The AAS Dataspace for Everybody: An Architecture Example for a Simple Dataspace – purely based on the Asset Administration Shell Concepts

As of October 2023

In the communities around the asset administration shell, development of AAS-based systems for collaborative data sharing is currently a major focus.

These developments can build upon existing drafts or published documents from both the IDTA e.V. and IEC TC65 WG24, and a lot of open-source components that have been shared.

This document describes an architecture example based on these building blocks. The main intentions are:

- clarify the stakeholder participating in the dataspace, their roles and operated software components
- give guidance about the role of the different APIs described in xyz-P2
- be a base for conversations about the implementation of cybersecurity measures in the AAS domain
- introduce some of the terms and definitions from IEC 63278-x, by using them in an architecture example.

Please note that the architecture is a description of an example, and not a prescription for all dataspaces using the AAS.

The document has been developed in the "Task Force Sichere Verwaltungsschale" of Plattform Industrie 4.0 (joint effort between AG1 and AG3), and it has been implemented as an open-source project "AASX server"[1] for demonstration purposes, e.g. for the DPP4.0 showcase of the ZVEI.

1 *https://github.com/admin-shell-io/aasx-server*

I. What is a Dataspace?

1.1 What is a Dataspace?

A *data space* (**→Glossary**) can be defined as a decentralized data ecosystem built around commonly agreed building blocks enabling an effective and trusted sharing of data among participants[2].

It is commonly defined, set up, and operated by a community of participants having an interest in sharing data, but without using central services for storing this data, thus enabling participants to maintain a certain sovereignty regarding their data.

The community agrees on all technical interoperability aspects, common cybersecurity agreements to establish trust, and the organizational and legal aspects of the dataspace.

1.2 The AAS Dataspace for Everybody: An Architecture Example

This document describes an architecture of a simple dataspace which uses only the concepts of the asset administration shell as a foundation.

As we use the AAS, and we want to provide an example for everybody to experience, we call it (→) *AAS Dataspace for Everybody (ADE)*.

The underlying goals are:
- demonstrate an effective exchange of information for collaboration between participating companies
- show the relationships and interactions of the major AAS concepts:
  - asset identification and AAS identification
  - the AAS meta-information model, submodels, and semantic definitions
  - serializations and communication protocols
  - data sets containing information about an asset, and the AASX package container format (AASX)
  - AAS servers and AAS discovery/registry, including their APIs
  - attribute based access control (ABAC)
  - mutual acceptance of data usage policies ("data business policies"[3]) for data
- demonstrate data sovereignty of participating companies regarding identifiers, holding and providing of information to other participants
- allow a decentralized infrastructure that is mainly operated by the participating stakeholders, using interoperable implementations that are based on a set of agreements among all participants
- ensure a level playing field for data sharing and exchange, reducing dominance of central players, also ensuring low entry barriers for participants.

Essential features of ADE include
- asset identification based on IEC 61406-x, using company administrated identifiers based on internet URLs
- a modular information modelling approach based on the asset administration shell (AAS) described in IEC 63278-x
- participant-operated AAS servers, offering http-REST interfaces (APIs) for AAS access that allow easy integration into existing IT landscapes.

Major design goals of ADE include
- easy to understand, to implement, and to maintain
- no use of application specific or sector specific building blocks
- a flexible approach to control access to asset data, covering the protection of intellectual property and trade secrets of participants.

2 *Source: Guidance on IoT and digital twin integrations in data spaces, ISO/IEC JTC 1/SC 41 N2335*
3 *https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Multilateral_Data_Sharing.html*

1.3 Described Scenario: Client-Server Pattern

In this document, we are focusing on a client-server-based data exchange between two participants.

The document "*Multilateral data sharing in industry*"[4] describes multilateral data exchange (between three or more participants) - the necessary technical interactions can always be broken down to a 1:1 relationship between one participant offering data access via servers and another participant implementing access using a *client application*.

This reduces the risk of a misunderstanding that data is exchanged via a central or shared "data pool" – data always has its home at one of the participants and is being provided from there[5].

For a simplification of language, we will describe the architecture from the perspective of a participant (→) *manufacturer* (see stakeholders):

- The participant (→) *manufacturer* has an interest in providing access to certain data (e.g. product data) related to a specific asset (e.g. a product, component, or a machine) that he is responsible for; the manufacturer is preparing the data to be shared for this asset and makes it accessible – he is also called the (→) *AAS responsible*
- another participant (→) *user* has an interest to access this data for the product and has an agreement with the manufacturer to access it.

From this perspective, the systems operated by the manufacturer are called in this document *internal systems*, other systems are called *external*.

1.4 Requirements Overview

Functional suitability	F1. Allow participants to share the information for which they are responsible F2. Allow decentralized data storage of exchange data at the participant that provides or uses it
Security, confidentiality and usage policies	S1. Allow participant authentication, conform the identity of participants. S2. Allow data verification, confirm the authenticity and integrity of the data S3. Ensure data sovereignty: The data provider is responsible for determining access to and use of the data generated S4. Ensure secure data storage and protection from unauthorized access S5. Logging and audit trail must be implemented to identify cybersecurity incidents. S6. Support mutual acceptance of data usage policies

4 *https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Multilateral_Data_Sharing.html*
5 *In the client-server setup, data is also not "broadcasted" to several participants, although an AAS-based publish/subscribe architecture may implement such interaction patterns in the future; the basic principles of using APIs, authentication, authorization, and access protection will remain.*

Interoperability	I1. Demonstrate a dataspace architecture based on the concepts of the AAS. I2. Use all definitions from the asset administration shell on data modelling, semantics, APIs, protocols, file formats
Modularity and modifiability	M1. Ensure flexibility to add/edit/remove participants, assets, or asset information
Accessibility	A1. Ensure participation opportunities for companies who do not have their own information system, e.g. by asking a cloud service provider to operate necessary components on their behalf.

