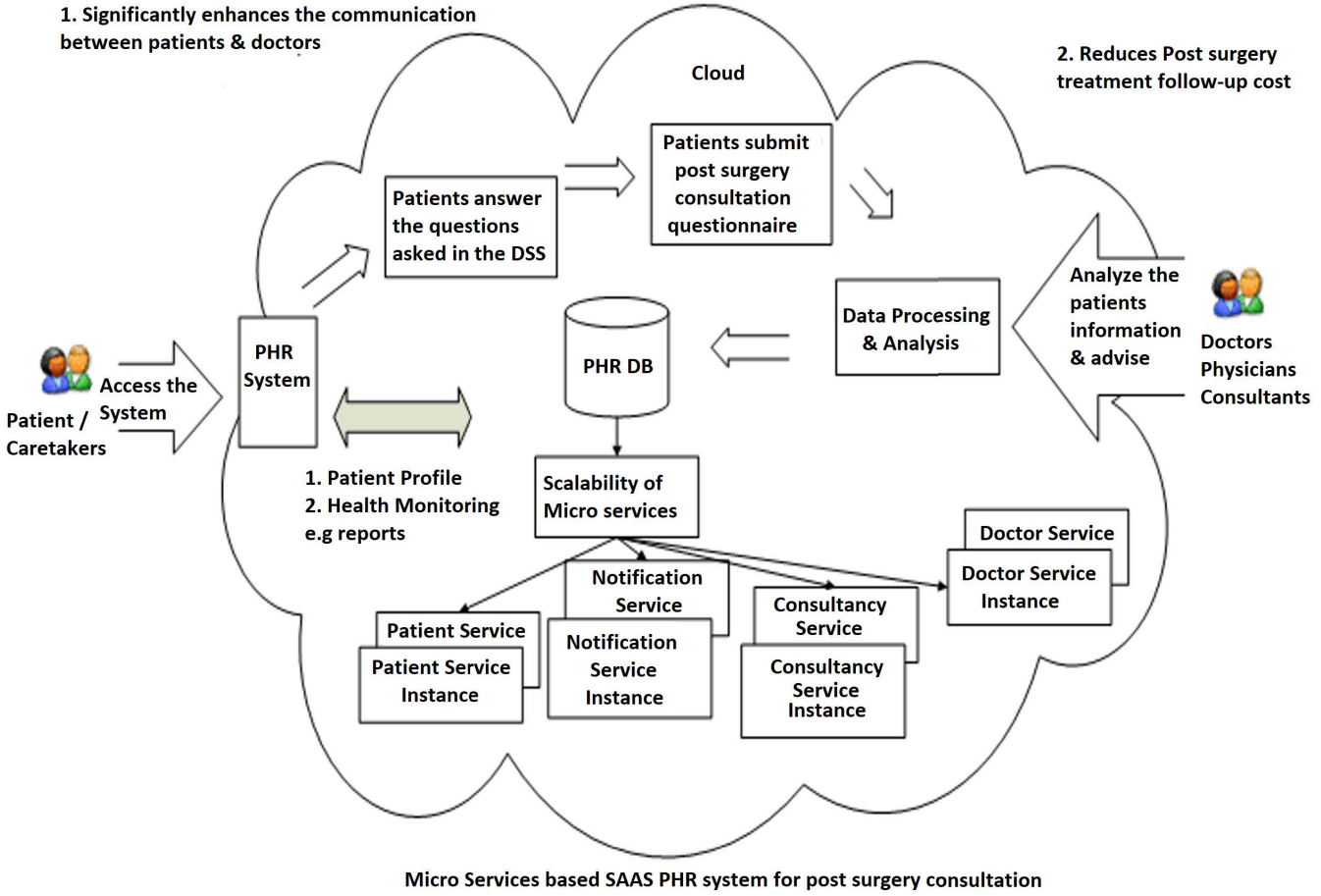

FIGURE 7. Proposed healthcare application design.

need post-surgery consultancy from their surgeons. The system does not only settle the interaction problem between patients and doctors but also is cost efficient in terms of being available, accessible and portable. This paper developed a proposed model by employing the micro services approach and using cloud computing infrastructure. The key benefits of the system are as follows.

- Simple integration with other healthcare services as the micro services incorporate new services effortlessly.
- Since the micro services are based on single responsibility principle so the logic of decomposition is fine grained by clearly separating the context of each service.
- Cloud services are employed by promising data confidentiality and security.
- Scalability of service are two fold, first the micro services consist of dedicated databases which make them more autonomous, second the cloud infrastructure also offer scalability feature.
- The services are interoperable easily by accessing each other through API's.

IV. RESULTS AND DISCUSSION

This research aimed to compare the SOA and microservice approaches for best utilization of the cloud resources and for best practical approach. For that purpose, both quantitative and qualitative case studies are used. Firstly, a case study of a healthcare application is deployed in the cloud using micro service and SOA approaches. Secondly, a survey of a number of professionals working in software development is conducted for qualitative comparison of the approaches. For cloud resource utilization, the micro service-based application is compared with SOA application using 3 parameters which are average response time, the possession of virtual machines and total number of dropped requests. Microsoft Azure was used, a PaaS layer cloud, for the deployment of the applications. The minimum number of virtual machines allotted to each deployed application is 1 and maximum is 5. The number of machines can fluctuate from 1 to 5 depending on the load.

