© Memi Lavi www.memilavi.com memi@memilavi.com

Software Architecture Series

Microservices

Check List

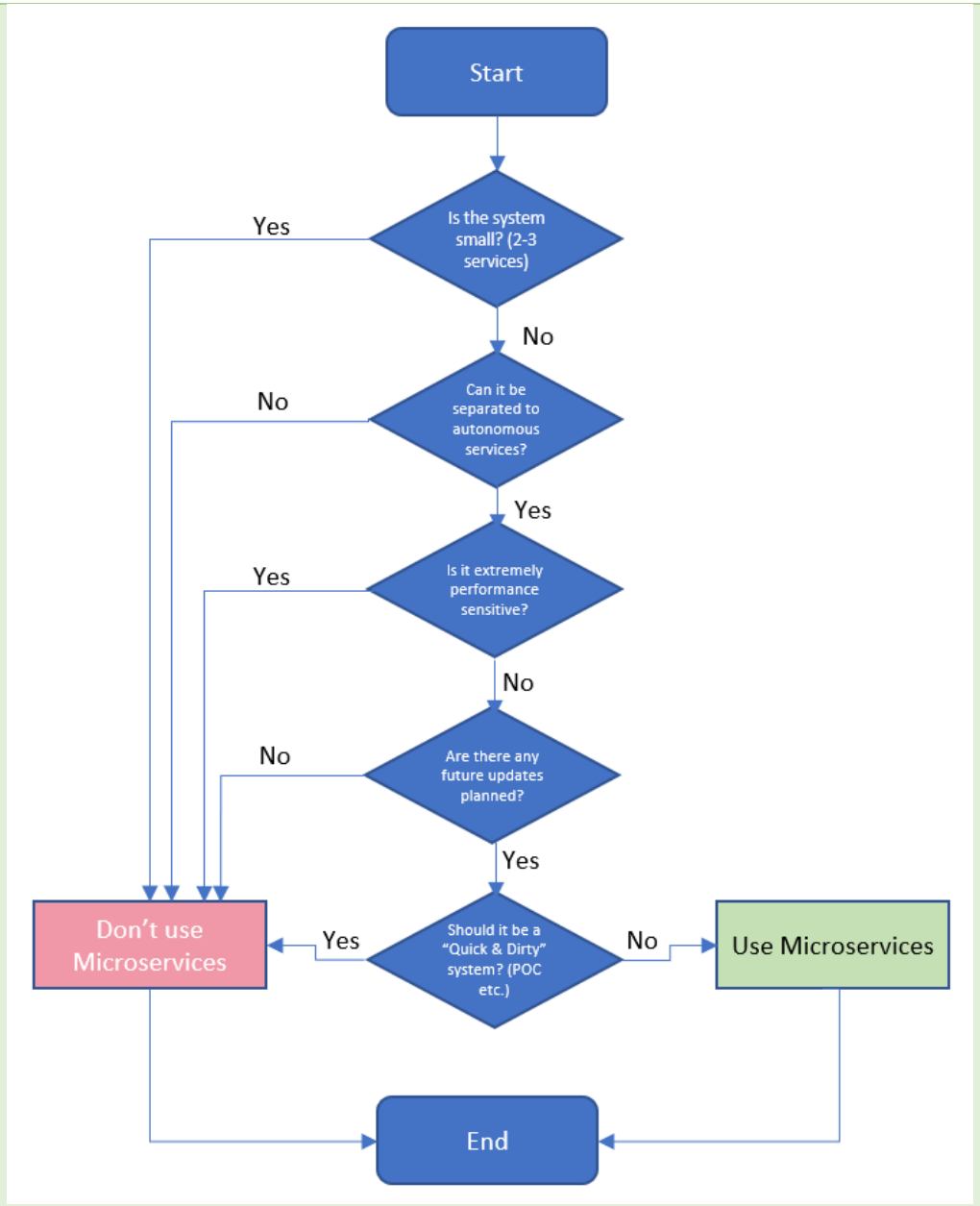
When designing Microservices architecture, use this checklist to make sure your Microservices architecture is robust and well-designed.

Topic	Description	Remarks
9 Attributes of Microservices	Based on Martin Fowler's article, these are the 9 attributes of Microservices: 1. Componentization via Services 2. Organized Around Business Capabilities 3. Products not Projects 4. Smart Endpoints and Dumb Pipes 5. Decentralized Governance 6. Decentralized Data Management 7. Infrastructure Automation	- Not all attributes are mandatory - The "Decentralized Data Management" is the most controversial, but also one of the most important - Even though REST API is mentioned in the

	8. Design for Failure 9. Evolutionary Design	article, there are other, modern API you should explore, such as GraphQL and gRPC.
Microservices Architecture Process	Follow these 4 steps when designing Microservices architecture: 1. Mapping the components – Decide what are the services and their boundaries in the system. This is perhaps the most important part of the process, and it's rarely reversible. 2. Set Communication Pattern -Design the methods for communicating with each service. First – decide if this service is synchronous or asynchronous, and then design the API. 3. Select the Technology Stack – Decide on the development platform, database, cache etc. of each service. Remember that each service can be technologically independent from other	

	services (“Decentralized Governance”) and they should not be based on the same stack. 4. Design the Architecture – Design the inner architecture of each service, usually using the layers paradigm.	

Microservices Flow Chart



Service Mesh	Goal: To abstract all communication aspects between the services. Service Mesh is an in-process / sidecar component that sits near the service and performs all communication aspects between services. Provides the following services: - Protocol Conversion - Security - Authentication & Authorization - Reliability - Monitoring - Service Discovery - Testing - Load Balancing	

	Uses Data Plane to actually perform the communication, and Control Plane to configure and monitor communication. Popular Implementations: - Sidecar: Istio, LinkerD - In-Process: DDS Use when there are a lot of services or if there are complex communication requirements.	
Breaking Monolith to Microservices	3 Approaches: 1. New Modules as Services – Add new modules as services, do not modify existing code 2. Separate Existing Modules to Services – Convert existing modules to services 3. Complete Rewrite – Discard existing code and rewrite the whole system as Microservices-based system	Choose your approach based on the complexity of the Monolith. The higher the complexity – the more attractive the rewrite.

I Hope you enjoyed the course, and that it made you a Microservices expert. I'm sure you'll now be able to design modular, robust systems, and that it made you a better Architect!

For any question or comment contact me at:

memi@memilavi.com

Thanks,

Memi