Eclipse Dataspace Connector

Peter Koen
Cloud Standards Architect

Microsoft Corporation
Corporate Standards Group

peter.koen@microsoft.com




Digital Sovereignty

A goal of Digital Sovereignty is autonomy, which is different from independence – it means **acting with choice**. It includes the control over when and where data is stored and how it can be accessed.

The goal of the EDC is to enable a federation of organizations to leverage their collective investment in data while maintaining their autonomy. To achieve that, each participant agrees to governing rules of behavior while avoiding centralized controls or technologies.

Why do we need a decentralized design?

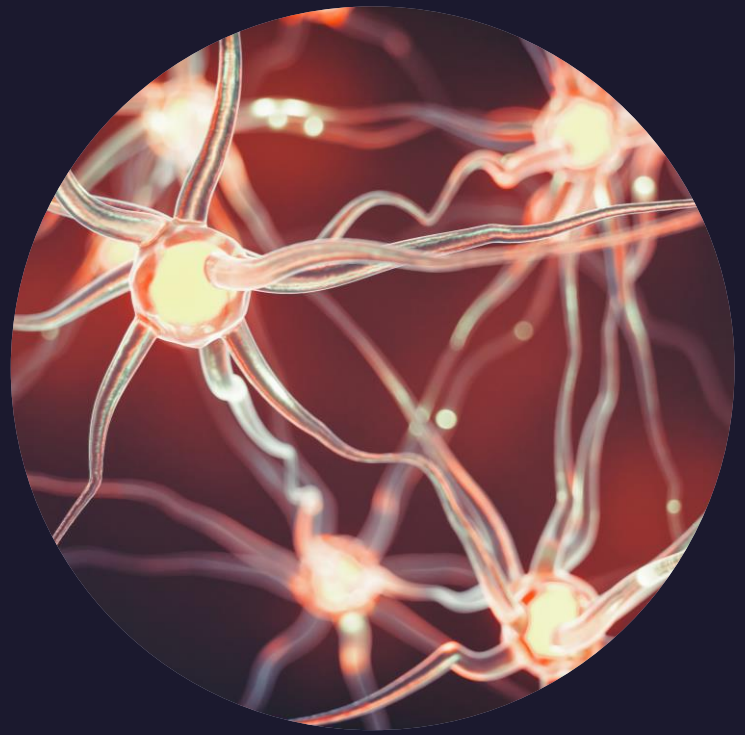
1. Participants of a dataspace must have **control** over which data they share with whom
2. Autonomy starts with controlling **identity**, if you are not in control of your identity you can't act autonomous
3. Participants need to decide who they **trust** on a case-by-case basis
4. Participation in a dataspace must be based on **rules** that apply to everyone
5. No central control system can make arbitrary **decisions** on individual participation
6. Decentralized systems are **resilient** and provide higher **availability**
7. No central system that holds the keys to the entire federation - to improve the **security**
8. **Interoperability** of heterogeneous environments with many different technologies and operating models
    - On Premises, Edge, Hyperscale Cloud, Multi-Cloud
9. Transitive trust based on common **trust anchors**

Further Aspects of Decentralization

- Low Cost – every participant is responsible for their own infrastructure to enable participation in the system – no need to pay for a very costly central system. Cost grows exponentially with centralization.
- Autonomy - Control over your participation – but keep in mind: this also includes full responsibility for High Availability and Disaster Recovery (however, this can be optimized for the needs of the individual participant and is not being enforced upon all at the same level through a central authority)
- Autonomy – Control over Information that you are sharing – if you want to be forgotten and your data should no longer be available in the system, just stop your node, which removes it from the federation.
- High Performance – event-based architecture enables independent work without blockers from other actors (central systems or other participants)