The micro service and SOA applications deployed in the cloud are overwhelmed with server request load through a client. The number of requests sent to SOA is 48K and to micro services is 59K before resource limits are reached as depicted in figure 8 and 9 respectively. There are roughly 11K more requests to micro service application compared to SOA. For that many requests, the behavior of both applications against the metrics is observed. Table 4 summarizes the behaviour of microservice against

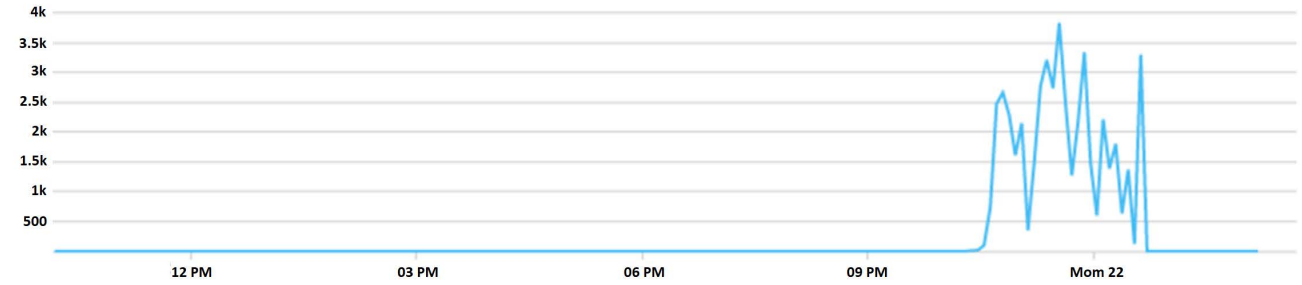

FIGURE 8. Total No. of requests for SOA (48.61K requests, PIDoctorRegistrationService).

FIGURE 9. Total No. requests for micro service, (59.58k requests, PLCDSSPatientRegistrationService).

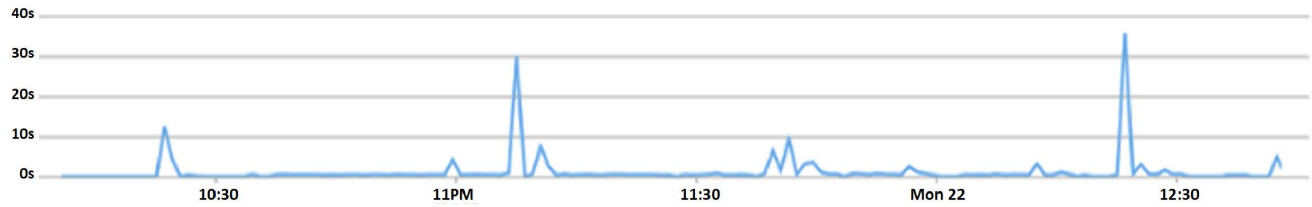

FIGURE 10. Average response time for SOA (1.2s, PIDoctorRegistrationService).

TABLE 4. Summary of the quantitative assessment of SOA and micro-service.

	SOA	Micro-Service
Requests	48K	59K
Response Time	1.2s	1.3s
Average # of servers	2	1
Max. # of servers	5	2
Dropped Requests	1.7K	93

SOA. The details of which is described in the following few paragraphs.

First, the impact of requests against the average response time of each application is observed. The average response time for SOA based application is found to be 1.2 seconds as shown in Figure 10 while the microservice is found to be 1.3 seconds as shown in Figure 11. Although, it may look like SOA is performing better than microservice in terms of average response time. It is crucial to investigate the resource utilization where the SOA kept expanding the load and fluctuating among the 5 servers as shown in Figure 12 with an average of 2 machines while the microservice, although had 11k more requests, was able to balance the load with a maximum of 2 servers and an average of 1 server as shown in Figure 13. If the results of SOA is compared to that of the microservice, it is found that despite microservice is handling 11,000 more requests, it efficiently utilized the resources. This clearly shows the advantage of the microservice approach over the SOA.

This paper approach explained earlier that the micro services structures are broken down at contextual level. Therefore, the microservice receiving overwhelmed number of requests is the only particular service of the application that needs to be scaled. This is clearly shown in the number of servers utilized to satisfy the requests.

In case of SOA application, which is a combination of more than one service, less efficient performance in case of cloud resource utilization can be seen. Although, the load was pointed at one specific service, the whole application was scaled. This reduced the ability of the SOA to handle resource utilization. In regards to the dropped requests, the microservice approach used provides a better quality of service represented in the number of dropped requests as shown in Figure 14. The microservice approach dropped 93 requests while the SOA approach dropped around 1.7k requests as shown in Figure 15 despite receiving 11k less requests than those of the microservice.

The previous figures showed better performance and resource utilization of microservice based approach in comparison to SOA.

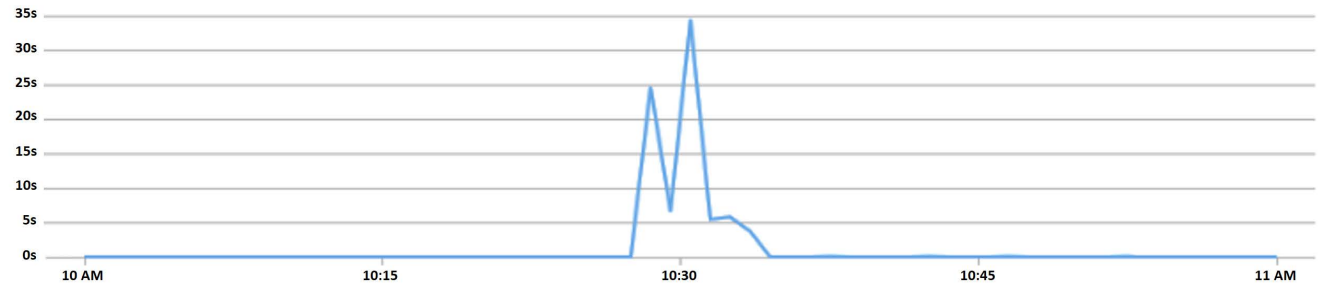

FIGURE 11. Average response time for micro service(1.3s, PLCDSSPatientRegistrationService).

