Engineer

Data mesh

надання аналітичних даних як продукту силами доменних команд

Що це таке

Data mesh — це спосіб організувати аналітичні дані у великій організації так, щоб команди, які ведуть кожну частину бізнесу, самі володіли даними цієї частини, публікували їх і відповідали за них. Ідею запропонувала Жамак Дегані (Zhamak Dehghani), тоді провідна технічна консультантка у Thoughtworks, у статті на сайті Мартіна Фаулера (Martin Fowler), яка виходила частинами в травні 2019 року. У грудні 2020 року вона сформулювала її заново як чотири принципи: доменне володіння даними та архітектурою навколо них, дані як продукт, інфраструктура даних самообслуговування як платформа і федеративне управління, правила якого перевіряються автоматично (англійською federated computational governance). На її думку, ці чотири принципи разом і необхідні, і достатні, а практики та інструменти змінюватимуться, тоді як самі принципи залишаться.

Ідея стосується аналітичних даних, а не операційних. Дегані розмежовує їх так: операційні дані живуть у базах даних за системами, на яких тримається щоденна робота бізнесу; вони транзакційні й показують, як усе стоїть саме зараз. Аналітичні дані — це історія бізнесу в часі, агрегована й змодельована; на них будують звіти й навчають моделі. Операційні дані вона залишає там, де вони є, усередині кожного сервісу. Змінити вона хоче саме аналітичний бік, бо саме там, на її думку, виникає тертя, коли бізнес росте.

Доменне володіння. Data mesh ділиться вздовж меж, які в організації вже є. Наскрізний приклад у Дегані — цифрова медіакомпанія з доменами на кшталт подкастів і виконавців. Команда, яка веде подкасти й надає інтерфейси для їх публікації, має також постачати історію опублікованих подкастів і того, скільки людей їх слухали з часом. Межі взято з предметно-орієнтованого проєктування (domain-driven design), де обмежений

контекст (bounded context) кожного домену окреслює ту ділянку, у межах якої зміна залишається локальною. Тому саме там доречно закріпити й володіння даними.

Дані як продукт. Коли дані розходяться між багатьма командами, їх стає важче знайти і важче їм довіряти. Тому кожна команда має ставитися до даних, якими ділиться, як до продукту, а до тих, хто ними користується, — як до клієнтів. Стаття 2019 року перелічує, що для цього потрібно: продукт даних має бути таким, щоб його можна було знайти, мати сталу адресу, заслуговувати на довіру, описувати сам себе, сполучатися з іншими за спільними стандартами й підлягати спільному контролю доступу. Власник продукту в домені відповідає за такі показники, як якість, час, який споживачі чекають на дані, і їхнє задоволення. Відповідальність за якість переходить вище за течією, якомога ближче до джерела. З погляду архітектури продукт даних — найменша одиниця, яку можна розгортати окремо, і складається він із трьох частин: коду, який перетворює й віддає дані та застосовує до них правила, самих даних разом із метаданими та інфраструктури, на якій усе це працює.

Платформа самообслуговування. Щоб збудувати продукт даних, експлуатувати його й стежити за ним, потрібні вузькоспеціалізовані знання інфраструктури, і відтворити їх у кожному домені не вийде. Тому команда платформи пропонує інструменти вищого рівня — зокрема для зберігання, схем продуктів, декларативних пайплайнів і відстеження того, звідки походять дані, — щоб розробник широкого профілю міг збудувати продукт даних без цих спеціальних знань.

Федеративне управління. Продукти все одно мають пасувати один до одного, інакше ніхто не зможе їх поєднати. Група власників продуктів із доменів і з платформи вирішує, які правила діють усюди, а які лишаються на розсуд окремого домену, і платформа автоматично стежить за виконанням спільних правил. Приклад Дегані — дані про прослуховування: як домен подкастів їх моделює, — його власна справа, але як ідентифікувати слухача, — спільне питання, бо та сама людина з’являється як користувач і в інших доменах.

Застосуйте поняття до рішення

Застосуйте «Що таке data mesh?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Дані продажів, операцій і фінансів зведені разом, щоб два відділи не приносили на одну нараду дві різні відповіді.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Прискорила пайплайн географічних даних у 50 разів — 50x швидші пайплайни географічних даних

Прискорила пайплайн географічних даних у 50 разів Зв'язатися з нами →

Чому це важливо

Аргумент Дегані починається з того, що йде не так, коли велика компанія тримає центральне озеро даних або центральне сховище даних. Дані з усіх доменів стікаються на одну платформу, яка володіє ними всіма, а обслуговує її команда фахівців з інструментів для даних, далека і від тих, хто дані створює, і від тих, хто ними користується. Дегані називає три способи, якими це ламається. Платформа централізована й монолітна, тож що більше стає джерел і споживачів, то важче їй дається. Вона поділена на етапи пайплайна — завантаження, очищення, агрегування, видачу, — тож навіть один новий показник, скажімо, як часто вмикають подкасти, вимагає змін на кожному етапі й випуску, який доводиться узгоджувати між командами. І володіння розбите на ізольовані, вузькоспеціалізовані групи: інженери, які мають видобути з даних сенс, погано знають домени, звідки ці дані надходять, а команди, які дані створюють, не мають жодного стимулу робити їх правильними.

