



Building Dataspace with EDC

Peter Koen

Cloud Standards Architect

Microsoft Corporation

Corporate Standards Group

peter.koen@microsoft.com





Designing and 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.

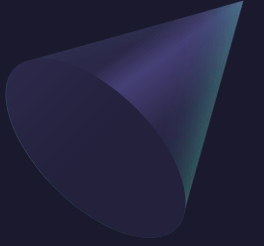
Why do we need a decentralized design?

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)

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

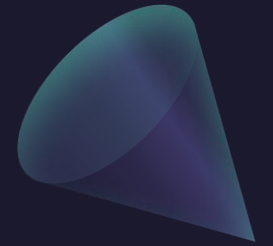
EDC Architecture & Components

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



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

- 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

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

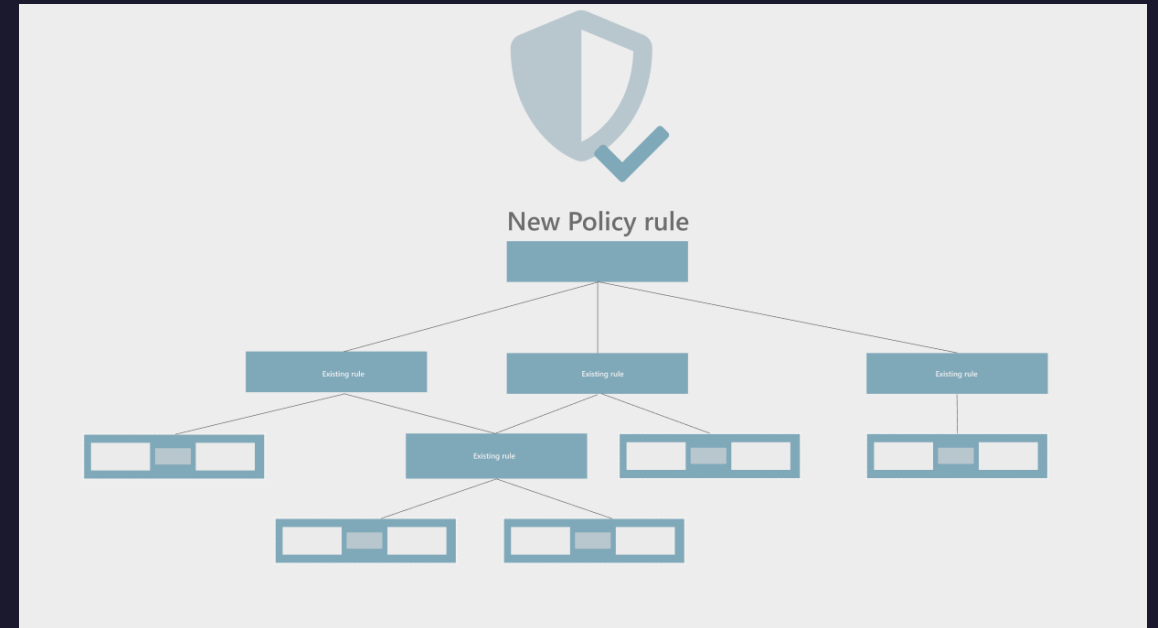


Managing Data Assets

- 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 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

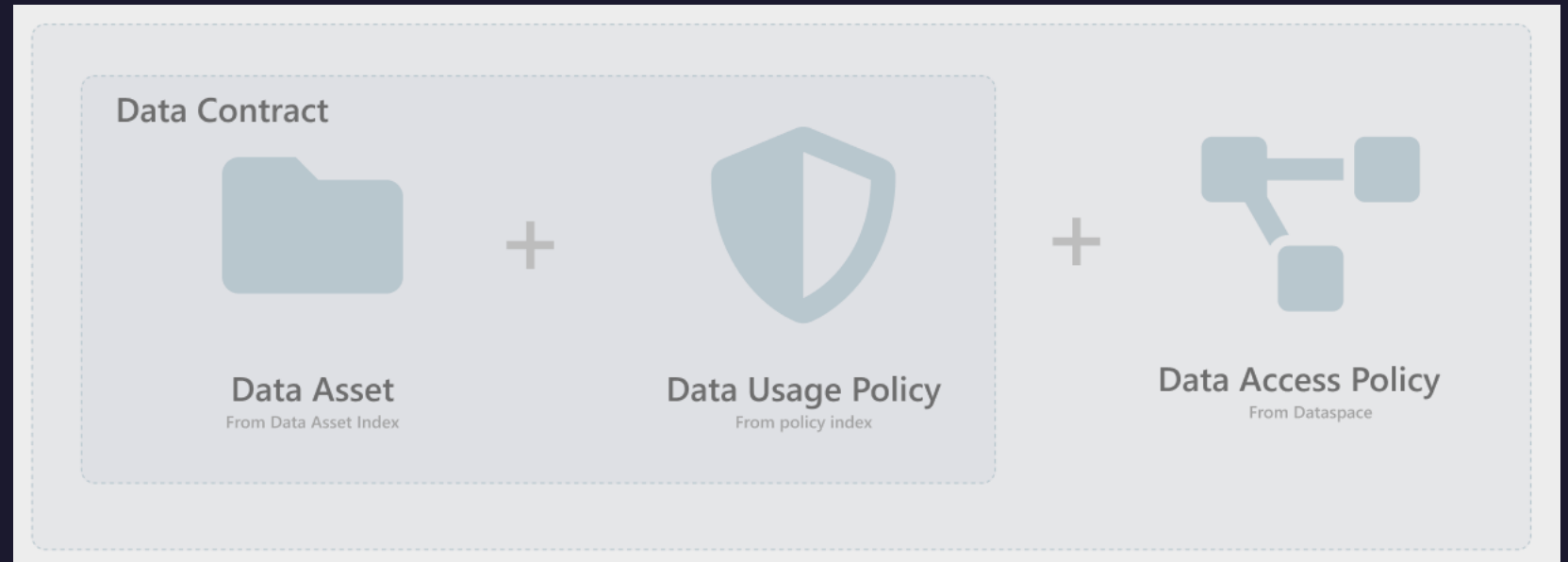
- Controls what can and cannot be done with data
- Obligations – ‘must be deleted after 7 days’, ‘must be stored in the EU’
- Prohibitions – ‘cannot be used for AI training purpose’, ‘must not be shared with other participants’
- Can be combined to form complex rule set

ACCESS POLICIES

- Controls who can discover the data contract in the Federated Catalog Node
- Easily filterable by Verifiable Credentials of crawlers from other participants

Publishing Data Contract Offers

- Published in participant's catalog
- Access Policy controls who can discover it in which catalog
 - Can be published to multiple dataspace
 - 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 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

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

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/UCYmjEHtMSzycheBB4AelTHg>

