Engineer

データメッシュ

ドメインの各チームが自分たちの分析用データを製品として提供する仕組み

どのようなものか

データメッシュは、大きな組織の分析用データを整理する方法です。事業の各部分を運営するチームが、その部分から生まれるデータも自ら管理し、公開し、その内容について説明する責任を負います。当時Thoughtworksのプリンシパル・テクノロジー・コンサルタントだったZhamak Dehghaniが、2019年5月にMartin Fowlerのサイトで連載として提案し、2020年12月に四つの原則として整理し直しました。ドメイン単位でデータとそのアーキテクチャに責任を持つこと、データを製品として扱うこと、セルフサービスのデータ基盤をプラットフォームとして提供すること、計算処理による連合型ガバナンスです。四つを合わせて必要十分とし、実践や道具は変わっても、原則は変わらないと考えていました。

対象は分析用データであり、業務を動かすデータではありません。Dehghaniは、次のように区別します。業務用データは、事業を動かすシステムの背後のデータベースにあり、トランザクションを扱い、現在の状態を保持します。分析用データは、事業の時間的な履歴を集計し、モデル化したもので、レポートの作成やモデルの学習に使います。業務用データは各サービス内の元の場所に残し、規模が大きくなると摩擦が生じると考えた分析側を、メッシュの対象にしています。

ドメインごとの責任。 メッシュは、組織にすでにある区分に沿って分かれます。Dehghaniが繰り返し使う例は、ポッドキャストやアーティストといったドメインを持つデジタルメディア企業です。ポッドキャストを運営し、公開用のインターフェースを提供するチームは、公開済みポッドキャストの履歴と、時間とともに何人が聴いたかも提供すべきだとします。この境界は、ドメイン駆動設計に由来します。各ドメインの境界づけられたコンテキストは、変更の影響をその中にとどめる範囲なので、データの責任もそこに置くのが適しています。

製品としてのデータ。 データを多数のチームに分散すると、見つけにくく、信頼しにくくなります。そのため、各チームは共有するデータを製品として扱い、その利用者を顧客として扱う必要があります。2019年の記事は、その条件を列挙しています。データ製品は、発見可能で、アドレスで参照でき、信頼でき、自己記述的で、全体共通の標準に従って相互運用でき、全体共通のアクセス制御で保護されているべきだとします。ドメインのデータ製品責任者は、品質、利用者がデータを待つ時間、満足度といった指標に責任を負います。品質の責任を負う場所を、可能な限りデータの発生源へ近づけます。アーキテクチャ上、データ製品は、単独でデプロイできる最小単位です。データを変換して提供し、ポリシーを適用するコード、メタデータを伴うデータ、それを実行するインフラという三つの部分を持ちます。

セルフサービスのプラットフォーム。 データ製品の構築、実行、監視には、インフラの専門知識が必要です。それをすべてのドメインにそろえることはできません。そこでプラットフォームチームが、ストレージ、製品のスキーマ、宣言したパイプライン、データの来歴などを扱う、より抽象度の高い道具を提供します。ドメインで幅広い仕事を担う開発者が、その専門知識なしにデータ製品を作れるようにするためです。

計算処理による連合型ガバナンス。 製品同士は、互いに整合している必要があります。そうでなければ、誰も組み合わせて使えません。ドメインの製品責任者とプラットフォームの責任者からなるグループが、全体に適用する規則と各ドメインに任せる規則を決め、全体共通の規則はプラットフォームが自動で適用します。Dehghaniの例は、聴取数です。ポッドキャストのドメインが、その数字をどうモデル化するかは、そのドメインの判断です。しかし、聴取者をどう識別するかは、全体で決める必要があります。同じ人が、ほかのドメインでは利用者として現れるためです。

概念を判断に生かす

「データメッシュとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 営業、運用、経理の数値をまとめ、同じ会議に部門ごとに異なる答えを持ち寄る状況を解消します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 地理情報パイプラインを50倍に高速化 — 50x 地理データパイプラインの高速化

地理情報パイプラインを50倍に高速化 お問い合わせ →

なぜ重要なのか

Dehghaniの議論は、大企業が中央集約型のデータレイクやデータウェアハウスを運用すると、何がうまくいかないかから始まります。あらゆるドメインのデータが、それらすべてに責任を持つ一つのプラットフォームへ集められます。運用するのはデータ用ツールの専門家チームで、データを作る人からも、使う人からも離れています。彼女は、三つの失敗の形を挙げます。中央集約された一体型のプラットフォームなので、発生源と利用者が増えると対応が難しくなります。取り込み、クレンジング、集計、提供といったパイプラインの段階に分かれているため、ポッドキャストの再生頻度という一つの新しい数字にも、すべての段階の変更と、チーム間で調整したリリースが必要です。さらに、責任が分断され、専門化されすぎています。データを理解しなければならないエンジニアは、その出所のドメインをよく知らず、データを生むチームには、正しく作る動機がありません。

