Engineer

arkitektur

hvordan et systems dele passer sammen, og hvorfor

Hvad det er

Arkitektur, i software ofte kaldt softwarearkitektur eller systemarkitektur, er det sæt af grundlæggende begreber og egenskaber, et system har i sit miljø, båret af dets elementer, relationerne mellem dem og de principper, der styrer, hvordan det udformes, og hvordan det ændrer sig. Formuleringen følger definitionen i NIST SP 800-160 Volume 1 Revision 1, Engineering Trustworthy Secure Systems, udgivet i november 2022.

Hver del af definitionen gør et stykke arbejde. Elementerne er delene: tjenester, databaser, køer, frontenden, modulerne inde i dem. Relationerne er, hvordan delene afhænger af hinanden, og hvad der går imellem dem. Principperne er de regler, der afgør, hvor nyt arbejde skal ligge, og de holder elementer og relationer fra at glide, efterhånden som systemet vokser. »I sit miljø« gør brugerne, nabosystemerne og begrænsningerne til en del af arkitekturen, og at definitionen også taler om udvikling over tid, betyder, at en beskrivelse, der kun er sand den dag, den blev tegnet, er ufuldstændig.

Ikke alle designvalg er arkitektoniske. Gruppen bag formatet Architectural Decision Record (ADR) kalder en arkitekturbeslutning et begrundet designvalg, der imødekommer et funktionelt eller ikke-funktionelt krav af arkitektonisk betydning, og et krav har den betydning, når det har en målbar effekt på et systems arkitektur og kvalitet. Skabelonen arc42 er mere ligefrem: dens afsnit om beslutninger er til vigtige, dyre, risikable eller omstridte beslutninger.

En enkelt tegning viser aldrig det hele, så arkitektur beskrives i visninger. C4-modellen, skabt af Simon Brown, tilbyder diagrammer, der fungerer som kort, man zoomer ind og ud på: et systemkontekstdiagram, der viser, hvordan systemet står mellem sine brugere og de systemer, der omgiver det, et containerdiagram, der viser applikationerne og datalagrene inde i det, et

komponentdiagram for én container og, hvor det hjælper, et kodediagram. arc42 ordner den skrevne beskrivelse i tolv afsnit, fra mål og begrænsninger til beslutninger, risici og teknisk gæld.

Brug begrebet i en beslutning

Brug »Hvad er arkitektur?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Systemets form lagt fast på papir først, så delene passer sammen, og prisen er kendt, før nogen skriver kode.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Én infrastrukturplan til 14 afdelinger — 14 forskningsafdelinger på én infrastrukturplan

Én infrastrukturplan til 14 afdelinger Kontakt os →

Hvorfor det betyder noget

Den beslutning, arkitektur stiller et team over for, handler mindre om, hvilken struktur der skal vælges, end om hvilke valg der skal træffes nu, bevidst og skriftligt, og hvilke der skal stå åbne.

Afgør man et strukturelt spørgsmål tidligt, betaler man, før man ved ret meget: valget træffes med den mindste viden, projektet nogensinde får, og det gør de første uger langsommere. Lader man det stå åbent, bliver strukturen afgjort alligevel, én bekvem ændring ad gangen, af den, der sidst rørte koden. Vejen ud er at skelne mellem to slags valg. Nogle få er dyre at ændre, når først koden afhænger af dem: hvordan systemet deles op, hvilken del der ejer forretningsreglerne, hvordan data nås, og hvilken standard hver ændring skal leve op til. De fortjener at blive truffet tidligt og med vilje. De fleste andre er billige at tage op igen og kan vente, til koden viser, hvad den har brug for.

Nedskrivningen er beslutningens anden halvdel, og det er dér, begrundelsen overlever. En ADR fanger én beslutning med dens begrundelse, dens afvejninger og dens konsekvenser, og samlingen af et projekts ADR'er udgør dets beslutningslog. ADR-sitet giver Michael Nygards indlæg »Documenting Architecture Decisions« fra 2011 æren for at have udbredt idéen. Prisen er en side tekst pr. beslutning. Gevinsten er, at en senere læser kan skelne et bevidst kompromis fra et uheld og tjekke, om begrundelsen stadig holder.

Kvalitetsmål afgør valget mellem strukturer, der alle ville virke. arc42 beder om de tre vigtigste, højst fem, kvalitetsmål for arkitekturen, dem de vigtigste interessenter lægger mest vægt på, og advarer om, at de ikke er det samme som

projektets mål. To designs med de samme funktioner kan opføre sig helt forskelligt, når de fejler, skalerer eller ændres, og kun kvalitetsmålene siger, hvilken forskel der tæller.

