**Engineer**

# Data mesh

at lade domæneteams levere deres analysedata som produkter

## Hvad det er

Et data mesh er en måde at organisere analysedata på i en stor organisation, så de teams, der driver hver del af forretningen, også ejer, udgiver og står inde for de data, som den del frembringer. Zhamak Dehghani, dengang principal technology consultant hos Thoughtworks, beskrev idéen på Martin Fowlers site i en artikel, der udkom i flere dele i maj 2019. I december 2020 samlede hun den i fire principper: domæneorienteret ejerskab af data og deres arkitektur, data som produkt, selvbetjent datainfrastruktur som platform og fødereret governance, som maskinerne håndhæver, på engelsk federated computational governance. Ifølge hende er de fire tilsammen både nødvendige og tilstrækkelige, og hun regner med, at praksis og værktøjer vil skifte, mens principperne står fast.

Idéen gælder analytiske data, ikke driftsdata. Dehghani trækker grænsen sådan: Driftsdata ligger i databaserne bag de systemer, der får forretningen til at køre, de er transaktionelle og viser, hvordan tingene står lige nu. Analytiske data er forretningens historik over tid, aggregeret og modelleret, og det er dem, rapporterne bygges på, og modellerne trænes med. Driftsdata lader hun blive, hvor de er, inde i hver tjeneste. Det er den analytiske side, hun vil have fat i, for det er dér, hun ser friktionen, når virksomheden vokser.

**Domæneejerskab.** Et data mesh skæres efter skel, organisationen allerede har. Dehghanis gennemgående eksempel er en digital medievirksomhed med domæner som podcasts og kunstnere. Det team, der driver podcasts og stiller grænsefladerne til at udgive dem til rådighed, bør også levere historikken over udgivne podcasts, og hvor mange der har lyttet til dem over tid. Skellene er hentet fra domænedrevet design, hvor hvert domænes bounded context afgrænser det område, hvor en ændring holder sig lokal. Derfor er det også et godt sted at placere ejerskabet af data.

**Data som produkt.** Når data spredes ud over mange teams, bliver de sværere at finde og at stole på. Hvert team må derfor behandle de data, det deler, som et produkt og dem, der bruger dem, som kunder. Artiklen fra 2019 opregner, hvad det kræver: Et dataprodukt skal kunne findes, have en fast adresse, være til at stole på, beskrive sig selv, kunne spille sammen med andre efter fælles standarder og være beskyttet af fælles adgangskontrol. En produktejer i domænet står til ansvar for mål som kvaliteten, hvor længe forbrugerne venter på data, og hvor tilfredse de er. Ansvaret for kvaliteten rykker opstrøms, så tæt på kilden som muligt. Arkitektonisk er et dataprodukt den mindste enhed, der kan udrulles for sig, og det rummer tre dele: koden, der omformer og udstiller data og håndhæver reglerne for dem, selve data med tilhørende metadata og den infrastruktur, der kører det hele.

**En selvbetjent platform.** At bygge, drive og holde øje med et dataprodukt kræver specialiseret viden om infrastruktur, og den kan ikke kopieres ind i hvert eneste domæne. Et platformteam stiller derfor værktøjer til rådighed på et højere niveau, blandt andet til lagring, produktskemaer, deklarerede pipelines og sporing af datas oprindelse, så en udvikler med en bred profil kan bygge et dataprodukt uden den specialistviden.

**Fødereret governance.** Produkterne skal stadig passe sammen, ellers kan ingen kombinere dem. En gruppe af produktejere fra domænerne og fra platformen afgør, hvilke regler der gælder overalt, og hvilke der overlades til det enkelte domæne, og platformen håndhæver de fælles regler automatisk. Dehghanis eksempel er lyttertal: Hvordan podcastdomænet modellerer dem, er dets egen sag, men hvordan en lytter identificeres, er et fælles anliggende, fordi den samme person også optræder som bruger i andre domæner.


