



Let's Solve



A Larsen & Toubro
Group Company

Whitepaper

Data Mesh on Azure

Get more out of distributed data with improved transparency, end-to-end compliance, flexibility & independence, faster access, and accurate data delivery.

Table of Content

1. Introduction Page **03**

2. What is Data Mesh? Page **04**

3. Data Mesh Architecture principles Page **05**

4. Data Mesh Architecture pattern Page **07**

5. How does an organization prepare for a Data Mesh? Page **08**

6. Operationalizing Data Mesh on Azure

6.1 Domain oriented and decentralized data ownership

6.2 Data as a Product

6.3 Self-service data infrastructure

Portal Architecture | ARM template based framework | User Personas

6.4 Federated computational governance

6.5 Data Mesh Conceptual Architecture

6.6 Data product marketplace

Page **10**

7. Conclusion Page **23**

8. References Page **24**

Introduction

Data warehouse, though dominant analytical paradigm, often creates challenges due to data volume, velocity, and variety. Centralized IT teams tend to become bottlenecks trying to cope with various business units' data hosting and processing demands. Often this leads to delay in go-to-market strategy and stale analytical product due to the time it takes for the IT team to understand and develop the analytical solution. To mitigate these challenges and accelerate the creation and sharing of Data as a product, organizations are looking for a shift in how analytical platforms are built, served, and governed.

The answer to the many issues and roadblocks in the traditional organizational setup and the monolithic architectures is addressed by a Data Mesh that focuses on building modern distributed architecture at scale by paving the way for modern architecture designed for scalability and flexibility and federated computational governance.



What is Data Mesh?

The concept of Data Mesh, a term coined by Zhamak Dehghani, encompasses data, technology, processes, and organization. On a conceptual level, it's a democratized approach to managing data where various domains operationalize their data, relieving the Central IT team from designing and developing Data products. Instead, IT teams focus on providing and governing IT resources using a self-service platform.

Data Mesh challenges the idea of conventional centralization of data. Rather than looking at data as one huge repository, Data Mesh considers the decomposition of independent data products. This shift is backed by a modern and self-service data platform, which is typically designed using cloud-native technologies from centralized to federated ownership. Overall, Data Mesh implementation is managed by federated computational governance through catalogs and policies.

Data Mesh

Architecture principles

Borrowing the concept from distributed computing, Data Mesh focuses on disseminated data products aligned around enterprise domains and possessed by cross-functional teams with data engineers and product owners, utilizing standard foundational infrastructure as a platform to host, prep, and serve their data-driven information.

Data Mesh architecture adheres to 4 principles. They are -

1

Decentralized data domain:

Fundamental units that represent each functional domain and host multiple data products about that functional domain. Domains are provisioned and governed by a centralized IT team but owned and maintained by individual domain-specific teams.

2

Data as a Product:

Data Mesh approaches data as an asset and offers the same as a product. Data Product works on logical architecture called a data quantum that controls and encapsulates all the structural components to share data as a product - data, metadata, code, policies, etc. Data Product should abide by the below-mentioned baseline usability attributes.



3

Self-service data infrastructure platform:

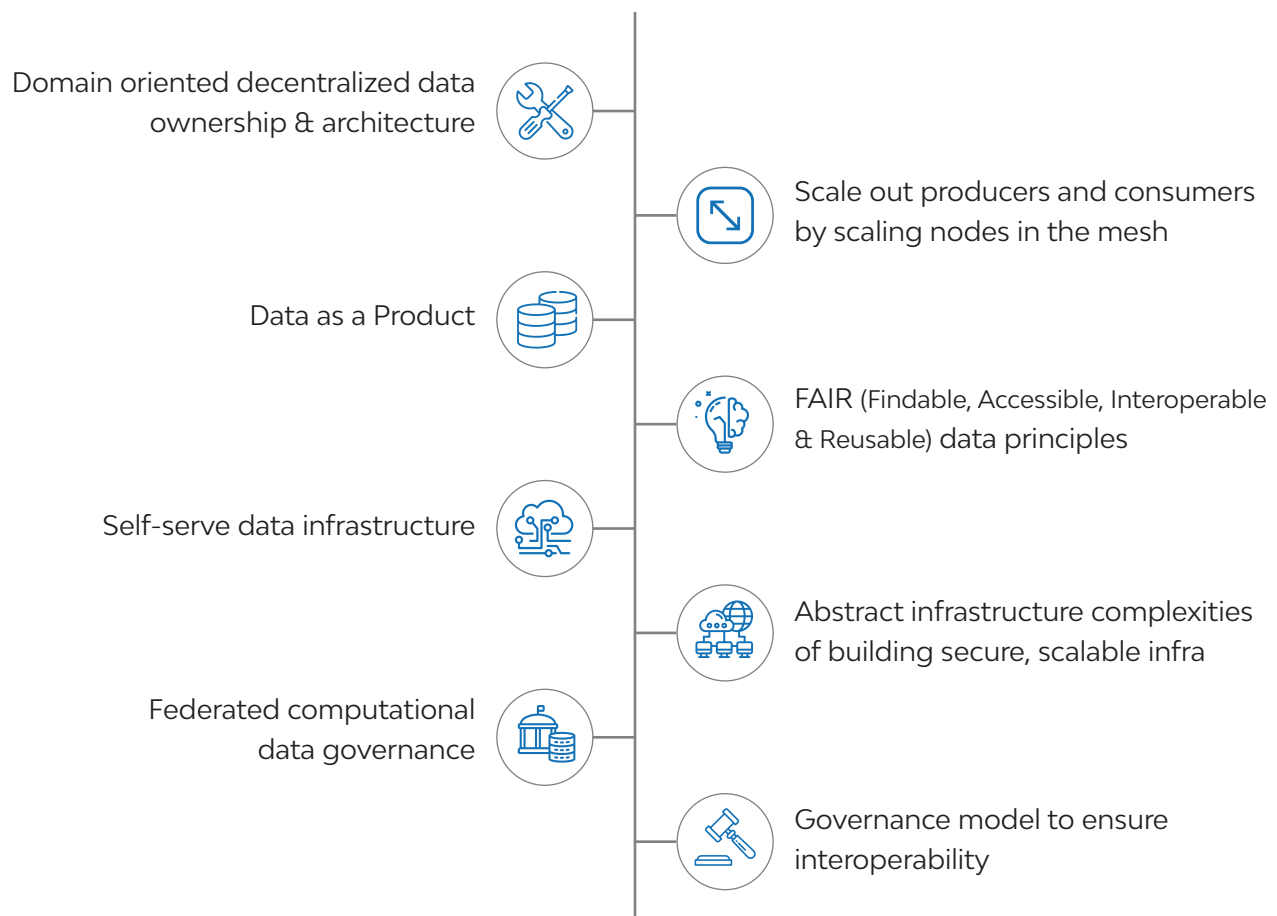
This platform is implemented using Infrastructure as a code, provisioning fast yet consistent domains and required services using best practices and best-of-breed technologies.

