International Journal of Engineering Sciences 2024 16(1) 15 - 30

INTERNATIONAL JOURNAL
OF
ENGINEERING SCIENCES

Design of an application architecture using infrastructure as code for the technological waste management system of the Universidad Nacional Federico Villarreal

Santos Sotelo Antaurco[1], Julio Sotomayor Abarca Antaurco[2], Daniel Alejandro Yucra Sotomayor[3] and Yeltsin Sotelo[4]

[1,2,3]Universidad Nacional Federico Villarreal, Lima Perú
[4]Universidad Nacional de Ingeniería

(Received 2 March 2024; Accepted 22 August 2024)

https://doi.org/10.36224/ijes.160103

Abstract

This research project presents the problem of choosing or defining a software architecture for the technological waste management system of the Universidad Nacional Federico Villareal and shows how it is possible to define the process of Architecture Design based on Infrastructure as code that is capable of being scalable, maintainable, secure and interoperable as principles for the Design.

In addition, it presents the Design of an application architecture using infrastructure as code - IaC, in the cloud and is able to support the capabilities of the components and modules of the Technological Waste Management System of the Universidad Nacional Federico Villarreal.

This project also implies, to determine the favourable acceptance to develop the Application Architecture Design using infrastructure as code - IaC.

From the technological point of view, the most critical dimensions are also presented to take into consideration the architecture design based on the functional modules of the Technological Waste Management System of the Universidad Nacional Federico Villarreal.

At the end of the work the presentation of results and discussion based on the instrument used is shown, to determine in the final part the fulfillment of the objective describing the conclusions and recommendations.

Keywords: Application architecture, Infrastructure as code, Technological waste management Technological waste; waste management; waste monitoring; monitoring system; software architecture; environmental contamination.

*Corresponding author
Email address: dyucra@unfv.edu.pe (Daniel Alejandro Yucra Sotomayor)
ISSN 0976 – 6693.

1. Introduction

According to the problem statement, the management of technological waste is receiving increasing attention on global, national, regional, and local levels. With the rapid development and changes in technology, more and more obsolete electronic devices are being discarded. These wastes contain toxic substances such as lead, mercury, and cyanide, and if not properly disposed of, they can be harmful to both the environment and human health.

The Growing Challenge of Technological Waste Management

Proper treatment of technological waste is an ever-growing challenge in modern society. As technology advances and electronic equipment ages rapidly, the generation of technological waste has reached an alarming level. In this context, the Universidad Nacional Federico Villarreal faces the need to improve its technical waste management system.

The Need for Effective Policies and Programs

It is crucial to implement effective policies and programs for managing technical waste, incorporating reuse, recycling, and safe disposal. Additionally, it is essential to promote awareness and reflection on the obligation for responsible management of technological waste and to initiate sustainable practices in technological production and consumption (Parrales, 2020).

Challenges in Designing Application Architectures Using IaC

To make appropriate decisions, it is inevitable to control, measure, and manage through technological platforms, requiring robust architectures for these applications. However, many issues can arise in architectural design, such as scalability, maintainability, performance, security, cost, and interoperability (Quintero, 2020).

Implementing IaC for Technological Waste Management

Designing an application architecture using infrastructure as code (IaC) for the management of technological waste at the Universidad Nacional Federico Villarreal can present several challenges. The complexity of configuring and deploying the infrastructure is one of the main issues. However, tools and techniques known as IaC can automate the implementation and configuration of applications, although this can be challenging for those unfamiliar with these tools.

Costs and Resources

Using IaC in designing an application architecture may require significant time and resources. The university must consider these costs when deciding to use this architecture.

Summary of Architectural Design Using IaC

Designing an application architecture using IaC for technological waste management can exhibit some difficulties, including the complexity of configuration and implementation processes, the need to ensure system scalability and security, and implementation costs.

Background and Justification

Regarding the design of an application architecture using IaC for technological waste management at the Universidad Nacional Federico Villarreal, no specific information was found. However, IaC has been applied to numerous cloud services and applications, making the implementation process more efficient and error-free (Quintero, 2020).

Importance of the Project

The significance of this project lies in its potential to enhance the operational efficiency of the university's system, optimizing the management of technological waste effectively. Additionally, using IaC will enable testing scenarios and software evaluation, aiding the continuous development of the management system.

Theoretical Framework

The theoretical framework for this research includes concepts related to technological waste management and IaC. Technological waste management involves the responsible and safe handling of electronic and electrical waste. On the other hand, IaC is a practice that uses tools and techniques to automate the implementation and configuration of applications.

