Modern Applications: Microservices architectures

Modern architecture for business agility

The proliferation of fast, affordable computing has allowed companies of all sizes to create internal efficiencies and reach more customers through digital products. However, the ubiquity of tools, multiple paths to market, and changing consumer preferences mean businesses must innovate faster than ever to remain competitive. Across industries, from agriculture to banking to telecommunications, rapid innovation is digital at its core. The way digital products have been traditionally built just isn't fast enough to enable the innovation that's needed to win in the market.

New architectural patterns

New architectural patterns like microservices allow organizations to accelerate the pace of innovation. Modern applications built with microservices architectures enable innovation by distributing the effort and investment over time and across smaller teams, increasing the speed of testing and delivering changes to the market. They allow fine-grained resource optimization and enable teams to rapidly scale both in how they build products and how they run them.

What defines a microservices architecture?

Specialized

Each service is designed for a set of capabilities and focuses on solving a specific problem. If developers contribute more code to a service over time and the service becomes complex, it can be broken into smaller services.

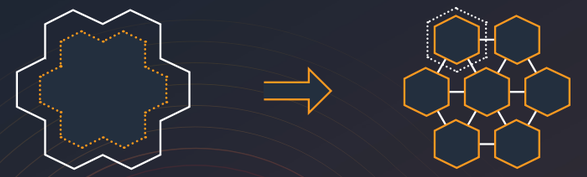
Distributed

A microservices architecture breaks your application from a single process into multiple components that work together to deliver value. Any communication between individual components happens via well-defined, loosely coupled APIs ,or through events and messaging.

Autonomous

Each component service in a microservices architecture can be developed, deployed, operated, and scaled without affecting the functioning of other services. Services do not need to share any of their code or implementation with other services. They act as self contained black boxes.

A monolithic architecture may work well today, but challenges often arise as your business grows. Microservices help you address common challenges like scaling up and deploying new features quickly.



Breaking a monolith

Breaking a monolith can be intimidating. Give your team some practice by inviting them to complete an AWS project that will walk them through the process of breaking a monolith into microservices. Try the tutorial >>

Moving to microservices

Learn how Mobvista adopted a microservices architecture to improve the scalability and reliability of it's platform. Read the case study >>

The microservices advantage

Microservices architectures are built with discrete, modular elements that work together. While this modularity does come with the challenge of increased code "surface area," it also offers key advantages for innovating faster; scaling independently; reducing the impact of failures; and allowing for distributed code development.

The benefits of microservices

1. Agility: A small team that's working on an individual service component is freed from the constraints of other components and thus can be more nimble and respond to issues or opportunities faster.

2. Easy Deployment: Microservices reduce the size of changes, making it easy to try out new ideas and to roll back if something doesn't work.

3. Technological Freedom: Microservices architectures give teams the freedom to choose the best tools to create each part of an application.

4. Scalability: Microservices allow each service to be independently scaled to meet demand for the application feature it supports.

5. Resilience: Microservices reduce the impact of failure to a single part of the application. This means any individual component failure only degrades functionality instead of crashing the entire application.

6. Reusable Code: Dividing software into small, well-defined service components enables teams to use these service components for multiple purposes within an application. This allows an application to build off itself, as developers can create new capabilities by leveraging existing services via their APIs without writing code from scratch or having to deal with implementation details.

Microservices for modern applications

Microservices help your organization improve application resiliency and optimize team productivity. As a result, development teams are able to experiment and innovate faster, to release the products and features that deliver a competitive advantage.

Where to start

Adopting microservices architectural patterns doesn't have to be an all-or-nothing endeavor. There are two common paths to a service oriented architecture: (a) wrapping the existing monolith in APIs and treating it as a black box while building net new functionality as microservices; (b) refactoring the monolith to microservices using the strangler pattern. Both avenues have benefits and downsides, but whichever path you choose it will require first setting up the appropriate development infrastructure. This includes building automated software delivery pipelines to independently build, test, and deploy executable services and the infrastructure to secure, monitor, operate, and debug a distributed system.

Keeping the monolith as is can work if it's a standalone system that won't require updates to its core functionality. In this case, most new development effort can go to building new microservices that simply connect to the monolith through APIs. If the monolith can't be maintained or thrown away but some of its parts need to be rewritten, using the strangler pattern is the best approach.

With the **strangler pattern**, development teams carve out functionality that is fairly decoupled from the monolith already. This functionality is decoupled from the monolith behind an API for easy replacement and decommissioning once the microservice is built. This means capabilities that don't require changes to many client facing apps and potentially don't need their own data store are ideal first candidates. In an e-commerce application, for example, a few potential services to consider are authentication, invoicing, or customer profiles. When carving out their first few microservices, most teams aim to test and optimize their software delivery pipelines, API approaches, and upskill team members, rather that optimizing for functionality.

Visit us to learn more about the potential of microservices in your business.