dm

dataminded

BETTER DATA ENGINEERING

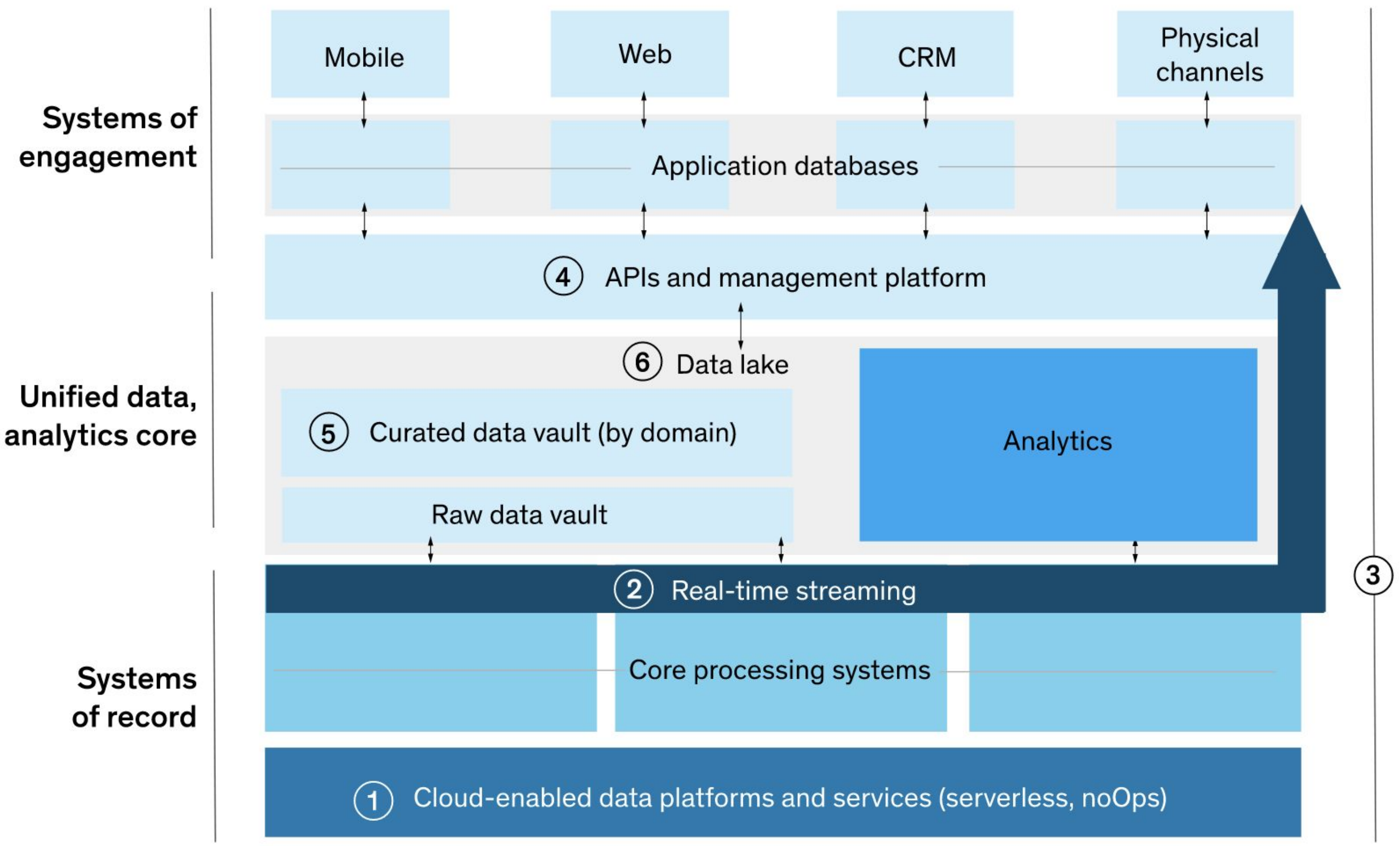
From

Fluffy Data Straty

To

Getting Data Done

Upgrade data architecture by making six foundational shifts.



Systems of engagement

Mobile

Web

CRM

Physical channels

Application databases

④ APIs and management platform

Unified data, analytics core

⑥ Data lake

⑤ Curated data vault (by domain)

Raw data vault

Analytics

② Real-time streaming

③

Systems of record

Core processing systems

① Cloud-enabled data platforms and services (serverless, noOps)

① From on-premise to cloud-based data platforms

② From batch to real-time data processing

③ From pre-integrated commercial solutions to modular, best-of-breed platforms

④ From point-to-point to decoupled data access

⑤ From an enterprise warehouse to domain-based architecture

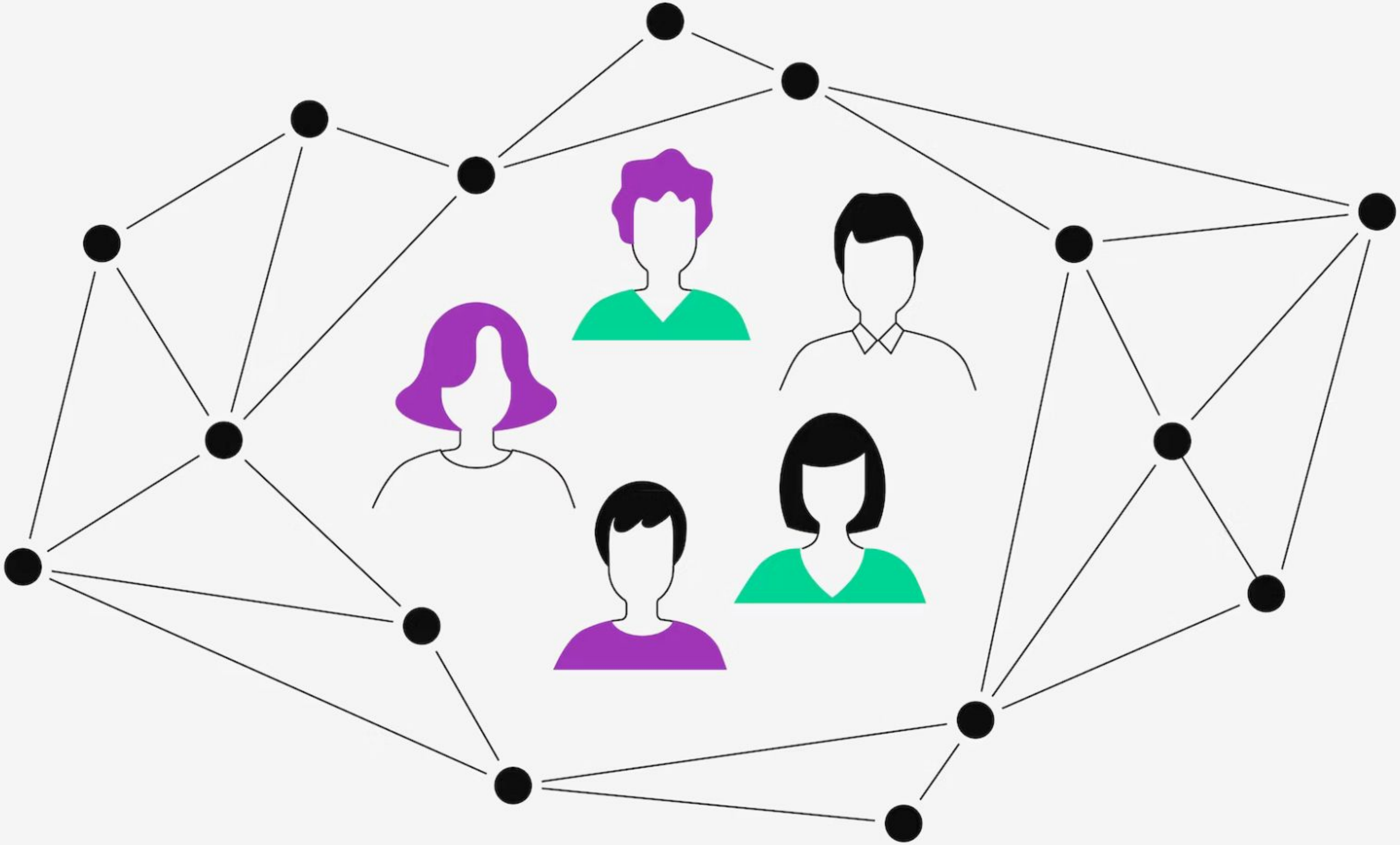
⑥ From rigid data models to flexible, extensible data schemas

Kubernetes clusters, AWS deployment strategy, Spark scaling, Terraform state files, Airflow, CI/CD, …

Quarterly business results, dashboards, insights, ML algorithms, experiments, ROC curve, …

Data platform team

Business analyst / data scientist



dataminded

BETTER DATA ENGINEERING

Practical approaches about how to make your data platform a success

About us

Jelle De Vleminck

Senior data engineer focused on making other engineers more productive

Kristof Martens

Senior data engineer focused on building secure, at scale data platforms in the cloud for organizations across different industries.

Business and IT are often acting like rivaling superpowers during the cold war

Business (decentralised)

IT department (centralised)

A difference in values

Business

IT department

Freedom

Speed
Agility
Experiments
User Experience
Ease of use
Self Service

CONTROL

Cost
Process & Procedures
Reuse
Standardization
Operations
Security, legal, compliance
Oversight

They carry dangerous weapons

Business

IT department

Mutually Assured Destruction

Business

IT department

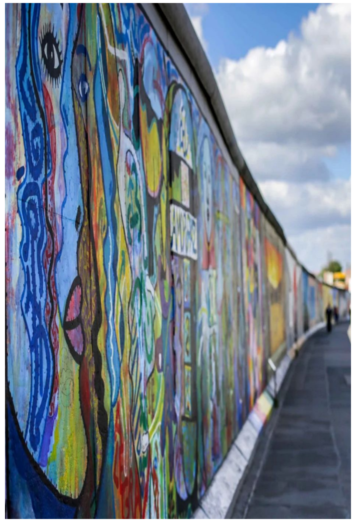
They build walls

Business

IT department

CONTROL

Yearly budget exercise
Maintain what we built
More functionality
Priority calls
Escalations



Freedom

Tickets
Queues
Inflated estimates (time / budget)
Compartmentalisation
No (budget/manpower)
Committees

As a consequence

Business

IT department

Slow and expensive

Cowboys

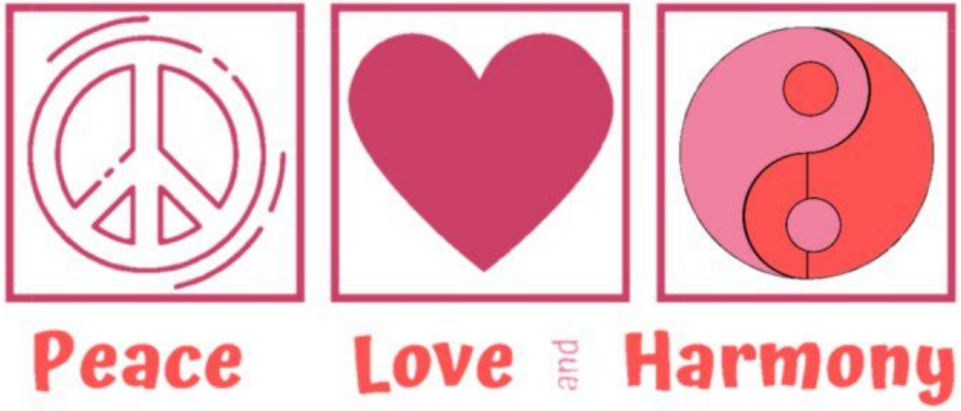
Let’s build it ourselves!

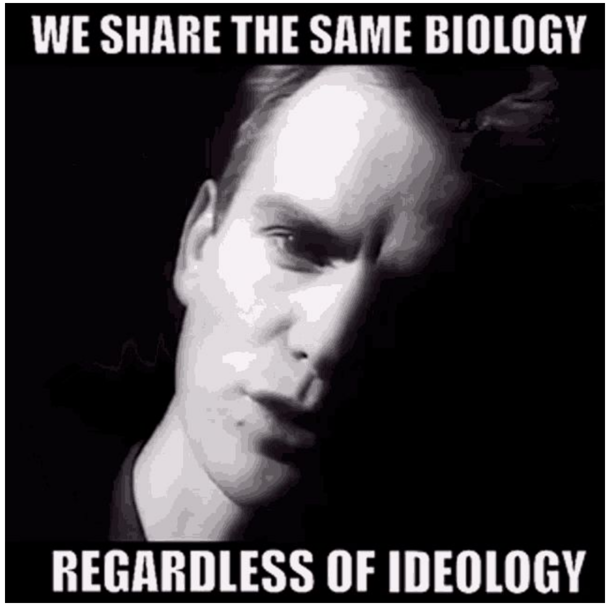
Why is nobody using our data platform?

In the end, we all want the same

Business

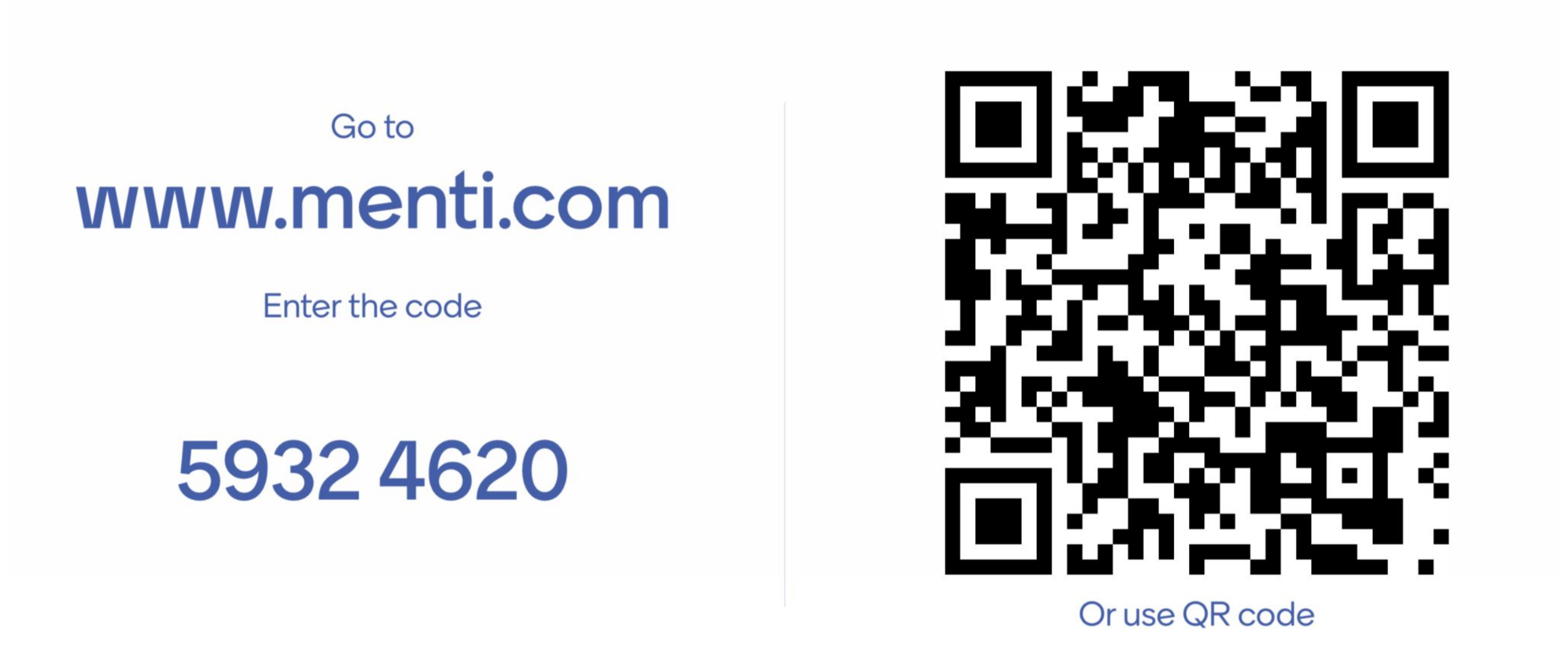
IT department





How to break down walls and create mutual understanding (in data)

Instructions



Go to

www.menti.com

Enter the code

5932 4620

Or use QR code

Practical approaches about how to make your data platform a success

1. **Tie your platform to the strategic goals of the company**
   Look for reasons to do things instead of finding reasons not to do it

2. **Embrace a platform thinking approach**
   The platform mindset and a roadmap on how to get there

3. **How do you get started?**
   Practical tips on how to approach this

4. **Balancing the act of centralization and decentralization**
   Values of Freedom and Control are not conflicting

Link data platform design to your strategic goals

Leverage your current company goals

Improve existing operations

Create innovative services

Become more data driven

Get closer to customers

All take a huge advantage of data analytics and data products

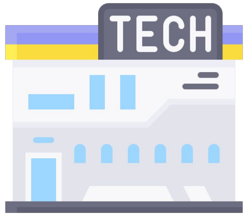
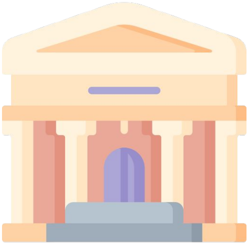
BUT…

How can we **leverage all data** that we have available to us?

How can we make sure that data projects are developed in a **technical sustainable and scalable** way?

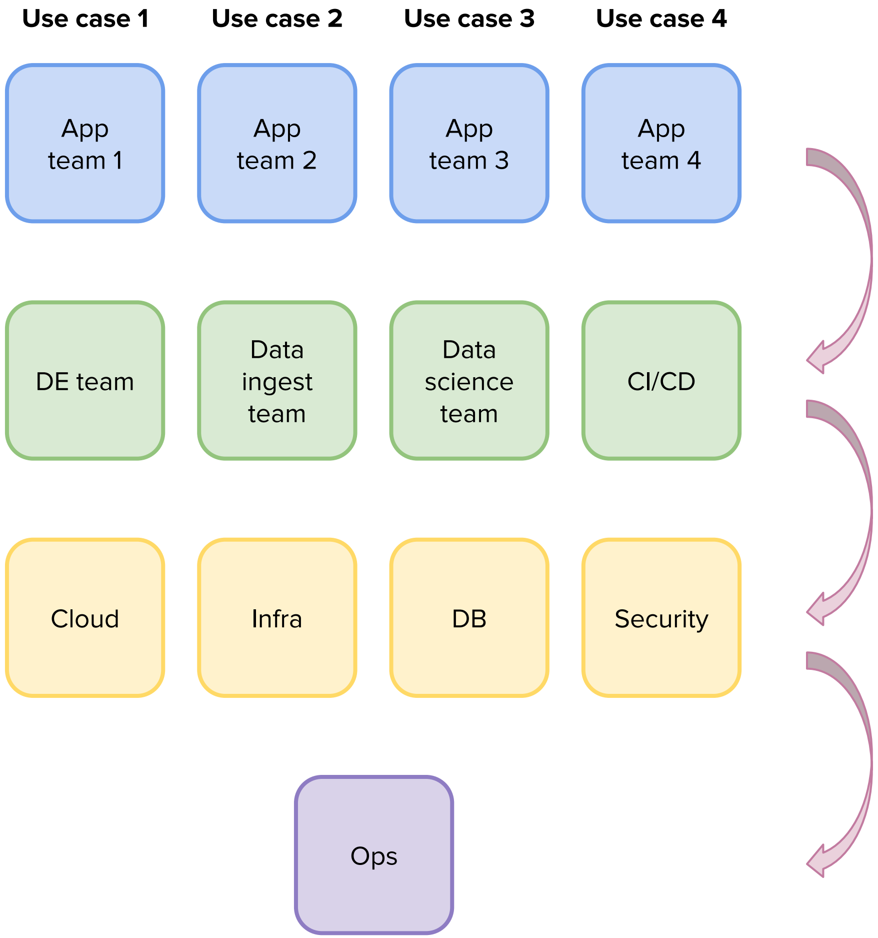
How can the business **focus on business challenges** instead of engineering and IT operational issues inherent to the data worlds?

