IEEE-SEM, Volume 12, Issue 2, February 2024
ISSN 2320-9151


# Choosing the Appropriate Cloud Architecture for Modern Applications

**Mehmet Altug AKGUL**

***Abstract*- Modern application development and deployment are generally based on cloud architecture infrastructure that provides flexibility, scalability and cost efficiency. This article provides research to help users choose the best cloud infrastructure architecture for modern applications. Examines the key elements, advantages and best practices of cloud architecture, focusing on cost efficiency, scalability and flexibility. The use of microservices architecture, continuous integration and deployment, and best practices and the need for separation of applications are also discussed in this article. Additionally, attention is drawn to the difficulties and important points that should be considered when choosing the best architecture. These include weighing trade-offs, addressing infrastructure complexity, and resolving specific challenges presented by cloud-native applications. In both individual and corporate elections, different approaches should be evaluated and the best decision should be made.**

***Index Terms*- cloud, architecture, information technologies, microservices, infrastructure**



## I. INTRODUCTION

This article includes a research on which infrastructure approach would be better. Nowadays, not only how an application works, but also the architecture behind the application has become extremely important. Years ago, while creating a monolithic application and focusing on the result, today the application development process has become a process that needs to be evaluated from end to end. Many different approaches and areas of expertise have emerged in this field. For example, the DevOps field is one of these fields. Because, as important as developing an application is, it is also important that it receives updates without any interruption and that large development teams develop different applications at different times to create an entire application. In fact, this is basically why microservice architecture is implemented in large teams. Having dozens of teams working on a single monolithic application can lead to organizational confusion. Maintaining and updating microservices created by such different teams can be difficult. That's why the 'cloud-native' approach has emerged. With cloud-native applications determined by CNCF, containerized applications can be deployed independently of the infrastructure.

## II. CLOUD APPROACHES

Before asking the question of what cloud architecture is, we need to understand what cloud means. To put it simply, it means using an infrastructure that is not yours, that is, renting it in different ways, without investing in physical infrastructure. In a traditional IT infrastructure, that is, on-premise architectures, the institution has its own data center and in this data center, both the management of the servers, the network and management of other hardware elements is provided. In addition, the institution should also manage issues such as air conditioning of the data center, protection against natural disasters in terms of disaster recovery, and redundancy of the network system and electrical grid. This naturally means the cost of human resources and hardware infrastructure. The increase in network speeds and the diversification of public cloud providers and their offering of product ranges in different types of services have enabled cloud architectures to become widespread. A cloud provider, with its availability zones and regions in many different parts of the world, allows you to deploy applications anywhere in the world, regardless of location. This gives all individual and corporate users, no matter how small or large, a chance to compete.

However, regulated sectors and GDPR compliance issues mostly worry large enterprise-level companies about cloud use. Because storing the data in a different data center in a different country instead of storing it in an on-premise environment creates the perception that the data is in someone else's hands. For this reason, for many years, enterprises have kept their distance from the cloud or contented themselves with watching it from afar. However, in recent years, a hybrid cloud approach has emerged, in which we can use on-premise architecture, that is, private cloud and public cloud together. The main reason for this is to keep the applications and data I want in an on-premise environment, but to benefit from the innovative features of cloud services wherever desired.

This approach has been widely adopted, especially by companies in the banking and finance field, and almost the entire fintech sector has started to work on how to transition their existing systems to hybrid architecture by establishing their own cloud teams. While data and services where confidentiality is at the forefront, such as customer and account information, are located in the private cloud of organizations, after the microservice transformation, most of the applications that are not blocked by GDPR can be run on public cloud services. This actually provides an advantage to large companies in terms of infrastructure costs. Because the biggest advantage of the cloud is that it is actually scalable. In a traditional infrastructure, the scalability limit is the maximum capacity of your data center. However, scalability in cloud services can be considered unlimited. In particular, storage solutions are limitless.