1.5 Stakeholders

Participants

Stakeholders are actively participating in the data exchange in the ADE are called (→) *participants*, examples are
- Machine suppliers: delivering complete machines
- component suppliers: delivering components to machine suppliers
- factory operators: users of machines (and the included components or products)

As described above, we are generalizing the participant roles to one (→) *manufacturer* who has produced an asset and wants to make asset information accessible to other participants, and one (→) *user*, who has an interest in the asset information and is allowed to access it at the manufacturer.

user

- easy identification of an asset, e.g. by means of an asset identifier included in a QR code (or RFID, NFC, or similar)
- easy access to asset information the user is allowed to access
- clarity about the allowed data usage, either by a bilateral contract with the manufacturer, or community agreed policies for all ADE participants

manufacturer

- clear description how to put together the asset
- information, and how to make it available
- control of access to asset data per participant
- protection of trade secrets and intellectual property
- regarding his product
- clarity about the allowed data usage, either by a
- bilateral contract with the manufacturer, or community
- agreed policies for all ADE participants

ADE Community

An additional group of stakeholders is the group of people and organizations involved in defining the game rules for ADE. They may or may not be participating in the data exchange.

We call this group (→) *community*, examples would be the IDTA, and involved industry associations like VDMA or the ZVEI.

In the end, the (→) *community* defines the setup and rules for ADE, and the participants will be using it, especially:

- agreements for the technical interoperability
- cybersecurity agreements to establish the necessary trust
- (→) *data usage policies*, describing what a participant is allowed to do with received data
- all legal and organizational governance aspects of the ADE
- if necessary, commercial agreements to cover operational cost of ADE.

II. Constraints

The structure of the document follows the arc42 template.

This section will be added in an updated version of the document.

III. System Scope and Context

The scope of ADE architecture includes:

- the participants that are involved in the data exchange
- the identity and access management system allowing participants to trustfully identify each other, restricting, or allowing access to data, and allowing to issue cryptographic certificates for signing or encrypting data
- options to integrate with other dataspaces like Catena-X
- the method used to identify (→) *assets* by using an (→) *asset identifier (asset-id)*
- *AAS discovery and registry* to find the necessary endpoints to access an AAS server, e.g. based on an asset-id
- the AAS data model that is used for structuring data related to an asset (→) (*asset data*)
- the data storage and data provisioning by (→) *AAS servers* operated by participants (or on their behalf by a 3rd party)
- usage of standardized application programming interfaces (API), protocols and data formats to find and access the AAS servers
- participant authentication, access authorization and fine-grained access control
- offering and acceptance of data usage policies for exchanged data
- the interactions and information flows that are used to exchange asset data.

Economic, legal, and organizational governance aspects are not discussed here – these need to be agreed in the community, e.g. the IDTA.

3.1 Business Context

Assuming the bilateral scenario, the information flow is the following:

- From his internal systems, the manufacturer makes *asset data* available to authorized participants by means of an (→) *AAS server.*
- After authentication, and accepting the data usage policy, the user may access asset data at the manufacturer by using the (→) *AAS interface*, an API that allows access to asset data of one or more assets.
- The user may then use the asset data in his own systems, along usage agreements that he has with the manufacturer[6].