Every company is different

			
Budget	✅	✅✅✅✅	✅✅
Synergy and reuse	✅	✅✅✅✅	✅✅✅
Agility	✅✅✅✅	✅✅	✅
Tech skills across organisation	✅✅✅✅	✅✅	✅
Legal / Governance requirements	✅	✅✅✅	✅✅✅✅
Data platform	❌	✅ (Shared)	✅ (# Instances)

Change how business and IT interact with each other

From delegating responsibilities to other teams



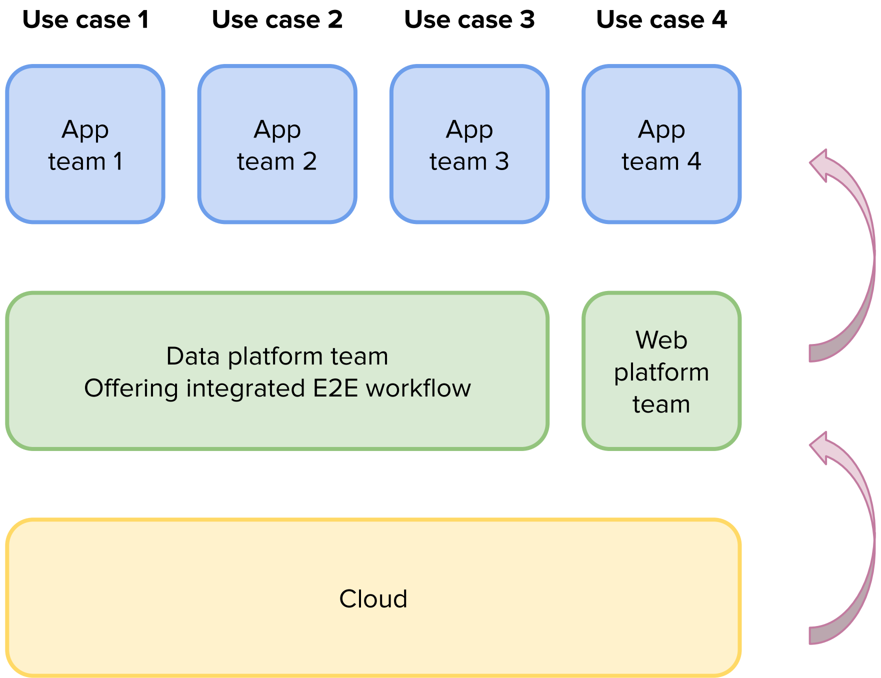
Use case 1	Use case 2	Use case 3	Use case 4
App team 1	App team 2	App team 3	App team 4
DE team	Data ingest team	Data science team	CI/CD
Cloud	Infra	DB	Security

Ops

Change how business and IT interact with each other

To offering capabilities that enable teams to work effectively

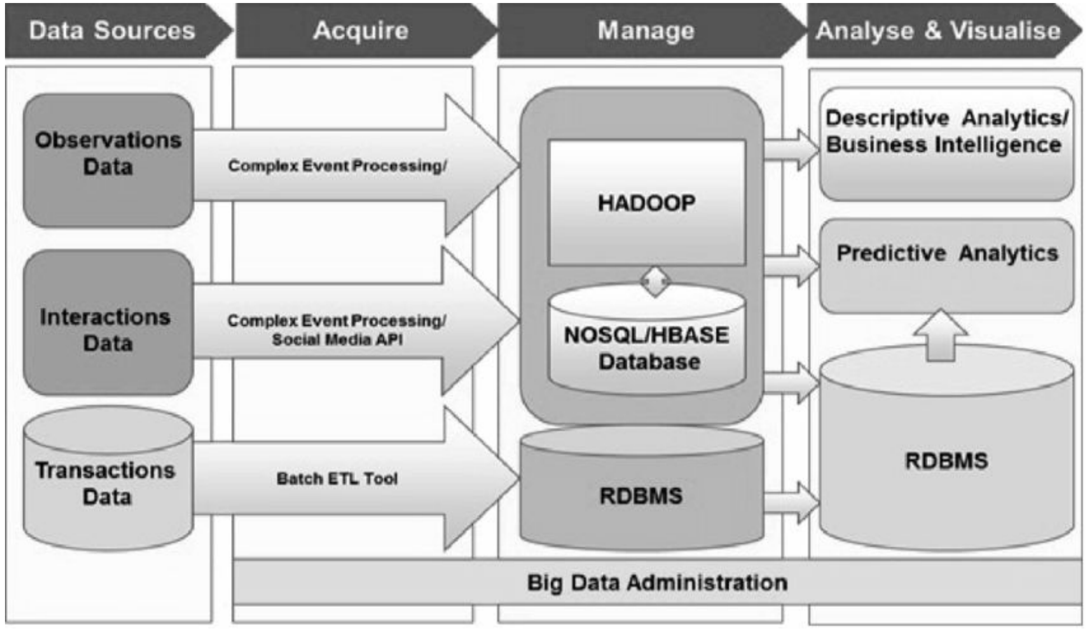
IT as enabler instead of bottleneck



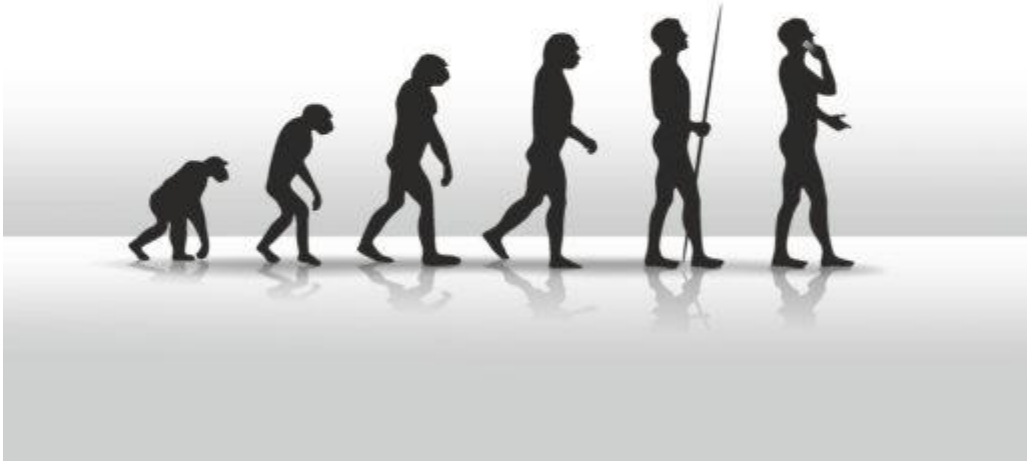
Make sure you are building the right capabilities

Beware Big Design Upfront & Analysis Paralysis

> “No architecture survives first contact with developers”



This is a wish list, not an architecture!

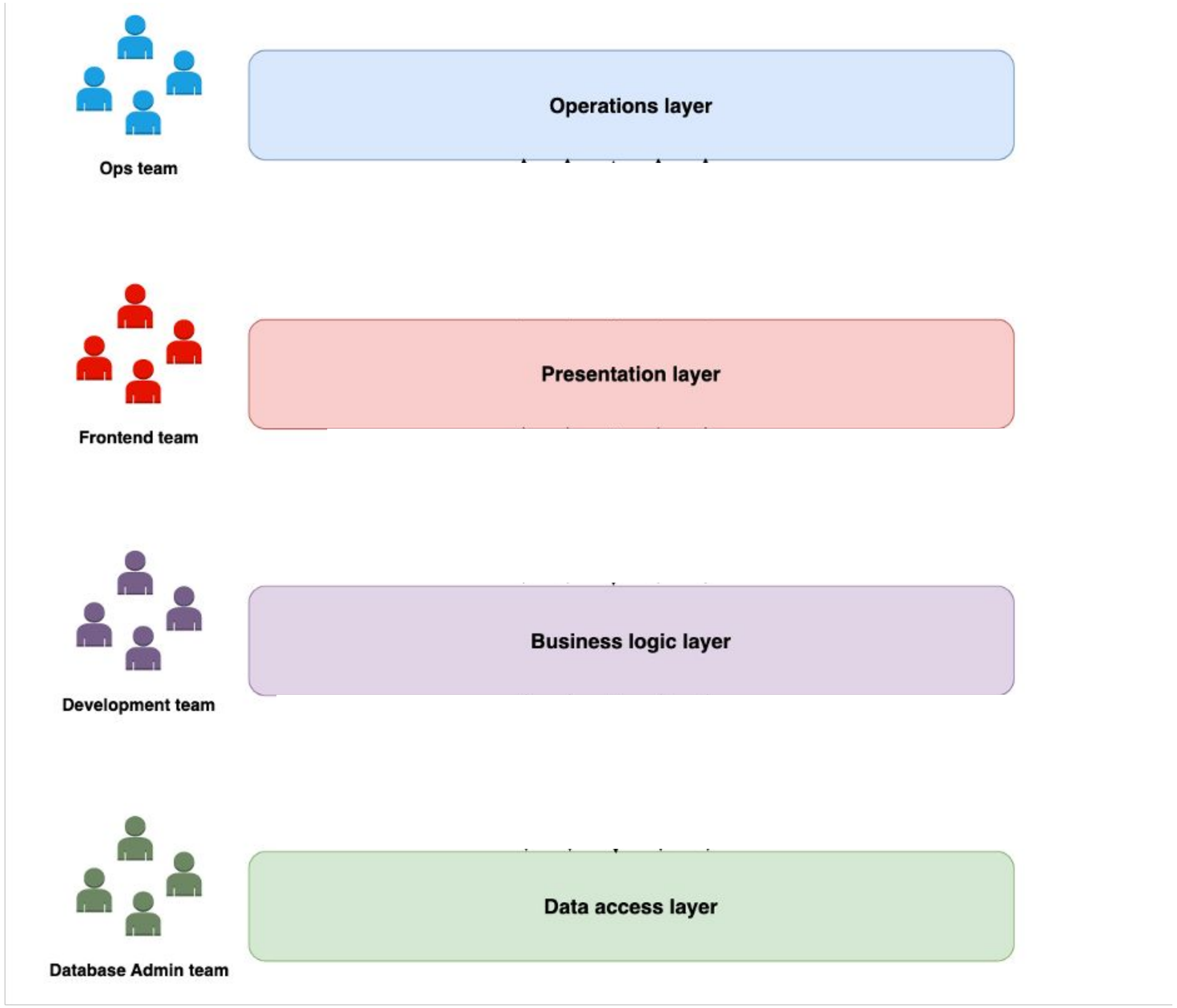


Accept from the beginning that your architecture will evolve, based on what you will learn. Start small, show value, lure in others!

“Build the plane while flying it!”

Embrace a platform thinking approach

Classic IT departments often operate in silos with different teams owning the different key layers of the end product



Ops team

Operations layer

Frontend team

Presentation layer

Development team

Business logic layer

Database Admin team

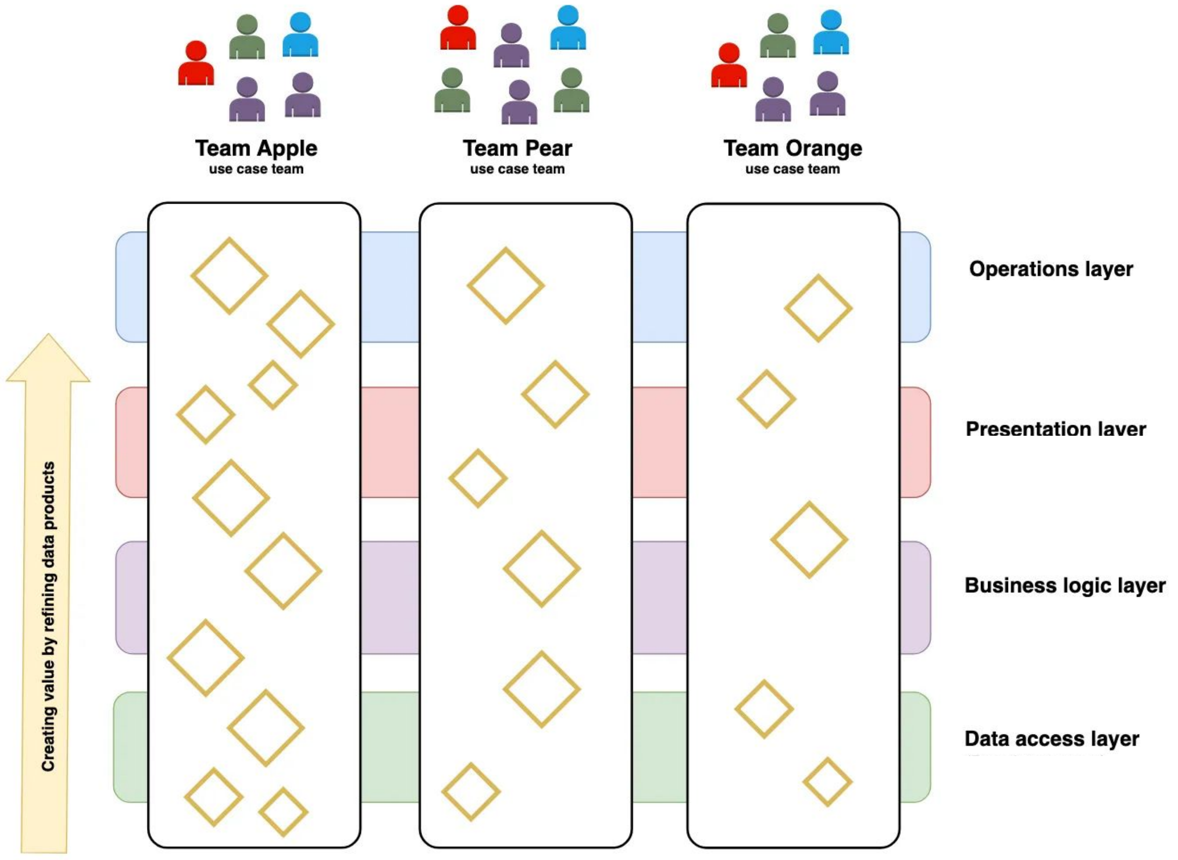
Data access layer

🤨 **Forgetting about the end-users.**

🧭 **Limited ownership and accountability on decisions.**

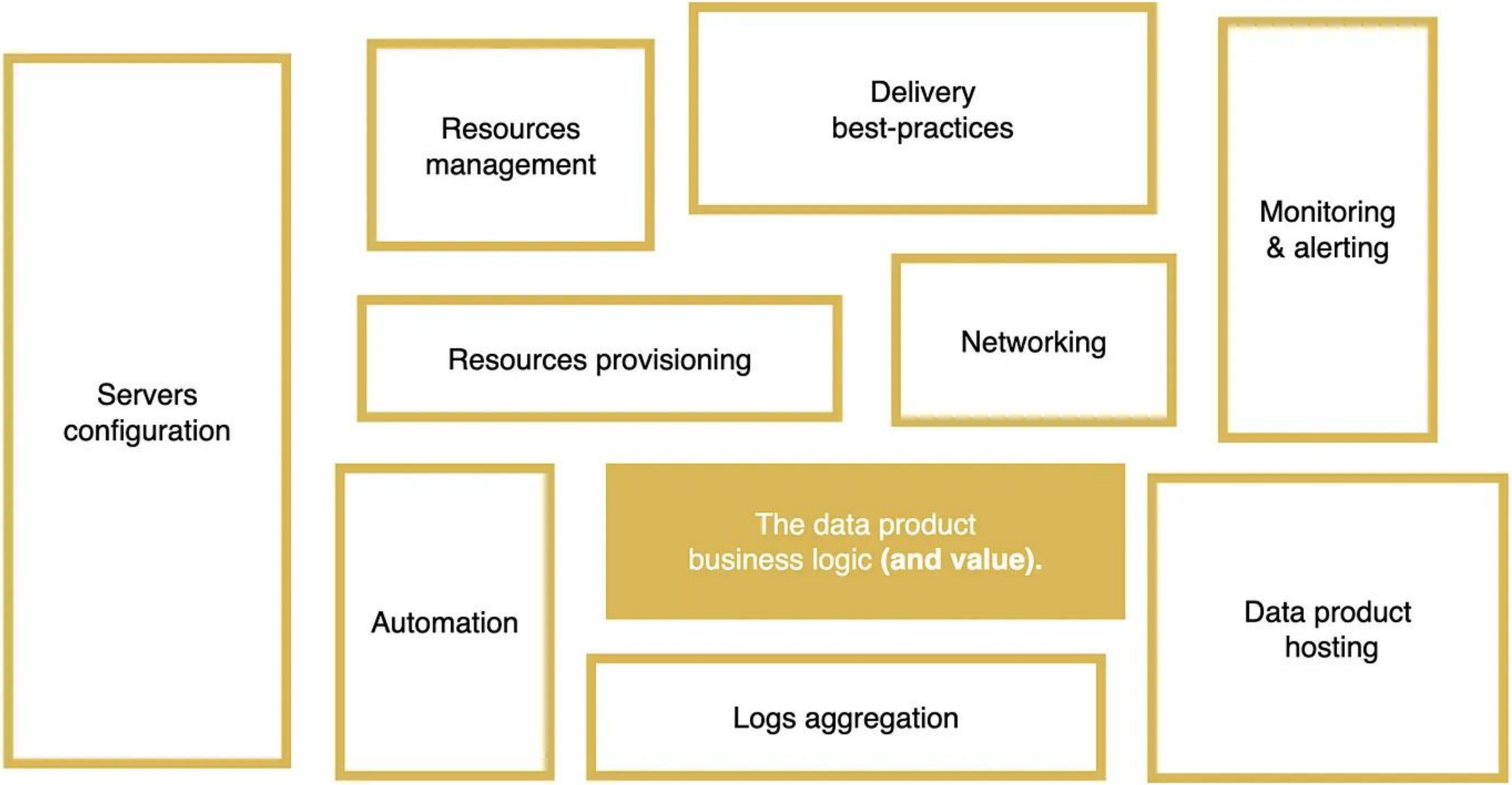
🚢 **Lack of agility & business adaptability.**

Moving towards long-lived product-oriented teams where people with different expertise work together towards a shared goal

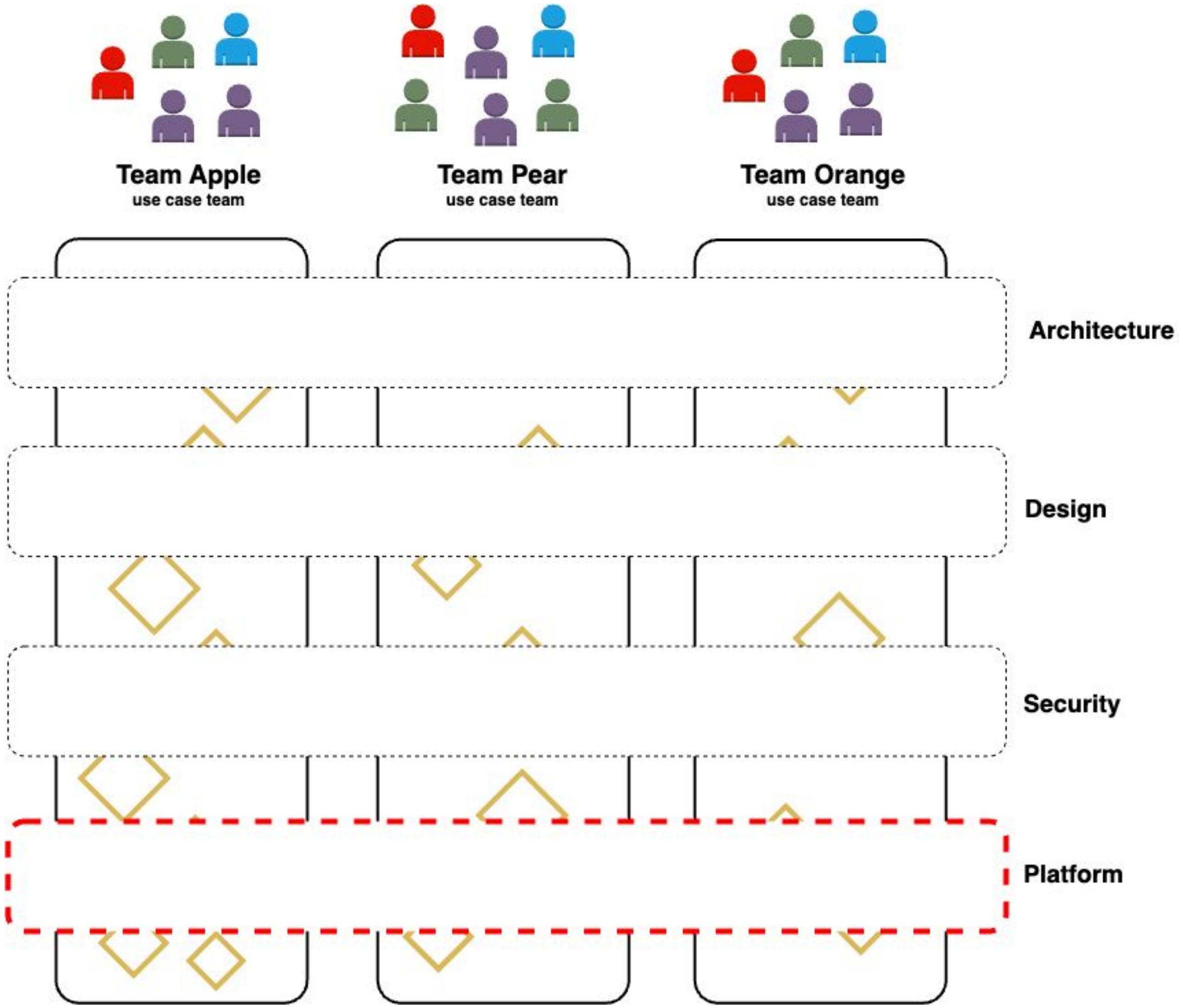


- A stronger feeling of ownership.
- Optimised for flow of change.
- More motivation in day-to-day tasks
- Developing skills enabling end-to-end value creation.