## III. ARCHITECTURE TYPES

As we have always stated in the cloud field, there is no single truth. Because the established structure and architecture varies according to the scenario to be realized. In some cases, we may need to deploy a monolithic architecture on the cloud, but modern applications are now rapidly transitioning towards microservice architecture. In particular, the emergence of container technologies and infrastructure-independent applications has led to a rapid move away from monolithic applications. Of course, there are intermediate approaches between microservice architecture and monolithic architecture such as modular monolithic. The emergence of the concept of "serverless" with the As a Service approach has led to deploying only the functions in the cloud, not the entire end-to-end applications, thus saving time by developing much faster. If we think cumulatively, there will definitely be systems where all these approaches are used together. These systems are examples of hybrid architecture approach. [1]

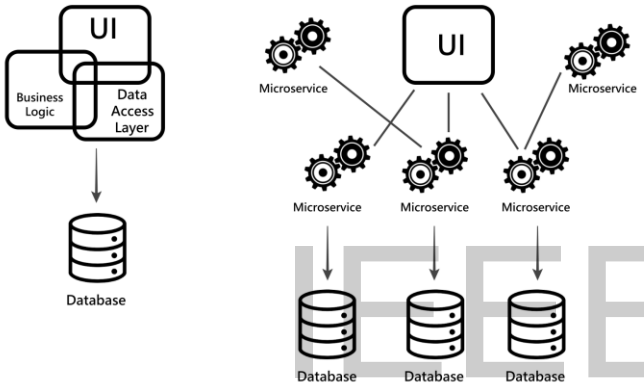


*Figure 1: Monolithic and Microservices Architecture*

In fact, when we look at traditional systems -we call these systems pre-cloud systems because the microservice approach became popular with the cloud- we see that most of the software is almost a monolith-like applications consisting of a single application and a single database running on a single server and server cluster. These applications are developed on a large repository and even a single change affects most parts of the system. Maintaining, updating and debugging the application can be quite difficult. Additionally, since the system grows vertically, performance problems and security risks are very common. These types of applications are ideal for simple and small applications that do not require frequent changes or high availability.

However, when we look at the microservices architecture, applications consist of many different sub-applications that communicate with each other through interfaces that we call APIs. It provides a very organizationally relaxing development process for modern application development and modern software teams. Hundreds of different teams can independently develop hundreds of different services and merge them into a single application. This allows us to move away from confusion and chaos and only improve on the determined target. Each service is responsible for its own function or domain. These services can be quickly deployed and scaled as needed. Each main service has its own database, which allows databases to grow distributed horizontally. It is very comfortable in terms of easy updating, flexibility and testability. It is a very useful architectural approach for complex, dynamic and very large applications that require high availability and scalability.

There is also a modular monolithic architecture, in which you create different services like in microservice, but you still connect them all to a single database. This is the type of architecture that is most confused with microservice architecture. However, in order to have a microservice architecture, databases must be separated from each other. The disadvantage here is that a single database grows vertically and becomes difficult to maintain.

Serverless architectures provide an ideal infrastructure for "small applications that do a single job" that have emerged in the sector in recent years. So, what is serverless? In fact, it is an approach that only allows developing applications on a function-based basis without managing the infrastructure and servers. Of course, there are servers in the background, so as the name suggests, development is not possible without a server, but instead of giving the management effort to the infrastructure, you can design applications in a more atomic structure and thus provide rapid development in transaction-based systems or applications that perform a single task. The cloud service provider manages the server on which your application runs, scales it, monitors its security and performance, and charges only for the resources you use. This provides you with lower costs, faster development and easier maintenance. It is especially ideal for applications that are not too complex in terms of business logic, are stateless, and have a short lifecycle. The best thing about these systems is that you have the opportunity to trigger different cloud services with a serverless API that you open and you can set up highly automated systems with this trigger mechanism.

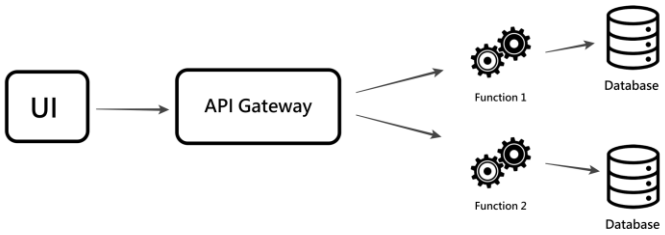


*Figure 2: Serverless Architecture*

Finally, we come to the hybrid architecture approach, as the name suggests, it is an approach to create a hybrid design by using different architectural approaches together to benefit from the strengths and minimize the weak features. For example, we can develop different types of applications in different data centers in different locations, and one is a monolithic application and the other is written with a microservice. We can combine these with a gateway and then create a serverless system on the cloud according to needs. This triple system means creating a large application by using different architectures together and applying hybrid architecture. In fact, the most commonly used architecture type today is hybrid architecture, because as applications grow, they begin to consist of multi-layered and discrete structures.

