Date of publication DEC-31, 2022, date of current version NOV-15, 2022.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.21.4.2783

A Microservice-based Software Architecture for Improving the Availability of Dental Health Records

ARCILA-DIAZ JUAN [1], CARLOS VALDIVIA[2]

[1]Postgraduate School, Universidad Nacional Pedro Ruiz Gallo. Lambayeque, Peru.
[2]School of Computer Engineering, Universidad Nacional Pedro Ruiz Gallo. Lambayeque, Peru.

Corresponding author: Arcila-Diaz Juan (e-mail: jcarlos.ad7@gmail.com).

ABSTRACT In order to keep accessible, the patient care information recorded by a dental provider, a software architecture must be designed to allow availability among the different providers.

In this research, a software architecture based on the Microservices approach is designed to enable the availability of dental medical records. The quality attributes and functional requirements were identified to design the architecture, determining that it should be composed of 4 Microservices, Patient, Dental Medical Record, Odontogram and Dental Service Provider; each microservice implements its database, the secure communication between the microservices and the clients is done through an API Gateway of HTTP resources and an authentication token.

To evaluate the software architecture, a prototype was developed in which each component was deployed in containers using the Microsoft Azure App Service. On this prototype load tests were performed to evaluate Availability and Performance determining that up to 21 dental records per second can be available with 100% availability, and if the demand of requests increases the architecture scales automatically.

KEYWORDS Dental health record; microservices; availability; software architecture.

I. INTRODUCTION

A person may seek dental care at private dental clinics, public health facilities or with different dental surgeons on an individual basis. If the patient goes to a dental provider for the first time, the patient's initial data is recorded, starting with the patient's data in his or her Dental Health Record (DHR) and odontogram following the corresponding health regulations, this information is stored either in a Paper Health Record (PHR) or Electronic Medical Record (EMR) using an information system, this health record is only managed by the service provider who made the registration. Many times, the patient must change dental providers, repeating the process of registering their DHR data, causing additional use of time and resources, in addition to the loss of information on the dental procedures performed.

In Peru, Law 30024 [1] created the National Registry of Electronic Medical Records (RENHICE), which is a database of the affiliation of each person with a list of the health facilities that have provided care and generated an EMR.

Research has been carried out to improve EMR management and patient data. In [2], a cloud-based EMR-exchange prototyping system using RESTful services was implemented on the basis of the Integrating the Healthcare Enterprise's Cross-Enterprise Document Sharing integration profile and the existing EMR exchange system in Taiwan. In the work [3], to improve the quality and completeness of data collected from the mother and child during their shared maternity care and to enable the exchange of health information between outpatient clinic, hospital and public health services in rural Kenya, an EMR management system was implemented.

There are even works to ensure the confidentiality of EMR, in paper [4], a solution for confidential EMR management in the cloud is proposed, whose basic idea is to deploy a trusted local server between the cloud and each medical information management system.

The DHR must be managed in the same way [5], so the software to be developed must have an architecture that allows the availability of DHR while maintaining the security and integrity of the information and a standard format [6].

An alternative software architecture that allows the availability of DHR is the Service Oriented Architecture (SOA) [7], although there is another style of software architecture called Microservices [8], which allows applications to be divided into its smaller components and be independent, microservices is presented as an approach promising approach for the development of innovative industrial applications [9], by offering high availability [10], improving availability and reliability as they can be used as gateways or load balancers

[11]. This microservices approach is being used for the design of enterprise applications [12], as an architecture for video game development [13], for development of route planning systems for unmanned aerial vehicles (UAV) [14], in the Internet of Things (IoT) scenario [15], Cattle-Health-Monitoring System using IOT devices in real time [16], microservices is also being used for the development of software applications related to the medical management of patients in the health sector, use of cloud-based healthcare-related data management [17], for the development of drug search systems [18], collect clinical data in the cloud [19] and integrate predictive algorithms [20].

