Fabric Automation

From Setup to CI/CD

Fabric Explorers User Group Norway, November 18th 2024

twoday

Fabric Automation

Fabric Explorers User Group Norway,
November 18th 2024

1. **Introduction to Fabric automation and the Fabric REST APIs**

2. **Designing our Fabric Data Lakehouse**
   - Components and architecture
   - Lakehouse layers
   - Fabric environments and stages

3. **Kickstart and uniform the Fabric data platform setup**

4. **Ways of working utilizing a metadata-driven framework**
   - Best practices
   - Feature development

5. **CI/CD – Deploying a Fabric solution with Azure DevOps**
   - Deployment approaches
   - Utilizing Azure DevOps CI/CD Pipelines
   - Current limitations, workarounds etc.

6. **Q&A**

twoday

Peer Grønnerup

Principal architect & Consulting manager, twoday, Data & AI

in https://www.linkedin.com/in/peergroennerup/

peer.gronnerup@twoday.com

https://peerinsights.hashnode.dev/

twoday

The 3 phases cover in this session

Design

Design of the architecture

Definition of storage layers, compute resources and governance

Environments and stages

Utilizing a metadata-driven framework

Git setup and repository structure

Build

Setup code base and environments including workspaces, lakehouse etc.

Implement features covering data ingestion, preparation and serving

Automate deployment with CI/CD

Documentation

Run

Operationalization of the platform

Collaborate and enhance the solution

Automating Fabric with Fabric REST APIs

Microsoft Fabric REST APIs

Build

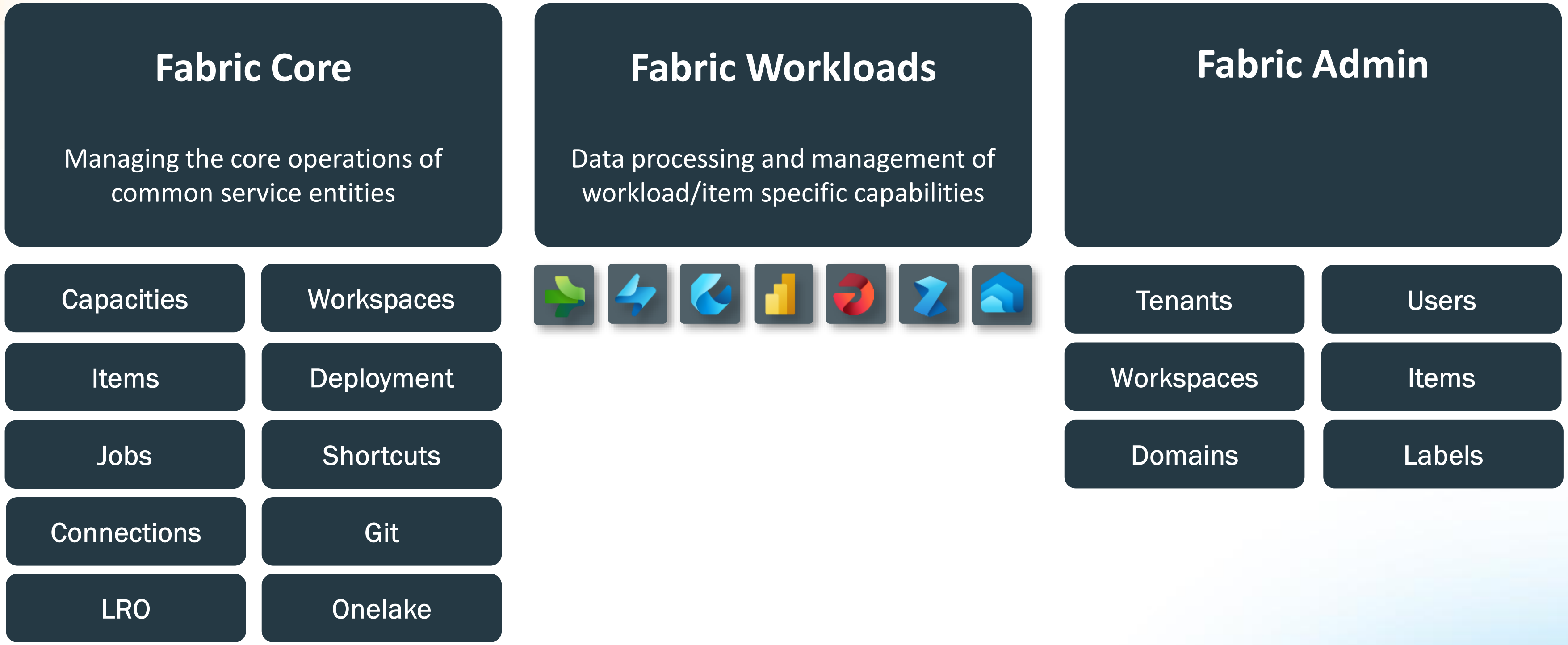
- Setup Fabric infrastructure
- Workspace management
- Item management
- Governance and security
- Automate documentation
- And much more...

Run

- Support our ways-of-working
- Setup development env.
- Deploy items
- Monitoring and alerting
- And much more...

twoday

Automating Fabric with Fabric REST APIs



Fabric Core

Managing the core operations of common service entities

- Capacities
- Workspaces
- Items
- Deployment
- Jobs
- Shortcuts
- Connections
- Git
- LRO
- Onelake

Fabric Workloads

Data processing and management of workload/item specific capabilities

Fabric Admin

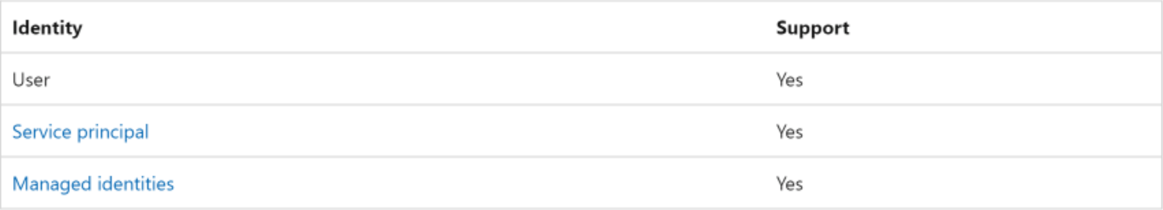
- Tenants
- Users
- Workspaces
- Items
- Domains
- Labels

→ https://learn.microsoft.com/en-us/rest/api/fabric/articles/

Fabric REST APIs and Service Principals

We can now use service principal for the following APIs:

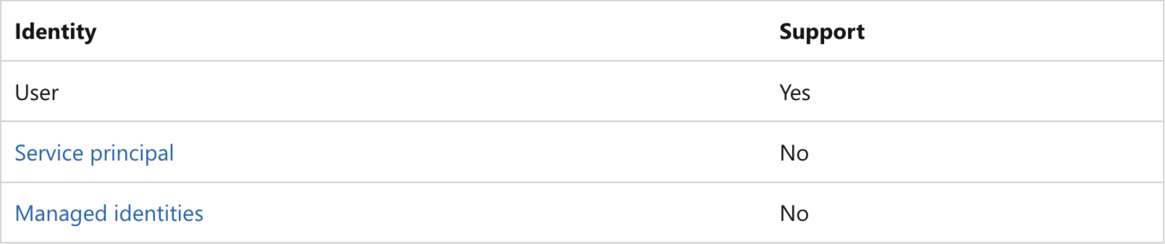
- Core: Workspaces, Capacities, Connections etc.
- Power BI: Reports & semantic models
- Data Engineering: Lakehouses, Notebook etc.
- RTI: Eventhouse, Eventstream, KQL Querysets etc.



Identity	Support
User	Yes
Service principal	Yes
Managed identities	Yes

But… There is still no support for service principal for:

- Git integration
- Items like Data Pipelines, Warehouse, ML Models & Experiments etc.



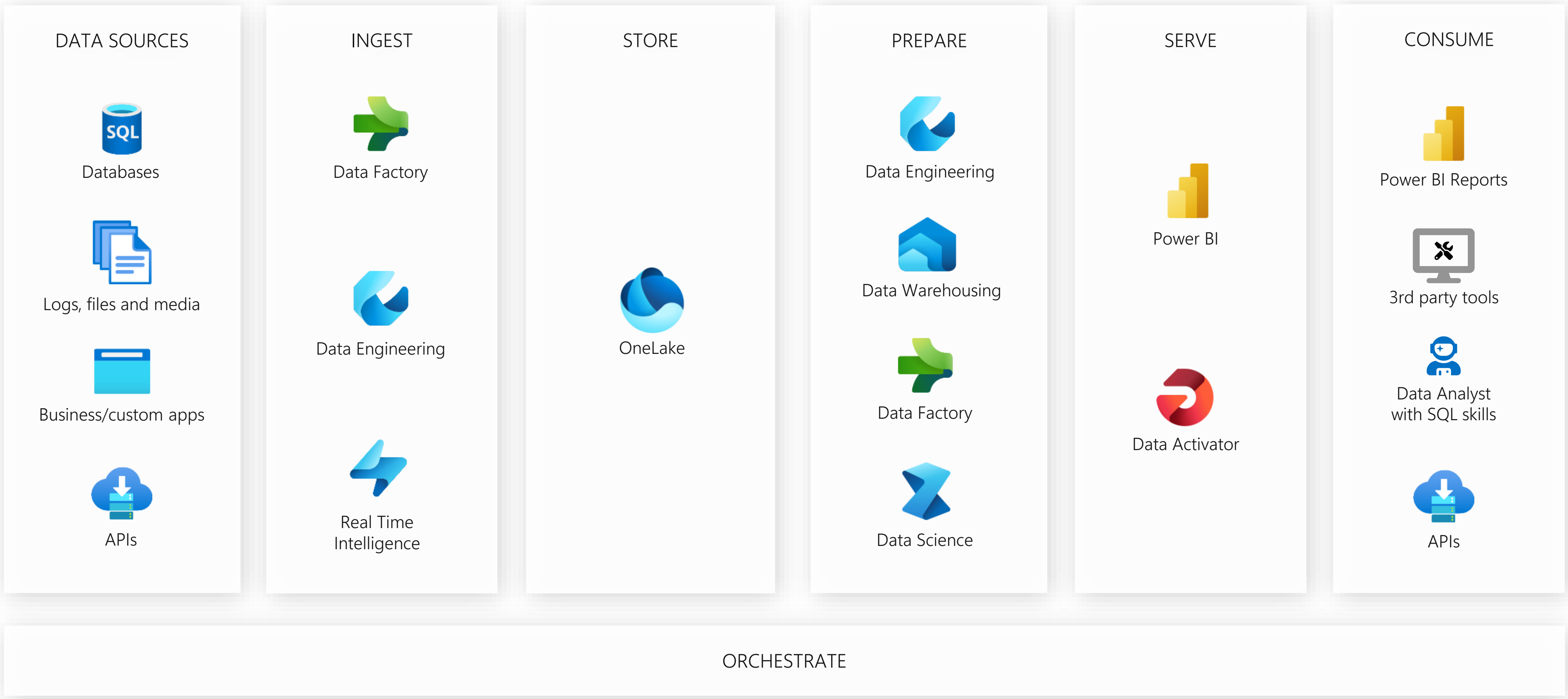
Identity	Support
User	Yes
Service principal	No
Managed identities	No

