Engineer

Data mesh

données d'analyse des équipes métier fournies comme produits

Ce que c’est

Un data mesh organise les données analytiques d’une grande organisation pour que les équipes responsables de chaque partie de l’activité possèdent aussi les données qu’elle produit, les publient et en répondent. Zhamak Dehghani, alors consultante principale en technologie chez Thoughtworks, a présenté l’idée sur le site de Martin Fowler dans une série de publications en mai 2019. Elle l’a reformulée en décembre 2020 en quatre principes : la propriété des données et de leur architecture par domaine, les données comme produit, l’infrastructure de données en libre-service comme plateforme et la gouvernance fédérée automatisée. Les quatre devaient, selon elle, être ensemble nécessaires et suffisants. Elle prévoyait que pratiques et outils changeraient, tandis que les principes resteraient inchangés.

Le mesh concerne les données analytiques, pas les données opérationnelles. Dehghani trace la limite ainsi. Les données opérationnelles sont dans les bases des systèmes qui font fonctionner l’activité ; elles sont transactionnelles et portent l’état actuel. Les données analytiques sont l’histoire de l’activité au fil du temps, agrégée et modélisée, sur laquelle on construit les rapports et entraîne les modèles. Elle laisse les données opérationnelles dans chaque service et vise le côté analytique, où elle situe les frictions à grande échelle.

La propriété par domaine. Le mesh suit les divisions que l’organisation possède déjà. L’exemple récurrent de Dehghani est une entreprise de médias numériques, avec des domaines comme les podcasts et les artistes. L’équipe qui gère les podcasts et propose les interfaces de publication devrait aussi fournir l’historique des podcasts publiés et le nombre de personnes qui les ont écoutés au fil du temps. La frontière vient de la conception pilotée par le domaine, où le contexte délimité de chaque domaine est le lieu dans lequel les changements restent locaux ; c’est donc aussi un bon endroit pour placer la responsabilité des données.

Les données comme produit. Lorsque les données sont réparties entre de nombreuses équipes, il devient plus difficile de les trouver et de leur faire confiance. Chacune doit donc traiter les données partagées comme un produit et leurs utilisateurs comme des clients. L'article de 2019 en donne les exigences : un produit de données devrait être découvrable, adressable, digne de confiance, autodescriptif, interopérable selon des normes communes à tous les domaines et protégé par un contrôle d'accès global. Un responsable de produit de données du domaine répond de mesures comme la qualité, le temps d'attente des consommateurs pour obtenir les données et leur satisfaction. La responsabilité de la qualité remonte aussi près que possible de la source. Dans l'architecture, un produit de données est la plus petite unité déployable seule. Il comprend trois parties : le code qui transforme et sert les données et applique leurs politiques, les données avec leurs métadonnées et l'infrastructure d'exécution.

Une plateforme en libre-service. Construire, exploiter et surveiller un produit de données demande des compétences spécialisées en infrastructure qu'on ne peut reproduire dans chaque domaine. Une équipe plateforme fournit donc des outils de plus haut niveau : stockage, schémas de produits, pipelines explicitement définis et filiation, par exemple. Un développeur généraliste du domaine peut ainsi construire un produit de données sans ce savoir spécialisé.

Une gouvernance fédérée automatisée. Les produits doivent encore s'assembler, sans quoi personne ne peut les combiner. Un groupe de responsables de produits des domaines et de responsables de plateforme décide des règles globales et de celles laissées à chaque domaine. La plateforme applique automatiquement les premières. Dehghani prend les chiffres d'écoute comme exemple : leur modélisation appartient au domaine podcasts, mais l'identification d'un auditeur est globale, puisque la même personne apparaît comme utilisateur dans d'autres domaines.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce qu'un data mesh ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Réunir les chiffres des ventes, des opérations et des finances pour que deux services ne présentent plus deux réponses à la même réunion.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. Pipeline géographique accéléré de 50 fois — 50x d'accélération des pipelines de données géographiques

Pipeline géographique accéléré de 50 fois Contactez-nous →

Pourquoi c’est important

Le raisonnement de Dehghani commence par les échecs d’un lac ou entrepôt central dans une grande entreprise. Les données de tous les domaines arrivent dans une plateforme qui les possède toutes, gérée par des spécialistes des outils de données séparés des producteurs et des consommateurs. Elle nomme trois modes d’échec. La plateforme centralisée et monolithique peine lorsque sources et consommateurs se multiplient. Elle est découpée en étapes d’ingestion, de nettoyage, d’agrégation et de mise à disposition ; un seul nouveau chiffre, comme la fréquence d’écoute des podcasts, exige donc une modification à chaque étape et une livraison coordonnée entre équipes. Enfin, sa responsabilité est cloisonnée et très spécialisée : les ingénieurs devant comprendre les données connaissent mal les domaines d’origine, tandis que les équipes productrices n’ont aucune incitation à en assurer l’exactitude.