Key Concepts

The theoretical framework for designing an application architecture using IaC for the technological waste management system is based on concepts such as IaC, DevOps, cloud computing, Service-Oriented Architecture (SOA), and microservices (Ramírez, 2013).

Objectives

The objectives of this study focus on designing an application architecture using IaC to improve the technological waste management system at the Universidad Nacional Federico Villarreal. This should allow for system scalability, interoperability, security, and maintainability.

2. Materials and Method

2.1 Method (Universe and sample, instrument and procedure)

The research will focus on the paradigm of an application architecture using infrastructure as code, will use the non-experimental and applied typographical design with an agile and flexible methodology used in the creation of numerous projects and that allows managing software development, of a descriptive and analytical fundamental type, and will aim to systematically describe the generation and management of technological waste, because it seeks to understand the facts from the reference point of the actors, allowing to know the area of study and the reality of pollution and waste of technological devices.

Using an agile methodology built on Terraform, a collaborative value creation framework that provides sufficient structure to allow individuals and teams to work according to their preferred methods while

incorporating best practices to meet their unique needs, a tool that allows automating the creation of infrastructures following the concept "Infrastructure as code". (Quintero, 2020).

To realize this contribution, firstly, process modeling and functional specification of tools for software architecture modeling and code generation using a microservices approach, where user (Architect/Developer) actions are detailed.

Also, to establish unbiased and reliable solutions that point in the direction of solving the problem, after which a solution is applied.

It involves two modes of research, bibliographic and field, because it happens where the problem actually arises, in the field. Members of the university have provided direct support and participation for this project. The deans and administrative staff will participate in the research by providing their standards and answering the relevant surveys. Bibliographic, since it has been used to investigate in diverse sources that contain scientific data related to the subject in order to concentrate on the topic, which has been consigned in the theoretical framework as well as in the foundation, which constitute the main components of the completed work. The scope of the research, both in terms of space and time, is determined by the study.

Geographical Coverage: The study will be carried out in the Peruvian capital of Lima.

Temporal Scope: According to the structure of the project, this research will be carried out from April to December 2023.

Very important factors are taken into account according to the methodology, among them:

Population: technical and professional personnel with a profile in Software Architecture, Technological Architecture and Cloud Architect of the Lima Region are taken into account.

Sample: this portion of the universe characterized in an identical way will be used to establish the volume of the sample according to the formula:

Where:

Population(N)	300
Security	95%
probability of success (p)	5%
maximum permissible error (d)	0.03

Para una seguridad del 95% se tiene un nivel de confianza (Z) = 1.96

$$n_{opt.} = \frac{300x\ 1.96^2\ x\ 0.05\ x\ 0.95}{0.03^2\ x\ 299 + 1.96^2\ x\ 0.05\ x\ 0.95}$$

$$n_{opt.} = 30.1 \approx 30\ people$$

Sample No.: 30 selected users

2.2 Instrument

Through the research report, the tool and the methods used for its use are disclosed to other team members, which facilitates the identification of the pattern used to record the observed facts and confirm the accuracy of the observations made (Chowdhury, et al, 2019; Bandalos, 2018).

In the process of the research, data collection instrument of technological waste will be applied, such as the survey, aimed at managers and administrative staff of the national university Federico Villarreal. For which the following will be carried out.

- Data Collection Sheet (survey)

- Data Tabulation

- Histogram chart

- Process Diagram

- Systematic Review

- Documentary analysis

- Sampling

2.3 Procedure

Once the data collection through the survey has been achieved, it will be processed through the SPSS statistical system, quantifying the required results through statistical tables and graphs (Castells, 2009), for which the following will be done:

- For the data collection procedure, the survey will be applied.

- The accumulation of information in each area will be verified and classified according to its validity.

- Systematization of the data and information process

- Review of the data obtained.

- Analysis and interpretation of data

3. Results

3.1 Representation of results by process and Representation of results by question

The following is the consolidated result for the main question on developing the Application Architecture Design.

Consolidated result for the main question

Do you agree with using the Application Architecture Design, using the infrastructure as code to improve the Technological Waste Management System of the Universidad Nacional Federico Villarreal?

Scale	Total	%
Strongly agree	3	10
Agree	22	73.3
Neither Agree Nor Disagree	3	10
Somewhat Disagree	1	3.3
Disagree	1	3.3
Total	**30**	**100.0**

Source: Own elaboration

Interpretation: According to table 03, it can be shown that 22 participants agree to develop the Application Architecture Design.

