100 XP

Performance constraints of a monolithic application

5 minutes

There are many reasons that you might choose to change the architecture of a system. Operational agility, cost, scalability, and performance are just some of the factors that play a role in determining the architecture of a system. In our example, we'll take a closer look at how performance becomes a factor for the drone shipping system.

As the Fabrikam drone shipping business grows, system load increases. The current architecture is straining under the load. Fabrikam wants to provide better flexibility in scaling the application that isn't available in the current monolithic architecture. Improving the application's scalability is one of the drivers for Fabrikam to look at moving their application to a microservices architecture.

Scaling monolith vs. scaling microservices

One of the primary benefits to a microservices architecture comes in the increased scaling capabilities. Because services are separated out, it's much easier to scale each service individually as load increases across them.

We can see this difference in capabilities in the drone delivery system. With a monolithic architecture, all services are contained within a single instance of the application. They expose an API interface to customers to submit and manage delivery requests. As customer requests increase, load on the system increases. More resources are required to be allocated to the system to avoid negatively affecting the end-user experience.

In a monolithic architecture, scaling this service individually also requires scaling the resources for the other services because they're contained within each application instance. This arrangement is inefficient because load for the other services might be minimal and not require the additional resource utilization.

In a microservices architecture, because each service is separate, we can scale the API independently of the other services. This arrangement increases efficiency because we don't need to consume the resources of unnecessary services.

Challenges with the Drone Delivery monolithic architecture

The package service was identified as a critical part of the business and was originally part of the monolith. Customers have dramatically increased their reliance on it, and performance is negatively affected as this load increases. To address this situation, Fabrikam has dedicated a development team with full control over this piece of the business. The team will develop and iterate on this service and be solely responsible for all aspects of the package service.

Because the package service is in the monolith, the team can't operate autonomously. They have to rely on shared data and data structures. They're also unable to iterate as quickly as they need. Along with the performance and scalability issues, this service is identified as a prime candidate for a microservice.

Let's take a closer look at how Fabrikam can analyze and decompose their application to take advantage of a microservices architecture.

Next unit: Analyze an application and identify decomposition boundaries

Continue >