In the project "Development of eHealth applications based on microservices in a Cloud architecture" [21], they present a proposal to strengthen the Electronic Medical Record (EMR) of Panama and the available information systems since they present a low availability in the access and processing of their data, thus limiting clinical-medical decision making. They developed two information systems with software architectures using the cloud computing paradigm (SaaS) and oriented to the implementation of microservices. To evaluate the efficiency of the software architectures (3 and 5 layers) of the information systems developed with this approach, performance tests and stress tests were used to obtain quantitative values and measure the overall performance of the application and its interaction with end users.

In [22], the architecture of a community health healthcare management system using IoT and microservices technologies is described, demonstrating efficiency in data collection and transfer in a simulation with 1000 requests, reducing traffic by 68% and processing time from 18.3 seconds to 7.3 seconds. Also in [23], in order to address the deployment of a suitable and efficient architecture for a Remote Healthcare Monitoring Systems (RHMS) propose the microservices approach to enable its rapid extensibility, maintainability, flexibility, in addition to fault isolation and security of healthcare data.

In this research work, a software architecture based on the Microservices approach has been designed to enable the availability of DHR. We started with the architecture design process by identifying the functional requirements and consolidating the following relevant quality attributes: functionality, availability, security, maintainability, and scalability. Using the decomposition into nouns it was determined that the software architecture to be designed should be composed of 4 microservices, Patient, Dental Health Record, Odontogram and Dental Service Provider, each microservice implements its independent database, they perform a secure communication between microservices and clients through an API Gateway of HTTP resources and a Token.

To evaluate the designed software architecture a prototype was developed that implements each of the components, the architecture was deployed in containers using the Microsoft Azure App Service with predefined features for each microservice and the API Gateway, each microservice is deployed and scales independently allowing them to be easy to change and maintain [24]. On this prototype black box tests were performed to evaluate the attribute of functionality and security, and tests to evaluate the scalability, managing to determine that the software architecture meets the requirements identified. Availability and Performance has been evaluated using load tests determining that with a single instance of the DHR Microservice up to 21 DHR per second can be available with 100% availability, and if the demand of requests increases the architecture scales horizontally automatically.

II. MATERIAL

Microsoft Azure [25] cloud service has been used to deploy the proposed architecture and test our implementation, four containers have been configured whose characteristics are shown in Table 1.

Table 1. Container characteristics to implement each microservice

Feature	Value
Name of service	App Service
Processor Speed	1 Core 3.7 Ghz.
RAM memory	1.75 GB.
Storage	50 Gb
Region	South Central USS.
Operating System	Linux
Libraries	.Net Core 3.1
Database Management System	SQL Server

The JMeter program was used to perform the load tests.

III. METHOD

A. DRIVERS OF QUALITY ATTRIBUTES

For the design of the Software Architecture that allows the availability of DHR, the quality attributes were identified, in Table 2 we can see the relevant quality drivers based on the terminology of the ISO/IEC25010 standard.

Table 2. Architecture drivers for HCO availability

Driver	Justification
Functionality	The architecture must allow the management of DHR, odontograms, patient and dental service providers according to formats established by the Peruvian Dental Association.
Security	Access to patient data, DHR and odontogram must use authentication and authorization mechanisms.
Maintainability	The architecture must support modifications and allow the addition of new functionalities with a low impact on the operation of the system.
Scalability	The architecture must allow to increase its processing capacity when DHR demand increases.
Performance	The availability of a patient's DHR must be ensured with optimal timing at a given time.
Availability	The architecture must allow the availability of DHRs and odontograms for each patient.

B. SOFTWARE ARCHITECTURE DESIGN

The microservices were identified using the decomposition based on nouns, respecting the Single Responsibility Principle (SRP) of Martin, Robert C. [26]. Table 3 presents the identified microservices grouping functionalities related to a particular entity (noun).