The main issue with autonomous product-oriented teams is that teams will spend a lot of their day-to-day work on tasks not directly impacting business value. We call that undifferentiated heavy lifting.



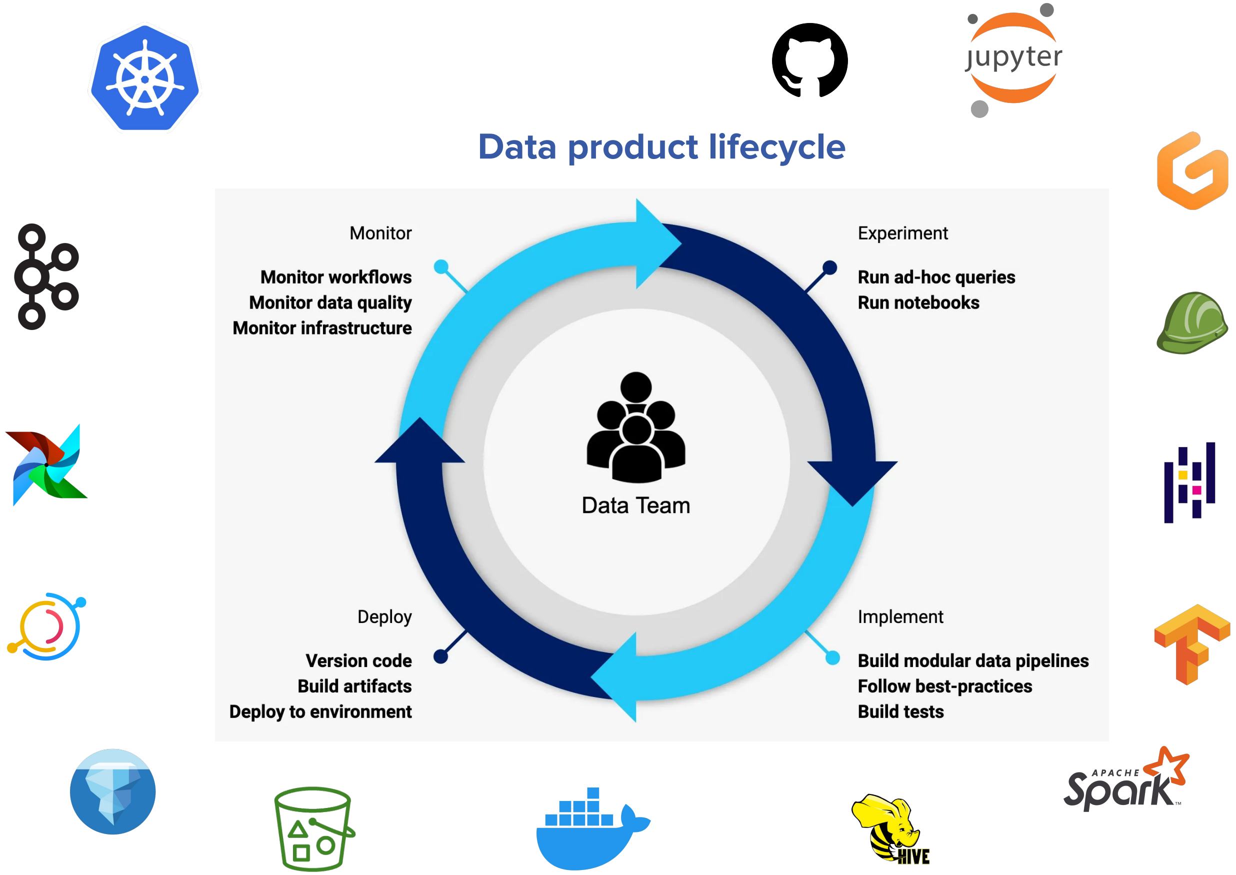
This is where the idea of a platform and enabling teams comes in. They support the product-oriented teams in doing their job.



https://medium.com/datamindedbe/the-data-product-lifecycle-4903c9752527

Make the easy things the rights things by providing paved roads

A data platform is not a bunch of tools randomly thrown together



Data product lifecycle

Monitor

Monitor workflows
Monitor data quality
Monitor infrastructure

Experiment

Run ad-hoc queries
Run notebooks

Data Team

Deploy

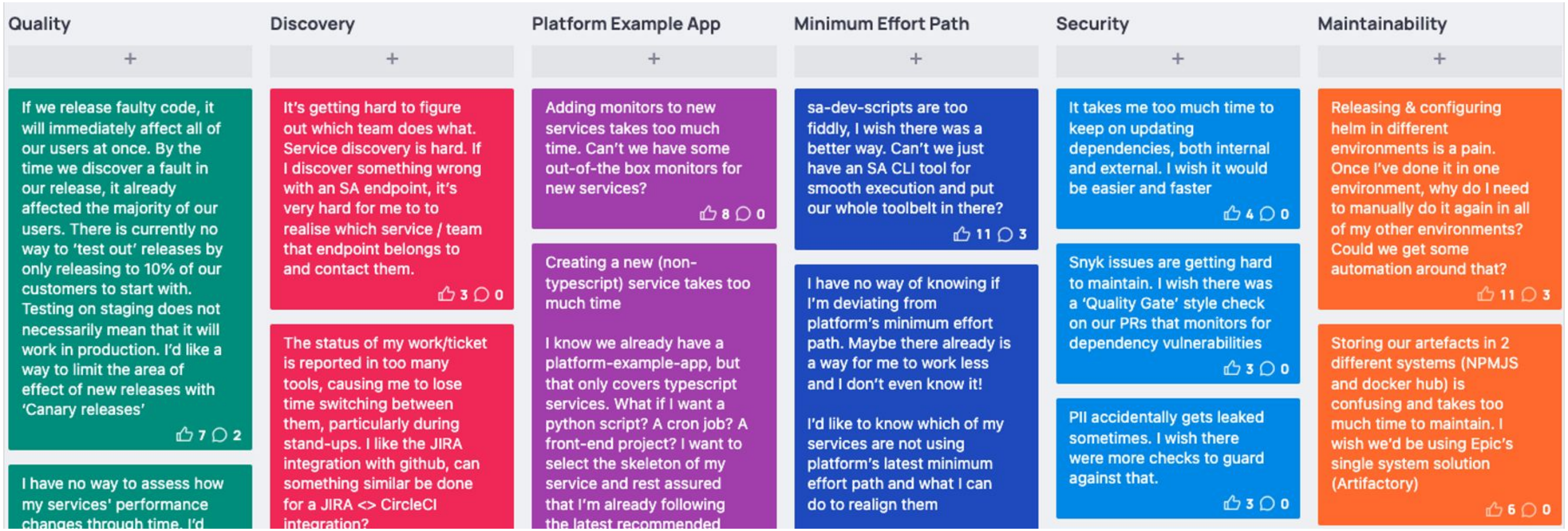
Version code
Build artifacts
Deploy to environment

Implement

Build modular data pipelines
Follow best-practices
Build tests

A data platform should offer an ***integrated experience*** that guides them through all steps of the data product lifecycle by offering ***paved roads***

Don’t force people to use the platform



Quality

+

If we release faulty code, it will immediately affect all of our users at once. By the time we discover a fault in our release, it already affected the majority of our users. There is currently no way to ‘test out’ releases by only releasing to 10% of our customers to start with. Testing on staging does not necessarily mean that it will work in production. I’d like a way to limit the area of effect of new releases with ‘Canary releases’

👍 7 💬 2

I have no way to assess how my services’ performance changes through time. I’d

Discovery

+

It’s getting hard to figure out which team does what. Service discovery is hard. If I discover something wrong with an SA endpoint, it’s very hard for me to to realise which service / team that endpoint belongs to and contact them.

👍 3 💬 0

The status of my work/ticket is reported in too many tools, causing me to lose time switching between them, particularly during stand-ups. I like the JIRA integration with github, can something similar be done for a JIRA <> CircleCI integration?

Platform Example App

+

Adding monitors to new services takes too much time. Can’t we have some out-of-the box monitors for new services?

👍 8 💬 0

Creating a new (non-typescript) service takes too much time

I know we already have a platform-example-app, but that only covers typescript services. What if I want a python script? A cron job? A front-end project? I want to select the skeleton of my service and rest assured that I’m already following the latest recommended

Minimum Effort Path

+

sa-dev-scripts are too fiddly, I wish there was a better way. Can’t we just have an SA CLI tool for smooth execution and put our whole toolbelt in there?

👍 11 💬 3

I have no way of knowing if I’m deviating from platform’s minimum effort path. Maybe there already is a way for me to work less and I don’t even know it!

I’d like to know which of my services are not using platform’s latest minimum effort path and what I can do to realign them

Security

+

It takes me too much time to keep on updating dependencies, both internal and external. I wish it would be easier and faster

👍 4 💬 0

Snyk issues are getting hard to maintain. I wish there was a ‘Quality Gate’ style check on our PRs that monitors for dependency vulnerabilities

👍 3 💬 0

PII accidentally gets leaked sometimes. I wish there were more checks to guard against that.

👍 3 💬 0

Maintainability

+

Releasing & configuring helm in different environments is a pain. Once I’ve done it in one environment, why do I need to manually do it again in all of my other environments? Could we get some automation around that?

👍 11 💬 3

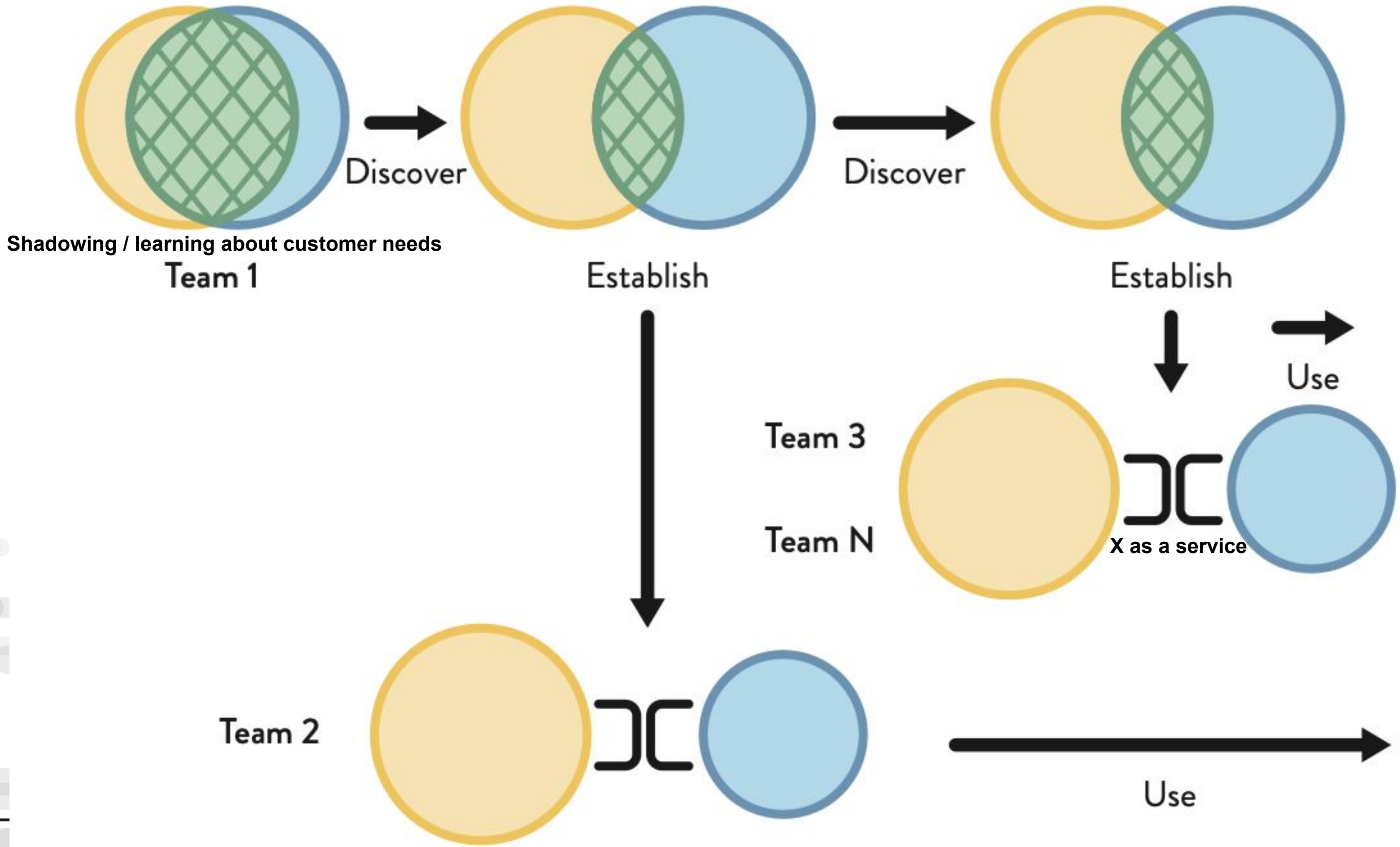
Storing our artefacts in 2 different systems (NPMJS and docker hub) is confusing and takes too much time to maintain. I wish we’d be using Epic’s single system solution (Artifactory)

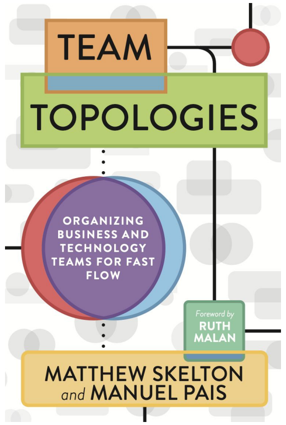
👍 6 💬 0

But make them want to use it.
Involve them in how the platform needs to evolve.

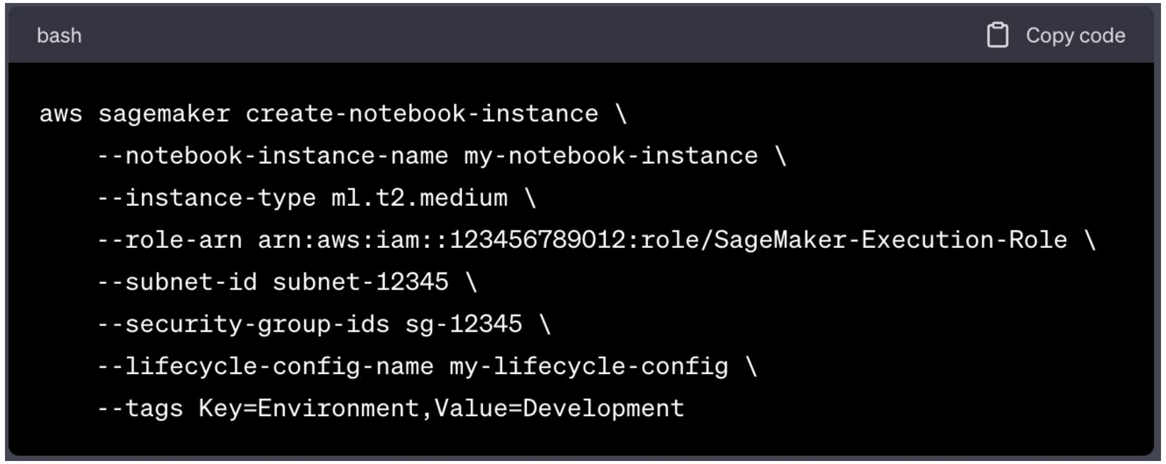
How do you get started?

