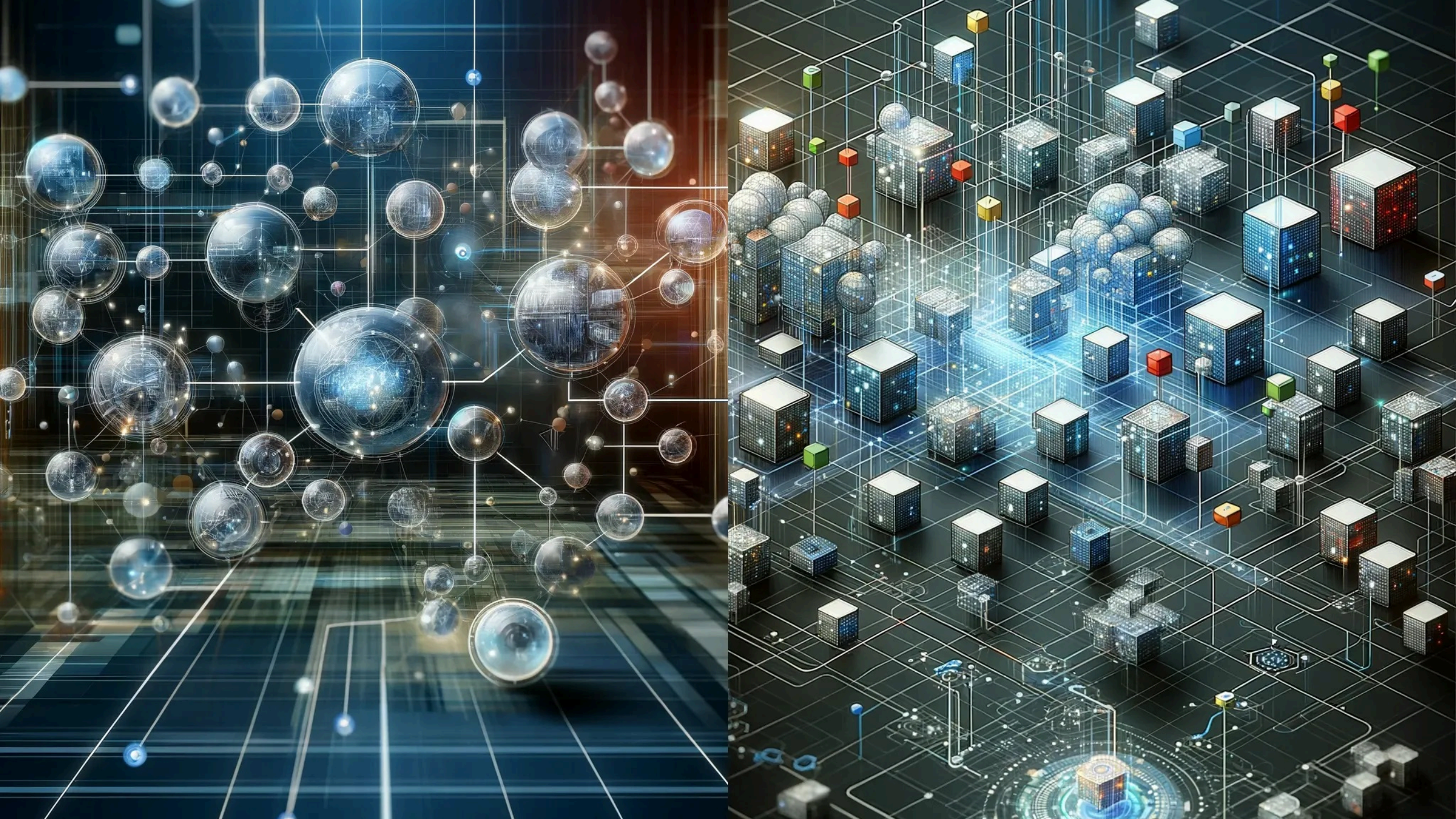


Hvad er Microservices?