Visit my blog post **"Automating Microsoft Fabric: Extracting Identity Support data"** on https://peerinsights.hashnode.dev

twoday

Designing our Fabric Data Lakehouse

Multiple components = multiple possibilities



DATA SOURCES

- Databases
- Logs, files and media
- Business/custom apps
- APIs

INGEST

- Data Factory
- Data Engineering
- Real Time Intelligence

STORE

- OneLake

PREPARE

- Data Engineering
- Data Warehousing
- Data Factory
- Data Science

SERVE

- Power BI
- Data Activator

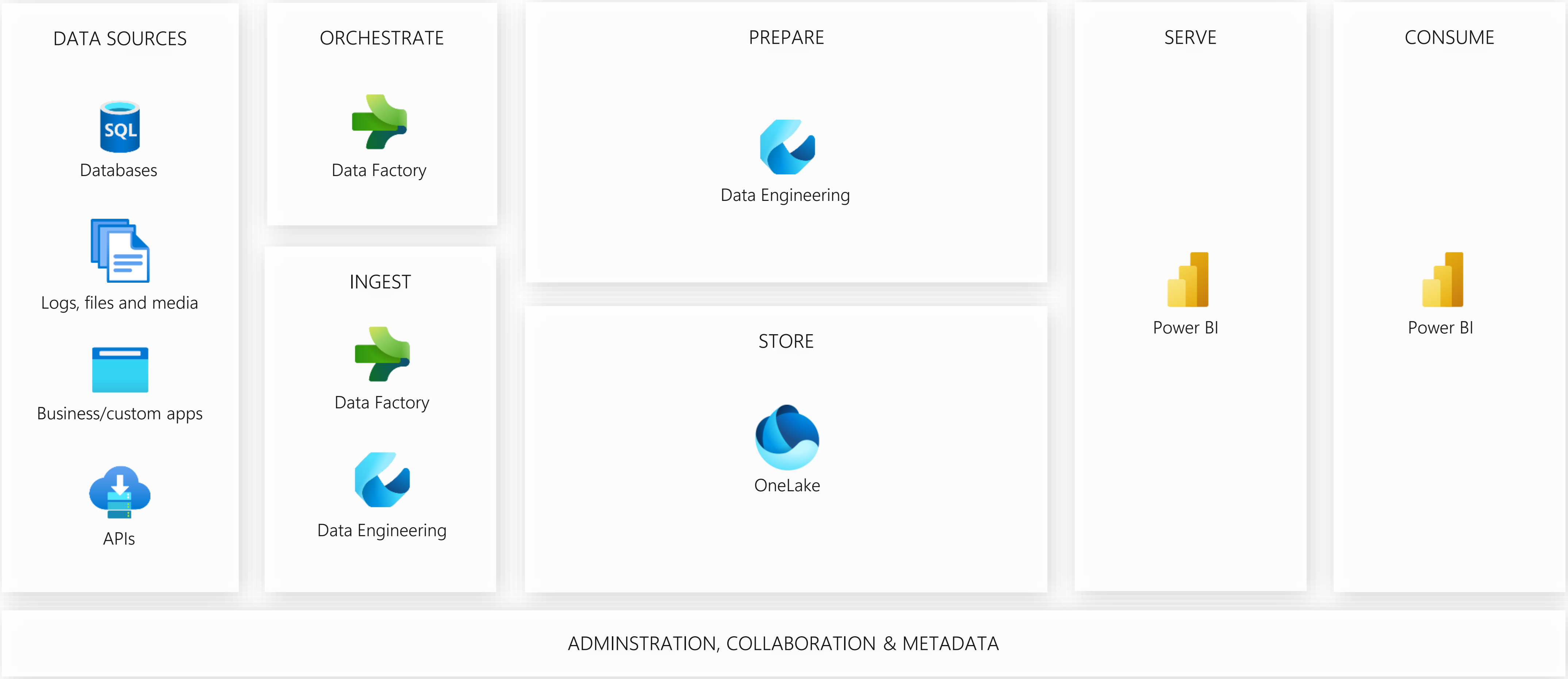
CONSUME

- Power BI Reports
- 3rd party tools
- Data Analyst with SQL skills
- APIs

ORCHESTRATE

twoday

The starting point architecture



DATA SOURCES

Databases

Logs, files and media

Business/custom apps

APIs

ORCHESTRATE

Data Factory

INGEST

Data Factory

Data Engineering

PREPARE

Data Engineering

STORE

OneLake

SERVE

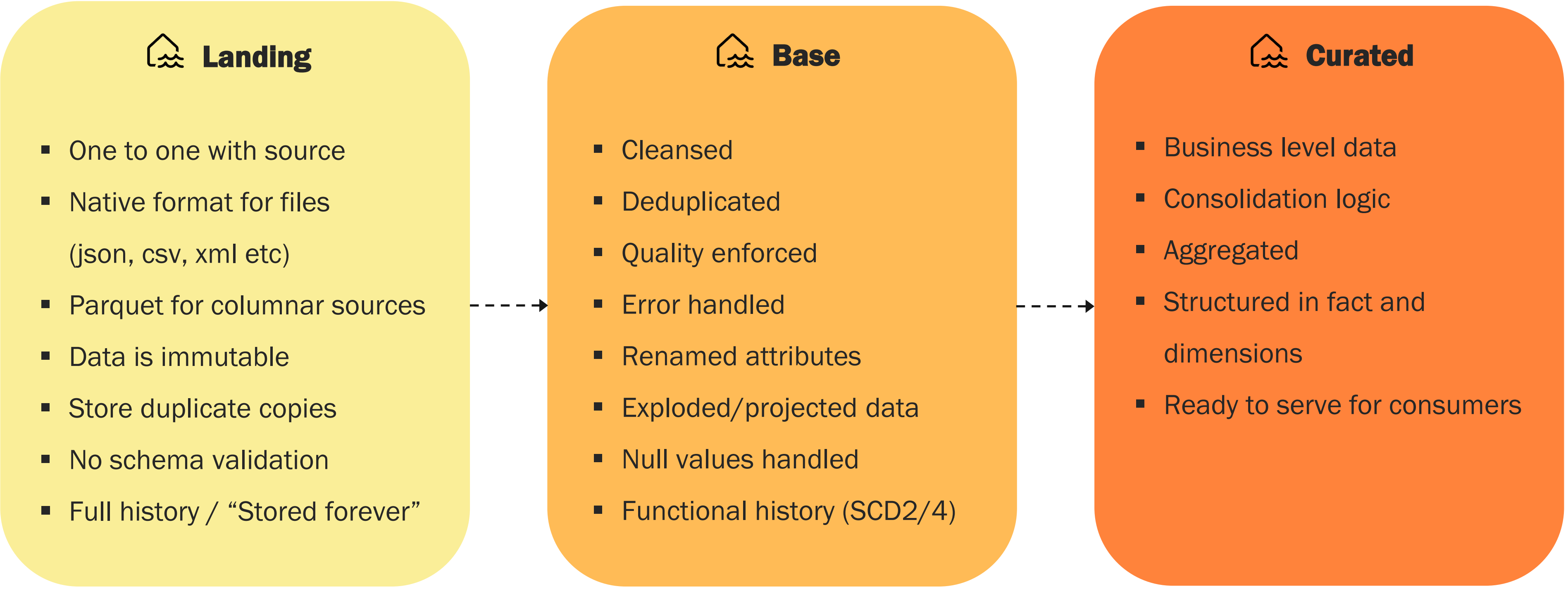
Power BI

CONSUME

Power BI

ADMINSTRATION, COLLABORATION & METADATA

The starting point architecture – Lakehouse layers



Landing

- One to one with source
- Native format for files (json, csv, xml etc)
- Parquet for columnar sources
- Data is immutable
- Store duplicate copies
- No schema validation
- Full history / “Stored forever”

