Fiorano®

WHITEPAPER

Driving Digital Strategy with Microservices

TABLE OF CONTENTS

EXECUTIVE SUMMARY

In today's age of technological disruption, every business needs to stay on top of digital transformations to stand out from their competitors and deliver an outstanding customer experience. Technology has come a long way from being a far-fetched luxury. It has now become the most fundamental necessity for businesses to merely survive. Today, modernization, innovation and agility are a part of core business principles that help organizations understand their stakeholders' expectations and achieve new heights. Any business that does not keep up with technological advancements and service upgradations, is destined to fail.

According to Gartner, in 2022, 91% of businesses are engaged in digital initiatives and approximately 87% of business leaders believe digital transformation is a priority. It is quite evident that technology is dictating strategic decisions and the growth pace of modern businesses. Technological innovations are all over the map; to stay ahead of their competition, businesses require to adapt these innovations quickly and effectively, thereby offering differentiation.

Not every business has the potential to capitalize on opportunities offered by digital transformation. To effectively adopt and implement technological innovations, organizations need to be highly agile, such that work contributing to business value is broken down into hyper-focused units, handled by specific experts, and is functioning in coordination to achieve a larger business goal, without any errors or delays. Simply put, every business function needs to be fragmented into micro units, thereby ensuring specialization and smoother flow of activities, independently within the organization.

This entire process requires businesses to set-up the relevant IT infrastructures and system-support mechanisms that can help maintain agility to meet business goals. Undoubtedly, every business thrives in an extremely dynamic environment that requires organizations to update their strategies and processes with time, minimizing the waiting period for their stakeholders.

This agility can be achieved when the IT team has set up the relevant digital assets to support core business competencies. When digital assets add irreplaceable value to business functions, IT becomes the true enabler of business growth and expansion.

Countless businesses have adopted the microservices architecture to foster a strong foundation for innovation and agility, thereby achieving an immaculate response mechanism towards ever-changing environments and incessant demands

INTRODUCTION

Previously, businesses adopted technology and software as per the demands and needs of their individual departments. However, due to lack of integration and alignment of technology architectures across various departments, organizations faced problems like missing information or inaccurate reporting, thereby affecting the overall efficiency of their systems. With technological advancements, system software became more reliable. When technology offered to integrate various business departments, it revolutionized the way businesses functioned. Today, multiple technology options are simplifying core business functions by taking responsibility for the coordination and integration of business information and processes.

One such ground-breaking technology is the 'microservices architecture' that offers to run every application by decomposing it into multiple loosely coupled services.

Ever since the term 'microservices' was first coined in 2011, it has been creating a compelling buzz among forward-looking, modern businesses. Microservices offer to redefine the core business architecture using independent but closely related services.

Unquestionably, microservices cannot deliver positive results overnight. However, in the longer run, microservices promise to form a digital strategy that adapts to the ever-fluctuating needs of every business.

According to a survey on microservices adoption - conducted by O'Reilly, in 2020 - 77% of businesses have already adopted and are enjoying the benefits of microservices architecture. The concept of microservices has gained traction in the past few years, as it offers greater value than purchasing or building separate applications for every business need. Once implemented, the microservice architecture smoothly addresses all the automation requirements of tech-enabled businesses.

WHAT ARE MICROSERVICES?

Microservices are short, self-sufficient parts of code that execute specific business functions.

Every microservice is like a self-contained single-function module having clearly defined user interfaces and operations, with some common characteristic features like decentralized control of data and languages, close customization to match business capabilities and operations, automated deployment, intelligence at business endpoints, and much more. Therefore, it can be deployed independently, and developers don't have to worry about how the whole application will be impacted due to changes in one microservice.

'Microservices Architecture' was born as a variant of the 'Service Oriented Architecture (SOA)' and operates as a distinctive method of designing software applications, using independently deployable, modular services.

Microservices refer to an organizational as well as an architectural approach to developing software systems, where the software is composed of multiple uncomplicated, independent services, and these services communicate over well-defined **APIs**, using simple messaging protocols like **JMS** or **REST**.

The control of each microservice is in the hands of small, self-contained teams that offer independent failure detection, exception handling, maintenance, upgradation and other operations for every specific service. Therefore, every microservice is decomposed and loosely connected with others, to form an overall modular application, highlighting business capacities.

The microservice architecture fosters innovation and accelerates the time-to-market. It makes any application easily scalable, quicker to develop, simple to modify and devoid of errors.

MONOLITHIC V/S. MICROSERVICES ARCHITECTURE

The modular structure used by microservices was designed to overcome the challenges of monolithic systems.

A monolithic application is a single, autonomous unit that cannot be modified or scaled in parts.

If the developer must scale a specific function, the entire application will have to be scaled. If changes are made to one part of the code, an entirely new version of the software might have to be built and deployed.

With microservices, developers can build applications as suites of services. There can be an individual microservice for a specific function, and each one can run with a different programming language and storage technique. The resulting system will be both flexible and scalable.

Thus, the microservices architecture is primarily in contrast to the traditional monolithic architecture, which is made up of tightly coupled processes, and runs as a single service.

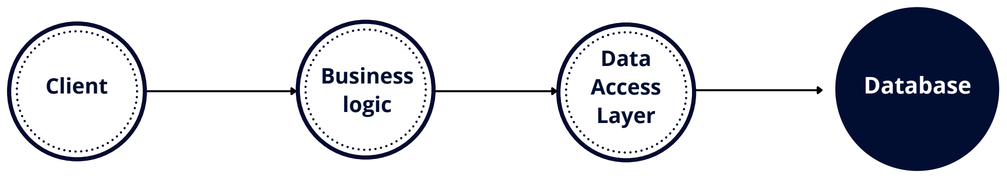
MONOLITHIC

In a monolithic architecture, if developers must modify or add new features to the application, the entire codebase grows, making the whole application much more complex and heavier, thereby limiting the potential for innovations. It hampers application availability due to the many tightly coupled and dependent processes, consequently heightening the impact of a single process failure.