4

Federated computational governance:

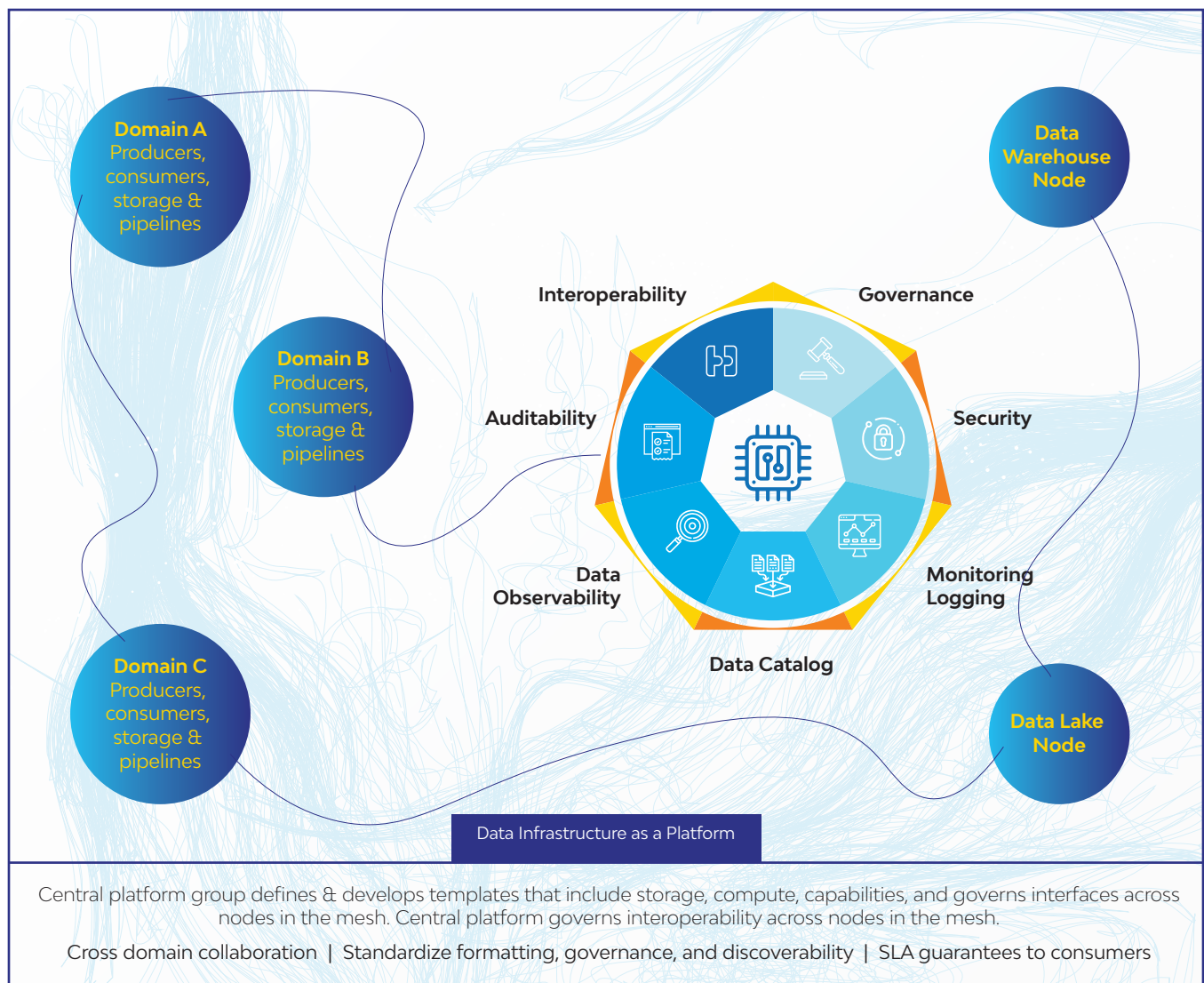
Due to the decentralized nature of data mesh, it is of prime importance to impose consistent and computational governance. It maintains required standards across different data domains and is required to be centrally managed and governed.

These principles can be briefly visualized as shown in the below diagram.



