Cloud Maturity Framework

A Guideline to Assess and Modernize Cloud Computing Applications and Workloads

Carlos Diego Cavalcanti Pereira
CESAR – Recife Center for Advanced Studies and Systems
Recife, Brazil
e-mail: cdcp@cesar.org.br

Carlos Alberto Costa Filho
Valcann - Cloud Intelligence
Recife, Brazil
e-mail: calberto@valcann.com.br

Felipe Silva Ferraz
CESAR – Recife Center for Advanced Studies and Systems
Recife, Brazil
e-mail: fsf@cesar.org.br

***Abstract*—Organizations tend to implement and maintain their architecture in the cloud the same way they operate on premises, wasting the opportunity to use the benefits of cloud computing. Although the literature proposes several prescriptive models on how to modernize cloud architectures, these models are invariably too generic and do not indicate the current maturity level of the cloud architecture, nor do they technically specify what actions should be taken. This paper focuses on the proposal of a Cloud Maturity Framework to help organizations understand their current maturity level regarding the best modern practices of cloud computing architecture, by applying a set of practices on how to elicit its current maturity level and, implement and manage improvements.**

Keywords- cloud computing; application modernization; cloud modernization.

I. INTRODUCTION

Cloud computing migration projects tend to apply strategies during the migration process so that the fewest possible changes to the architecture are made in order to minimize the already natural frictions that any process of change applies [1]. In this sense, organizations keep their architectures in the cloud the same way they operated when on premises, wasting the opportunity to use the benefits of cloud computing and modern cloud architectures [2]. Cloud native or modern cloud architecture is a software approach of building, deploying, and managing modern applications in cloud computing environments [3]. Modern companies want to build highly scalable, flexible, and resilient applications that they can update quickly to meet customer demands [3].
Amazon Web Services, a leading cloud computing company, states that when carrying out modernization projects, enterprises can reduce payback periods to 6 months and reduce Total Cost of Ownership (TCO) by 64% [4]. Although the literature proposes several prescriptive models on how to modernize cloud architectures, those models are invariably too generic, focusing more on describing what to do rather than how to do it. They also do not indicate the current maturity level of the cloud architecture, nor do they technically specify how improvements should be implemented [5]. To address those opportunities, the present work proposes a maturity model for cloud computing architectures, covering a set of practices and processes on: how to assess a cloud workload summary and examine its components, how to establish a maturity baseline and a roadmap to cloud modernization, and how to deploy improvements and manage change processes.

The rest of the paper is structured as follows. In Section 2, we present the concept of maturity models. In Section 3, we present the Cloud Maturity Framework, covering the general model, workload layers, the modernization process and maturity classification. In Section 4, we discuss advances in cloud maturity models and proposed future work related to how to modernize cloud architectures. Finally, in Section 5, we present our conclusions.

II. MATURITY MODELS

A maturity model is an approach in which the effectiveness of a process or a general architecture is assessed to understand its current level in reference to a classification range [6]. Applying maturity models has become a common practice in computer science, especially in software engineering. So much that the literature presents a specification of a meta model for creating maturity models like the ISO 33001 standard [7]. Implementing a maturity model helps evaluate the achievement of process quality characteristics and the application of the results to the conduct of improvements.

Regarding cloud maturity evaluation, several models propose different approaches on how to evaluate the current state of cloud adoption, such as OACA Cloud Maturity Model [8]. Even the industry also provides several different approaches: AWS Well-Architected Framework [9], AWS Cloud Adoption Framework [10], Azure Well-Architected Framework [11], among others. Although those frameworks

 ISBN: 978-1-61208-997-3

help to identify the level of adoption from a governance and processes perspective, they do not focus on a technical level and on how broad is the application of more modern cloud technologies, such as the use of containers, functions as services, event-driven architectures, among other implementations. That is, there is the opportunity to delve into how to classify the level of sophistication of a workload in the cloud, as well as the path to be followed to make it even more modern.

III. Cloud Maturity Framework

The proposal of Cloud Maturity Framework intends to help organizations understand their current maturity level regarding the best practices of modern cloud computing architecture. It states that organizations should first list their high-value assets and analyze how they could benefit from improved scalability, security, and performance. The framework helps organizations understand how modernization can benefit business operations.