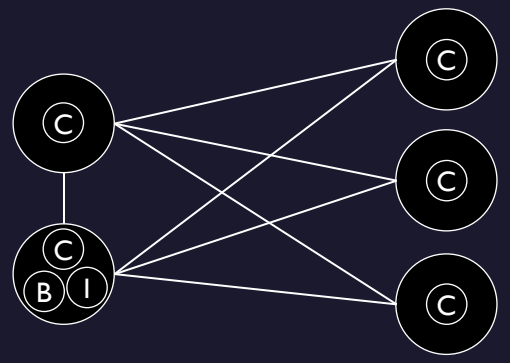
Dataspaces

Multi-Cloud Federations

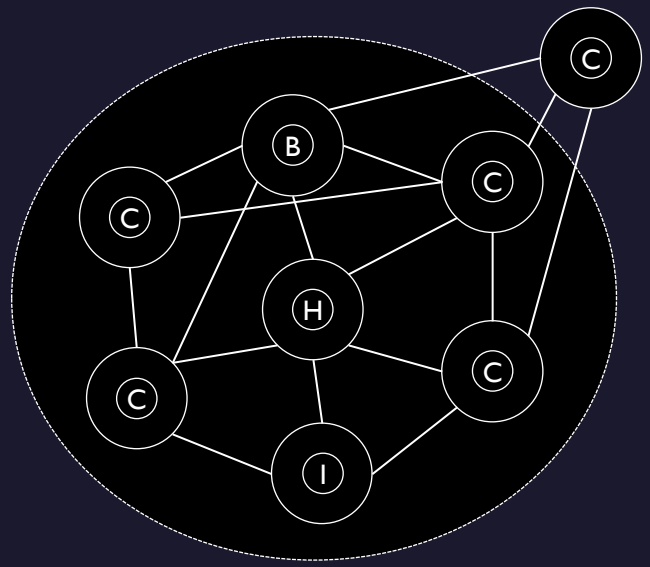


Change of Data Sharing

Bilateral data exchange

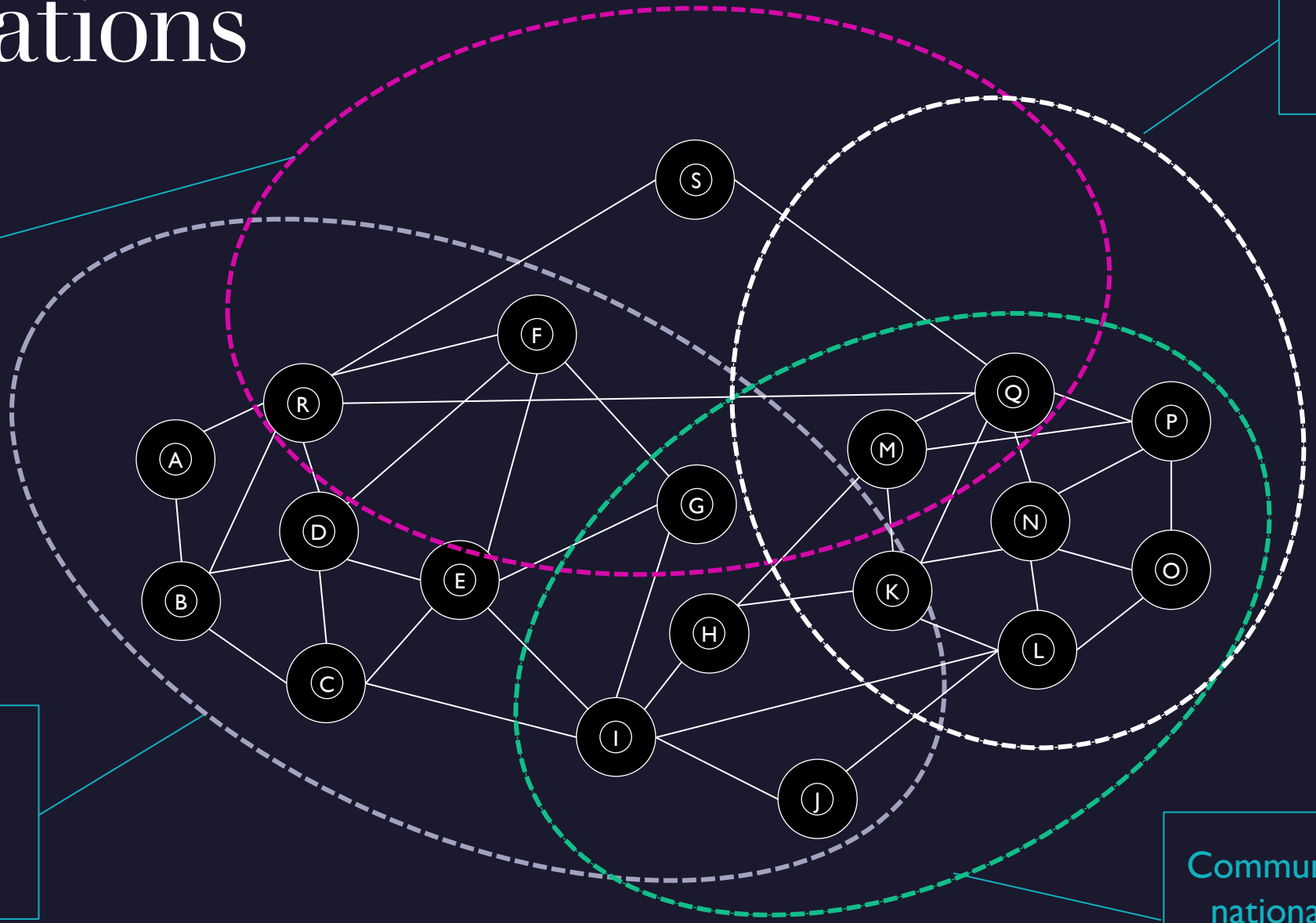


Closed group data exchange



Open and dynamic data exchange

Federations



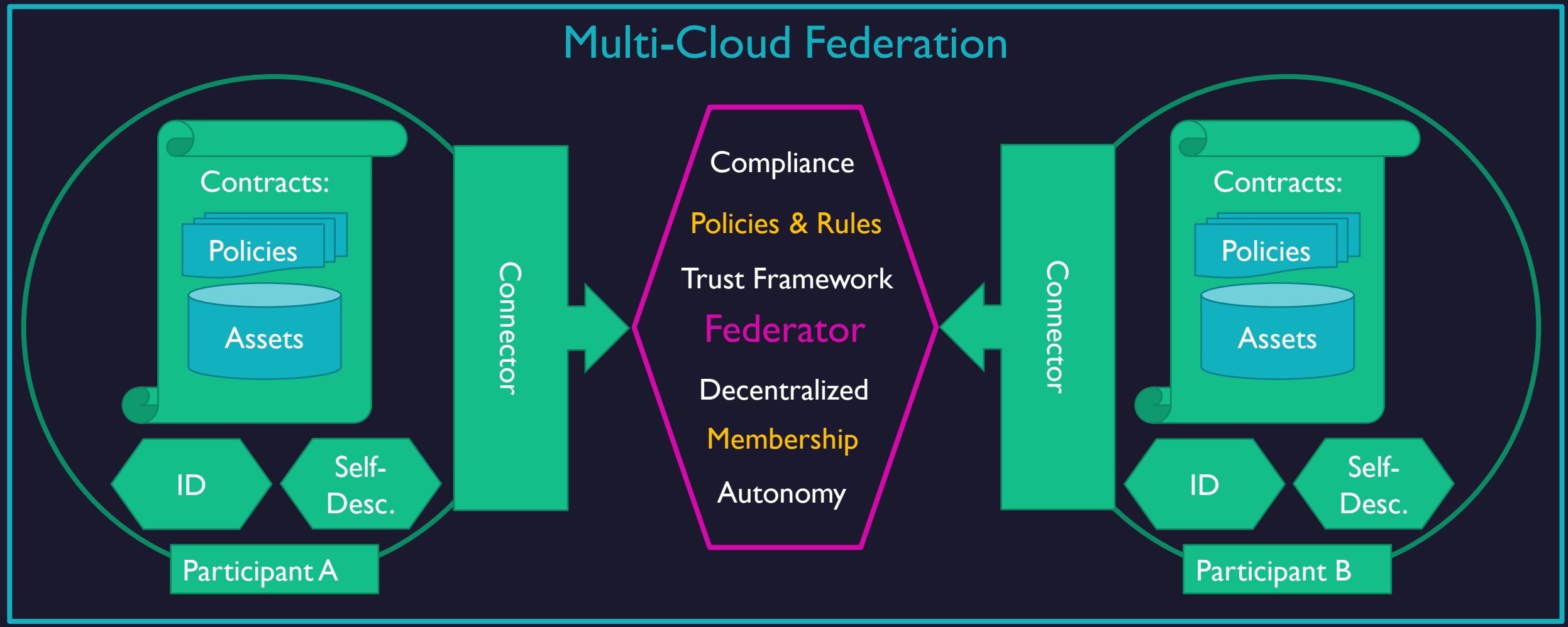
Jurisdictional Community

Private Community

Industry Group

Community around a multi-national company and its supply chain

Participating in a Federation



“The way to get started is to quit talking and begin doing.”

- Walt Disney

EDC Overview

The essential information of the project

EDC Quick Facts

Open Source

Started in June 2021
Official Eclipse Foundation project
Apache 2.0 License

Community

8 Committers
30+ Contributors
>250 pull requests

Technology

Based on Java
Extensible Architecture

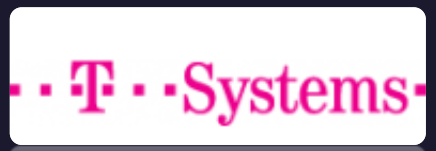
Enterprise Ready

Scalable
Highly Available
Customizable

Current State

Runnable Code
Basic End to End Sample

Contributors, Committers & Partners



Alignment with Gaia-X

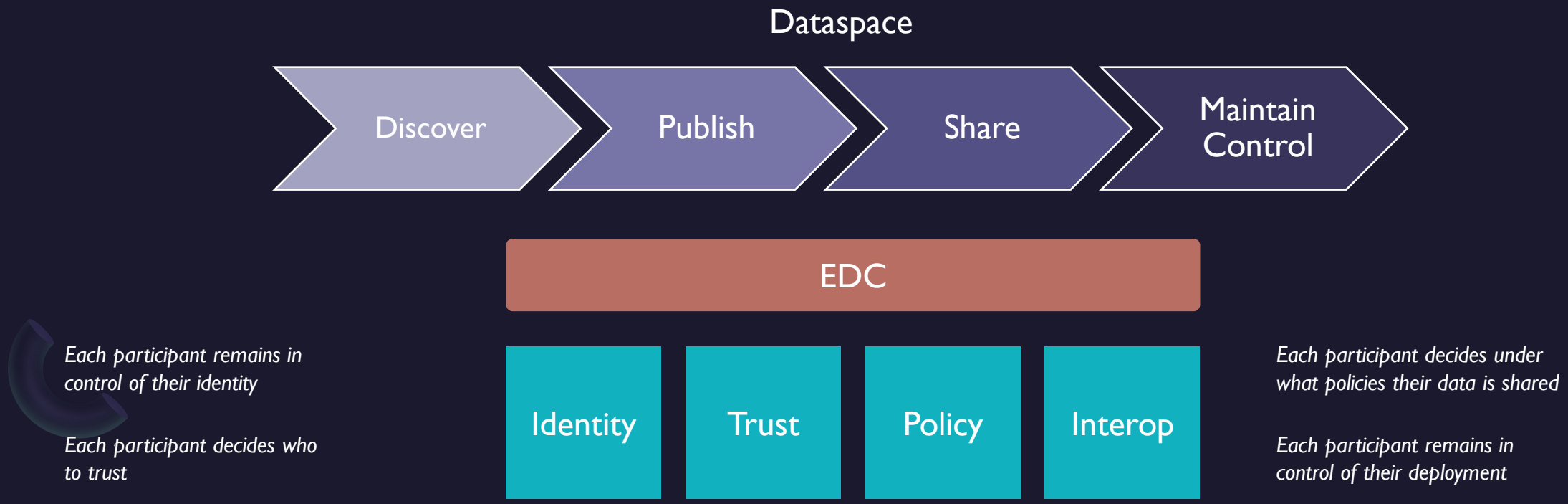
- Fulfillment of the mandatory and further criteria
  - Support for Gaia-X Trust Framework
  - Support Gaia-X compliant Self-Description
  - Gaia-X Registry Extension and support for VC
  - Currently planning for alignment with Gaia-X Federation Services v2, pending finalization of specification
- Possible Integration with GXFS-DE implementation project
  - Evaluated existing specifications
  - Waiting for tangible code to evaluate integration
- Alignment with lighthouse projects to meet their requirements for building economically viable dataspaces and using Gaia-X

Alignment with International Data Spaces Association

- Already supporting IDS-based messages and policy definitions
- Support development of IDSA Reference Architecture Model 4.0
- Part of IDS Open Source Landscape
- Participating in IDSA committees and working groups
  - Architecture WG
  - Rule Book WG

Pillars of a Dataspace

- A dataspace is a way for organizations to securely share data with other participants.
- Dataspaces are built on *identity*, *trust*, *policy*, and *interoperability*
- Dataspaces enable data cooperation in a multi-cloud federation



Dataspace Characteristics

- Connections in a dataspace are always peer-to-peer
- Multiple participants can cooperate, but data is always exchanged 1:1
- Dataspaces enable a multi-cloud federated environment
  - On Premises, Edge, Hyperscale Cloud, Multi-Cloud
- Participants can have multiple roles
  - Data Owner, Data Holder, Data Processor, Data Recipient, Algorithm Provider,…
- Patterns
  - Aggregator – combining data from multiple sources for computation at one partner (Specialization: Data Trustee)
  - Supply Chain – data moves through multiple participants adding value along the way with potential aggregation, anonymization,…
  - Code to data – code packages can be transferred by EDC to where data resides, execution controlled through policies and custom extensions

Decentralized Dataspaces

DATASPACE AUTHORITY

- Defines the rules of the dataspace
- Acts as root of trust
- Anchors trust in Trust Frameworks
- Has an Identity
- Has a self-description
- Issues Verifiable Credential of membership
- Provide a member registry

CONNECTOR

- Connects 2 participants
- Provides Control over data sharing
- Enforces Policies
- Orchestrates Data Transfer

