Engineer

arkitektur

hvordan et systems deler henger sammen, og hvorfor

Hva det er

Arkitektur, i programvare ofte kalt programvarearkitektur eller systemarkitektur, er de grunnleggende begrepene og egenskapene et system har i omgivelsene sine, slik de kommer til uttrykk i elementene, i relasjonene mellom dem og i prinsippene for hvordan systemet utformes og endrer seg. Ordlyden følger definisjonen i NIST SP 800-160 Volume 1 Revision 1, Engineering Trustworthy Secure Systems, som kom i november 2022.

Hvert ledd i definisjonen har en jobb å gjøre. Elementene er delene: tjenester, databaser, køer, frontenden og modulene inni dem. Relasjonene er hvordan delene avhenger av hverandre, og hva som går mellom dem. Prinsippene er reglene som avgjør hvor nytt arbeid skal ligge, og de holder elementer og relasjoner på plass etter hvert som systemet vokser. «I omgivelsene» gjør brukerne, nabosystemene og begrensningene til en del av arkitekturen. At definisjonen også tar med utvikling over tid, betyr at en beskrivelse som bare stemmer den dagen den ble tegnet, er ufullstendig.

Bare noen av designvalgene er arkitektoniske. Gruppen som forvalter formatet Architectural Decision Record (ADR), kaller en arkitekturbeslutning et begrunnet designvalg som tar hånd om et funksjonelt eller ikke-funksjonelt krav av arkitektonisk betydning. Et krav har den betydningen når det har en målbar effekt på arkitekturen og kvaliteten til et system. Malen arc42 sier det rett ut: avsnittet om beslutninger er for viktige, dyre, risikable eller omstridte beslutninger.

Hver tegning viser bare ett utsnitt, så arkitektur beskrives i visninger. C4-modellen, laget av Simon Brown, gir diagrammer som fungerer som kart du zoomer inn og ut på. Et systemkontekstdiagram viser hvor systemet står blant brukerne sine og systemene rundt det, et containerdiagram viser applikasjonene

og datalagrene inni det, et komponentdiagram tar for seg én container, og der det hjelper, kommer et kodediagram til slutt. arc42 deler den skriftlige beskrivelsen inn i tolv avsnitt, fra mål og begrensninger til beslutninger, risikoer og teknisk gjeld.

Bruk begrepet i en beslutning

Bruk «Hva er arkitektur?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Formen på et system tegnet opp på papiret først, slik at delene passer sammen og kostnaden er kjent før noen skriver kode.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Én infrastrukturplan for 14 avdelinger — 14 forskningsavdelinger på én felles infrastrukturplan

Én infrastrukturplan for 14 avdelinger Ta kontakt →

Hvorfor det er viktig

Det arkitektur egentlig ber et team om å avgjøre, er sjeldnere hvilken struktur det skal velge, og oftere hvilke valg som skal tas nå, med vilje og skriftlig, og hvilke som får stå åpne.

Tar dere et strukturelt valg tidlig, betaler dere før dere vet særlig mye: valget tas med det minste kunnskapsgrunnlaget prosjektet noen gang vil ha, og de første ukene går tregere. Lar dere det stå åpent, blir strukturen avgjort likevel, én beleilig endring om gangen, av den som sist rørte koden. Veien ut er å skille mellom to slags valg. Noen få blir dyre å endre så snart koden avhenger av dem: hvordan systemet deles opp, hvilken del som eier forretningsreglene, hvordan data nås, og hvilken terskel hver endring må over. De fortjener å bli tatt tidlig og med vilje. De fleste andre er billige å ta opp igjen og kan vente til koden viser hva den trenger.

Å skrive beslutningen ned er den andre halvdelen av den, og det er der begrunnelsen overlever. En ADR fanger én beslutning med begrunnelse, avveininger og konsekvenser, og samlingen av ADR-er i et prosjekt utgjør beslutningsloggen. ADR-nettstedet gir Michael Nygards innlegg «Documenting Architecture Decisions» fra 2011 æren for å ha gjort ideen kjent. Prisen er en side tekst per beslutning. Gevinsten er at en senere leser kan skille en bevisst avveining fra et uhell, og sjekke om begrunnelsen fortsatt holder.

Kvalitetsmålene avgjør valget mellom strukturer som alle ville ha fungert. arc42 ber om de tre viktigste kvalitetsmålene for arkitekturen, høyst fem, altså dem de viktigste interessentene bryr seg mest om, og minner om at de er noe annet enn prosjektets mål. To design med de samme funksjonene kan oppføre seg helt ulikt når de feiler, skalerer eller endres, og det er kvalitetsmålene som sier hvilken forskjell som teller.