メッシュは、誰がデータを整えるかを変えます。従来のモデルでは、誤りを含んだデータが届き、取り込み後に中央チームがクレンジングします。メッシュでは、ドメイン自身が、合意した品質指標と全体共通の規則に照らして検査して初めて、そのドメインのデータセットがデータ製品になります。フィールドが何を意味すべきかを知る人が、その内容を保証します。

ガバナンスの形も変わります。2020年の記事は、二つのモデルを表で対比しています。中央チームは、自ら品質、セキュリティー、コンプライアンスに責任を持ちます。連合型のチームは、品質をどうモデル化し、どのセキュリティー規則や法規制上の規則をプラットフォームに組み込んで監視するかを定義します。以前は、管理対象のテーブル数で成功を数えていました。メッシュでは、製品が実際にどれだけ使われるかで数えます。誤りを防ごうとする手作業のレビューに代わり、自動化した処理が誤りを検出し、そこからの復旧も行います。

万能の答えとして提案しているわけではありません。2019年の記事でDehghaniは、ドメインが比較的単純で、利用者の種類が少ない組織なら、中央集約型でも機能すると認めています。一方、ドメインが複雑で、発生源が多く、利用者が多様な企業では行き詰まると述べます。メッシュの追加負担に見合う価値があるかは、その組織がどちらに当たるかで決まります。

実際の活用例

本物のメッシュは、多数のチームとデータストアにまたがります。以下の例は、一つのPostgreSQLデータベースに限定しています。各文をマニュアルと照合できるようにし、ドメインが製品を公開することと、プラットフォームが全体共通

の規則を一つ適用すること、という最小の二つの動きを示します。

注文チームは、`orders` スキーマの業務用テーブルに責任を持ちます。分析用データは、自分たちのスキーマ内で、説明と明示的な権限付与を伴う、バージョン付きのビューとして公開します。

```
-- 注文チームがこのスキーマとその中のすべてを管理する
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
  '日別・通貨別の支払い済み注文。管理者：注文チーム。';

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

PostgreSQLのマニュアルは、利用者側にはこれで十分な理由を説明しています。ビューを使う人がUSAGE権限を必要とするのは、ビューを置くスキーマだけで、ビューが読み取る先のスキーマには不要です。分析担当者は製品を問い合わせられますが、その背後の業務用テーブルは、引き続き直接利用できません。Evansも、境界づけられたコンテキストの境界が取り得る物理的な形の一つとして、データベースのスキーマを挙げています。

プラットフォーム側は、すべてのドメインに当てはまる規則を、クエリーとして書き、リリースのたびに実行します。製品のスキーマには、説明のないビューを置いてはいけません。`obj_description` は、オブジェクトに保存されたコメントを返し、`pg_class` 内の `relkind` が `v` なら、ビューを表します。