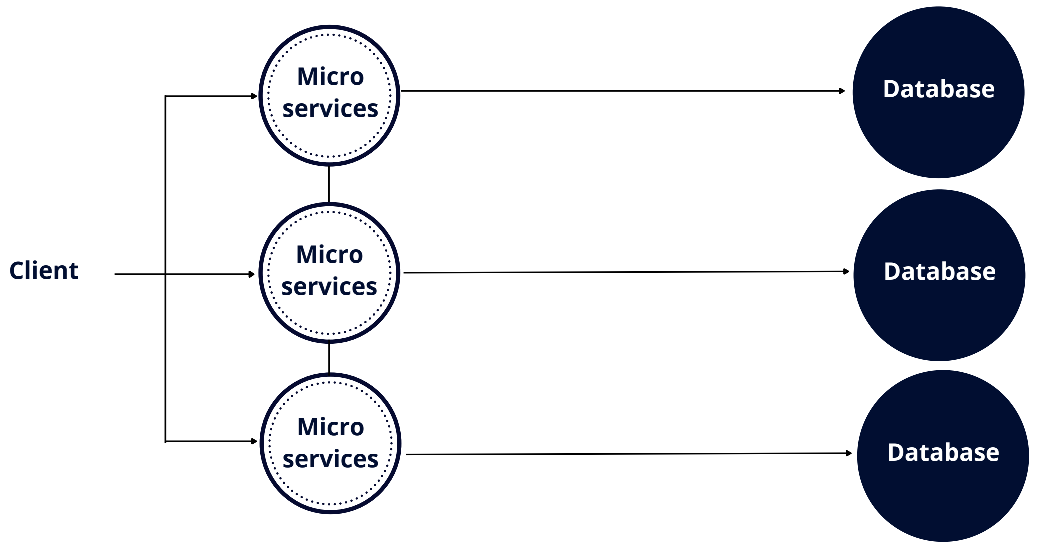
MICROSERVICE

Microservices architecture enables the deployment of multiple components, each running independently as a service/process within the application. Each service is loosely coupled and communicates using a well-defined protocol. Each of these services is built for and represents a core business capability, which can be easily modified, deployed, updated, and scaled, to meet the ever-changing business demands.

MONOLITHIC ARCHITECTURE



MICROSERVICE ARCHITECTURE



WHY MICROSERVICES?

The need for microservices roots out of necessity. With business environments and customer expectations growing more complex over time, modern businesses increasingly relied on technology to become more agile and innovative. This led to business software and applications growing larger and heavier, becoming highly cumbersome for developers to modify and scale, to meet business demands. Consequently, business applications were broken down into multiple, smaller services that are now used in the form of microservices.

These multiple, individual services can be easily integrated by developers to create an innovative and specialized application for every business. Developers can easily scale and modify services within applications, as business requirements and customer expectations evolve. Additionally, this modular style of app development offers various benefits, against traditional, monolithic architecture styles.

HOW TO IMPLEMENT MICROSERVICES?

The most fundamental consideration behind successful implementation of a microservice architecture is the ability to define all microservices with logical boundaries and align them with the businesses' core capabilities and customer expectations. It is also important to understand that every single microservice is dedicated to one specific feature or function. While implementing a microservice architecture, every organization must thoroughly consider the following:

1. Communication

The microservices architecture constitutes multiple independent services that perform single business functions and require communication with other services in the process, to achieve business goals. Communication is an integral aspect of this approach as it can either accomplish tasks on time or cause delays in the overall operations.

Microservices communicate in two different ways:

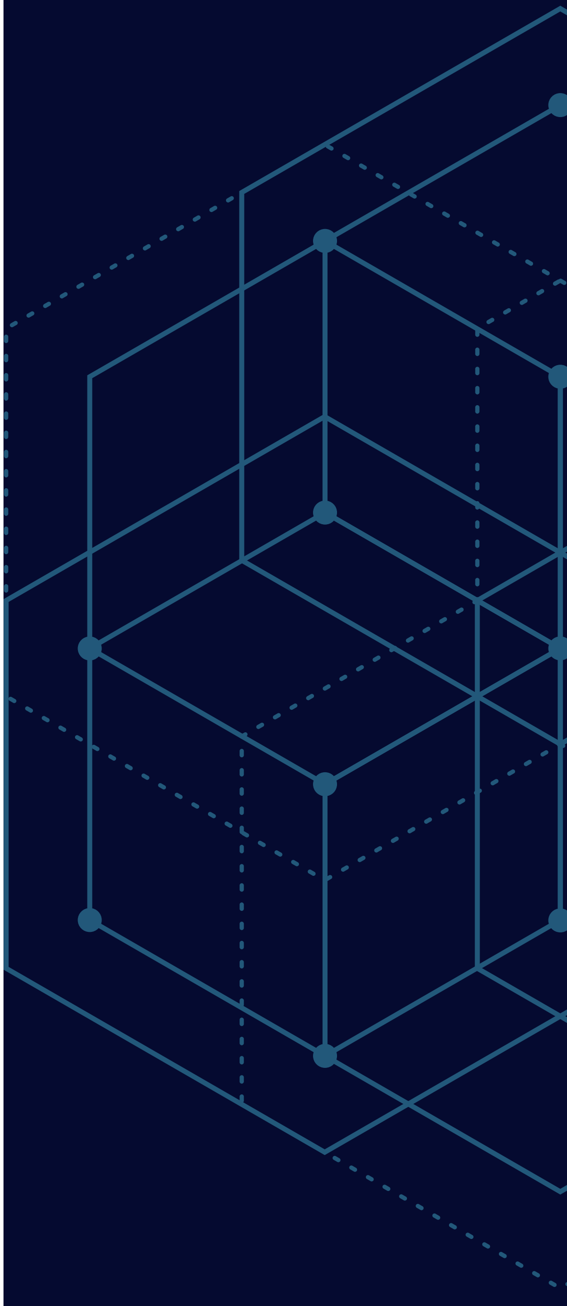
Synchronous Communication

- In this type, communication happens in real-time. The data moves to and from an endpoint of a service in a blocking interaction. It happens in a coordinated and patterned manner, thereby creating a chain of dependencies where services are called for, one after the other. This communication might involve delays, latencies, or errors if a particular service becomes unavailable.