Cloud Maturity Framework goes beyond just defining generic goals of improvement processes and good architecture. It consists of a specific analysis in which it is possible to explore breadth and maturity of the cloud computing stack, integrating various related areas, and clarify the means to achieve a higher maturity in modern cloud architecture.

A. General Model

The Cloud Maturity Framework establishes a predefined structure as presented on Figure 1: a) Workload Layers; b) Modernization process; and c) Maturity classification.

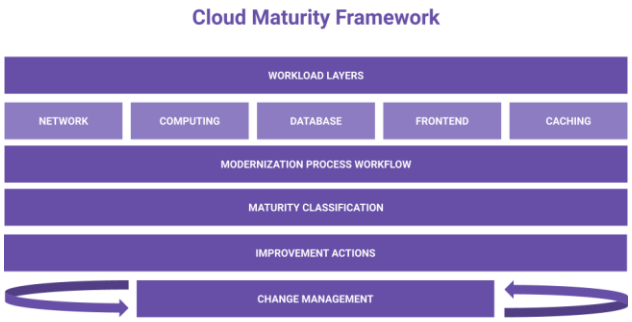

Figure 1. Cloud Maturity Framework

B. Workload Layers

One of the challenges when evaluating a cloud workload is to understand its blueprint and implications regarding integration between components [12]. Cloud computing provides access to large pools of distributed components - such as computing (servers, containers, functions as services) - for building high-quality applications [13].

Architectural patterns cover the architectural principles found in most cloud-native applications to enable the cloud application properties [14]. Service-based cloud architecture patterns target settings which components a workload may have [15]. Although the literature predicts different types of component categories, monolithic and distributed computing architectures, they usually consider the partitioning of their structure in contexts that involve: computing, data and communication [16]. In this sense, In order to ensure that these common components categories are covered in the maturity model, the framework proposes five categories of cloud components:

a) **Network Layer**: Foundation of cloud communications and networking. It establishes a set of network connections, access control lists, routing policies, and the application's boundaries.
b) **Computing Layer**: Usually related to the use of application components, such as virtual machines, containers, functions as services and others. Evaluating this layer helps to understand how it implements computing best practices.
c) **Database Layer**: The data tier, also known as the database layer, is where all of the information about user data and transactions are kept. It basically contains any data that has to be stored. The data is delivered to the computing layer for logic processing and then for rendering to the user.
d) **Front-end Layer**: Front-end layer is related to components at the edge of the application's architecture, such as Content Delivery Networks (CDN) and Object Storage.
e) **Caching Layer**: Components that process a time-consuming request, generally in memory, so that the requests can be handled faster. Caching components manage the data in memory rather than loading it directly from a database, network, or a storage volume.

C. Modernization Process

Cloud modernization is becoming more essential for migrating to public clouds [17]. Transitioning from monolithic, tightly connected architectures to modern distributed and serverless services requires a mix of high-demand technologies and expertise.

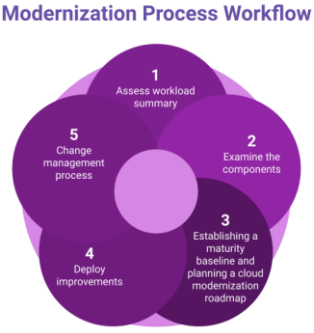

Figure 2. Modernization Process

 ISBN: 978-1-61208-997-3

Cloud Maturity Framework dictates that to perform a cloud modernization process - as presented on Figure 2 - a set of steps should be performed:

a) **Assess workload summary** - To understand the workload, the first step is to assess it and make an inventory of all components. This will make it possible to identify each component in the architecture and how it relates to each of the workload layers.

b) **Examine the components** - From the results of resource inventory, the next step will be to establish the general summary of the architecture. The purpose is to build, customize, and share detailed architecture insights of the workload based on live data retrieved from Cloud provider Application Programming Interface (APIs). The outcomes expected from this examination are architecture diagrams, query cost and usage reports (CURs), and resource classification from a functional perspective.

