Microservices

- Part of **Cloud Native** journey

Our OKRs?

- To become a Thought Leader in providing services
  - Business placing increased demands to IT
  - IT department is not able to deliver future innovation
  - How can we provide SAAS products

- To become financially independent
  - Accelerate services to Time to market
  - Reduce IT operational costs
  - Minimize execution wastes

How can we achieve our goals?

- To meet these demands,
  - IT must digitally transform the enterprise through the adoption of **Cloud Native** practices
  - Cloud native practices allows to optimize and transform the existing infrastructure and applications
  - IT modernization is crucial to addressing emerging requirements of the digital business
  - Expect to deliver business results like selling our bill break down as a service to potential customers (SAAS)
  - Reducing business operating costs, which can boost the net income.
  - Faster time to market can boost sales and increase revenue

What is Cloud Native ?

- Best practices to adopt as part of our IT strategy
- Cloud Native is an ongoing journey; adopt and adhere over time
- The more cloud native we become, greater the benefits
- Increases the velocity, agility and innovation
- Gaining access to needed wider capabilities (prepackaged services)
- Two important terms
    1. **Cloud Native Applications**
    2. **Cloud Native Infrastructure**

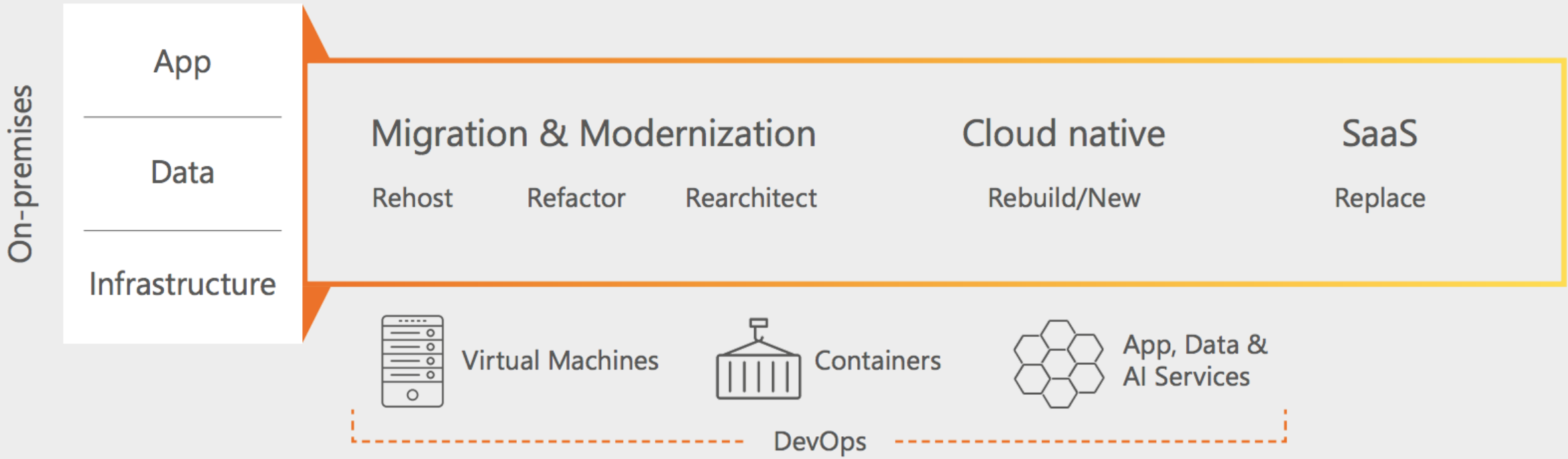
What is Cloud Native ?

- **Cloud Native Applications**
  - Architected, designed, developed, packaged, delivered and managed in a manner that is consistent with the cloud environment.
  - They are developed using DevOps practices, a **microservices** architectural style that takes advantage of PaaS platforms

What is Cloud Native ?