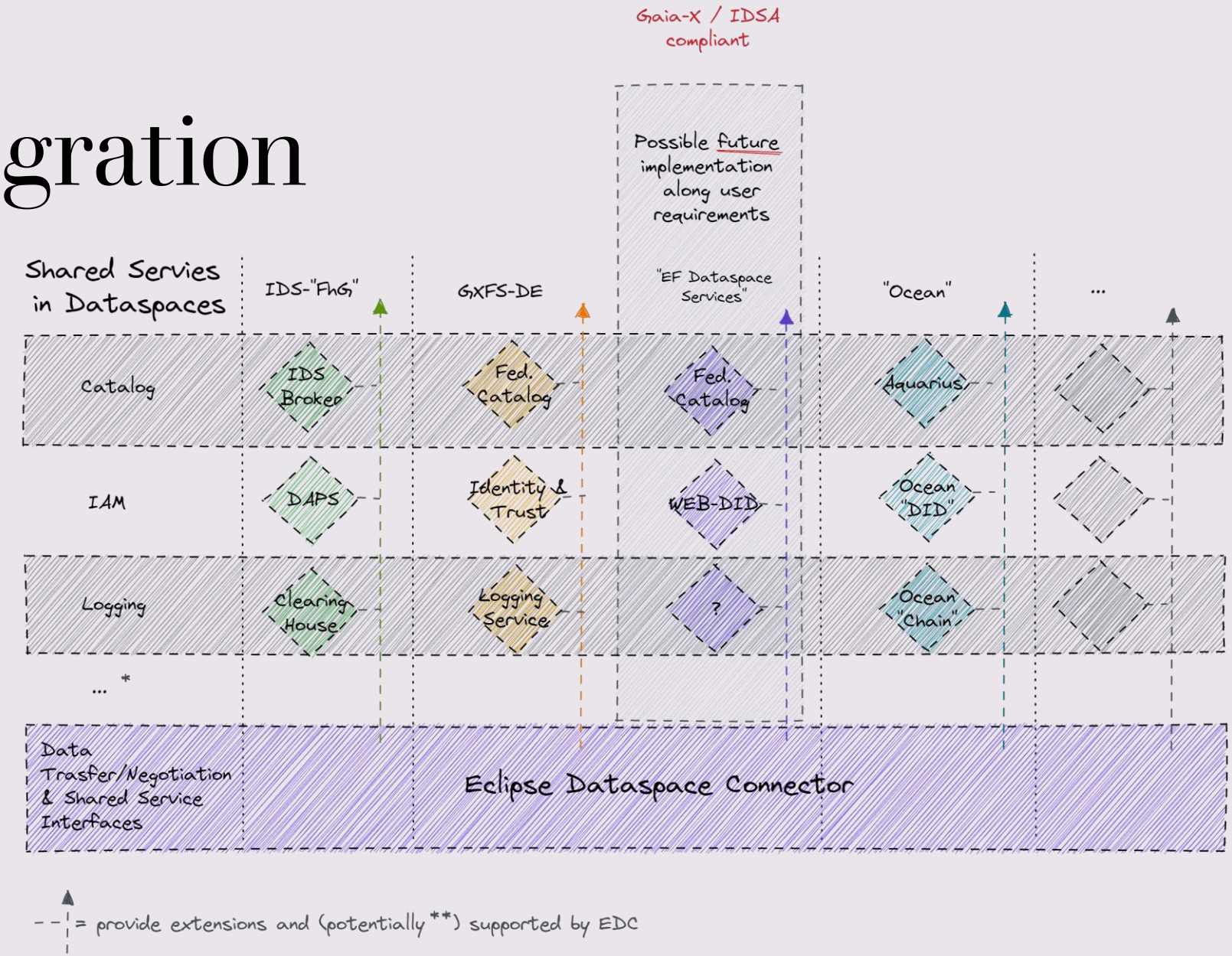
CATALOG

- Provides Data Contract Offers
- Federated Catalog, each participants holds their own catalog entries
- Filtered by Access Rights based on Verifiable Credentials
- Crawler scans for catalog entries of other participants
- Highly customizable and filterable

Ecosystem Integration

- EDC has a flexible, modular system
- Modules can be exchanged
- Custom modules can be created
- Existing modules can be extended
- Can be fully decentralized

  or partially centralized



EDC Architecture

Concepts & Technical Foundations

EDC Architecture & Components

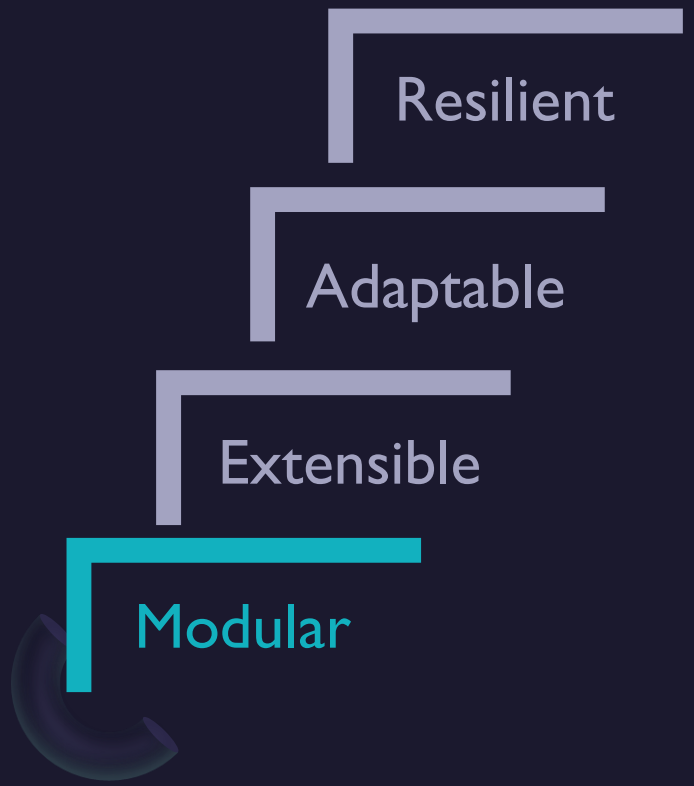
ARCHITECTURE

- Separation of Control and Data Plane
- Extensible through Java SPI
- IDS based communication standard
- Acts as an orchestrator for data transfers
- Asynchronous processing for maximum scalability
- Decentralized Identity Management with customizable Trust Anchors

COMPONENTS

- EDC Control Plane
- EDC Data Plane
- Federated Catalog Node
- Federated Catalog Crawler
- Policy Management
- Data Asset Management
- Identity Hub

EDC architectural principles



- Written in Java
- All functionality is contributed as a module
- Lightweight, composable runtime
- Minimal dependencies

EDC architectural principles

- Defines extension points for all features
- Swap implementations, e.g., database, security
- Create your own features and capabilities

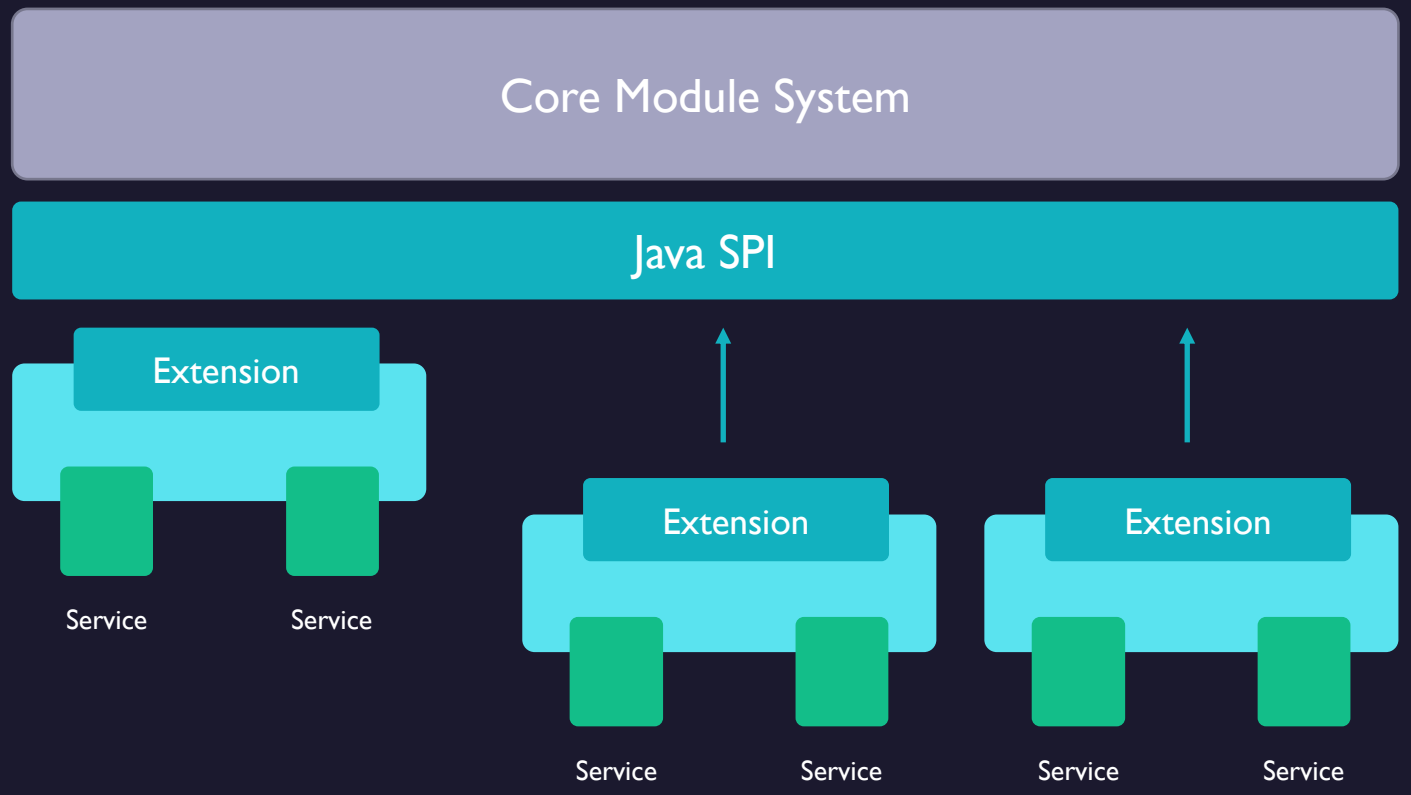
EDC architectural principles

- Deploy to diverse environments
  - Cloud, on-premises, edge
- Deploy with different capabilities
- Scales up and down