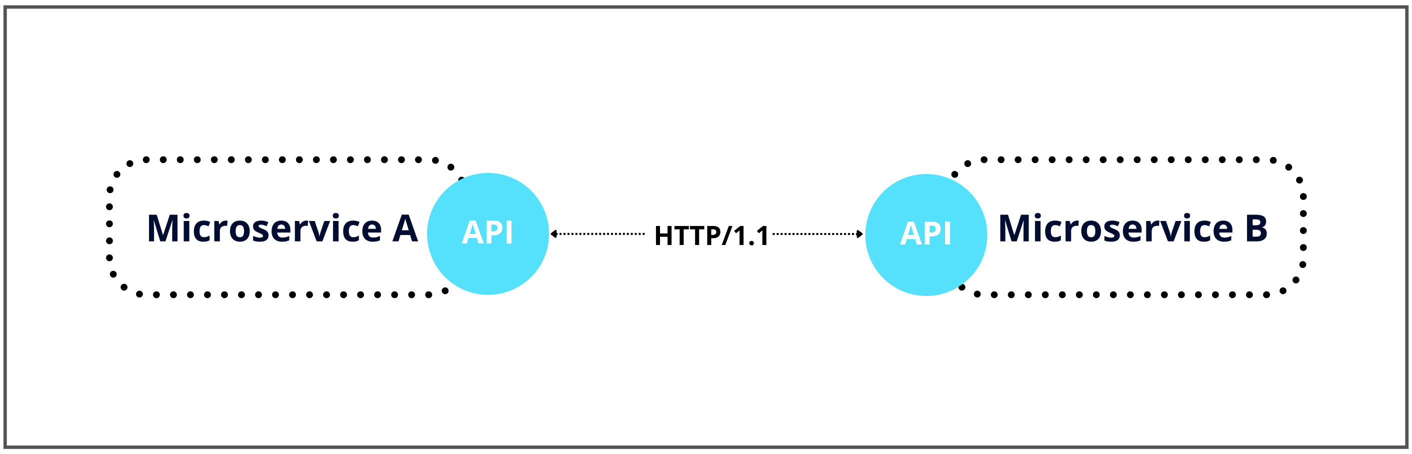
- REST protocols, adopted by Fiorano, are a good example of synchronous communication microservices.

Asynchronous Communication

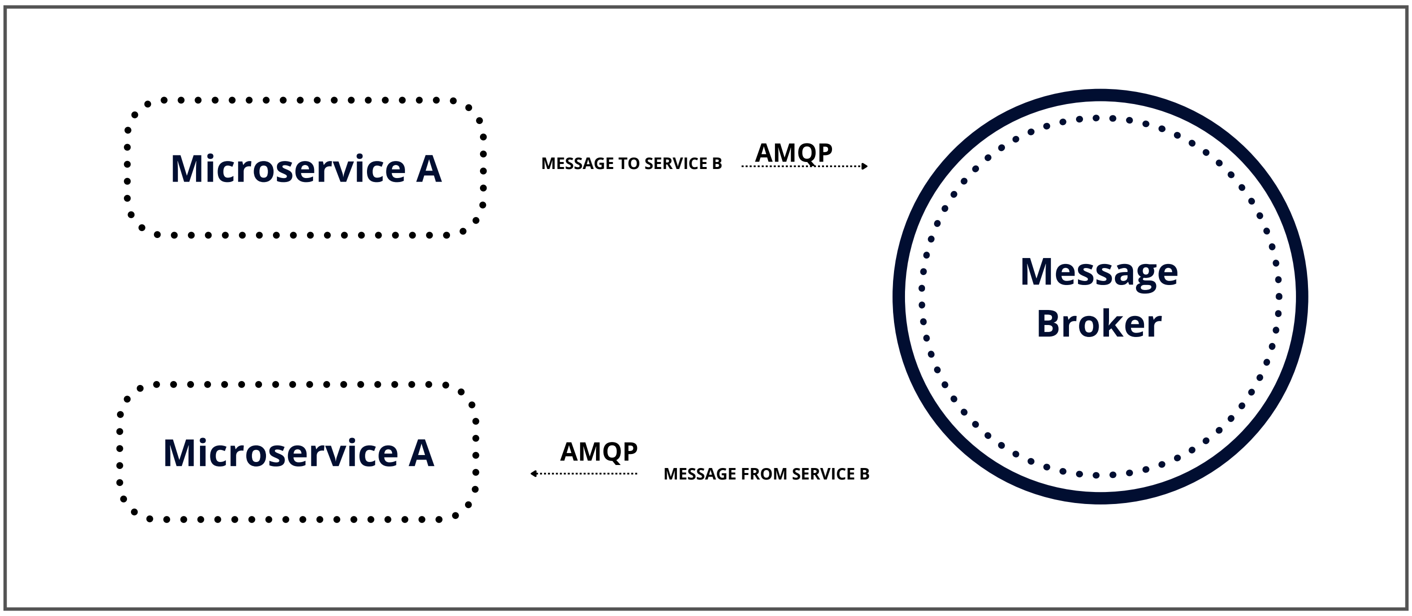
- In the case of asynchronous communication, the request to service and subsequent response happens independently. The general manner of using asynchronous microservice is by deploying a message broker technology like FioranoMQ.



Synchronous inter-service communication



Asynchronous inter-service communication



2. Integration

Since the microservices architecture involves decoupled services, integrating these services is crucial for the smooth functioning of the application software. Integration can be done using an API, Messages or using File Transfer.

An API style of integration uses synchronous communication, resulting in subsequent responses. The messaging style involves exchanging data through messages, written in a pre-defined format. Lastly, the File Transfer Integration style involved exporting the source file to the target system. It works best when loads of data require to be transferred and executed at predetermined times.

