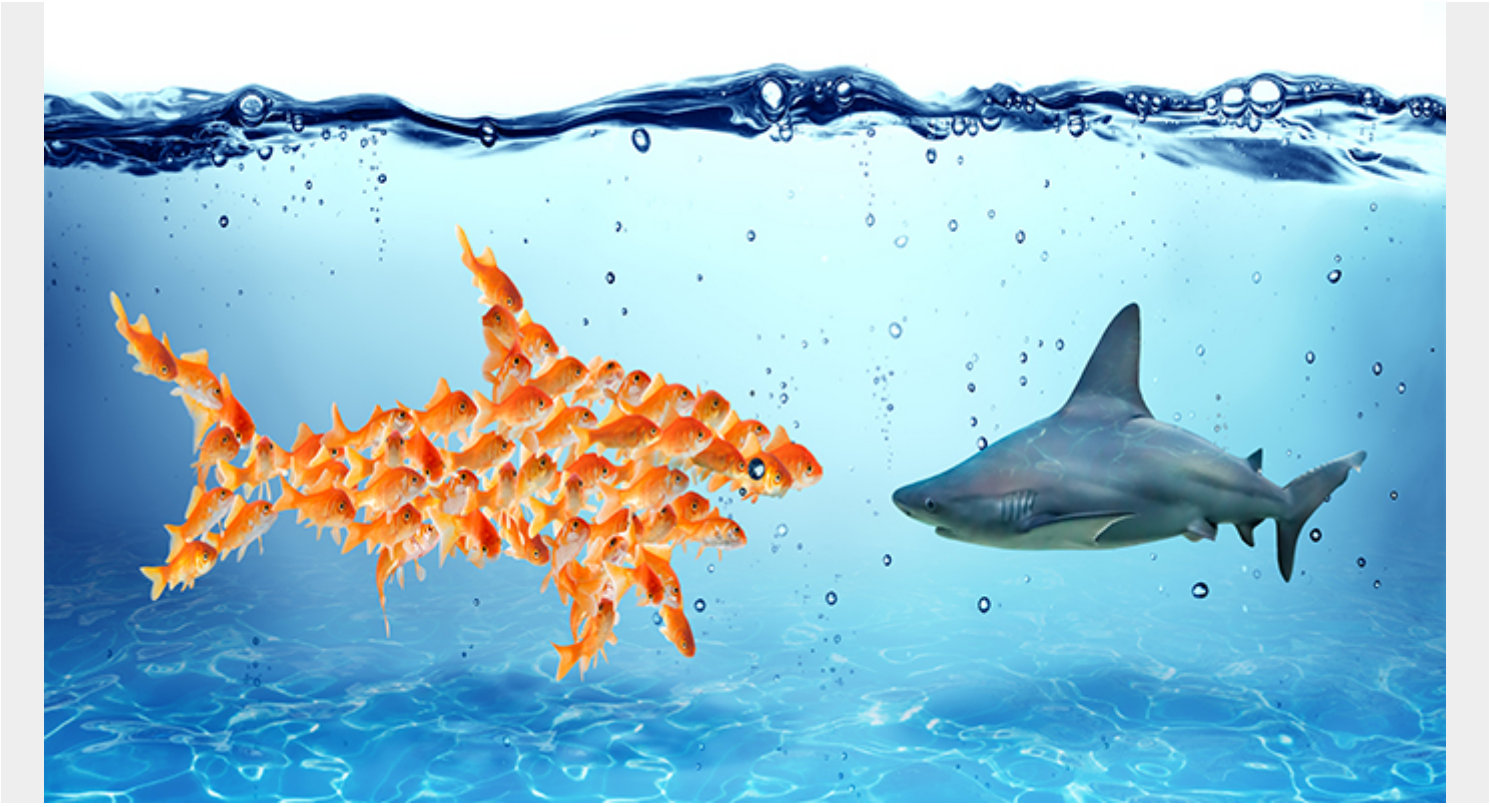
MONOLITHIC VS MICROSERVICES ARCHITECTURE (MSA)



Microservices, also known as microservice architecture, are a specific method of designing software systems that structures an application as a collection of loosely coupled services.

The main idea behind microservices is that some types of applications are easier to build and maintain when they are broken down into many small pieces that work together. Each component has its own small team working on it so they are completely decoupled and separated from each other, allowing each service to run its own unique process and communicate autonomously without having to rely on the other teams or applications as a whole.