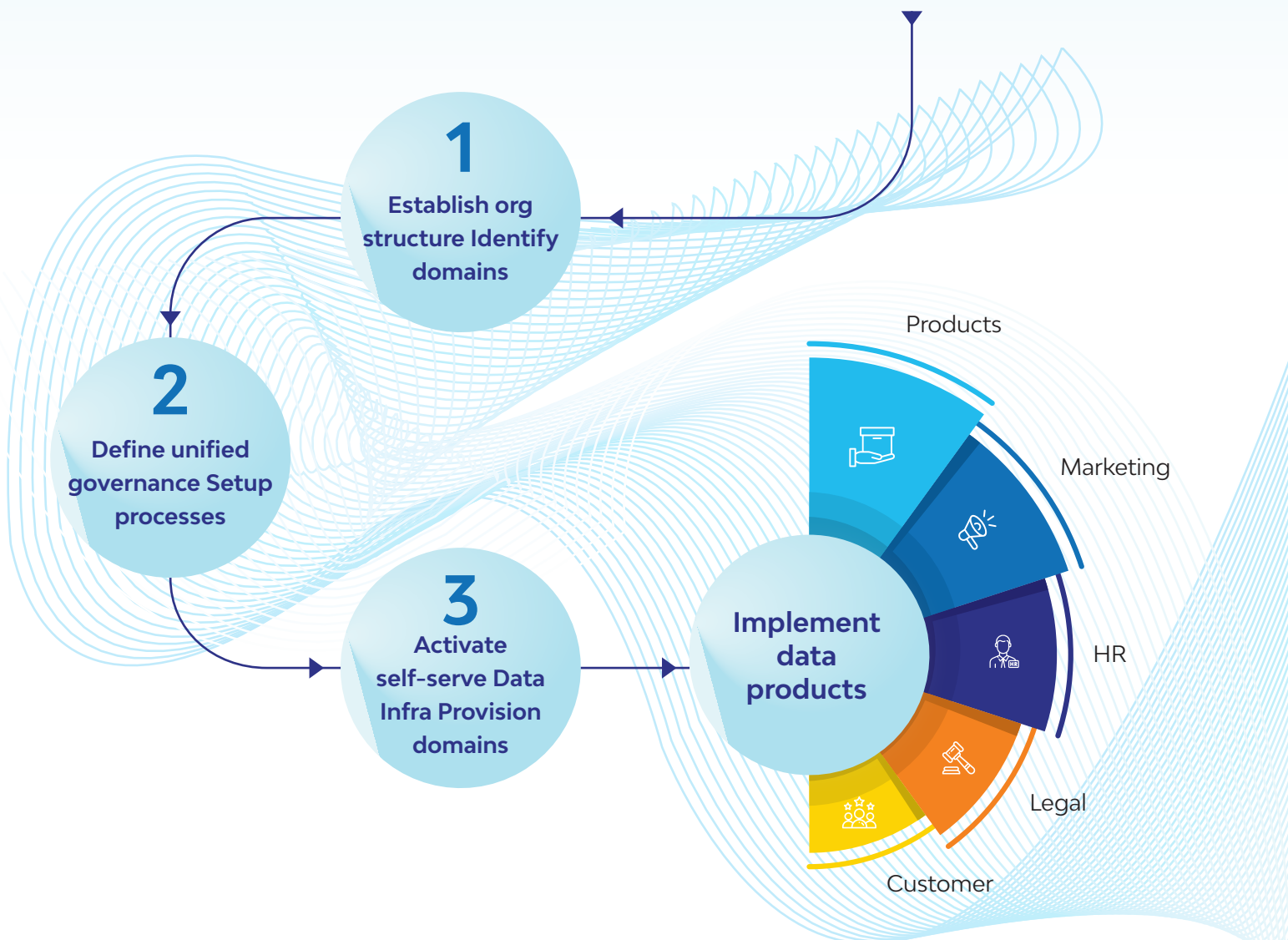
Data Mesh Architecture pattern

Data Mesh, in essence, is an architecture paradigm where each functional data domain is represented as nodes and is interconnected, managed, and governed by a centralized IT/Governance node. Each data domain can host multiple data products that can be shared across different data domains using the same centralized IT/governance mode. The conceptual architecture of a Data Mesh is depicted as shown below.



How does an organization prepare for a Data Mesh?

Although a technical concept, Data Mesh demands a change in work culture and a different perspective to observe and share the data. Hence, Data Mesh implementation needs to be approached from technical and behavioral aspects. Before defining any technical architecture or framework, it is essential to visualize the organization as different functional domains to define and fulfill the first principle of Data Mesh, i.e., domain oriented decentralized data ownership. Data Mesh can be approached in 4 linear steps.



1

Establish org structure and identify domains

This is very important and intensive work that needs a thorough understanding of business and its modularity. Defining domains with clear ownership and data products requirements can eventually produce Data Mesh with high-performing data products and seamless governance.

2

Define unified governance setup process

Ideally, a governance committee will be established and represented by participants from different business units like security, compliance, legal, IT, and data domains. This team aims to identify, define, and impose governance policies that adhere to organizational and regional /government guidelines/policies. Having centralized federated computational governance can establish consistent and uniform policies across different data domains.

3

Activate self-service infrastructure provisioning

Establish a centralized platform to help self-service infrastructure provisioning for faster domain and data product creations. This centralized platform will govern data mesh as per defined governance policies.

A self-service infrastructure platform provides a user-friendly, quick and compliant environment, taking away all the complexity of the underlying technical platform.

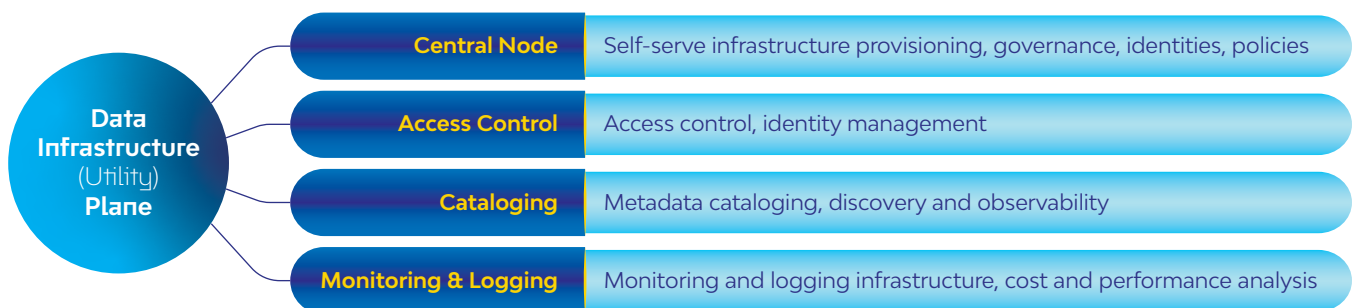
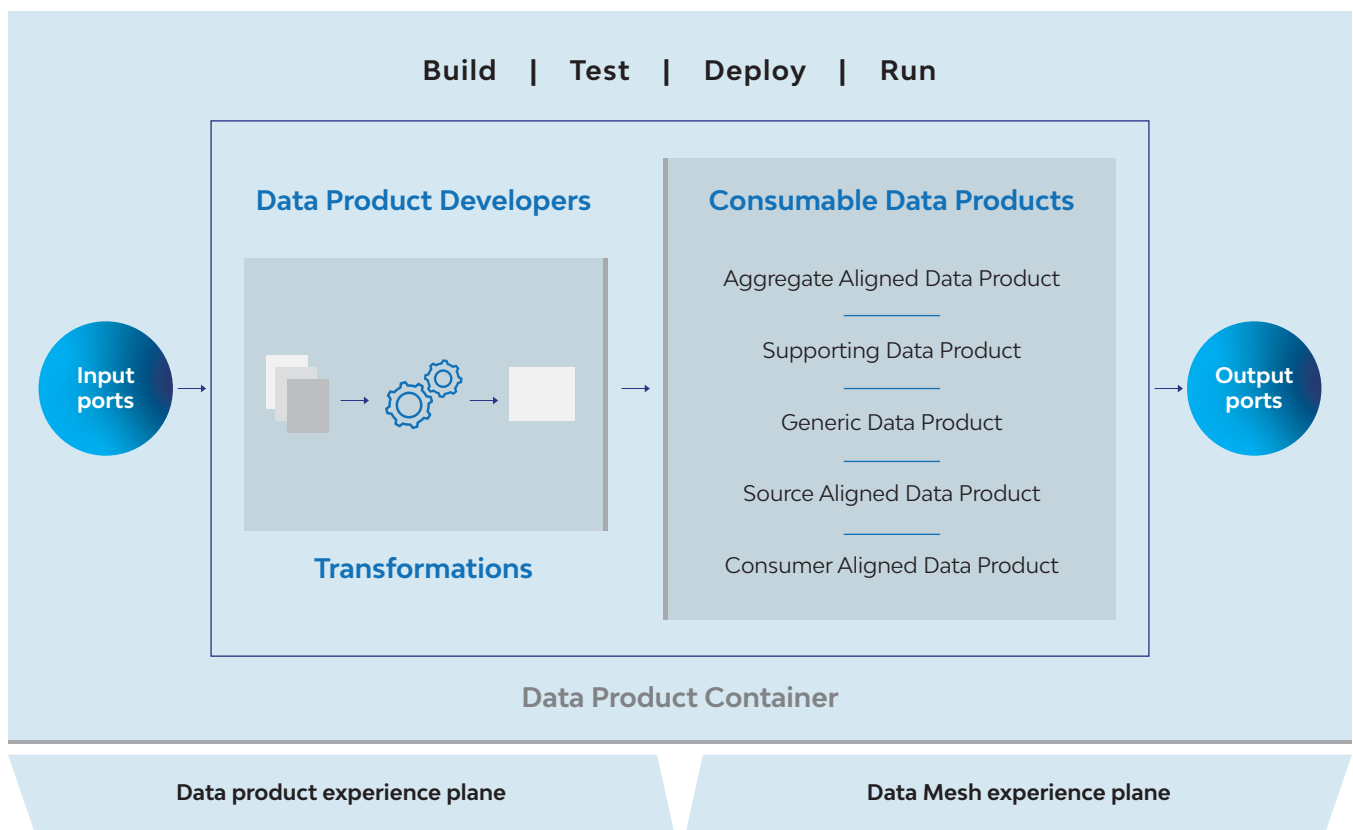
4