3. Deployment

The last and most crucial decision involved in the implementation of microservices architecture is the deployment strategy. The resulting flexibility, scalability, agility, and efficiency, after adopting microservices, depend on the choice of deployment strategy. The various microservices deployment strategies are- Server-less deployment, Service Instance Per Container, Multiple Service Instance Per Host, and Service Instance Per Host.

Microservices have become the best digital transformation approach for modern business. However, decisions related to implementation, management, security, and scalability will finally determine how successful and beneficial microservices prove to be for any organization.

FEATURES OF MICROSERVICES

01. Easier to Develop and Modify

Single microservice codes are smaller and more focused than monolithic applications. As a result, experimentation and testing get easier with incremental code updates. While the overall functionality of the service remains very similar to a monolithic application, the enforced decoupling of the code makes it easier to develop, modify and enhance. In this case, there is hardly any risk of the applications becoming 'black boxes,' wherein the source code is too complex to dive into.

02. Simplified Cross-Team Coordination

Any large-scale business IT infrastructure comprises multiple integration points, coordination, and maintenance of which becomes an unmanageable task. Even in the case of an SOA, one requires elaborate planning to determine how the data reaches its destination integration point. In contrast, microservices deal with low volumes of data, communicated using simple messaging systems like the JMS. These lightweight messaging protocols, used within the API, avoid complications in the coordination topology.

03. Enhances Performance and Scalability

In the case of microservices, developers must concentrate on individual services, rather than the whole application. Therefore, it gets easier to scale a few services, as per the changing business demands, rather than scaling an entire application at once. The concept uses the divide-and-conquer strategy, whereby integrating multiple microservices helps to harness power from multiple CPUs and RAM, leading to efficient and quicker processing of data. In this manner, microservices offer high scalability and enhanced performance.

04. Improved Data Security

Microservices allow for decentralized data handling and storage. Small teams take accountability and responsibility for specific services, avoiding centralized storage or handling of data. It leads to increased data security as all teams work in silos. Every service is only loosely connected to other services, thereby ensuring maximum security and reliability for the safety of stored data.

05. Ease of Maintenance, Troubleshooting, and Extension

Due to the decentralized and federal nature of microservices, running ongoing maintenance checks and troubleshooting issues in a particular microservice can be done easily, without disrupting the operational flow of the other microservices. One can easily add new functions or update existing functions without requiring complicated changes or understanding of the source code, as in the case of monolithic architecture. Therefore, maintenance, problem-solving and adding codes are easier and risk-free in the case of microservices architecture.

06. Freedom to Reuse Codes

Microservice Architecture does not follow the "one-size fits all" approach. Every service is a well-defined, self-sufficient component that performs a specific function within an application. Therefore, developers can use these microservice codes in every application that calls for a particular function. Also, a service written for one application, can become a foundation block for another function. This allows applications to bootstrap off themselves, while developers can reuse the same codes in innovative ways and create new capabilities, without having to write codes from scratch.

07. Multiple Components

Microservices software is designed with multiple component services that can be independently deployed, modified, and redeployed without compromising the entire application. Some downsides of this approach might be coarser-grained remote APIs and more complexity in redistributing responsibilities among components.

08. Simple Routing

Microservices have smart endpoints for processing the info and applying logic and dumb pipes for the information to flow through. Like the classical UNIX system, microservices receive requests, process them, and deliver a relevant response. This routing process is much more straightforward than ESBs (Enterprise Service Buses), which have high-tech message routing systems, choreography, and business rules.

09. Failure Resistant

Many diverse services communicating together in microservices may result in the possibility of some failures. However, when one service fails, the neighboring services continue to function seamlessly, thereby preventing complete software system failure. By simply monitoring microservices, even individual service failure can be prevented. Therefore, the microservices system architecture is more failure-resistant than a monolithic systems architecture.

10. Decentralized

Since various technologies are involved in microservices, decentralized governance is favored. Microservice developers try to create tools that are also useful for solving the same problems in different systems. Decentralized data management is also preferred in a microservices application, with each service typically managing its own unique database.

11. Built For Business

A microservice software is usually designed around business priorities and capabilities. Microservices applications use cross-functional teams, with each team trying to make specific products based on individual services communicating via message bus. The team that creates a product owns it for its lifetime.

12. Evolutionary

Microservices architecture is ideal for systems where it is impossible to predict the types of devices accessing an application. Many apps start with a monolithic architecture but require an evolutionary system when unforeseen requirements surface. Accordingly, these apps can slowly switch to a microservices architecture, with individually changeable services built over a monolithic architecture using APIs.

DIFFERENCE BETWEEN APIS AND MICROSERVICES

APIs, short for application programming interfaces, are responsible for enabling communication between applications. Microservices is an application-building approach whereby its functionalities are broken down into modular components.

Often, businesses confuse Microservices with APIs. Although thought of as synonymous, these two concepts differ in functionality. Put simply, APIs and microservices complement each other. APIs are required to make any microservice architecture work.

API

An API is a part of a web application that enables communication with other applications. Every software's API outlines a set of acceptable requests made to the API and responses to these requests. Basically, APIs allow applications to communicate and transfer information to one another. APIs also define how applications should interact. Therefore, APIs can be thought of as connectors, coupling or integrating separate applications.