## IV. ARCHITECTURE TYPES

Deploying and updating applications is often a difficult process for both developers and users because some problems such as interruptions and access problems arise during the updating and maintenance of applications. It is at this point that the concept of container comes into our lives. In fact, most people are familiar with virtual machines. What do virtual machines offer us? It provides a place where we can install and run different operating systems in isolated environments. So, we can manage and scale as many virtual machines as we want. The container approach allows us to run applications in independent container environments, just like a virtual machine, and to distribute them quickly.

While it has been possible to isolate processes since 1979, everything changed completely with the emergence of Docker in 2013. Now you could quickly containerize your applications and distribute them quickly. Basically, you choose which operating system your application will use in a container config file, then specify which commands and what operations it should perform after the operating system installation, and place your application into a ready-to-use container environment. Thus, the person who installed this container could start using the applications very quickly, without any settings or minimal changes. This convenience, especially in scalable systems, has completely changed the game. Of course, since managing these containers one by one would be a very difficult process, container orchestration applications began to emerge. Kubernetes is one of these applications. In cases where the network load increases, instead of scaling the virtual machines, declaratively telling and monitoring the operations to be performed by an orchestration tool such as Kubernetes has made the systemic processes much easier. Thus, highly accessible, easy-to-maintain, scalable, fast-deployable, self-healing systems can be built.

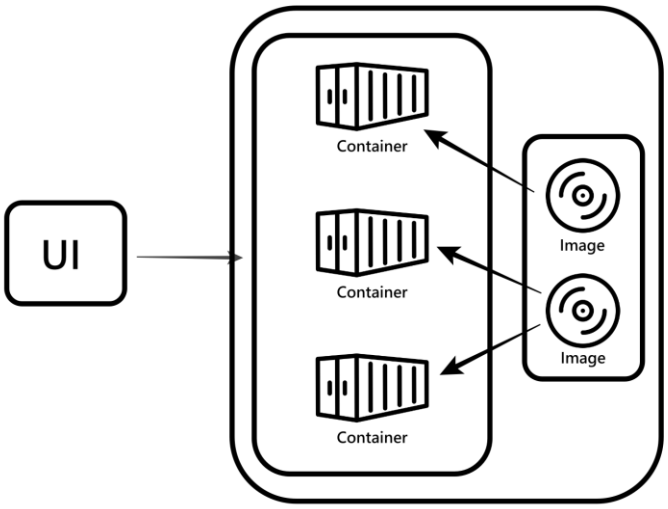


*Figure 3 Basic Container Architecture*

If we look at container technologies, we can give examples of popular applications such as Docker, Containerd, Podman. Kubernetes has now started using Containerd as the container runtime by default. Applications such as Kubernetes, Docker Swarm, Apache Mesos, OpenShift, Nomad can be given as examples to orchestrate these containers. Many open-source and licensed applications based on Kubernetes are also available in the market. We packaged our applications, isolated them from each other by placing them in containers, and then managed these containers. Now it's time for maintenance and delivery. In fact, the DevOps culture welcomes us here too. If you ask what DevOps is, it is an approach that will cover the entire software development life cycle. It is the abbreviation of the words Development Operations. While developing an application, different teams make different updates and improvements. They keep these developments in a common version control system. During the life cycle, these updates are regularly received from the version control system and deployed to the production environment.

We call this process the CI/CD process. Its definition is Continuous Integration and Continuous Delivery/Continuous Deployment. A pipeline is set up and in case of a possible software update, the codes taken from the repository are first analyzed for static code, then built and put into testing. If all the rules are successful, the release phase is reached and deployment takes place. Afterwards, it is necessary to carry out monitoring operations and continue this cycle continuously. The most important stage of DevOps is this cycle. Deployment is generally stored in the registry as the container images we explained at the beginning, and these current applications are kept running with container orchestration. In fact, we have summarized the process from the development phase to the delivery phase to the user.

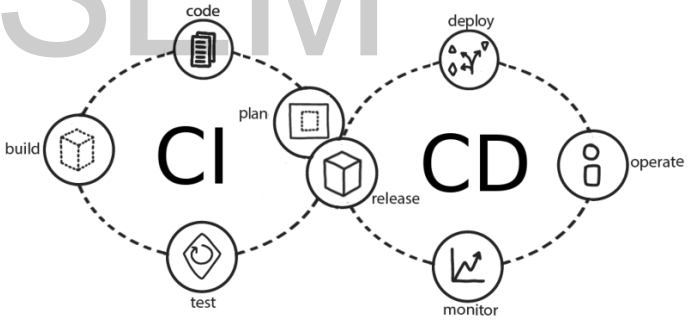


*Figure 4 CI/CD Process*

## V. CLOUD NATIVE

We have moved from traditional software development and architectures to modern architectures and come to the “Cloud Native” approach, which is one of the most popular architectural approaches today. Cloud Native actually means cloud-native architecture. A cloud-based application must be scalable and suitable for automation. It is very important that it can be easily deployed in any cloud environment and moved when necessary. We talked about public cloud, private cloud and hybrid cloud approaches. It is important that a cloud native application adapts to all cloud infrastructures here. Applications that are compatible with the cloud native approach are listed according to categories on the Cloud Native Computing Foundation (CNCF) website.