From close collaboration to limited collaboration (discovery) through to X-as-a-Service for established, predictable delivery

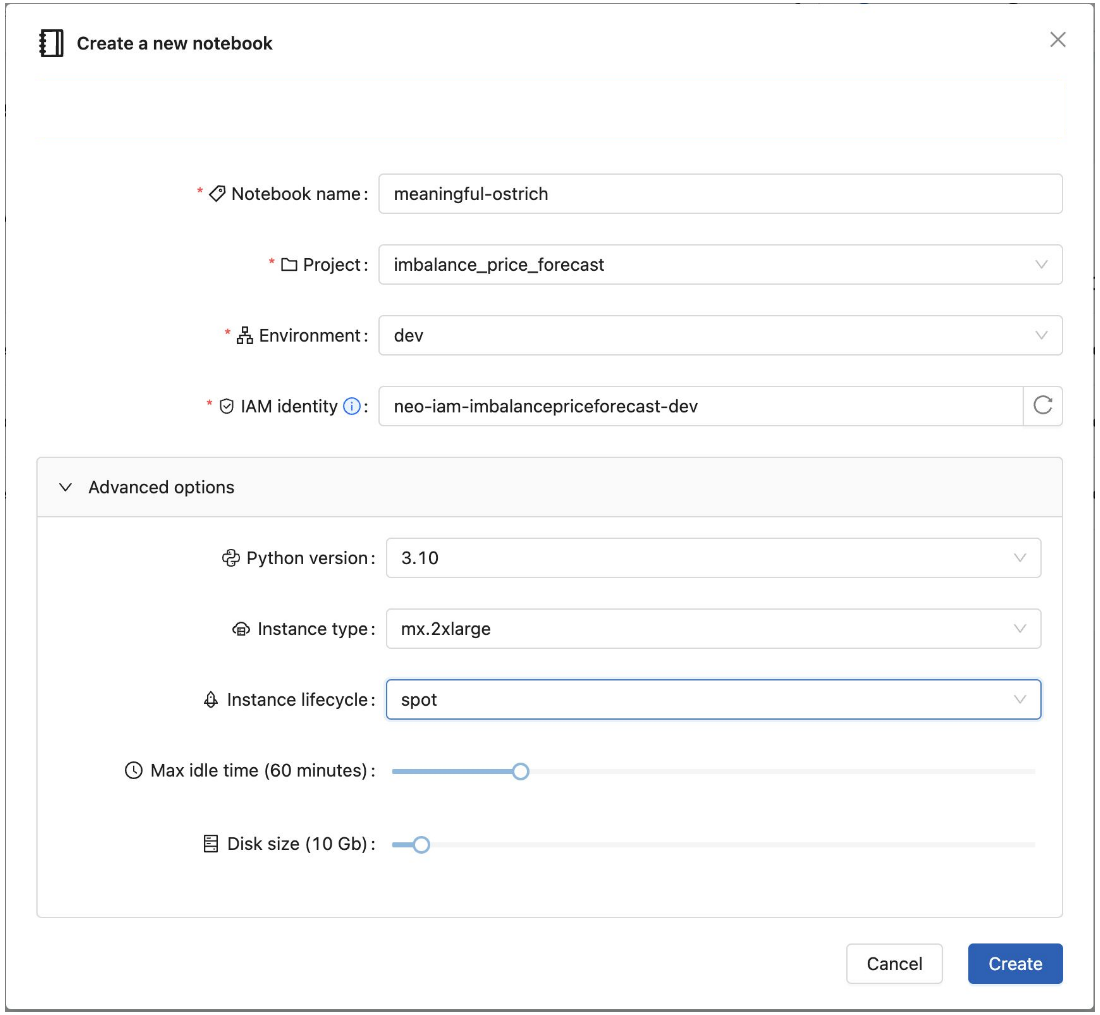



TEAM
TOPOLOGIES
ORGANIZING BUSINESS AND TECHNOLOGY TEAMS FOR FAST FLOW
Foreword by RUTH MALAN
MATTHEW SKELTON and MANUEL PAIS

An example: Going from documenting API calls to experimentation as a service



```bash
aws sagemaker create-notebook-instance \
    --notebook-instance-name my-notebook-instance \
    --instance-type ml.t2.medium \
    --role-arn arn:aws:iam::123456789012:role/SageMaker-Execution-Role \
    --subnet-id subnet-12345 \
    --security-group-ids sg-12345 \
    --lifecycle-config-name my-lifecycle-config \
    --tags Key=Environment,Value=Development
```



Create a new notebook

* Notebook name: meaningful-ostrich

* Project: imbalance_price_forecast

* Environment: dev

* IAM identity: neo-iam-imbalancepriceforecast-dev

Advanced options

Python version: 3.10

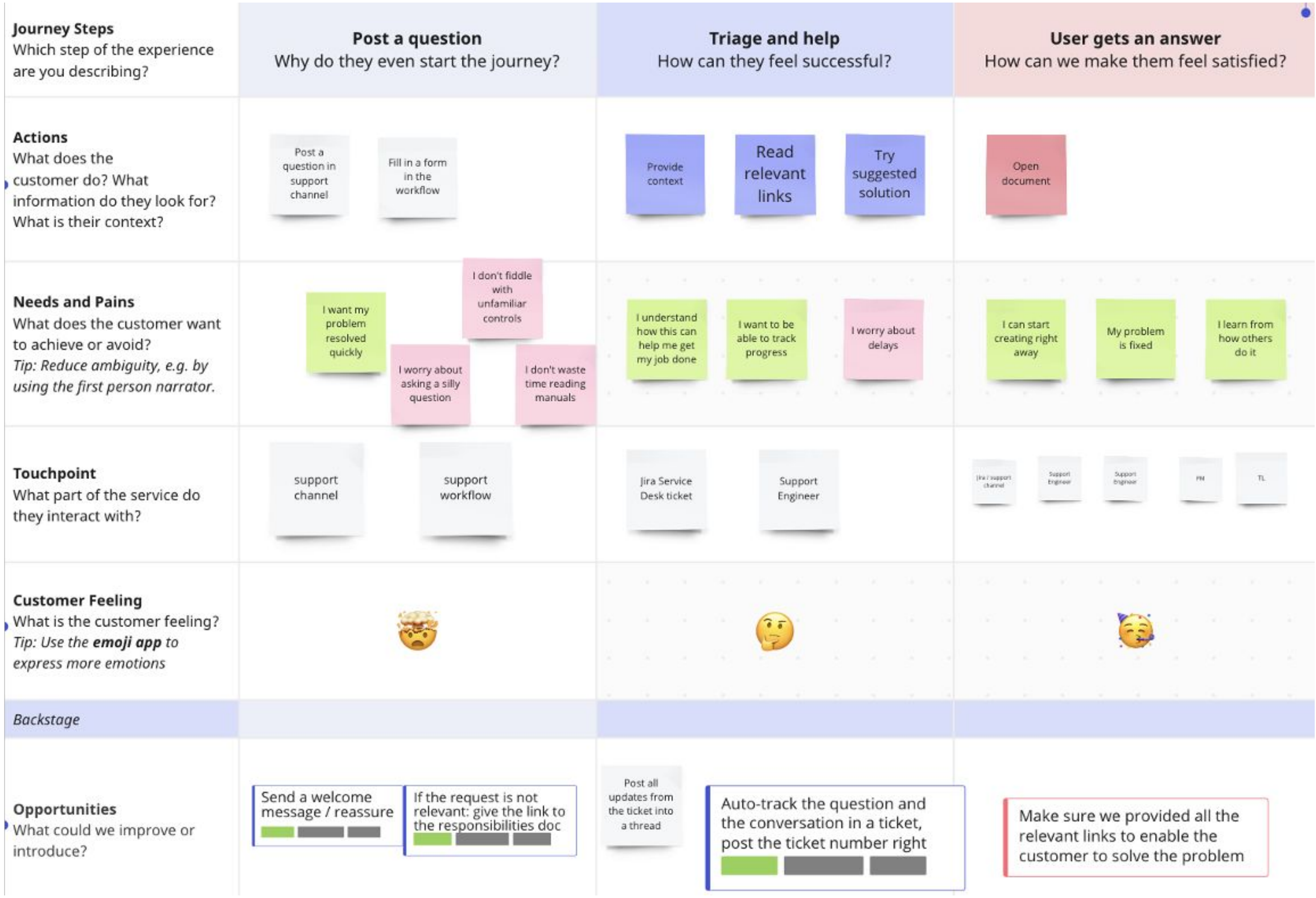
Instance type: mx.2xlarge

Instance lifecycle: spot

Max idle time (60 minutes):

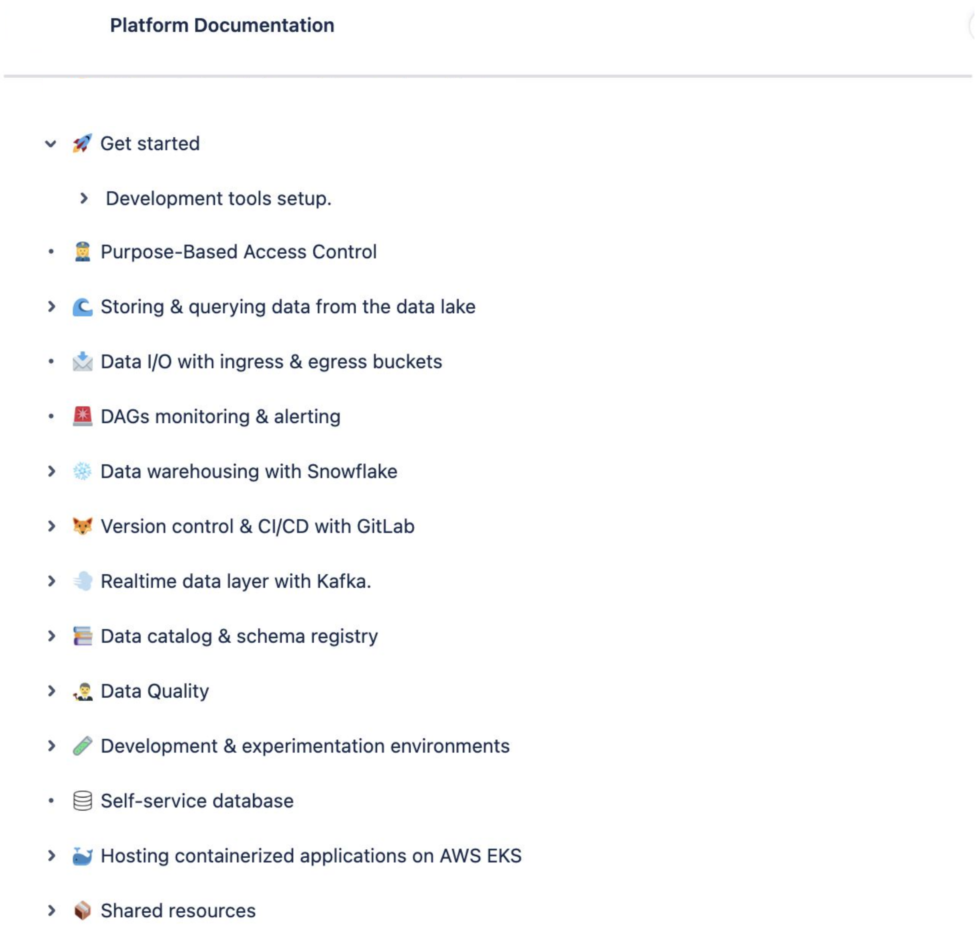
Disk size (10 Gb):

Put yourself in your users shoes



Journey Steps Which step of the experience are you describing?	Post a question Why do they even start the journey?	Triage and help How can they feel successful?	User gets an answer How can we make them feel satisfied?
Actions What does the customer do? What information do they look for? What is their context?	Post a question in support channel Fill in a form in the workflow	Provide context Read relevant links Try suggested solution	Open document
Needs and Pains What does the customer want to achieve or avoid? *Tip: Reduce ambiguity, e.g. by using the first person narrator.*	I don't fiddle with unfamiliar controls I want my problem resolved quickly I worry about asking a silly question I don't waste time reading manuals	I understand how this can help me get my job done I want to be able to track progress I worry about delays	I can start creating right away My problem is fixed I learn from how others do it
Touchpoint What part of the service do they interact with?	support channel support workflow	Jira Service Desk ticket Support Engineer	Jira / support channel Support Engineer Support Engineer PM TL
Customer Feeling What is the customer feeling? *Tip: Use the **emoji app** to express more emotions*	🤯	🤔	🥳
Backstage			
Opportunities What could we improve or introduce?	Send a welcome message / reassure If the request is not relevant: give the link to the responsibilities doc	Post all updates from the ticket into a thread Auto-track the question and the conversation in a ticket, post the ticket number right	Make sure we provided all the relevant links to enable the customer to solve the problem

Self-service documentation is a first class deliverable.



Platform Documentation

- Get started
  - Development tools setup.
- Purpose-Based Access Control
- Storing & querying data from the data lake
- Data I/O with ingress & egress buckets
- DAGs monitoring & alerting
- Data warehousing with Snowflake
- Version control & CI/CD with GitLab
- Realtime data layer with Kafka.
- Data catalog & schema registry
- Data Quality
- Development & experimentation environments
- Self-service database
- Hosting containerized applications on AWS EKS
- Shared resources

Audit regularly.

Make it simple.

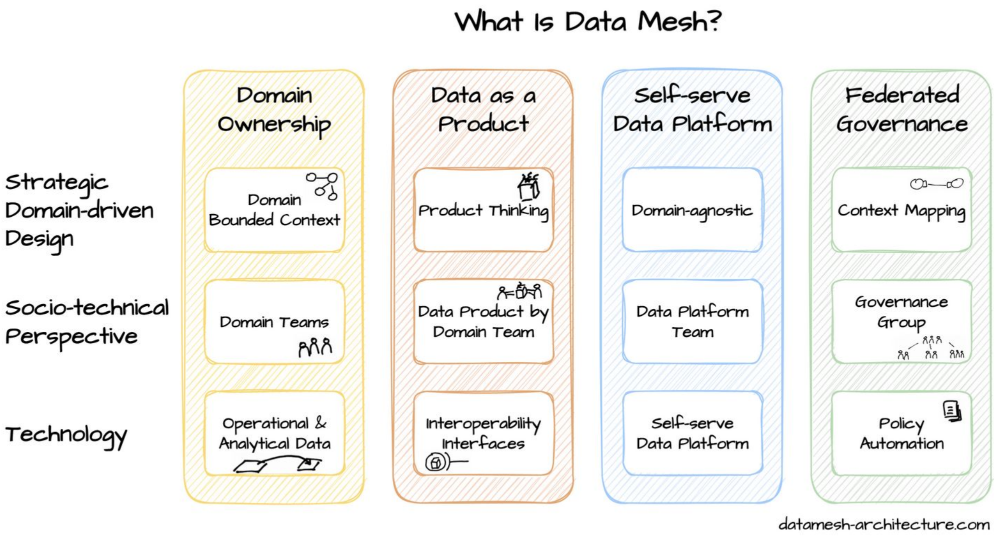
Have an onboarding exercise.

Have consistent formats.

Collect feedback on every page.

Balancing the act of centralization and decentralization

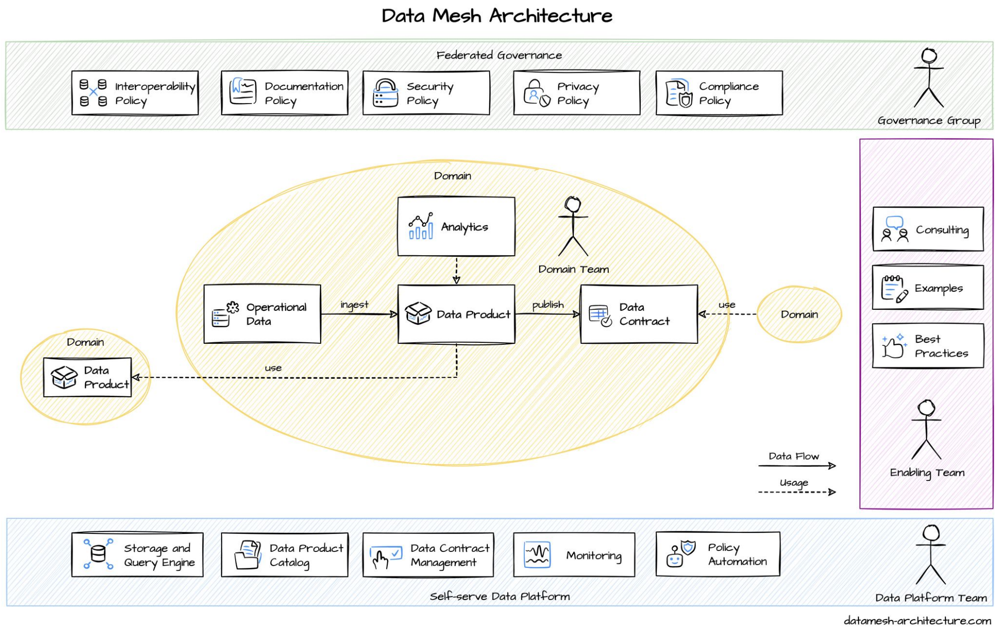
Data Mesh or Data Mess?



What Is Data Mesh?

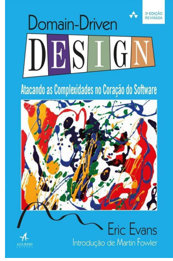
	Domain Ownership	Data as a Product	Self-serve Data Platform	Federated Governance
Strategic Domain-driven Design	Domain Bounded Context	Product Thinking	Domain-agnostic	Context Mapping
Socio-technical Perspective	Domain Teams	Data Product by Domain Team	Data Platform Team	Governance Group
Technology	Operational & Analytical Data	Interoperability Interfaces	Self-serve Data Platform	Policy Automation

datamesh-architecture.com



Data Mesh Architecture

datamesh-architecture.com



Domain-Driven
DESIGN
Atacando as Complexidades no Coração do Software
Eric Evans
Introdução de Martin Fowler

Data mesh is not a religion: Don't be dogmatic in your principles and keep a flexible mindset.

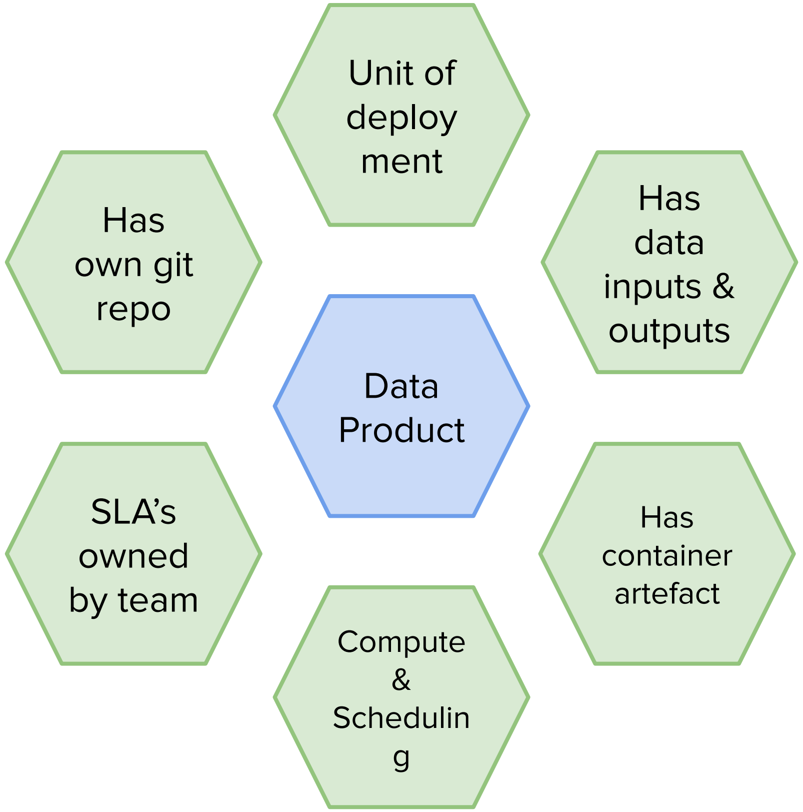
Take a practical approach: See what works and skip what doesn't work for you

Do not decentralise everything: Data mesh principles allow for shared capabilities

? **Be careful of ambiguous terminology**: What is a data domain and how granular is it?

Processing vs Data ownership: Who "owns" a derived data product using data from other domains?

Introduce simple concepts that are repeatable and easy to understand



Data
Product

Unit of
deploy
ment

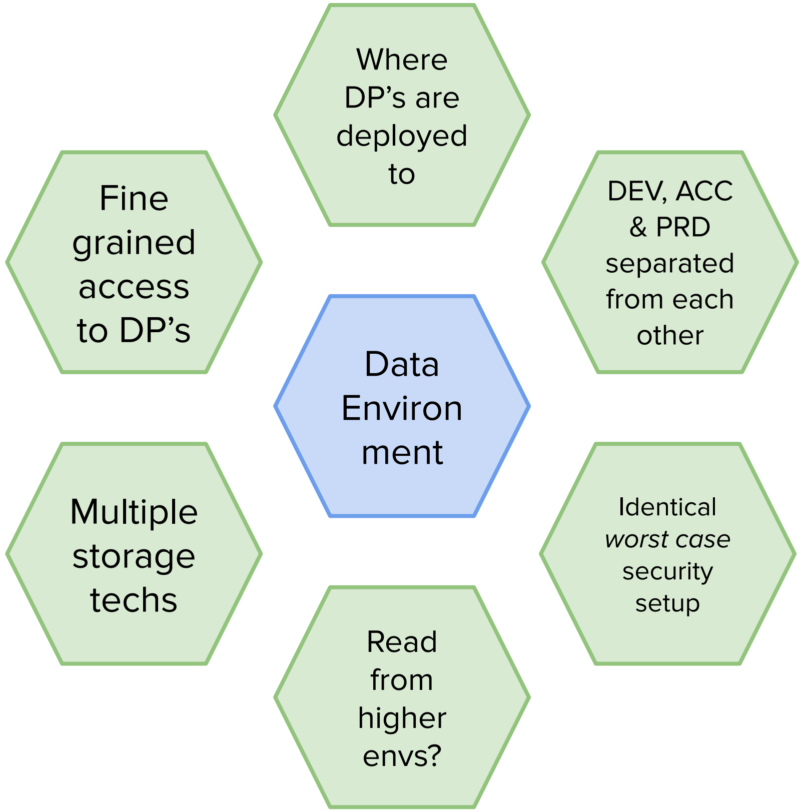
Has
data
inputs &
outputs

Has
container
artefact

Compute
&
Schedulin
g

SLA's
owned
by team

Has
own git
repo



Data
Environ
ment

Where
DP's are
deployed
to

DEV, ACC
& PRD
separated
from each
other

Identical
worst case
security
setup

Read
from
higher
envs?