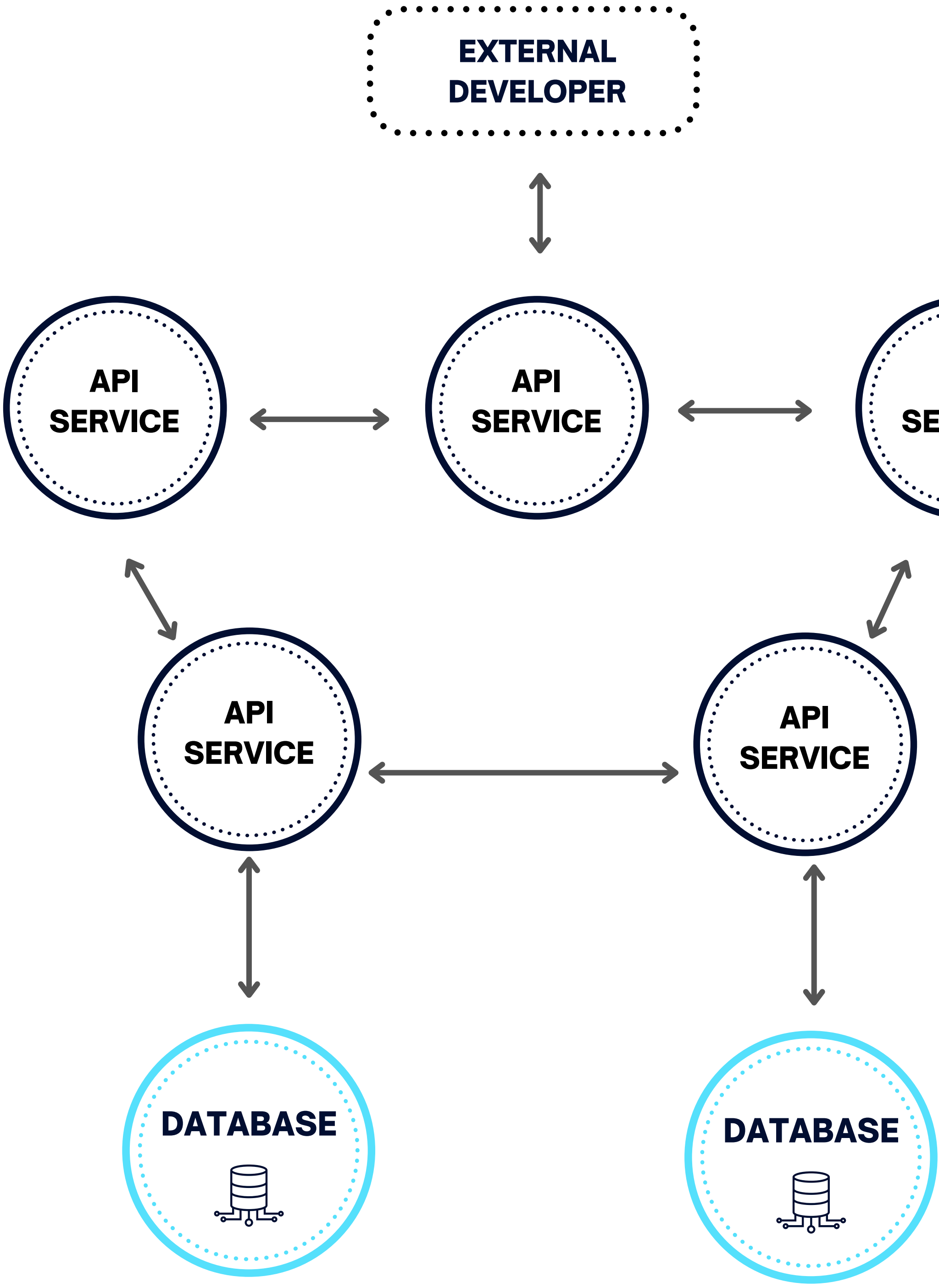
MICROSERVICE

Microservices are about providing agility and scalability. The microservices architecture breaks down applications into modular, self-contained components that have their own UI and database back-ends. However, to make multiple, independent microservices work within a single software, one requires connectors, i.e., APIs.

Every microservice is loosely coupled with another, using a private or light-weighted API, that integrates the various services, thereby forming an application software.

It is important to understand that all microservices use APIs differently. Some may deploy a single API for multiple services, while others may deploy multiple APIs for a single service. Also, APIs have uses beyond microservices. For example, web APIs allow data sharing among web applications.

APIs can be used to enable efficient communications between all the individually deployable services within a microservices application. They help codify interactions between various departments, reducing ad hoc communication and minimizing the complexities of coordination.



Microservice Architecture v/s. APIs

In a nutshell, APIs are needed to make decoupled services work together. Therefore, efficient management and security of APIs are crucial to the smooth functioning of a microservice architecture.

BUSINESS IMPACT OF MICROSERVICES

The success of any business today depends on its ability to be agile and adapt to the ever-changing needs of its customers. Companies that continue using a monolithic application structure need to put the entire app on hold even to make minor modifications.

In today's competitive market, exiting the scene even for short durations can lead to customers giving up and trying out competitors' apps. Therefore, to stay relevant among customers, businesses need to find a way to be agile and make changes on the go without pausing their apps or taking breaks to add new features.

The Solution

Microservices allow businesses to create modular applications with various individually deployable services connected with APIs. When one service in the app needs to be modified or revamped, it can be quickly done without interrupting the availability of other connected services. Multiple teams can work on separate services, enabling quick and easy deployment.

An excellent analogy for how microservices work is the coordination in a theater play. During a play, if one of the actors requires a costume change or needs medical attention due to an unexpected injury, they bow out and exit the scene gracefully and unobtrusively. The play then continues with the rest of the actors continuing to play their roles and improvising to cover up for the lack of one crew member. Similarly, when one microservice needs modification or replacement, it can be quickly detached from the rest of the app and modified in real-time. The other services continue to function as usual, and business doesn't stop.

When an app needs to be improved to suit new market demands, a new service can be developed independently and then connected to the rest of the services in the app using APIs. In this scenario, too, the app doesn't need to be paused at any time, and business continues to run efficiently despite modifications and upgrades.

The efficiency of microservices not only minimizes downtime; it also reduces costs. De-grouping services and working on them individually eliminates the need for expensive machines. The cost of modifying an app will also be less as changes to one service won't affect any other services in the app.
Therefore, microservices also allow you to cut infrastructure costs and optimize resources.