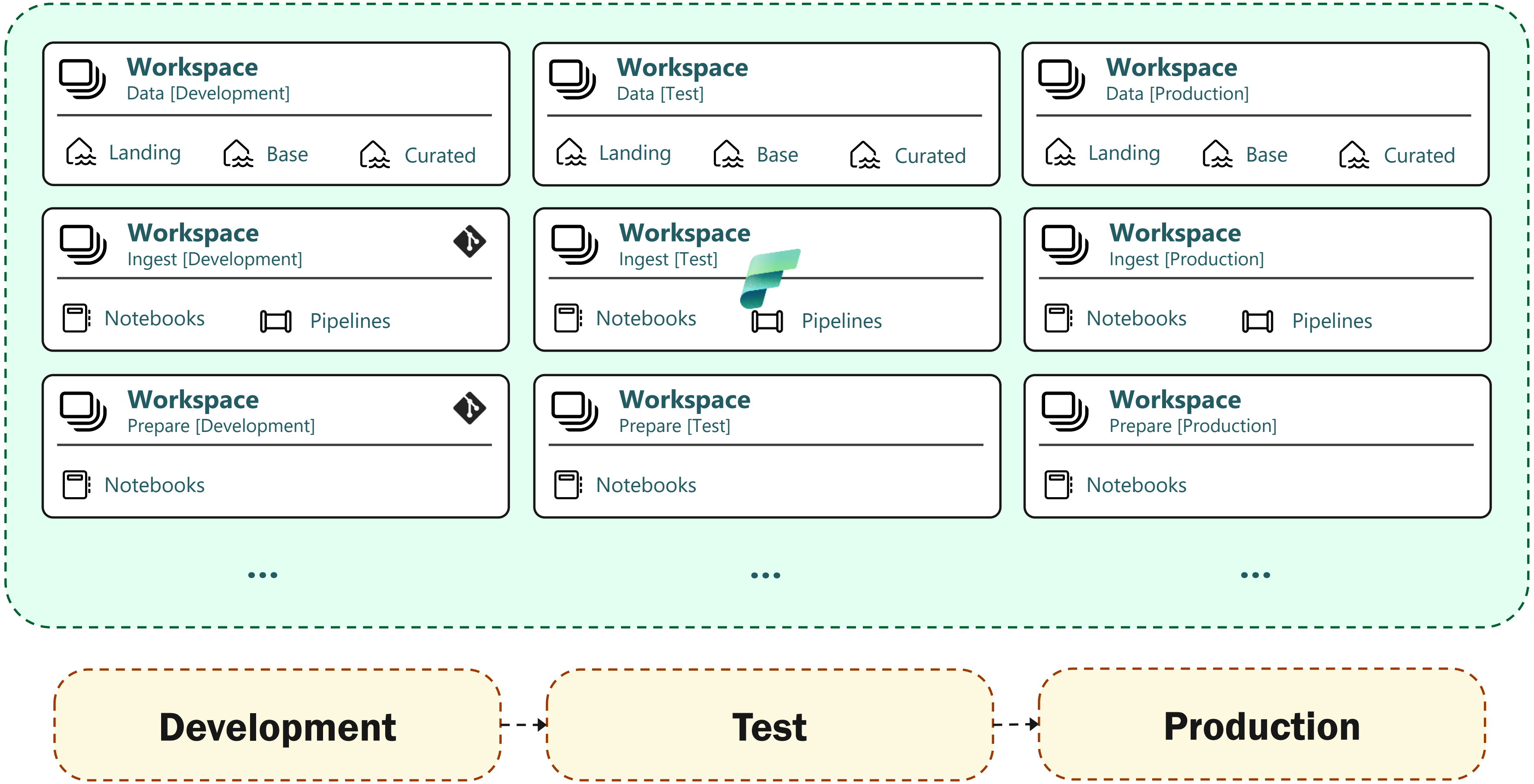
Base

- Cleansed
- Deduplicated
- Quality enforced
- Error handled
- Renamed attributes
- Exploded/projected data
- Null values handled
- Functional history (SCD2/4)

Curated

- Business level data
- Consolidation logic
- Aggregated
- Structured in fact and dimensions
- Ready to serve for consumers

Environments, stages and Git structure



Workspace Data [Development]

Landing Base Curated

Workspace Data [Test]

Landing Base Curated

Workspace Data [Production]

Landing Base Curated

Workspace Ingest [Development]

Notebooks Pipelines

Workspace Ingest [Test]

Notebooks Pipelines

Workspace Ingest [Production]

Notebooks Pipelines

Workspace Prepare [Development]

Notebooks

Workspace Prepare [Test]

Notebooks

Workspace Prepare [Production]

Notebooks

... ...

Development → **Test** → **Production**

Git structure

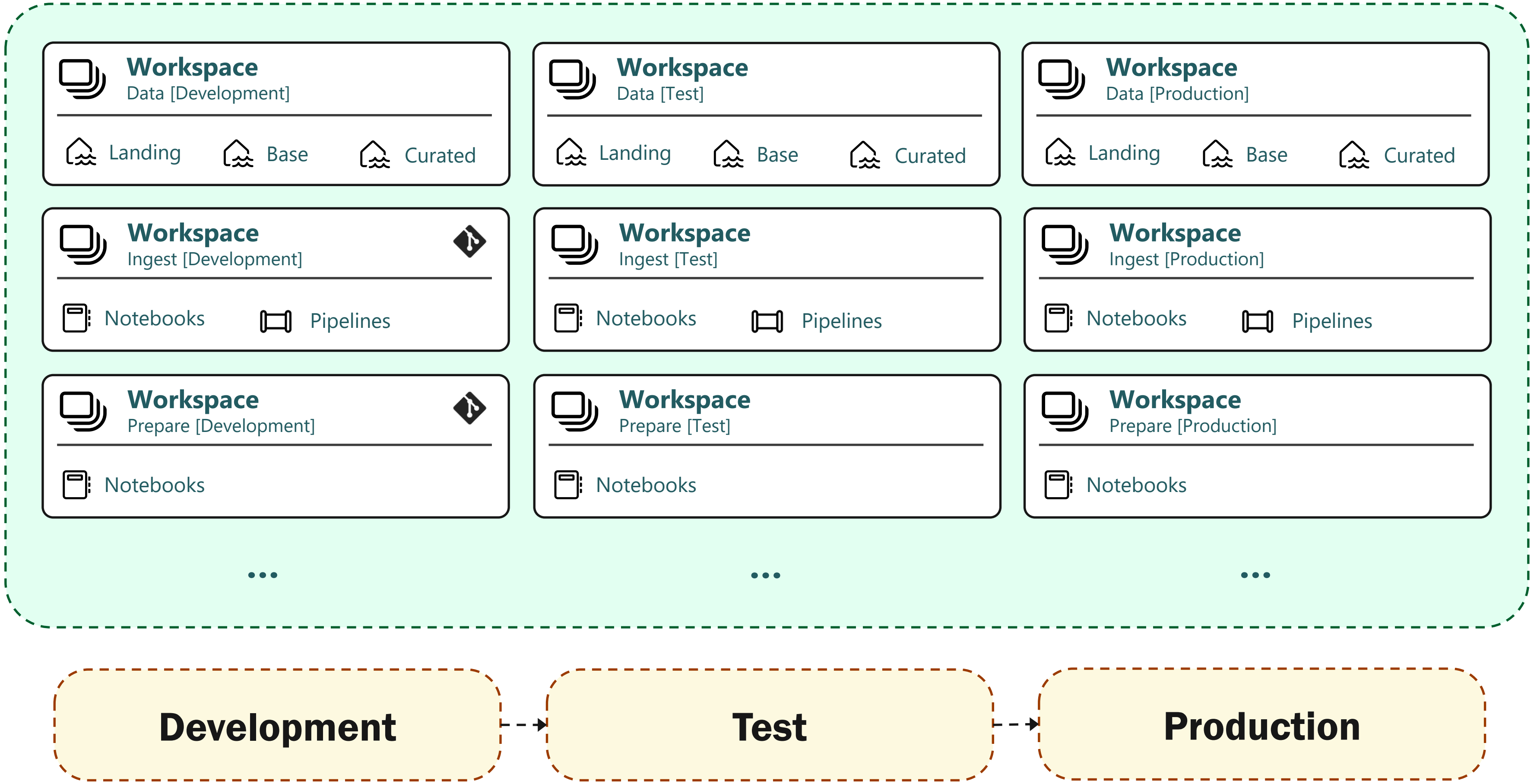
Single repository approch:

- Automation
- Documentation
- Solution
  - Ingest
  - Prepare
  - Serve
  - Cons
  - ...
- ...

twoday

Kickstart and uniform the Fabric data platform setup

Environments, stages and Git structure



Workspace
Data [Development]
Landing · Base · Curated

Workspace
Data [Test]
Landing · Base · Curated

Workspace
Data [Production]
Landing · Base · Curated

Workspace
Ingest [Development]
Notebooks · Pipelines

Workspace
Ingest [Test]
Notebooks · Pipelines

Workspace
Ingest [Production]
Notebooks · Pipelines

Workspace
Prepare [Development]
Notebooks

Workspace
Prepare [Test]
Notebooks

Workspace
Prepare [Production]
Notebooks

...

...

...

Development → **Test** → **Production**

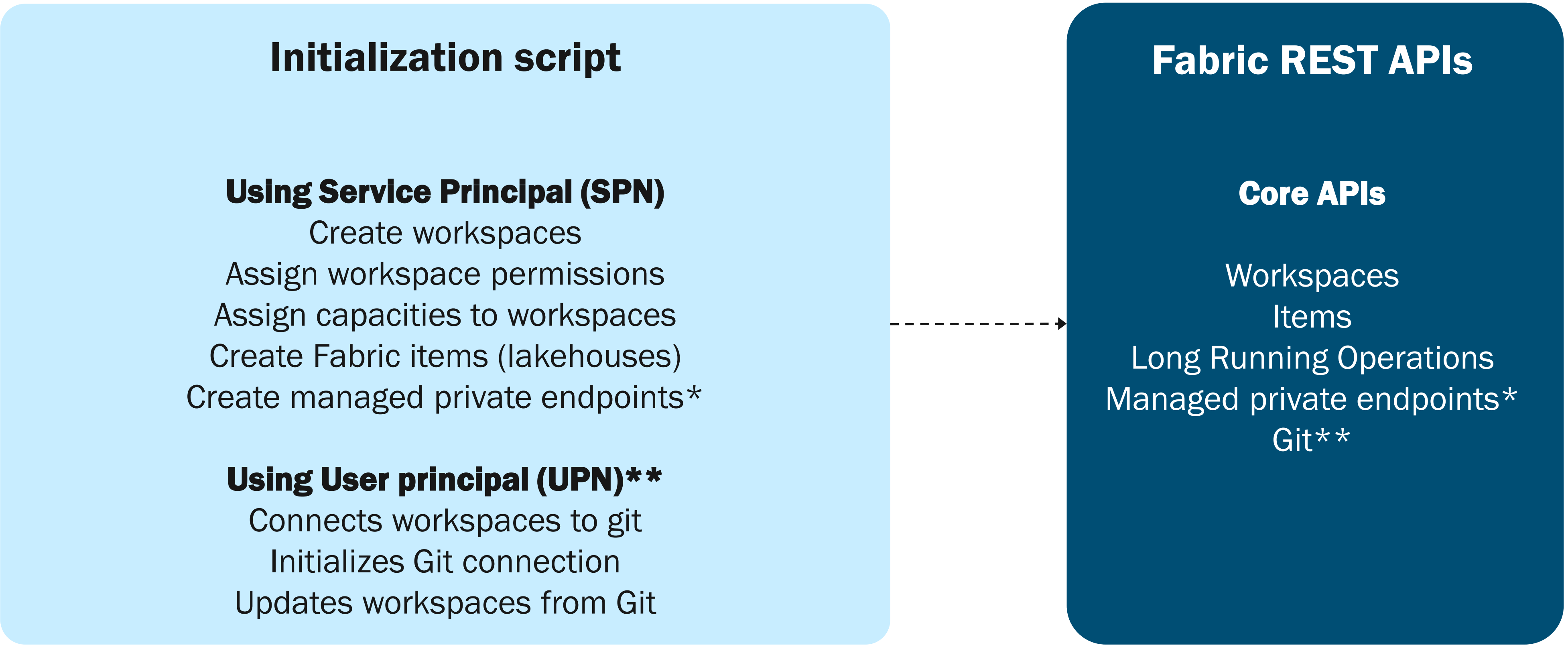
Git structure

Single repository approch:

- Automation
- Documentation
- Solution
  - Ingest
  - Prepare
  - Serve
  - Cons
  - ...
- ...

twoday

Setup you Fabric lakehouse structure in 1 minute!