Table 3. Architecture drivers for HCO availability

Entity	Tasks
Dental Service Provider	Register login data Update profile data Update access password Authenticate and obtain token Retrieve access data
Patient	Register patient data Update patient data Obtain patient data by their identity document
Dental Health Record	Register Dental History Update Dental History Obtain DHR by patient's identity document Obtain DHR by its identification code
Odontogram	Register Odontogram Update Odontogram Consult Odontogram by DHR identifier Obtain Dental Treatments

In order to be able to make requests to each of the identified microservices these have a public endpoint (endpoint) or final routes of the resource that can be accessed by making a request using the Representational State Transfer protocol (HTTP REST), using the endpoints and the HTTP REST protocol an API has been built for each microservice. Table 4 shows the microservices with the final routes of the resource and the type of request.

Table 4. Microservices and tasks of the proposed model

Microservice	Tasks	Type of action	Final Routes
Dental Service Provider	Register access data	POST	/ ProviderDentalService
	Update profile data	PUT	/ ProviderDentalService /{id}
	Update access password	PUT	/ProviderDentalService / ChangePassword/{id}
	Authenticate and obtain token	POST	/Security/Login
Patient	Record patient data	POST	/Patient
	Update patient data	PUT	/Patient /{id}
	Obtain patient data by their identity document	GET	/Patient/getByDoc/{document}
Dental Health Record	Record Dental History	POST	/DentalRecord
	Update Dental History	PUT	/DentalRecord/{id}
	Obtain DHR by patient ID	GET	/DentalRecord/getByPatientDoc/{document}
	Obtain DHR by its identification code	GET	/DentalRecord/ id}
Odontogram	Register Odontogram	POST	/Odontogram
	Update Odontogram	PUT	/Odontogram /{id}
	Query Odontogram by DHR identifier	GET	/Odontogram/getByDentalRecordId/{id}
	Obtain dental treatment	GET	/Treatment /getByName/{name}

The HTTP status codes returned by the APIs each time a request is made will be those provided by the RFC 2616 standard.

The communication between the clients and the microservices will be done through an API Gateway, the use of an API Gateway has been chosen since it allows encapsulating the internal structure of the software architecture, decoupling the clients from the microservices, keeping the architecture secure since not all the microservices are exposed [27]. The secure communication between the microservices, the API Gateway and the client is performed using authentication tokens, the token is a security mechanism during the user session [28], the microservices approach can facilitate service encryption and key management for authentication and authorization using secure communication protocols [29].

The tokens are stored in the client, there is no state information allowing the architecture to be fully scalable and avoiding Cross-Site Request Forgery (CSRF) attacks. Authentication will be worked in the Dental Service Provider Microservice. The software architecture client will send and receive response and request requests in JavaScript Object Notation (JSON) format. Each microservice will have responsibility over its database, the microservice will be the only component of the software architecture that performs insert, update and select queries on the database to expose them through its REST API.

Fig. 1 shows the physical architecture of the proposed model and Table 5 describes each of the elements that compose it, considering that each logical component will be inside a container, thus allowing automatic scalability.

C. SOFTWARE ARCHITECTURE IMPLEMENTATION

A prototype implementation of each of the components that make up the proposed software architecture has been developed. Fig. 2 shows the physical software architecture taking into account the programming languages, libraries, frameworks, database management system used for the implementation of the prototype.