- **Cloud Native Infrastructure**
  - Standardized, scalable, multi-tenant environment that provides all of the hardware and services required to support applications;
  - Provides redundancy and failover capabilities to prevent disruption of service and application availability.

Journey to the cloud



Re-host:

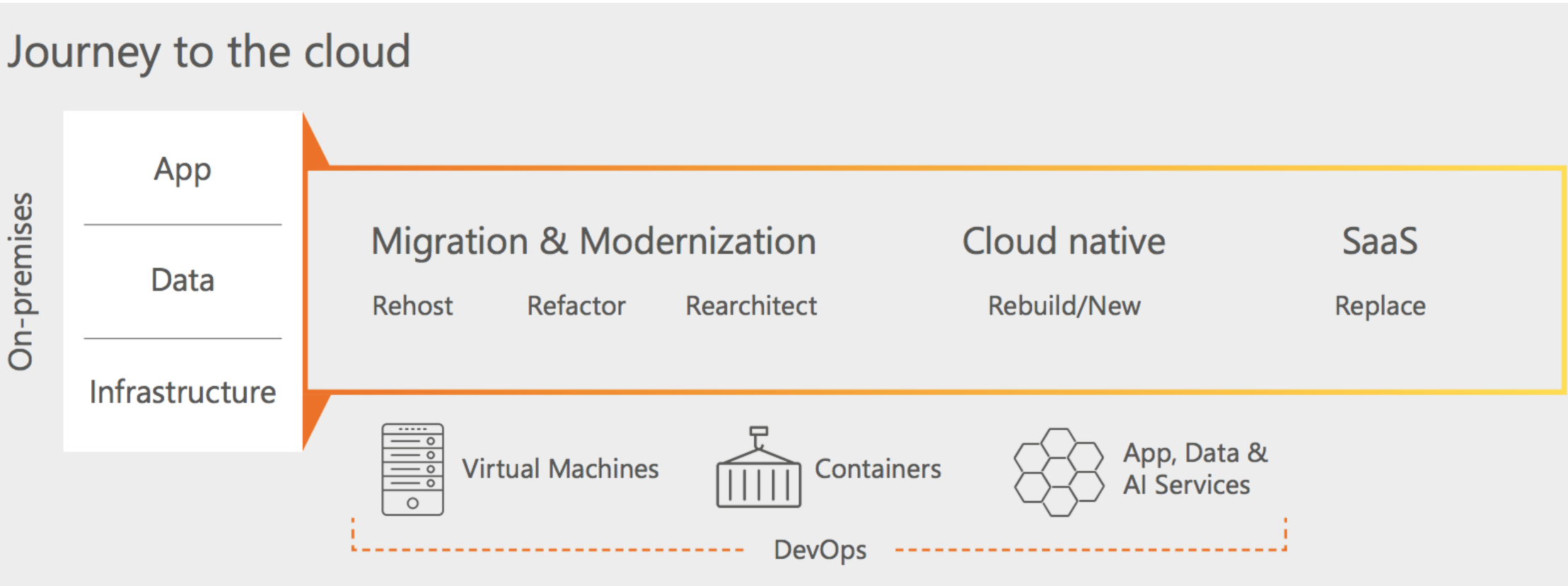
*Re-hosting is typically carried out with an infrastructure as a service (IaaS) offering, and is often referred to as a "**lift and shift**".*

-> With this approach you move your existing applications out of your data center and off your hardware into someone else's data center and on to their hardware

-> The quick and easy way to access the cloud without incurring a great deal of cost

-> NOTE: This is typically only the first step to realizing the full benefits of native cloud.