Multiple
storage
techs

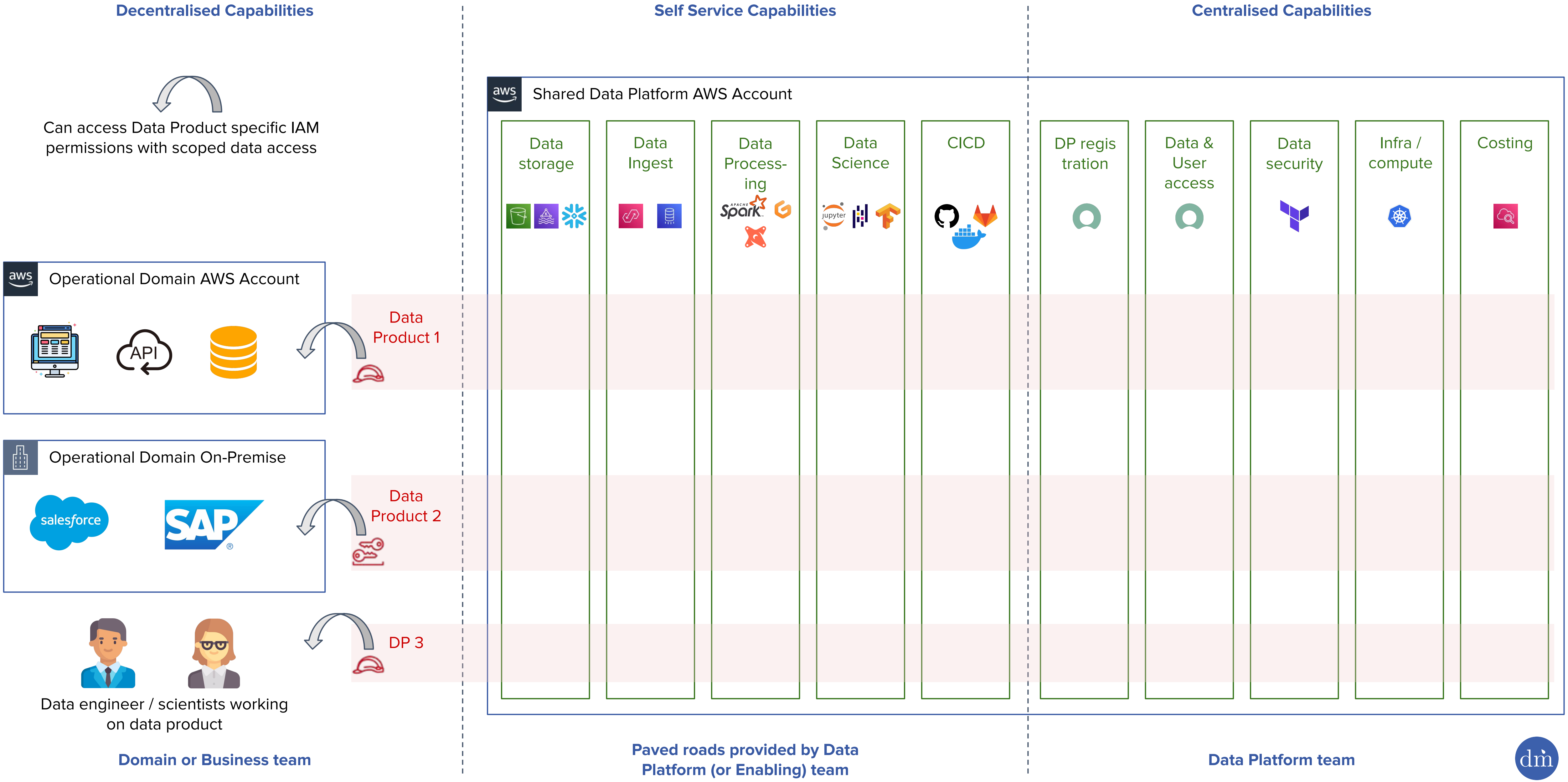
Fine
grained
access
to DP's

Infrastructure environment

Decide upfront how you want to offer the capabilities of your data platform

Data platform capability	Centralised	Self service	Decentralised
Ingestion		●	
Processing	●		
Access & sharing	●		
Security	●		
ML / AI capabilities		●	
BI capabilities		●	
Data modelling			●
Dashboards		●	
Build/release/deploy			●
API hosting		●	
Data product serving		●	
Governance	●		

An example (AWS Focussed)



Decentralised Capabilities

Self Service Capabilities

Centralised Capabilities

Can access Data Product specific IAM permissions with scoped data access

Operational Domain AWS Account

Operational Domain On-Premise

Data engineer / scientists working on data product

Data Product 1

Data Product 2

DP 3

Shared Data Platform AWS Account

Data storage

Data Ingest

Data Process-ing

Data Science

CICD

DP regis tration

Data & User access

Data security

Infra / compute

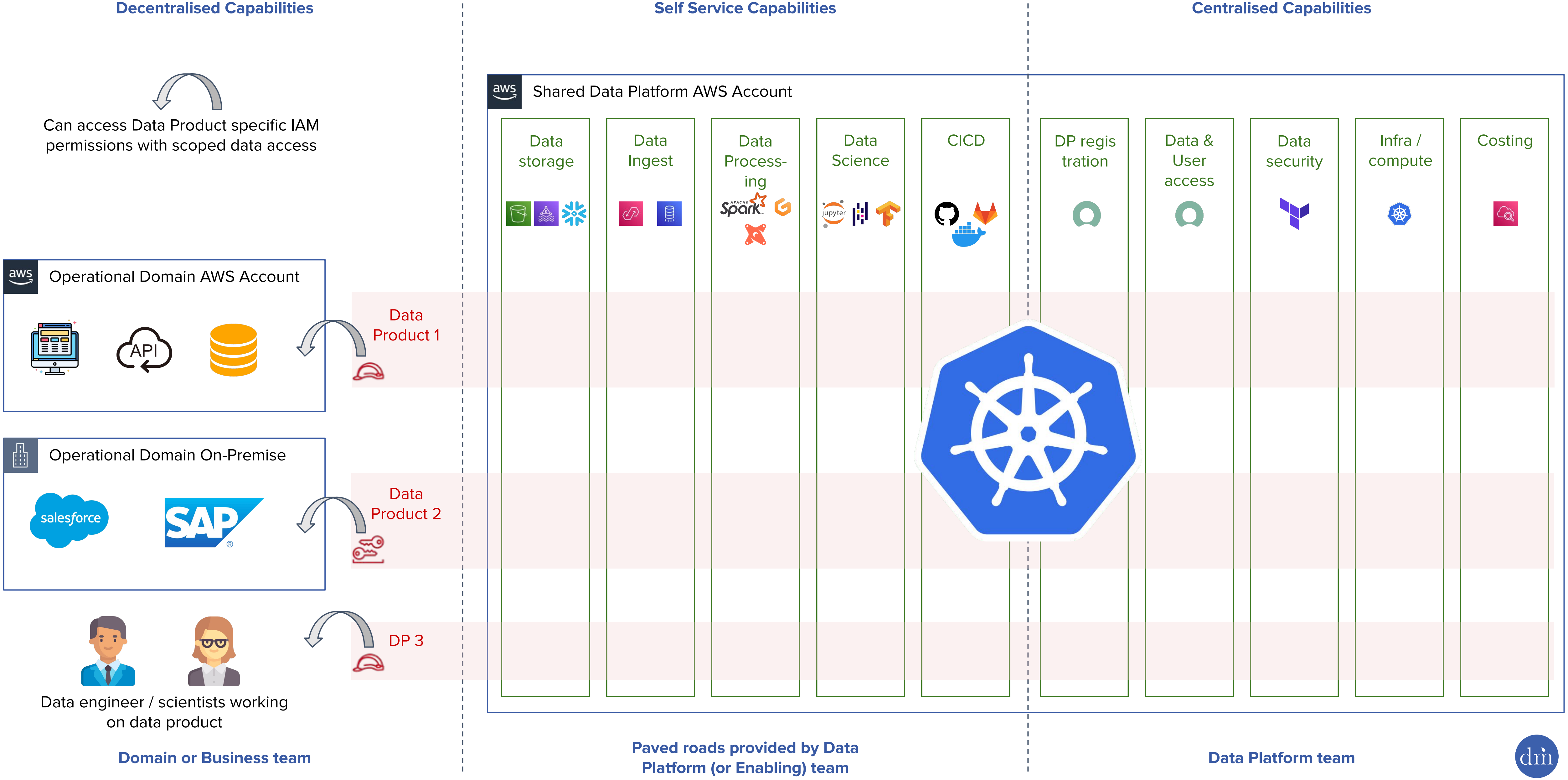
Costing

Domain or Business team

Paved roads provided by Data Platform (or Enabling) team

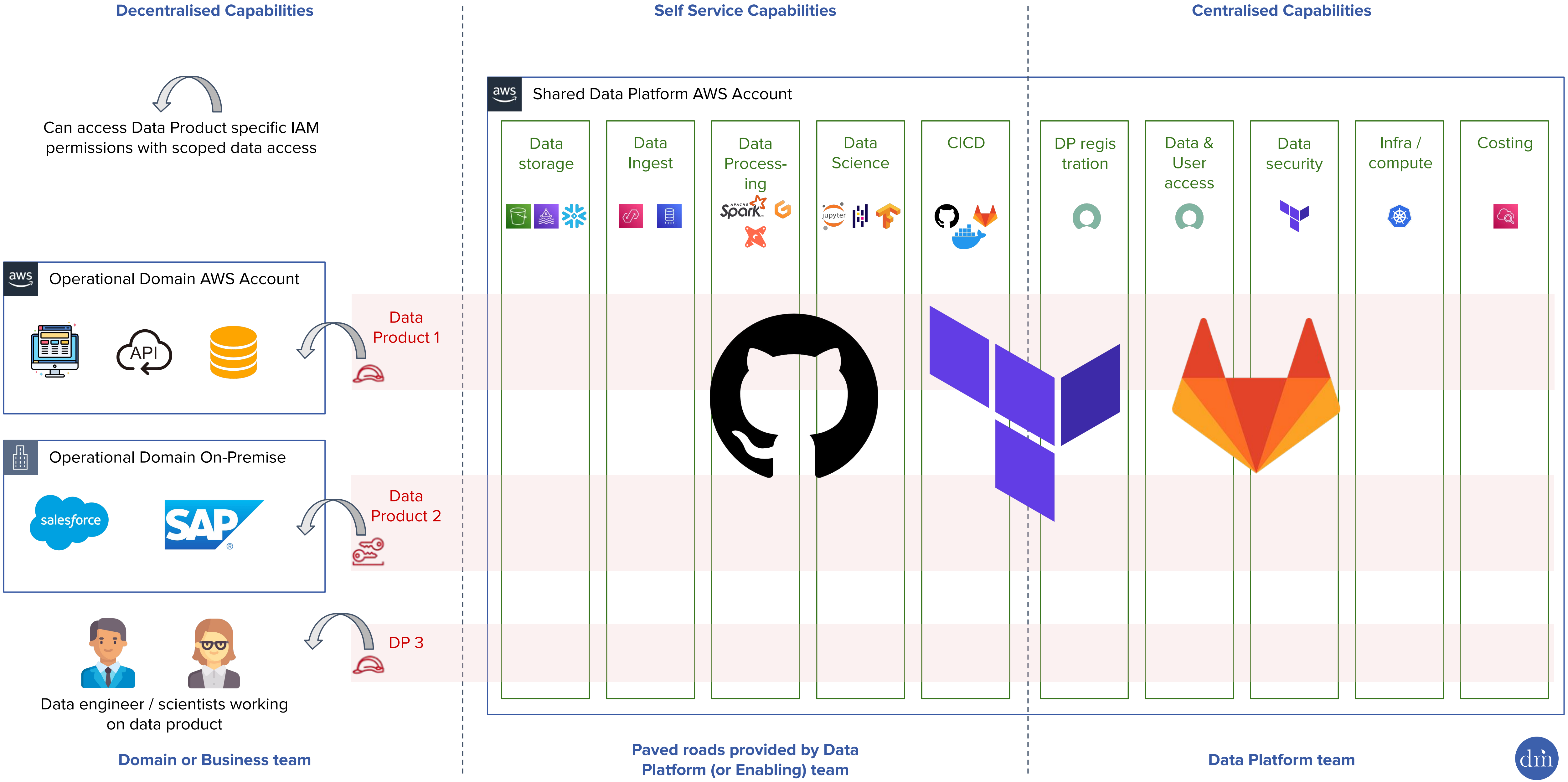
Data Platform team

Pro tip: Leverage Kubernetes to manage separation



Decentralised Capabilities
Self Service Capabilities
Centralised Capabilities
Can access Data Product specific IAM permissions with scoped data access
Shared Data Platform AWS Account
Data storage
Data Ingest
Data Processing
Data Science
CICD
DP registration
Data & User access
Data security
Infra / compute
Costing
Operational Domain AWS Account
API
Data Product 1
Operational Domain On-Premise
salesforce
SAP
Data Product 2
DP 3
Data engineer / scientists working on data product
Domain or Business team
Paved roads provided by Data Platform (or Enabling) team
Data Platform team

Pro tip: Leverage automation to do this at scale and integrate all components



Decentralised Capabilities

Self Service Capabilities

Centralised Capabilities

Can access Data Product specific IAM permissions with scoped data access

Operational Domain AWS Account

Operational Domain On-Premise

Data engineer / scientists working on data product

Domain or Business team

Shared Data Platform AWS Account

Data storage

Data Ingest

Data Process-ing

Data Science

CICD

DP registration

Data & User access

Data security

Infra / compute

Costing

Data Product 1

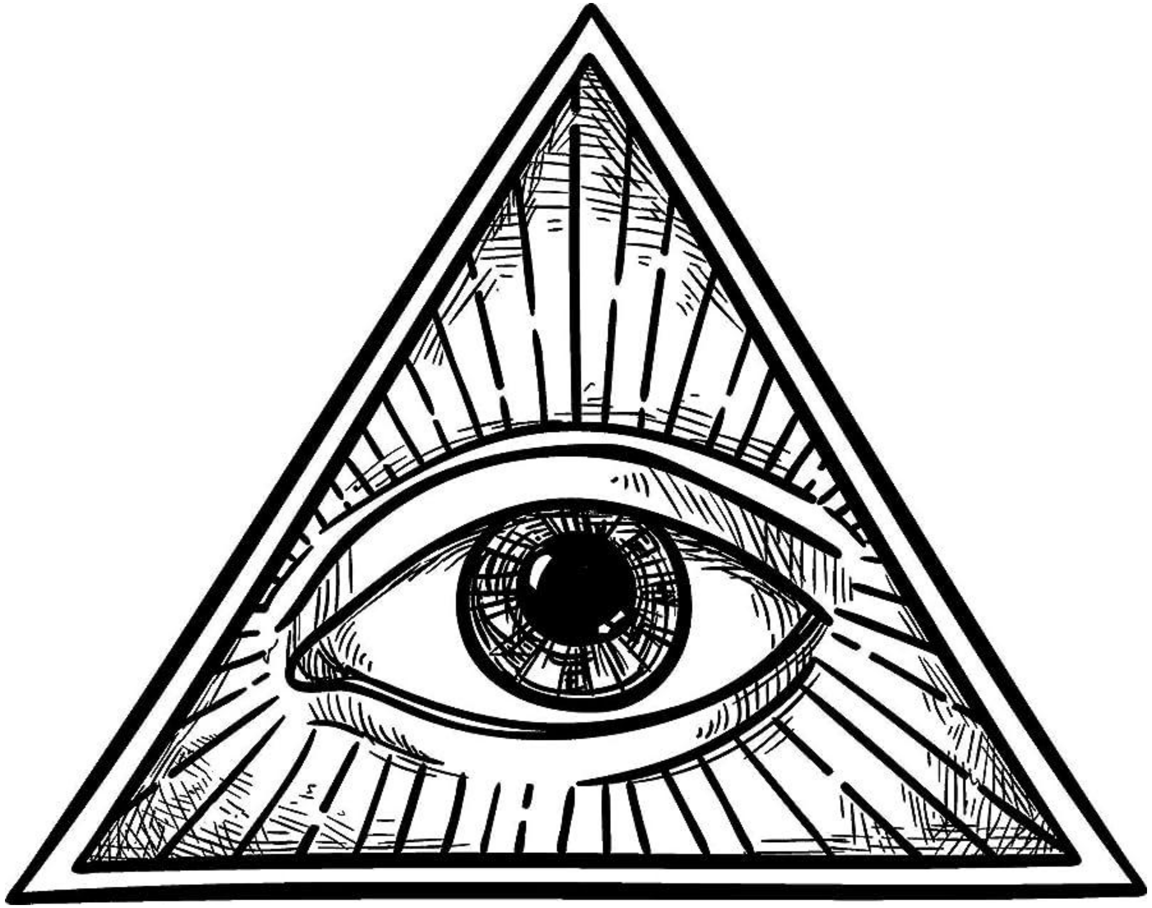
Data Product 2

DP 3

Paved roads provided by Data Platform (or Enabling) team

Data Platform team

If done properly you can strike the right balance...