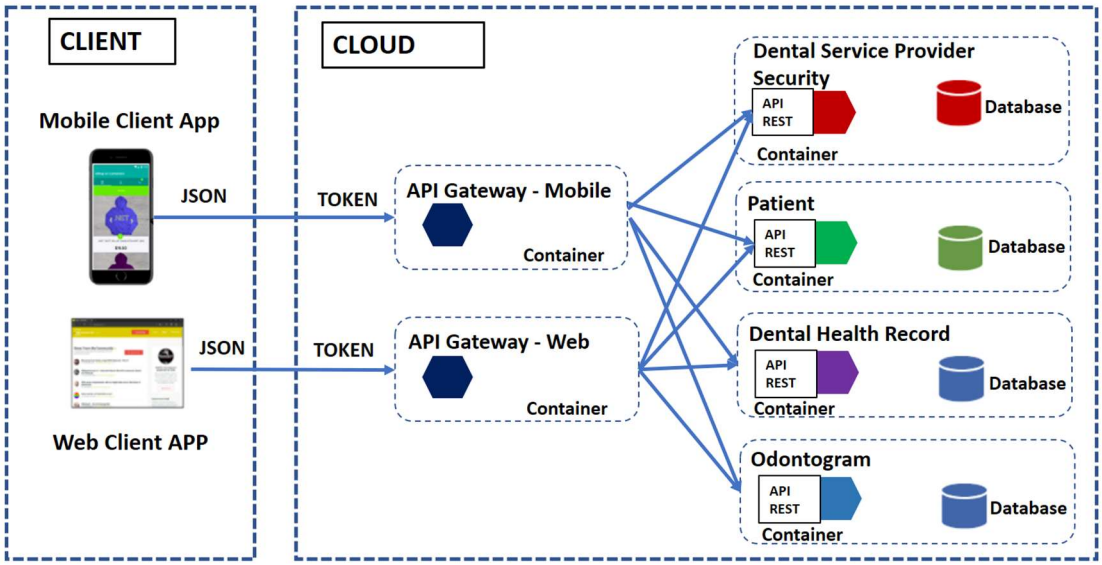

Figure 1. Physical architecture of the microservices composition model

Table 5. Elements of software architecture

Element	Responsibility	Properties
API Gateway	The gateway provides a single point of connection or URL for client applications, then internally maps requests to microservices. In this API Gateway has load balancer functionality.	HTTP Requests
Client	Clients are consumers of the resources provided by a microservice, they must authenticate themselves to generate a communication token, this token must be sent in each of their Response and Request requests to the API Gateway.	Response and request requests in JSON format
Token	The client authenticates to the architecture through the Dental Service Provider's access account. If the data is correct, the architecture returns a unique token. Each HTTP request made by the client is accompanied by a token.	Software Component
Microservice	Independent service that communicates through a REST API and performs specific functionalities to satisfy the functional requirements of the business.	REST protocol for API calls
Database	The microservice will be the only component of the software architecture that performs insert, update and select queries on the database to expose them through its REST API.	Database Management System (DBMS)