Journey to the cloud



Refactoring:

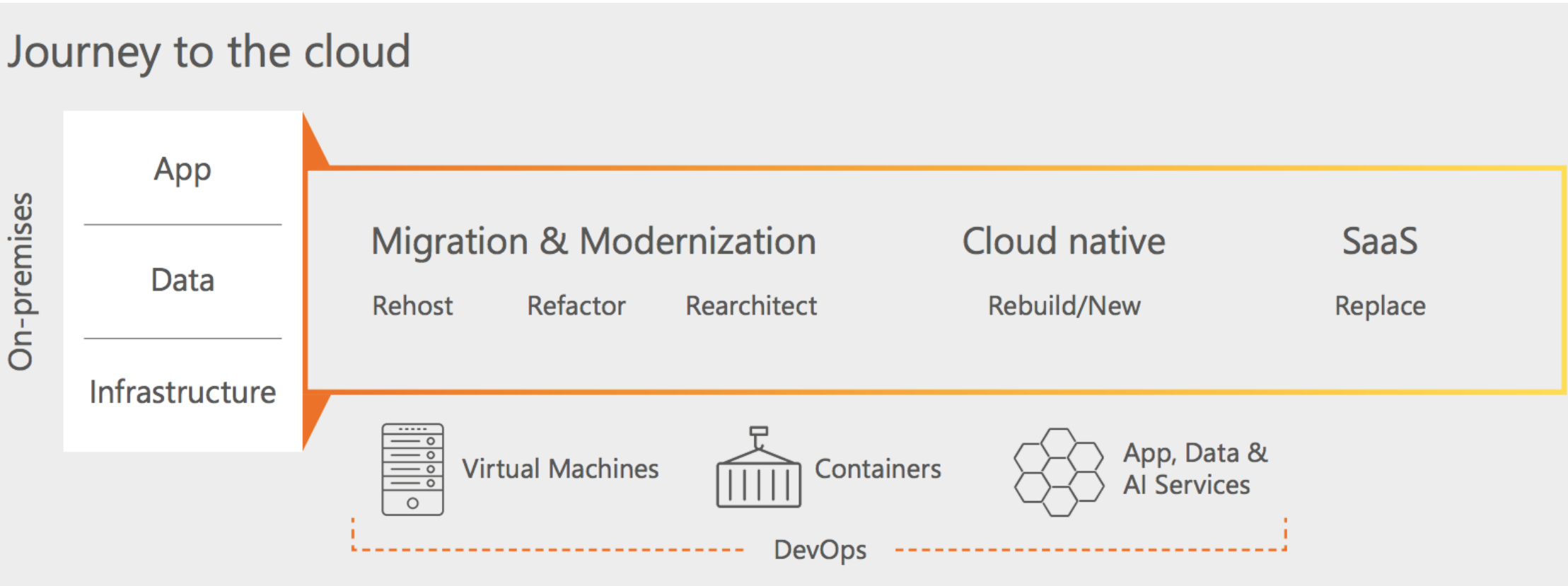
“Application refactoring is the process of altering an application’s source code without changing its external behavior”

-> The purpose of code refactoring is to improve many of the non-functional properties of the code, such as readability, complexity, maintainability and extensibility.

-> A popular approach to refactoring is the use of containers (Dockers)

->With containers you can speed up a data center exit without having to address updates to all of your applications at the same time.

