



TIGERTEAM®

Service Orienteret Arkitektur



Udfordringer og problemstillinger

En liste over de udfordringer og problemstillinger der er ved indførelse af SOA



Generelt

om

SOA



Generelt, I

- ❏ For stor tillid til modenheten af de forskellige leverandørers produkter og specifikations implementationer
- ❏ For stor tillid til, at WS*-standarderne har gjort det nemt at integrere heterogene miljøer
- ❏ Nedvurderer værdien og vigtigheden af serviceprincipper og løbende justering af disse

Generelt, II

- ❏ Man undervurderer værdien, omkostningerne og vigtigheden ved en stærk service governance
 - ❏ Service versionering, orchestration, ændringer, etc.
- ❏ Man undervurderer hvor længe det tager at blive enige om hvilken service model der skal vælges
 - ❏ 'Canonical data model'/Ontology eller 'The one true schema'
- ❏ Man undervurderer omkostningerne ved manglende reliability og transaktioner
 - ❏ Hvornår skal man selv kode idempotence, retry, compensation – er der produkter der kan hjælpe og virker de på tværs af platforme?

Generelt, III

- ❏ Man forgaber sig i smarte features, der ikke tjener projektet
- ❏ Man overdesigner, fordi man kan
- ❏ Man prøver at få det hele på een gang
- ❏ Hvad med sikkerheden?



Test

test

TEST



Test, Test, Test

- ❏ Man glemmer at sikre ordentlige tværgående test data
- ❏ Man glemmer omkostninger ved fejlsøgning i en løst koblet verden
- ❏ Man glemmer at sikre sig tværgående audit trails
- ❏ Man mangler ordentlige test værktøjer til tværgående tests

Og

løsningen

ER!



Hvad kan man gøre, I

- ❏ SOA er et princip, ikke produkter eller Web services
- ❏ Definer Service typer (forretnings og tekniske) og lav klare aftaler om ansvar (hvor ligger forretningsreglerne, hvor ligger data, etc.)
- ❏ Definer Service Principper så hurtigt som muligt (navngivning, data formater, versionering, service governance, etc.)
- ❏ Fokuser på at etablere tværgående audit trails og logging
- ❏ Brug interne og eksterne erfaringer til at definere en grundlæggende arkitektur, der dækker løsningens kendte behov
 - ❏ Behovs dreven arkitektur

Hvad kan man gøre, II

- 🍯 Lav en "deep and narrow" arkitektur i de tidlige iterationer
 - 🍯 Tænk stort, implementer i små bidder efter behov
- 🍯 Lav proff-of-concepts eller spikes af de største tekniske udfordringer så tidligt som muligt
- 🍯 Prioriter test og automatisering højt
- 🍯 Vær kritisk overfor teknologi og produkt valg



TIGERTEAM®

LEAN THINKING

www.tigerteam.dk

Copyright 2007