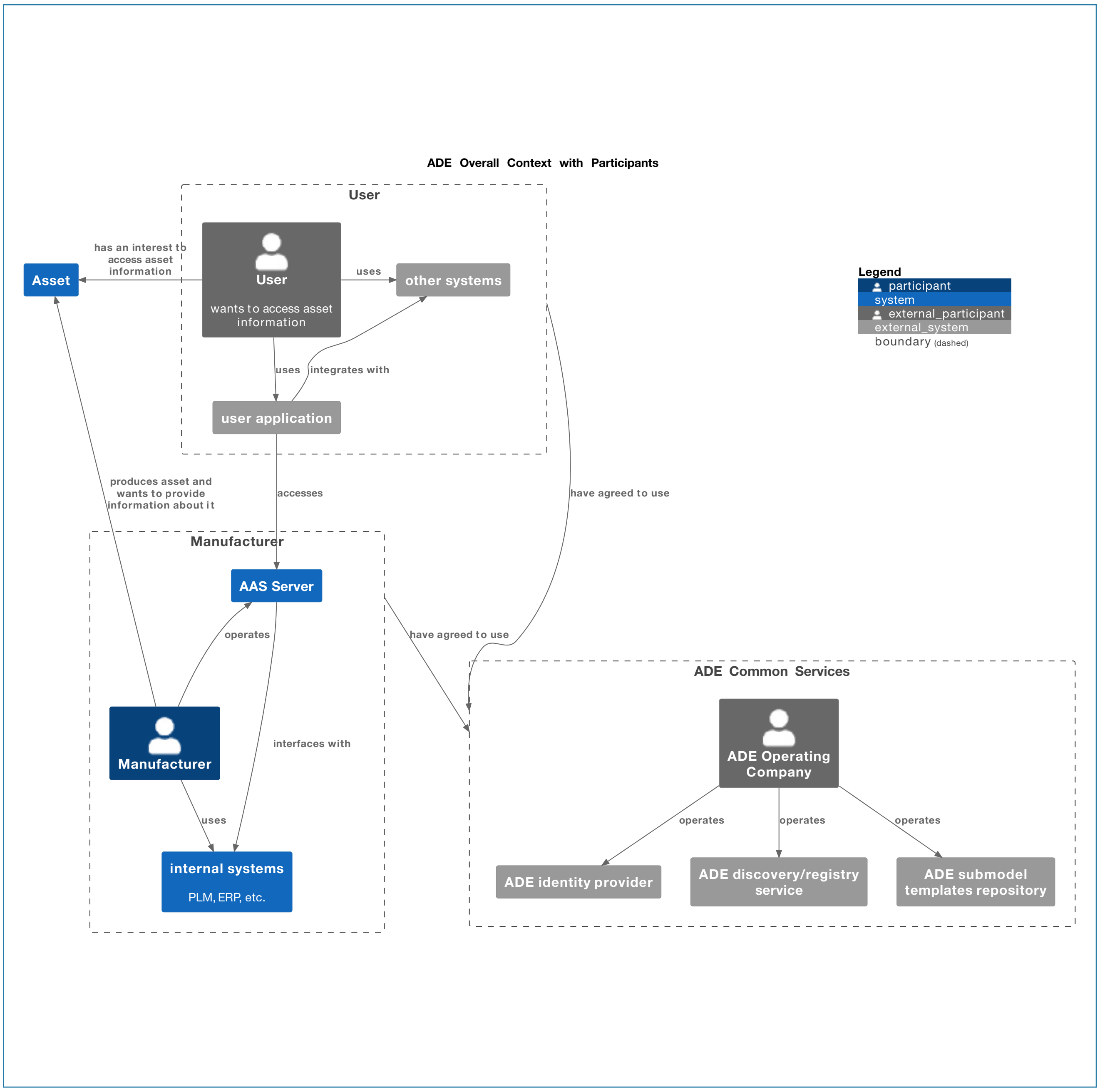
6 *We assume in this architecture that these agreements are not enforced by technical means (i.e. no implementation of usage policies or similar).*

For the data exchange in ADE, participants have agreed to use three ADE common services:

- (→) *ADE authentication service* allows participants to identify each other, based on a common trust anchor.
- (→) *ADE discovery/registry* can translate asset-ids into AAS server and submodel endpoints.
- (→) *ADE submodel template repository* is a repository for submodel templates that participants have agreed to use. Submodel templates are used to provide clear unambiguous semantic definitions for all AAS instance data to be shared.

Figure 1 gives an overview of the complete system, showing participants, ADE common services, and the asset.

Figure 1: ADE Overall context



3.2 Technical Context

To access asset data at the manufacturer, the user must follow these steps:

1. Identify the asset that he is interested in, e.g. by reading a QR code affixed on the product and thus receiving the (→) *asset identifier*[7] (asset-id). The asset-id is a unique URL, used only for identification of the asset. Typically, the user does this by reading a 2D-code (QR code, data matrix code) on the asset that contains the asset-ID, or he has prepared a list of asset-IDs he is interested in.

2. Authenticate with the ADE authentication service to receive an access token that must be used to get access to the ADE discovery/registry and then the manufacturer's AAS server.

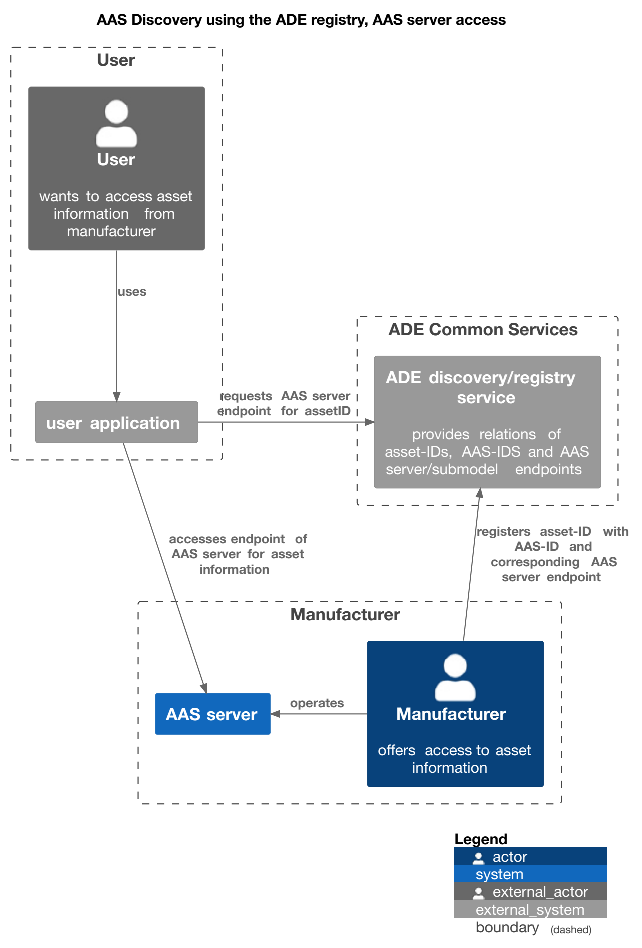
3. Find the endpoint of the AAS server by accessing the ADE discovery/registry, providing the asset-id as a parameter.