Microservices help businesses stay agile and adapt to the ever-changing needs of customers and the market. With microservices apps, Amazon can update its offerings on the go, Spotify can engage its customers through other apps like Uber, and Google can create new revenue streams by selling its APIs for other apps to use.

Apart from each service within a microservices app being able to communicate with each other, the entire app also has inherent interoperability. Microservices apps can communicate with each other if standards like HTTP and JSON are adopted across industries. Through this promise of connectivity and communication for seamless adjustment to change, microservices offer businesses a clear way forward and a smart way to stay relevant.

Thus, Microservices have steadily evolved into the most suitable application development approach for designing and deploying an IT architecture that meets the best practices and core capabilities of every business. It has been a catalyst for businesses to adopt a resilient digital strategy that meets the challenges posed by its external as well as internal environment.

With increasing competition and complexity of business environments, all modern businesses must have IT solutions that deliver a holistic and synonymous customer experience, across all platforms - physical as well as digital.

Thus, businesses need to set up technology infrastructure that aligns the businesses' core competencies and purpose with customer experiences. It will require businesses to actively take up constant upgrades and modifications of the technology set up, to remain in sync with the constantly changing business and customer demands.

Microservices have become an imperative part of every business's digital strategy because the architecture offers to automate and enhance every function within the businesses' software, making the system devoid of operational breakdowns, thereby making the business services agile and prompt. For example, microservices offer the opportunity to decompose the coupling between businesses' supply chains and the back-end systems that fuel these chains with customer information. They also offer to automate the transfer of customer data across the core value points within a business, thereby simplifying physical as well as digital supply routes.

Similarly, microservices have been proven to offer several benefits for businesses, which are unique from those offered by SOAs or monolithic architectures. Some key advantages for businesses adopting microservices architecture are:

1. Improved Business Agility and Time-to-Market

The decoupling of business processes allows businesses to allocate greater time, cost, and resources to functions that are crucial to the present business goals and objectives. Similar choices can be made to scale up services that are in demand, rather than all the services at once. Eventually, it leads to exponential savings. Therefore, the microservice architecture helps businesses maintain laser focus and prioritize services as well as resources crucial to the current market demands, thereby making businesses more prompt and agile.

2. Optimization of Business Resources

Undoubtedly, adopting the microservice architecture summons a capital investment for any business, as it involves a crucial digital transition that would result in a competitive strength for the businesses' future. Despite some significant costs of setting up an expert team and the digital infrastructure, migrating to microservices is seen as a source of remarkable savings in the long run. By pushing the business to prioritize the most crucial processes and avoiding any additional expenses for repairs or system failures, microservice architecture proves to be highly time and cost-effective in the longer run.

"Additionally, by reducing the time to market, costs of maintaining and upgrading heavy source codes, etc., only leads to optimization of human and financial business resources."

3. Hassle-free Migration and Upgrade

Migration to a microservice architecture is not as complicated as developing a monolithic business software application from scratch. The federal nature of microservices allows businesses to quickly choose the useful microservice bundles, which developers can loosely integrate into business software, without any coding or language barriers. Additionally, the most attractive benefit of using microservices is the ability to quickly upgrade to faster-moving updates and functions, without requiring altering a complicated source code.

4. Technological Freedom

The microservice architecture comprises multiple loosely coupled, independent application services. Each of these can be built using a custom set of technologies. This architecture does not follow the 'one-size fits' approach. It supports the use of multiple programming languages and database back-ends. Developers are free to choose the set of technologies suited best for the function they want to develop. Therefore, businesses and developers have the flexibility to choose their tech stack for the business function they wish to build.

5. Continuous Delivery

Microservices can significantly speed up the continuous delivery pipelines. In the traditional monolithic architectures, a single team of developers had to focus on all the aspects of the business software- the UI, the database, and the server-side logic. In the case of Microservices architecture, a single team focuses only on one specific functionality. Thus, multiple teams can simultaneously focus on distinctive functions of the same application, thereby increasing its agility and leading to enhanced continuous delivery.

Therefore, microservices architecture has successfully helped business organizations build high-performing, sustainable applications, effectively adapt to new changing technology, and increase agility with fast-tracking time-to-market.

WHY ARE MICROSERVICES IMPERATIVE TO A COMPANY’S DIGITAL STRATEGY?

Companies' digital offerings must evolve at the same pace as their customers’ demands to stay relevant. Applications with old monolithic architectures only lent themselves to rigid sharing and scaling, a slow and inefficient process that doesn't stand a chance in the volatile digital landscape. On the other hand, microservices apps have a modular structure, allowing companies to modify parts of the code independently without affecting the entire application.

The modular sharing and scaling options offered by microservices allow companies to make iterations on the go and scale their offerings at par with users' ever-changing expectations. Microservices offer the flexibility to use ideal tools for specific tasks. Each service uses a unique language, framework, and ancillary services while communicating with the other services in an application. Individual microservices can also be tested and debugged to deliver error-free applications on an ongoing basis.

You can also optimize resources and reduce downtime with microservices, as multiple teams can work together on independent services. While microservices have their fair share of advantages, they cannot work efficiently unless they are monitored closely and carefully. Since each service uses its own language and APIs, the entire app will suffer if one service is lacking. At the full-failure stage, it might be challenging to identify the root cause of the issues, so companies must monitor their microservices applications robustly to ensure reliability.

HOW DO MICROSERVICES HELP WITH DIGITAL TRANSFORMATION?

Microservices offer the reliability, agility, scalability, maintainability, and deploy-ability that allow enterprises to sail through a process of digital transformation.

The ideal way to go about this transformation would be to take on a pragmatic mindset from an enterprise architecture standpoint, with IT strategy leadership and business as primary stakeholders.

Many large-scale websites like Amazon, Netflix, and eBay have changed their software approach from a monolithic architecture to a microservices one to keep up with their digital user bases.

1
Amazon originally had a two-tier architecture but recently migrated to a microservices architecture with hundreds of backend services to scale their offerings. The Amazon website app calls 100-150 microservices to get the data used to build a web page.