CNCF has more than 400 members, including cloud providers, enterprise software companies and technology entrepreneurs. Microsoft, Oracle, VMware, Intel are among these members. A cloud native system must be durable, manageable and observable. Industry-leading cloud providers also offer cloud tools and services so that developers can have faster development processes.

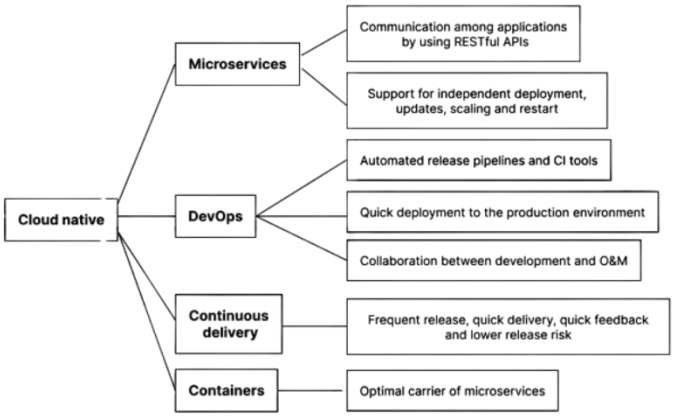


*Figure 5 Cloud Native Components*

Cloud computing has become a game-changer in the world of technology and has completely changed the industry. Almost all technology is now concentrated on the cloud, from the first point of software development to the last point reached to the user and maintenance services. Developing applications without being tied to a single software language, operating system and a single tool, packaging these applications and delivering them to all clients from a single place, instantly presenting an update to the whole world and realizing this on more than one cloud, migrating from cloud to cloud and all this very quickly. Realizing it in some way really explains the importance of the cloud much better. When we consider how difficult this process is in traditional architectures, we understand how the market potential of cloud systems has an upward trend in 10-year charts. Global cloud market volume will increase to 4x today's volume on average by the 2030s

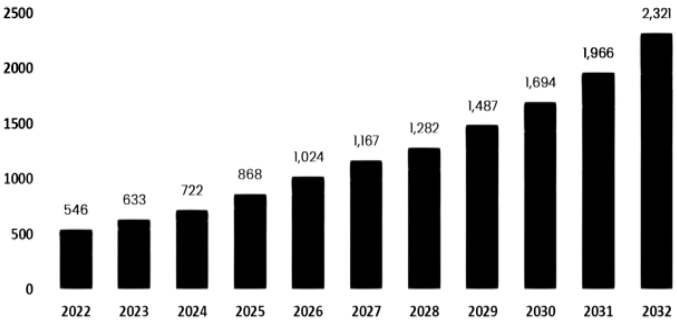


*Figure 6 Global Cloud Infrastructure Services Market Size Insights Forecasts to 2032[2]*

Looking at Flexera's cloud market research, 24% of cloud users use only public cloud, while only 4% use only private cloud. 72% of them use hybrid cloud. Hybrid cloud was preferred because private cloud is generally required in regulated sectors. In general terms, hybrid cloud is an ideal architectural approach for most sectors. [2]

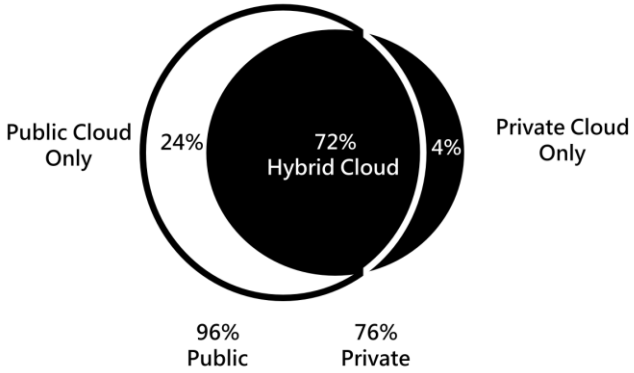


*Figure 7 Public vs Private Cloud Usage [2]*

After taking a look at public, private and hybrid cloud issues, there is a final approach where we can use many cloud environments as an alternative. Multi-cloud architecture. In this architecture, services offered by more than one public cloud provider are used together. Thus, when a hyperscaler offers services that are much more advantageous and performant than others, the chance to use this service is not missed. It is a very beautiful architectural style and companies are slowly starting to switch to this structure. Services that are not available in some public cloud environments may be available in others, or it may be necessary to evaluate different regional availability zones and data centers on a country-by-country basis. Therefore, multi-cloud is a very logical and useful architectural solution according to today's agility approach.