The results by process and associated specific processes per scale are presented below:

3.2 Development

Development Methodology

Development Methodology for Infrastructure-as-Code Architecture Design - IaC

This strategy has been established for the development of the Architecture Design based on Infrastructure as Code according to the following stages:

(a) Requirements Definition:

Understand the infrastructure requirements and translate them into declarative code. This involves defining what resources are required, how they should be configured and their interdependencies.

(b) Tool Selection:

Choosing the right IaC tools based on the team's requirements and preferences. Terraform is known for its multi-cloud support, Ansible for its focus on configuration, and CloudFormation for its native integration with AWS, for example.

(c) Coding:

Write code that describes the desired infrastructure. Use clean coding practices, proper documentation, and testing to ensure code quality.

(d) Architecture Design:

Define the conditions, dimensions, requirements, scalability, security and maintainability of the software Architecture.

(e) Architecture Implementation

Apply the code to create and update the infrastructure. Automate this process to achieve a consistent and repeatable implementation in different environments.

(f) Architecture Validation and Testing

Perform extensive testing to ensure that the infrastructure meets the requirements. This may include integration testing, performance testing, and security validations.

(g) Continuous Solution Monitoring

Implement monitoring tools to continuously track the status of the infrastructure. This facilitates early detection of problems and implementation of quick fixes.

3.1 Analysis and design of the architecture

The following are the results of the analysis and design of the application architecture using infrastructure as code for the Technological Waste Management system, as detailed below:

3.2.1. Application Architecture design requirements.

In order to define the requirements for the design of the Application Architecture, it will be necessary to consider the following requirements for the process:

a) Scalability

b) Resilience

c) Architecture performance monitoring and logging

d) Regulatory compliance

e) Integration

f) Documentation

g) Functional and non-functional system requirements

h) Infrastructure security

i) Cloud architecture

j) Infrastructure as code (IaC)

k) Application Automation and Orchestration Considerations

l) High availability

3.2.2 Application architecture design characteristics.

Once the requirements have been identified it is necessary to characterize the design that make possible a more efficient and scalable implementation, the same has been identified these technical characteristics that it must support:

(a) Automation

b) Reproducibility

c) Versioning and change control

d) Flexibility and scalability

e) Portability

f) Automated documentation

g) Collaboration and standardization

h) Ease of testing and validation

i) Security and compliance

j) Simplified maintenance

3.2.3. Technology for the design of the Application Architecture.

Different tools and platforms are defined that allow the automated management of the infrastructure; therefore, it is possible to consider and evaluate the following technologies:

(a) Terraform

b) AWS cloud formation

c) Azure Resource Manager Templates

d) Google Cloud Deployment Manager.

e) Docker Compose

f) Kubernetes

Identified considerations to consider for scalability of architecture design 3.2.4.

Considerations for Infrastructure as Code (IaC) has been identified enables the adoption of practices and strategies that facilitate horizontal and vertical scalability to cope with growth and demand fluctuations, described below:

(a) Monitoring and metrics

b) Continuous deployment

c) Vertical and horizontal scalability

d) Data partitioning

e) Resource optimization

f) Modular design

g) Use of containers

h) Container orchestration

i) Load balancing

j) Automatic scaling

3.2.5. Considerations identified to consider Resilience for architecture design.

For the action of deploying the architecture and system it is necessary to have the ability of the system to withstand, recover and adapt to possible failures and disturbances, therefore, the following considerations have been identified:

(a) Error and log management

b) Capacity to respond to unexpected load

c) Documentation and contingency planning

d) Backup and recovery

e) Fault tolerance

f) Resilience testing

g) Monitoring and alerts

3.2.6. Considerations identified to consider the security of the architecture design

Since the application will work in the Cloud, this aspect refers to the measures and strategies implemented to protect data, systems and users against possible security threats and vulnerabilities, which are listed below these considerations for the system:

(a) Updates and patches

b) Cloud security

c) Protection against DDoS attacks

d) Backup and recovery

e) Layered security

f) Access control

g) Data encryption

h) Identity and authentication management

i) Monitoring and intrusion detection

j) Security testing

3.2.7. Considerations identified for considering the Modularity of the Architecture Design

Each module represents a specific functionality of the system and can be treated as an isolated entity, modularity facilitates development, maintenance and scalability, these aspects have been taken into consideration below:

a) Well-defined interface

b) Scalability

c) Facilitates collaboration

d) Unit testing

e) Abstraction of components

f) Decoupling

g) Reusability

h) Fault isolation

i) Simplified maintenance

j) Clear documentation

3.2.8. Considerations identified for automation of architecture design.

This refers to the extent to which the infrastructure provisioning, configuration and management processes are carried out automatically, without manual intervention, and the items identified are listed below:

(a) Version management and change control

b) On-demand provisioning

c) Standardization and consistency

d) Automation of operational tasks

e) Automatic resource provisioning

f) Automated configuration and management

g) Continuous deployment

h) Automatic elasticity and scalability

i) Monitoring and self-recovery

3.2.9. Considerations identified for architecture design monitoring and logging.

These are fundamental practices to ensure the correct operation, performance and security of the system, therefore, the following items have been identified for consideration:

a) Infrastructure monitoring

b) Performance metrics

c) Security monitoring

d) Error and exception logging

e) Performance analysis and optimization

f) Compliance and audits

g) Event logs

h) Alerts and notifications

i) Data visualization

j) Availability and uptime monitoring

3.3 Application Architecture Design

3.3.1 Layers

The resources to be provisioned for the Technological Waste Management System of the Federico Villarreal National University REHUSA are detailed below.

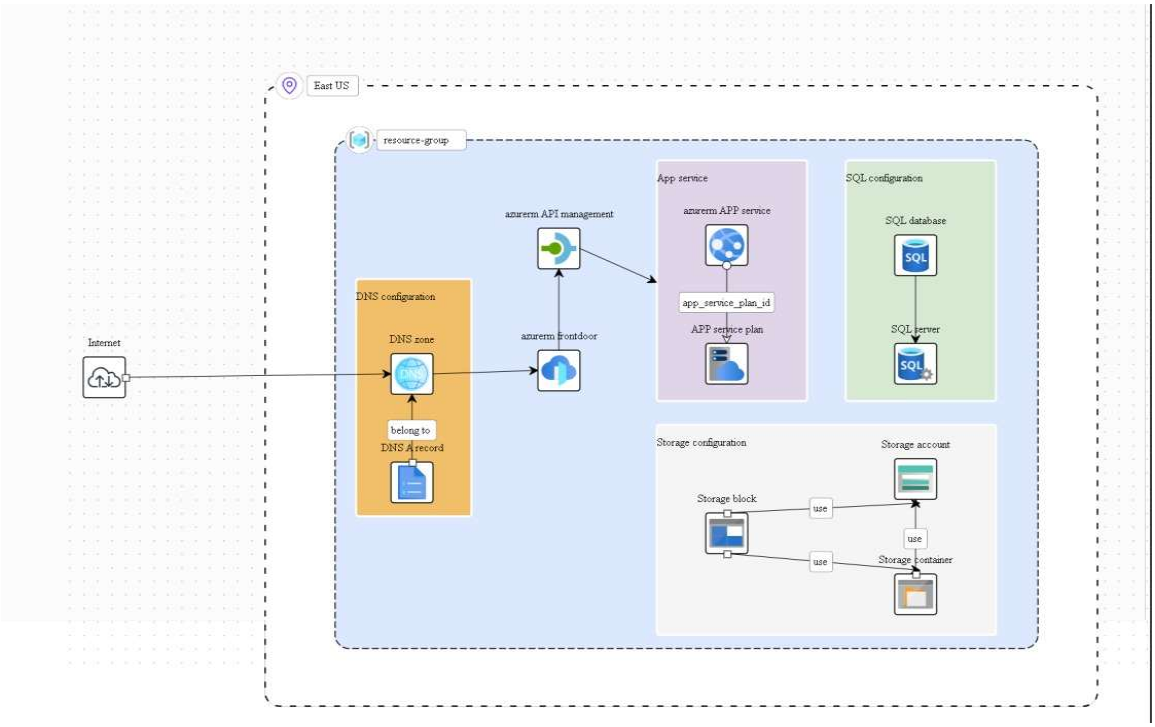

Figure 1: Design of Application Architecture at the Infrastructure as code level.

Source: Own elaboration

Where it describes:

(a) Application security layer:

DNS Zone:

A DNS Zone allows you to manage and control domain name resolution for your web application. Here, you will configure DNS records to bind your domain to the Azure resources you are using.