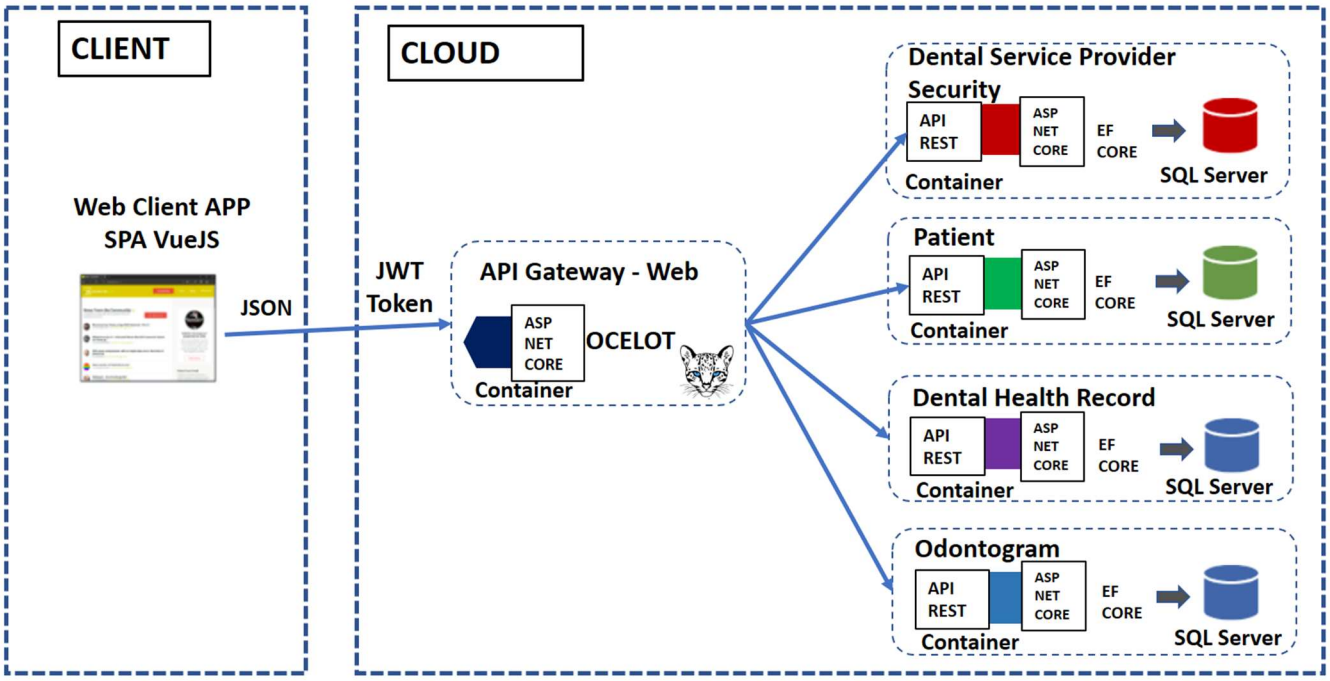

Figure 2. Physical architecture of the prototype implementation of the software architecture

The prototype has been implemented to evaluate the software architecture considering only necessary to create a web client and a single gateway API Gateway for communication with the microservices and the web client.

Table 6 describes each of the elements that make up the prototype. The frameworks, libraries and other technologies for implementing the architecture have been selected taking into account the software development experience of the authors of this research. The SQL Server relational database management system has been selected taking into account the research [30], where they determined that a Relational Database Management System (RDBMS) shows better benefits for storing medical records.

Table 6. Elements of the software architecture prototype

Element	Responsibility	Properties
API Gateway Web	The gateway provides a single point of connection or URL for client applications to make requests to services. This gateway also performs the task of load balancer.	Framework ASP Net Core 3.1 Library Ocelot
Web Client SPA VueJS	In order to test the proposed software architecture by means of request requests, a Single Page Application (SPA) web client has been developed.	VueJS Framework JSON request and JSON response requests using Axios library Vuetify library
Token	The client is authenticated in the architecture through the Dental Service Provider's access account, if the data is correct the architecture returns a unique token using the JWT library, after authentication each HTTP request must contain the token in its header.	JSON Web Token (JWT)
Microservice	Each microservice handles information according to the functional requirements of the business, exposing these functionalities through an HTTP resource API.	Framework ASP Net Core 3.1 Framework Entity Core 3.1
Database	Each microservice manages its own information, stores it and manages it in its own database.	Microsoft SQL Server

Container	Each microservice and the API Gateway will be deployed in different containers, this allows for independent scalability of each Microservice by applying scalability policies during periods of high demand.	Microsoft Azure – App Service 1 Core 3.7 GHz 1.75Gb RAM 50GB Storage South Central USS Region

IV. RESULTS

For the design of the software architecture proposed in this research, different decisions have been taken with the objective of satisfying the identified quality attribute drivers, tests have been performed to validate the software architecture according to each attribute.

A. FUNCTIONALITY

To validate the software architecture according to the quality attribute of Functionality and determine that the software architecture meets the identified requirements, 3 functional tests have been performed on the designed prototype, to evaluate the management of Dental Service Providers, Patients, DHR and Odontograms taking into account the established formats. It has been determined that the software architecture meets the identified requirements.

B. SECURITY

To validate the software architecture according to the Security quality attribute and determine that the software architecture complies with the identified authentication and authorization requirements, 2 tests have been performed, one of them allows to validate the access and token generation and the other one the authorization to consult a DHR and odontogram. After performing the black box tests on the prototype designed and implemented in this research, it has been determined that the software architecture meets the identified requirements.

C. MAINTAINABILITY

The proposed architecture has been designed under the Microservices approach thus allowing low coupling, each microservice is deployed independently resulting in easier frequent updates affecting less the rest of the system and enabling shorter delivery cycles.