Initialization script

Using Service Principal (SPN)
Create workspaces
Assign workspace permissions
Assign capacities to workspaces
Create Fabric items (lakehouses)
Create managed private endpoints*

Using User principal (UPN)**
Connects workspaces to git
Initializes Git connection
Updates workspaces from Git

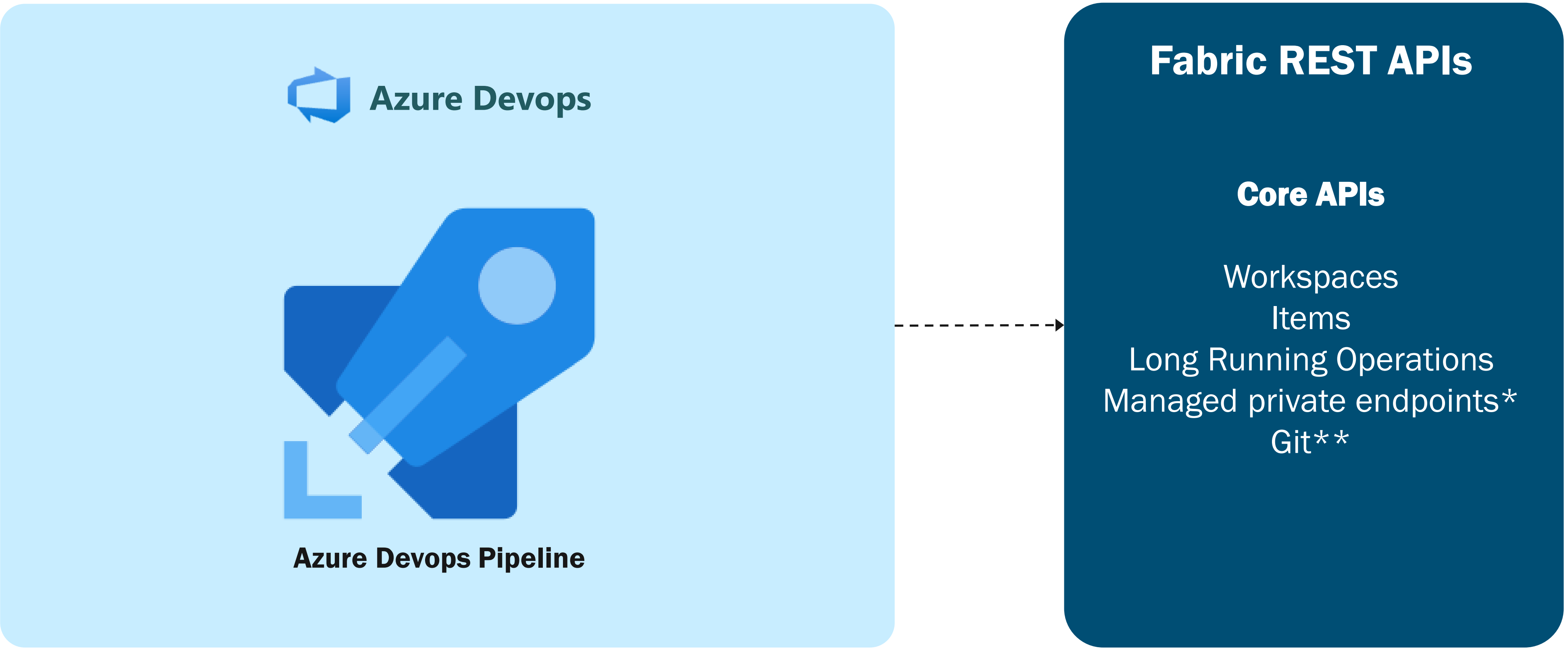
Fabric REST APIs

Core APIs

Workspaces
Items
Long Running Operations
Managed private endpoints*
Git**

Visit my blog post **”Automating Fabric: Kickstart your Fabric Setup with Python and Fabric REST APIs”**
on https://peerinsights.hashnode.dev

Setup you Fabric lakehouse structure in 1 minute!



Azure Devops

Azure Devops Pipeline

Fabric REST APIs

Core APIs

Workspaces
Items
Long Running Operations
Managed private endpoints*
Git**

Visit my blog post **"Automating Fabric: Kickstart your Fabric Setup with Python and Fabric REST APIs"** on https://peerinsights.hashnode.dev