Le mesh change la répartition du travail de nettoyage. Dans l’ancien modèle, les données arrivent avec leurs erreurs et une équipe centrale les nettoie après ingestion. Dans un mesh, le jeu de données d’un domaine ne devient un produit qu’une fois vérifié par ce domaine selon les mesures de qualité convenues et les règles globales. Les personnes qui savent ce qu’un champ devrait signifier sont celles qui en répondent.

La gouvernance change aussi de forme. L’article de 2020 compare les modèles dans un tableau. Une équipe centrale répond elle-même de la qualité, de la sécurité et de la conformité ; une équipe fédérée définit la modélisation de la qualité, ainsi que les règles de sécurité et les exigences réglementaires que la plateforme devrait intégrer et surveiller. Le succès se mesurait au nombre de tables gouvernées ; dans un mesh, il se mesure à la consommation réelle des produits. La revue manuelle visant à prévenir les erreurs cède la place aux processus automatiques qui les détectent et rétablissent un fonctionnement normal.

Rien de tout cela n’est proposé comme réponse universelle. Dans l’article de 2019, Dehghani reconnaît que le modèle central peut fonctionner pour une organisation au domaine plus simple avec moins de types de consommateurs. Elle dit qu’il échoue dans les entreprises aux domaines riches, aux nombreuses sources et aux consommateurs divers. L’intérêt d’un mesh au regard des moyens supplémentaires qu’il exige dépend du cas auquel appartient l’organisation.

À quoi cela ressemble en pratique

Un vrai mesh couvre de nombreuses équipes et de nombreux stockages. L'esquisse suivante se limite à une base PostgreSQL, pour que chaque instruction soit vérifiable dans le manuel. Elle montre les deux opérations élémentaires : un domaine publie un produit et la plateforme impose une règle globale.

L'équipe commandes possède les tables opérationnelles du schéma `orders`. Elle publie ses données analytiques dans son propre schéma sous forme de vue versionnée, avec une description et une autorisation explicite :

```
-- the orders team owns this schema and everything in it
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
  'Paid orders per day and currency. Owner: orders team.';

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

Le manuel PostgreSQL explique pourquoi cela suffit côté consommateur : l'utilisateur d'une vue n'a besoin du privilège USAGE que sur le schéma qui contient la vue, pas sur les schémas qu'elle lit. Les analystes peuvent interroger le produit tandis que les tables opérationnelles leur restent inaccessibles. Evans cite pour sa part les schémas de bases de données parmi les formes physiques que peut prendre la frontière d'un contexte délimité.

Côté plateforme, une règle écrite sous forme de requête est exécutée à chaque livraison et vaut pour tous les domaines : aucun schéma de produit ne doit contenir une vue sans description. `obj_description` renvoie le commentaire d'un objet, et un `relkind` de `v` marque une vue dans `pg_class` :