```
-- 全体共通の規則：公開するすべてのビューに説明を付ける
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;
-- 一行でも返ればリリースを止める
```

この役割分担が、連合型という部分の縮図です。何を支払い済みの注文と見なすかは、注文チームが決めます。どの製品にも自己説明を付けること、製品スキーマの末尾を `_data` にすることは、全体共通の約束です。誰かが覚えておかなくても、プラットフォームが検査します。

データガバナンスと品質のサービスは、各数字に誰が責任を持ち、どこから来たのかを明確にするとしています。これは、データ製品責任者が説明できなければならないこと、そのものです。納品した仕事として、関数を中心にしたPostgreSQLのデータ層を挙げています。読み書きは、テーブルを直接経由せず、データベースがアクセス権を付与できる関数を必ず通るように設計しました。上のビューの背景にある考え方と同じです。利用者に渡すのは、背後のテーブルではなく、権限を付与したインターフェースです。

ポートフォリオの事例14部門に共通する一つのインフラ計画は、データメッシュを名乗ってはいませんが、メッシュのガバナンスグループが向き合う緊張関係に直面しています。ある研究機関では、14の研究グループが、それぞれ異なるデータ源、形式、規模、道具を使っていました。技術力とITリソースにも大きな差があり、一部は独自の仕組みを作っていましたが、多くは苦労していました。計画は、研究者が研究に集中できることを目指し、柔軟性と標準化の釣り合いを取りました。モジュール構成のアーキテクチャ、大きく異なるデータ源からの取り込み経路、習熟度に合ったインターフェース、拡張可能なストレージと処理、導入支援、サポート、ガバナンスの戦略を組み合わせています。各グループに何を任せ、何を全体で共有するかを決めることが、Dehghaniが連合型ガバナンスに委ねる調整です。グループに求める専門知識の水準を下げることは、セルフサービスのプラットフォームに求めている役割です。

問題が起きるところ

中央チームの名前だけを変えます。 データプラットフォームのチームをメッシュと呼んでも、同じチームが、全ドメインの生データを受け取り、整え、品質に責任を負うままなら、何も変わりません。Dehghaniのガバナンスの比較表は、実用的な確認項目になります。品質の責任者は誰か、セキュリティー規則を誰が決めるか、成功を今も管理対象のテーブル数で数えていないか、という点です。各データ製品の責任を、データを生むドメインへ移し、中央チームはプラットフォームと規則を担当するようにします。

誰も責任を持たない製品になります。 責任者、説明、データの鮮度に関する約束を付けずにテーブルを出力するドメインは、以前の問題を自分の場所へ移しただけです。Dehghaniは、各データ製品が、適時性や誤り率といったサービスレベル目標で品質を示すことを求めています。また、同じイベントについて、ドメ

インが二つの製品を提供してもよいとします。一つはほぼリアルタイムですが、イベントを取りこぼしたり重複させたりする可能性があり、もう一つは届くのが遅い代わりに、より正確です。すべての製品に責任者を置き、目標を公開し、データを生む場所で品質を検査します。

共通の識別子がありません。 各ドメインが同じものを独自に識別すると、どの二つの製品も結合できません。2019年の例では、再生を記録するドメインと、アーティストに支払うドメインで、アーティストの表し方が異なります。Dehghaniが多義語と呼ぶ、こうした共通の実体をどう識別するかは、全体で合意します。各モデルの残りは、そのドメインに任せます。フィールドの形式、アドレスの規約、共通メタデータのフィールドも、同じ全体共通の一覧に含めます。

すべてのドメインが、基盤を作り直します。 セルフサービスのプラットフォームがなければ、各ドメインが、自前のストレージ、パイプライン、監視を用意する必要があります。一つの中央チームでも不足していた専門知識が、すべてのチームに必要になります。ドメインに公開を求める前にプラットフォームを作り、ドメインがどれほど少ない専門知識で製品を届けられるかで評価します。

利用中の製品を、その場で変更します。 データ製品はインターフェースなので、同じ場所で変更すると、それに基づくレポートが壊れます。PostgreSQLの `CREATE OR REPLACE VIEW` は、この誤りの一部を拒否します。新しいクエリーは、同じ名前、順序、データ型の同じ列を出力する必要があり、列を追加できるのは末尾だけです。意味の変更は検出できません。列の計算内容は、まったく別のものに変わり得るためです。列名を変える、列を消す、列の定義を変える変更は、`orders_by_day_v2` という新しいビューに分け、古いビューと並べます。利用者が移行してから、古い方を廃止します。

必要のない場所に、メッシュを導入します。 四つの原則、プラットフォームチーム、連合型ガバナンスのグループ、各ドメインの製品責任者をそろえるには、大がかりな体制が必要です。Dehghaniの2019年の記事も、終盤で「企業規模での採用には、まだ長い道のりがある」と認めています。少数のデータ源と一つのチームで足りるなら、各テーブルに明確な責任者を置き、中央のデータストアを適切に運用します。どんな規模でも役立つ部分、つまり責任者、説明のあるデータセット、機械が検査する規則を取り入れます。

参考資料

- データメッシュの原則と論理アーキテクチャ — Zhamak Dehghaniによる2020年の記事です。四つの原則、データ製品の構成要素、中央集約型と連合型のガバナンスを比べる表を説明します。
- 一体型のデータレイクから、分散したデータメッシュへ進む方法 — 2019年5月の最初の記事です。中央のデータプラットフォームが失敗する三つの形と、データ製品に必要な性質を説明します。
- ドメイン駆動設計リファレンス — Eric Evansによるパターンの要約です。メッシュで責任を分ける際の境界になる、境界づけられたコンテキストも含みます。
- PostgreSQL：CREATE VIEW — ビューを置き換える際の規則と、利用者に必要な権限です。例で使った部分を説明します。
- PostgreSQL：COMMENT — データベースのオブジェクトに説明を保存する方法です。`obj_description` は、その説明を読み出します。

用語集

実際の制約に合わせて選ぶ

代表的なデータセット、そこから答えを得たい問い、現在のボトルネックをお知らせください。品質、処理量、復旧の測り方を合意できます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「データメッシュとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

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

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/data-mesh/