D. ESCALABILITY

Scalability tests have been performed to determine if the architecture increases its processing capacity when necessary to meet the DHR demand. In Table 7 we can see that with 3 instances of the DHR microservice it can support up to 63 requests per second with 100% availability and meeting the established time indicator to answer a DHR.

Table 7. Availability result with 3 instances of the DHR microservice

Requests/Second	Average time (ms)	Availability (%)
45	1099	100.00%
50	1021	100.00%
55	1074	100.00%
60	1042	100.00%
62	1166	100.00%
63	1199	100.00%
65	2632	**98.33%**
70	**4638**	**92.05%**

E. PERFORMANCE

Table 8 shows that the architecture can respond to up to 21 requests in a given second with a single instance, thus meeting the acceptance indicator that indicates that the architecture should allow the availability of up to 20 DHR in 1 second.

Table 8. Response time for obtaining a DHR

Requests/Second	Time response (ms)
2	849
5	748
10	696
15	686
20	680
21	662
22	4393
23	13765

F. AVAILABILITY

Table 9 shows that the architecture can have up to 21 DHRs with a single instance of the Dental Heatlh Record microservice without any margin of error, and from 22 DHRs onwards the requests would be resolved incorrectly, but if 100% availability is to be maintained, scalability rules would be applied to automatically increase in the number of instances of the microservice.

Table 9. Percentage of availability of a DHR

Requests/Second	Availability (%)
2	100.00%
5	100.00%
10	100.00%
15	100.00%
20	100.00%
21	100.00%
22	99.84%

V. CONCLUSIONS

The quality attributes for the design of the software architecture have been identified taking into account Peruvian regulations, background information in the literature on medical records systems, formats established by the Peruvian Dental Association, and a survey of local dental surgeon specialists, consolidating the following relevant quality attributes: functionality, availability, security, maintainability and scalability.

The Microservices approach has been used for the design of the proposed architecture thus allowing the functionality, availability, maintainability and scalability independent of each identified microservice.

Applying the use case diagram and the decomposition into nouns it was determined that the software architecture to be designed should be composed of 4 Microservices, Patient, Dental Health Record, Odontogram and Dental Service Provider. Each microservice implements its independent database, secure communication between the microservices and the clients is performed by means of an API Gateway of HTTP resources and an authentication and authorization token.

The implementation of the software architecture designed in this research can be done independently for each microservice and considering the technological knowledge of the development team to implement each component.

In order to validate the software architecture according to the quality attribute Functionality and Security, black box tests have been performed on the developed prototype, determining that the software architecture complies with the identified requirements.

In order to validate the software architecture according to the quality attribute Scalability, tests have been performed on the developed prototype, and it has been determined that the designed software architecture can automatically scale horizontally when there is an increase in the demand of requests.

The Availability attribute was evaluated using load tests on the developed prototype that implements the designed Software architecture, obtaining as a result that with only one instance of DHR microservice up to 21 DHR can be accessed in a given second with an availability of 100%, taking into account the scalability test it is determined that with 3 instances of the Dental Health Record microservice up to 63 DHR can be accessed with an availability of 100%.

References

[1] Congress of the Republic of Peru, *Law 30024*. Peru, 2013.

[2] C. H. Wu, R. K. Chiu, H. M. Yeh, and D. W. Wang, "Implementation of a cloud-based electronic medical record exchange system in compliance with the integrating healthcare enterprise's cross-enterprise document sharing integration profile," *Int. J. Med. Inform.*, vol. 107, pp. 30–39, 2017. https://doi.org/10.1016/j.ijmedinf.2017.09.001.

[3] J. Haskew *et al.*, "Implementation of a cloud-based electronic medical record for maternal and child health in rural Kenya," *Int. J. Med. Inform.*, vol. 84, no. 5, pp. 349–354, 2015, https://doi.org/10.1016/j.ijmedinf.2015.01.005.

[4] Z. Wu, S. Xuan, J. Xie, C. Lin, and C. Lu, "How to ensure the confidentiality of electronic medical records on the cloud: A technical perspective," *Comput. Biol. Med.*, vol. 147, p. 105726, 2022. https://doi.org/10.1016/j.compbiomed.2022.105726.

