**Monolithic to Microservices -why or why not ?**

**Noopur Agarwal – IBM**

[**noagarwa@in.ibm.com**](mailto:noagarwa@in.ibm.com)

**+91-8055743507**

# SESSION ABSTRACT

Monolithic architecture has been the backbone of all the enterprises softwares till now.Client+ Server+Database and thats all.The stack is pretty straight-forward but it's not.Even a small change requires the entire monolith application to be rebuilt and deployed.Over time the monolith applications grew larger as business needs and as new functionality is added.So we moved to break monolithic applications to microservices which are smaller and simpler,can be built and deployed independently. Microservices can be scaled quickly and precisely to production.Its an architecture which embraces DevOps practices.If there is a demand scale the service instances to meet.

But having microservices is not always win-win.it has its own challenges.Developing distributed systems is complex.Refactoring a monolith to microservices creates many small components that constantly communicate,the complexity is interconnections between services.Microservices gives flexibility of developing and deploying the code whereas a monolith is for simple, lightweight applications but components cannot be scaled independently.

# SESSION FLOW

15-20 mins

# AUDIENCE

Students,.industry experts ,software enginners ,cloud enginners , seniors ,etc.

This can be useful for beginner or intermediate .

• Beginner – The session is targeted for individuals with basic knowledge in the subject and is appropriate for a beginner in the area. This level is ideal for submissions targeting an introduction to a particular technology/subject.

• Intermediate – The session is targeted for those with prior knowledge and some working experience on the topic. The speaker will expound the latest advancements or detail the topic further to help attendees develop a greater understanding.

**INTRODUCTION**

Monolithic architecture has been the backbone of all the enterprises softwares till now.Client+ Server+Database and thats all.The stack is pretty straight-forward but it's not.Even a small change requires the entire monolith application to be rebuilt and deployed.Over time the monolith applications grew larger as business needs and as new functionality is added.So we moved to break monolithic applications to microservices which are smaller and simpler,can be built and deployed independently. Microservices can be scaled quickly and precisely to production.Its an architecture which embraces DevOps practices.If there is a demand scale the service instances to meet.

But having microservices is not always win-win.it has its own challenges.Developing distributed systems is complex.Refactoring a monolith to microservices creates many small components that constantly communicate,the complexity is interconnections between services.Microservices gives flexibility of developing and deploying the code whereas a monolith is for simple, lightweight applications but components cannot be scaled independently.

**Benefits**

**1.**Need to release an update to your order service? Well, it’s “micro,” so it’s simple to understand, and its developers can focus on just that functionality. And when it’s ready, you can just test and deploy that service – no need to bundle it up with the rest of your application, and no waiting for a release window.

**2.**Is the same service creating bottlenecks when demand spikes in the evenings? Scale out by deploying a few more instances of just that microservice. This is very different from monolithic applications, which may have very diverse requirements but must still be deployed together as a single unit.

**3.**Microservices are just as much about team process and organization as technology. From Conway’s Law, microservices are changing how teams are structured, allowing organizations to create teams centered on specific services and giving them autonomy and responsibility in a constrained area. This approach helps the company rapidly adjust in response to fluctuating business demand, without interrupting core activities. It also makes it easier to onboard new staff quickly.

**4.**Microservices are simpler, developers get more productive and systems can be scaled quickly and precisely, rather than in large monolithic blobs; an architecture that embraces DevOps practices.

**Challenges**

**1.**But, microservices are not the silver bullet that will solve all architectural problems in your applications. Developing distributed systems is complex. More granularity means more moving parts. Refactoring a monolithic application to microservices creates many small components that constantly communicate; the complexity is shifted around to the interconnections between services.

**2.**When more services are interacting, you increase possible failure points. Smart developers stay one step ahead and plan for failure.

**3**.Tracing performance problems across tiers for a single business transaction can be difficult. This can be handled by correlating calls with a variety of methods including custom headers, tokens or IDs.

**4.**Distributed logic with distributed data increases the effort of finding the root cause of issues. Traditional logging is ineffective because microservices are stateless, distributed and independent — you would produce too many logs to easily locate a problem. Logging must be able to correlate events across several platforms.

**5.**Operational complexity is also increased due to the increased demands on managing these services and monitoring them. The ability to quickly deploy small independent services is a win for development, but it puts additional strain on operations as half-a-dozen applications now turn into hundreds of little microservices. Coordinating a large number of rapidly changing services necessitates automated continuous integration and continuous delivery**.**

**OUTCOMES/CONCLUSION**

When to move from monolithic to microservices

**PARTICIPATION STATEMENT**

yes

# BIO

Noopur is having 10 years of experience in Cloud and also a practitioner of microservices .She has been working on Agile and Cloud technologies for these many years .She has immense knowledge of microservices architecture ,docker ,kuebernetes and using this in her recent projects .She has been speaker in various industry events for mircroservices ,app modernisation ,and other cloud related technologies..