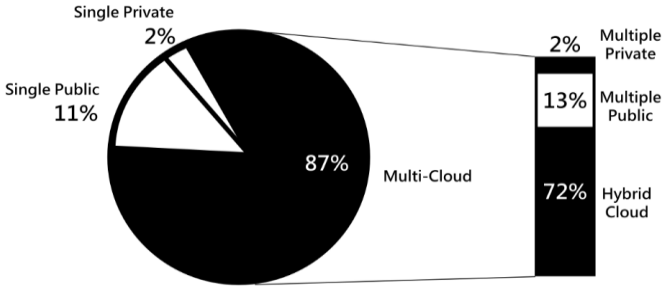


*Figure 8 Multi-cloud Usage[2]*

The question 'Which architectural approach should I choose?' is generally asked by all system architects and developers, and in fact, the better and stronger we lay the foundation of a building, the more robust it will remain as the application grows and scales. Likewise, the choice of architecture in IT systems is also very important, just like a building. Especially if you are a company that comes from the past and has legacy systems, switching directly to modern architecture will take a lot of time and cause serious problems in terms of resources and accessibility. First of

all, it is necessary to plan this transition very well. In this planning, it would be good for the first stage to create a private cloud environment from legacy architectures. If we can provide a private cloud environment and manage resources in the SDDC environment, our job will be much easier. Because most of the SDDC infrastructures have cloud connections. It has hybrid and multi-cloud management features. VMware and HPE platforms develop end-to-end enterprise solutions in this regard. [3]

After the private cloud transition, you will now manage your resources from a common pool. Likewise, you will manage your network and storage resources from a common pool. You can start your corporate cloud migration by relocating your applications in your SDDC-based datacenter. Starting transformation projects will be a good step for you to gradually transition monolithic applications to a modular, monolithic and microservices structure. As your applications gain a modern architectural approach, you can now move your applications located on the private cloud to the public cloud infrastructure. Since cloud is a very comprehensive subject, you can get support from experts and consultants trained in this field or establish your own cloud team. The SDDC tool you have used will probably give you an advantage in multi-cloud management and you will be able to manage all your nodes and clusters from a single center. With microservices transformation, you have turned your applications into a cloud native feature and can move your containerized applications to the cloud environment of your choice. DevOps teams and cloud teams will work together on this issue and design your CI/CD processes and service management in an interconnected way.

Some of your applications may still remain as monoliths, and that's okay. You don't have to convert all your services into microservices. You may have APIs that do a single job, and these are internal APIs that do not open to the outside. These applications can remain the same and be scaled in transformation, but transforming your applications that are exposed to heavy traffic with a microservices approach will be very good for the long-term health and maintenance of your company and applications.

## VI. CONCLUSION

In this article, we have discussed the end-to-end journey of the software and took a look at both the software architectures and the infrastructure architectures in which this software is deployed, from monolithic applications to microservices, from private cloud to multi-cloud, and we created an idea about the future with the statistics prepared on this subject. Cloud space is a very wide area and there is no single truth. Each strategy brings with it different architectural approaches. You should analyze the systems well and design your architectures by foreseeing the medium and long-term future, not today.

## REFERENCES

[1] Imran, Hamza & Latif, Usama & Ikram, Aziz & Ehsan, Maryam & Ikram, Ahmed & Khan, Waleed & Wazir, Saad. (2020). Multi-Cloud: A Comprehensive Review. 1-5. 10.1109/INMIC50486.2020.9318176.

[2] Flexera, "2023 State of the Cloud | Report," https://info.flexera.com/CM-REPORT-State-of-the-Cloud?lead_source=Organic+Search

[3] Hong, Jiangshui & Dreibholz, Thomas & Schenkel, Joseph & Hu, Jiaxi. (2019). An Overview of Multi-cloud Computing. 10.1007/978-3-030-15035-8_103.



## AUTHOR

**Mehmet Altug Akgul** – Mehmet Altug Akgul, Cloud Solutions Architect, info@altug.dev