```
-- global rule: every published view carries a description
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;
-- any row returned stops the release
```

Cette séparation est la partie fédérée en miniature. Ce qui compte comme commande payée relève de l'équipe commandes. Le fait que tout produit se décrive, et que les schémas de produits finissent par `_data`, relève de conventions globales que la plateforme vérifie sans que quelqu'un ait à s'en souvenir.

La page de prestation Gouvernance et qualité des données promet qu'on saura clairement qui possède chaque chiffre et d'où il vient, précisément ce qu'un responsable de produit de données doit pouvoir dire. Parmi les réalisations livrées, elle cite une couche PostgreSQL organisée autour de fonctions, pour que lectures et écritures passent par des fonctions auxquelles la base peut accorder l'accès, jamais par les tables elles-mêmes. C'est l'intention derrière la vue ci-dessus : les consommateurs reçoivent une interface avec une autorisation, pas les tables sous-jacentes.

L'étude de cas Un plan d'infrastructure pour 14 départements ne se présente pas comme un data mesh, mais rencontre la tension à laquelle est confronté le groupe de gouvernance d'un mesh. Dans un institut de recherche, 14 groupes utilisaient des sources, formats, échelles et outils différents. Leurs compétences techniques et ressources informatiques différaient beaucoup : quelques-uns avaient créé leurs solutions, beaucoup peinaient. Le plan visait à laisser les chercheurs se concentrer sur la recherche et équilibrait flexibilité et standardisation : architecture modulaire, prise en charge de sources très diverses, interfaces adaptées à différents niveaux de compétence, stockage et traitement évolutifs, stratégies d'intégration, d'assistance et de gouvernance. Décider de ce que chaque groupe garde et de ce que tous partagent est l'équilibre confié par Dehghani à la gouvernance fédérée. Réduire le niveau de compétences spécialisées requis d'un groupe est ce qu'elle demande à la plateforme en libre-service.

Où les problèmes surviennent

Une équipe centrale change de nom. Renommer l'équipe plateforme en mesh ne change rien si elle reçoit toujours les données brutes de tous les domaines, les nettoie et répond de leur qualité. Le tableau de gouvernance de Dehghani fournit une liste de contrôle utilisable : qui répond de la qualité, qui décide des

règles de sécurité et si le succès se compte encore en tables gouvernées. Transférez la propriété de chaque produit au domaine qui produit ses données, et gardez l'équipe centrale pour la plateforme et les règles.

Personne ne répond des produits. Un domaine exportant des tables sans propriétaire, description ni engagement de fraîcheur n'a fait que déplacer l'ancien problème dans son coin. Dehghani demande à chaque produit d'énoncer sa qualité sous forme d'objectifs de niveau de service, par exemple ponctualité et taux d'erreur. Elle note qu'un domaine peut proposer deux produits des mêmes événements : un presque en temps réel, qui peut omettre ou répéter des événements, et un plus tardif et plus exact. Nommez un propriétaire par produit, publiez ses objectifs et testez la qualité là où les données sont produites.

Pas d'identifiants partagés. Si chaque domaine identifie les mêmes choses à sa façon, aucun produit ne peut être joint à un autre. Dans l'exemple de 2019, un artiste n'est pas représenté de la même façon dans le domaine qui enregistre les écoutes et dans celui qui le rémunère. Accordez-vous globalement sur l'identification de ces entités partagées, appelées polysèmes par Dehghani, et laissez le reste du modèle à son domaine. Les formats de champs, conventions d'adresses et champs de métadonnées communs relèvent de la même liste globale.

Chaque domaine reconstruit sa pile. Sans plateforme en libre-service, chacun doit provisionner son stockage, ses pipelines et sa supervision ; les compétences rares dans une équipe centrale sont désormais nécessaires partout. Construisez la plateforme avant de demander aux domaines de publier. Évaluez-la d'après le niveau minimal de connaissances spécialisées nécessaire à un domaine pour livrer un produit.

Le produit change sous les pieds de ses consommateurs. Un produit de données est une interface ; le modifier sur place casse les rapports qui en dépendent. PostgreSQL refuse une partie de cette erreur avec `CREATE OR REPLACE VIEW` : la nouvelle requête doit produire les mêmes colonnes, avec les mêmes noms, ordre et types, et ne peut en ajouter qu'à la fin. Il ne détecte pas un changement de sens : le calcul derrière une colonne peut être entièrement différent. Toute modification renommant, supprimant ou redéfinissant une colonne doit aller dans une nouvelle vue, `orders_by_day_v2` , à côté de l'ancienne. Retirez celle-ci après la migration de ses consommateurs.

Un mesh inutile. Quatre principes, une équipe plateforme, un groupe de gouvernance fédérée et un responsable de produit par domaine représentent un dispositif conséquent. Vers la fin, l'article de 2019 reconnaît que « l'adoption à l'échelle de l'entreprise a encore beaucoup de chemin à faire ». Si quelques sources et une équipe suffisent, gardez un stockage central bien tenu avec un propriétaire nommé pour chaque table. Reprenez les parties utiles à toute échelle : propriétaires, jeux de données décrits et règles vérifiées par la machine.

Pour aller plus loin

- Principes du data mesh et architecture logique — Zhamak Dehghani, 2020 : les quatre principes, les parties d'un produit et le tableau comparant gouvernance centrale et fédérée.
- Passer d'un lac monolithique à un data mesh distribué — l'article initial de mai 2019, les trois modes d'échec d'une plateforme centrale et les qualités requises d'un produit de données.
- Référence de conception pilotée par le domaine — le résumé des modèles d'Eric Evans, dont le contexte délimité, sur les frontières duquel un mesh répartit la propriété.
- PostgreSQL : CREATE VIEW — les règles de remplacement d'une vue et les privilèges nécessaires à ses utilisateurs, employés dans l'exemple.
- PostgreSQL : COMMENT — le stockage d'une description sur un objet, ensuite relue par `obj_description` .

Glossaire

Choisissez selon vos contraintes réelles

Présentez un jeu de données représentatif, les questions auxquelles il doit répondre et le goulot d'étranglement actuel. Nous pouvons convenir de la manière de mesurer la qualité, le débit et la reprise.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce qu'un data mesh ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

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

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/data-mesh/