Define data product

The final step is to identify and develop data products for each data domain. Data products should imbibe all 8 characteristics as defined in an earlier section.

Operationalizing Data Mesh on Azure

Below diagram shows the logical architecture that comprises different modules of Data Mesh and their relevance within Data Mesh.



Architecture Components

Explanation

Input Ports

Represents ports to pull data from different sources. These sources could be native to Azure or external, e.g., on-premises.

Output Ports

Represents ports to expose/share data products. These can be native, e.g., SQL DB or API based.

Data Product Containe

It encompasses all the structural units required to create data products, such as services to ingest data, transform data and host final consumable data products.

Data Product Experience Plane

This is a higher-level abstraction built using the infrastructure plane to build, maintain, and consume data products.

Data Mesh Experience Plane

A marketplace where consumers can browse and subscribe to data products. This plane will enlist all relevant metadata for a given Data Product.

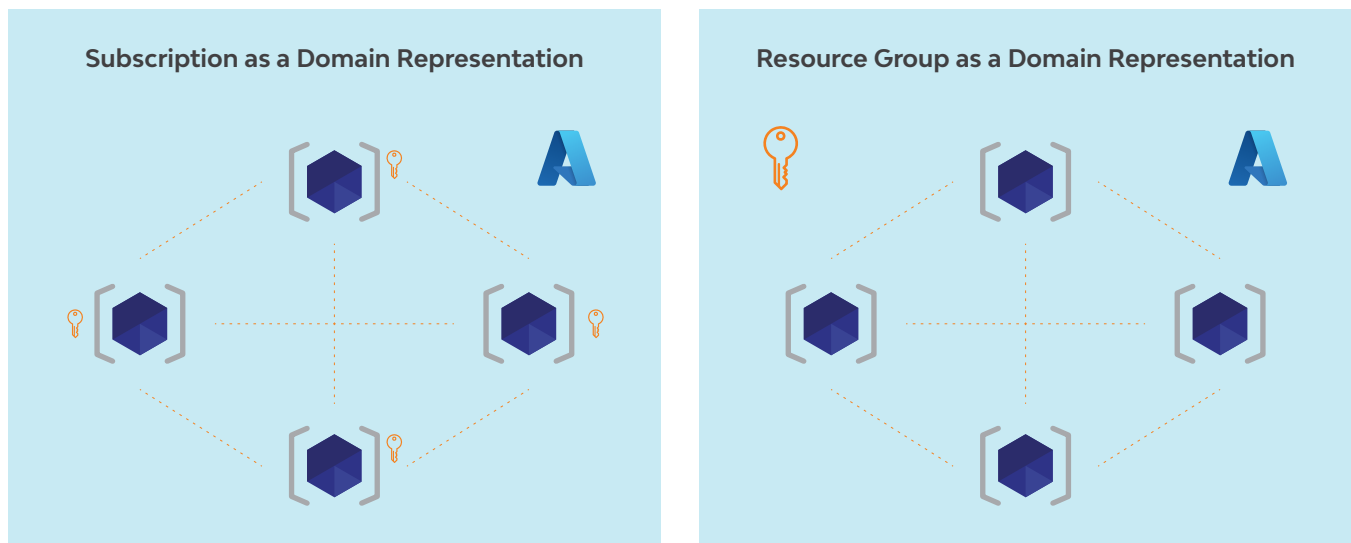
Data Infrastructure Plane

Self-service platform to define, manage and govern Data Mesh.

In subsequent sections, we'll provide an implementation perspective on Azure for each Data Mesh principals.

Domain oriented and decentralized data ownership

Data ownership / independent Domain Nodes can be built-in Azure at the resource group level or subscription level. Based on organization's needs, the domain can be implemented in either way.



Subscription as a domain could lead to org-level management of cost, whereas any lower level than the resource group could result in complex governance and maintenance. Hence, resource group is an ideal choice to be used as a Data Domain.

Note: For further modularity, sub-domains can be created under each domain using VNets. However, it is out of scope for the context of this whitepaper.