c) **Establishing a maturity baseline and planning a cloud modernization roadmap** - Architecture, design, and technology stack directly influence the modernization approach [12]. While it depends on the organization's desired goal, the path taken may depend on various archetypes. Achieving a simplified architecture, for example, requires a change in thinking, especially when considering options for new systems and integrations [6]. A detailed understanding of the current state of the legacy system, comparing current and future architecture, will help identify gaps and changes needed to be successfully modernized. Each workload layer will be evaluated from a maturity perspective. As the organization identifies its current maturity level, it will be possible to define which path they want to take, from simply moving one or a few levels up or even achieving excellence going forward to the higher possible maturity level.

d) **Deploy improvements** - After understanding the maturity level for each workload layer, it will be possible to define the gaps as adherent to a modern cloud application and infrastructure approach. More than simply applying changes and improvement directly to cloud provider console or CLI, for the Cloud Maturity Framework it is essential to apply infrastructure as code as a common practice because it is the essential characteristics of modern and cloud native infrastructure [3], as presented in Figure 3.

```
Resources:

 RemoveIngressRuleRole:
   Type: AWS::IAM::Role
   Properties:
     RoleName: RemoveIngressRule
     AssumeRolePolicyDocument:
       Statement:
         - Effect: Allow
           Principal:
             Service: lambda.amazonaws.com
           Action: sts:AssumeRole
     Policies:
       - PolicyName: RemoveIngressRulePolicy
         PolicyDocument:
           Version: "2012-10-17"
           Statement:
             - Effect: Allow
               Action: ['ec2:DescribeSecurityGroupRules',
'ec2:DescribeSecurityGroups', 'ec2:RevokeSecurityGroupIngress']
               Resource: '*'

 RemoveIngressRuleFunction:
   Type: AWS::Lambda::Function
   Properties:
     FunctionName: RemoveIngressRule
     Role: !GetAtt RemoveIngressRuleRole.Arn
     Runtime: python3.9
     Handler: index.lambda_handler
     Timeout: 60
     Code:
       ZipFile: |
         import json
         import boto3

         def lambda_handler(event, context):
             client = boto3.client('ec2')
             security_groups = client.describe_security_groups(
                 Filters=[
                     {
                         'Name' : 'ip-permission.cidr',
                         'Values': ['0.0.0.0/0']
                     }
                 ]
             )

             for security_group in security_groups["SecurityGroups"]:
                 security_group_rules =
client.describe_security_group_rules(
                                                 Filters=[
                                                     {
                                                         'Name' : 'group-
id',
                                                         'Values':
[security_group["GroupId"]]
                                                     }
                                                 ]
                                             )
                 for security_group_rule in
security_group_rules["SecurityGroupRules"]:
                     if("CidrIpv4" in security_group_rule.keys() and
security_group_rule["CidrIpv4"] == "0.0.0.0/0"):
                         client.revoke_security_group_ingress(
                             GroupId=security_group_rule["GroupId"],

SecurityGroupRuleIds=[security_group_rule["SecurityGroupRuleId"]]
                         )
             return {
                 'statusCode': 200,
                 'body': json.dumps('Hello from Lambda!')
             }
```

Figure 3. AWS Cloudformation improvement script to remove “Public open ports to virtual machine instances”, level 1 maturity criteria for Computing Layer

 ISBN: 978-1-61208-997-3

e) **Change management process** - Cloud IT change management processes facilitate changes to IT systems in order to minimize risks to the production environment while adhering to policies, audit, and risk controls [18]. All tests, validations, approvals, and rejections must be documented as part of the pipeline deployment.

D. Maturity Classification

The maturity classification serves to analyze the maturity level of the organization's cloud computing practices and architecture [19]. Each of the five layers previously presented are evaluated from the perspective of specific maturity criteria. Each one of the layers have a set of criteria that should be validated, classified, and have an improvement directly related to it. Table 1 presents an example of maturity criterias of Computing Layer.

After being evaluated, the organization will have a classification score, as presented in Figure 4, which shows the roadmap that the organization will take to be more adherent to what is called modern architecture in the cloud [3].