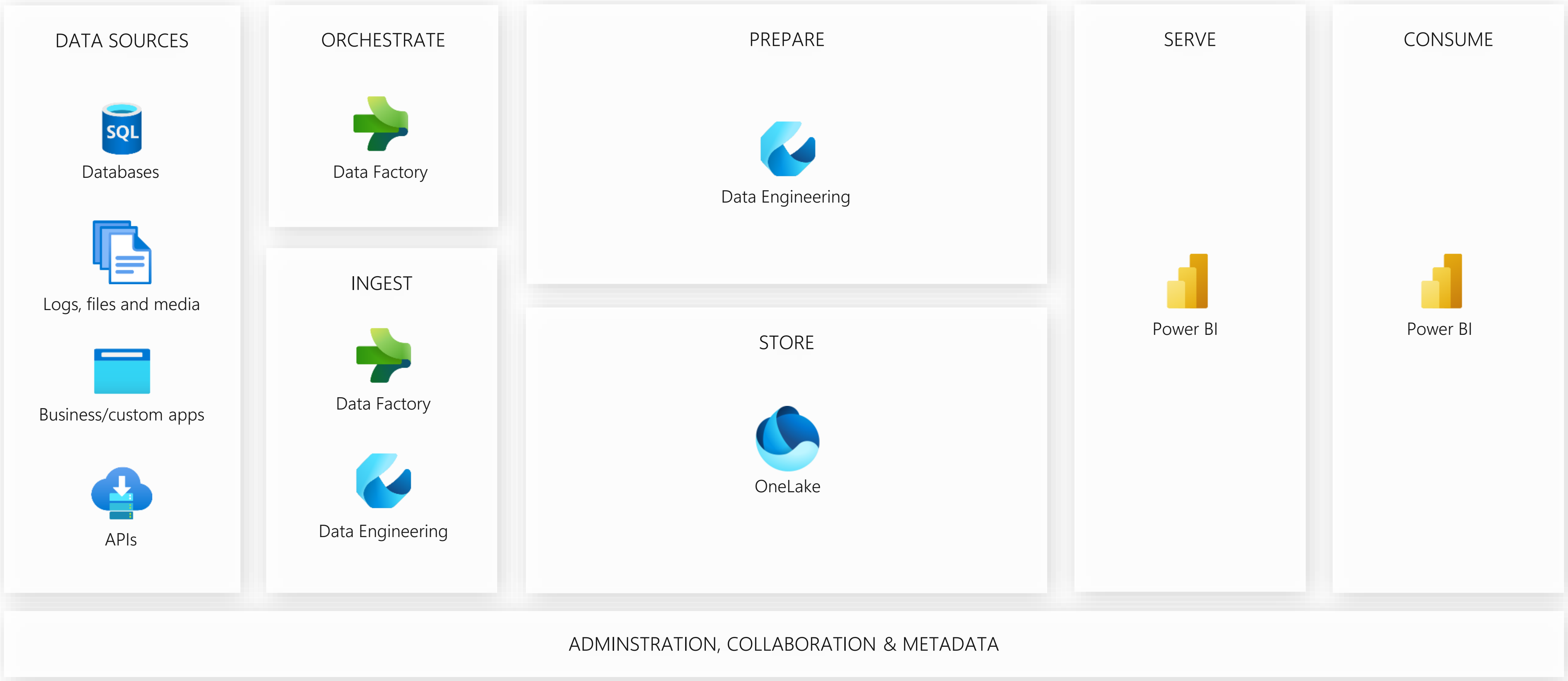
Demo

Setup of a Fabric Lakehouse
Data Platform

twoday

Ways of working utilizing a metadata-driven framework

The starting point architecture



DATA SOURCES

Databases

Logs, files and media

Business/custom apps

APIs

ORCHESTRATE

Data Factory

INGEST

Data Factory

Data Engineering

PREPARE

Data Engineering

STORE

OneLake

SERVE

Power BI

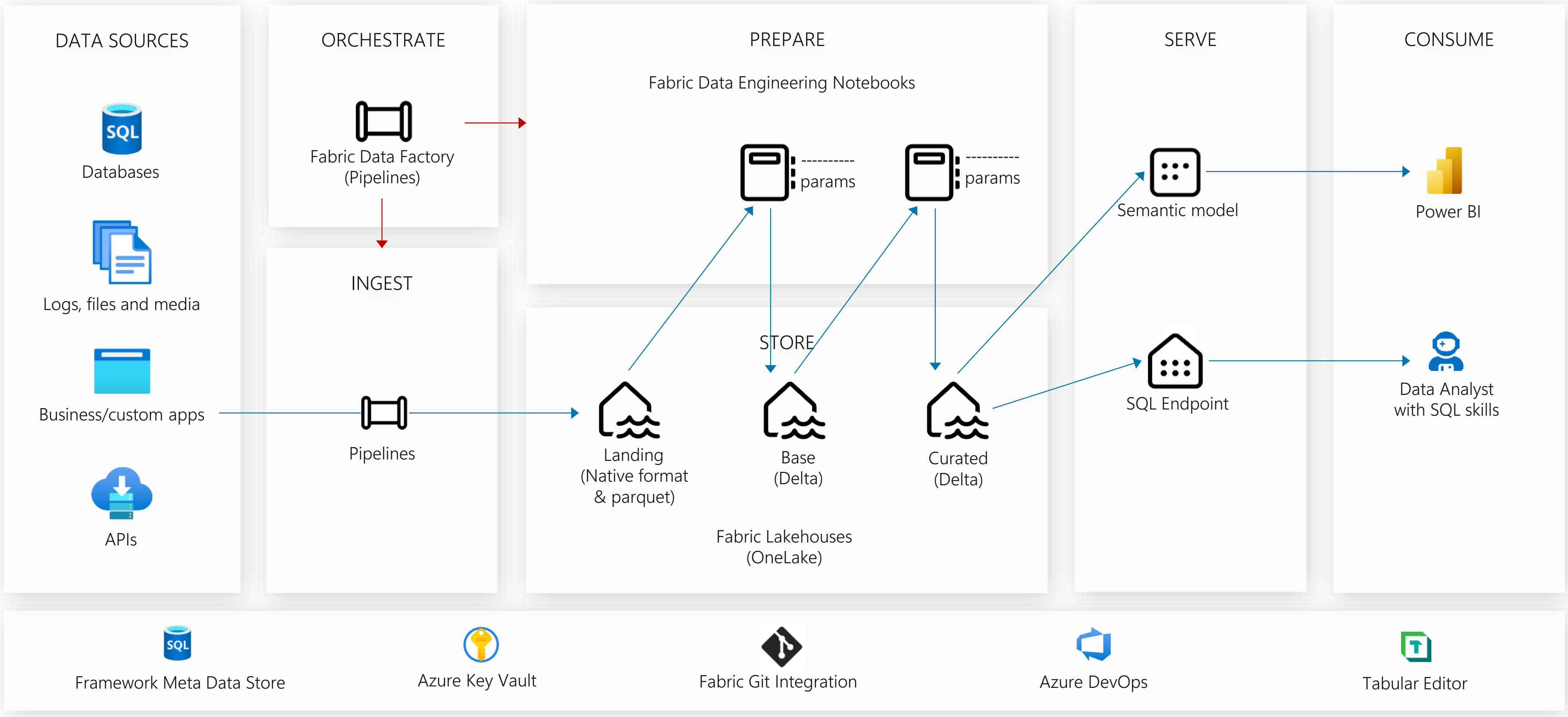
CONSUME

Power BI

ADMINSTRATION, COLLABORATION & METADATA

twoday

The starting point architecture – The data flow



twoday

Selected features of a best practice framework (AquaVilla)

- **Automated setup:** Full setup of infrastructure, workspaces and items
- **Metadata:** Integration, processing and orchestration with metadata
- **Ingestion**: Batch ingest from multiple sources using delta og full.
- **Common transformations:** Deduplication, flattening of complex data structures, data conversion, translations and more.
- **Update strategies:** Append, overwrite, upsert type 1 & 2 etc.
- **Data warehousing:** Functions specifically for dimensional modelling
- **Samples:** Sample implementations for all functionality and layers
- **Orchestration:** Orchestration through Data pipelines and Notebooks
- **Deployment: Full enterprise** CI/CD using ADO Pipelines or GitHub Actions

twoday

Accelerate development using a metadata-driven framework

Metadata store

Data ingestions
Source definition
Source object definition
Ingestion patterns

Data preparation
Landing to Base definition
Orchestration

Function library

Generic & helper functions

Reader and writer functions
Full, incremental etc.
Overwrite, append, SCD1, SCD2, etc

BI workload functions
Load dimension
Load fact table
Handle surrogate key lookups etc.

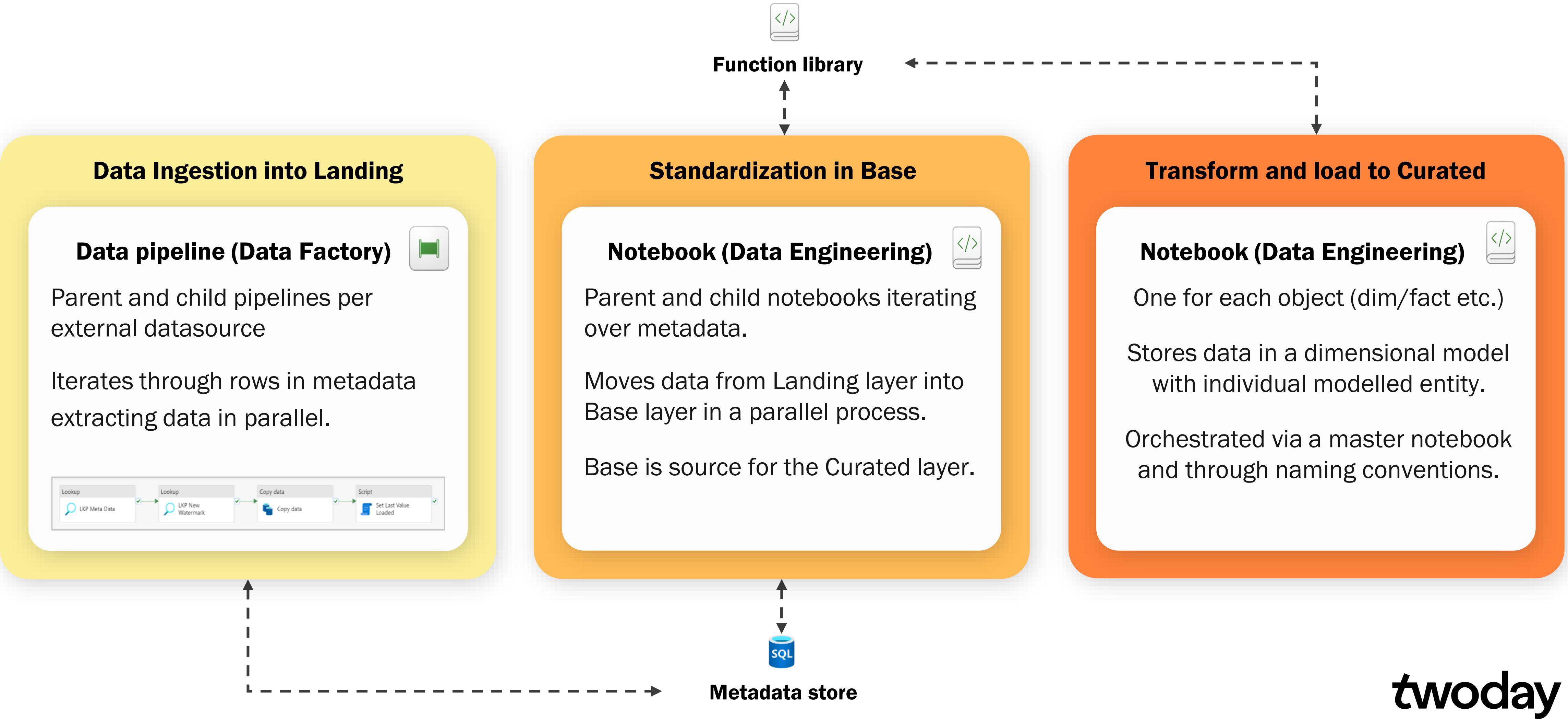
Template pipelines

Information schema pipelines
Fetches information schema
from sources

Data ingestion pipelines
Parent & Child per source
Supports full and incremental
Ingest data into Landing Lakehouse

Orchestration pipelines
Orchestrates data ingestion, data
preparation and model refresh

Accelerate development using a metadata-driven framework



Function library

Data Ingestion into Landing

Data pipeline (Data Factory)

Parent and child pipelines per external datasource

Iterates through rows in metadata extracting data in parallel.

Standardization in Base

Notebook (Data Engineering)

Parent and child notebooks iterating over metadata.

Moves data from Landing layer into Base layer in a parallel process.

Base is source for the Curated layer.

Transform and load to Curated

Notebook (Data Engineering)

One for each object (dim/fact etc.)

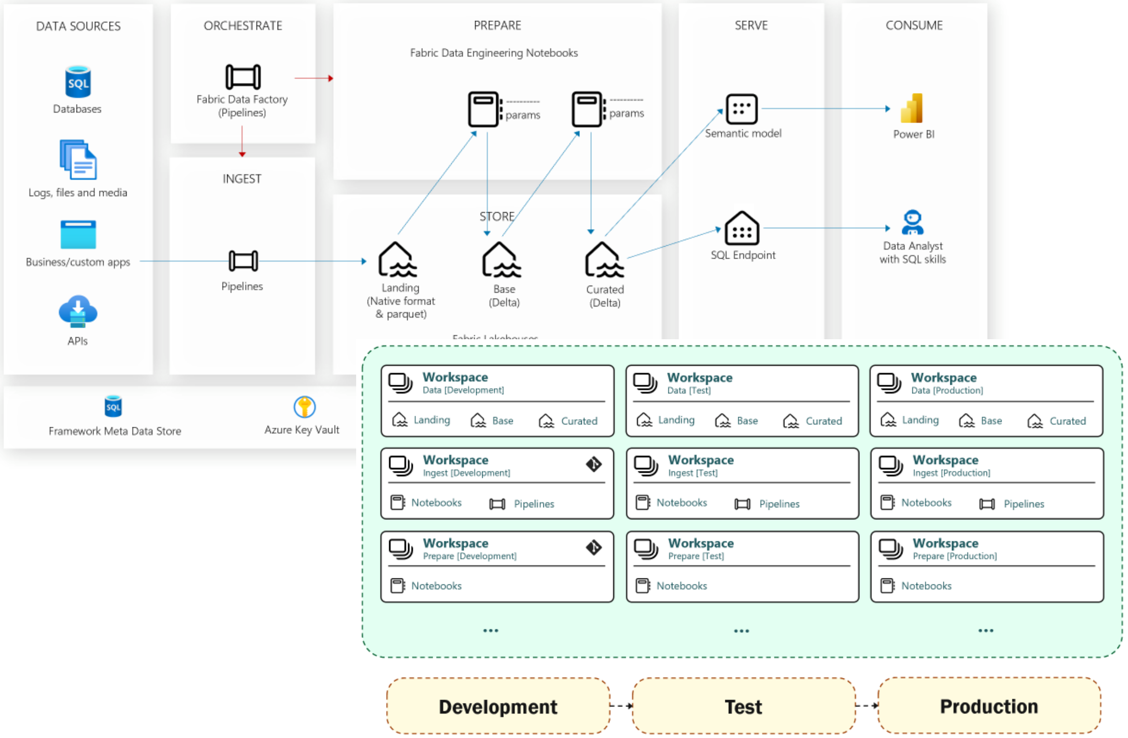
Stores data in a dimensional model with individual modelled entity.

Orchestrated via a master notebook and through naming conventions.

