Engineer

Data mesh

å la domeneteam levere sine analysedata som produkter

Hva det er

Et data mesh er en måte å organisere analysedata på i en stor organisasjon, slik at teamene som driver hver del av virksomheten, også eier, publiserer og står ansvarlige for dataene den delen produserer. Zhamak Dehghani, den gang principal technology consultant i Thoughtworks, la fram ideen på Martin Fowlers nettsted i en artikkel som kom ut i flere deler i mai 2019. I desember 2020 formulerte hun den på nytt som fire prinsipper: domeneorientert eierskap til data og arkitekturen rundt dem, data som produkt, selvbetjent datainfrastruktur som plattform og føderert styring som maskinene håndhever, på engelsk federated computational governance. De fire skulle etter hennes syn være både nødvendige og tilstrekkelige til sammen, og hun regnet med at praksis og verktøy kommer til å endre seg mens prinsippene står fast.

Ideen gjelder analytiske data, ikke operative. Dehghani trekker skillet slik: Operative data ligger i databasene bak systemene som holder virksomheten i gang, de er transaksjonelle og viser hvordan ting står akkurat nå. Analytiske data er virksomhetens historikk over tid, aggregert og modellert, og det er dem rapportene bygger på og modellene trenes med. De operative dataene lar hun ligge der de er, inne i hver tjeneste. Det er den analytiske siden hun retter modellen mot, for det er der hun ser friksjonen når virksomheten vokser.

Domeneeierskap. Et data mesh deles langs skillelinjer organisasjonen allerede har. Dehghanis gjennomgående eksempel er et digitalt medieselskap med domener som podkaster og artister. Teamet som driver podkastene og tilbyr grensesnittene for å publisere dem, bør også levere historikken over publiserte podkaster og hvor mange som har lyttet til dem over tid. Skillelinjene er hentet fra domenedrevet design, der hvert domenes bounded context avgrenser området der en endring holder seg lokal. Derfor er det også et godt sted å legge eierskapet til dataene.

Data som produkt. Når data spres over mange team, blir de vanskeligere å finne og å stole på. Hvert team må derfor behandle dataene det deler som et produkt, og dem som bruker dataene som kunder. Artikkelen fra 2019 lister opp hva som skal til: Et dataprodukt skal være mulig å finne, ha en fast adresse, være til å stole på, beskrive seg selv, kunne spille sammen med andre etter felles standarder og være beskyttet av felles tilgangskontroll. En produkteier i domenet står ansvarlig for mål som kvalitet, hvor lenge konsumentene må vente på data og hvor fornøyde de er. Ansvaret for kvaliteten flyttes oppstrøms, så nær kilden som mulig. Arkitektonisk er et dataprodukt den minste enheten som kan rulles ut for seg selv, og det består av tre deler: koden som transformerer og leverer dataene og håndhever reglene for dem, selve dataene med tilhørende metadata og infrastrukturen som kjører det hele.

En selvbetjent plattform. Å bygge, drifte og følge med på et dataprodukt krever spesialisert infrastrukturkompetanse, og den lar seg ikke kopiere inn i hvert eneste domene. Et plattformteam tilbyr derfor verktøy på et høyere nivå, blant annet for lagring, produktskjemaer, deklarerte pipelines og sporing av dataenes opphav, slik at en utvikler med bred profil kan bygge et dataprodukt uten den spesialistkunnskapen.

Føderert styring. Produktene må fortsatt passe sammen, ellers kan ingen kombinere dem. En gruppe produkteiere fra domenene og fra plattformen avgjør hvilke regler som skal gjelde overalt, og hvilke som overlates til det enkelte domenet, og plattformen håndhever de felles reglene automatisk. Dehghanis eksempel er lyttertall: Hvordan podkastdomenet modellerer dem, er domenets egen sak, men hvordan en lytter identifiseres, er en felles sak, fordi den samme personen også dukker opp som bruker i andre domener.

Bruk begrepet i en beslutning

Bruk «Hva er et data mesh?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Tall fra salg, drift og økonomi ført sammen, slik at to avdelinger slutter å komme med to ulike svar til det samme møtet.

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. Gjorde en geodatapipeline 50 ganger raskere — 50x raskere pipelines for geografiske data

Gjorde en geodatapipeline 50 ganger raskere Ta kontakt →

Hvorfor det er viktig

Dehghanis argument starter med det som går galt når et stort selskap driver en sentral datasjø eller et sentralt datavarehus. Data fra alle domener trekkes inn på én plattform som eier alt sammen, og den driftes av et team av spesialister på dataverktøy som sitter langt unna både dem som skaper dataene, og dem som bruker dem. Hun peker på tre måter dette svikter på. Plattformen er sentral og monolittisk, så den får det tungt etter hvert som kilder og konsumenter blir flere. Den er delt opp i pipeline-trinn som innlasting, rensing, aggregering og levering, så et enkelt nytt tall, for eksempel hvor ofte podkaster blir spilt av, krever en endring i hvert trinn og en utgivelse som må koordineres på tvers av team. Og eierskapet er splittet i siloer og høyt spesialisert: Ingeniørene som skal få mening ut av dataene, kjenner lite til domenene de kommer fra, mens teamene som produserer dataene, ikke har noe insentiv til å gjøre dem riktige.