4. Access the endpoint of the AAS server. The AAS server allows access to the asset data of the requested asset if the user is authorized to access it.

Please note that access to the ADE discovery/registry service and to the AAS server must be authorized – the authentication and authorization process is described in chapter 4.5.

Figure 2 shows the involved components and the main interactions.

Figure 2: Data flow for AAS discovery and AAS server access, authentication and authorization not shown


AAS Discovery using the ADE registry, AAS server access
User
User
wants to access asset information from manufacturer
uses
user application
requests AAS server endpoint for assetID
ADE Common Services
ADE discovery/registry service
provides relations of asset-IDs, AAS-IDS and AAS server/submodel endpoints
accesses endpoint of AAS server for asset information
registers asset-ID with AAS-ID and corresponding AAS server endpoint
Manufacturer
AAS server
operates
Manufacturer
offers access to asset information
Legend
actor
system
external_actor
external_system
boundary (dashed)

7 *IEC 61406-x describes 2D-Codes (e.g. QR codes, data matrix codes) and RFID as data carriers.*

IV. Solution Strategy

4.1 Participants that are involved in the data exchange

The participants that are active in ADE have already been outlined in chapter stakeholders.

To meet the requirements on data security, authentication of participants is required before they access system components of ADE.

4.2 Method used to identify assets: the asset-id

The AAS concept uses URIs (universal resource identifiers, RFC 3986) to identify assets; more precisely it uses URLs (RFC 1738), a specific form of URIs that simplifies the resolution of the identifier to the final source of AAS content.

The manufacturer manages asset identification under his own responsibility.
This choice has several advantages:
- the identifiers can be assigned without a central registration authority; except for the manufacturer domain name which must be registered in the DNS system
- no incremental cost for each created identifier is incurred.

As data carrier to be affixed on the product, 2D codes (QR or data matrix) may be used, alternatively RFID of NFC. These data carriers can be read with common reading devices, and with modern smart phones.

IEC 61406-x specify the relevant principles and restrictions for the identifier and the data carrier used in ADE.

4.3 AAS servers

In ADE, storage of the asset information is implemented in AAS servers operated by the participants (e.g., the manufacturer of a product). The servers can also be operated by a 3rd party (e.g., a cloud service provider) on behalf of the participant.

The AAS server holds (stores) the asset data for one or more assets. Each individual asset-id is linked to exactly one (→) *AAS instance data set*.

The AAS server offers an (→) *application programming interface* (API) that can be accessed from other participants in the ADE after authentication. The API can be accessed via a defined (→) *endpoint* belonging to the AAS server. The AAS content related to one asset-id can be accessed as a complete AAS instance data set (including all AAS content), or partially on submodel or submodel element level (see section Data Model).

The server also verifies actor's authentication, and enforces access authorization (see below).
The AAS server includes an asset data storage with asset data for each asset-id. It is connected to further systems:
- to the asset itself if the asset can provide online information like variables or parameters
- to other systems that hold information about the asset.

4.4 ADE Discovery/Registry

To find the endpoint for a given identifier, the (→) *ADE discovery/registry* is used. When asked for an asset-id, the ADE discovery/registry returns the AAS server endpoint. After that, accessing the AAS server endpoint and its submodel endpoints, asset data can be accessed.

The ADE registry offers an application (→) *programming interface* (API) that can be accessed from other participants in the ADE after authentication.

4.5 Identity and access management

Participant Authentication

To ensure a trusted data exchange between participants, they must agree on a common way of authentication.

In the ADE, all participants agree to trust one common trust anchor, e.g. a trust service provider with a PKI. The ADE community may agree to use one (ore several) commercially available PKI offerings.

This trust service provider issues X.509 certificates to all participants, which they can use as credentials to authenticate.

Authentication in the ADE is implemented by using (↗ Appendix) *OpenID Connect*. Participants need to authenticate with an OpenID Connect server, using their X.509 certificate or other credentials the ADE community has agreed upon[8].

In return, they receive an (→) *access token* that they will present when accessing systems at the manufacturer, e.g. the AAS server. ADE uses (↗) *JSON web tokens* (JWT) as access tokens.

In ADE, access is also possible without authentication – in this case only publicly available AAS content is accessible.

Access Permissions and Access Authorization

Users may have different information requirements, and the manufacturer controls the access to his information - therefore different parts of the AAS content might be restricted to certain participants.