Metadata store

twoday

Ways of working in Fabric – Building our solution



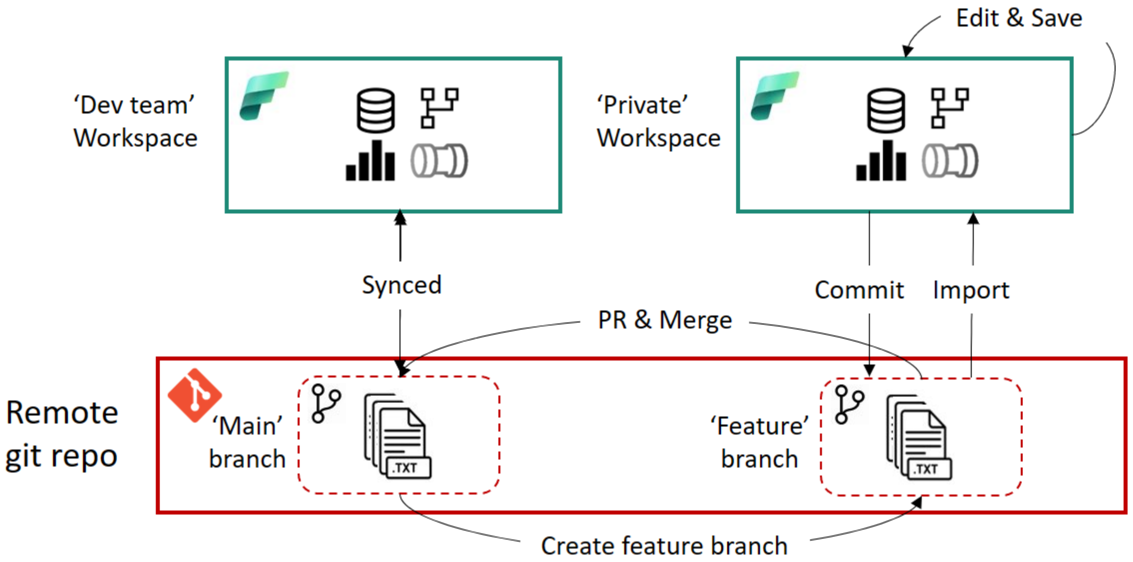
Fabric Best practice solution implementation

- Branch out to a feature branch
- Implement changes in isolated Fabric workspace
- Commit changes back into Git repository
- Create pull request
- Review, rework and test
- Merge into main and delete development branch etc.
- Deploy changes through a CI/CD pipeline

https://learn.microsoft.com/en-us/fabric/cicd/best-practices-cicd

https://learn.microsoft.com/en-us/power-bi/guidance/powerbi-implementation-planning-usage-scenario-enterprise-content-publishing

Ways of working in Fabric



Fabric UI

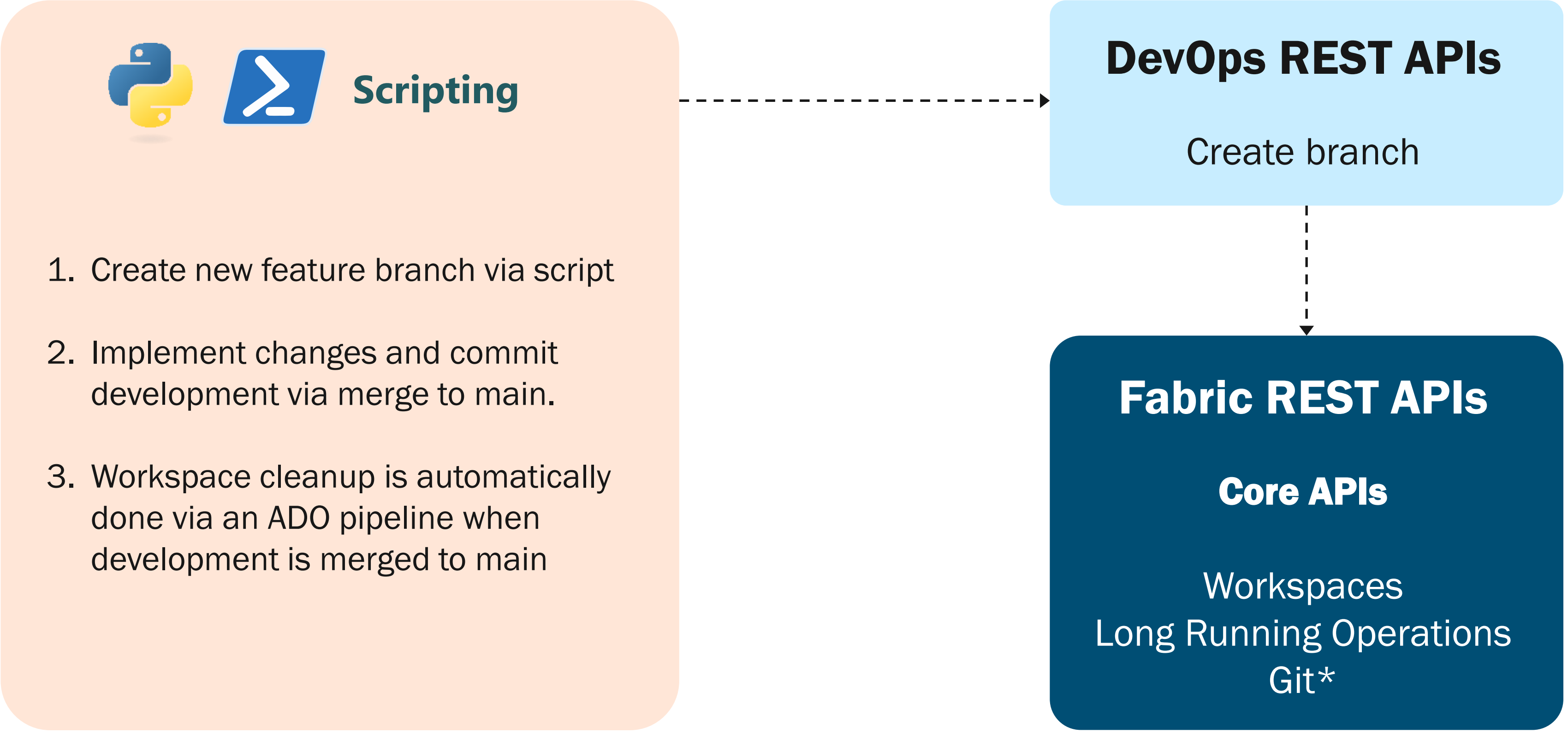
Scripting

Azure Devops

→ https://learn.microsoft.com/en-us/fabric/cicd/best-practices-cicd

→ https://learn.microsoft.com/en-us/power-bi/guidance/powerbi-implementation-planning-usage-scenario-enterprise-content-publishing

Ways of working in Fabric – Automate and accellerate



Scripting

1. Create new feature branch via script
2. Implement changes and commit development via merge to main.
3. Workspace cleanup is automatically done via an ADO pipeline when development is merged to main

