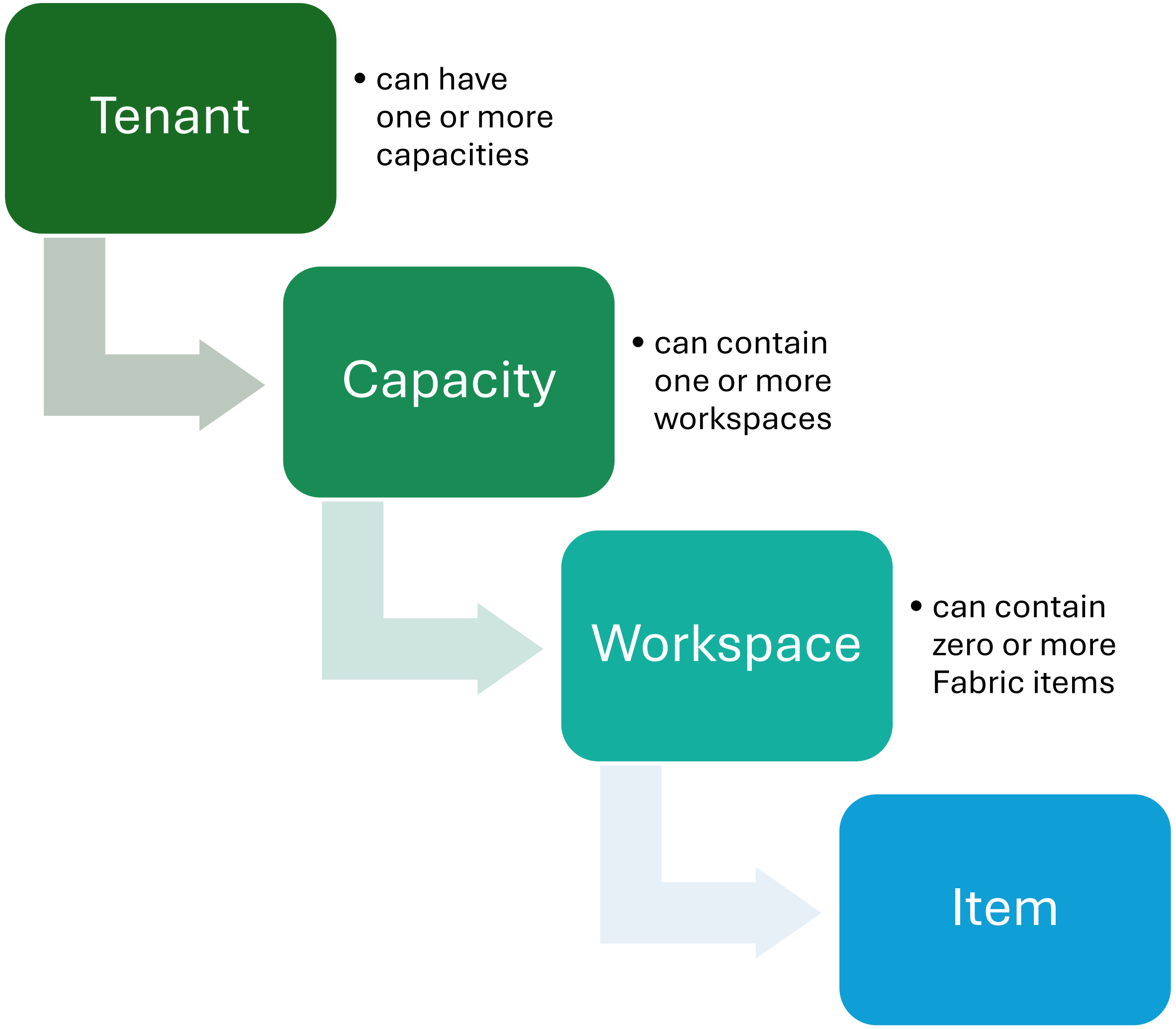
Deployment patterns

Andrea Benedetti
Sr Cloud Architect, Microsoft

in /in/abenedetti X @anBenedetti https://github.com/anbened

#FabricGarage | 002 | 2024.05.23

4 Levels



Tenant

- can have one or more capacities

Capacity

- can contain one or more workspaces

Workspace

- can contain zero or more Fabric items

Item

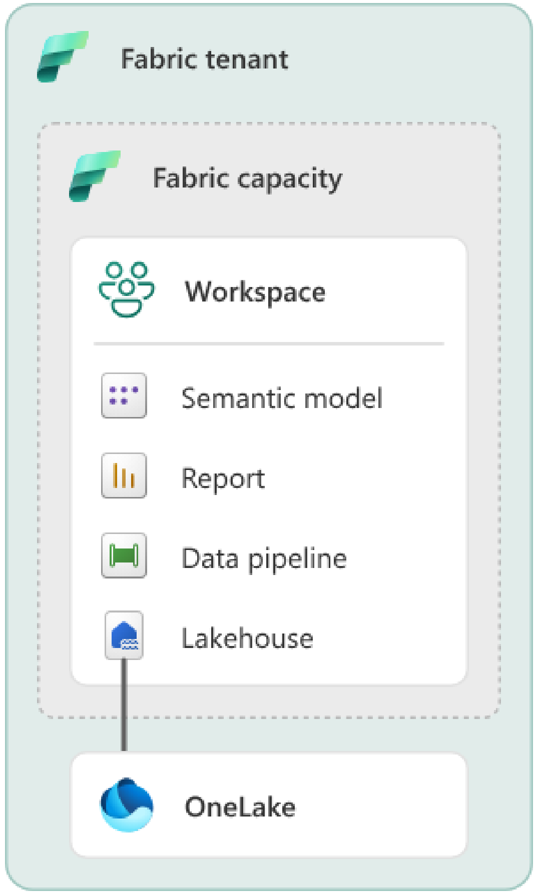
An org structure or objectives in the areas of security, scale, governance, and application lifecycle might influence its choice of deployment pattern

All Fabric deployment patterns

- Workspaces → boundaries for scale, governance, and security
- Domains → to manage multiple workspaces
  - When same business unit
  - When data (business domain) spans more than one workspace
- Capacities → to scale compute resources
  - dedicated capacities per ws when specific perf levels must be met
- OneLake data hub → to promote discovery / use of data assets
- OneSecurity → to set up data security policies for data assets