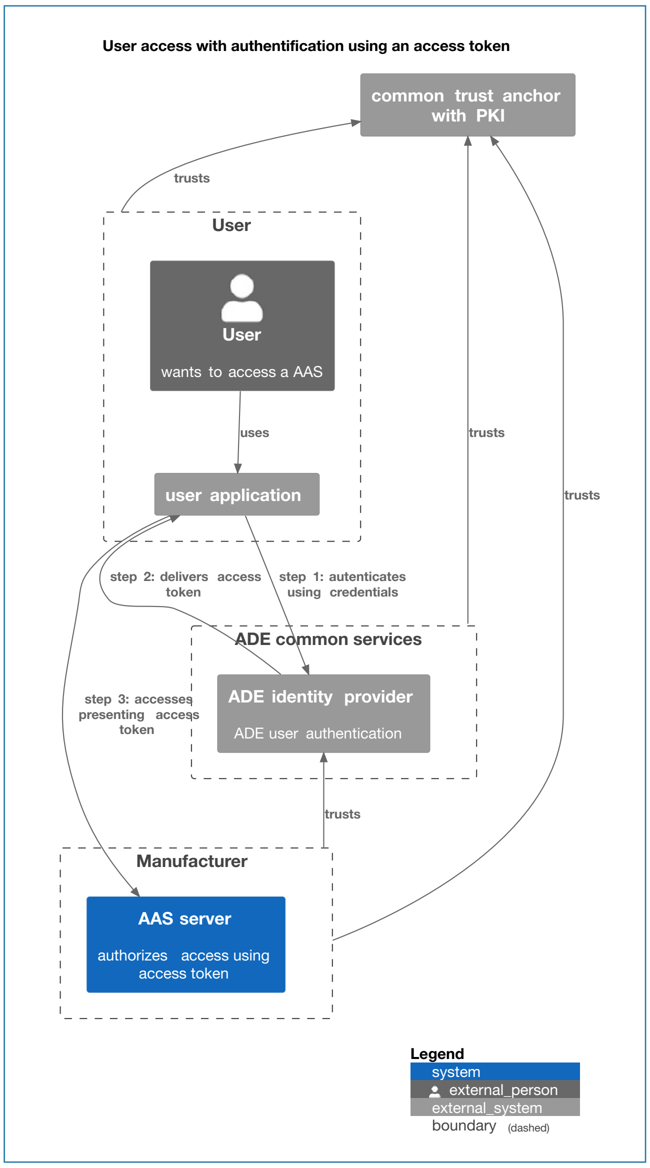
By verifying the presented access token, both the AAS registry and AAS server can decide which parts of the AAS content can be accessed by the authenticated participant.

The enforcement of access permissions is completely in the hands of the participant that is providing access to the AAS, thus allowing data sovereignty for the AAS holder.

Figure 3 shows the activity flow for authorization.

Two layers of access authorization are useful to implement:
- access permissions that allow access to manufacturer's AAS server
- another set of access permissions that relate to the content of each individual AAS, submodel or submodel element.

Figure 3: Activities for authentication and authorization


User access with authentification using an access token
common trust anchor with PKI
trusts
User
User
wants to access a AAS
uses
user application
trusts
trusts
step 2: delivers access token
step 1: autenticates using credentials
ADE common services
step 3: accesses presenting access token
ADE identity provider
ADE user authentication
trusts
Manufacturer
AAS server
authorizes access using access token
Legend
system
external_person
external_system
boundary (dashed)

Data Usage Policies

To achieve a trustful data exchange, the involved participants need to agree how the data may be used on the receiving side. The participant that is offering data may want to restrict it to a certain purpose, duration, or other criteria, and passing on data to 3rd parties needs to be explicitly agreed.

These agreements are documented in "data usage policies", which are documents that contain the details about the agreement.

8 *Alternative authentication methods are possible – we have demonstrated authentication via verifiable credentials and also the Eclipse Dataspace Connector (EDC). In the end, a trusted JWT is needed for AAS server access.*

Data usage policies can be mutually accepted in several ways:

- as part of a “paper contract” between participants
- as part of the authentication process with the ADE identity provider by displaying a general data usage policy with an “accept” button
- as part of the client/server interactions during the data exchange.

Integrating the agreement into the client/server interactions has the advantage that the policies can be specific for each data element that will be exchanged. In this case, the mutual agreement is achieved with the following steps:

1. Before the data exchange happens, the participant that wants to access[9] data informs the data provider about the data it wants to access . Access policies are also enforced for this step.

2. The data provider then sends a list of offered usage policies for this data.

3. When finally accessing the requested data, the receiver confirms in his request the usage policy he has accepted. This receiver request includes the receiver’s signature

4. The data provider can now check if an offered usage policy was chosen, delivers the requested data along with the accepted usage policy, and log the access for documentation.

5. If the confirmed usage policy is not identical to one that was offered, access is rejected.

After step 4, the data is exchanged, and both parties have an ascertainable mutual manifestation of their agreement on the usage policy.

For the technical flow of providing and accepting usage policies, the actual content of the usage policy document is not relevant, participants only exchange hash values of the agreed documents, which allows a proof of the integrity of the document on both sides. Such usage policy document can be e.g. a PDF, but also machine readable, e.g. provided as JSON or other.

Please note that this process allows the technical enforcement of the acceptance of a data usage policy, but the content of the policy is not technically enforced.