Data mesh змінює те, хто прибирає. У старій моделі дані надходять разом зі своїми помилками, і центральна команда очищає їх уже після завантаження. У data mesh набір даних домену стає продуктом даних лише тоді, коли сам домен перевірив його на відповідність узгодженим цілям якості та спільним правилам. Ті, хто знає, що має означати поле, за нього й ручаються.

Управління теж змінює форму. Стаття 2020 року зіставляє дві моделі в таблиці. Центральна команда сама відповідає за якість, безпеку й дотримання регуляторних вимог. Федеративна команда визначає, як моделювати якість і які вимоги безпеки та законодавства платформа має вбудувати й відстежувати. Раніше успіх вимірювали кількістю таблиць під управлінням. У data mesh його вимірюють тим, наскільки активно продуктами справді користуються. Ручні перевірки, які намагаються запобігти помилкам, поступаються автоматизованим процесам, що виявляють помилки й виправляють їхні наслідки.

Ніщо з цього не подається як універсальна відповідь. У статті 2019 року Дегані визнає, що централізована модель може працювати в організаціях із простішим доменом і меншою кількістю типів споживачів, і пише, що вона не витримує в бізнесі з багатими доменами, великою кількістю джерел і споживачами з дуже різними потребами. Чи вартий data mesh зусиль, залежить від того, до якого з двох типів належить організація.

Як це виглядає на практиці

Справжній data mesh охоплює багато команд і багато сховищ. Нарис нижче обмежується однією базою даних PostgreSQL, щоб кожну SQL-інструкцію можна було звірити з документацією, і показує два найменші кроки: домен публікує продукт, а платформа стежить за одним спільним правилом.

Команда замовлень володіє операційними таблицями у схемі `orders`. Аналітичні дані вона публікує в окремій схемі — як версіоноване представлення (view) з описом і явно наданими правами:

```
-- команда замовлень володіє цією схемою і всім, що в ній лежить
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
  'Оплачені замовлення за днями й валютами. Власник: команда замовлень.';

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

Документація PostgreSQL пояснює, чому з боку споживача цього досить: тому, хто користується представленням, потрібен привілей USAGE лише на схему, у якій це представлення лежить, а не на схеми, з яких читає його запит. Отже, аналітикам відкрито дані продукту, а операційні таблиці за ним для них залишаються закритими. Ерік Еванс (Eric Evans), зі свого боку, зараховує схеми баз даних до фізичних форм, яких може набувати межа обмеженого контексту.

Частина платформи — це правило, яке діє для всіх доменів. Воно записане як SQL-запит і виконується під час кожного випуску: жодна схема продукту не може містити представлення без опису. `obj_description` повертає коментар, збережений для об’єкта, а в `pg_class` значення `v` у стовпці `relkind` означає, що об’єкт є представленням:

```
-- спільне правило: кожне опубліковане представлення має опис
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;
-- кожен повернений рядок зупиняє випуск
```

Такий поділ — це федеративна частина в мініатюрі. Що вважати оплаченим замовленням, вирішує команда замовлень. Те, що кожен продукт описує сам себе, а назви схем продуктів закінчуються на `_data`, — спільні домовленості, які платформа перевіряє сама, і нікому не треба про них пам’ятати.

Сторінка послуги Управління даними та їх якість обіцяє, що в кожної цифри буде відомий власник і відоме джерело, а саме за це має вміти відповісти власник продукту даних. Серед виконаної роботи там названо функційний рівень даних у PostgreSQL, у якому всі операції читання й запису проходять через функції, доступ до яких база даних видає окремо, а не безпосередньо через таблиці. Це та сама думка, що й у представленні вище: споживач отримує інтерфейс зі своїм доступом, а не таблиці під ним.

Проєкт Один інфраструктурний план для 14 підрозділів не описано як data mesh, але він стикається з тією самою напругою, з якою живе група управління в data mesh. У науково-дослідному інституті 14 дослідницьких груп працювали кожна зі своїми джерелами даних, форматами, масштабами й інструментами, а технічна підготовка та IT-ресурси в них дуже різнилися: кілька груп збудували власні рішення, тоді як багато інших не давали собі ради. План мав дати науковцям змогу зосередитися на дослідженнях і шукав рівновагу між гнучкістю та стандартизацією: модульна архітектура, шляхи інтеграції для дуже різних джерел даних, зручні інтерфейси для різного рівня технічних навичок, масштабовані рішення для зберігання й обробки даних, а також стратегії онбордингу, підтримки та управління. Вирішувати, що кожна

група залишає собі, а що всі мають спільним, — це саме той вибір, який Дегані покладає на федеративне управління, а знижувати поріг знань для групи — саме те, чого вона вимагає від платформи самообслуговування.

Де виникають проблеми

Центральна команда під новою назвою. Нічого не зміниться, якщо назвати команду платформи даних data mesh, а вона й далі прийматиме сирі дані з усіх доменів, очищатиме їх і відповідатиме за якість. Таблиця управління зі статті Дегані працює як чекліст: хто відповідає за якість, хто визначає правила безпеки і чи досі успіх вимірюють кількістю таблиць під управлінням? Передайте володіння кожним продуктом даних доменові, який ці дані створює, а за центральною командою залиште платформу й правила.

Продукти, за які ніхто не відповідає. Домен, який вивантажує таблиці без власника, опису чи обіцянки щодо свіжості даних, просто переніс стару проблему у свій куток. Дегані просить, щоб кожен продукт даних описував свою якість мовою цілей рівня обслуговування, наприклад свіжістю й часткою помилок, і зауважує, що домен може пропонувати два продукти з тими самими подіями: один майже в реальному часі, який може губити або дублювати події, і другий, що надходить пізніше, зате точніший. Призначте кожному продукту власника, опублікуйте цілі й перевіряйте якість там, де дані виникають.

Немає спільних ідентифікаторів. Якщо кожен домен ідентифікує ті самі речі по-своєму, жодні два продукти не вдасться поєднати. У прикладі 2019 року виконавець виглядає по-різному в домені, що фіксує відтворення, і в домені, що виплачує виконавцям гроші. Домовтеся між доменами, як ідентифікувати такі спільні сутності — Дегані називає їх полісемами, — а решту моделі залиште кожному доменові. Формати полів, правила адресації та спільні поля метаданих належать до того самого спільного переліку.

Кожен домен заново будує весь стек. Без платформи самообслуговування кожен домен мусить сам налаштовувати зберігання, пайплайни й моніторинг, а вузькоспеціалізовані знання, яких бракувало навіть одній центральній команді, тепер потрібні в кожній команді. Збудуйте платформу, перш ніж просити домени публікувати продукти, і оцінюйте її тим, як мало спеціальних знань потрібно доменові, щоб випустити продукт.

Продукт, що змінюється під споживачами. Продукт даних — це інтерфейс, і якщо змінити його на місці, звіти, які на ньому тримаються, зламаються. У PostgreSQL `CREATE OR REPLACE VIEW` відхиляє частину таких змін: новий запит має повертати ті самі стовпці з тими самими назвами, у тому самому порядку й тих самих типів, і може лише додавати нові стовпці в кінці. Змінене значення він не помітить, бо обчислення за стовпцем може бути вже зовсім іншим. Виносьте кожну зміну, яка перейменовує, прибирає або переозначує стовпець, у нове представлення `orders_by_day_v2` поруч зі старим, а старе виводьте з ужитку, коли споживачі перейдуть.

Data mesh там, де він не потрібен. Чотири принципи, команда платформи, федеративна група управління й власник продукту в кожному домені — це чимало механізмів. Наприкінці статті 2019 року Дегані визнає, що «до впровадження в масштабах підприємств ще далеко». Якщо вистачає кількох джерел і однієї команди, залиште добре налагоджене центральне сховище з названим власником кожної таблиці й позичте те, що окупається за будь-якого масштабу: власників, описані набори даних і правила, які перевіряє машина.

Що почитати далі

- Data Mesh Principles and Logical Architecture — Жамак Дегані, 2020: чотири принципи, складники продукту даних і таблиця, що зіставляє централізоване й федеративне управління.
- How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh — первісна стаття травня 2019 року: три вади центральної платформи даних і властивості, які має мати продукт даних.
- Domain-Driven Design Reference — стислий виклад шаблонів від Еріка Еванса, серед них обмежений контекст, уздовж меж якого data mesh розподіляє володіння.
- PostgreSQL: CREATE VIEW — правила заміни представлення і привілеї, потрібні його користувачам, саме в тому вигляді, у якому їх використано в прикладі.
- PostgreSQL: COMMENT — як опис зберігається на об’єкті бази даних, щоб `obj_description` міг прочитати його назад.

Словник

Обирайте за фактичними умовами

Надайте репрезентативний набір даних, запитання до нього та поточне вузьке місце. Узгодимо вимірювання якості, пропускної здатності та відновлення.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке data mesh?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

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

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/data-mesh/