Data as a Product

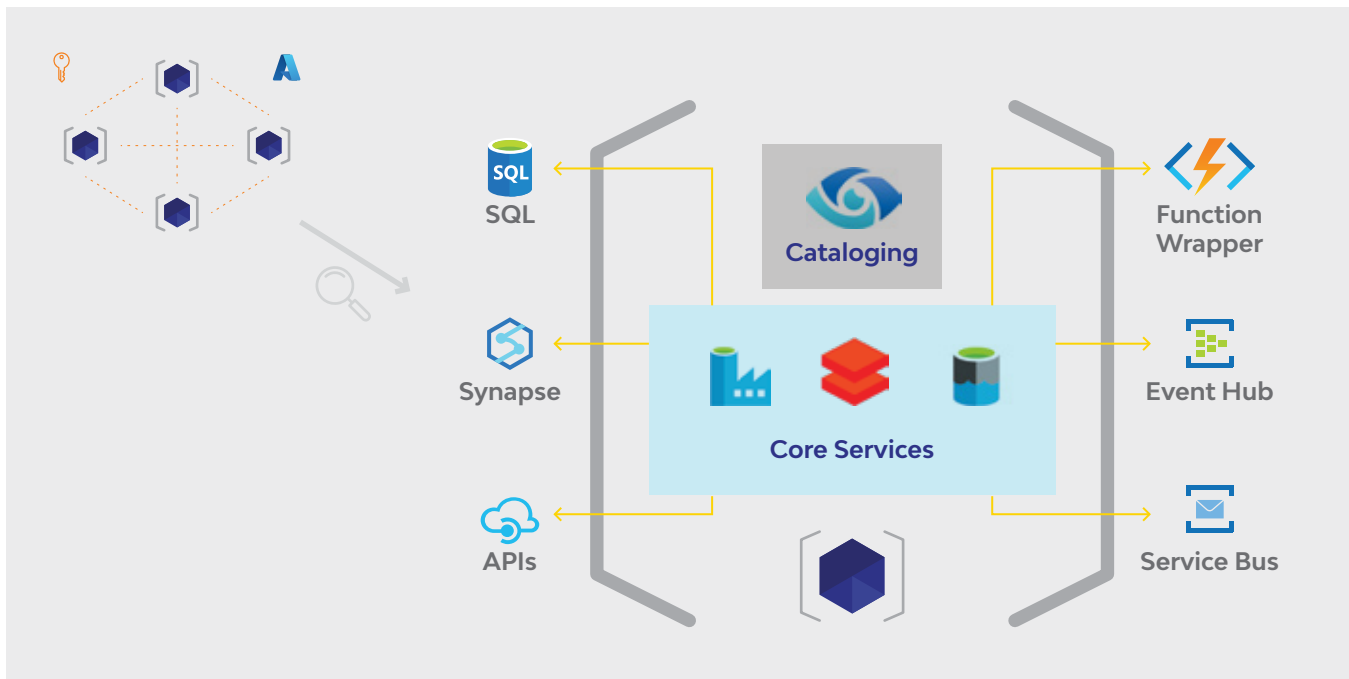
As described earlier, data product quantum refers to all structural units that present a data product with the final aim to produce a consumable data product that can be subscribed to and accessed by data product consumers.

For simplicity, we'll explain Data Product services in 3 buckets:

1. Primary | 2. Secondary | 3. Tertiary

1. Primary:

Represents services that produce final consumable data products that can be developed by developers and accessed by consumers directly. Choice of these services are at the discretion of a given domain.



Each data product will contain core services to ingest and process the data. In addition, mandatory cataloging will be enabled on supported services using Azure Purview.

Ways to consume Data Products can be categorized as:

Edges:

- Services that provide access to Data Products to consumer domains.
- Always available for Data Product consumption.
- Services Like SQL DB, Synapse, APIs can be used as Edges to Data Domain.

Interfaces:

- Services that publish the data to consumer domains.
- Publishes on completion of tasks or predefined events.
- Services like Event Hub, Service Bus, Functions, ADF can be used as Interfaces for Data Domain

SQL endpoint is the preferred method of accessing Data Products due to its industry-wide usage and familiarity, making it easy and faster to consume data as per business needs. In Azure, SQL endpoint can be delivered using the below options.

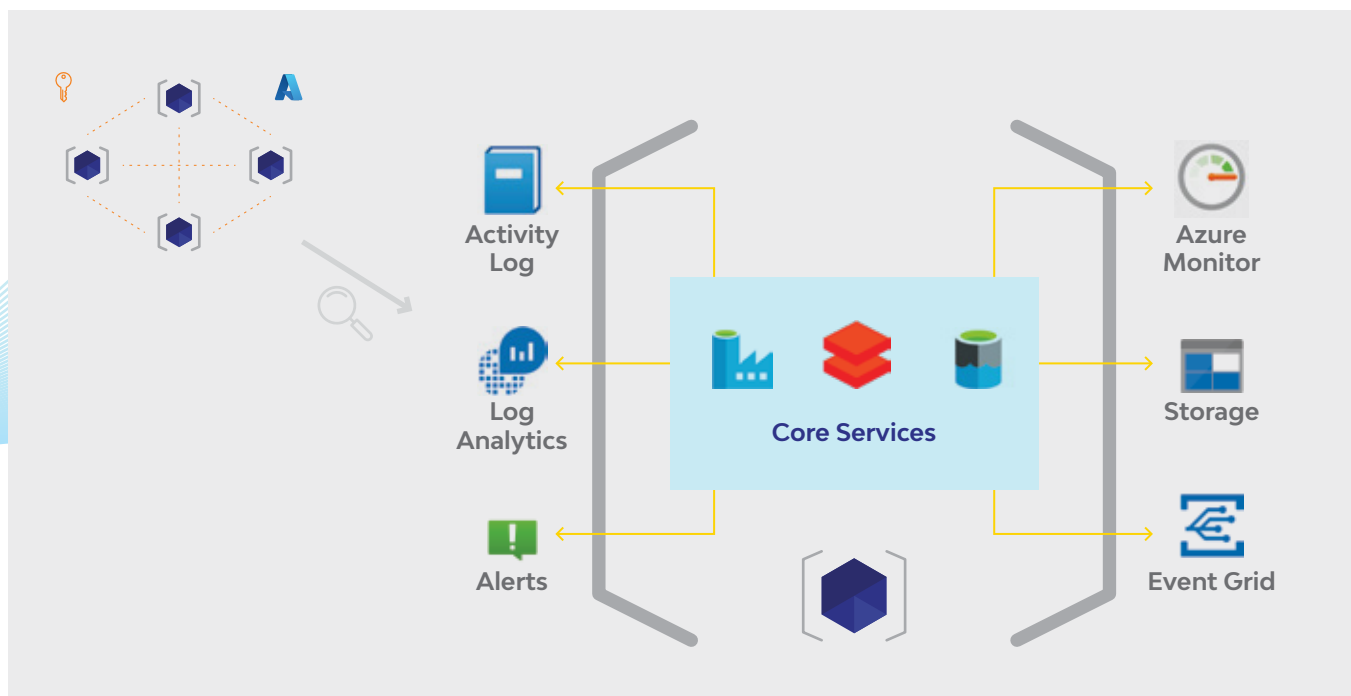
SQL

- 1 Azure Synapse Analytics (**SQL** Serverless or Dedicated SQL Pools)
- 2 Azure Databricks **SQL**
- 3 Azure **SQL** Database
- 4 Azure **SQL** Managed Instance(s)
- 5 **SQL** Server running on an Azure Virtual Machine

Depending on business use cases, different Data Products within a domain can be independently implemented in SQL DB/Synapse, OR as other schemas/databases within a single instance of DB with an API extraction layer to control the access.