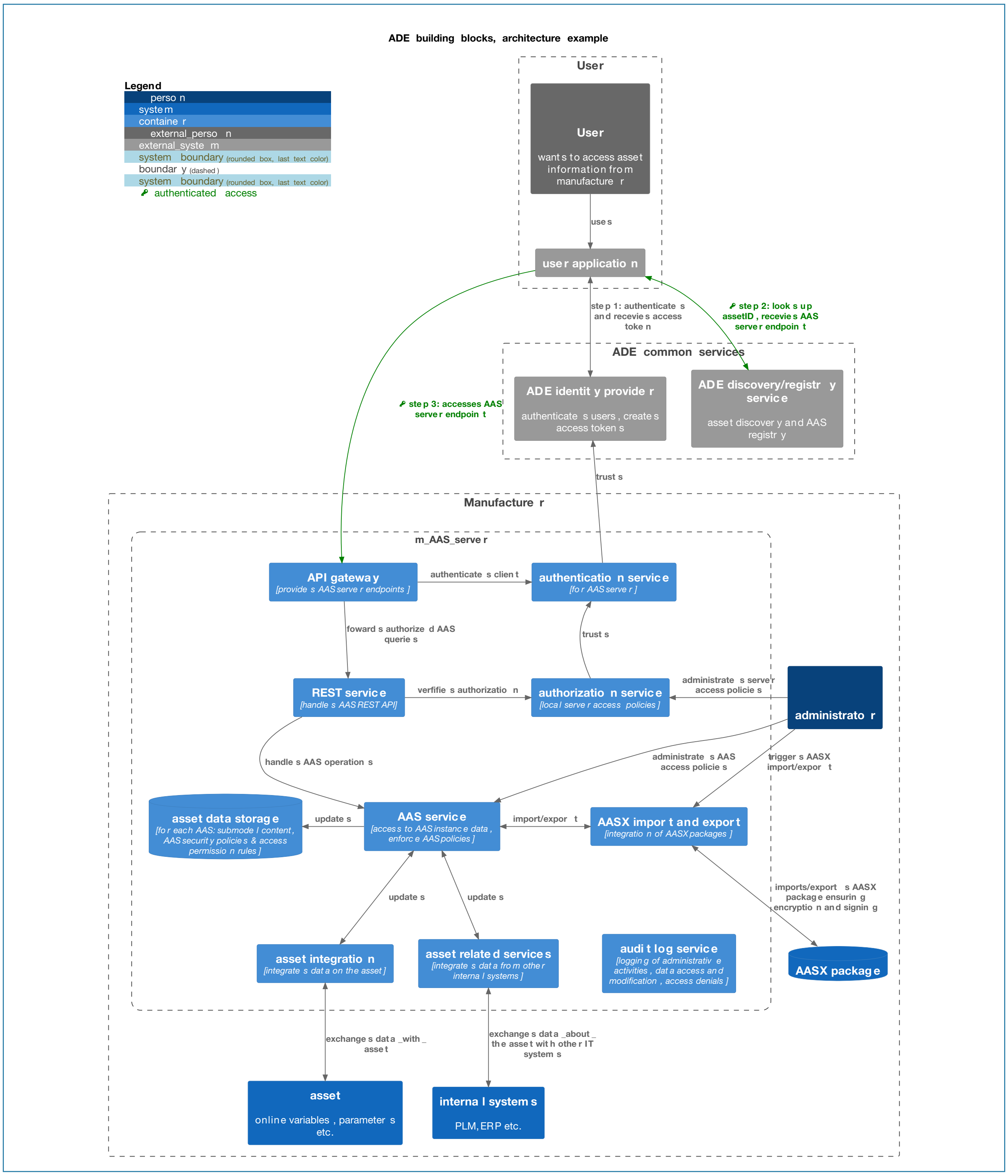
9 An HTTP HEAD request is used to query the available access policies. The HTTP HEAD method has the same parameters as the HTTP GET method, but the answer contains only headers (which contain the offered hashes of the usage policies), and not the requested data.

V. Building Block View

Figure 1 has already shown the setup of participants in ADE, and Figure 2 gave an overview about the main involved system components.

Figure 4 gives an overview about the main building blocks interacting in the ADE as a simplified complete example.

Figure 4: Details of ADE building blocks and their interactions



ADE building blocks, architecture example
Legend
person
system
container
external_person
external_system
system boundary (rounded box, last text color)
boundary (dashed)
system boundary (rounded box, last text color)
authenticated access
User
User
wants to access asset information from manufacturer
uses
user application
step 1: authenticates and recevies access token
step 2: look sup assetID, recevies AAS server endpoint
step 3: accesses AAS server endpoint
ADE common services
ADE identity provider
authenticates users, creates access tokens
ADE discovery/registry service
asset discovery and AAS registry
trusts
Manufacturer
m_AAS_server
API gateway
[provides AAS server endpoints]
authenticates client
authentication service
[for AAS server]
foward authorized AAS queries
trusts
REST service
[handles AAS REST API]
verfifies authorization
authorization service
[local server access policies]
administrates server access policies
administrator
handles AAS operations
administrates AAS access policies
triggers AASX import/export
asset data storage
[for each AAS: submodel content, AAS security policies & access permission rules]
updates
AAS service
[access to AAS instance data, enforce AAS policies]
import/export
AASX import and export
[integration of AASX packages]
imports/export AASX package ensuring encryption and signing
updates
updates
asset integration
[integrates data on the asset]
asset related services
[integrates data from other internal systems]
audit log service
[logging of administrative activities, data access and modification, access denials]
AASX package
exchanges data _with_ asset
exchanges data _about_ the asset with other IT systems
asset
online variables, parameters etc.
internal systems
PLM, ERP etc.

