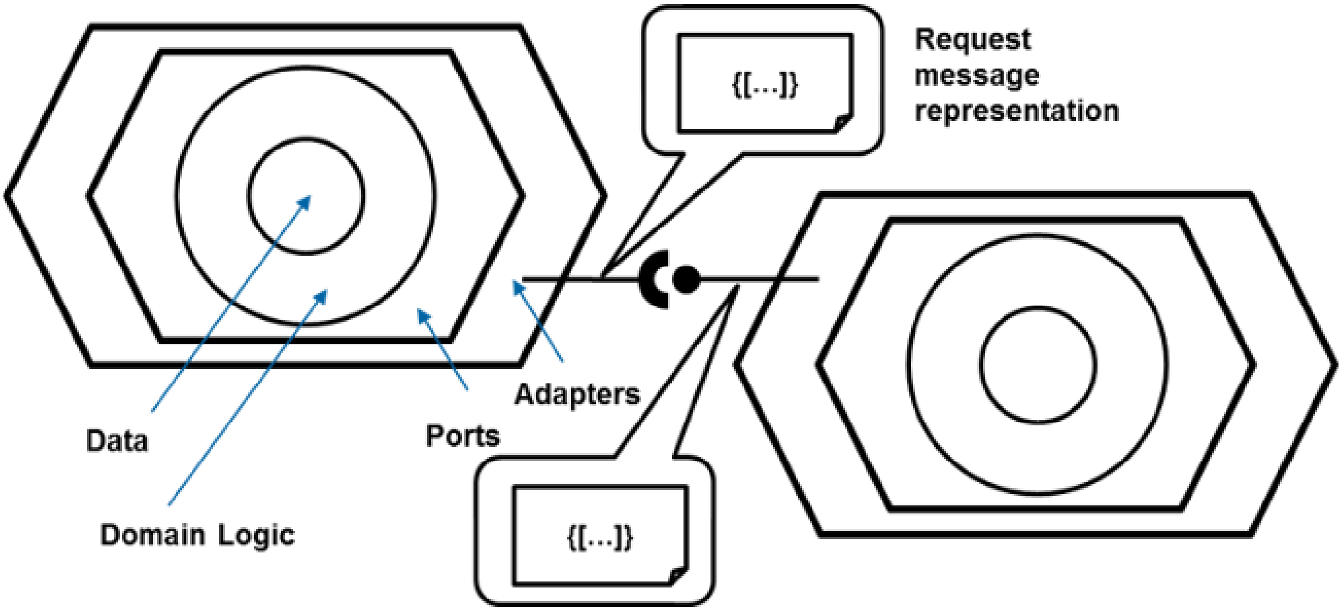
Microservices vs. SOA

Three positions and evidence for one of them

Doc SoC Follow
Jul 8 · 3 min read

Microservice architectures evolved from previous incarnations of Service-Oriented Architectures (SOAs) to promote agility and elasticity. They are often visualized with hexagons and onions (to emphasize service autonomy):



I see such microservices (defined by seven tenets here) as a contemporary implementation approach for Service-Oriented Architecture (SOA) principles and patterns, emphasizing and benefiting from modern software engineering practices such as domain-driven design, containerization and DevOps.

Please read on if you would like to learn why — or disagree.

Revolution! Stagnation! Evolution!

Three rather different positions can be observed, both online and in print:

1. Microservices as a *new architectural style* competing with SOA and to be contrasted with it (“better SOA”). See this example of such positioning.
2. Microservices as a *synonym for SOA* (“nothing new”), see for instance this blog post and this one.
3. Microservices as a *substyle, variant, and/or implementation approach to SOA* (“SOA done right”), see several positions from a SATURN 2015 microservices workshop.

Radically Different? Nothing New? Somewhere in Between.

Going through the various online resources, I found — and continue to find — much evidence for Position 3:

- In this Software Engineering Radio interview, James Lewis highlights the close connection from/to SOA (assuming that SOA is understood and done right).
- In this goto 2014 presentation, Martin Fowler points out that it is fair to say that the microservices approach has been done under the name of SOA for at least a decade if not more (with the term SOA being too broad, and meaning different things to different people). Note that this also holds for microservices now — semantic diffusion striking again!
- In his SATURN 2017 keynote, Chris Richardson makes similar remarks and defines (micro-)services as *loosely coupled* and organized around *business capabilities*, which very much is in line with my SOA definition, as well as SOA introductions such as that in my OOPSLA tutorials (slides).

More Evidence for Position 3

Still not convinced? A Jan./Feb. 2017 interview with James Lewis, Mike Amundsen and Nicolai Josuttis in IEEE Software, facilitated by the Insights department editors (note: I am one of them), contains a side bar that comes to similar conclusions. The article also discusses the relationship of Domain-Driven Design and microservices and other service design issues. Part 2 of the interview covers architectural and organizational concerns such as service composition, data integrity and versioning.

You need even more arguments and rationale? Here are two recent blog posts pointing out that services may have any size — and that modular monoliths very much have a place in the modern architect’s toolbox:

- Kyle Brown (IBM Cloud Garage) and Shahir Daya ask What's the right size for a Microservice? and comment that services should be small — but not too small (here on Medium). They even go back to the roots of the word "micro" in ancient Greek to make their case!
- Vaughn Vernon's (author of "Implementing Domain-Driven Design") advises that "following a purpose leads to useful cohesion" in Microservices and [Micro]services (and it is worth noting that some purposes simply cannot be codified in a few lines of code).

I could not agree more! Context and requirements drive the architectural decisions about service granularity and size; for instance, 16 coupling and cutting criteria are collected in the Service Cutter wiki, and I am working on a light domain-driven service design method right now (demo).

Outlook

Publication/research and development projects that I currently lead include Microservice API Patterns and Microservices Domain-Specific Language (MDSL) and Design Practice Repository (DPR). You can read more about them in my personal blog.

I hope you found this post useful (and worth sharing?). An extended version of it (that also contains the seven tenets) appears here.

Whether you agree or disagree with my analysis, I'll be very interested in your arguments, please let me know what you think!

Microservices Soa Eai DevOps Containers