DevOps REST APIs

Create branch

Fabric REST APIs

Core APIs

Workspaces
Long Running Operations
Git*

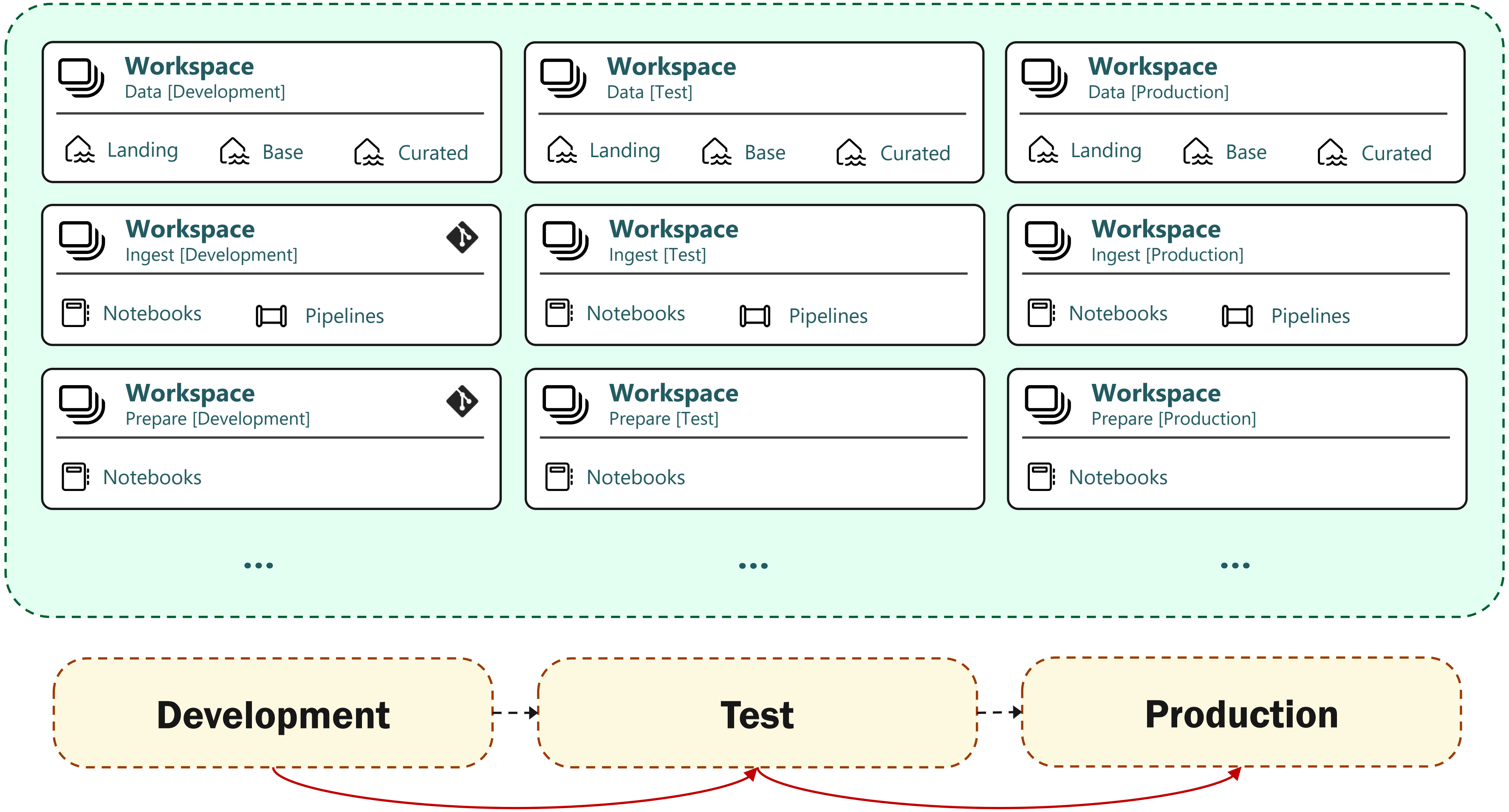
Demo

Ways-of-working
A live feature development flow

twoday

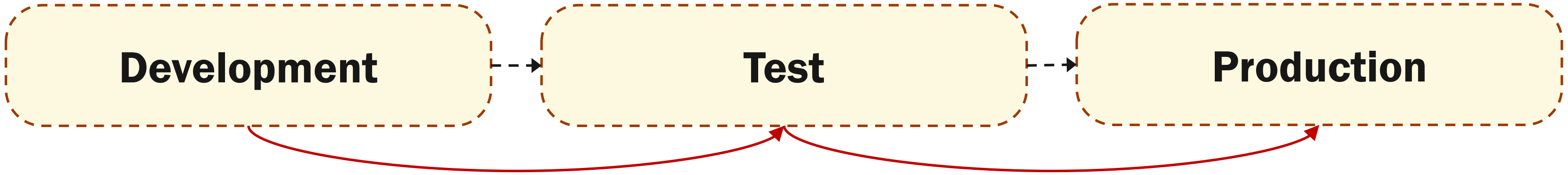
CI/CD
Deploying your solution

Environments, stages and Git structure



twoday

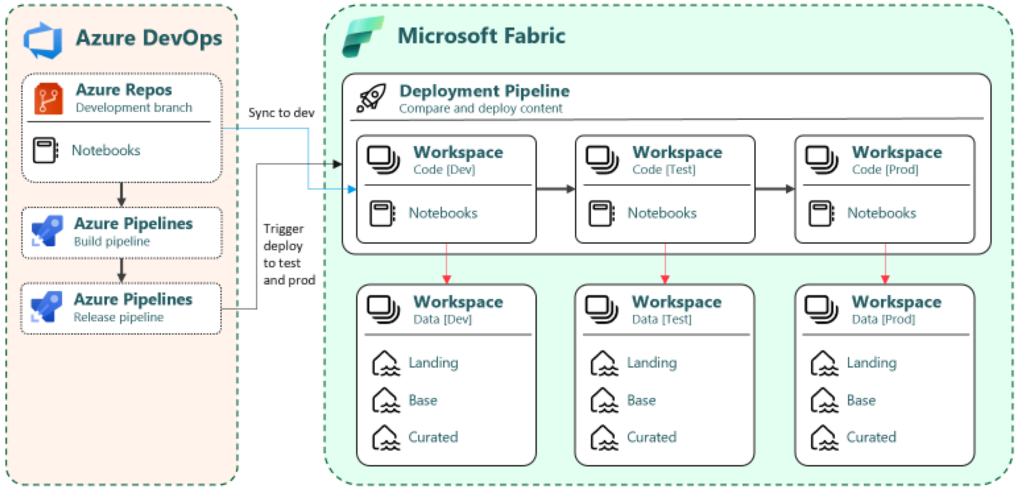
Environments, stages and Git structure



Development

Test

Production



CI/CD process with Fabric Deployment pipelines

Azure DevOps

Azure Repos
Development branch

Notebooks

Azure Pipelines
Build pipeline

Azure Pipelines
Release pipeline

Sync to dev

Trigger deploy to test and prod

Microsoft Fabric

Deployment Pipeline
Compare and deploy content

Workspace
Code [Dev]
Notebooks

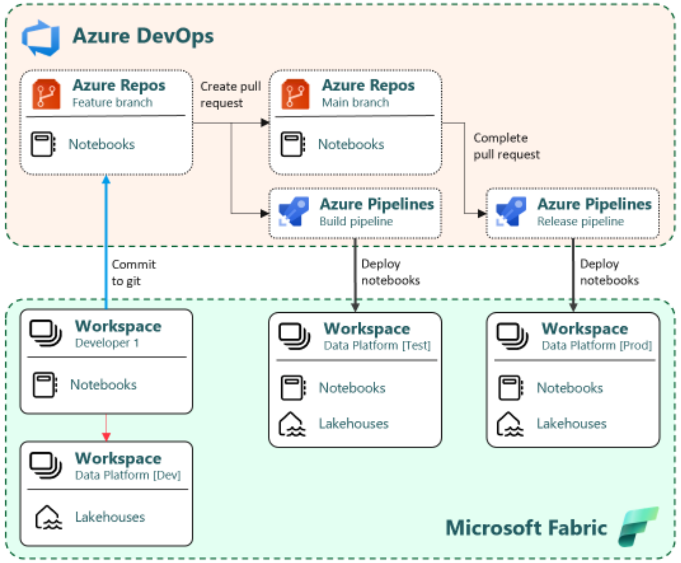
Workspace
Code [Test]
Notebooks

Workspace
Code [Prod]
Notebooks

Workspace
Data [Dev]
Landing
Base
Curated

Workspace
Data [Test]
Landing
Base
Curated

Workspace
Data [Prod]
Landing
Base
Curated



CI/CD process with Azure DevOps / GitHub

Azure DevOps

Azure Repos
Feature branch
Notebooks

Create pull request

Azure Repos
Main branch
Notebooks

Complete pull request

Azure Pipelines
Build pipeline

Azure Pipelines
Release pipeline

Commit to git

Deploy notebooks

Deploy notebooks

Workspace
Developer 1
Notebooks

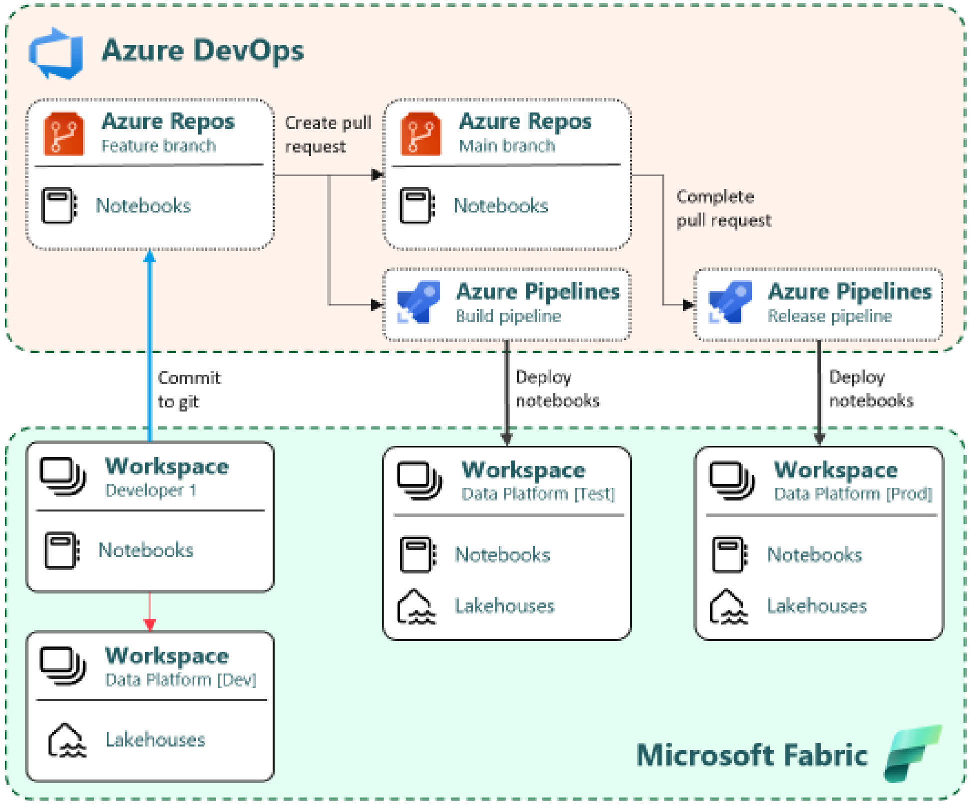
Workspace
Data Platform [Dev]
Lakehouses

Workspace
Data Platform [Test]
Notebooks
Lakehouses

Workspace
Data Platform [Prod]
Notebooks
Lakehouses

Microsoft Fabric

The deployment process (simplified)