Sådan ser det ud i praksis

Platformens repository blev oprettet den 25. november 2025 og var helt tomt, og siden om dens grundlæggende valg slår fast, at den slags valg er »billige at træffe på dag tre og næsten umulige at lave om i måned seks«. To af de tre, der blev truffet dengang, formede strukturen. Stakken blev lagdelt, så hver del har én opgave: en Next.js-frontend, et Go-API, der validerer og videresender, og et PostgreSQL-funktionslag, der ejer forretningsreglerne. Dataadgang blev lagt bag stored functions fra starten, for »at indføre det bagefter ville have betydet at skrive samtlige handlers om«. Ni måneder og flere tusinde commits senere er lagene ifølge siden ikke flydt sammen, og ingen handler taler direkte med en tabel. Dens prøve for en grundlæggende beslutning er ikke, om den lød rigtig, men om den holdt, da den mødte den mængde arbejde, der fulgte.

Struktur, der overlades til tilfældighederne, bliver fastlagt ved et uheld, som siden om GIS-kortet viser. Kortet var vokset frem stykke for stykke: spatiale data i PostgreSQL, tegning i Mapbox, ReactJS i frontenden og NodeJS bagved. Hver del virkede for sig; de var »bare ikke bygget til at passe sammen«. Rollen dækkede full-stack-udviklingen og den tekniske retning, og arkitekturen blev fastlagt først. Derefter blev ansvar ifølge siden »flyttet tilbage til det lag, det hørte hjemme i, hver gang et lag begyndte at lække ind i det næste«, og en god del af arbejdet bestod i »at fange de små uoverensstemmelser, før de størknede til arkitektur«.

Siden om containerlaget behandler den nedskrevne begrundelse som det egentlige resultat. Branchestandarden blandt containermotorer placerer containerporte over værtens firewallregler, så laget er en dæmonløs containermotor, der kører enheder, som genereres af styresystemets egen supervisor, uden nogen publicerede porte overhovedet. Én afvejning er noteret som sådan: enhederne kører på systemniveau og ikke rootless, hvilket holder automatiseringen enkel på bekostning af den strengere isolation, rootless ville give, og noten siger, hvilken vej man skal gå, hvis udbrud fra containere nogensinde kommer til at betyde mere. Det er netop, hvad en ADR rummer: en beslutning med dens begrundelse, dens afvejninger og dens konsekvenser. Siden sætter det at skrive valget ned med sin begrundelse over selve valget, for »den næste person vil få at vide, at man skal bruge standarden, af hver eneste artikel, vedkommende læser«.

Hvor det går galt

En omhyggelig nytilkommen vender en bevidst beslutning om. Begrundelsen fandtes kun i nogens hoved, så valget ligner en fejl, og den standard, hver vejledning anbefaler, ligner rettelsen. Skriv en ADR for hver beslutning, der er dyr at lave om: konteksten, de afvejede muligheder, den valgte, dens konsekvenser og det, der ville gøre den værd at tage op igen. Hold optegnelserne ved siden af koden, så en ændring i den ene bliver gennemgået sammen med den anden.

Forretningsregler dukker op i tre lag på én gang. Hver ændring lagde sin logik der, hvor det nu var bekvemt den eftermiddag, indtil den samme regel blev tjekket i frontenden, API'et og databasen, lidt forskelligt hvert sted. Navngiv, hvad hvert lag ejer, skriv det ned, og flyt et ansvar tilbage, så snart det lækker, mens den vildfarne kopi stadig er lille. Hvor platformen tillader det, så håndhæv også grænsen, for eksempel ved at lade API'et kalde databasefunktioner uden at give det rettigheder til tabellerne.

En regel, der indføres i måned seks, bliver et oprydningsprojekt. Den beslutning, alt afhænger af, blev stående åben, og hver commit siden har bygget på det, der tilfældigvis var sandt. Beslut de få dyre ting først, og lad resten stå åbent med vilje. Skriv det i beslutningsloggen, så et åbent spørgsmål synligt er åbent i stedet for stille at blive besvaret af den næste ændring.

Arkitekturdiagrammet forklarer ikke noget for nogen. Én tegning blander brugere, tjenester, klasser og tabeller, og linjerne mellem kasserne står enten uden etiket eller med bare ordet »Uses«. Tegn ét diagram pr. zoomniveau, som C4 gør, og giv hver linje en etiket, der passer til relationens retning og hensigt; mellem containere skal teknologien eller protokollen også stå. C4's vejledning om notation beder om netop det og fraråder etiketter på ét enkelt ord.

Læs mere

- architecture, NIST CSRC Glossary — definitionen, der bruges ovenfor, ved siden af FIPS-definitionen og definitionen af sikkerhedsarkitektur, som deler opslaget.
- NIST SP 800-160 Vol. 1 Rev. 1 — Engineering Trustworthy Secure Systems, den publikation, definitionen kommer fra.
- Architectural Decision Records — hvad der gør en beslutning arkitektonisk, skabeloner til at skrive en ned og værktøjer, der hjælper.

- The C4 model — fire niveauer af diagrammer, fra systemkontekst ned til kode, med råd om notation og en tjekliste til review.
- arc42 — praktiske tips og eksempler til hvert af de tolv afsnit i arc42-skabelonen.

Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er arkitektur?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/architecture/