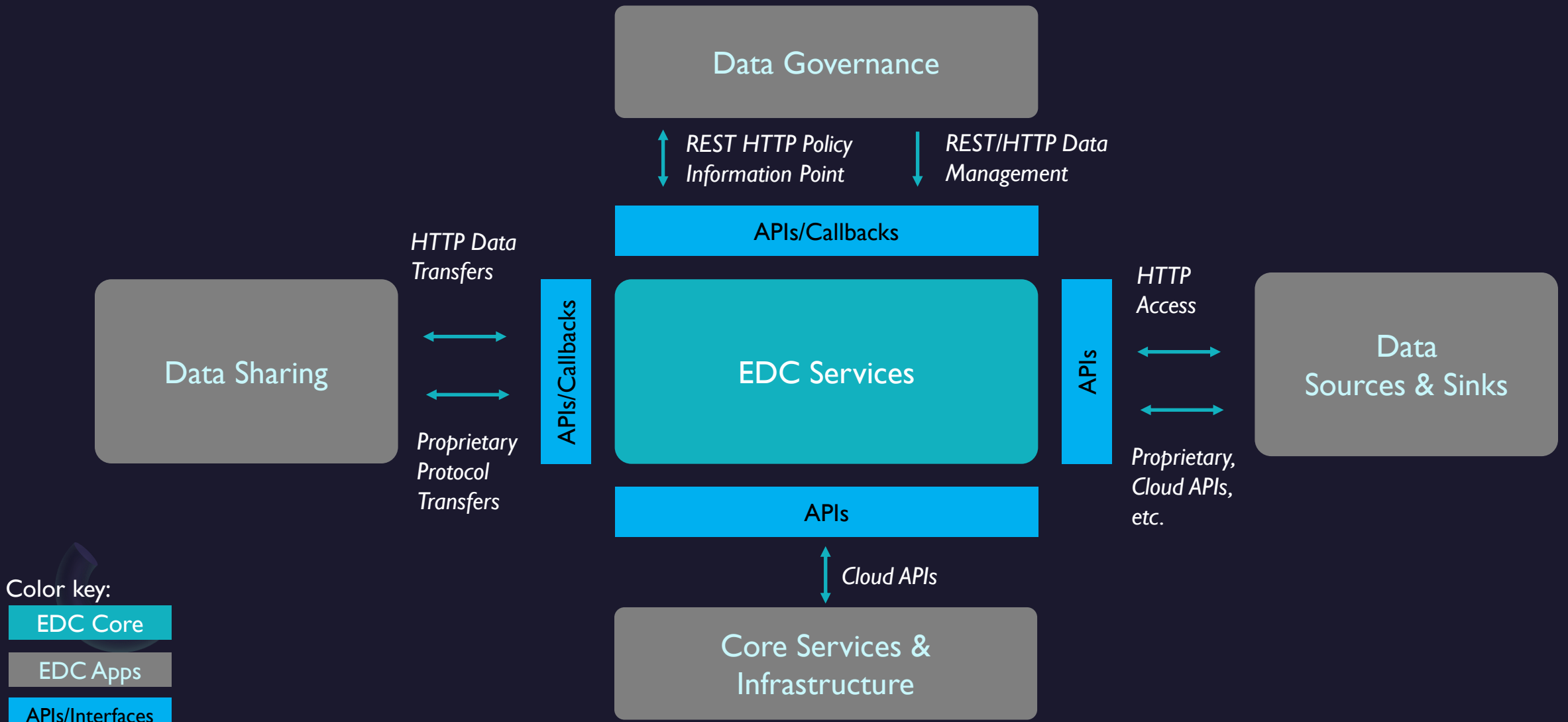
EDC architectural principles

- Leverages high-availability infrastructure that you have already invested in
  - Cloud services
  - Data storage
  - Data transfer technologies

EDC design: The Foundation



EDC APIs and Callbacks (EDC core vs. EDC applications)



EDC design: Dataspace Services

Registry

Registration and discovery

Core Module System

Catalog Services

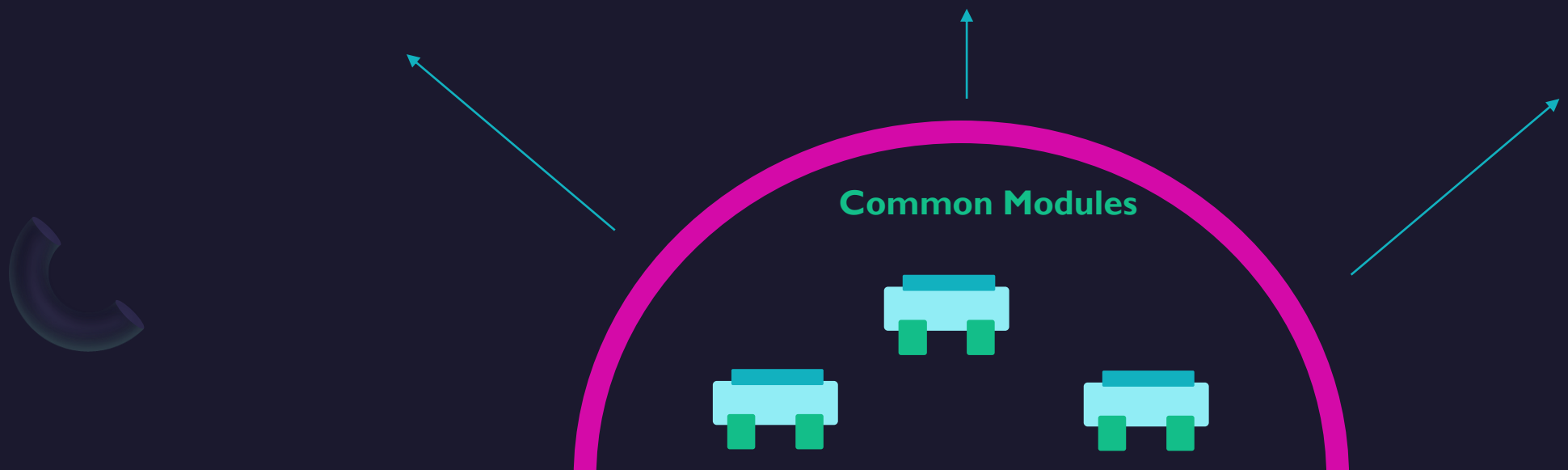
Publish and search

Core Module System

Connector

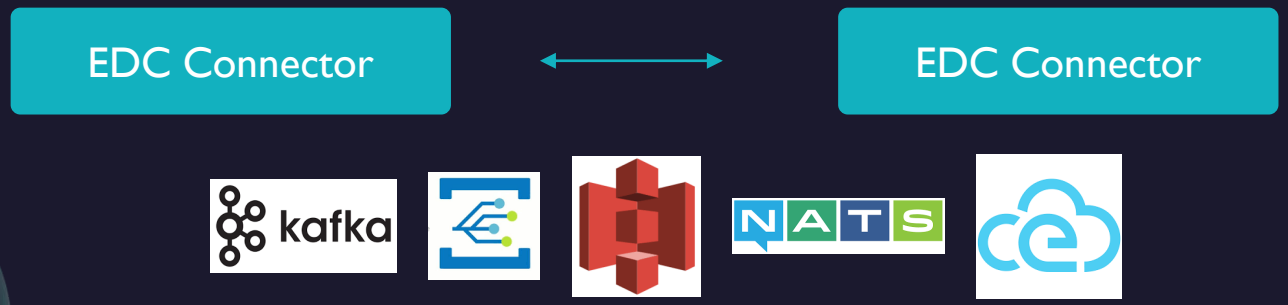
Contract negotiation and data sharing

Core Module System



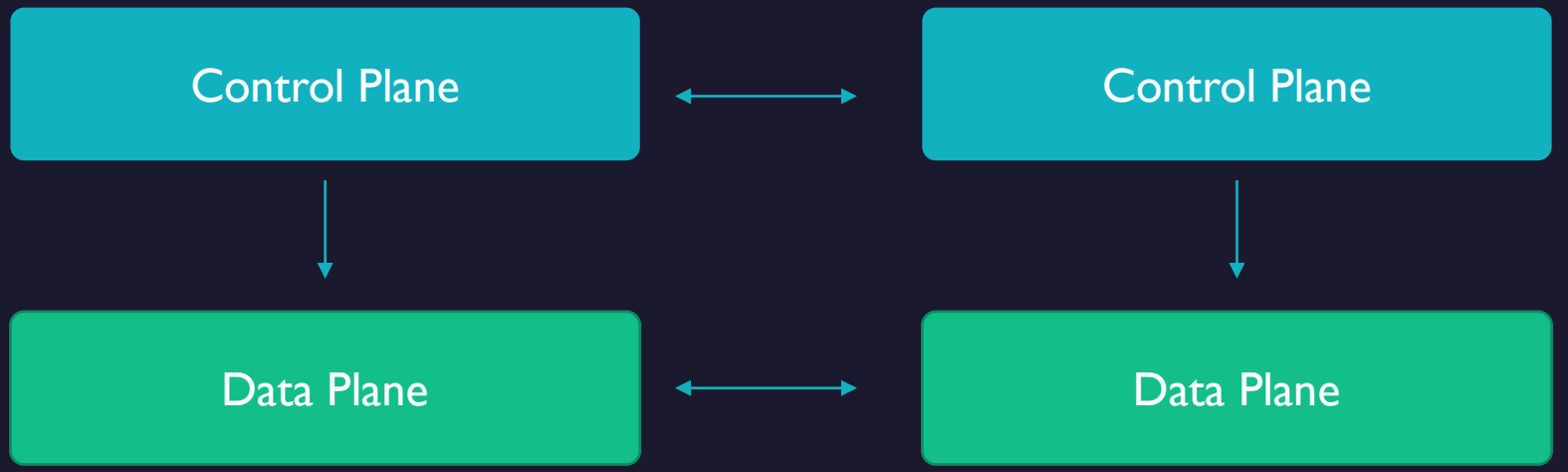
EDC: The Connector

Sharing Data



Control Plane and Data Plane

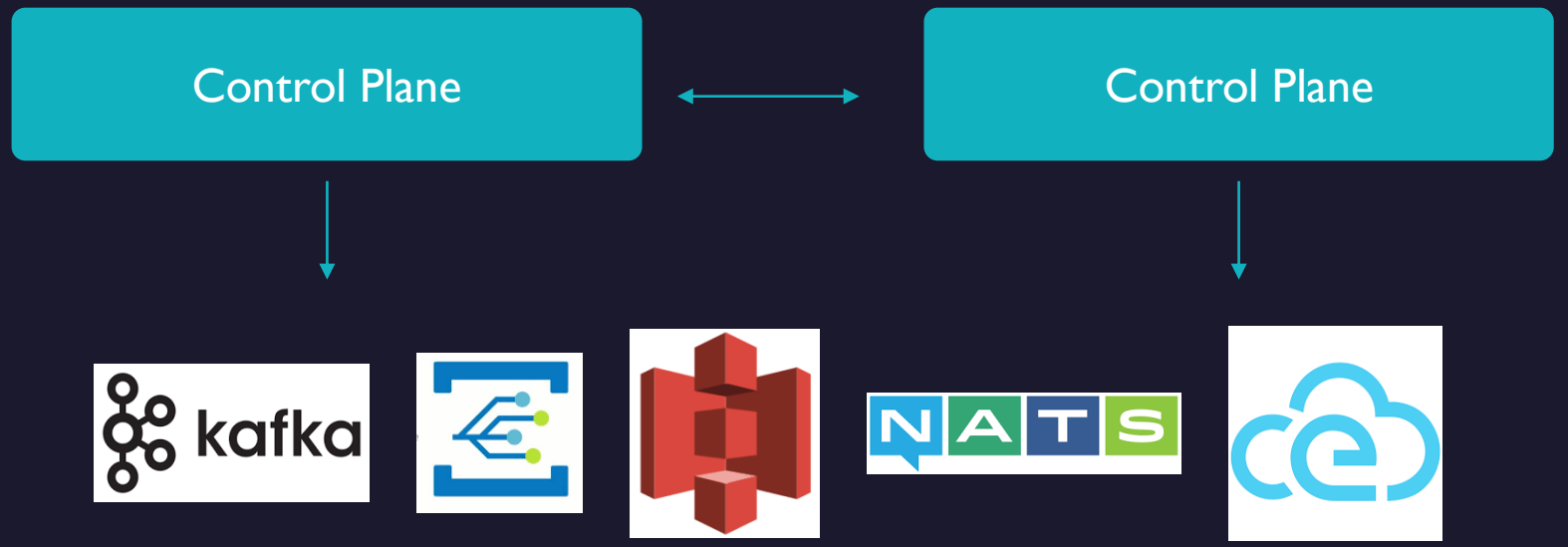
- The Connector is divided into two logical subsystems, a **control plane** and a **data plane**