Definition

Microservices er en arkitekturstil, der strukturerer en applikation som en samling af små, autonome tjenester, udviklet omkring specifikke forretningsmål.

Egenskaber

- Hver mikroservice er selvstændig
- Kommunikerer via lette protokoller
- Uafhængigt deployable

Hvad er SOA?

Definition

SOA er en arkitektonisk model, der anvender distribuerede tjenester, som kommunikerer over et netværk for at tilbyde funktionalitet til applikationskomponenter.

Egenskaber

- Tjenester er løst koblet
- Kan involvere flere applikationer og datakilder
- Fokuserer på genbrug af eksisterende tjenester

Fordele ved Microservices

- Fleksibilitet i udvikling og vedligeholdelse
- Skalerbarhed og fejltolerance
- Hurtigere deployment og innovation

Ulemper ved Microservices

- Kompleksitet i styring og opsætning
- Udfordringer ved transaktionshåndtering
- Potentiel overhead for inter-service kommunikation

Fordele ved SOA

- Integration på tværs af forskellige systemer og platforme
- Genbrug af eksisterende tjenester
- Mulighed for at håndtere store og komplekse systemer

Ulemper ved SOA

- Tung og kompleks governance struktur
- Risiko for performance-problemer på grund af netværkskald
- Kan blive dyrt og ressourcekrævende at vedligeholde