ApiManagement:

Azure API Management is a service that allows you to securely create, publish, manage and secure APIs. It is useful for exposing your APIs in a controlled way and managing their access.

FrontDoor:

Azure FrontDoor is a global application delivery service that provides scalability, security and high availability. It acts as an entry point for your web applications and APIs, and intelligently distributes traffic to optimize gain and availability.

(b) Application Layer:

App Service:

Azure App Service is a managed service for hosting web apps, RESTful services, and mobile apps. It allows you to easily deploy and scale your applications without worrying about the underlying infrastructure.

AppService Plan:

An App Service Plan defines the resources and features available to your applications hosted on Azure App Service. You can choose the level of scalability and performance that best suits your needs.

(c) Data Layer:

SQL Server:

SQL Server in Azure allows you to host relational databases in the cloud. You can choose from different levels of performance and capabilities depending on your application requirements.

Storage account:

A storage account in Azure provides a place to store and manage unstructured data, such as images, files, videos, etc. You can use it to back up your application and store static files.

3.3.2 Components

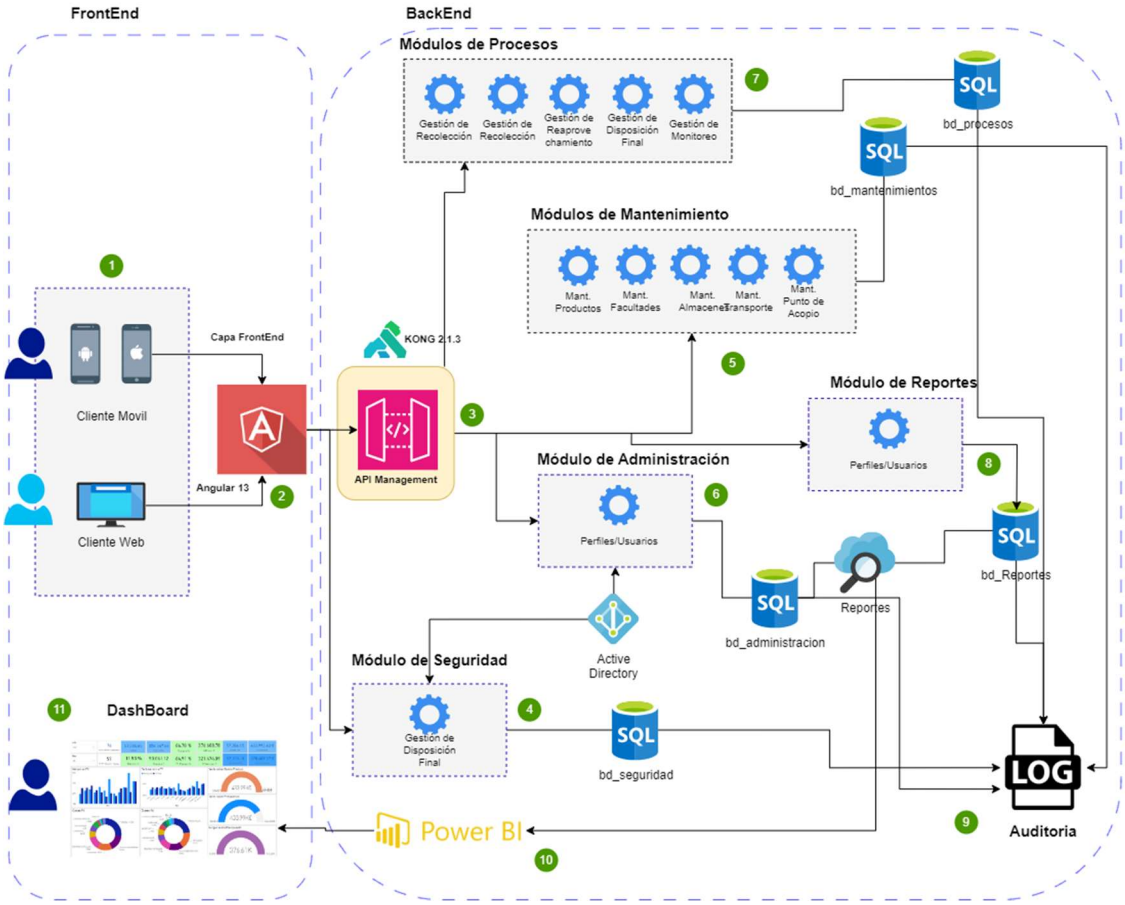

Figure 14: Architecture design at the infrastructure as code level (Source: Own elaboration)