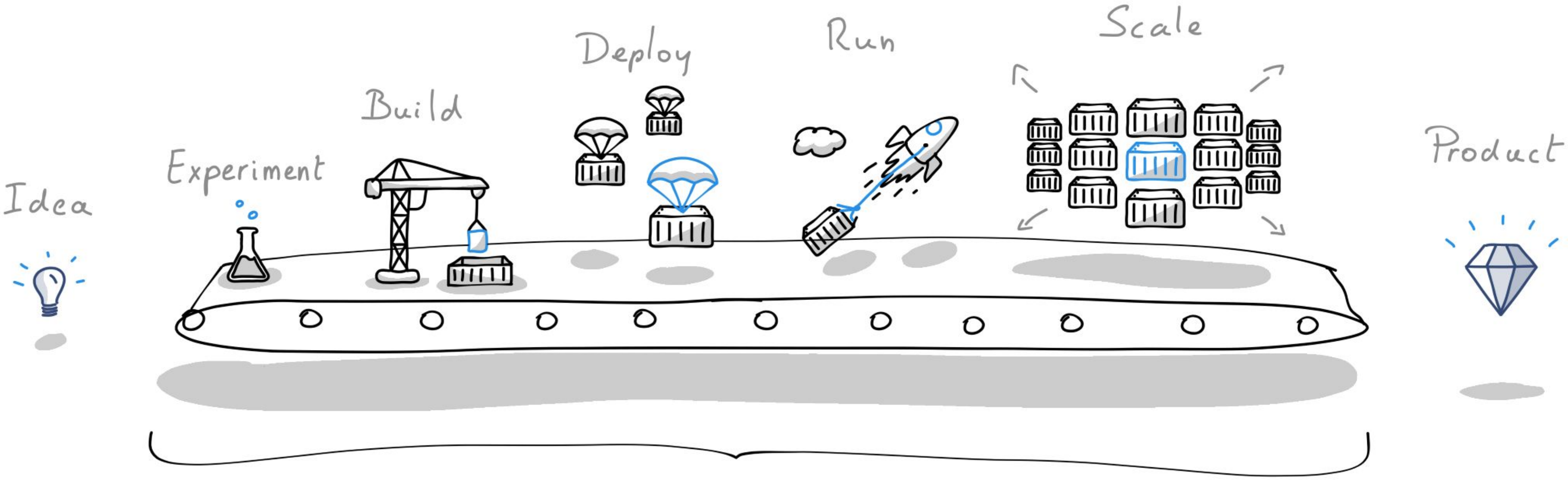


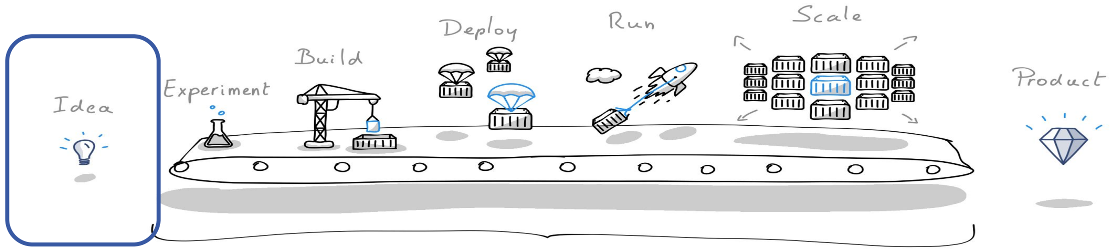
Centralised

Self Service

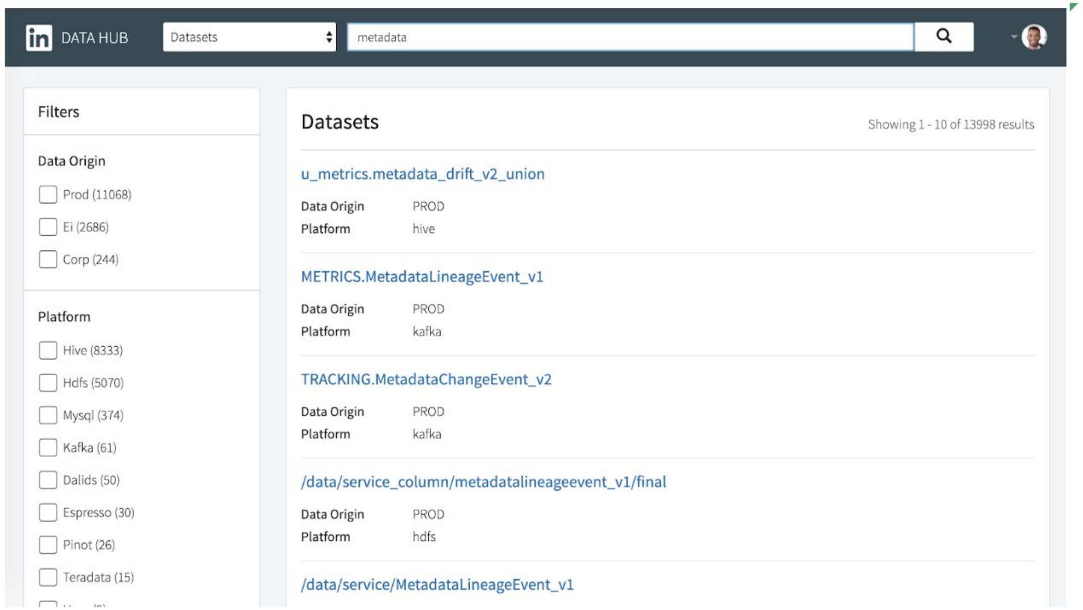
Decentralised

Example of a paved road


Idea
Experiment
Build
Deploy
Run
Scale
Product



Browse available data sources in the data catalog



Filters

Data Origin

- □ Prod (11068)
- □ Ei (2686)
- □ Corp (244)

Platform

- □ Hive (8333)
- □ Hdfs (5070)
- □ Mysql (374)
- □ Kafka (61)
- □ Dalids (50)
- □ Espresso (30)
- □ Pinot (26)
- □ Teradata (15)

Datasets

Showing 1 - 10 of 13998 results

u_metrics.metadata_drift_v2_union

Data Origin	PROD
Platform	hive

METRICS.MetadataLineageEvent_v1

Data Origin	PROD
Platform	kafka

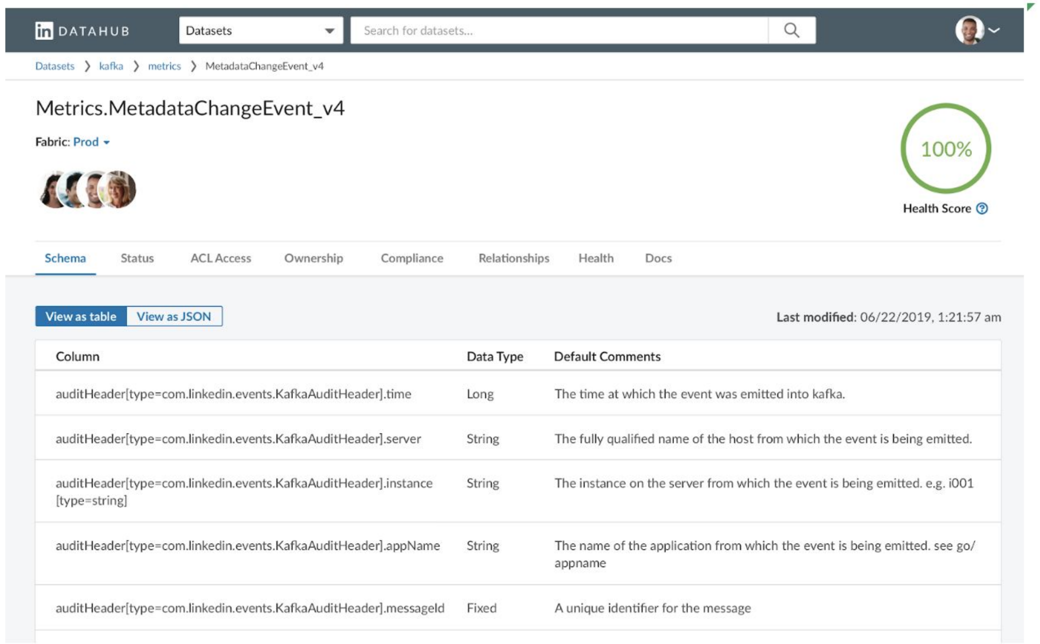
TRACKING.MetadataChangeEvent_v2

Data Origin	PROD
Platform	kafka

/data/service_column/metadatalineageevent_v1/final

Data Origin	PROD
Platform	hdfs

/data/service/MetadataLineageEvent_v1

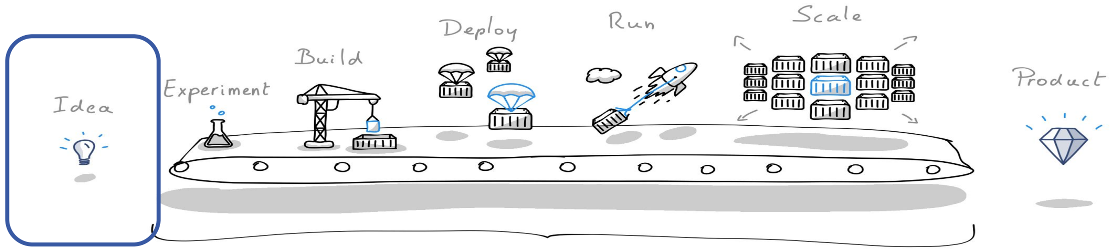


Metrics.MetadataChangeEvent_v4

Fabric: Prod

Last modified: 06/22/2019, 1:21:57 am

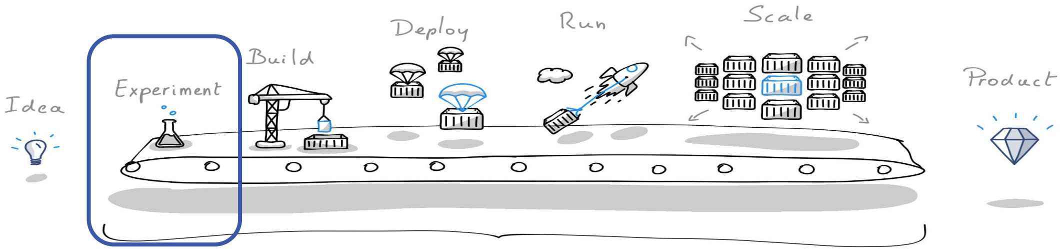
Column	Data Type	Default Comments
auditHeader[type=com.linkedin.events.KafkaAuditHeader].time	Long	The time at which the event was emitted into kafka.
auditHeader[type=com.linkedin.events.KafkaAuditHeader].server	String	The fully qualified name of the host from which the event is being emitted.
auditHeader[type=com.linkedin.events.KafkaAuditHeader].instance [type=string]	String	The instance on the server from which the event is being emitted. e.g. i001
auditHeader[type=com.linkedin.events.KafkaAuditHeader].appName	String	The name of the application from which the event is being emitted. see go/ appname
auditHeader[type=com.linkedin.events.KafkaAuditHeader].messageId	Fixed	A unique identifier for the message



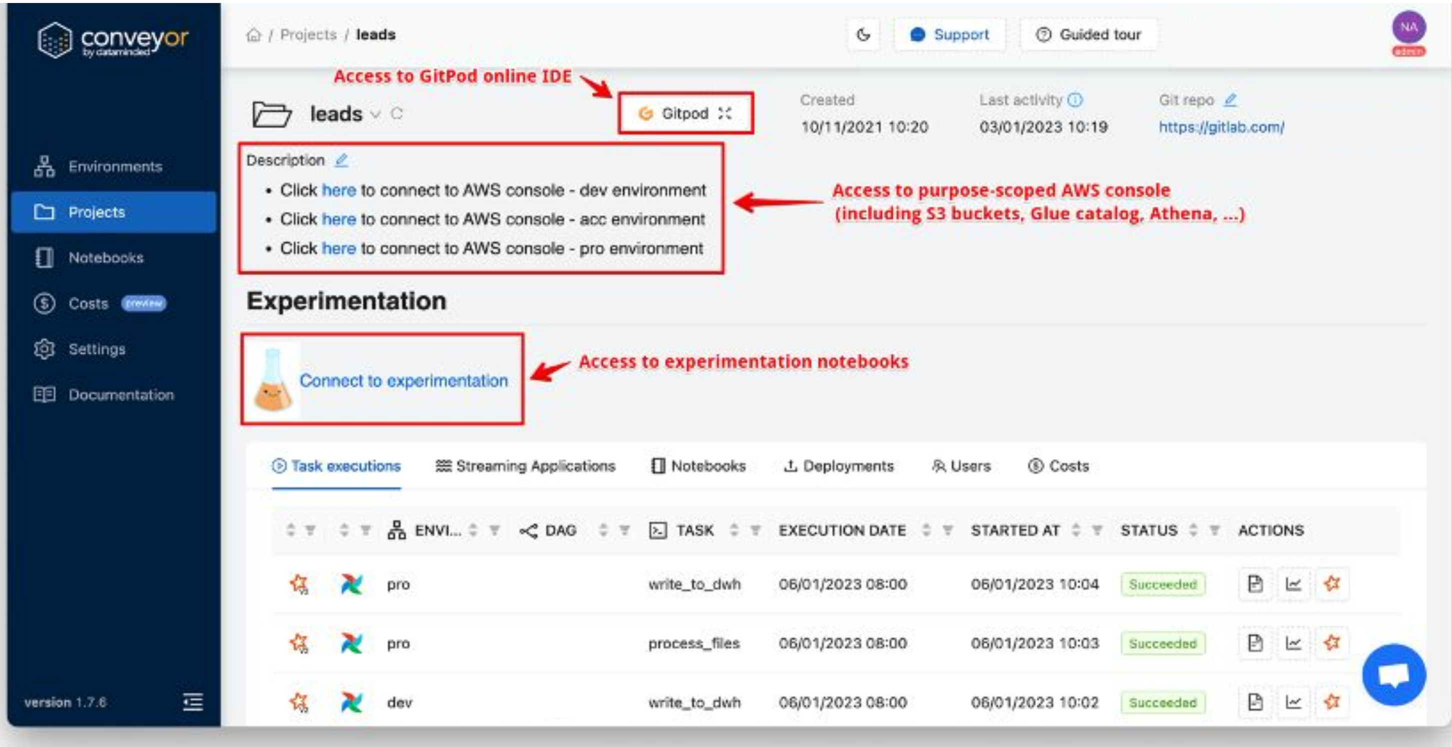
Register a new potential data product purpose & request data access.

servicenow®

Data owners validate the purpose and whether data can be used for that.
After approval everything is set up automatically.



Ideally offer an integrated user experience for data scientists/engineers/analysts



Access to GitPod online IDE

Created 10/11/2021 10:20

Last activity 03/01/2023 10:19

Git repo https://gitlab.com/

Description

- Click here to connect to AWS console - dev environment
- Click here to connect to AWS console - acc environment
- Click here to connect to AWS console - pro environment

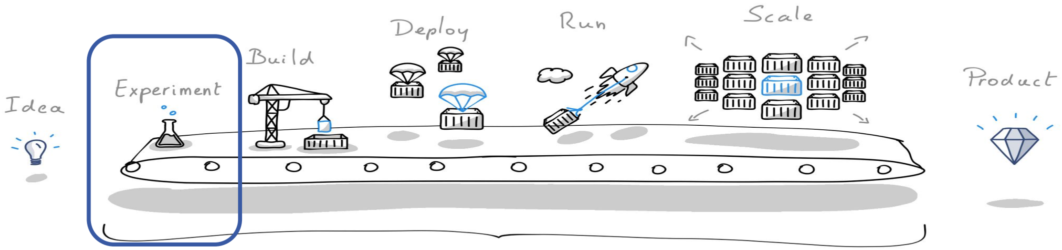
Access to purpose-scoped AWS console (including S3 buckets, Glue catalog, Athena, ...)

Experimentation

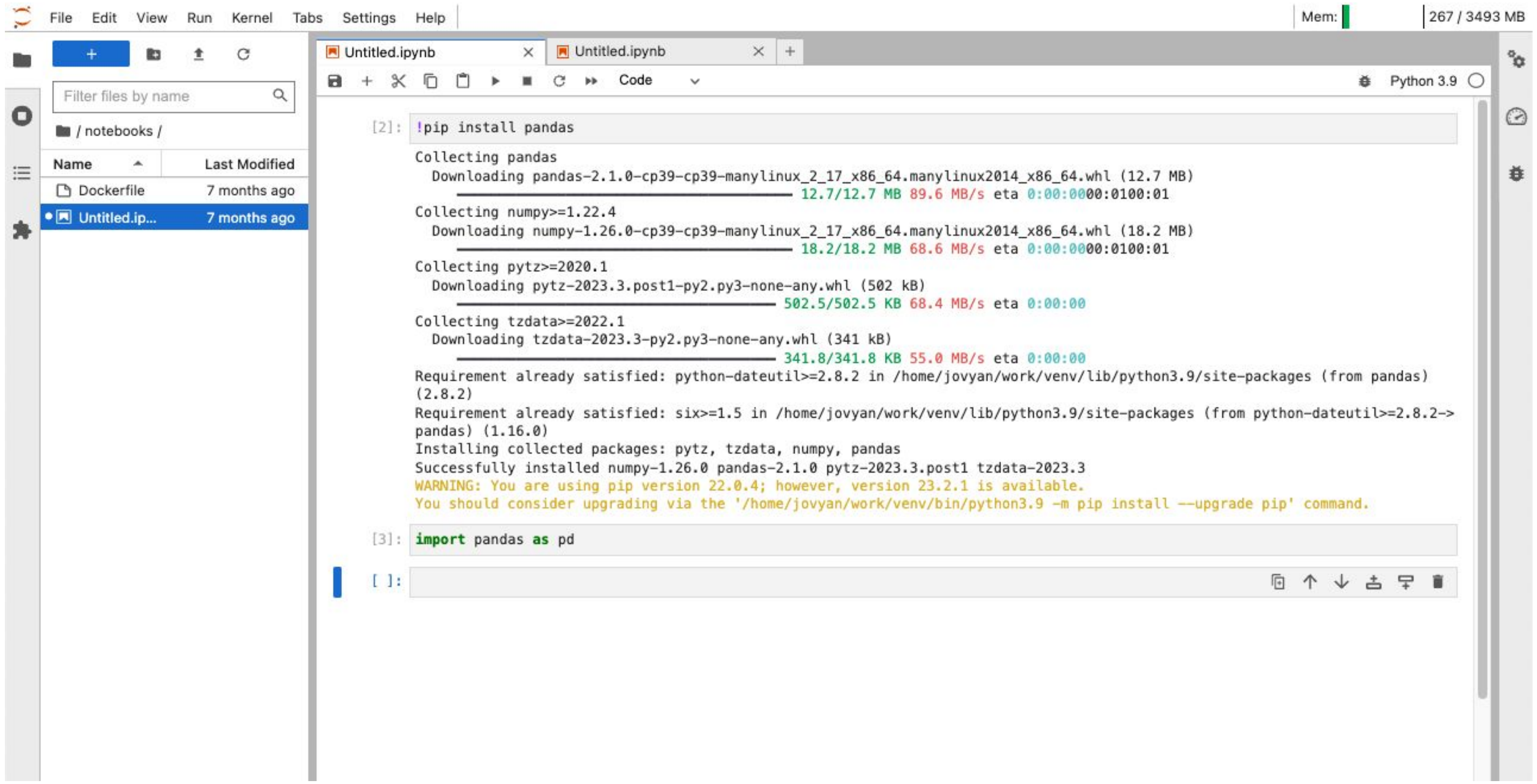
Connect to experimentation

Access to experimentation notebooks

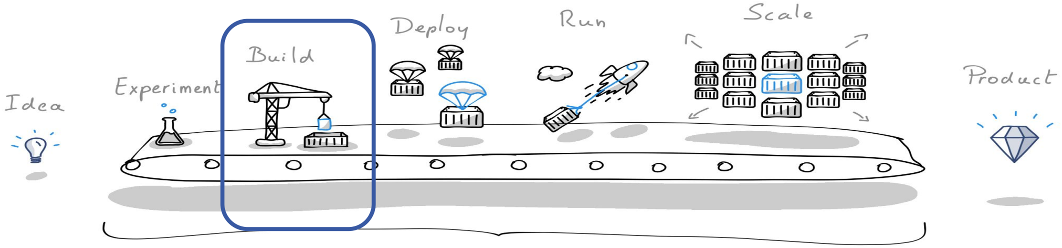
ENVI...	DAG	TASK	EXECUTION DATE	STARTED AT	STATUS	ACTIONS
pro		write_to_dwh	06/01/2023 08:00	06/01/2023 10:04	Succeeded	
pro		process_files	06/01/2023 08:00	06/01/2023 10:03	Succeeded	
dev		write_to_dwh	06/01/2023 08:00	06/01/2023 10:02	Succeeded	