Integration af Microservices og SOA

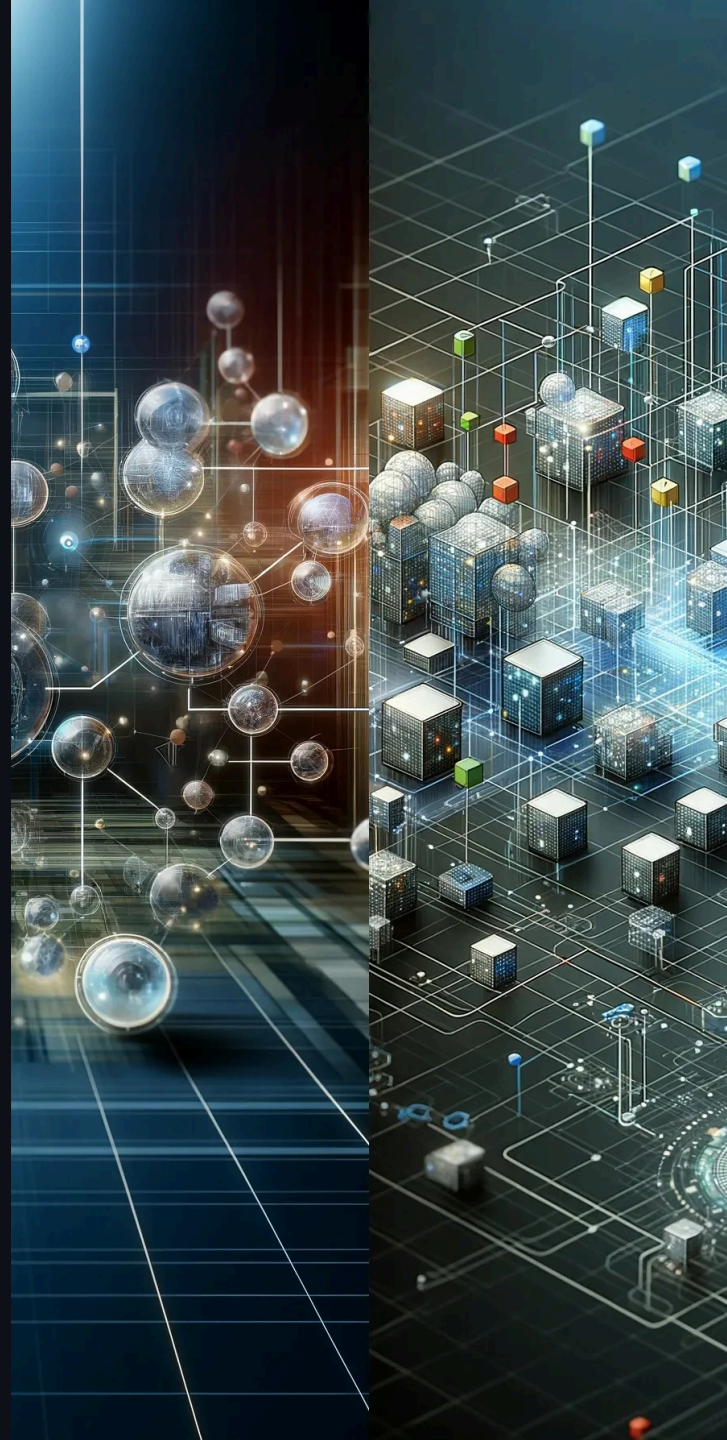
Komplementære styrker

Brug

- *SOA til at facilitere store systemintegrationer*
- *Microservices håndterer specifikke forretningsbehov.*

Teknologisk stack

- Brug af API Gateways, Service Meshes, og containeriseringsteknologier til at understøtte begge arkitekturer.



Eksempel - Finansiel teknologivirksomhed

Problem

- Integration af ny kundeservice platform med eksisterende legacy systemer.

Løsning

- SOA til at orkestrere tjenester
- Microservices til at implementere nye, selvstændige tjenester for bedre kundeinteraktioner.

Resultat

- Fleksibilitet i udviklingen af nye tjenester, forbedret skalering og vedligeholdelse, og effektiv integration med ældre systemer.

Konklusion

Opsummering

- Microservices og SOA kan succesfuldt integreres for at kombinere deres respektive fordele.

Anbefalinger

- Vurdering af forretningens specifikke behov for at vælge den mest passende arkitektur eller en kombination.

Microservices Diagramtyper

Component Diagram

- Viser hvordan en applikation er opdelt i individuelle services med deres afhængigheder. Det er nyttigt til at illustrere, hvordan services interagerer internt i en microservices arkitektur.

Deployment Diagram

- Skitserer hvordan forskellige microservices deployes på infrastrukturen. Det kan vise relationer mellem services og de underliggende maskiner eller containere.

Microservices Diagramtyper

Sequence Diagram

- Viser hvordan data flows gennem forskellige microservices ved specifikke operationer. Dette er nyttigt for at forstå interaktionerne mellem services under forskellige brugsscenarier.

Network Diagram

- Illustrerer det netværkslayout, der understøtter microservices, inklusiv gateways, load balancers og forbindelser mellem services.

SOA Diagramtyper

Service Component Architecture (SCA) Diagram

- Bruges til at fremvise komponenter og deres interaktioner inden for en SOA. Det kan inkludere services, referencer, bindings og komponenter.

Integration Flow Diagram

- Fremviser, hvordan forskellige services er integreret og kommunikerer med hinanden og med eksterne systemer. Dette er vigtigt for at forstå datastrømme og afhængigheder mellem services.

SOA Diagramtyper

Enterprise Service Bus (ESB) Diagram

- Dette diagram viser, hvordan ESB fungerer som rygraden i en SOA ved at facilitere kommunikation og middleware-løsninger mellem forskellige services.

BPMN (Business Process Model and Notation) Diagram

- Illustrerer forretningsprocesser og deres sekvenser, der er implementeret via SOA. BPMN-diagrammer er nyttige til at visualisere høj-niveau processer og hvordan de nedbrydes i individuelle services.

Fælles Diagramtyper

For begge arkitekturer kan følgende diagrammer også være relevante.

Architectural Diagram

- Viser den overordnede arkitektur og komponenternes placering.

UML Diagrams (såsom Use Case Diagrams)

- Bruges til at beskrive systemets funktionaliteter og de aktører, der interagerer med systemet.