- Verification
- Contract negotiation
- Oversee policy enforcement
- Manages provisioning

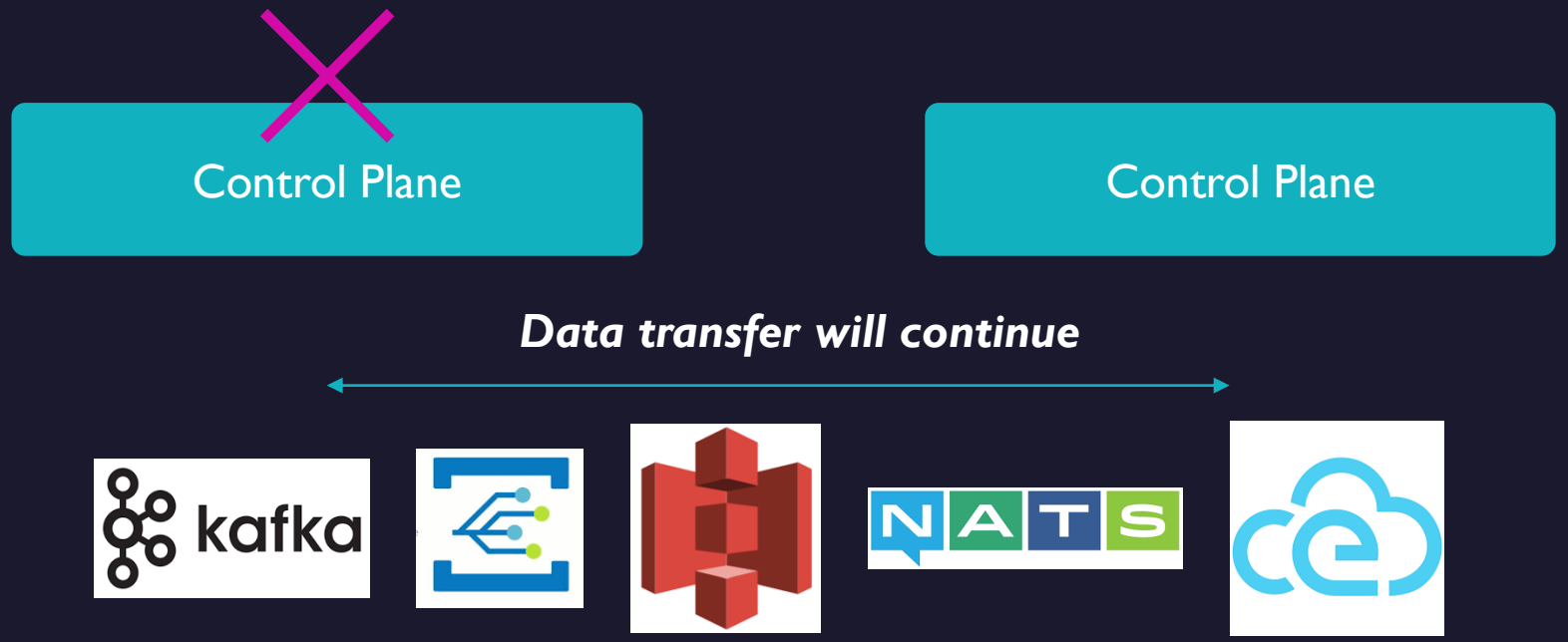
- Moves bits
- Big Data
- Streaming
- Events

Leverage existing infrastructure

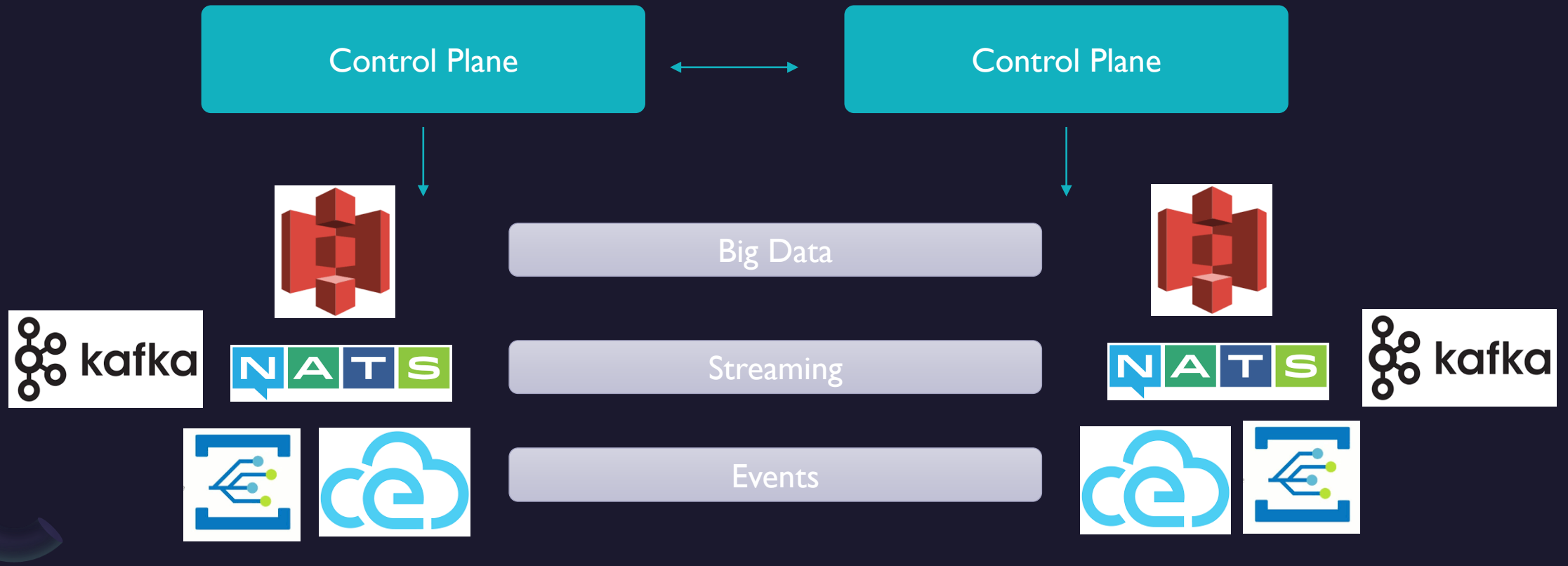


Examples of data technology: Streaming: Apache Kafka, Eventing: Azure Event Grid, Storage: AWS S3, Edge Connectivity: NATS, Eventing: CNCF cloudevents, and many more are possible!

High Availability

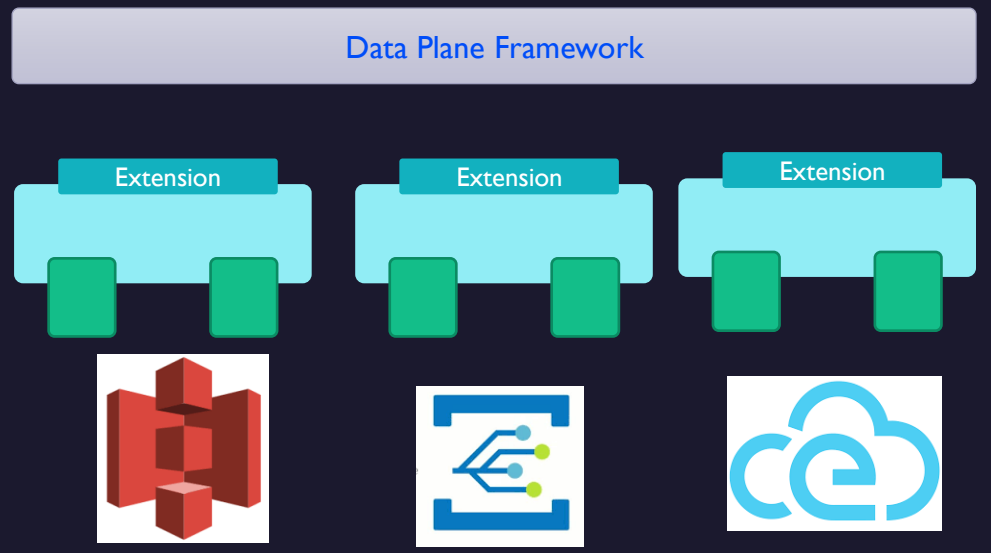


One size does not fit all



The Data Plane Framework (DPF)

- A dynamic routing data plane
  - Optimized for big data and eventing
  - N-way transfers
- Built on the EDC foundation
- Initial release in Milestone 2