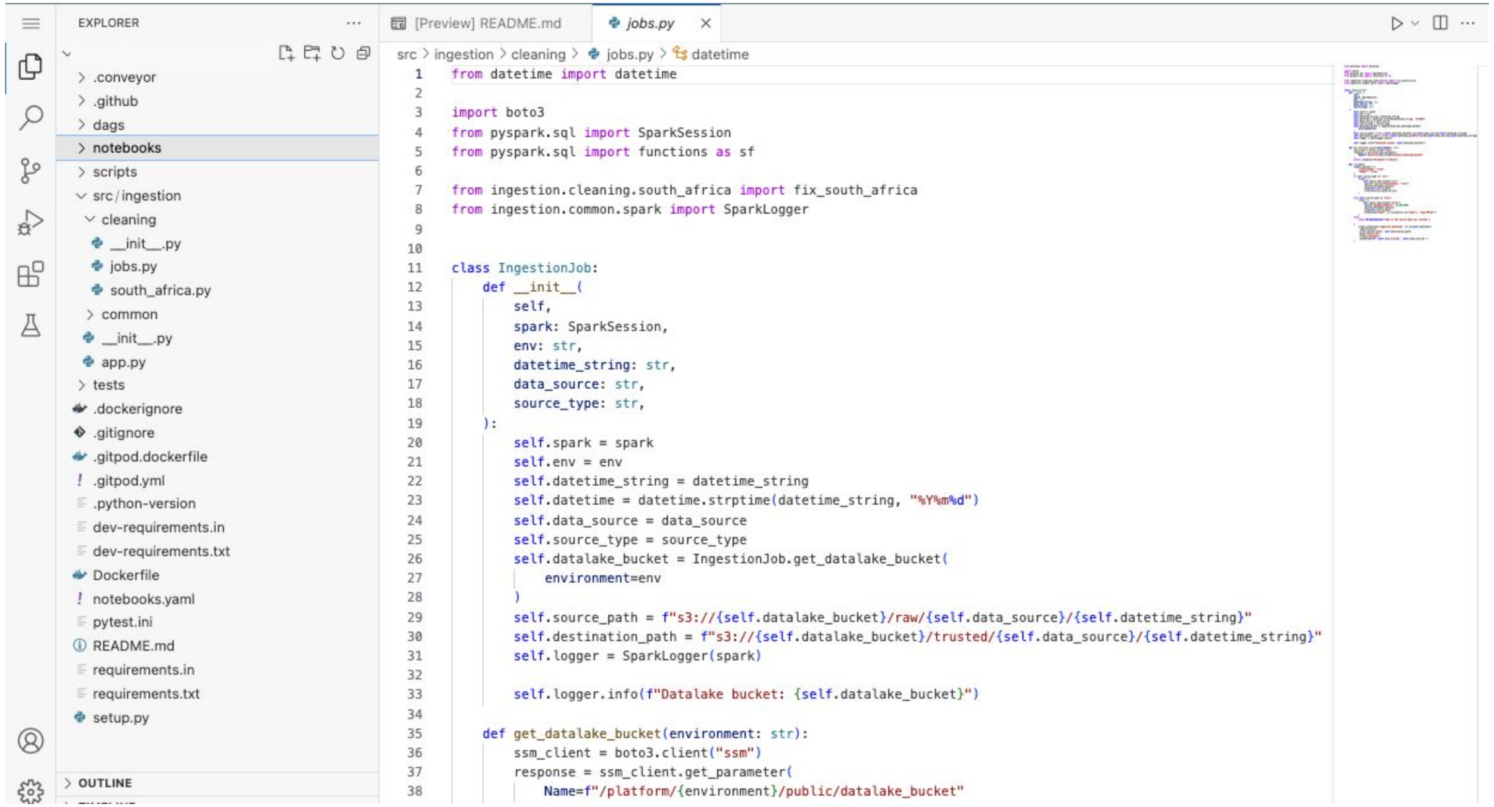
Experiment with data via one-click access to notebooks scoped to that project



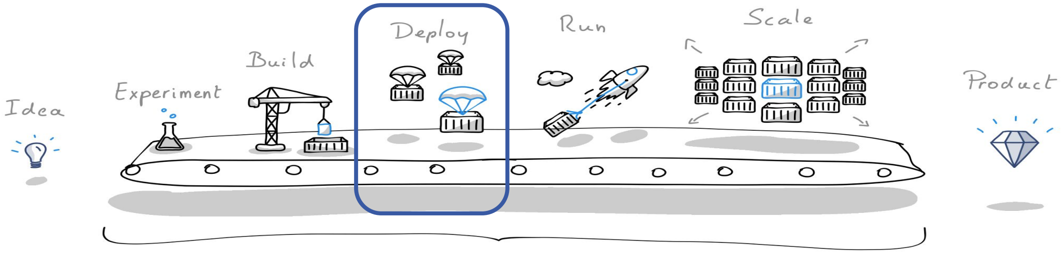
```
[2]: !pip install pandas
Collecting pandas
  Downloading pandas-2.1.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.7/12.7 MB 89.6 MB/s eta 0:00:0000:0100:01
Collecting numpy>=1.22.4
  Downloading numpy-1.26.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 18.2/18.2 MB 68.6 MB/s eta 0:00:0000:0100:01
Collecting pytz>=2020.1
  Downloading pytz-2023.3.post1-py2.py3-none-any.whl (502 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 502.5/502.5 KB 68.4 MB/s eta 0:00:00
Collecting tzdata>=2022.1
  Downloading tzdata-2023.3-py2.py3-none-any.whl (341 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 341.8/341.8 KB 55.0 MB/s eta 0:00:00
Requirement already satisfied: python-dateutil>=2.8.2 in /home/jovyan/work/venv/lib/python3.9/site-packages (from pandas)
(2.8.2)
Requirement already satisfied: six>=1.5 in /home/jovyan/work/venv/lib/python3.9/site-packages (from python-dateutil>=2.8.2->
pandas) (1.16.0)
Installing collected packages: pytz, tzdata, numpy, pandas
Successfully installed numpy-1.26.0 pandas-2.1.0 pytz-2023.3.post1 tzdata-2023.3
WARNING: You are using pip version 22.0.4; however, version 23.2.1 is available.
You should consider upgrading via the '/home/jovyan/work/venv/bin/python3.9 -m pip install --upgrade pip' command.

[3]: import pandas as pd

[ ]:
```



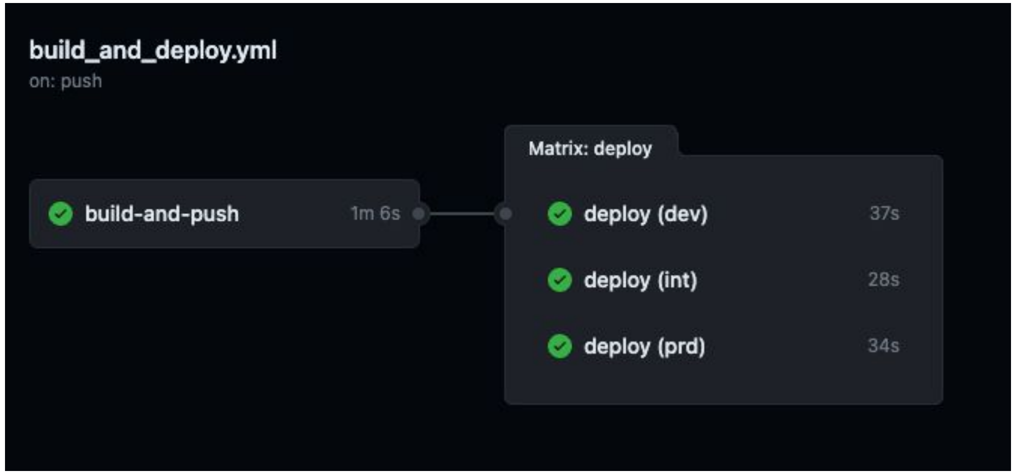
Industrialise via Cloud IDE’s that are scoped to that project



```python
from datetime import datetime

import boto3
from pyspark.sql import SparkSession
from pyspark.sql import functions as sf

from ingestion.cleaning.south_africa import fix_south_africa
from ingestion.common.spark import SparkLogger


class IngestionJob:
    def __init__(
        self,
        spark: SparkSession,
        env: str,
        datetime_string: str,
        data_source: str,
        source_type: str,
    ):
        self.spark = spark
        self.env = env
        self.datetime_string = datetime_string
        self.datetime = datetime.strptime(datetime_string, "%Y%m%d")
        self.data_source = data_source
        self.source_type = source_type
        self.datalake_bucket = IngestionJob.get_datalake_bucket(
            environment=env
        )
        self.source_path = f"s3://{self.datalake_bucket}/raw/{self.data_source}/{self.datetime_string}"
        self.destination_path = f"s3://{self.datalake_bucket}/trusted/{self.data_source}/{self.datetime_string}"
        self.logger = SparkLogger(spark)

        self.logger.info(f"Datalake bucket: {self.datalake_bucket}")

    def get_datalake_bucket(environment: str):
        ssm_client = boto3.client("ssm")
        response = ssm_client.get_parameter(
            Name=f"/platform/{environment}/public/datalake_bucket"
```



Build and deploy via standardized tooling and templates offered via paved roads



build_and_deploy.yml
on: push

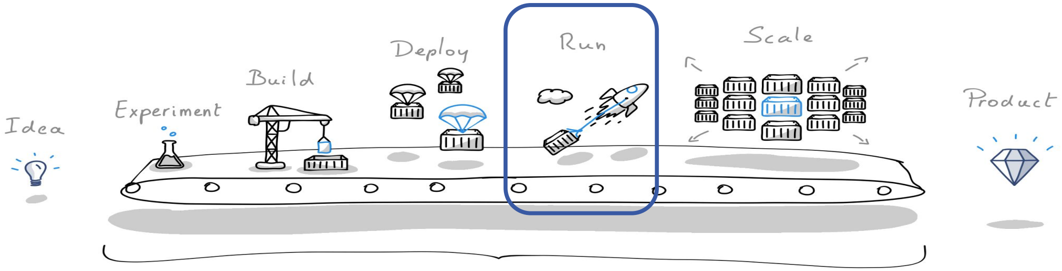
build-and-push 1m 6s

Matrix: deploy

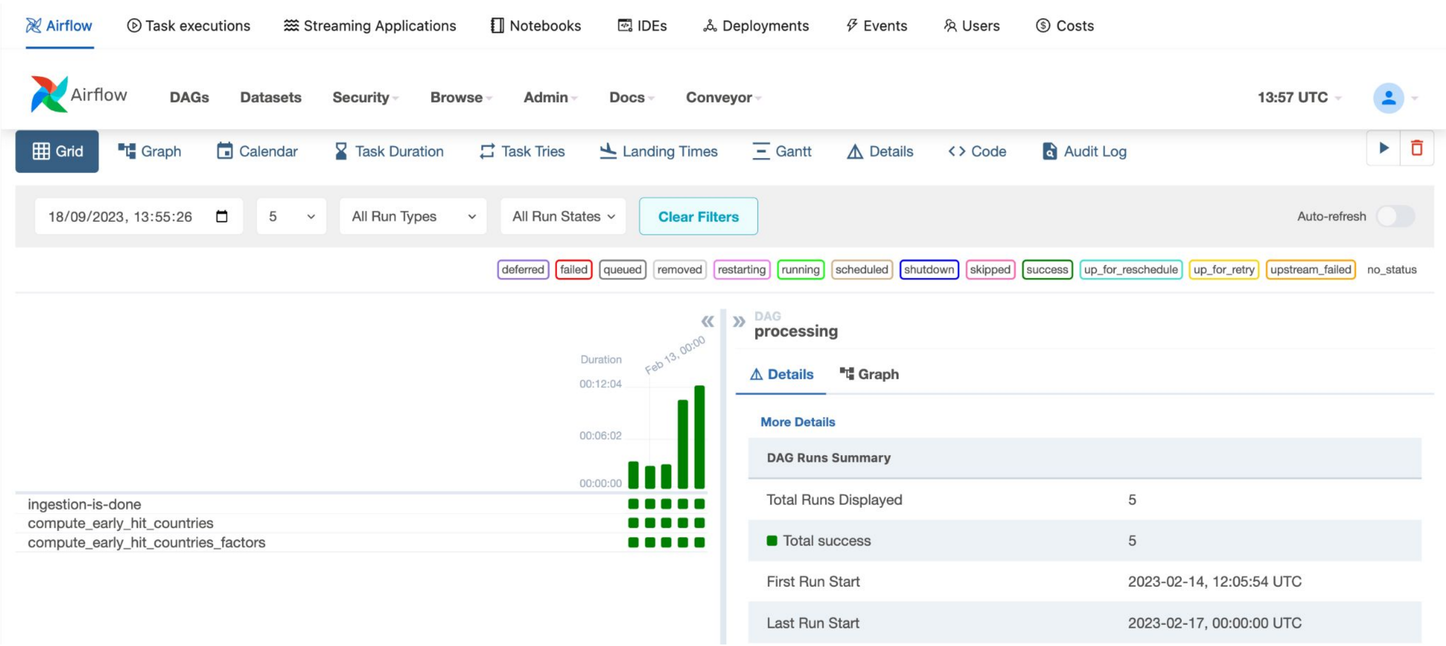
deploy (dev) 37s

deploy (int) 28s

deploy (prd) 34s



Schedule your data pipelines via Airflow or a different scheduler

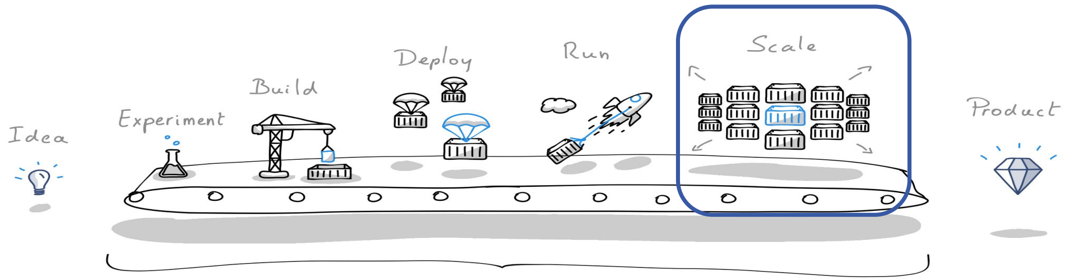


18/09/2023, 13:55:26 | 5 | All Run Types | All Run States

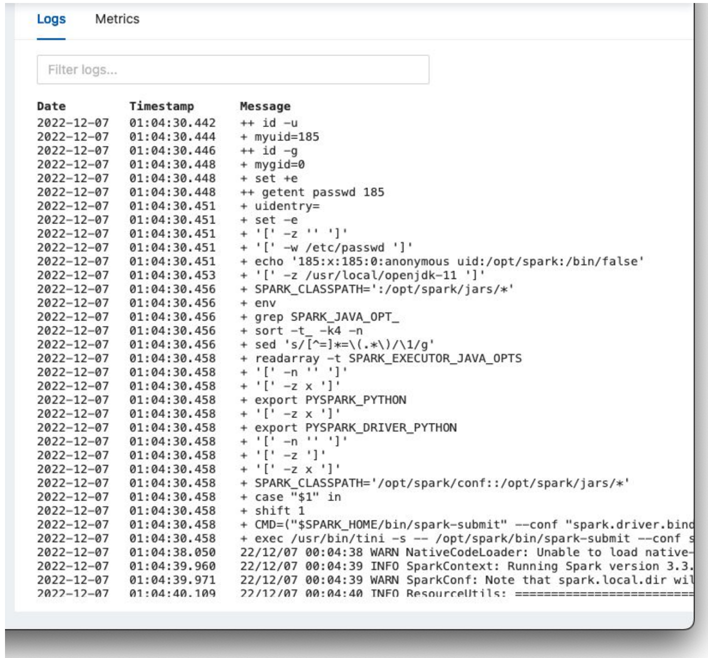
ingestion-is-done
compute_early_hit_countries
compute_early_hit_countries_factors

DAG
processing

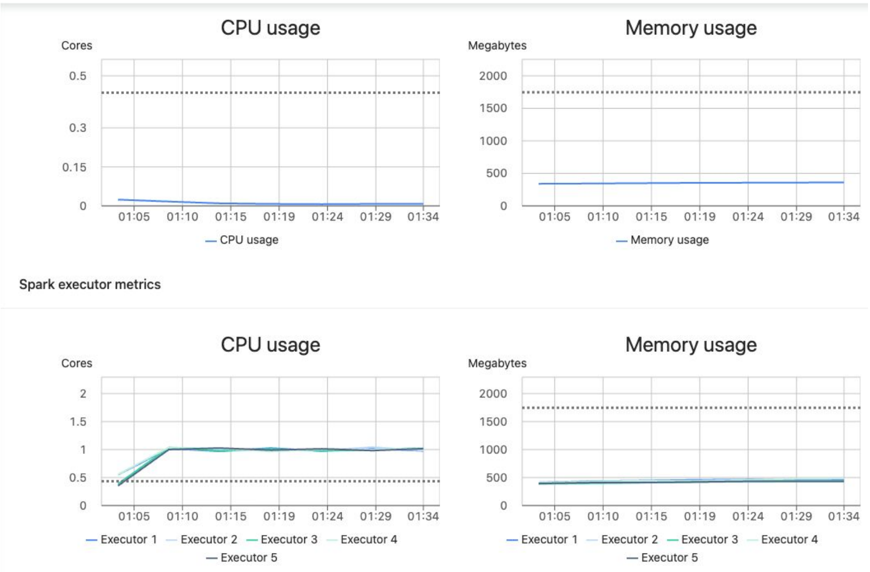
DAG Runs Summary	
Total Runs Displayed	5
🟩 Total success	5
First Run Start	2023-02-14, 12:05:54 UTC
Last Run Start	2023-02-17, 00:00:00 UTC