2. Secondary:

Mandatory services to facilitate federated computational governance and logging/monitoring.



Edges:

- Services that provide access to Data Products to consumer domains.
- Always available for Data Product consumption.
- Services Like SQL DB, Synapse, APIs can be used as Edges to Data Domain.

Interfaces:

- Services that publish the data to consumer domains.
- Publishes on completion of tasks or predefined events.
- Services like Event Hub, Service Bus, Functions, ADF can be used as Interfaces for Data Domain

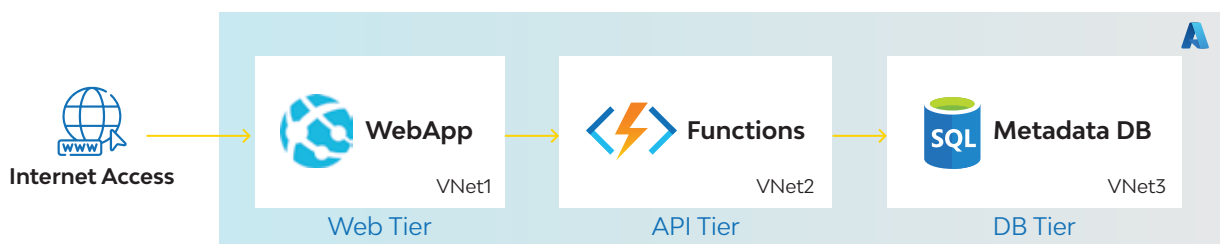
3. Tertiary:

Services that create/manage networks, subnets, and security groups. As the domain itself represents a logical unit, adding network-level complexities could lead to management overhead and a complex self-service platform. And hence, for the context of this whitepaper, details on tertiary services for data products are out of scope.

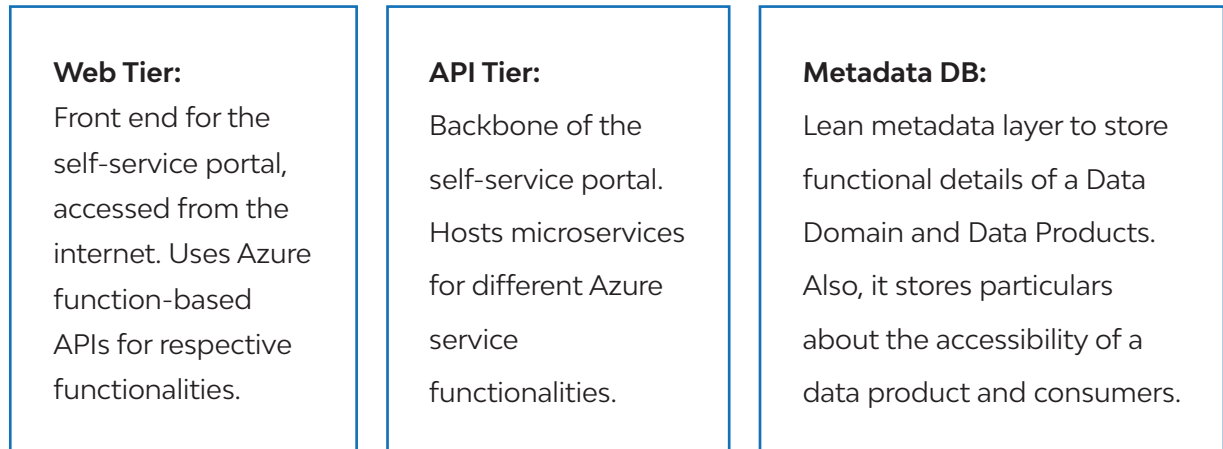
Self-service data infrastructure

Portal Architecture

A self-service platform can be created using preferred technologies like HTML, CSS, Python, etc. However, it is advisable to follow industry-recognized 3-layer architecture to isolate the front, back, and database tiers into different networks. Microservice-based architecture is recommended for better control and reusability. All microservices about a service functionality, e.g., creating a resource or cataloging a DB, can be written in Azure Functions, and exposed as an API to Web Tier for consumption.

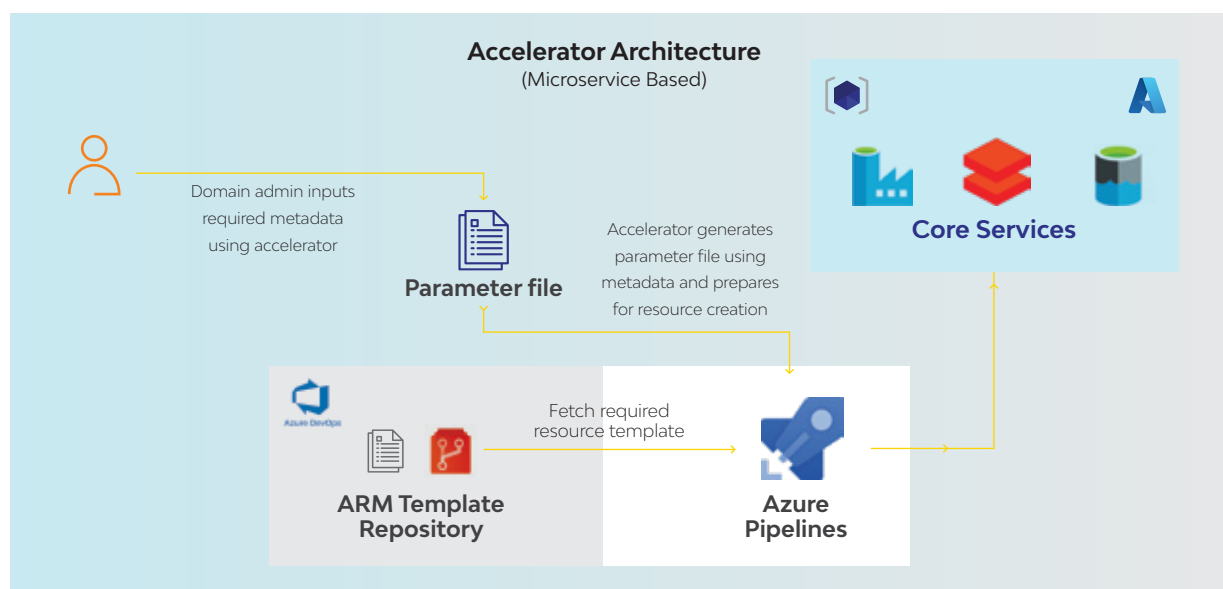


As shown, only Web Tier can be accessed from the public internet. API and DB Tier are isolated from the internet.