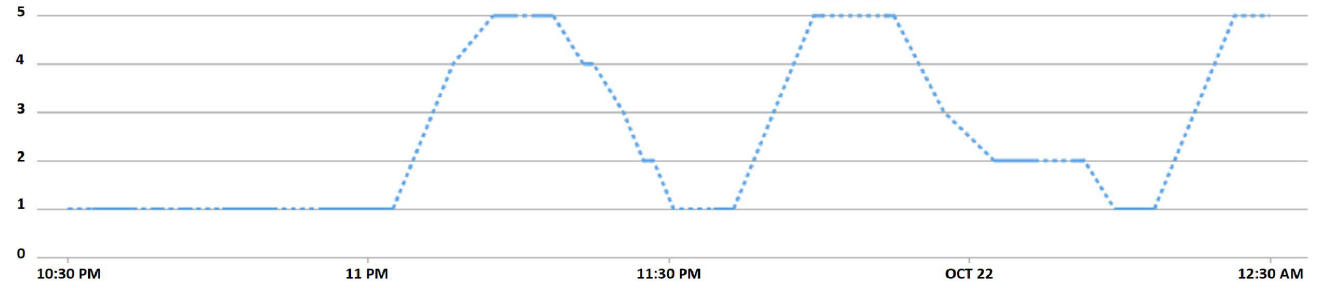

FIGURE 12. Average observed capacity for resource utilization in SOA (Number of autoscaling servers - 2.26).

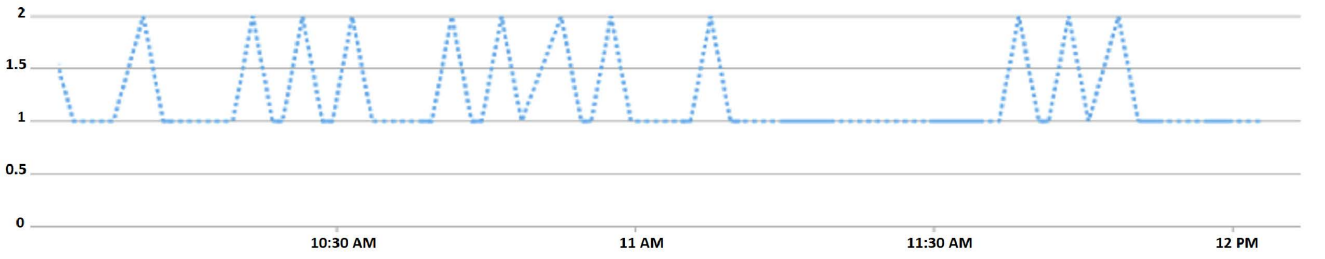

FIGURE 13. Average observed capacity for resource utilization in micro services (Number of autoscaling servers - 1.2).

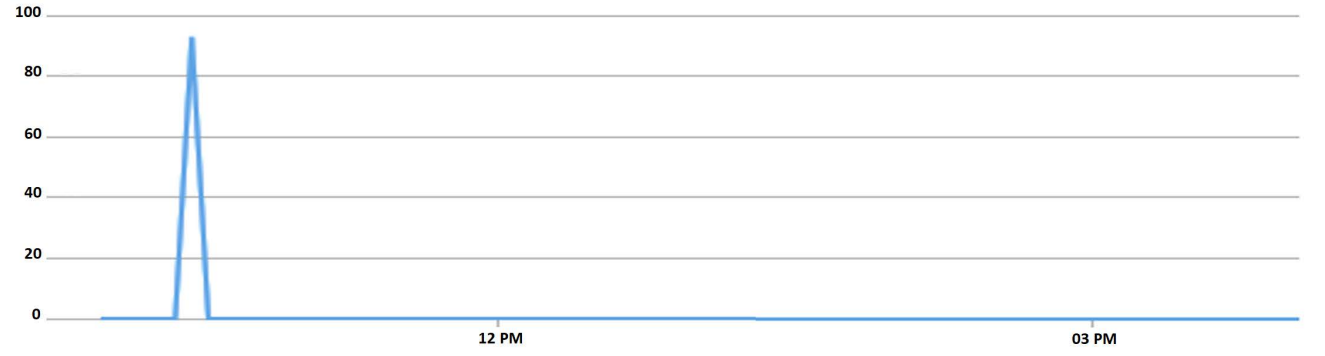

FIGURE 14. Server errors - number of dropped requests in micro services (93, PLCDSSPatientRegistrationService).

To qualitatively compare both approaches, a case study based on industry expert opinion reviews for the proposed work is conducted. Six (6) success factors are chosen, then a questionnaire is designed based on those success factors to evaluate the proposed work. The success factors that were taken into account are: Requirements Management, Interoperability, Resilience, Testing Agility, Operational Complexity, Maintenance and Resource Utilization in cloud. The questions asked were semi structured. The respondents had to Strongly Agree, Agree or Not Agree (neutral) on any of the asked questions. The classification of respondents comprised 28 people with different job titles such as Requirements Engineers, Software Developers, Testers, Project Managers and Software Architects. The minimum

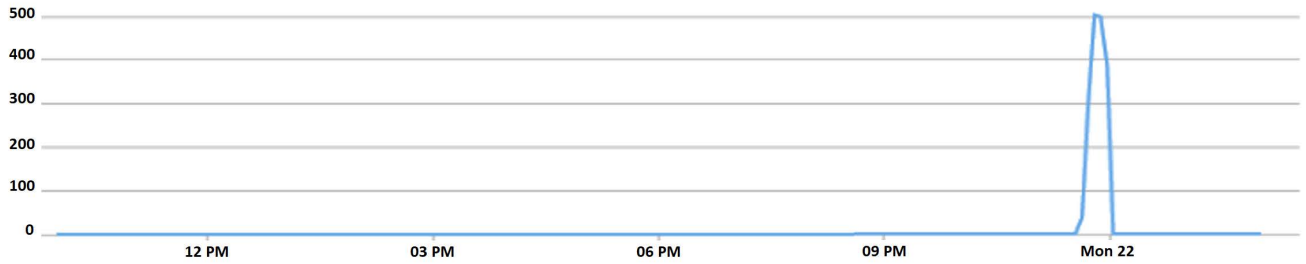

FIGURE 15. Server errors - number of dropped requests in SOA (1.73k, PIDoctorRegistrationService).