### Brug begrebet i en beslutning

Brug »Hvad er et data mesh?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Tal fra salg, drift og økonomi samlet ét sted, så to afdelinger ikke længere kommer til det samme møde med hver sit svar.

**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. Har gjort en geodatapipeline 50 gange hurtigere — 50x hurtigere geografiske datapipelines

**Har gjort en geodatapipeline 50 gange hurtigere** Kontakt os →

# Hvorfor det betyder noget

Dehghanis argument begynder med det, der går galt, når en stor virksomhed driver en central data lake eller et centralt data warehouse. Data fra alle domæner trækkes ind på én platform, som ejer det hele og drives af et team af specialister i dataværktøjer, der sidder langt fra både dem, der skaber data, og dem, der bruger dem. Hun peger på tre måder, modellen svigter på. Platformen er central og monolitisk, så den får det svært, efterhånden som kilder og forbrugere bliver flere. Den er delt op i pipelinetrin som indlæsning, rensning, aggregering og udstilling, så et enkelt nyt tal, for eksempel hvor tit podcasts bliver afspillet, kræver en ændring i hvert trin og en udgivelse, der skal koordineres på tværs af teams. Og ejerskabet er delt op i siloer og højt specialiseret: De ingeniører, der skal få mening i data, ved kun lidt om de domæner, data kommer fra, mens de teams, der frembringer data, ikke har nogen grund til at gøre det ordentligt.

Et data mesh flytter, hvem der gør rent. I den gamle model ankommer data med deres fejl, og et centralt team renser dem efter indlæsningen. I et data mesh bliver et domænes datasæt først et dataprodukt, når domænet selv har tjekket det op mod de aftalte kvalitetsmål og de fælles regler. De mennesker, der ved, hvad et felt burde betyde, er også dem, der står inde for det.

Governance skifter også form. Artiklen fra 2020 stiller de to modeller over for hinanden i en tabel. Et centralt team har selv ansvaret for kvalitet, sikkerhed og overholdelse af regler. Et fødereret team fastlægger, hvordan kvalitet skal modelleres, og hvilke regler for sikkerhed og lovgivning platformen skal bygge ind og overvåge. Før blev succes målt i antallet af tabeller under governance. I et data mesh måles den i, hvor meget produkterne faktisk bliver brugt. Manuel kontrol, der prøver at forhindre fejl, afløses af automatiske processer, der opdager fejlene og retter op efter dem.

Intet af det præsenteres som et universalsvar. I artiklen fra 2019 medgiver Dehghani, at den centrale model kan fungere i organisationer med et enklere domæne og færre slags forbrugere, og hun skriver, at den kommer til kort i virksomheder med rige domæner, mange kilder og forbrugere med vidt forskellige behov. Om et data mesh er besværet værd, afhænger af, hvilken af de to slags organisationen hører til.

# Sådan ser det ud i praksis

Et rigtigt data mesh spænder over mange teams og mange datalagre. Skitsen nedenfor holder sig til én PostgreSQL-database, så hver SQL-sætning kan efterprøves i manualen, og den viser de to mindste træk: et domæne, der udgiver et produkt, og en platform, der håndhæver én fælles regel.

Ordreteamet ejer driftstabellerne i skemaet `orders`. Sine analysedata udgiver det i et skema for sig, som et versioneret view med en beskrivelse og en udtrykkelig tildeling af rettigheder:

```
-- ordreteamet ejer dette skema og alt, hvad der ligger i det
CREATE SCHEMA order_data;

CREATE VIEW order_data.orders_by_day_v1 AS
SELECT order_date, currency, count(*) AS orders, sum(total) AS revenue
FROM orders.orders
WHERE status = 'paid'
GROUP BY order_date, currency;

COMMENT ON VIEW order_data.orders_by_day_v1 IS
  'Betalte ordrer pr. dag og valuta. Ejer: ordreteamet.';

GRANT USAGE ON SCHEMA order_data TO analysts;
GRANT SELECT ON order_data.orders_by_day_v1 TO analysts;
```