ARM template based framework

Creating a generalized framework to execute different Azure service-related requests is necessary. E.g., creating a resource, cataloging a DB, or granting access to a data product. Azure ARM templates are one probable solution to make such a framework. All templates can be stored in the DevOps repository and executed using Azure pipelines and parameters as depicted in the below diagram.



Advantages of DevOps based Framework:



Centralized storage and management of ARM Templates.



Lean and modularize architecture for Accelerator.



Easy to deploy accelerator as PaaS service.



Faster development due to decoupling of accelerator and resource creation processes.



DevOps CI/CD processes can be leveraged for development and enhancement.

User Personas

Depending on the type of responsibilities, users can be categorized into 4 different personas.



Domain Provisioner

As step 1 of the self-service process, the Central IT team provisions Data Domain, enable Azure services access, and enforces governance.



Domain Owner

In step 2, leverages Data Domain and creates vital services. Grants access to developers and oversee performance and cost of Data Domain.



Data Product Developer

In step 3, the developer community that develops data products based on functional requirements.



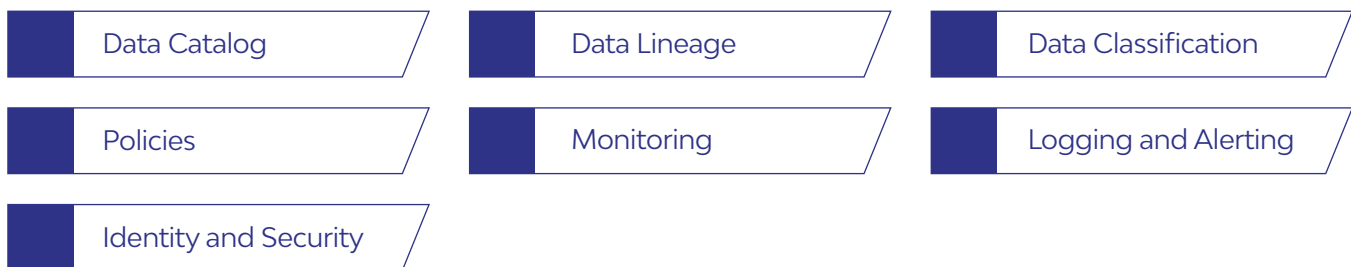
Data Product Consumers

As a final step, consumers that browse, subscribe, and consume data products. Data Domain marketplace is the primary access point for Data Product consumers.

Federated computational governance

For a consistent behavior of the Data Mesh and to be compliant with the organizational policies, it is essential to implement governance in a federated computational manner, i.e., it should be governed in an automated fashion and enforced automatically as per the configuration of policies and scope. Also, governance manages automated metadata capture to facilitate searchability and observability of Data products.

A few essential governance aspects of being considered are:



Data Catalog

Here are the features of a Data Catalog:

- Automatic technical metadata collection.
- Business metadata via predefined tag templates.
- Default all datasets across domains published.
- Domains can restrict based on the sensitivity.
- A local catalog with entire data set attributes can be leveraged by the technical team for analysis and development.
- Enterprise catalog with published data products that are intended for end-user consumption. This catalog will be integrated with the marketplace to provide consumers with user-friendly navigation of published data products.

Azure Purview is a preferred Azure service to implement data cataloguing features.

Data Lineage

Here are the features of Data Lineage.

- Provide end-to-end lineage for given data products.
- It should be easily accessible in a user-friendly way.

Azure Purview is a preferred Azure service to implement the Data Lineage feature.

Data Classification

Data Product attributes should be classified based on organizational data security policies.

Azure Purview is a preferred Azure service to implement Data Classification features.

Policies

For consistent enforcement of data compliance, it is necessary to enforce policies, global and local, on the Data Domain and Data Products. There are 2 steps, policy definition, and policy enforcement. Azure policies can be leveraged. Alternatively, open-source like Open Policy Agent can also be used to create a framework for policy definition and enforcement. The governance committee is primarily responsible for defining and enforcing policies.

There are 3 types of policies that can be considered.

Type 1

Policy related to code.

Example: Table names should be in the upper code.

Enforcement: CI/CD pipeline.

Implementation: DataOps platform.

Type 2

Policy related to Infra.

E.g., Warehouse size cannot be more than XL.

Enforcement: Self-service provisioner.

Implementation: IaaC script.

Type 3

Policy related to Data.

E.g., Emp_Num must be positive.

Enforcement: Data pipeline.

Implementation: DataOps platform.

Monitoring

Azure monitor with built-in/pre-configured charts/statistics can be used for monitoring purposes.