Where it is described:

1. Corresponding to the Front-End Web and Mobile Client component 2.
2. Corresponds to part of the Frontend technology, in this case Angular 13.
3. Corresponds to the Backend component, API Management that orchestrates the solution through the components of the solution, also allows interoperability with internal or external processes and even other systems.
4. Corresponds to the Architecture Security, Active Directory is used to control access to users.
5. Corresponds to the Maintenance Module of the UNFV's Technological Waste Management System, REHUSA.
6. Corresponds to the Administration Module of the Technological Waste Management System of the UNFV, REHUSA.
7. Corresponds to the Processes Module of the UNFV's Technological Waste Management System. REHUSA.
8. Corresponds to the Reports Module of the UNFV's Technological Waste Management System, REHUSA.
9. Corresponds to the Audit component (Logs) of the UNFV's Technological Waste Management System, REHUSA.
10. Corresponds to the component of the Power BI (Business Intelligence) solution for data analysis of the system.
11. Corresponds to the visualization of the Dashboard of the Technological Waste Management System of the UNFV, REHUSA.

4. Discussion

As we have appreciated, in a general way, the valuation of the different tabulations in the presentation of results, shows us an acceptance in the Development of the Design of the Architecture of applications, which imply to define the scalability, interoperability, security and maintainability of the Software, that is to say it is "In Agreement", and it is necessary to take action in the course of the elaboration of the design of the present project of investigation.

Regarding the scalability of the Application Architecture Design, it has been considered as a necessary requirement to consider or add more modules horizontally and increase the technological capabilities at the hardware level vertically.

Regarding the interoperability scalability of the Application Architecture Design, it has been considered as a requirement for data exchange and integration with internal processes and systems.

With respect to the security of the Application Architecture Design, it has been considered as a requirement to provide the security of the Architecture Design of modules and data at internal and external level.

With respect to the maintenance of the Application Design Architecture, it has been considered as a requirement to provide modifications, updates, version control and support Architecture design changes.

With respect to the Software Architecture, the different technologies have been selected based on the requirements of the identified processes, in addition to scaling and sustainability.

As main conclusion:

The Application Architecture Design has been developed using infrastructure as code for the technological waste management system of the Universidad Nacional Federico Villarreal.

As secondary conclusions:

It has been positively determined that scalability is a critical dimension in the Application Architecture Design for the technological waste management system of the Universidad Nacional Federico Villarreal, the same will allow increasing new component modules to the system.

It has been positively determined that interoperability is a critical dimension in the Application Architecture Design, it will allow the system to exchange data and information with other systems or applications.

It has been positively determined that security is a critical dimension in the Application Architecture Design, it will allow the system to protect, certify and endorse the security of the information of the system in question.

It has been positively determined that maintenance is a critical dimension in the Application Architecture Design, the same will allow to support changes, updates and corrections of the system.

Recommendations:

Apply the Application Architecture Design using infrastructure as code for the development and implementation of the technological waste management system of the Universidad Nacional Federico Villarreal with the purpose of making it scalable, interoperable, secure and maintainable.

Continue to carry out the development and implementation of the design of the cloud architecture to scale for the technological waste management system of the UNFV.

Continue to perform the development and implementation of the architecture design to interoperate with other systems the UNFV technological waste management system.

Continue to carry out the development and implementation of the architecture design to provide security for UNFV's technological waste management system.

Continue to develop and implement the architecture design for the maintenance of UNFV's technological waste management system.

References

1. Castells, X. (2009). Reciclaje de Residuos Instriciales (Segunda ed.). Diaz de Santos.
2. Cruz, A. R. (2020). Herramienta para despliegue y gestión de plataformas en la Nube. Sevilla, España: Universidad de Sevilla.
3. Parrales, J. (2020). Manejo de desechos tecnológicos en instituciones públicas de la cabecera del Cantón. Ecuador: Jipijapa.
4. Pereira, A. M. (2020). Uso de una arquitectura basada en eventos como capa de comunicación para microservicios. Lima, Perù: PUCP.

5. Quintero, J. R. (2020). Terraform como herramienta para automatizar la creación de infraestructuras siguiendo el concepto “Infraestructura como código”. Esmeraldas, Ecuador: Pontificia Universidad Catolica del Ecuador.
6. Ramírez, K. R. (2013). Propuesta de una Arquitectura SOA para la integraciòn de sistemas en una empresa de traslado de dinero. Lima, Perù: UNMSM.