PostgreSQL-manualen forklarer, hvorfor det er nok på forbrugerens side: Den, der bruger et view, skal kun have USAGE-rettigheden til det skema, viewet ligger i, ikke til de skemaer, som viewets forespørgsel læser fra. Analytikerne kan altså hente data fra produktet, mens driftstabellerne bag det forbliver lukket for dem. Evans nævner for sin del databaseskemaer blandt de fysiske former, som grænsen om en bounded context kan antage.

Platformens side er en regel, der gælder for alle domæner. Den er skrevet som en forespørgsel og køres ved hver udgivelse: Intet produktskema må indeholde et view uden beskrivelse. `obj_description` returnerer den kommentar, der er gemt på et objekt, og i `pg_class` betyder en `relkind` med værdien `v`, at objektet er et view:

```
-- fælles regel: hvert udgivet view har en beskrivelse
SELECT n.nspname, c.relname
FROM pg_class AS c
```

```
JOIN pg_namespace AS n ON n.oid = c.relnamespace
WHERE c.relkind = 'v'
  AND n.nspname ~ '_data$'
  AND obj_description(c.oid, 'pg_class') IS NULL;
-- hver række, der kommer tilbage, standser udgivelsen
```

Den opdeling er den fødererede del i lille målestok. Hvad der tæller som en betalt ordre, bestemmer ordreteamet. At hvert produkt beskriver sig selv, og at navnene på produktskemaer ender på `_data`, er fælles konventioner, som platformen tjekker, uden at nogen behøver at huske dem.

Ydelsessiden Data governance & datakvalitet lover, at det skal stå klart, hvem der ejer hvert tal, og hvilken kilde det har, og netop det skal en produktejer for data kunne gøre rede for. Blandt det udførte arbejde står et funktionsbaseret datalag i PostgreSQL, bygget sådan, at al læsning og skrivning sker gennem funktioner, databasen kan give adgang til, og aldrig direkte gennem tabellerne. Det er samme tankegang som i viewet ovenfor: Forbrugerne får en grænseflade med en rettighed på, ikke tabellerne nedenunder.

Porteføljecasen Én infrastrukturplan til 14 afdelinger kalder sig ikke et data mesh, men den rammer den spænding, som governancegruppen i et data mesh lever med. På et forskningsinstitut arbejdede 14 forskningsgrupper med hver deres datakilder, formater, skalaer og værktøjer, og deres tekniske kunnen og it-ressourcer var meget forskellige: Nogle få havde bygget deres egne løsninger, mens mange kæmpede. Planen skulle lade forskerne koncentrere sig om forskningen, og den afvejede fleksibilitet mod standardisering med en modulær arkitektur, veje ind for meget forskellige datakilder, grænseflader til forskellige kompetenceniveauer, skalerbar lagring og behandling samt strategier for onboarding, support og governance. At afgøre, hvad hver gruppe beholder for sig selv, og hvad alle deler, er netop den afvejning, Dehghani lægger hos den fødererede governance, og at sænke den kunnen, en gruppe skal have, er det, hun forlanger af en selvbetjent platform.

## Hvor det går galt

**Et centralt team med et nyt navn.** Det ændrer intet at kalde dataplatformteamet et data mesh, hvis det samme team stadig tager imod rådata fra alle domæner, renser dem og står inde for kvaliteten. Dehghanis governancetabel kan bruges som tjekliste: Hvem har ansvaret for kvaliteten,

hvem fastlægger sikkerhedsreglerne, og måles succes stadig i tabeller under governance? Flyt ejerskabet af hvert dataprodukt til det domæne, der frembringer data, og behold det centrale team til platformen og reglerne.