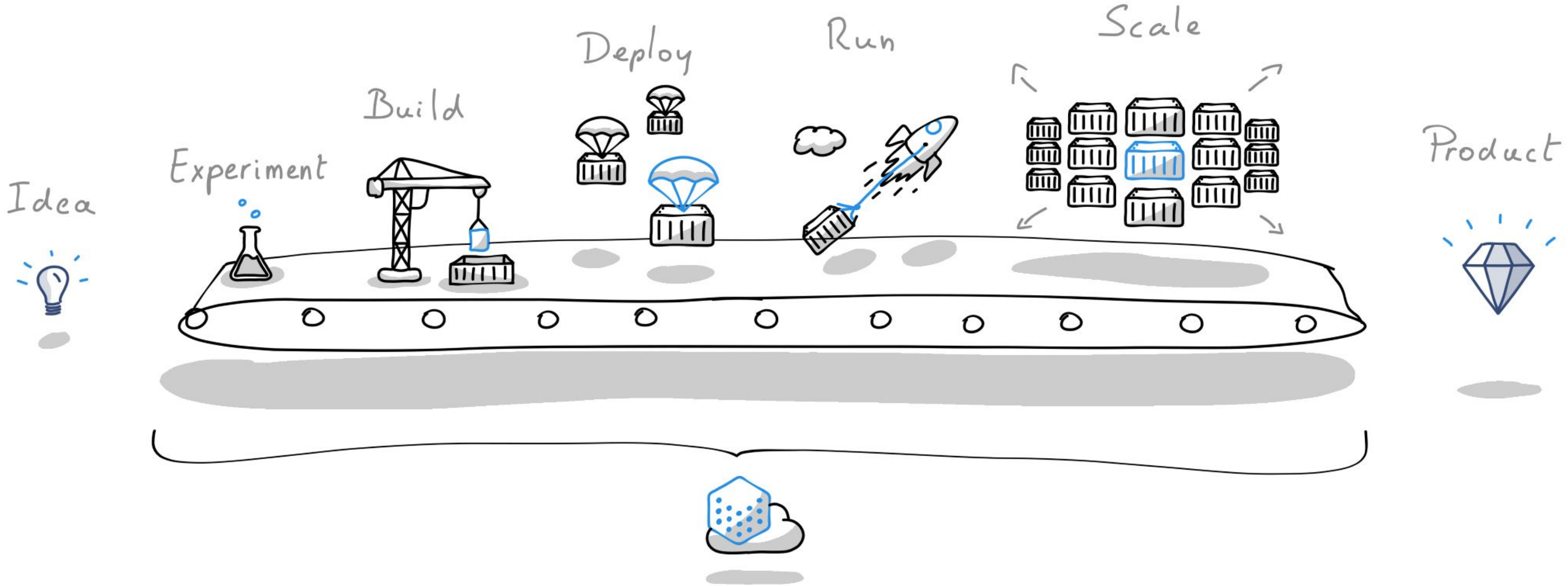
Follow up on alerts and offer easy access to logs and metrics if needed



```
Date        Timestamp      Message
2022-12-07  01:04:30.442   ++ id -u
2022-12-07  01:04:30.444   + myuid=185
2022-12-07  01:04:30.446   ++ id -g
2022-12-07  01:04:30.448   + mygid=0
2022-12-07  01:04:30.448   + set +e
2022-12-07  01:04:30.448   ++ getent passwd 185
2022-12-07  01:04:30.451   + uidentry=
2022-12-07  01:04:30.451   + set -e
2022-12-07  01:04:30.451   + '[' -z '' ']'
2022-12-07  01:04:30.451   + '[' -w /etc/passwd ']'
2022-12-07  01:04:30.451   + echo '185:x:185:0:anonymous uid:/opt/spark:/bin/false'
2022-12-07  01:04:30.453   + '[' -z /usr/local/openjdk-11 ']'
2022-12-07  01:04:30.456   + SPARK_CLASSPATH=':/opt/spark/jars/*'
2022-12-07  01:04:30.456   + env
2022-12-07  01:04:30.456   + grep SPARK_JAVA_OPT_
2022-12-07  01:04:30.456   + sort -t_ -k4 -n
2022-12-07  01:04:30.456   + sed 's/[^=]*=\(.*\)/\1/g'
2022-12-07  01:04:30.458   + readarray -t SPARK_EXECUTOR_JAVA_OPTS
2022-12-07  01:04:30.458   + '[' -n '' ']'
2022-12-07  01:04:30.458   + '[' -z x ']'
2022-12-07  01:04:30.458   + export PYSPARK_PYTHON
2022-12-07  01:04:30.458   + '[' -z x ']'
2022-12-07  01:04:30.458   + export PYSPARK_DRIVER_PYTHON
2022-12-07  01:04:30.458   + '[' -n '' ']'
2022-12-07  01:04:30.458   + '[' -z ']'
2022-12-07  01:04:30.458   + '[' -z x ']'
2022-12-07  01:04:30.458   + SPARK_CLASSPATH='/opt/spark/conf::/opt/spark/jars/*'
2022-12-07  01:04:30.458   + case "$1" in
2022-12-07  01:04:30.458   + shift 1
2022-12-07  01:04:30.458   + CMD=("$SPARK_HOME/bin/spark-submit" --conf "spark.driver.bind
2022-12-07  01:04:30.458   + exec /usr/bin/tini -s -- /opt/spark/bin/spark-submit --conf s
2022-12-07  01:04:38.050   22/12/07 00:04:38 WARN NativeCodeLoader: Unable to load native-
2022-12-07  01:04:39.960   22/12/07 00:04:39 INFO SparkContext: Running Spark version 3.3.
2022-12-07  01:04:39.971   22/12/07 00:04:39 WARN SparkConf: Note that spark.local.dir wil
2022-12-07  01:04:40.109   22/12/07 00:04:40 INFO ResourceUtils: =========================
```



It is possible to build this yourselves, or give yourselves a head start



conveyor

Practical approaches about how to make your data platform a success

1. **Tie your platform to the strategic goals of the company**
   Look for reasons to do things instead of finding reasons not to do it

2. **Embrace a platform thinking approach**
   The platform mindset and a roadmap on how to get there

3. **How do you get started?**
   Practical tips on how to approach this

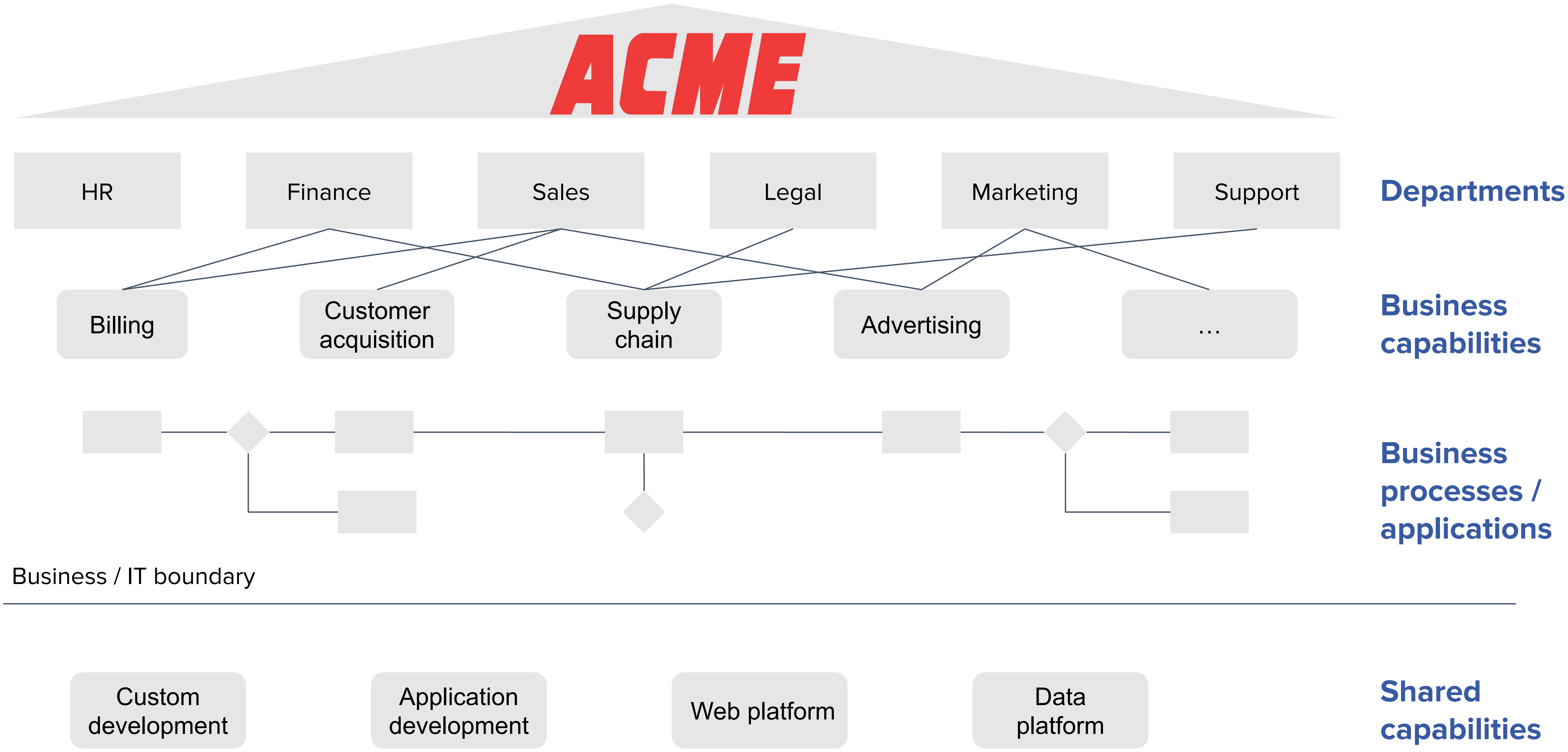
4. **Balancing the act of centralization and decentralization**
   Values of Freedom and Control are not conflicting

👋

The end

Backup slides

Make sure you are building capabilities the business needs



ACME

HR

Finance

Sales

Legal

Marketing

Support

Departments

Billing

Customer acquisition

Supply chain

Advertising

…

Business capabilities

Business processes / applications

Business / IT boundary

Custom development

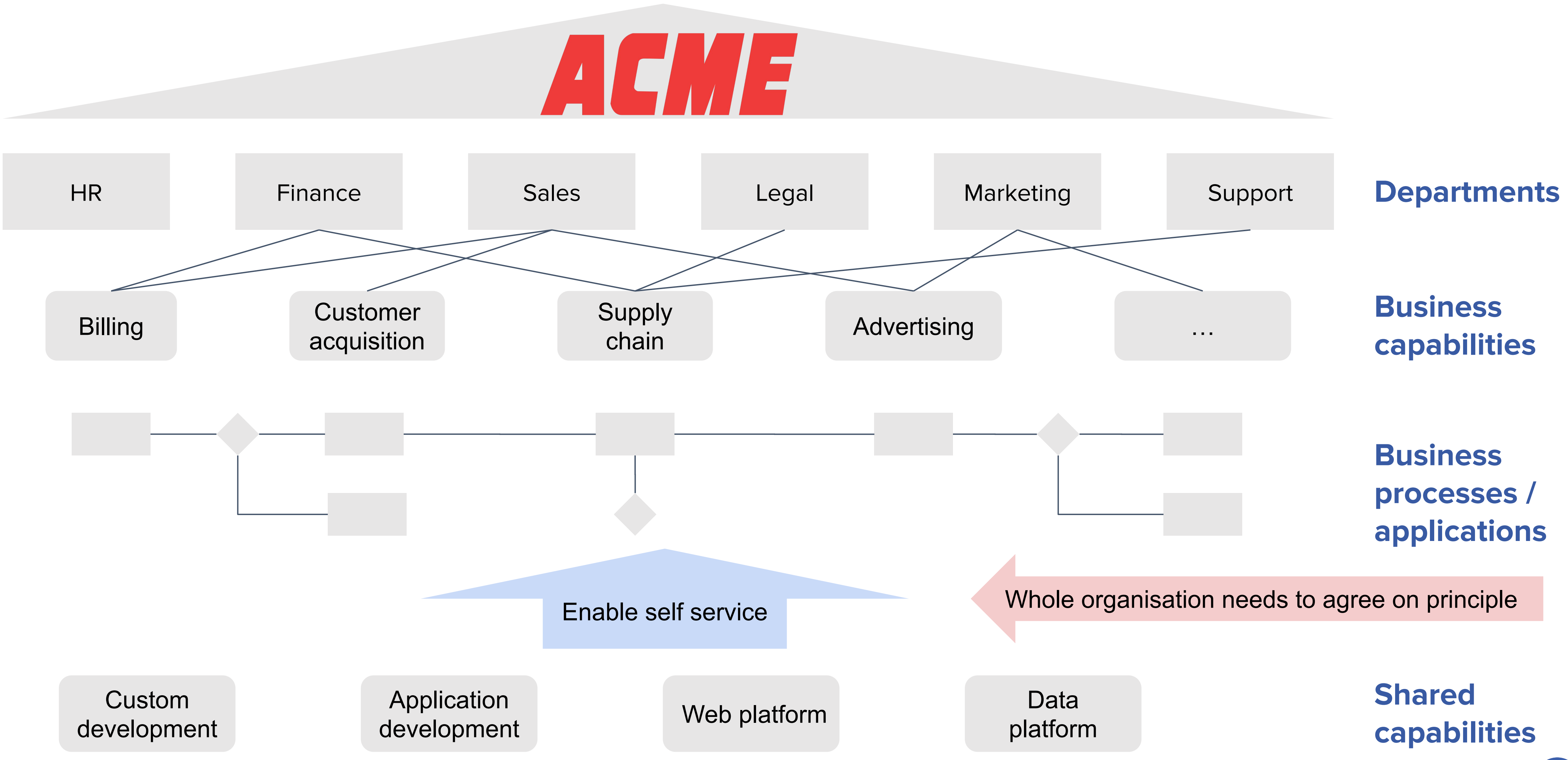
Application development

Web platform

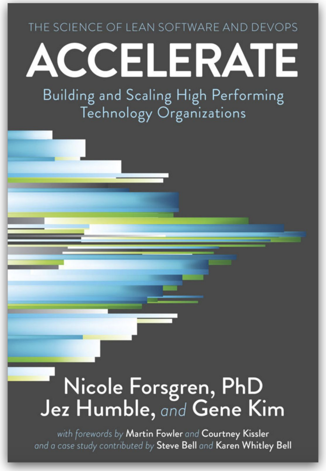
Data platform

Shared capabilities

Make sure you are building capabilities the business needs



“we must ensure delivery teams are cross-functional, with all the skills necessary to design, develop, test, deploy, and operate the system on the same team.”



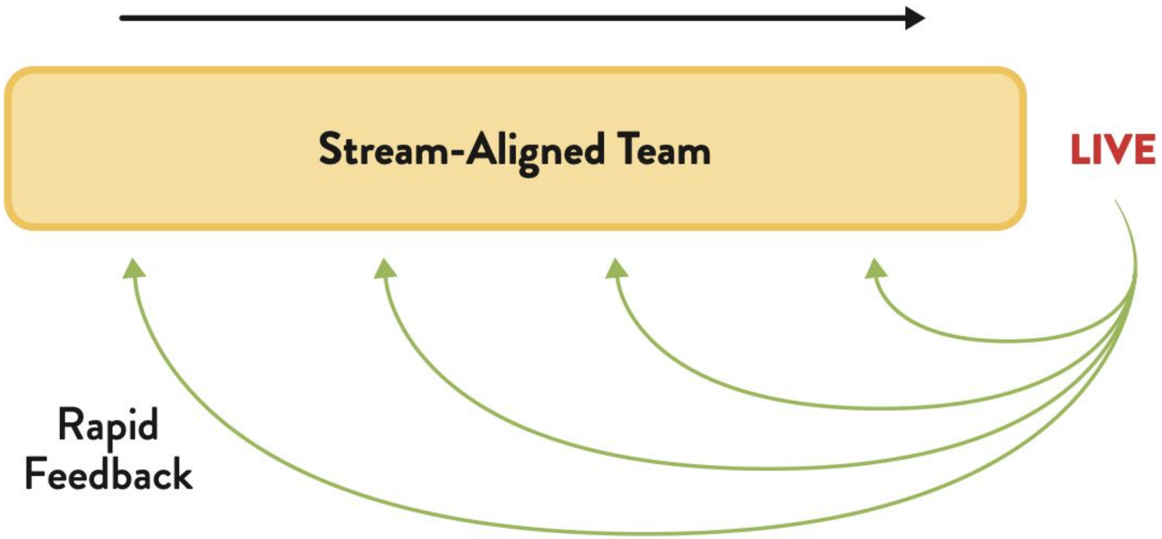
THE SCIENCE OF LEAN SOFTWARE AND DEVOPS

ACCELERATE

Building and Scaling High Performing Technology Organizations

Nicole Forsgren, PhD
Jez Humble, *and* Gene Kim

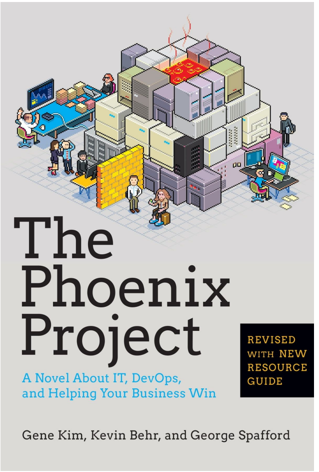
with forewords by Martin Fowler *and* Courtney Kissler
and a case study contributed by Steve Bell *and* Karen Whitley Bell



Stream-Aligned Team

LIVE

Rapid Feedback



The Phoenix Project

A Novel About IT, DevOps, and Helping Your Business Win

REVISED WITH NEW RESOURCE GUIDE

Gene Kim, Kevin Behr, and George Spafford

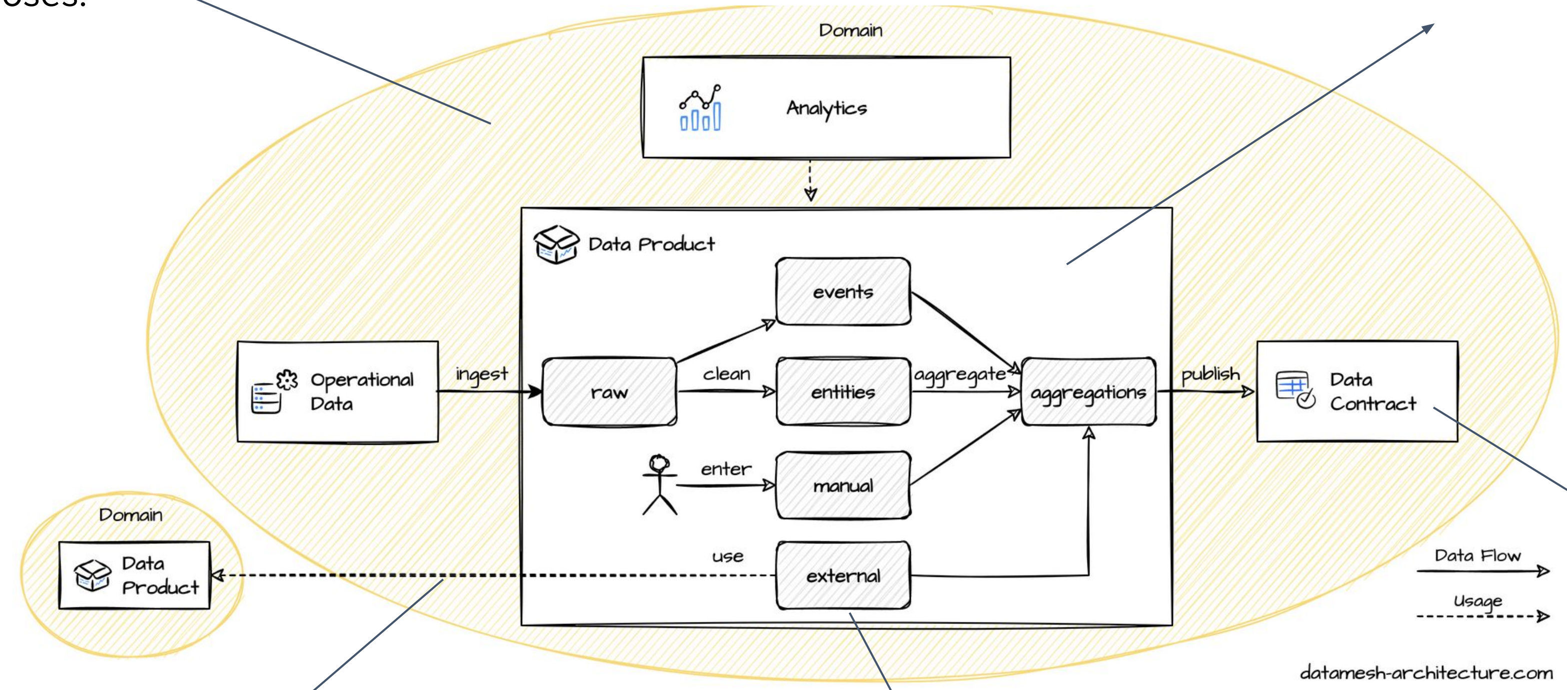
Pro tip: own the perception of how teams look at you as a platform team

Support your users by defining your commitments, support channels and support workflow for each type of request

Data Mesh is not an excuse to decentralise everything (AWS focused)

❌🔫🦶
Domain is not the equivalent of AWS account, especially for analytical purposes!

🚗
Capabilities to build this can be offered by the data platform team via paved roads.



❌🔫🦶
Please don't use dedicated API's for all X-domain interactions! S3/Glue, Athena/DWH SQL,... are also API's.

🤝
Data can be stored in shared data stores (S3/DWH's). This makes it easier to share data across data products/domains and to standardise data access patterns

🤷
There is no standardized way to share data contracts