[5] A. Acharya, N. Shimpi, A. Mahnke, R. Mathias, and Z. Ye, "Medical care providers' perspectives on dental information needs in electronic health records," *J. Am. Dent. Assoc.*, vol. 148, no. 5, pp. 328–337, 2017. https://doi.org/10.1016/j.adaj.2017.01.026.

[6] O. Tokede, R. B. Ramoni, M. Patton, J. D. Da Silva, and E. Kalenderian, "Clinical documentation of dental care in an era of electronic health record use," *J. Evid. Based Dent. Pract.*, vol. 16, no. 3, pp. 154–160, 2016. https://doi.org/10.1016/j.jebdp.2016.07.001.

[7] N. Niknejad, W. Ismail, I. Ghani, B. Nazari, M. Bahari, and A. R. B. C. Hussin, "Understanding Service-Oriented Architecture (SOA): A systematic literature review and directions for further investigation," *Inf. Syst.*, vol. 91, p. 101491, 2020. https://doi.org/10.1016/j.is.2020.101491.

[8] S. Li *et al.*, "Understanding and addressing quality attributes of microservices architecture: A systematic literature review," *Inf. Softw. Technol.*, vol. 131, p. 106449, 2021. https://doi.org/10.1016/j.infsof.2020.106449.

[9] F. Siqueira and J. G. Davis, "Service computing for industry 4.0: State of the art, challenges, and research opportunities," *ACM Comput. Surv.*, vol. 54, no. 9, 2021. https://doi.org/10.1145/3478680.

[10] L. Abdollahi Vayghan, M. A. Saied, M. Toeroe, and F. Khendek, "Microservice based architecture: Towards high-availability for stateful applications with Kubernetes," *Proceedings of the 2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS)*, 2019, pp. 176–185. https://doi.org/10.1109/QRS.2019.00034.

[11] S. R. Boyapati and C. Szabo, "Self-adaptation in Microservice Architectures: A Case Study," *Proceedings of the 2022 26th International Conference on Engineering of Complex Computer Systems (ICECCS)*, 2022, pp. 42–51. https://doi.org/10.1109/ICECCS54210.2022.00014.

[12] R. Kharbuja, "Designing a Business Platform using Microservices," Technische Universität München, 2016.

[13] A. A. Yunanto, F. F. Hardiansyah, A. A. Anantha Putra, M. B. Afridian Rasyid, and S. Arifiani, "Development of sandbox components with microservices architecture and design patterns in games," *Procedia Comput. Sci.*, vol. 197, pp. 354–361, 2022. https://doi.org/10.1016/j.procs.2021.12.150.

[14] L. Matlekovic, F. Juric, and P. Schneider-Kamp, "Microservices for autonomous UAV inspection with UAV simulation as a service," *Simul. Model. Pract. Theory*, vol. 119, p. 102548, 2022. https://doi.org/10.1016/j.simpat.2022.102548.

[15] S. Aydin and M. Nafiz Aydin, "Design and implementation of a smart beehive and its monitoring system using microservices in the context of IoT and open data," *Comput. Electron. Agric.*, vol. 196, p. 106897, 2022. https://doi.org/10.1016/j.compag.2022.106897.

[16] I. Shabani, T. Biba, and B. Çiço, "Design of a cattle-health-monitoring system using microservices and IoT devices," *Computers*, vol. 11, no. 5, 2022. https://doi.org/10.3390/computers11050079.

[17] C. Esposito, A. Castiglione, C. A. Tudorica, and F. Pop, "Security and privacy for cloud-based data management in the health network service chain: A microservice approach," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 102–108, 2017. https://doi.org/10.1109/MCOM.2017.1700089.

[18] J. Sadek, D. Craig, and M. Trenell, "Design and implementation of medical searching system based on microservices and serverless architectures," *Procedia Comput. Sci.*, vol. 196, pp. 615–622, 2021. https://doi.org/10.1016/j.procs.2021.12.056.