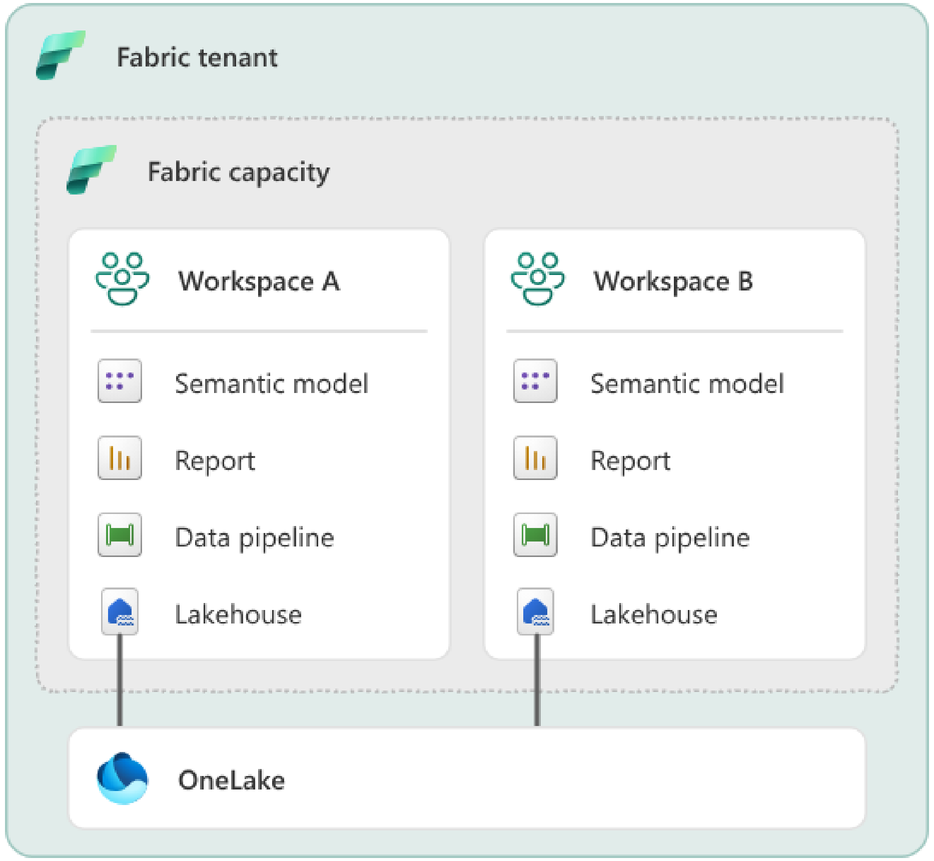
Pattern 1: Monolithic deployment

- For organizations that want to have faster (or slower) time to market by organizing teams that can cross-collaborate, with lower restrictions on data usage
- In this deployment pattern, you provision a single workspace to cater to all your use cases. All business units work within the same, single workspace.



Pattern 2: Multiple workspaces backed by a single Fabric capacity

- For organizations that want to provide isolated environments for teams to work in, with a central team that is responsible for providing and managing infrastructure.
- In this deployment pattern, you use separate workspaces. Because a single capacity is shared across workspaces, workloads that run concurrently at any time might affect the performance of jobs and interactive queries.