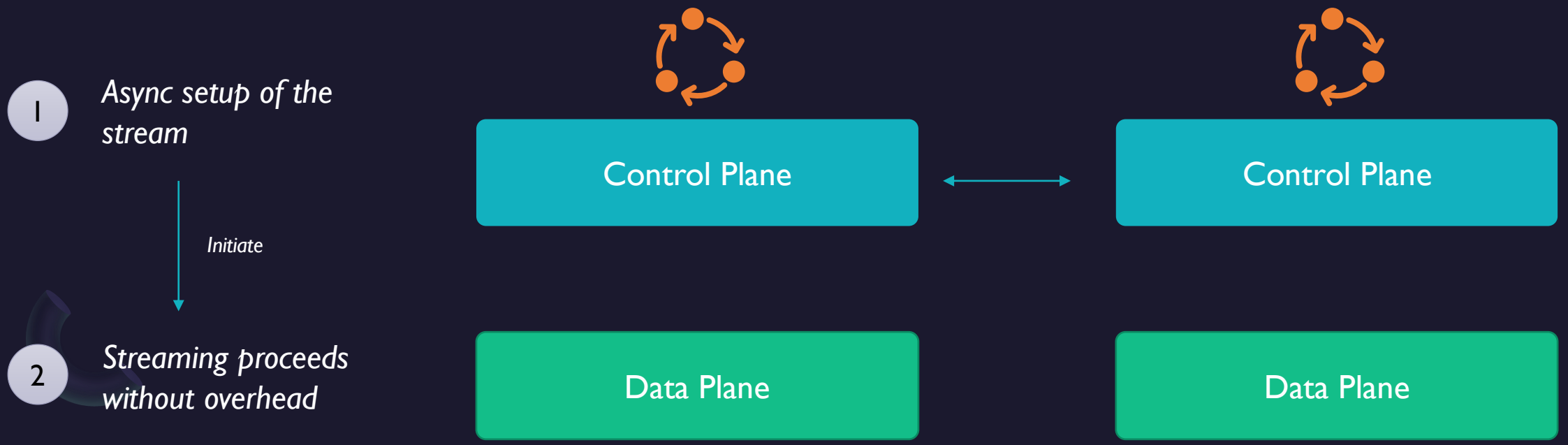
The Connector: Asynchrony

- The Connector is an asynchronous system
- Requests asynchronously transition through a series of predefined states on the client and provider connectors

- Allows for potentially lengthy data preparation
  - Infrastructure provisioning
  - Policy provisioning
  - Data pre-processing
- Reliability through idempotent retries
- High availability via clustered workers
- Natural throttling
- Fairness by distributing processing across states

Data Transfer Performance

- The data plane is separate from the control plane!
- Let's see how this works with streams…



Policy Enforcement – Controlling your Data

"These assets need to be deleted after 7 days"

"Data can only be used for AI training"

Policy Engine

"Data must remain in Austria"

"Data needs to be stored in a C5 compliant cloud environment"

"These assets should only be shared with my partners"

"Data cannot be used for ML Application"

"Data needs to be anonymized before further sharing"

Operating Dataspaces

Designing and operating a decentralized federation of data services leveraging EDC creates a dataspace. This model is extensible at any time to include further capabilities and functionalities. It enables governance through policies while granting full autonomy to all participants without operating central runtime components.

Establishing a dataspace

Identity

Establish a DID
Define DID Method
Define Roots of Trust
Trust Framework
DID Document points to Self-Description

Self-Description

Attributes of the Dataspace
Policies and Rules
Location and Type of Member Registry

Member Registry

List of Member DIDs
Revocation List

Creating a dataspace

WITH CENTRAL DATASPACE AUTHORITY

- **“Association Model”**
  - Ideal for association with a real-world membership organization envelope for the dataspace
- Establishes a DID
  - Anchored in a Trust Framework (e.g. Gaia-X)
- Creates Participation Rules
  - Expressed as Policies
- Publishes Self-Description
- Manages Participant Registry

100% DECENTRALIZED DATASPACE AUTHORITY

- Requires at least 3 members
  - 2 would be a direct contract, not a dataspace
  - 3+ needed to build a quorum and ensure agreement on rules and provide verification of dataspace identity
- Verifiable Credentials of membership needs to be signed by x amount of members
- Member Registry and Dataspace Rules get replicated among members
  - Potentially blockchain type solution
  - Also possible in an implementation like DNS

Joining a dataspace

Identity

Establish a DID
DID Document points to Self-Description
Submit DID to Dataspace Authority

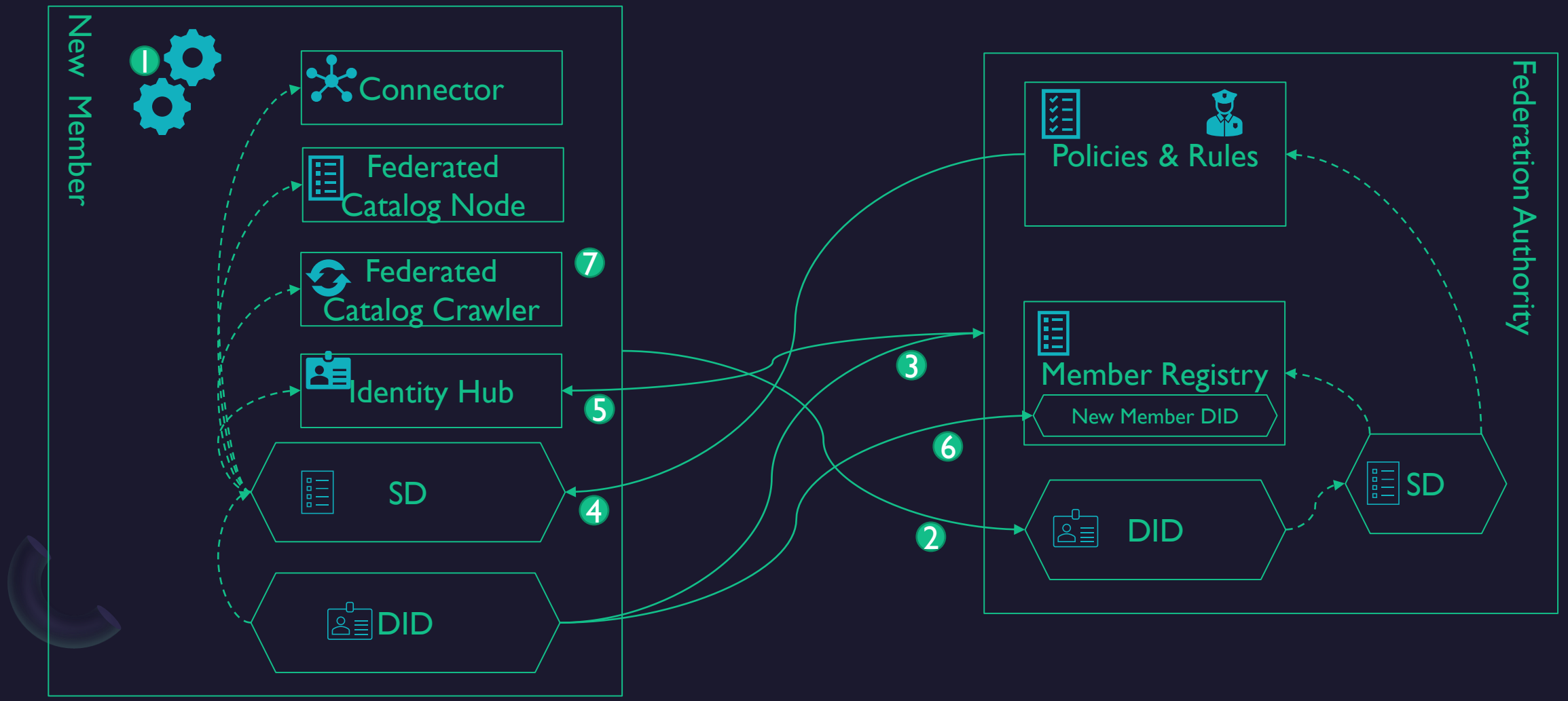
Self-Description

Attributes of the Member
List of Verifiable Credentials
Pointer to Identity Hub
Pointer to Claims and Proofs
URLs of required Endpoints

Member Registry

Member DID is added to the Registry

Joining a Dataspace



Joining a dataspace

FULLY AUTOMATED PROCESS

- Provide DID of dataspace you wish to join to Participants EDC
  - DID Document > Service Entry > Self-Description
  - Self-Description contains all policies > matching
  - Self-Description contains proof of identity and roots of trust > Identity verification
- Provide Participant DID to Dataspace Authority
  - DID Document > Service Entry > Self-Description
  - Self-Description points to Identity Hub > verify VCs
  - Does the self-description match the policy and is the Identity verified and trusted?
- Issue Verifiable Credential of Membership
  - Stored in Identity Hub of Participant
- Add Participant DID to Member Registry

MANUAL PROCESS

- Go to website for the dataspace
- Apply to the dataspace
  - Fill out form, provide URL to DID and/or Self-Description
- After successful validation of applicant Membership Verifiable Credential is issued
  - E-Mail, Download, Push to Participants Identity Hub, etc…
- Can have additional manual workflows like approval processes

Managing Data Contracts

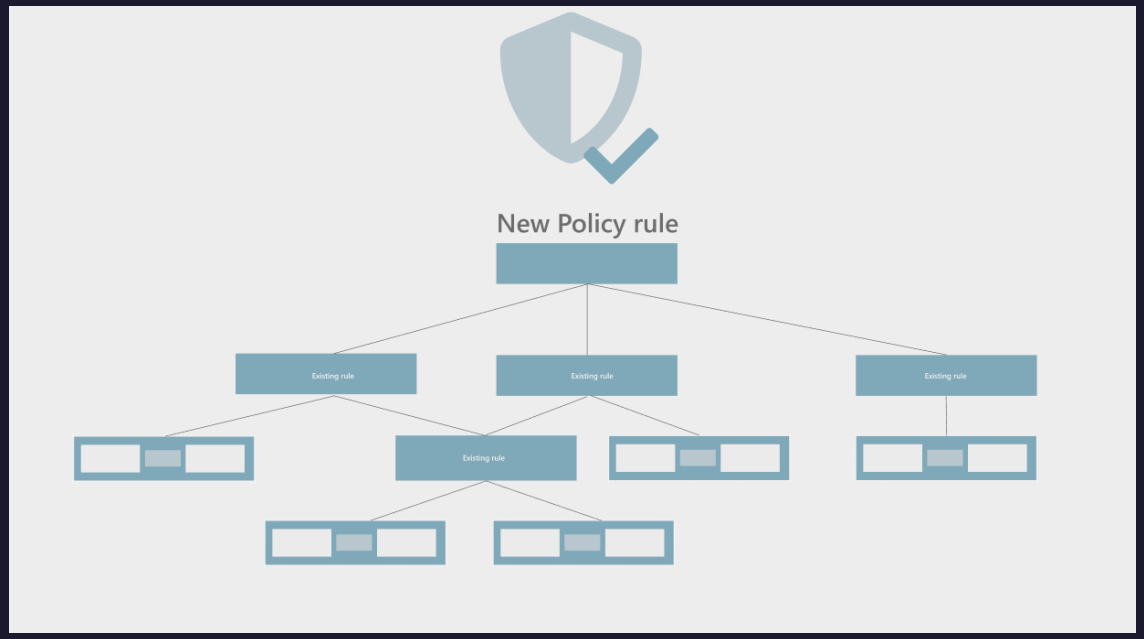
Data Assets	Usage Policies	Access Policies
Symbolic Links to Datasets	Obligations Prohibitions	Visibility based on claims: - Verifiable Credentials - Attributes in the Self-Description