Et data mesh endrer hvem som rydder. I den gamle modellen kommer dataene inn med feilene sine, og et sentralt team vasker dem etter innlastingen. I et data mesh blir et domenes datasett først et dataprodukt når domenet selv har kontrollert det mot de avtalte kvalitetsmålene og de felles reglene. De som vet hva et felt skal bety, er også de som går god for det.

Styringen skifter også form. Artikkelen fra 2020 stiller de to modellene opp mot hverandre i en tabell. Et sentralt team har selv ansvaret for kvalitet, sikkerhet og etterlevelse av regelverk. Et føderert team bestemmer hvordan kvalitet skal modelleres, og hvilke sikkerhetskrav og lovkrav plattformen skal bygge inn og overvåke. Før ble suksess målt i antall tabeller under styring. I et data mesh måles den i hvor mye produktene faktisk blir brukt. Manuell kontroll som prøver å hindre feil, erstattes av automatiserte prosesser som oppdager feil og retter opp etter dem.

Ingenting av dette legges fram som et universalsvar. I artikkelen fra 2019 innrømmer Dehghani at den sentrale modellen kan fungere for organisasjoner med et enklere domene og færre typer konsumenter, og skriver at den svikter i virksomheter med rike domener, mange kilder og konsumenter med svært ulike behov. Om et data mesh er bryet verdt, avhenger av hvilken av de to typene organisasjonen hører til.

Slik ser det ut i praksis

Et ekte data mesh strekker seg over mange team og mange datalagre. Skissen nedenfor holder seg til én PostgreSQL-database, slik at hver SQL-setning kan sjekkes mot manualen, og den viser de to minste grepene: et domene som publiserer et produkt, og en plattform som håndhever én felles regel.

Ordreteamet eier driftstabellene i skjemaet `orders`. Analysedataene sine publiserer det i et eget skjema, som et versjonert view med en beskrivelse og en uttrykkelig tildeling av rettigheter:

```
-- ordreteamet eier dette skjemaet og alt som 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 per dag og valuta. Eier: ordreteamet.';

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

PostgreSQL-manualen forklarer hvorfor dette er nok på konsumentsiden: Den som bruker et view, trenger bare USAGE-privilegiet på skjemaet viewet ligger i, ikke på skjemaene som viewets spørring leser fra. Analytikerne kan altså hente data fra produktet, mens driftstabellene bak det forblir stengt for dem. Evans regner på sin side databaseskjemaer blant de fysiske formene grensen rundt en bounded context kan ta.

Plattformens side er en regel som gjelder for alle domener. Den er skrevet som en spørring og kjøres ved hver utgivelse: Ingen produktskjemaer får inneholde et view uten beskrivelse. `obj_description` returnerer kommentaren som er lagret på et objekt, og i `pg_class` betyr en `relkind` med verdien `v` at objektet er et view:

```
-- felles regel: hvert publisert 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 rad som kommer tilbake, stopper utgivelsen
```

Den oppdelingen er den fødererte delen i miniatyr. Hva som teller som en betalt ordre, bestemmer ordreteamet. At hvert produkt beskriver seg selv, og at navnene på produktskjemaer slutter på `_data`, er felles konvensjoner som plattformen kontrollerer uten at noen trenger å huske dem.

Tjenestesiden Datastyring og datakvalitet lover at hvert tall skal ha en kjent eier og en kjent kilde, og det er nettopp det en produkteier for data må kunne gjøre rede for. Blant det leverte arbeidet står et funksjonsbasert datalag i PostgreSQL, bygd slik at all lesing og skriving går gjennom funksjoner databasen kan gi tilgang til, og aldri rett mot tabellene. Det er samme tankegang som i viewet over: Konsumentene får et grensesnitt med en tilgang knyttet til seg, ikke tabellene under.

Oppdraget Én infrastrukturplan for 14 avdelinger beskrives ikke som et data mesh, men det møter den samme spenningen som styringsgruppen i et data mesh lever med. Ved et forskningsinstitutt arbeidet 14 forskningsgrupper med hver sine datakilder, formater, skalaer og verktøy, og både den tekniske kompetansen og IT-ressursene varierte mye: Noen få hadde bygd egne løsninger, mens mange slet. Planen skulle la forskerne konsentrere seg om forskningen, og den veide fleksibilitet mot standardisering, med modulær arkitektur, integrasjonsveier for svært ulike datakilder, grensesnitt for ulike ferdighetsnivåer, skalerbar lagring og prosessering og strategier for onboarding, støtte og styring. Å avgjøre hva hver gruppe beholder for seg selv, og hva alle deler, er nettopp den avveiningen Dehghani legger til den fødererte styringen, og å senke kompetansekravet for en gruppe er det hun krever av en selvbetjent plattform.

Hvor det går galt

Et sentralt team med nytt navn. Det endrer ingenting å kalle dataplattformteamet et data mesh hvis det samme teamet fortsatt tar imot rådata fra alle domener, vasker dem og står ansvarlig for kvaliteten. Dehghanis styringstabell fungerer som sjekkliste: Hvem har ansvaret for kvaliteten, hvem

bestemmer sikkerhetsreglene, og måles suksess fortsatt i tabeller under styring? Flytt eierskapet til hvert dataprodukt til domenet som produserer dataene, og behold det sentrale teamet til plattformen og reglene.

Produkter ingen står inne for. Et domene som eksporterer tabeller uten eier, beskrivelse eller løfte om hvor ferske dataene er, har bare flyttet det gamle problemet inn i sitt eget hjørne. Dehghani ber hvert dataprodukt oppgi kvaliteten sin som tjenestenivåmål, for eksempel aktualitet og feilrater, og påpeker at ett domene kan tilby to produkter med de samme hendelsene: ett nesten i sanntid som kan miste eller gjenta hendelser, og ett som kommer senere og er mer nøyaktig. Utpek en eier for hvert produkt, publiser målene, og test kvaliteten der dataene oppstår.

Ingen felles identifikatorer. Hvis hvert domene identifiserer de samme tingene på sin egen måte, kan ingen to produkter kobles sammen. I eksempelet fra 2019 ser en artist forskjellig ut i domenet som registrerer avspillinger, og i domenet som betaler artistene. Bli enige på tvers om hvordan slike delte entiteter, som Dehghani kaller polysemer, identifiseres, og overlat resten av modellen til hvert domene. Feltformater, adressekonvensjoner og felles metadatafelt hører hjemme på den samme felles listen.

Hvert domene bygger hele stakken på nytt. Uten en selvbetjent plattform må hvert domene selv sette opp lagring, pipelines og overvåking, og den spesialiserte kompetansen som var mangelvare i ett sentralt team, må nå finnes i hvert eneste team. Bygg plattformen før domenene blir bedt om å publisere, og mål den på hvor lite spesialistkunnskap et domene trenger for å levere et produkt.

Et produkt som endres under konsumentene. Et dataprodukt er et grensesnitt, og endrer man det på stedet, ryker rapportene som bygger på det. I PostgreSQL avviser `CREATE OR REPLACE VIEW` en del av denne feilen: Den nye spørringen må gi de samme kolonnene med samme navn, rekkefølge og datatyper og kan bare legge til kolonner på slutten. En endret betydning fanger den ikke, for beregningen bak en kolonne kan godt være en helt annen. Legg enhver endring som gir nytt navn til, fjerner eller omdefinerer en kolonne, i et nytt view, `orders_by_day_v2`, ved siden av det gamle, og fas ut det gamle når konsumentene har flyttet over.

Et data mesh der det ikke trengs. Fire prinsipper, et plattformteam, en føderert styringsgruppe og en produkteier i hvert domene er mye maskineri. Mot slutten av artikkelen fra 2019 innrømmer Dehghani at «enterprise scale adoption still has a long way to go». Dekker en håndfull kilder og ett team behovet, så behold et velfungerende sentralt lager med en navngitt eier for hver tabell, og lån de delene som lønner seg uansett størrelse: eiere, beskrevne datasett og regler som maskinen kontrollerer.

Les mer

- Data Mesh Principles and Logical Architecture — Zhamak Dehghani, 2020: de fire prinsippene, delene i et dataprodukt og tabellen som stiller sentral og føderert styring opp mot hverandre.
- How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh — den opprinnelige artikkelen fra mai 2019, med de tre svakhetene ved en sentral dataplattform og egenskapene et dataprodukt må ha.
- Domain-Driven Design Reference — Eric Evans' sammendrag av mønstrene, blant dem bounded context, langs hvis grenser et data mesh deler eierskapet.
- PostgreSQL: CREATE VIEW — reglene for å erstatte et view og privilegiene brukerne av det trenger, slik eksempelet bruker dem.
- PostgreSQL: COMMENT — hvordan en beskrivelse lagres på et databaseobjekt, slik at `obj_description` kan lese den tilbake.

Ordliste

Velg ut fra deres faktiske rammer

Ta med et representativt datasett, spørsmålene det skal besvare og dagens flaskehals. Vi kan avtale målinger av kvalitet, gjennomstrømning og gjenoppretting.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er et data mesh?» 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/data-mesh/