Fabric tenant

Fabric capacity

Workspace A

Semantic model

Report

Data pipeline

Lakehouse

Workspace B

Semantic model

Report

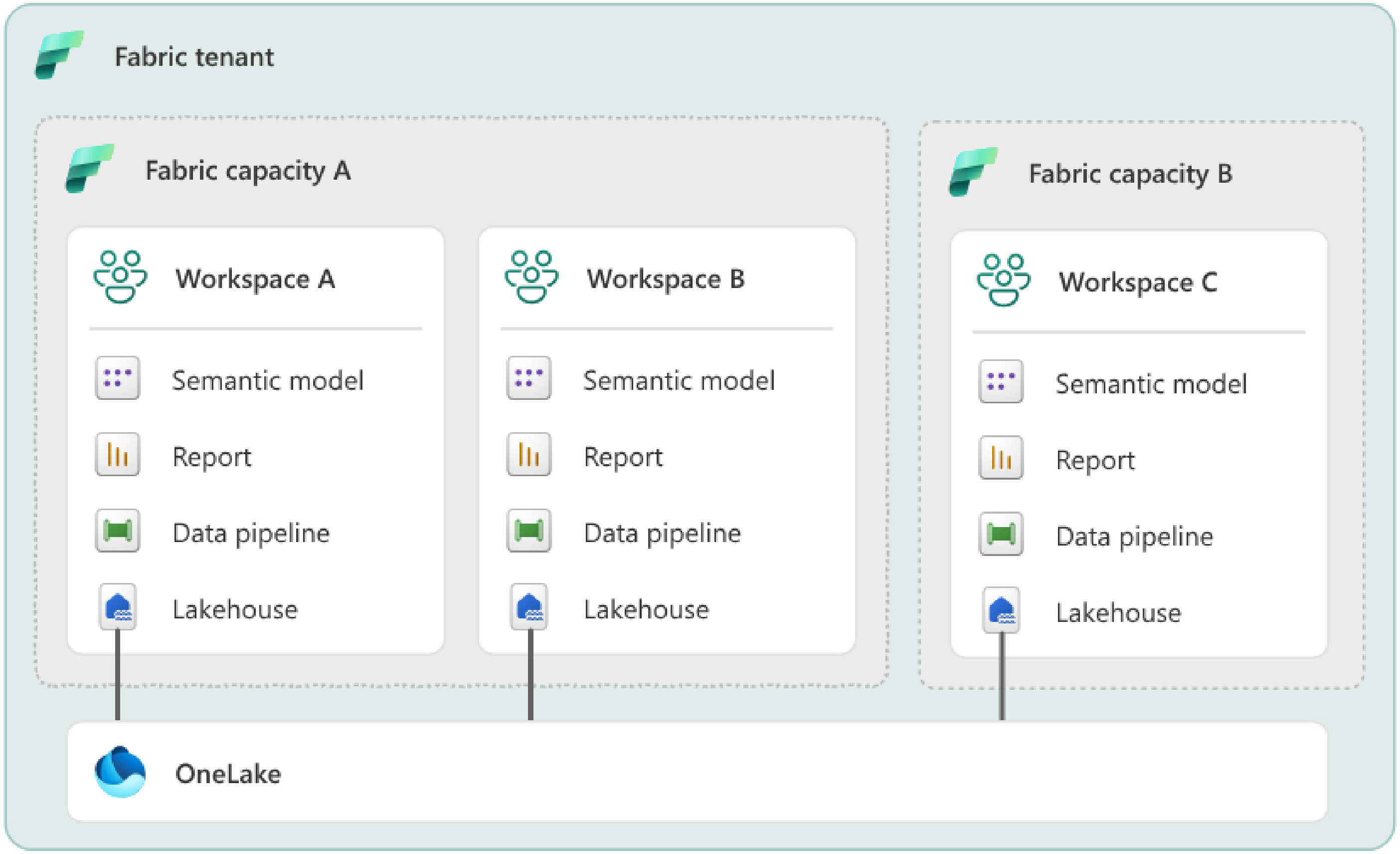
Data pipeline

Lakehouse

OneLake

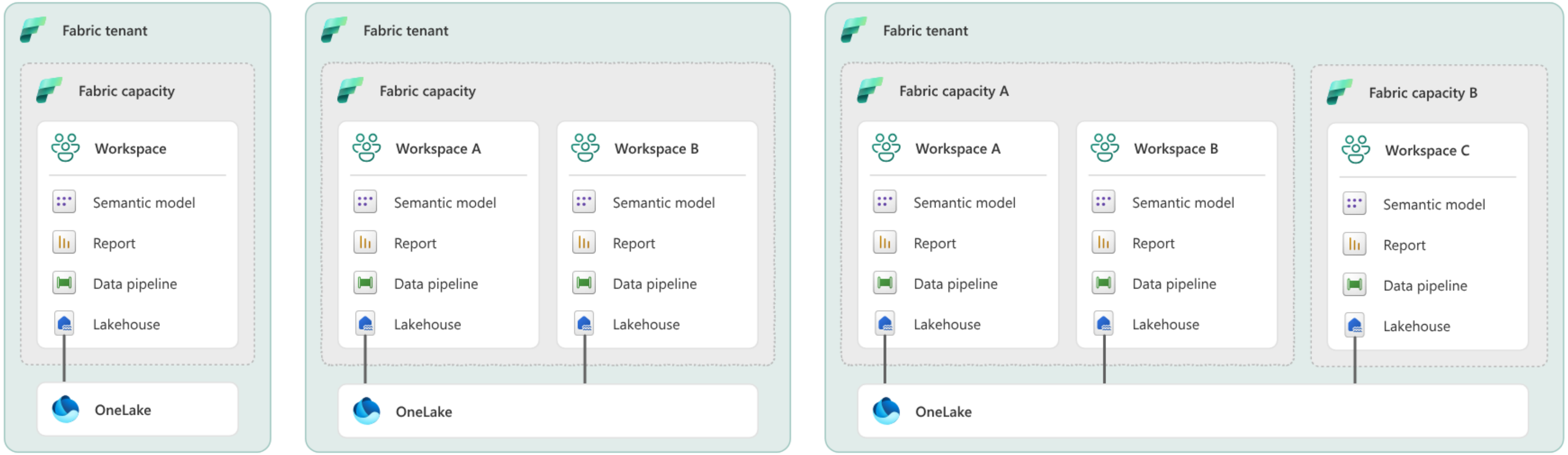
Pattern 3: Multiple workspaces backed by separate capacities

- For organizations that want an entirely decentralized model that gives business units or teams the freedom to control and manage their own data platforms.
- In this deployment pattern, you achieve separation between business units for governance and performance.



Pattern 4: Multiple Fabric tenants

- An organization might choose to use a hybrid approach in which it combines multiple patterns to achieve its requirements.
- When separate Fabric tenants are deployed, all instances of Fabric are separate entities with respect to governance, management, administration, scale, and storage.



Links

- Deployment patterns for MS Fabric - Azure Architecture Center

Andrea Benedetti

Sr Cloud Architect, Microsoft

/in/abenedetti @anBenedetti https://github.com/anbened

#FabricGarage | 002 | 2024.05.23