Managing Data Assets

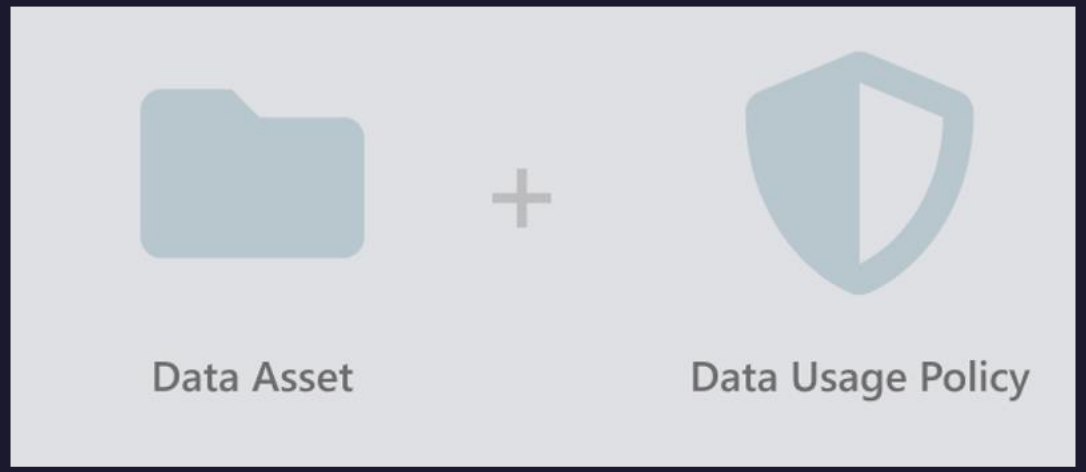
- Create a Data Asset
  - Provide mandatory properties, e.g.: Name
  - Add 1..n Datasets
    - URL to dataset
    - Provide Properties, e.g.: Type, Query Templates, etc..
- Store Data Asset in the EDC Asset Index
- Data Assets can also be Self-Descriptions of Services
- Data Assets can also be Compute to Data Packages

Managing Policies

- Create simple rules and combine them to more complex ones.
- Usage Policies
  - Controls what can be done with the data
  - Controls what must be true
  - Examples:
    - Must be stored in Germany
    - Storage must be C5 certified
    - Data must be deleted after 7 days
- Access Policies
  - Controls visibility to and access from other members



Data Contract Templates



Data Contract Definitions

DATA ASSETS

- Consists of multiple datasets
- A data set is a pointer to the data
- Supports diverse storage and data technologies
- E.g.: a folder with images and text file with metadata form a dataset for machine learning

USAGE POLICIES

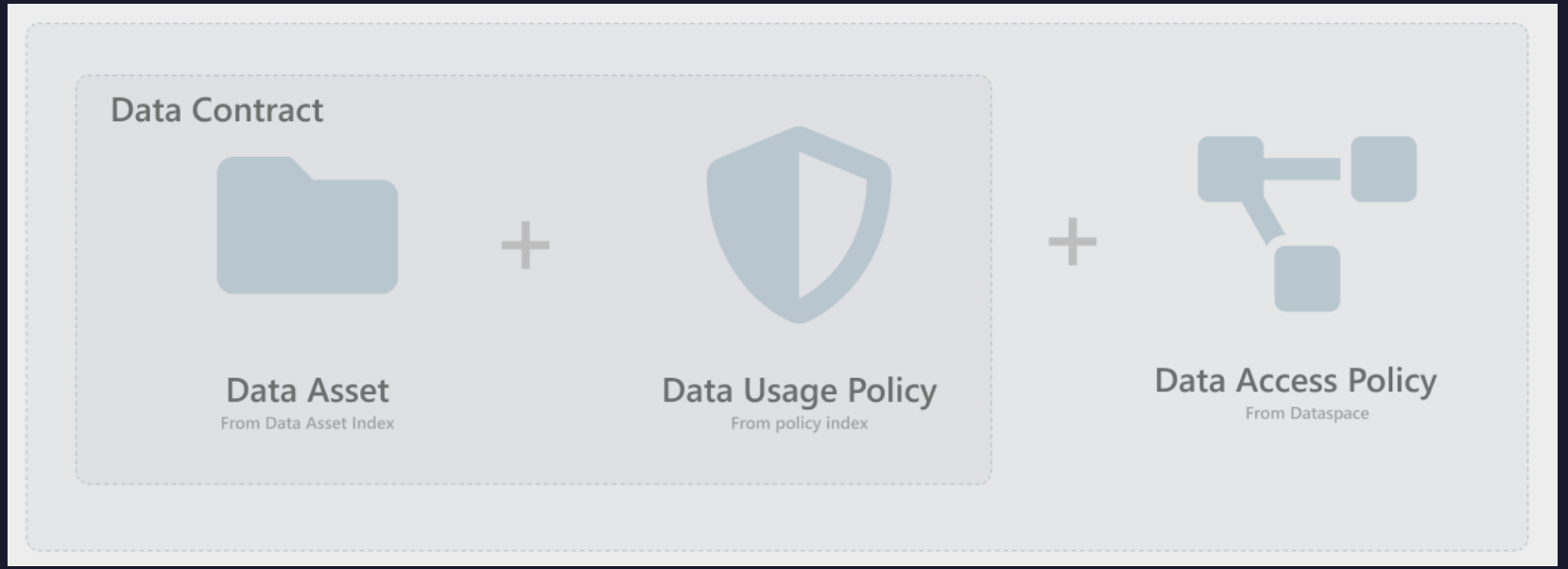
- Controls what can and cannot be done with data
- Obligations – 'must be deleted after 7 days', 'must be stored in the EU'
- Prohibitions – 'cannot be used for AI training purpose', 'must not be shared with other participants'
- Can be combined to form complex rule set

ACCESS POLICIES

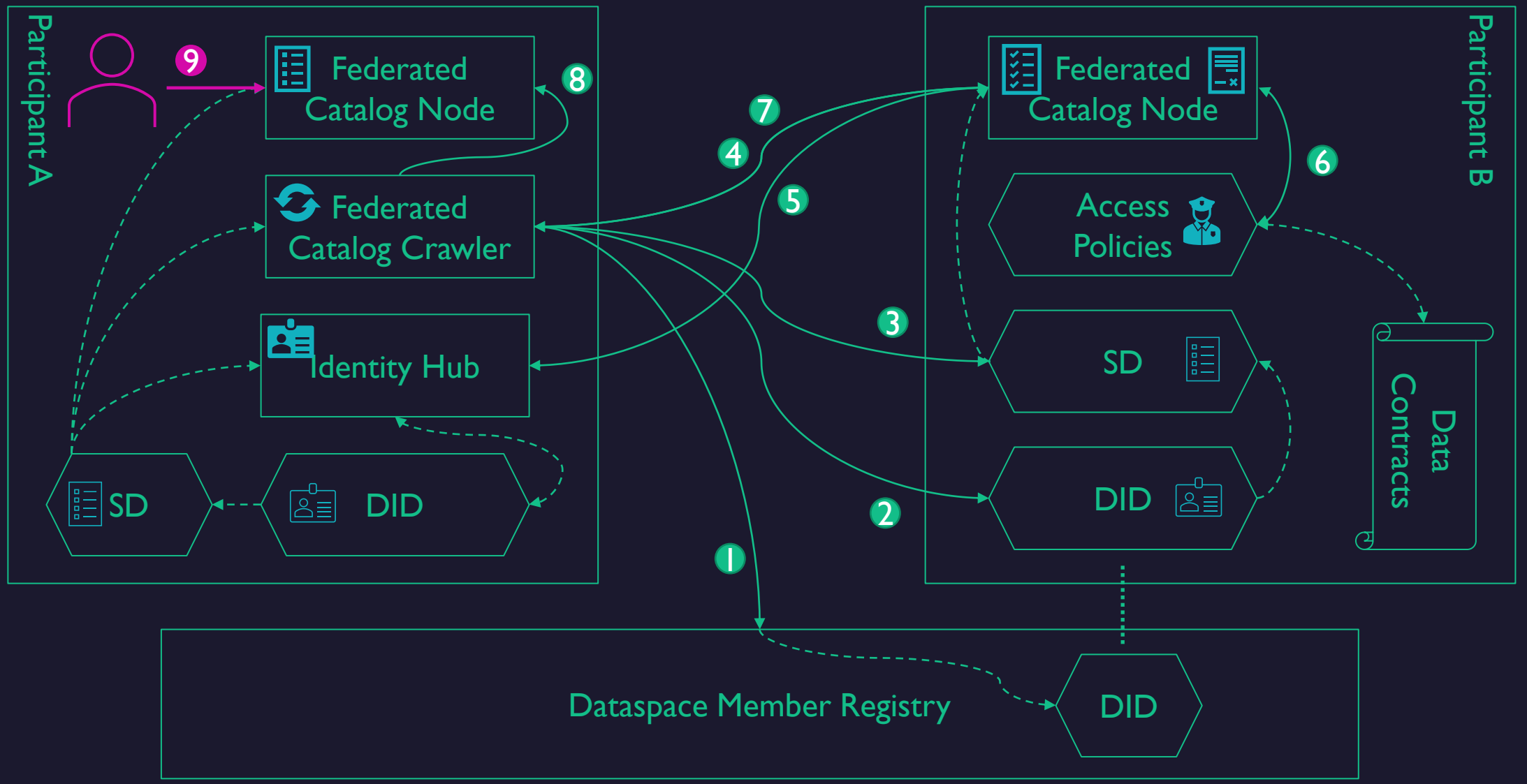
- Controls who can discover the data contract in the Federated Catalog Node
- Easily filterable by Verifiable Credentials of crawlers from other participants
- Dynamically generates resulting Data Contract Definition based on presented VCs and Claims