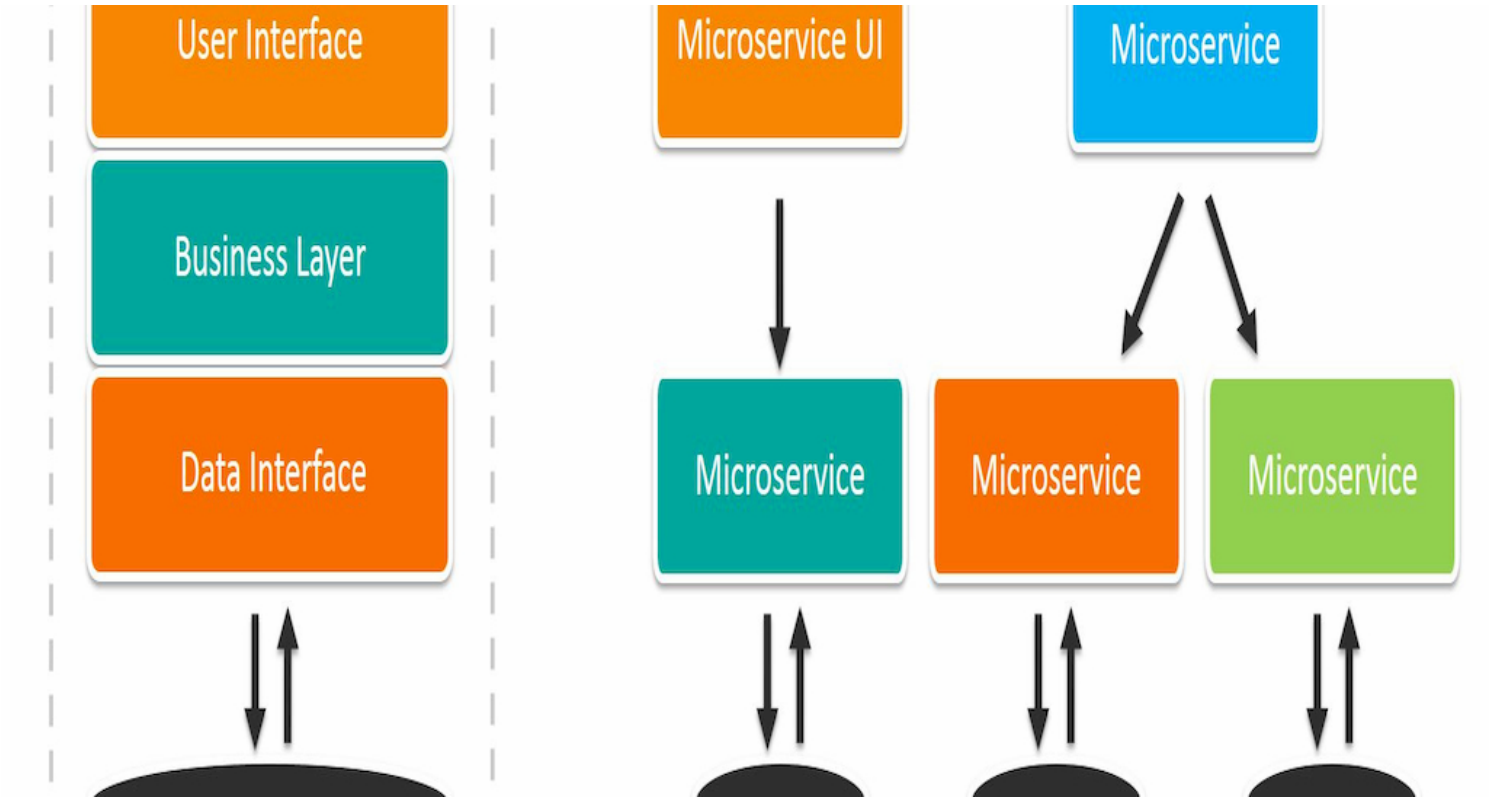
This ability to be separated and recombined protects the entire system against decay and better facilitates agile processes, making it an appealing method for organizations to adopt, especially those who are still utilizing monolithic architecture.

Monolithic Architecture vs. Microservice Architecture

The monolithic architecture pattern is the traditional architectural style that many systems utilize, with the monolith application built as a single, autonomous unit. While this style has been an integral part of many businesses, its numerous limitations and issues are motivating more and more to make the switch to microservices.

Monolithic structures make any changes to the application extremely slow as it often affects the entire system. It can require a completely rebuilt and deployed version of software whenever a

modification is made to a small section of code. If developers wish to scale certain functions of an application, they must scale the entire application, further complicating changes and updates. Microservices help to solve these challenges and more.



Microservices are often considered to be a logical evolution of Service Oriented Architecture (SOA). In a related blog post, we go into greater detail on the differences between SOA and Microservices.