2
Netflix has a large-scale microservices architecture that allows it to handle more than a billion calls every day to its video-streaming API from 800+ types of devices. Each API call is attached to an average of six calls to backend microservices.

3
Ebay also moved from a monolithic app architecture to a microservices architecture with multiple independent applications. Each app implements the business logic for a specific function, like buying or selling.

Reports have suggested that 90% of all apps will feature microservices architectures by 2022 to enable more efficient designing, debugging, updating, and leveraging third-party code. While many companies have already started using APIs to connect their existing set of monolithic applications, it is still difficult and expensive to modify and redeploy the applications when users demand changes.

Companies can tap into new business opportunities by applying microservices in an anticipative and proactive way, ensuring an impactful digital transformation that is effective in the present and the long run.

CHALLENGES TO IMPLEMENTING MICROSERVICES

The Microservices architecture is undeniably easier to implement and offers more advantages as compared to the traditional monolithic app development approaches. However, the deployment procedure requires more discipline and effort to design, build, and effectively manage. Just as every technology comes with a learning curve, the microservice architecture also poses some challenges that businesses need to overcome, as they adopt the microservice architecture. Although many businesses have already been experiencing the benefits of microservices, those starting from scratch must also know the challenges involved in the process of creating and using a microservices application for their business.

Challenges include:

01

Achieving Data Consistency

In the case of microservice architecture, every service handles its data independently, across its data stores, causing a problem of data redundancy. For example, when one service stores data for specific transactions, the same data may get stored across various services for reasons like analysis, reporting or even archiving. Moreover, when multiple services are tied to the same database, alterations in one service may cause a cascading effect across multiple linked services. This negates the purpose of having independent service functions.

To overcome this challenge, one does not require every microservice to have a separate physical database. Every service must, however, have complete ownership over its data. Microservices would require every type of database: NoSQL, graph, and in-memory. Therefore, having a relational database would not bring out the desired results in a microservice architecture.

02

The Problem of Inter-Service Communication Breakdown

The various microservices that rely on each other within an application software communicate using well-defined APIs. However, these do not share the same technology stacks, frameworks, or data libraries. For smooth internal communication, developers are required to set up infrastructure layers that enable effective resource sharing across multiple services. The method of communication and interactions need to be explicitly defined. It demands principles for security, serialization, error handling, requests, etc. Therefore, a poor configuration can lead to increased latency, thereby invalidating the purpose of microservices.

As a result, companies deploying microservices architecture would require some form of service mesh capabilities like reliable CI/CD servers, application performance management tools, configuration management platforms, etc. to ensure effective internal communication, among services

03

Need for Expert Teams

Transitioning to a microservice architecture requires expert teams who have experience working with distributed systems. Switching to microservices could fail if the organization has an ill-prepared design and development team. Moreover, the team also needs to promptly coordinate with independent teams involved in developing the stand-alone service functions. To cover the whole cycle and get the microservice architecture smoothly running, all teams must apply concepts like CAP, BASE, functional interfaces, CQRS, and sagas.

To accomplish error-free integration and data persistence, teams also require a holistic understanding of concepts like persistence ignorance, polyglot persistence, and event-driven messaging. Lastly, organizations also need developers with a good understanding on how to build and maintain polyglot microservices using complex CI/CD pipelines.

04

Difficulty in Maintaining Microservices

Microservices bring flexibility by allowing the use of diverse programming languages and different technological bases, within a single business application software. When the software transitions into maintenance mode, the flexibility and ease may end up being shadowed by high maintenance costs due to the diverse tools and technologies used.

Therefore, to prevent such costs, developers must ensure that a microservice failure should not bring down the entire system. All design rules must be strictly followed and checked with new versions of microservices before they are updated to prevent any form of breakdown, affecting the entire application.

05

Operational Complexity Leading to Poor Network Management

An application software designed using the microservices architecture approach entails a multitude of modular services. In this case, managing all these services would require some serious, disciplined efforts to avoid failovers and ensure that the system is resilient to changing business demands. If there is a sudden spike in application usage, coordination of services becomes even more challenging. Moreover, if a single microservice fails, it may cause a cascading effect on the remaining services.

Thus, developers must ensure that each of these microservices are resilient in themselves, without relying on other services. Sophisticated management tools should be used for automated provisioning, making the entire system bullet-proof and agile. By deploying and using the right kind and quality of automation software, tools and technologies, CI frameworks and expert staff, every business can overcome most of the above-mentioned challenges, in their quest to adopt the microservice architecture. With the right support and knowledge, every business having large applications can successfully break their infrastructure down into smaller components, thereby bringing flexibility and potential for higher scalability.

06

Need for Rapid Provisioning and App Deployment

As microservices enable incremental development and continuous delivery, organizations' staff also need to be able to provision resources instantly to keep up. You should also be able to deploy new services or applications quickly to make the most of microservices.

07

Robust Monitoring Requirements

Since each service in a microservices application uses its own language, platform, and APIs, multiple teams will be working simultaneously on different services. If any one service or machine fails, it will be difficult, if not impossible, to track down issues in retrospect. Therefore, robust monitoring is essential for effectively monitoring and managing the entire infrastructure.

08

Complex Testing Demands

Each service in a microservices app has its own dependencies. The addition of new features may cause new dependencies and increase complexity. To ensure effectiveness, microservices architectures need to have resiliency testing and fault injection to handle network latency, database errors, caching issues, service unavailability, etc.

09

Designing with Failure in Mind

While creating a microservices architecture, it is important to factor in possible failure issues like slow service, system downtime, and unexpected responses. Despite load balancing measures, the company must also have an alternate plan if a failure arises. In such a situation, the affected service should be able to keep running with degraded functionality rather than crashing the whole system.

10

New Acceptance of DevOps Culture

In traditional companies with monolithic apps, developers were solely focused on features and functionalities, while ops teams worked on production challenges. Microservices apps, however, require businesses to work in cross-functional teams under the common banner of DevOps, where everyone is responsible for both service provisioning and failure.