Publishing Data Contract Offers

- Published in participant's catalog
- Access Policy controls who can discover it in which catalog
  - Can be published to multiple dataspaces
  - Based on VCs required
- Access Policy can further restrict discoverability
  - E.g. “only companies that have a VC issued by me designating them as my supplier”



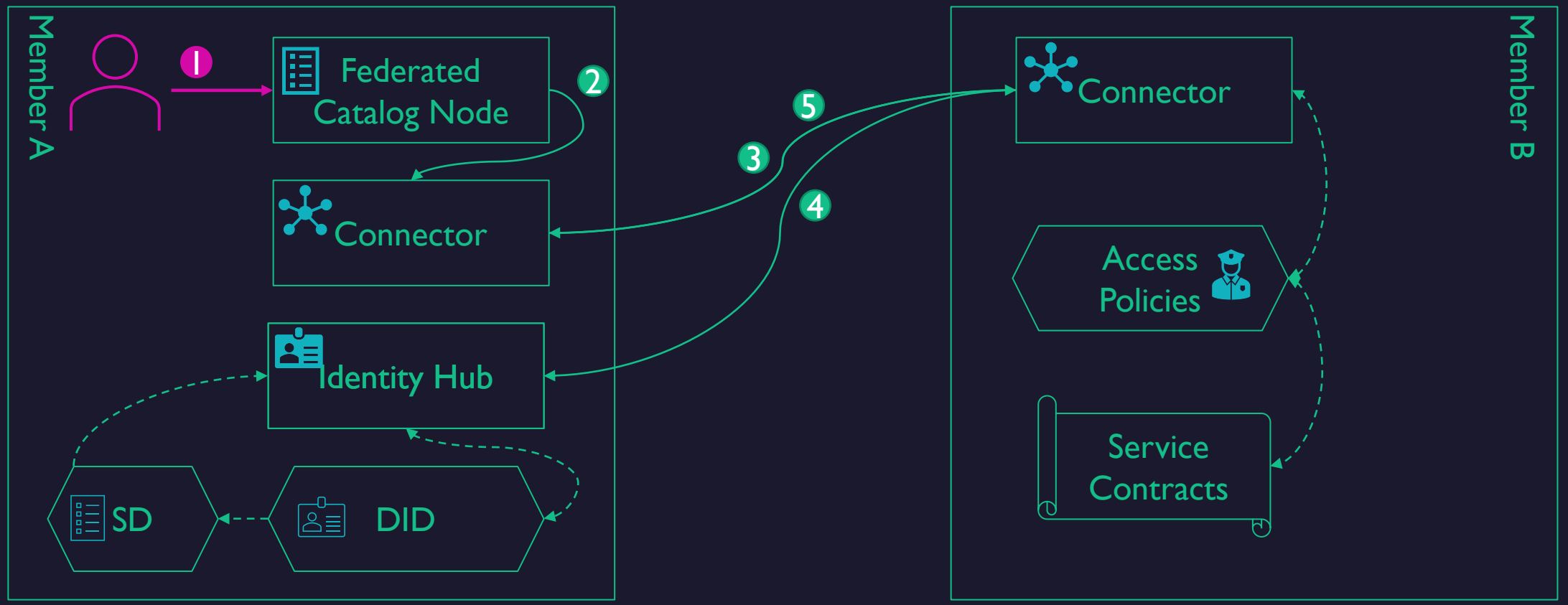
Discovering Data Contracts



Discovering Data Contract Offers

- Federated Catalog Crawler (FCC) reads Member Registry
  - DID > DID Document > Service Entry > Self-Description > Information on Catalog (Location, Access Policies, Interfaces)
- FCC crawls Federated Catalog Nodes (FCNs) of other participants
  - Presents Verifiable Credentials for Dataspace Membership and additional relevant credentials and information through Identity Hub
  - FCNs return filtered view based on access rights of the FCC's VCs
  - FCC caches discovered Data Contract Offers in local FCN
  - FCC can be provided filters to limit scanning of FCNs based on the other participants self-description (e.g. "only scan French participants")
- FCC regularly checks Member Registry and respective catalogs for updates
- Data User can query the local FCN to discover Data Contract Offers

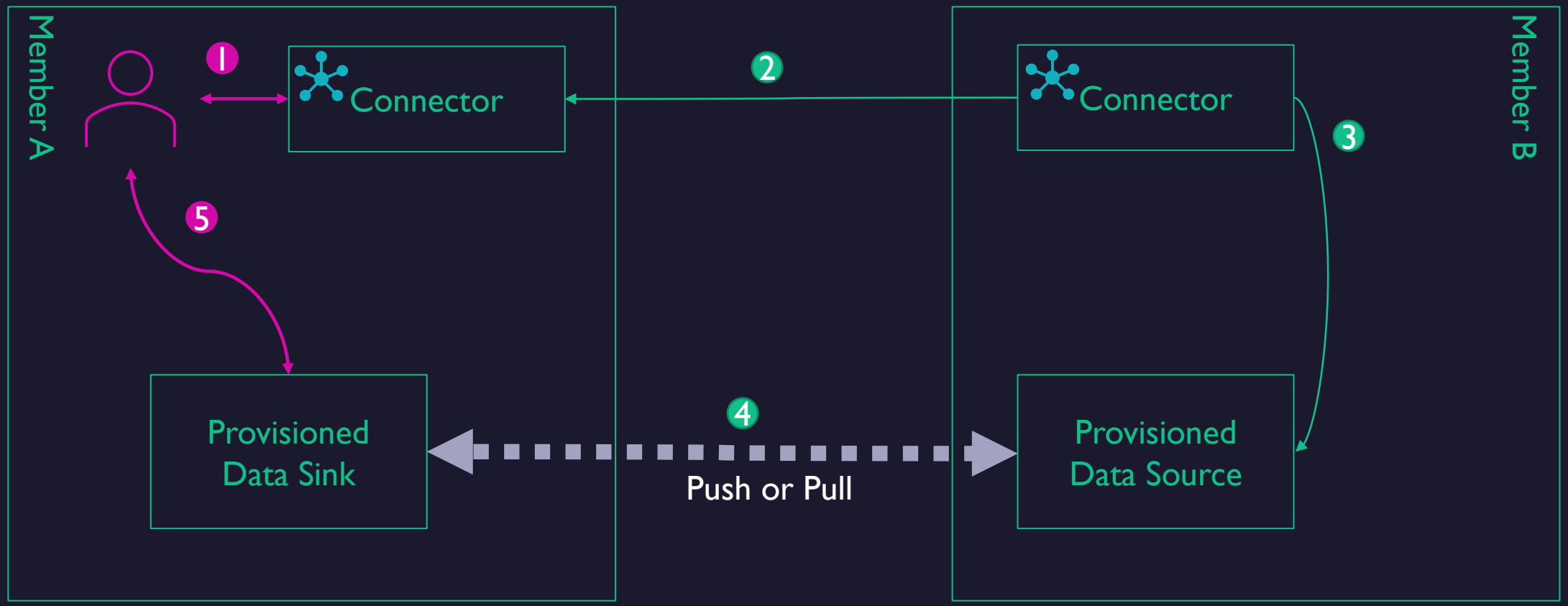
Data Contract Negotiation



Contract Negotiation

- A Data Consumer selects a Data Contract Offer from the local FCN which they would like to consume
- The local EDC contacts the other participants EDC to start negotiation of the data contract
- VCs and Self-Descriptions are checked
- Any additional approval workflows are executed
- Data Contract Offer turns into a Data Contract Agreement and is issued a unique ID

Data Consumption



Contract Execution

- Data can be pushed or pulled
- Target environment is provisioned, verified and proven to conform with policies
- Source environment is provisioned, verified and proven to conform with policies
- Data Transfer is executed

Observing Contract Execution

- Federated Logger operating per EDC
- Log entries linked to DIDs of the executing party to ensure proper access control
- Classic Transaction Monitoring
  - Success, Failure, Aborted, etc…
- Policy Monitoring
  - Hooks provide communication channels to confirm policy execution even after a long time has passed
    - E.g.: "delete data asset after one year"
  - Auditable and Traceable via unique ID of Data Contract Agreement
- Telemetry data for ongoing performance monitoring or billing purposes
- Log data can be exchanged between participants and/or additional service providers as data contracts following all rules and procedures that apply to any other data contract

Federation Services

- Dataspaces or Individual Participants can be anchored in multiple Trust Frameworks
  - industry associations, regulations, federations
- Hierarchical nesting with strong links of trust are possible
  - Dataspaces within dataspaces
  - Multi-company, multi-division, multi-national participants
- EDC can be used to exchange metadata of services in a compliant and secure way
  - Service information can be represented as a Data Asset and thus be discoverable and access controlled through the Federated Catalog Node
  - The EDC policy system can also be used to broker access to and verify properties of any type of service
- Any additional Federation Service can be built on this model
  - E.g. Compliance (exchange of audit and compliance data, verification of VCs and proofs of conformance)

Any Federation designed on Self-Descriptions & Decentralized Trust can be implemented as a dataspace of metadata.

Thank You

https://projects.eclipse.org/projects/technology.dataspaceconnector

https://github.com/eclipse-dataspaceconnector

https://www.youtube.com/channel/UCYmjEHtMSzycheBB4AeITHg