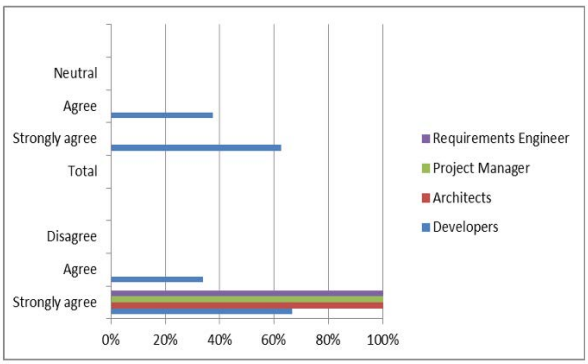

FIGURE 16. Requirements management.

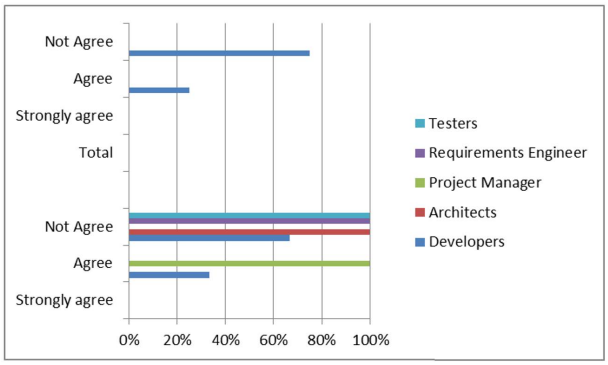

FIGURE 17. Interoperability.

years of experience for all the respondents was 1 year in the domain of software development using service-oriented approach. The questions and their respective responses are specified as below.

A. QUESTION 1: HOW WELL THE MSbHA MANAGE THE STAKEHOLDERS' NEEDS?

There are too many stakeholders in the domain of healthcare application development, each with his own domain expertise and contextual requirements. Figure 16, bottom bars show the responses per respondent category while to blue top bars show the total responses. It shows that 63% of the respondents strongly agreed to the fact that requirements management are well taken care of in the proposed MSbHA approach. It suggests a detailed requirement management process. The requirements are gathered and broken down into two parts; domain breakdown and context breakdown as described in section 4. The breakdown of the requirements is registered in the repository to ensure the availability of these artifacts throughout the life cycle of the project. It ensures that the work carried out by different experts is completed according to the micro service requirements breakdown regardless of the expert domain. The MSbHA framework is based on three major components: information repository, micro services and cloud computing. Every layer of component provides unique and comprehensive functions. The overall intricacy is lower as the proposed framework encourages the breakdown of the requirements into contexts and then based on those contextual entities the independent micro services structures are developed. Hence, 63% of the evaluators strongly agreed to the fact that this approach simplifies the information management and helps the developers to focus on small tasks. However, three of the evaluators mentioned that the requirements breakdown is a daunting task. It requires intense analytical and technical skills to perform. Also, they raised concerns about the consistency in requirements management. They argued that issues may arise when attempting to execute the approach in smaller teams. Nonetheless, all agreed that the approach handles requirements management in the right manner.

B. QUESTION 2: IS THE INTEROPERABILITY REDUCED BY THE PROPOSED APPROACH?

The interoperability of services is all about sharing of data among different services. Figure 17 shows that 75% of the respondents' opinions were that interoperability does not improve by using this approach. Either use of micro service-based approach for development or SOA approach, the interoperability remains the same. However, two of the developers pointed out that micro services use light weight messaging protocol such as JSON with Restful API, in which case the sharing of service process become very responsive and agile.

C. QUESTION 3: DOES THE PROPOSED APPROACH OFFER BETTER RESILIENCE THAN MONOLITH AND SOA APPROACHES?

This is the area where almost all of the experts declared an agreement on resilience. The result is shown in Figure 18. There are 88% of the experts who strongly agreed that the proposed approach offered more resiliency than the existing

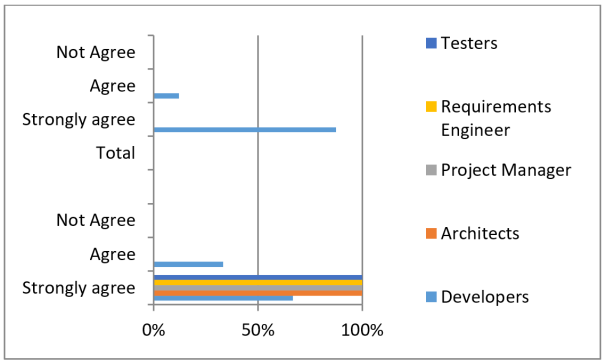

FIGURE 18. Resilience.

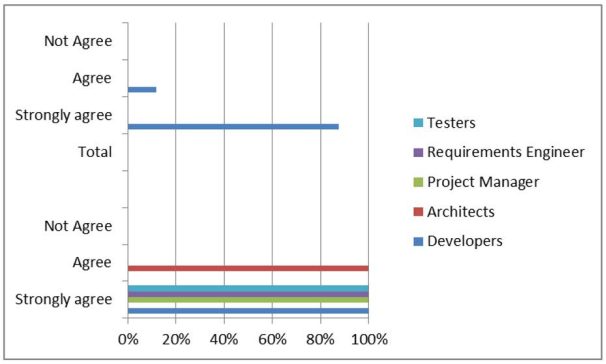

FIGURE 19. Testing.

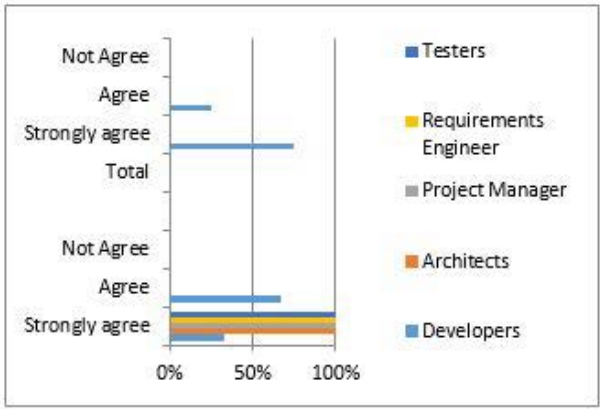

FIGURE 20. Operational complexity.