Slik ser det ut i praksis

Repoet til plattformen ble opprettet 25. november 2025 og var helt tomt. Siden om grunnvalgene slår fast at slike valg er «billige å ta på dag tre og nesten umulige å omgjøre innen måned seks». To av de tre valgene som ble tatt den gangen, formet strukturen. Stakken ble lagdelt slik at hver del har én oppgave: en Next.js-frontend, et Go-API som validerer og videresender, og et funksjonslag i PostgreSQL som eier forretningsreglene. Datatilgangen ble lagt bak lagrede funksjoner fra starten av, for å ettermontere den senere «ville ha betydd å skrive om hver eneste handler». Ni måneder og flere tusen commits senere har lagene ifølge siden holdt seg adskilt, og hver handler går via en funksjon i stedet for rett til en tabell. Prøven siden setter for en grunnleggende beslutning, er om den overlevde møtet med mengden arbeid som kom etterpå, og ikke om den hørtes riktig ut.

Struktur som overlates til tilfeldighetene, blir bestemt ved et uhell, slik siden om GIS-kartet viser. Kartet hadde vokst frem bit for bit: romlige data i PostgreSQL, tegningen i Mapbox, ReactJS i frontenden og NodeJS bak. Hver del fungerte for seg, men delene var «bare ikke bygget for å passe sammen». Rollen omfattet fullstack-utviklingen av kartet og den tekniske retningen, og arkitekturen ble lagt først. Deretter ble ansvar ifølge siden flyttet tilbake «til laget det hørte hjemme i hver gang ett lag begynte å lekke inn i det neste», og en god del av arbeidet var å fange opp «de små inkonsekvensene før de stivnet til arkitektur».

Siden om containerlaget regner den nedskrevne begrunnelsen som selve resultatet. Bransjestandarden blant containermotorer publiserer containerporter forbi reglene i vertens brannmur, så laget er en daemonløs containermotor som driver enheter generert av operativsystemets egen tjenestestyrer, uten publiserte porter i det hele tatt. Én avveining er ført opp som nettopp det: enhetene kjører på systemnivå og ikke rootless. Det holder automatiseringen enkel på bekostning av den strengere isoleringen rootless ville gitt, og notatet sier hvilken vei man skal gå hvis utbrudd fra containeren noen gang blir viktigere. Det er akkurat det en ADR rommer: en beslutning med begrunnelse, avveininger

og konsekvenser. Siden setter det å skrive ned valget med begrunnelse høyere enn selve valget, fordi «den neste som kommer, vil få beskjed om å bruke standarden av hver eneste artikkel vedkommende leser».

Hvor det går galt

En grundig nykommer omgjør en beslutning som var tatt med vilje. Begrunnelsen fantes bare i hodet til én person, så valget ser ut som en feil, og standarden som hver veiledning anbefaler, ser ut som rettelsen. Skriv en ADR for hver beslutning som er dyr å omgjøre: konteksten, alternativene som ble veid, det som ble valgt, konsekvensene og hva som ville gjøre det verdt å ta saken opp igjen. Hold ADR-ene ved siden av koden, slik at en endring i den ene får gjennomgang sammen med den andre.

Forretningsreglene dukker opp i tre lag samtidig. Hver endring la logikken sin der det var lettvint den ettermiddagen, helt til den samme regelen ble sjekket i frontenden, i API-et og i databasen, litt forskjellig hvert sted. Sett navn på hva hvert lag eier, skriv det ned, og flytt et ansvar tilbake så snart det lekker, mens den villfarne kopien fortsatt er liten. Der plattformen tillater det, bør dere også håndheve grensen, f.eks. ved å la API-et kalle databasefunksjoner mens rettighetene på tabellene holdes tilbake.

En regel som innføres i måned seks, blir et opprydningsprosjekt. Beslutningen alt avhenger av, ble stående åpen, og hver commit siden har bygget på det som tilfeldigvis var sant. Bestem de få dyre tingene først, og la resten stå åpent med vilje. Skriv det inn i beslutningsloggen, så et åpent spørsmål er synlig åpent i stedet for å bli besvart i det stille av neste endring.

Arkitekturdiagrammet viser alt på én gang og forklarer lite. Én tegning blander brukere, tjenester, klasser og tabeller, og linjene mellom boksene står uten etikett eller med bare ordet «uses». Tegn ett diagram per zoomnivå, slik C4 gjør, og gi hver linje en etikett som sier hvilken retning relasjonen går og hva den er til. Mellom containere skal teknologien eller protokollen også stå. Veiledningen om notasjon i C4 ber om nettopp dette og fraråder etiketter på ett enkelt ord, som «Uses».

Les mer

- architecture, NIST CSRC Glossary — definisjonen som brukes ovenfor, sammen med FIPS-definisjonen og definisjonen av sikkerhetsarkitektur i samme oppslag.
- NIST SP 800-160 Vol. 1 Rev. 1 — Engineering Trustworthy Secure Systems, publikasjonen definisjonen er hentet fra.
- Architectural Decision Records — hva som gjør en beslutning arkitektonisk, maler for å skrive den ned og verktøy som hjelper.
- The C4 model — fire nivåer av diagrammer, fra systemkontekst ned til kode, med råd om notasjon og en sjekkliste for gjennomgang.
- arc42 — praktiske tips og eksempler for hvert av de tolv avsnittene i arc42-malen.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er arkitektur?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/architecture/