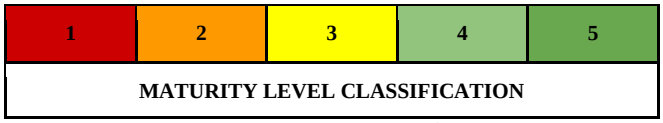

Figure 4. Maturity Level Classification

TABLE 1. MATURITY LEVEL CLASSIFICATION FOR COMPUTING LAYER

LAYER	MATURITY LEVEL	MATURITY CRITERIA
COMPUTING LAYER	MATURITY LEVEL 1	- Auto scaling disabled for virtual machine instances; - Load Balancer disabled for virtual machine instances; - Missing snapshots for block storage volumes; - Unencrypted block storage volumes; - Public open ports to virtual machine instances.
	MATURITY LEVEL 2	- Public accessible serverless function; - Block storage snapshots with public access; - Public virtual machine images; - Virtual machine instances without automation for status check failure; - Serverless functions environment variables without encryption enabled.
	MATURITY LEVEL 3	- Underutilized (<10%) container cluster; - Underutilized (< 30% capacity on average for last week) of virtual machine instances; - Automation for status check failure on virtual machines; - Out of date virtual machine images; - Pending scheduled maintenance events on virtual machines.
	MATURITY LEVEL 4	- Virtual machines are right sized; - Block storage is encrypted by default; - Block storage lifecycle management is enabled for backup; - Block storage volumes not attached to virtual machines; - Logging and alarms are configured.
	MATURITY LEVEL 5	- Virtual machine have termination protection; - Load Balancers placed in multiple availability zones; - There is no unused EIP; - Virtual machine images are private; - All resources are tagged.

IV. Towards Cloud Maturity And Future Works

A successful cloud modernization strategy starts with the business need in mind, having a tight connection to technical practices and tools [17]. As the journey to the cloud gathers pace, organizations have been looking for what is their current cloud maturity landscape so that they can accelerate cloud modernization [20].

As cloud computing has a natural entropy as a technological and incipient field, this research will continue to develop on Cloud Maturity Framework. Some initiatives were already mapped and are part of the development of the framework:

- Applying the framework in cloud modernization projects in different industries and collecting data on the implementation process, as well identifying benefits and challenges.
- Benchmarking the framework with other cloud modernization approaches.
- Open source the framework, so it can be used by both academia and industry, especially to continue the development of maturity criteria as well as developing and automating improvements for those criteria elicited.

V. Conclusions

Cloud modernization is both a technical and organizational goal, so it is important to develop well-defined strategies to ensure that both occur [12]. Both literature and industry had proposed different approaches to help organizations modernize their cloud workloads. Those approaches focus on defining what should be done to modernize, focusing on governance and processes perspective, but do not guide how to do it or what is the technical roadmap to be taken towards a modern cloud architecture, especially on a technical level.

As presented in this research, to define a roadmap it is essential to understand the current baseline and where the project is starting. Understanding the workload's current state enables the knowledge about the environment and the overall impact of the changes at different layers of the architecture. Cloud Maturity Framework addresses those challenges previously presented by defining a three part structure: a) Workload Layers; b) Modernization process; and c) Maturity classification. Moreover, the framework proposes a detailed set of maturity criteria and correlates it to technical improvements that, once applied, can overcome the architecture violations previously found in the target workload [21].

 ISBN: 978-1-61208-997-3

REFERENCES

[1] P. Jamshidi, A. Ahmad, and C. Pahl, "Cloud Migration Research: A Systematic Review," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 142–157, 2013, doi: 10.1109/TCC.2013.10.

[2] S. Bhardwaj, L. Jain, and S. Jain, "Cloud Computing : a Study of Infrastructure As a Service (Iaas)," *Int. J. Eng.*, vol. 2, no. 1, pp. 60–63, 2010.