5.1 ADE Common Services

ADE Identity Provider

The ADE identity provider identifies the user (e.g. by asking for registered credentials) and grants an access token that the user application uses to access common ADE services or AAS servers.

It is implemented using the OpenID Connect standard.

As access token, we are using JSON Web Tokens (JWT).

ADE Discovery/Registry Service

ADE discovery/registry service handles the discovery of assets, i.e. translating an asset-id into an AAS-id and resolving the AAS-id to a communication endpoint where the AAS server can be accessed.

It implements the discovery interface and the registry interface of AASSpec-Part-2-V3.

ADE registry access must be authenticated, as not all participants are allowed to reveal all information.

5.2 Building Blocks at the User

The user wants to access AAS content based on asset-ids, either for viewing it, or for further processing in other systems that he is using.

The user application implements at least the following functions:

- authenticate the user at the ADE identity provider, in return receive an access token
- use the ADE discovery/registry to translate asset-ids to endpoints of AAS servers;
- access the AAS server to access the asset data via the AAS service API, accessing content of submodels as needed
- this access must be authorized by presenting the user's access token to the AAS server (including the acceptance of the data usage policy if implemented)
- present the AAS content to the user, or forward it to other systems the user may have for further processing.

5.3 Building blocks at the Manufacturer

We are describing the details of the AAS server as an architecture example, implementations may look differently depending on the manufacturer's needs.

The following table describes the responsibilities of all components of the AAS server in figure 4.

Name	Responsibility
AAS server	• handle all tasks related to processing, preparing, storing, and providing access to AAS content
API gateway	• accept user's access from user's client app • authenticate the user verifying the validity of the access token presented the local authentication service • forward all AAS-API request to the REST-server
authentication service	• confirm the accessing user's identity by verifying the presented access token

authorization service	• enforce server access policies: allow (or forbid) AAS server access based on the user’s identity and AAS server access policies • allow the administrator to manage and store access policies for the AAS server
REST server	• handle all authorized AAS-API requests by using the the AAS service • implements the AAS interface and submodel registry interface and submodel interface from AAS-Spec-P2-V3
AAS service	• look up AAS and its submodels in the asset data storage • enforce AAS access policies: allow (or forbid) AAS instance data access according to AAS access policies • allow the administrator to manage and store access policies for each element in each AAS and submodel that the AAS server is holding • acquire and evaluate additional attributes required for AAS access policies (which is based on attribute based access control, ABAC), like local time etc. • executes CRUD operations with the asset data storage • interfaces with the asset integration to exchange asset information with the asset, and update the asset data storage accordingly • interfaces with the asset related services to exchange asset data with systems holding information about the asset, and update the asset data storage accordingly
Administrator	• manage AAS server access policies with the authorization service • manage AAS access policies for each AAS instance, by adopting these policies to the environment of the manufacturer with the AAS service • monitors the system by analyzing audit logs from the audit log service • triggers AASX package import and export when needed, e.g. when new assets are integrated into the manufacturer
Asset integration	• allows data exchange with the asset, e.g. online variables, parameters etc.

Asset related services	• allows data exchange with system that hold information about the asset, e.g. ERP systems holding financial information about the asset or service
Audit log service	• logs administrative activities, data access and modifications, user authentication denials

5.4 Integration with other Dataspaces like Catena-X

As presenting a valid JWT is the only condition for accessing the AAS server, AAS servers in ADE can be integrated into other dataspaces.

JWTs may also be created by other processes, e.g. the authentication and usage policy negotiation process of Catena-X.

The only condition: The access authentication of the “foreign” dataspace yields a JWT which the AAS server can validate, and the service that issues the JWT is included in the AAS server’s trust relationships.

The JWT may also contain additional attributes that another dataspace requires, like the BPN in Catena-X - these additional attributes can be used inside the AAS server for additional functionality like access authorization.

VI. Runtime View

The structure of the document follows the arc42 template.

This section will be added in an updated version of the document.

VII. Deployment and Operation View

The following principles for deployment should be applied in the ADE:

- ADE common services (ADE Discovery/Registry, ADE authentication service and ADE submodel template repository) are operated by the ADE community, e.g. the IDTA. The ADE community may ask a 3rd party to operate these on its behalf.
- ADE common services should be containerized using the technologies of the Open Container Initiative to allow easy deployment into any cloud or on-premise system.
- Participants are responsible for operating their own AAS servers or ask a 3rd party to operate them on their behalf.

IIX. Cross Cutting Concepts

8.1 AAS Data Model (Information Meta-Model)

All asset related content in ADE is structured along the principles of the asset administration shell (AAS), described in IEC 63278-x [10].

Signing AAS content

Cryptographic signing allows the verification of integrity and authenticity of AAS content; thus it can be proven that the AAS content was originally provided by the participant who signed it, and that it has not been modified.

Further sematic meanings of the signature (e.g. non-repudiation) have not been defined, they can be added if needed for certain use cases.