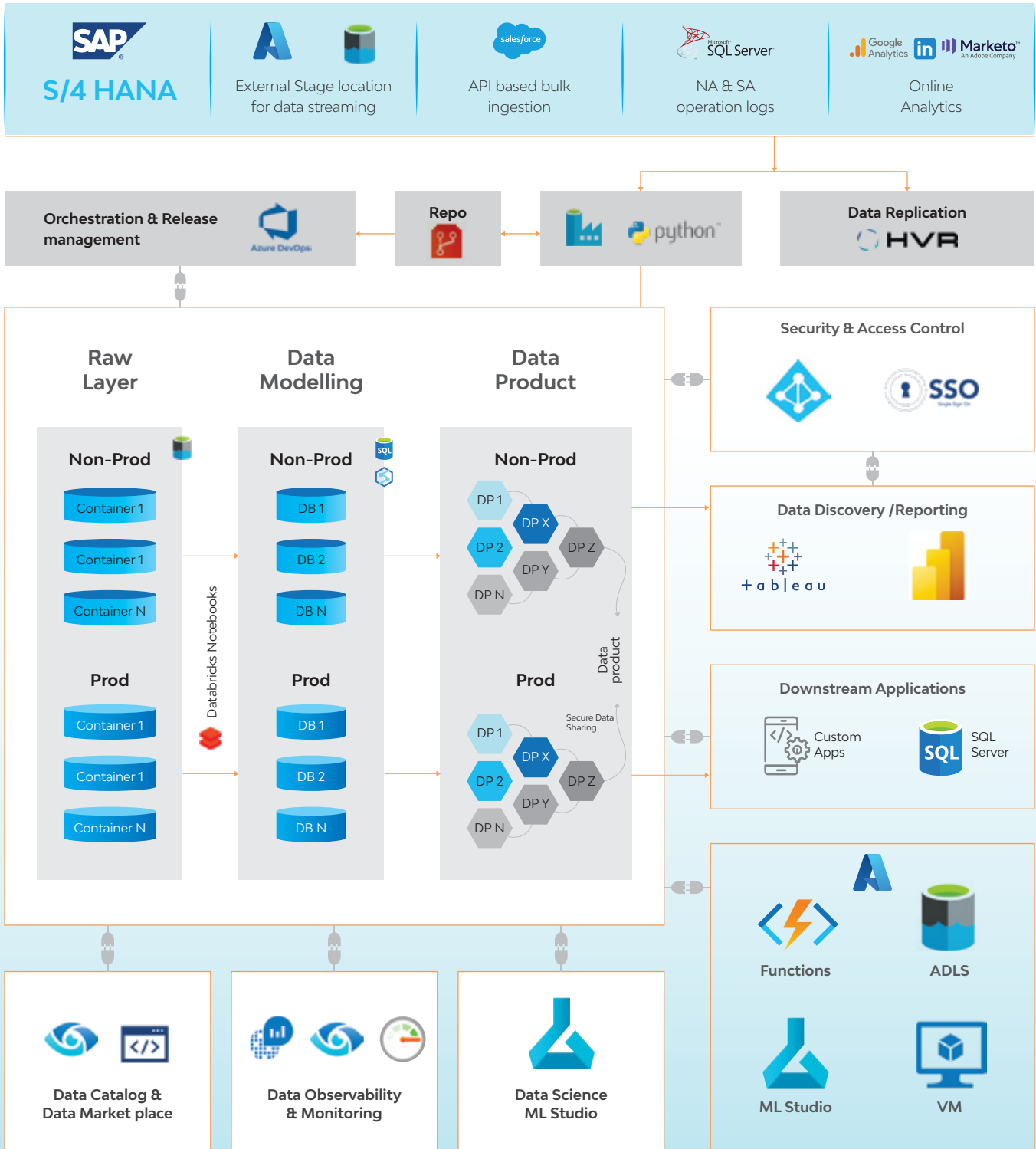
Logging and alerting

Organizational level log analytics service can be set up and mandatory enabled for all domains to collect logs for alerting and operational analytics purposes. Kushto query-based framework can be considered to enable domain-level operational analytics and alerting.

Identity and security

Azure AD should be used as a single service to manage identities and access to the resources. Please note that specific elevated AD access like Directory Reader will have to be enabled for managing resource accesses through a self-service portal.

Data Mesh Conceptual Architecture



Data product marketplace

To successfully adopt Data Mesh, it is essential to create an easy-to-use and intuitive Data Product marketplace for consumers to browse and subscribe to required data products. The marketplace platform leverages catalogs and metadata generated by the self-service infrastructure platform and creates a functional view of underlying data products for easy consumption. It also facilitates the approval request workflow to raise and grant requests for a particular Data Product. Please note that only the enterprise catalog, as described earlier, qualifies and integrates with marketplace. Integrating native catalogs can lead to too many unwanted data products and potentially create confusion and improper data product selection.

Also, the marketplace should track and enlist statistics around 8 properties of a data product, as mentioned in an earlier section. E.g., it should capture the number of subscriptions, consumers, and access to a data product to understand the value and discoverability of that data product. Data products that are not used or have poor user ratings could be considered for enhancement or decommissioning.



Search

Provide search based on keyword, domain etc.



Explore

Bookmarks, download, favorites.



Access

Ability to request and grant access.



Collaborate

Ability to share and publish data products.



Personalize

Recently viewed, recommendations for you.



Trust

Certification for the quality & authenticity of data products.



Preview

Ability to preview data products and features.



Insights

Track and present data product performance.










Conclusion

Data Mesh is a new way of creating and sharing 'Data as a Product' by decentralizing data ownership and accountability. However, extensive research, thinking and decision making are required to define proper data domain strategy as a pre-requisite to Data Mesh implementation. Next critical business task is to define trustworthy and accurate Data Products that can be easily discovered and shared across organization.

Important yet complex task is to create self-service data infrastructure that executes technical requirements in the form of Data Mesh functionality. Azure with its wide range of services enables simple and effective implementation of Data Mesh in cloud. LTI's Azure Data Mesh accelerator simplifies the centralized platform infrastructure and integrates federated computational governance in easy-to-use UI portal, therein reducing overall cost and efforts to implement Data Mesh.

References

- Q Data Mesh – Delivering Data-Driven Value at Scale (by Zhamak Dehghani) 
- Q <https://mrpaulandrew.com/> (Article by Paul Andrew) 
- Q <https://docs.microsoft.com/en-us/azure/purview/> 
- Q <https://docs.microsoft.com/en-us/azure/governance/policy/> 
- Q <https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/> 
- Q <https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops> 
- Q <https://www.openpolicyagent.org/> 



About the Author



Abhishek Patel

Principal - Data Engineering
LTI

Abhishek has over 21 years of experience helping customers build data platforms and analytical systems. He has architected multi-year large-scale projects on data modernization and migration to cloud data platforms like Azure and Databricks. His expertise lies in data modernization & cloud transformation for customers across banking and financial services, manufacturing & engineering, and healthcare industries, with core competencies in data strategy development, architecture design, application development, data warehousing, data integration, and cloud modernization. He holds a bachelor's degree in Computer Science.



LTI (NSE: LTI) is a global technology consulting and digital solutions Company helping more than 475 clients succeed in a converging world. With operations in 33 countries, we go the extra mile for our clients and accelerate their digital transformation journeys. Founded in 1997 as a subsidiary of Larsen & Toubro Limited, our unique heritage gives us unrivalled real-world expertise to solve the most complex challenges of enterprises across all industries. Each day, our team of more than 40,000 LTItes enable our clients to improve the effectiveness of their business and technology operations and deliver value to their customers, employees and shareholders. Find more at <http://www.Lntinfotech.com> or follow us at @LTI_Global.

info@Lntinfotech.com



A Larsen & Toubro
Group Company