[3] J. Garrison and K. Nova, *Cloud Native Infrastructure*. O'Reilly Media, Inc., 2017. [Online]. Available: https://www.oreilly.com/library/view/cloud-native-infrastructure/9781491984291/

[4] Amazon Web Services, "Modernize Your Applications, Drive Growth and Reduce TCO." https://aws.amazon.com/pt/enterprise/modernization/ (accessed Jun. 10, 2022).

[5] L. Wang *et al.*, "Cloud computing: A perspective study," *New Gener. Comput.*, vol. 28, no. 2, pp. 137–146, 2010, doi: 10.1007/s00354-008-0081-5.

[6] W. S. Humphrey, "Characterizing the software process: a maturity framework," *IEEE Softw.*, vol. 5, no. 2, pp. 73–79, 1988, doi: 10.1109/52.2014.

[7] International Organization for Standardization, "ISO/IEC 33001:2015 - Information technology — Process assessment — Concepts and terminology." ISO/IEC, 2015. Accessed: Sep. 10, 2022. [Online]. Available: https://www.iso.org/standard/54175.html

[8] M. Williams, R. Skipp, T. Scott, M. Estes, and W. Dupley, "OACA Cloud Maturity Model v4.0," Jun. 2018.

[9] "AWS Well-Architected Framework." Amazon Web Services. Accessed: Sep. 10, 2022. [Online]. Available: https://aws.amazon.com/pt/architecture/well-architected/

[10] "AWS Cloud Adoption Framework." Amazon Web Services. Accessed: Sep. 10, 2022. [Online]. Available: https://aws.amazon.com/pt/professional-services/CAF/

[11] "Azure Well-Architected Framework." Microsoft. Accessed: Sep. 10, 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/architecture/framework

[12] W. Lloyd *et al.*, "The cloud services Innovation platform - Enabling service-based environmental modeling using infrastructure-as-a-service cloud computing," *IEMSs 2012 - Manag. Resour. Ltd. Planet Proc. 6th Bienn. Meet. Int. Environ. Model. Softw. Soc.*, pp. 1208–1215, 2012.

[13] Y. Zhang, Z. Zheng, and M. R. Lyu, "Real-Time Performance Prediction for Cloud Components," in *2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, 2012, pp. 106–111. doi: 10.1109/ISORCW.2012.29.

[14] C. Fehling, F. Leymann, R. Retter, W. Schupeck, and P. Arbitter, "Cloud Application Architecture Patterns," in *Cloud Computing Patterns: Fundamentals to Design, Build, and Manage Cloud Applications*, Vienna: Springer Vienna, 2014, pp. 151–238. doi: 10.1007/978-3-7091-1568-8_4.

[15] P. Jamshidi, C. Pahl, S. Chinenyeze, and X. Liu, "Cloud Migration Patterns: A Multi-cloud Service Architecture Perspective," in *Service-Oriented Computing - ICSOC 2014 Workshops*, Cham, 2015, pp. 6–19.

[16] M. Richards and N. Ford, *Fundamentals of Software Architecture*. O'Reilly Media, Inc., 2020.

[17] G. Reese, *Cloud Application Architectures: Building Applications and Infrastructure in the Cloud*. O'Reilly Media, Inc., 2009.

[18] Amazon Web Services, "Implement change management process for cloud." Amazon Web Services, Jun. 2020. Accessed: Sep. 10, 2022. [Online]. Available: https://docs.aws.amazon.com/wellarchitected/latest/financial-services-industry-lens/implement-change-management-process-for-cloud.html

[19] CMMI Institute, "Capability Maturity Model Integration." ISACA, Mar. 2019. Accessed: Sep. 10, 2022. [Online]. Available: https://cmmiinstitute.com/cmmi

[20] V. Thumma, "Strategy for modernizing applications in the AWS Cloud." Amazon Web Services, Dec. 2020. Accessed: Jun. 10, 2022. [Online]. Available: https://docs.aws.amazon.com/prescriptive-guidance/latest/strategy-modernizing-applications/welcome.html

[21] N. J. Gunther, *Guerilla Capacity Planning: A Tactical Approach to Planning for Highly Scalable Applications and Services*, 1st ed. Springer Publishing Company, Incorporated, 2010.

 ISBN: 978-1-61208-997-3