**Questions about Service Oriented Architecture and Microservices:**

**Why, in a SOA, long-lived transactions are discouraged and Sagas are suggested instead?**

*https://blog.bernd-ruecker.com/saga-how-to-implement-complex-business-transactions-without-two-phase-commit-e00aa41a1b1b*

**What are the differences between Soa and Microservices?**

*The other microservices will continue to handle requests. ... Last but not least, the main difference between SOA and microservices lies in the size and scope. Microservice has to be significantly smaller than what SOA tends to be and mainly is a small(er) independently deployable service.*

**Let's talk about web services versioning, version compatibility and breaking changes.**

*Web services are bound to change and evolve over time. ... Introduce different patterns that can be considered for Web services versioning. Provide guidelines for applying those patterns in real-world solutions.*

*I prefer the Salesforce.com method of versioning. Each version of the Web Services gets a distinct URL in the format of:*

*http://api.salesforce.com/{version}/{serviceName}*

*So you'll have Web Service URLs that look like:*

*http://api.salesforce.com/14/Lead*

*http://api.salesforce.com/15/Lead*

*and so on...*

*With this method, you get the benefits of:*

*You always know which version you're talking to.*

*Backwards compatibility is maintained.*

*You don't have to worry about dependency issues. Each version has the complete set of services. You just have to make sure you don't mix versions between calls (but that's up to the consumer of the service, not you as the developer).*

**What's the difference between a transaction and a compensation operation in a saga, in SOA?**

[*https://medium.com/@tomasz\_96685/saga-pattern-and-microservices-architecture-d4b46071afcf*](https://medium.com/@tomasz_96685/saga-pattern-and-microservices-architecture-d4b46071afcf)

**When is a Microservice too micro?**

[*https://www.altoros.com/blog/over-engineering-when-microservices-are-too-micro/*](https://www.altoros.com/blog/over-engineering-when-microservices-are-too-micro/)

**What are the pros and cons of MicroService architecture?**

*Pros:*

*It is easier to deploy individual Microservice compared to a monolith.*

*It is easier to develop a Microservice for a specific task.*

*It is much easier to change the underlying implementation of a Microservice.*

*It is easier to scale a Microservice.*

*It is easier to debug any issues with a specific Microservice.*

*Cons:*

*It is very difficult to maintain multiple Microservices.*

*It is very difficult to maintain versioning among Microservices.*

*It is extremely difficult to find good architects for creating Microservice architecture in right way.*