Journey to the cloud



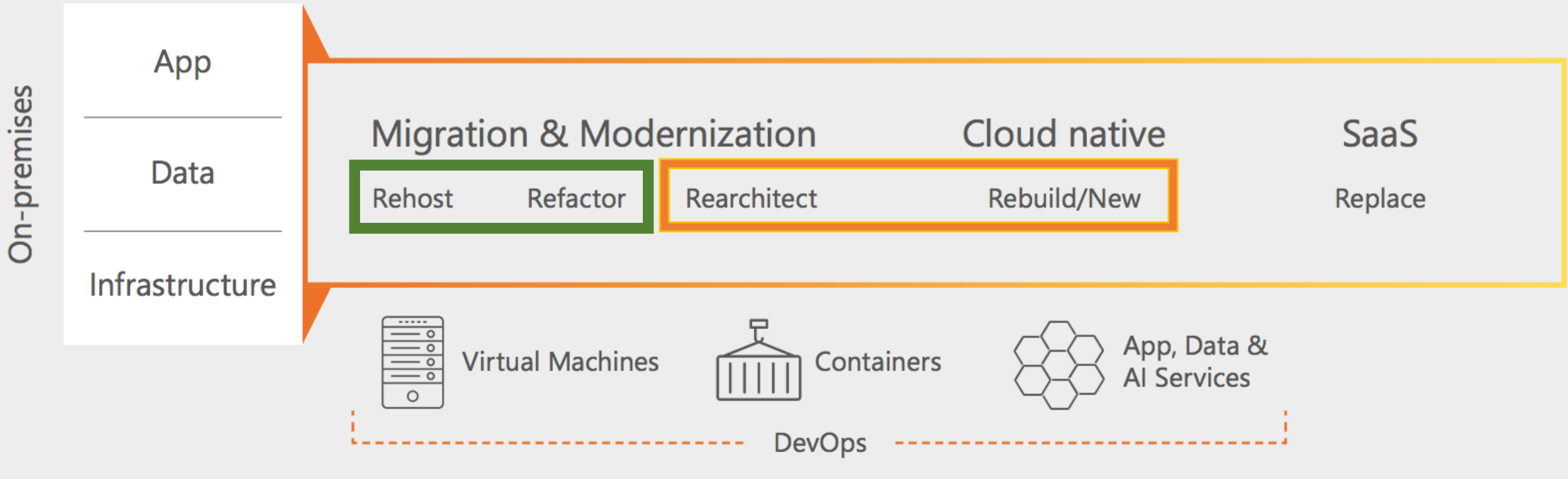
Re-architecting (making the old new)

*"Introducing cloud native aspects to the application, such as PaaS services, new architectural styles ("**microservices**"), and also altering the code to give it many of the characteristics and benefits of a cloud native application."*

-> Start with an existing application that is performing a valuable business function but due to its architecture is preventing you from being more responsive to changing business needs.

->These monolithic applications can be broken down into smaller **microservices**, which leverage modern frameworks and platforms

Journey to the cloud



Microservices

“Microservices architecture is essentially about building (or breaking up) an application into smaller, discrete pieces that exist independently from one another. ”

“Multiple services work together as a system to provide business valuable features”

This fundamental trait of microservices is also the basis of many of its benefits, especially relative to monolithic approaches to building and managing these services (applications).

5 Benefits of Microservices

1. Microservices can reduce risk and improve stability
2. Microservices can reduce time to market
3. Microservices are more scalable
4. Microservices enable greater technology flexibility
5. Microservices create greater team flexibility

Bonus Benefit:-

“The biggest benefit of microservices is that it allows multiple teams to deploy their changes independently without requiring much coordination between those teams,”

5 De merits of Microservices

1. Microservices is complex
2. Requires a stable team to run and maintain services
3. Choosing multiple language and frameworks (hard to maintain)
4. Transaction management can lead data inconsistency (nightmare)
5. More services – increases complexity – Interface can be broken

Note:-

“Deciding what capability to decouple when and how to migrate incrementally are some of the architectural challenges of decomposing a monolith to an ecosystem of microservices”

Our Microservice Strategy: 3 Pillars

1. API Centric
    - To serve distinct use case, build its solution around REST APIs

- Services
    - Single responsibility and distinct life cycle with automated testing to fit in microservices platform

- Events Based
    - Services should support asynchronous communication by events

Tech Stack & Infrastructure

- Java
- Spring boot
- Java + GraalVM
- Quarkus
- Managed Kubernetes
  - backed by big companies
  - It is going to stay here for at-least next 5-8 years

Conclusion

- Moving to cloud native microservices architecture
  - Accelerates the innovation
  - Helps business demands
  - Enables us to become thought leader in providing services (OKR)

- Moving to cloud native multitenant infrastructure
  - Keeps the TCO in check
  - We can be financially independent (OKR)