MICROSERVICES ON FIORANO’S HYBRID INTEGRATION PLATFORM

Fiorano's Hybrid Integration Platform deploys microservices, high-speed messaging, and a unique peer-to-peer architecture to design a customized integration and API management strategy for digital organizations requiring real-time solutions. Fiorano’s digital strategies are suitable for every kind of business environment. It ensures high control and regulation compliance (suitable for systemic environments), as well as federated, high-speed service solutions (suitable for an adaptive business environment).

With increasing complexity of business environments, the significance of business data has skyrocketed. Every organization is striving to collect data from thousands of endpoints, leading to large volumes of data generation, requiring speedy velocity for processing the same. Fiorano offers an integrated solution that proliferates cloud, mobile, social, and big data projects into a single API, backed by multiple, independent microservices.

KEY FEATURES OF FIORANO’S MICROSERVICE ARCHITECTURE

Fiorano deploys unique software development architecture styles in the form of independent microservices, that are device and platform agnostic. Instead of developing centralized applications in the monolithic format, Fiorano integrates multiple microservices into a single application that has data back-ends accessible on all mobile, the cloud, and other devices. This microservice architecture is an improved or specialized variant of the Service Oriented Architecture (SOA) that enables businesses to promptly connect or adapt to their dynamic environment.

01

Microservices are independent 'Products'

Every microservice available on the Fiorano platform is offered as an independent, self-contained product, having all its dependencies (e.g., libraries, etc.) within a single package. Every microservice offering is developed, tested, and deployed independently of the other microservice products. Additionally, upgrades or modifications to any service offering can be made without disrupting the smooth functioning of other services. Lastly, modifications to microservices can be made during the run-time, without stopping the whole application for a single microservice. Therefore, microservices make applications truly decentralized or federated.

02

Process-Specific, Coarse-Grained Components

Each of Fiorano's microservices represents an independent process, which can be run alone or alongside other processes within an application. Every microservice is like a coarse-grained component that needs to be set up inside an application, for a defined purpose. It accepts some inputs and provides a few outputs. These perform simple functions or services and are not meant to offer complex services involving hundreds of interfaces.

03

Multi-Language Support

As all microservices are completely decoupled from each other, they can be developed in any supported language. Fiorano allows the development of microservices in Java, C, C++ and C# language. This offers a high degree of flexibility and eases all developers as well as deployers.

04 Business Logic Management

Fiorano’s microservice architecture acts as an agent between the back-end data storage applications like the cloud, and the organization's internal network/IT architecture. Microservices manage business intelligence by enabling ‘infrastructure specific’ functions like routing, coaching, application of business rules, etc. On Fiorano, applications are built by choreographing microservices, by using industry standard protocols like REST, SOAP, JMS etc.

05 Infrastructure Automation

Every application created on the Fiorano platform by assembling microservices, can easily move across the various environments, namely- development, QA, staging and production, in one click. The profiles of each of these environments and the relevant microservices are set up in advance, thereby allowing the platform to search for and apply the accurate profile at runtime. This movement across environments is achieved using graphical tools or scripts.

06 Focus on Business Capabilities

Fiorano offers independent microservices as a product for specific application functionalities. Therefore, every time a large system application is to be built, developers map out and integrate individual microservices, and deploy the microservices. If a specific application function or component is to be modified, it can be done easily by modifying the specific microservice component.

07 Easy Failure Detection and Exception Handling Services

The Fiorano platform has deployed JMS messaging as the default transport solution across Microservices, at runtime. Unlike monolithic application architectures, in the case of Fiorano, microservices are independently developed and deployed.

Therefore, handling failures on applications becomes comparatively simpler and targeted to specific microservice products. Fiorano deploys a central exception handling microservice for every single application. The exception handling microservice is responsible for the smooth functioning of the application flow.

It monitors and configures exception alerts in real-time, using specific, measurable parameters like backlogs on queues (pending messages to be processed by the microservice), run status of components, and so on.

08 Decentralization of Data Management

Every Fiorano microservice manages its own data with aid from the business logic implemented in the microservice. Each service offering has its own persistent datastore accessible by that service itself. Therefore, there is no centralized database system for these applications. All information is managed as a part of the payload that flows among the various microservices, distributed across the platform.

09 Decentralized Governance

Since every microservice offering on the Fiorano platform is independent of other services, the overall governance of any resultant application is also truly decentralized. Every microservice is governed, modified, upgraded, deployed or even replaced independently, thereby ensuring flexible and smooth running of the entire application with its many microservice bundles.

10 Microservices Integration

Individual microservices can be configured, managed, updated, and redeployed independently without disrupting other processes or services.

11 Elastic Deployment

Regardless of size, type, and location, systems can be integrated with cloud-to-cloud, cloud-to-on-premises, and on-premises-only environments.

12 Security

Controlled service deployment across network endpoints follows a distributed security model.

13 Integration as per Industry Standards

Integration is carried out only after industry standards for connectivity, communication, transformation, and security application interoperability are met.

CONCLUSION

Fiorano enables businesses to develop highly modular and reusable microservices. Developers can easily avail flexible solutions for every single process required within an application.

It not only saves costs and time but also helps to create applications with flexible scalability features. Additionally, the platform comes with tools that offer business users the luxury of visual drag-and-drop design elements, to monitor, create and modify their business applications and processes.

Therefore, all alterations to existing applications can be made as quickly as in real-time.

The technology also offers a self-healing solution for node failures, making the application resilient and robust to external as well as internal factors that require real-time exception-handling services; thereby making the application highly adaptable to the constantly changing business environment.

ABOUT FIORANO

Connecting People Through Technology

Fiorano is a leading enterprise integration, microservices and API management software provider.

Our products and solutions connect applications, devices, and data to streamline business processes, helping companies scale, adopt emerging technologies, and improve customer experiences.

Fiorano operates worldwide through its offices in 9 countries and network of partners across the globe.

Drop us a line:

www.fiorano.com
info@fiorano.com