[19] A. Garcés-Jiménez *et al.*, "Medical prognosis of infectious diseases in nursing homes by applying machine learning on clinical data collected in cloud microservices," *Int. J. Environ. Res. Public Health*, vol. 18, no. 24, pp. 1–16, 2021. https://doi.org/10.3390/ijerph182413278.

[20] E. Nageba, M. Hilka, R. Gozlan, J. Dubiel, C. Baudoin, and C. Daniel, "Microservices-Based Architecture to Support the Adaptive Records-Trial," *Stud. Health Technol. Inform.*, vol. 294, pp. 283–284, 2022. https://doi.org/10.3233/SHTI220458.

[21] H. Calderón-Gómez *et al.*, "Development of eHealth applications-based on microservices in a cloud architecture," *RISTI – Iber. J. Inf. Syst. Technol.*, no. December, p. 14, 2019.

[22] R. Hill, D. Shadija, and M. Rezai, "Enabling community health care with microservices," *Proceedings of the 15th IEEE International Symposium on Parallel and Distributed Processing with Applications and 16th IEEE International Conference on Ubiquitous Computing and Communications, ISPA/IUCC 2017*, 2017, pp. 1444–1450. https://doi.org/10.1109/ISPA/IUCC.2017.00220.

[23] M. Ianculescu and A. Alexandru, "Microservices – A catalyzer for better managing healthcare data empowerment," *Stud. Informatics Control*, vol. 29, no. 2, pp. 231–242, 2020. https://doi.org/10.24846/v29i2y202008.

[24] N. Santos and A. Rito Silva, "A complexity metric for microservices architecture migration," *Proceedings of the 2020 IEEE International Conference on Software Architecture (ICSA)*, 2020, pp. 169–178. https://doi.org/10.1109/ICSA47634.2020.00024.

[25] Microsoft, "App Service Microsoft Azure." [Online]. Available at: https://docs.microsoft.com/es-es/azure/app-service/ (accessed Jan. 20, 2022).

[26] R. C. Martin, J. M. Rabaey, A. P. Chandrakasan, and B. Nikolić, *Agile Software Development: Principles, Patterns, and Practices*. University of California, 2003.

[27] N. Mateus-Coelho, M. Cruz-Cunha, and L. G. Ferreira, "Security in microservices architectures," *Procedia Comput. Sci.*, vol. 181, pp. 1225–1236, 2021. https://doi.org/10.1016/j.procs.2021.01.320.

[28] M. Krämer, S. Frese, and A. Kuijper, "Implementing secure applications in smart city clouds using microservices," *Futur. Gener. Comput. Syst.*, vol. 99, pp. 308–320, 2019. https://doi.org/10.1016/j.future.2019.04.042.

[29] R. S. de O. Júnior, R. C. A. da Silva, M. S. Santos, D. W. Albuquerque, H. O. Almeida, and D. F. S. Santos, "An extensible and secure architecture based on microservices," *Proceedings of the 2022 IEEE International Conference on Consumer Electronics (ICCE)*, 2022, pp. 1–2. https://doi.org/10.1109/ICCE53296.2022.9730757.

[30] J. E. Díaz Montejo, D. F. Bellon Cely, and J. S. González Sanabria, "A comparative study between temporary databases and relational databases applied to Electronic Health Records," *Rev. Ing. USBMed*, vol. 6, no. 1, pp. 46–53, 2015. https://doi.org/10.21500/20275846.1723.

***ARCILA-DIAZ JUAN, a** Master in Systems Engineering, Universidad Nacional Pedro Ruiz Gallo, Peru. He works as a software project leader and software development teacher. His research areas are Information Systems, Software, Intelligence Artificial and Computer Vision.*

***VALDIVIA CARLOS, a** Master in Systems Engineering. He works as a Professor of the School of Computer Engineering, Universidad Nacional Pedro Ruiz Gallo, Peru. His research areas are Information Systems, Software, Intelligence Artificial and Computer Vision.*