SOA and monolith approaches. Only 1 of the respondents did not strongly agree. The main component of the framework was the scheme of micro services. Micro services-based software development technique provides better autonomy to services orientation and is more focused on fine-grained modular structures. Each micro service maintains and manages its own individual database and uses APIs in order to access or retrieve data from other micro services. Additionally, most of the respondents agreed to the fact that micro services have brought stability in terms of application development process. Each micro service is responsible to act and function independently of other entities. If a specific micro service is down then it does not affect other services.

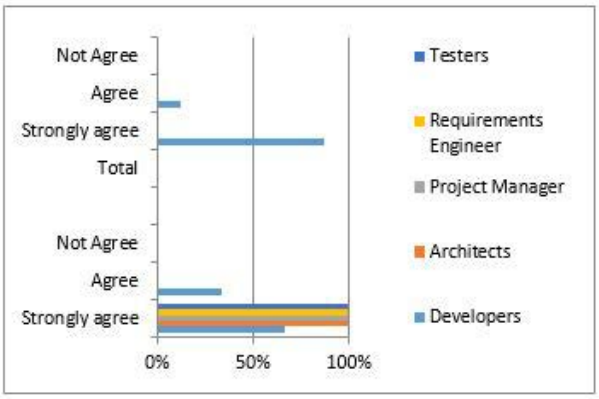

FIGURE 21. Maintenance.

D. QUESTION 4: HOW DOES TESTING AFFECT THE OVERALL AGILITY OF HEALTHCARE APPLICATIONS DEVELOPMENT USING MICRO SERVICES?

This again is one of the areas where micro service approach surpasses SOA software development approach. All the respondents, as shown in Figure 19, agreed that because micro services are tiny, and independent structures; once the requirements phase is completed and all the requirements for a specific micro service are finalized then the development becomes much more agile as the interdependencies of services among each other is removed. The development for each micro service becomes autonomous and is completed very swiftly, so as the unit testing. That is why 88% of the experts strongly agreed that microservice approach is more agile and testing friendly. However, one of the testers pointed out that ultimately, we have to unify the micro services as well to make the system work. In that case, the integrated testing becomes a little tough in terms of combining large population of micro services.

E. QUESTION 5: DOES THE PROPOSED APPROACH AFFECT THE DESIGN AND DEVELOPMENT OPERATIONAL COMPLEXITY?

The whole emphasis is to design micro services that address the context of the business capabilities and not about making the service smaller in term of code or cost. The business capability should define the size of the micro-service. Figure 20 shows 88% of the respondents strongly agreed to the fact that micro services do minimize the overall complexity of the system. Some respondents commented that the overall complexity of the system will be minimized but the organizations need to have a robust mechanism to monitor the colossal amount of micro services. The requirement engineers and project managers also pointed out that in the proposed approach, the complexity becomes apparent only at the very early stage of the process. That is, when the requirements are being gathered and broken down into domains and context levels. Once that stage is over, the proposed methodology shows more agility in terms of development and also the complexity becomes much less.

TABLE 5. Summary of the proposed microservice framework qualitative assessment.

Question	Outcome	Advantages	Disadvantages
Requirement Management	63%	approach takes good care of requirement management	requirement breakdown is daunting and requires technical and analytical skills
Interoperability	75%	utilizes light weight messaging services provides agility and responsiveness	interoperability does not improve much
Resilience	88%	Provides stability in application development	-
Overall Agility	88%	Interdependency is removed and development is autonomous	Harder in integrated overall testing
Operational Complexity	88%	Minimizes overall complexity	Require robust monitoring mechanism for the large number of services
Maintenance Robustness	88%	eaiser to maintain each microservice	-

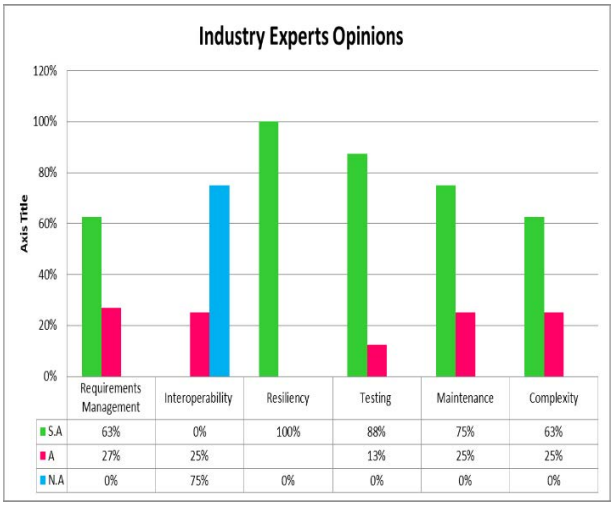

FIGURE 22. Industry experts opinion.

F. QUESTION 6: DOES THE PROPOSED APPROACH OFFER MORE ROBUSTNESS IN TERMS OF MAINTENANCE?

The decentralized data management gives micro services the luxury of choosing its own distinct data management mechanism. The micro-services are built as fully independent and decoupled services with the variety of technologies and platforms. In figure 21, it is shown that 88% of the experts strongly agreed that maintenance is much easier in micro services approach.

The industry-based interview summary, shown in Figure 22 and Table 5 together with the proposed approach advantages and disadvantages, is developed by using 6 parameters. According to most of the respondents the MSbHA framework does offer more flexibility as far as the accommodation of supplementary stakeholders is concerned. The flexibility of the framework invites the stakeholders to interoperate with the existing services structures. However, the project managers viewed that the teams must keep track of existing and prospective stakeholders as well as their requirements in the requirement repository. This is to keep the existing and new stakeholders and their requirements isolated and well traced. 63% of the experts viewed that the proposed approach better manages the product owners' requirements. Additionally, 75% completely disagreed that there is any effect by using the proposed approach on interoperability among services. Micro services-based software development technique gives more autonomy to services orientation and is more focused on fine-grained modular structures. Each micro service maintains and manages its own individual database and uses APIs in order to access or retrieve data from other micro services. 88% of the respondents strongly agreed to the fact that micro services have induced more stability and offered more resiliency. Each micro service is responsible to act and function independently of other entities. If a specific micro service is downgraded then it does not affect other services. The testing application is an overwhelming task and takes more time in the completion of application. The larger and tightly coupled a system is, the more time it takes to test all the cases. The experts ranked this approach superior by strongly agreeing. There are 88% of the respondents who ranked the micro services approach to make the unit testing easier and more agile. The smaller individual test cases for the micro services take less time to test. The development for each micro service is autonomous and gets completed very swiftly, so as the unit testing. Micro services break down the business logic at contextual level and performance of the overall system gets enhanced when it comes to the complexity of the system. The services in the system are loosely coupled, without being dependent on other services thus offering less complexity. However, if the system becomes larger, it does not mean that it will be difficult to maintain. In this case the maintenance of the services is much easier due to decentralized database, loosely coupled structures, and communication of services through APIs and separation of business logic at context level.