Benefits of Microservices

Although some developers may be resistant to change, or hesitant about trying something different in their applications, the benefits of microservices far outweigh the potential disadvantages for many applications.

Applications built as a set of independent, modular components are easier to test, maintain, and understand. They enable organizations to increase agility while improving workflows and the amount of time to takes to improve production. Microservices have already proven to be an extremely superior system, especially for large enterprise applications that are developed by distributed and diverse teams. Outside of this, they have many other distinct advantages that are outlined below.

Developer Independence

With small teams working on a microservice, it contributes to much more developer freedom and independence. Small teams that are working in parallel can iterate faster than large teams. They can also scale the services on their own without having to wait for a larger and more complex team.

Isolation and Resilience

Another big advantage of microservices is their qualities of isolation and resilience. If one of the components should fail, due to issues including the technology becomes outdated or the code in the service cannot be developed any further, developers are able to spin up another while the rest of the application continues to function independently. This capability gives developers the freedom to develop and deploy services as needed without having to wait on decisions concerning the entire application.

Scalability

Due to the fact that microservices are made of much smaller components, they are able to take up fewer resources and therefore more easily scale to meet increasing demand of that specific component only. As a result of their isolation, microservices can properly function even during large changes in size and volume, making it an ideal method for enterprises dealing with a wide range of platforms and devices.

Autonomously Developed

As opposed to monoliths, individual components are much easier to fit into continuous delivery pipelines and complex deployment scenarios. Only the pinpointed service needs to be modified and redeployed when a change is needed, and if a service should fail, the others will continue to function independently. Outside of the obvious benefits this offers the system, this autonomous nature is also beneficial to the team as it enables scaling and development without requiring much coordination between other teams, particularly an advantage as companies become more distributed and workers more remote.

Relationship to the Business

Microservice architectures are split along business domain boundaries, organized around capabilities such as logistics, billing, etc. This increases independence and understanding across the organization as different teams are able to utilize a specific product and then own and maintain it for its lifetime.

Evolutionary

A final benefit of microservice architecture is the fact that it is highly evolutionary. Microservices are an excellent option for situations where developers can't fully predict what devices will be accessed by the application in the future, and they allow quick and controlled changes to the software without slowing the application as a whole.

Examples of Microservices Architecture

As Martin Fowler points out, there are many large companies that now utilize microservices within their architecture. Some of these include:

- Netflix
- eBay

- Amazon
- Twitter
- PayPal
- SoundCloud
- Gilt
- The Guardian

Netflix and Amazon are known for their diverse architectures which have evolved from monolithic to microservices in order to handle their large volume of data. eBay has also gone through the same transition.

How Netflix Handles Microservices

In this video from QCon 2016, Josh Evans explains how Netflix manages their microservices architecture – both challenges and benefits for the microservices approach.

Conclusion

In the end, microservices are part of a complete shift in the larger IT and DevOps movement, which allows strong collaborations between development and operations teams. Microservices are not just a technology being used but rather they are an entire concept that organizations need to adopt culture, knowledge, and structures for development teams to fully be able to incorporate this model.

Resources: Learning More About Microservices

SlideShare hosts very useful presentations on subjects related to microservices. Here are some of the most useful decks on this topical area.

Introduction to Microservices from **Amazon Web Services**

Architecture: Microservices from **Amazon Web Services**

Introduction to Microservices from **Amazon Web Services**

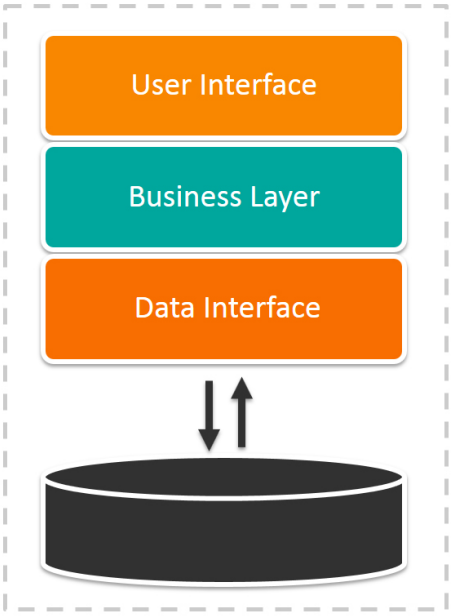
Introduction to Microservices from **Amazon Web Services**

An Overview of Designing Microservices Based Applications on AWS - March 2017 AWS Online Tech Talks from **Amazon Web Services**

A Modern Data Architecture for Microservices from **Amazon Web Services**

Original Reference Image:

Monolithic Architecture



Microservices Architecture