Azure DevOps

Azure Repos
Feature branch
Notebooks

Create pull request

Azure Repos
Main branch
Notebooks

Complete pull request

Azure Pipelines
Build pipeline

Azure Pipelines
Release pipeline

Commit to git

Deploy notebooks

Deploy notebooks

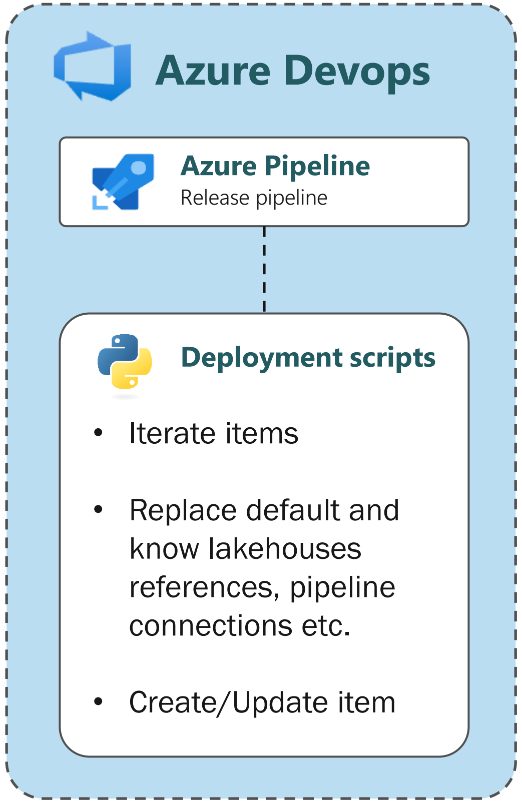
Workspace
Developer 1
Notebooks

Workspace
Data Platform [Dev]
Lakehouses

Workspace
Data Platform [Test]
Notebooks
Lakehouses

Workspace
Data Platform [Prod]
Notebooks
Lakehouses

Microsoft Fabric



Azure Devops

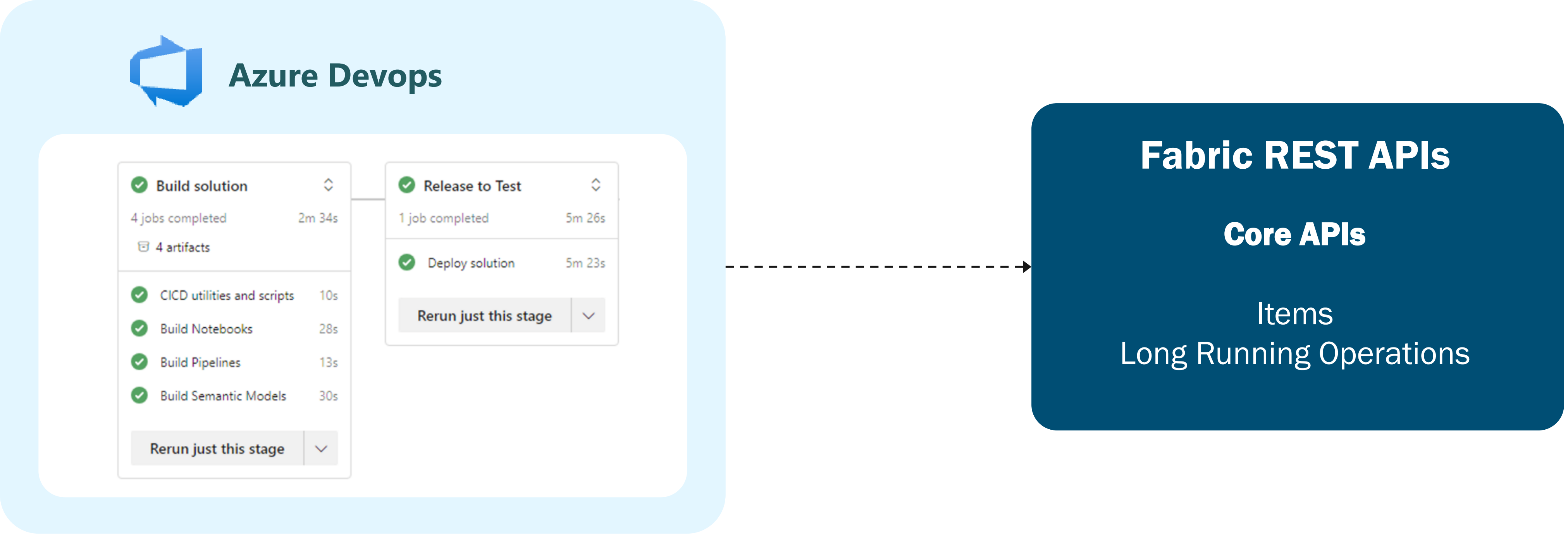
Azure Pipeline
Release pipeline

Deployment scripts

- Iterate items
- Replace default and know lakehouses references, pipeline connections etc.
- Create/Update item

* Deployment of Data Pipelines and other items currently not supporting Service Prinicpal through an Azure DevOps pipeline can be accomplished using a temporary workaround.

Utilizing Fabric REST APIs for build and release (CI/CD)



Azure Devops

Build solution

4 jobs completed 2m 34s

4 artifacts

CICD utilities and scripts 10s

Build Notebooks 28s

Build Pipelines 13s

Build Semantic Models 30s

Rerun just this stage

Release to Test

1 job completed 5m 26s

Deploy solution 5m 23s

Rerun just this stage

Fabric REST APIs

Core APIs

Items

Long Running Operations

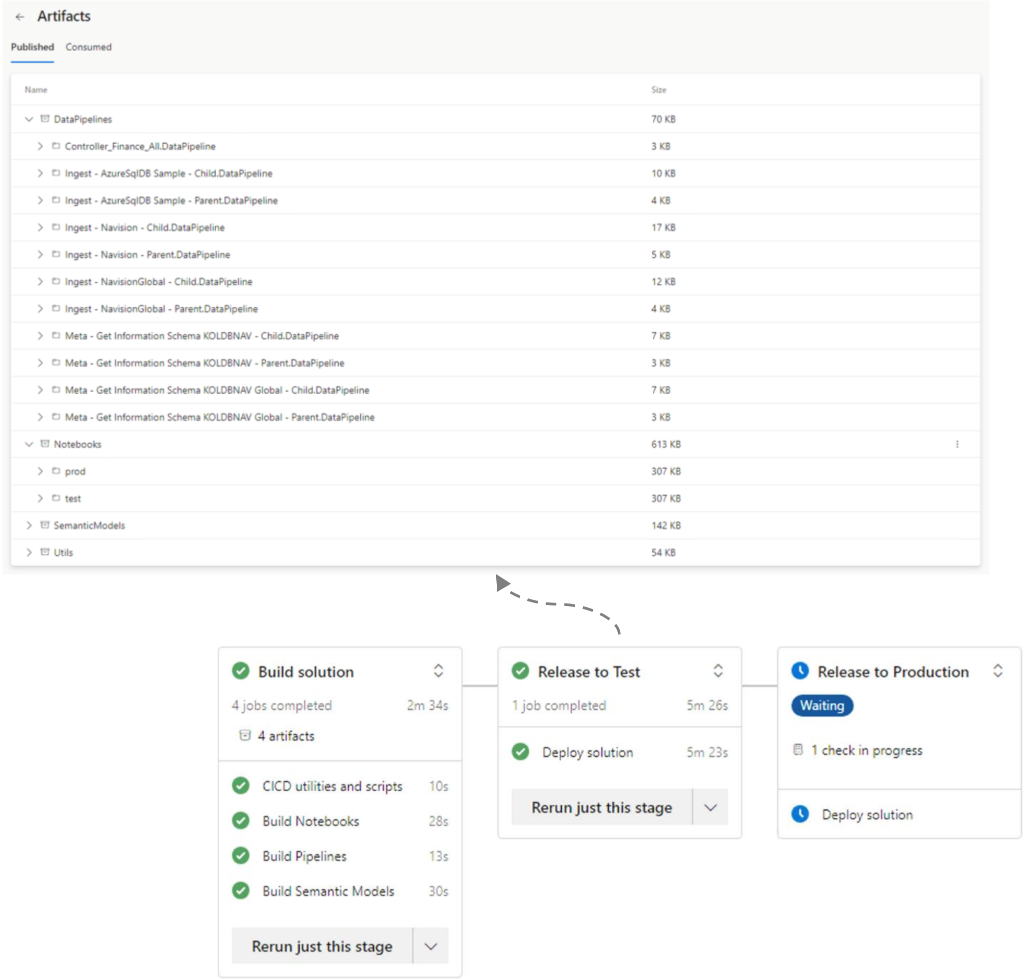
Azure DevOps – An example of activities of a release pipeline

Build and releases activities

1. Notebooks
   - Iterate repository folder holding notebook items
   - Replace default and known lakehouses
   - Release using Fabric REST APIs

2. Data Pipelines *
   - Iterate repository folder holding data pipeline items
   - Replace notebook, lakehouse and connection references
   - Release in a sorted requence using Fabric REST APIs

3. Semantic Models
   - Iterate repository folder holding semantic models
   - Utilize Tabular Editor portable version to build bim-file and change source connection information
   - Release using Tabular Editor portable version



← Artifacts

Published Consumed

Name	Size
DataPipelines	70 KB
Controller_Finance_All.DataPipeline	3 KB
Ingest - AzureSqlDB Sample - Child.DataPipeline	10 KB
Ingest - AzureSqlDB Sample - Parent.DataPipeline	4 KB
Ingest - Navision - Child.DataPipeline	17 KB
Ingest - Navision - Parent.DataPipeline	5 KB
Ingest - NavisionGlobal - Child.DataPipeline	12 KB
Ingest - NavisionGlobal - Parent.DataPipeline	4 KB
Meta - Get Information Schema KOLDBNAV - Child.DataPipeline	7 KB
Meta - Get Information Schema KOLDBNAV - Parent.DataPipeline	3 KB
Meta - Get Information Schema KOLDBNAV Global - Child.DataPipeline	7 KB
Meta - Get Information Schema KOLDBNAV Global - Parent.DataPipeline	3 KB
Notebooks	613 KB
prod	307 KB
test	307 KB
SemanticModels	142 KB
Utils	54 KB

Build solution
4 jobs completed 2m 34s
4 artifacts
CICD utilities and scripts 10s
Build Notebooks 28s
Build Pipelines 13s
Build Semantic Models 30s

Release to Test
1 job completed 5m 26s
Deploy solution 5m 23s

Release to Production
Waiting
1 check in progress
Deploy solution

twoday

Workaround for releasing Data Pipelines

- No Service principal support for Data Pipelines
  NOTE! Deployment must be performed in the context of a User identity

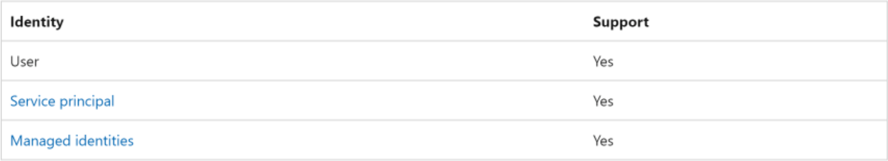
- Deployment can be accomplished using different approaches… But are limited and/or require workarounds…
  - Using Deployment Pipelines
  - Using Azure DevOps Service Connection and User indentity without MFA activated
  - Local deployment through Python, Powershell or simular utilzing the Fabric REST APIs
  - 3rd party/custom application utilzing the Fabric REST APIs
  - Using Azure DevOps with runtime parameter or key vault secret holding user identity token

Demo

Deploying Fabric items using Azure DevOps Pipelines

twoday

Status, current limitations and future CI/CD flow



Identity	Support
User	Yes
Service principal	Yes
Managed identities	Yes

- Not quite there yet on fullblown CI/CD setup

- The goal is a CI/CD flow fully supported by Azure DevOps, including:
  - Release of all item types (notebooks, pipelines etc.)
  - Workspace setup and cleanup

- Primary limitations preventing this is missing support for Service Principals
  - But we are approaching… And many items just got support for SPN and MSI.

- Terraform Provider for Fabric will be my recommendation on future IaC tool for setting up not only Fabric solutions but also other related services like Azure KeyVault etc.

Also check out my blog which holds a few articles on automating Microsoft Fabric – and stay tuned… more will come ☺
https://peerinsights.hashnode.dev/

Q&A

twoday

Where tomorrow is made