**Produkter, ingen står inde for.** Et domæne, der eksporterer tabeller uden en ejer, en beskrivelse eller et løfte om, hvor friske data er, har blot flyttet det gamle problem ind i sit eget hjørne. Dehghani beder hvert dataprodukt om at angive sin kvalitet som service level objectives, for eksempel aktualitet og fejlrater, og bemærker, at ét domæne kan tilbyde to produkter med de samme hændelser: et næsten i realtid, der kan miste eller gentage hændelser, og et, der kommer senere og er mere præcist. Udpeg en ejer for hvert produkt, offentliggør produktets mål, og test kvaliteten dér, hvor data opstår.

**Ingen fælles identifikatorer.** Hvis hvert domæne identificerer de samme ting på sin egen måde, kan ingen to produkter kobles sammen. I eksemplet fra 2019 ser en kunstner forskellig ud i det domæne, der registrerer afspilninger, og i det domæne, der betaler kunstnerne. Aftal fælles, hvordan den slags delte entiteter, som Dehghani kalder polysemer, identificeres, og overlad resten af modellen til hvert domæne. Feltformater, konventioner for adresser og fælles metadatafelter hører hjemme på den samme fælles liste.

**Hvert domæne bygger hele stakken om.** Uden en selvbetjent platform må hvert domæne selv sætte lagring, pipelines og overvågning op, og de specialiserede kompetencer, der var en mangelvare i ét centralt team, skal nu findes i hvert eneste team. Byg platformen, før domænerne bliver bedt om at udgive, og mål den på, hvor lidt specialistviden et domæne behøver for at levere et produkt.

**Et produkt, der ændres under forbrugerne.** Et dataprodukt er en grænseflade, og ændrer man den på stedet, går de rapporter i stykker, der bygger på den. I PostgreSQL afviser `CREATE OR REPLACE VIEW` en del af den fejl: Den nye forespørgsel skal give de samme kolonner med samme navne, rækkefølge og datatyper og må kun tilføje kolonner til sidst. En ændret betydning fanger den ikke, for beregningen bag en kolonne må gerne være en helt anden. Læg enhver ændring, der omdøber, fjerner eller omdefinerer en kolonne, i et nyt view, `orders_by_day_v2`, ved siden af det gamle, og udfas det gamle, når forbrugerne er flyttet.

**Et data mesh, hvor der ikke er brug for et.** Fire principper, et platformteam, en fødereret governancegruppe og en produktejer i hvert domæne er meget maskineri. Mod slutningen af sin artikel fra 2019 indrømmer Dehghani, at

»enterprise scale adoption still has a long way to go«. Hvis en håndfuld kilder og ét team dækker behovet, så behold et velfungerende centralt lager med en navngiven ejer for hver tabel, og lån de dele, der betaler sig uanset størrelse: ejere, beskrevne datasæt og regler, som maskinen tjekker.

## Læs mere

- Data Mesh Principles and Logical Architecture — Zhamak Dehghani, 2020: de fire principper, et dataprodukts dele og tabellen, der stiller central og fødereret governance over for hinanden.
- How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh — den oprindelige artikel fra maj 2019 med en central dataplatforms tre svagheder og de egenskaber, et dataprodukt skal have.
- Domain-Driven Design Reference — Eric Evans' sammenfatning af mønstrene, herunder den bounded context, hvis grænser et data mesh deler ejerskabet efter.
- PostgreSQL: CREATE VIEW — reglerne for at erstatte et view og de rettigheder, brugerne af det skal have, som i eksemplet.
- PostgreSQL: COMMENT — hvordan en beskrivelse gemmes på et databaseobjekt, så `obj_description` kan læse den igen.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring et repræsentativt datasæt, de spørgsmål det skal besvare, og den aktuelle flaskehals. Vi kan aftale målinger af kvalitet, gennemløb og gendannelse.

**Se, hvad det koster** Kontakt os →

# Gør begreberne praktiske

Spørg, hvordan »Hvad er et data mesh?« 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/data-mesh/