V. CONCLUDING REMARKS

In this work, we studied the healthcare application development, resulting in the identification of the problems of the traditional approaches in service availability, remote access to services, service provisioning, scalability, healthcare systems integration with each other. To address these issues, we introduced a microservice framework for developing healthcare applications. Our experimental evaluation of our implementation of the microservice framework highlighted the feasibility of deploying the proposed scheme in the cloud; the proposed approach shows a significant reduction in response time and number of dropped requests and substantial increases in resource utilization compared to service-oriented architecture. To validate our approach from software development perspectives, we performed a qualitative survey among a number of developers; the analysis of the survey reveals

the preferences of using the microservice-based framework because of the proven advantages mentioned above.

In the future, the proposed framework can be extended by incorporating artificial intelligence (AI)-based autoscaling that can dynamically adjust the required computational resources delivered to a cloud workload on-demand. It is not straightforward to identify the required level of granularity for microservices architecture. It also usually requires appropriate customizations. As such, it is hard to implement autoscale microservices. Therefore, the use of AI techniques [40] can be used to optimize the service model and find the appropriate autoscaling parameters. Such extension is indeed possible; for example, authors in [41] proposed a cloud application auto-scaling approach based on a reinforcement learning method for optimal resource allocation dynamically.

REFERENCES

[1] S. M. Riazul Islam, D. Kwak, M. Humaun Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015, doi: 10.1109/ACCESS.2015.2437951.

[2] M. Hossain, S. M. R. Islam, F. Ali, K.-S. Kwak, and R. Hasan, "An Internet of Things-based health prescription assistant and its security system design," *Future Gener. Comput. Syst.*, vol. 82, pp. 422–439, May 2018, doi: 10.1016/j.future.2017.11.020.

[3] I. Karabey Aksakalli, T. Celik, A. Burak Can, and B. Tekinerdogan, "Systematic approach for generation of feasible deployment alternatives for microservices," *IEEE Access*, vol. 9, pp. 29505–29529, 2021, doi: 10.1109/ACCESS.2021.3057582.

[4] A. Abdel Khaleq and I. Ra, "Intelligent autoscaling of microservices in the cloud for real-time applications," *IEEE Access*, vol. 9, pp. 35464–35476, 2021, doi: 10.1109/ACCESS.2021.3061890.

[5] H. Vural and M. Koyuncu, "Does domain-driven design lead to finding the optimal modularity of a microservice?" *IEEE Access*, vol. 9, pp. 32721–32733, 2021, doi: 10.1109/ACCESS.2021.3060895.

[6] H. Calderón-Gómez, L. Mendoza-Pittí, M. Vargas-Lombardo, J. Manuel Gómez-Pulido, J. Luis Castillo-Sequera, J. Sanz-Moreno, and G. Sención, "Telemonitoring system for infectious disease prediction in elderly people based on a novel microservice architecture," *IEEE Access*, vol. 8, pp. 118340–118354, 2020, doi: 10.1109/ACCESS.2020.3005638.

[7] S. Pandey, W. Voorsluys, S. Niu, A. Khandoker, and R. Buyya, "An autonomic cloud environment for hosting ECG data analysis services," *Future Gener. Comput. Syst.*, vol. 28, no. 1, pp. 147–154, 2012, doi: 10.1016/j.future.2011.04.022.

[8] J. Hanen, Z. Kechaou, and M. B. Ayed, "An enhanced healthcare system in mobile cloud computing environment," *Vietnam J. Comput. Sci.*, vol. 3, no. 4, pp. 267–277, 2016, doi: 10.1007/s40595-016-0076-y.

[9] J. H. Abawajy and M. M. Hassan, "Federated Internet of Things and cloud computing pervasive patient health monitoring system," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 48–53, Jan. 2017, doi: 10.1109/MCOM.2017.1600374CM.

[10] A. Forkan and I. Khalil, "A clinical decision-making mechanism for context-aware and patient-specific remote monitoring systems using the correlations of multiple vital signs," *Comput. Methods Programs Biomed.*, vol. 139, pp. 1–16, Feb. 2017, doi: 10.1016/j.cmpb.2016.10.018.

[11] M. Bitsaki, G. Koutras, H. Heep, and C. Koutras, "Cost-effective mobile-based healthcare system for managing total joint arthroplasty follow-up," *Healthcare Inform. Res.*, vol. 23, no. 1, pp. 67–73, 2017, doi: 10.4258/hir.2017.23.1.67.

[12] H.-K. Ra, H. J. Yoon, S. H. Son, J. A. Stankovic, and J. Ko, "HealthNode: Software framework for efficiently designing and developing cloud-based healthcare applications," *Mobile Inf. Syst.*, vol. 2018, Apr. 2018, Art. no. 6071580, doi: 10.1155/2018/6071580.

[13] H. F. El-Sofany and I. A. T. F. Taj-Eddin, "A cloud-based model for medical diagnosis using fuzzy logic concepts," in *Proc. Int. Conf. Innov. Trends Comput. Eng. (ITCE)*, Feb. 2019, pp. 162–167, doi: 10.1109/ITCE.2019.8646624.

[14] A. Darwish, A. E. Hassanien, M. Elhoseny, A. K. Sangaiah, and K. Muhammad, "The impact of the hybrid platform of Internet of Things and cloud computing on healthcare systems: Opportunities, challenges, and open problems," *J. Ambient Intell. Humanized Comput.*, vol. 10, pp. 4151–4166, Dec. 2019, doi: 10.1007/s12652-017-0659-1.

[15] I. Singh, D. Kumar, and S. Kumar Khatri, "Improving the efficiency of e-healthcare system based on cloud," in *Proc. Amity Int. Conf. Artif. Intell. (AICAI)*, Feb. 2019, pp. 930–933, doi: 10.1109/AICAI.2019.8701387.

[16] H. Zhao, S. Deng, Z. Liu, J. Yin, and S. Dustdar, "Distributed redundancy scheduling for microservice-based applications at the edge," *IEEE Trans. Services Comput.*, early access, Aug. 3, 2020, doi: 10.1109/TSC.2020.3013600.

[17] M. Ejaz, T. Kumar, I. Kovacevic, M. Ylianttila, and E. Harjula, "Healthblockedge: Blockchain-edge framework for reliable low-latency digital healthcare applications," *Sensors*, vol. 21, no. 7, p. 2502, Apr. 2021, doi: 10.3390/s21072502.

[18] A. Lakhan, M. A. Mohammed, A. N. Rashid, S. Kadry, T. Panityakul, K. H. Abdulkareem, and O. Thinnukool, "Smart-contract aware ethereum and client-fog-cloud healthcare system," *Sensors*, vol. 21, no. 12, p.4093, 2021, doi: 10.3390/s21124093.

[19] R. M. Aileni, G. Suciu, C. A. V. Sukuyama, and S. Pasca, "Internet of Things and communication technology synergy for remote services in healthcare," in *IoT and ICT for Healthcare Applications* (EAI/Springer Innovations in Communication and Computing), N. Gupta and S. Paiva, Eds., 1st ed. Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-030-42934-8_5.

[20] S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, "Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 939–951, Mar. 2021, doi: 10.1109/TMC.2019.2957804.

[21] W. Jin, R. Xu, T. You, Y.-G. Hong, and D. Kim, "Secure edge computing management based on independent microservices providers for gateway-centric IoT networks," *IEEE Access*, vol. 8, pp. 187975–187990, 2020, doi: 10.1109/ACCESS.2020.3030297.

[22] A. Samanta and J. Tang, "*Dyme*: Dynamic microservice scheduling in edge computing enabled IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6164–6174, Jul. 2020, doi: 10.1109/JIOT.2020.2981958.

[23] I.-D. Filip, F. Pop, C. Serbanescu, and C. Choi, "Microservices scheduling model over heterogeneous cloud-edge environments as support for IoT applications," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2672–2681, Aug. 2018, doi: 10.1109/JIOT.2018.2792940.

[24] J. Islam, T. Kumar, I. Kovacevic, and E. Harjula, "Resource-aware dynamic service deployment for local IoT edge computing: Healthcare use case," *IEEE Access*, vol. 9, pp. 115868–115884, 2021, doi: 10.1109/ACCESS.2021.3102867.

[25] E. Harjula, P. Karhula, J. Islam, T. Leppänen, A. Manzoor, M. Liyanage, J. Chauhan, T. Kumar, I. Ahmad, and M. Ylianttila, "Decentralized IoT edge nanoservice architecture for future gadget-free computing," *IEEE Access*, vol. 7, pp. 119856–119872, 2019, doi: 10.1109/ACCESS.2019.2936714.

[26] M. Fazio, A. Celesti, R. Ranjan, C. Liu, L. Chen, and M. Villari, "Open issues in scheduling microservices in the cloud," *IEEE Cloud Comput.*, vol. 3, no. 5, pp. 81–88, Sep./Oct. 2016, doi: 10.1109/MCC.2016.112.

[27] L. Bao, C. Wu, X. Bu, N. Ren, and M. Shen, "Performance modeling and workflow scheduling of microservice-based applications in clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 9, pp. 2114–2129, Sep. 2019, doi: 10.1109/TPDS.2019.2901467.

[28] A. Khan, "Key characteristics of a container orchestration platform to enable a modern application," *IEEE Cloud Comput.*, vol. 4, no. 5, pp. 42–48, Sep./Oct. 2017, doi: 10.1109/MCC.2017.4250933.

[29] J. P. Delgado Preti, A. N. Araujo Souza, E. C. Freiberger, and T. De Almeida Lacerda, "Monolithic to microservices migration strategy in public safety secretariat of Mato Grosso," in *Proc. Int. Conf. Electr., Commun., Comput. Eng. (ICECCE)*, Jun. 2021, pp. 1–5, doi: 10.1109/ICECCE52056.2021.9514268.

[30] O. Al-Debagy and P. Martinek, "Extracting microservices' candidates from monolithic applications: Interface analysis and evaluation metrics approach," in *Proc. IEEE 15th Int. Conf. Syst. Syst. Eng. (SoSE)*, Jun. 2020, pp. 289–294, doi: 10.1109/SoSE50414.2020.9130466.

[31] C.-Y. Fan and S.-P. Ma, "Migrating monolithic mobile application to microservice architecture: An experiment report," in *Proc. IEEE Int. Conf. AI Mobile Services (AIMS)*, Jun. 2017, pp. 109–112, doi: 10.1109/AIMS.2017.23.

[32] G. Mazlami, J. Cito, and P. Leitner, "Extraction of microservices from monolithic software architectures," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 524–531, doi: 10.1109/ICWS.2017.61.
[33] M. Lin, J. Xi, W. Bai, and J. Wu, "Ant colony algorithm for multi-objective optimization of container-based microservice scheduling in cloud," *IEEE Access*, vol. 7, pp. 83088–83100, 2019, doi: 10.1109/ACCESS.2019.2924414.
[34] B. Ma, H. Ni, X. Zhu, and R. Zhao, "A comprehensive improved salp swarm algorithm on redundant container deployment problem," *IEEE Access*, vol. 7, pp. 136452–136470, 2019, doi: 10.1109/ACCESS.2019.2933265.
[35] R. Liu, P. Yang, H. Lv, and W. Li, "Multi-objective multi-factorial evolutionary algorithm for container placement," *IEEE Trans. Cloud Comput.*, early access, Dec. 21, 2021, doi: 10.1109/TCC.2021.3137400.
[36] P. Ricardo Rodrigues De Souza, K. J. Matteussi, A. Da Silva Veith, B. F. Zanchetta, V. R. Q. Leithardt, Á. L. Murciego, E. Pignaton De Freitas, J. C. S. Dos Anjos, and C. F. R. Geyer, "Boosting big data streaming applications in clouds with BurstFlow," *IEEE Access*, vol. 8, pp. 219124–219136, 2020, doi: 10.1109/ACCESS.2020.3042739.
[37] A. Bahga and V. K. Madisetti, "A cloud-based approach for interoperable electronic health records (EHRs)," *IEEE J. Biomed. Health Informat.*, vol. 17, no. 5, pp. 894–906, Sep. 2013, doi: 10.1109/JBHI.2013.2257818.
[38] S. Oh, J. Cha, M. Ji, H. Kang, and S. Kim, "Architecture design of healthcare software-as-a-service platform for cloud-based clinical decision support service," *Healthcare Informat. Res.*, vol. 21, no. 2, p. 102, 2015, doi: 10.4258/hir.2015.21.2.102.
[39] D. Hayn and G. Schreier, *Health Informatics Meets EHealth: Digital Insight–Information-Driven Health & Care. Proceedings of the 11th EHealth2017 Conference*, vol. 236. Amsterdam, The Netherlands: IOS Press, 2017.
[40] F. Ali, S. El-Sappagh, S. M. R. Islam, A. Ali, M. Attique, M. Imran, and K.-S. Kwak, "An intelligent healthcare monitoring framework using wearable sensors and social networking data," *Future Gener. Comput. Syst.*, vol. 114, pp. 23–43, Jan. 2021, doi: 10.1016/j.future.2020.07.047.
[41] Y. Wei, D. Kudenko, S. Liu, L. Pan, L. Wu, and X. Meng, "A reinforcement learning based auto-scaling approach for SaaS providers in dynamic cloud environment," *Math. Problems Eng.*, vol. 2019, Feb. 2019, Art. no. 5080647, doi: 10.1155/2019/5080647.

JOHN ZAKI received the master's degree from Cambridge University, the M.B.A. degree from Glasgow University, and the Ph.D. degree from Mansoura University. He is currently the Vice Director of the Communication and Information Technology Centre and the Director of E-learning with Mansoura University, where he is also an Assistant Professor with the Department of Computer and Systems. He worked in both technical and managerial positions in various multinational companies. He is experienced in establishing tech start-ups, project management, and business processes. His research interests include artificial intelligence, software engineering, control systems, and e-learning technology.

S. M. RIAZUL ISLAM (Member, IEEE) has been working as an Assistant Professor with the Department of Computer Science and Engineering, Sejong University, South Korea, since March 2017. From 2014 to 2017, he worked with the Wireless Communications Research Center, Inha University, South Korea, as a Postdoctoral Fellow. In 2016 and 2017, he was also affiliated with Memorial University, Canada, as a Postdoctoral Fellow. From 2005 to 2014, he was with the University of Dhaka, Bangladesh, as an Assistant Professor and a Lecturer with the Department of Electrical and Electronic Engineering. In 2014, he worked with the Department of Solution Laboratory for Advanced Research, Samsung Research and Development Institute Bangladesh, as a Chief Engineer. His research interests include wireless communications, the Internet of Things, and applied artificial intelligence.

NORAH SALEH ALGHAMDI received the bachelor's degree in computer science from Taif University, Taif, Saudi Arabia, and the master's degree in computer science and the Ph.D. degree from the Department of Computer Science, La Trobe University, Melbourne, Australia. She is currently an Associate Professor with the Department of Computer Science, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University (PNU), Riyadh, Saudi Arabia, where she is also the Vice-Dean of Quality Assurance, since 2019. Her research interests include data mining, machine learning, text analytics, image classification, bioengineering, and deep learning. She has participated in organizing the international conference on computing (ICC 2019). She is a member of the Reviewer Committee of several journals, such as IEEE Access, *Journal of Computer Science*, and *International Journal of Web Information Systems*. She has authored or coauthored many articles published in a well-known journals in the research field.

M. ABDULLAH-AL-WADUD (Member, IEEE) received the B.S. degree in computer science and the M.S. degree in computer science and engineering from the University of Dhaka, Bangladesh, in 2003 and 2004, respectively, and the Ph.D. degree in computer engineering from Kyung Hee University, South Korea, in 2009. He worked as a Faculty Member of the Department of Industrial and Management Engineering, Hankuk University of Foreign Studies, South Korea, from 2009 to 2014. He also worked as a Lecturer with the Faculty of Sciences and Information Technology, Daffodil International University, Bangladesh, in 2003, and the Faculty of Sciences and Engineering, East West University, Bangladesh, in 2004. He is currently an Associate Professor with the Department of Software Engineering, King Saud University, Saudi Arabia. His research interests include pattern recognition, optimization, computer vision, cloud computing, recommender systems, sensor, and *ad-hoc* networks.

KYUNG-SUP KWAK (Life Senior Member, IEEE) received the B.S. degree from Inha University, Inchon, South Korea, in 1977, the M.S. degree from the University of Southern California, in 1981, and the Ph.D. degree from the University of California at San Diego, in 1988, under the Inha University Fellowship and the Korea Electric Association Abroad Scholarship Grants, respectively. He worked at Hughes Network System, San Diego, CA, USA, and the IBM Network Research Center, Research Triangle Park, NC, USA, from 1988 to 1990. Since 1990, he has been with Inha University as an Inha Fellow Professor. He is currently an Inha Hanlim Professor, South Korea. His research interests include multiple access communication systems, mobile communication systems, UWB radio systems and *ad-hoc* networks, and high-performance wireless internet. He is a member of IEICE, KICS, and KIEE.