The complete AAS or selected AAS submodels can by cryptographically signed to ensure data authenticity (confirm the source of the data), reliability and integrity (detection of incidental or intentional modifications).

To allow signing, the common trust anchor/PKI needs to issue certificates suitable for signing content.

8.2 Semantics for Submodels and Submodel Elements

Concept dictionaries in line with IEC 61360 (IEC CDD or ECLASS) are being used to clarify the semantic meaning of submodels and submodel elements.

Each submodel and submodel element is tagged with the IRDI referring to the corresponding CLASS concept according to IEC 61360 (IEC CDD, or also ECLASS).

8.3 Exchange Protocols and Formats

For the data exchange between participants, a client-server architecture is used.

Client applications actively query ADE services and AAS servers.

All communication transactions are stateless.

HTTPS is being used as a secure communication protocol.

JSON is being used for HTTP payloads in queries and responses.XML may be used for exchanging AAS content sets for one or more specific DPPs, using the AASX format to package the information for one or several DPPs into a single file.

10 The current detailed description of the AAS information meta-model can be found in the specification "Details of the Asset Administration Shell - The exchange of information between partners in the value chain of Industrie 4.0 " published by the IDTA e.V. and Plattform Industrie 4.0.

IX. Architectural Decisions

9.1 Discovery and AAS Registry Services combined into one module

The AAS Registry Interface and the AAS Basic Discovery interfaces have been integrated into one software component, as the main purpose is determining the AAS server endpoint for a given asset-id.

9.2 AAS interface, Submodel Registry and Submodel interface combined into one module

The AAS server follows the pattern described in AAS-Spec-Part-2-V3 Figure 8, integrating AAS interface and submodel interface into one software component.

9.3 Using ECLASS and CDD as a semantic dictionary

As the AAS community in the IDTA has agreed to use ECLASS (and optionally IEC CDD) as their preferred semantic dictionary, we have followed this preference in the ADE.

Please note that using different semantic dictionaries still allows to exchange data, but the interpretation of data is not unambiguous anymore.

X. Quality Requirements

The structure of the document follows the arc42 template.

This section will be added in an updated version of the document.

XI. Risks and Technical Debt

11.1 Cybersecurity Risks

- As each participant is responsible for the reliability of his own systems, a total ADE reliability cannot be assured. The ADE community should define guidelines about expected response times etc.
- The usage of open-source SDKs and reference implementations is recommended; these can be reviewed to confirm cybersecurity requirements.

XII. Glossary

Multiple entries in the “Term” column are synonyms, also used to align the terms with IEC 63278-x.

Term	Definition
dataspace	a decentralized data ecosystem built around commonly agreed building blocks enabling an effective and trusted sharing of data among participants
AAS content	The information which is contained in a digital product passport and that is exchanged between participants.
ADE	A system concept for implementing a simple dataspace by means of the asset administration shell concept
asset	a physical entity or digital entity that has value to a participant
Asset identifier, asset-id	a unique identifier that is used to identify an asset
component	a product intended to be incorporated into another product
data carrier	a linear bar code symbol, a two-dimensional symbol or other automatic identification data capture medium that can be read by a device
(unique) product identifier	means a unique string of characters for the identification of products that also enables a web link to the product passport. Can be applied on model, batch, or item level.
participant	a person or organization that has an interest to exchange asset related information in the ADE

User, AAS user application operator	a participant (person or company) that has an interest to access data for an asset, and has an agreement with the manufacturer to access it
manufacturer	A participant (most likely a company) that has interest to share data about an asset that he is responsible for
AAS responsible	human or organization having interest in an asset and governing an Asset Administration Shell related to one specific asset
user application	software application which accesses an AAS via its AAS interface(s) for use by humans or for automatic processing
Asset data	asset information related to one specific asset
AAS instance data set	One complete set of asset information related to one specific asset bearing one specific asset identifier
community	All persons or organizations agreeing on the game rules for the ADE
registry	A software service to find the server endpoints for a given product identifier
AAS server	A software service that makes AAS content accessible for one or more assets
AAS repository	An AAS server implementing the AAS Repository API
ADE authentication service	A software service allowing participants to identify each other, based on a common trust anchor

ADE registry	A software service that can translate asset-ids into
ADE submodel template repository,	a repository for submodel templates that participants have agreed to use. Submodel templates are used to provide clear unambiguous semantic definitions for all AAS instance data to be shared.
Application programming interface, API, AAS interface	An application programming interface (API) is a way for two or more computer programs to communicate with each other. It is a type of software interface, offering a service to other pieces of software. (Source: Wikipedia)
Endpoint, communication endpoint	the resource identifier (URI) of one or several resources used as starting points
Access token	an access token contains the security credentials for a login session and identifies a user and potentially his client application
Asset service	service that is provided by the considered asset
Asset related service	service that is not provided by the considered asset, but by software service outside of the considered asset
Data usage policy	mutual agreement between participants on allowed data usage and usage restrictions for data after access has been granted

XIII. Appendix

Standards and Normative References

Term	Definition
OpenID Connect	OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol. It allows Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner. **Source: https://openid.net/connect/**
JSON web token	JSON Web Tokens are an open, industry standard RFC 7519 method for representing claims securely between two parties. **https://jwt.io/**
IEC 63278-x Asset administration shell	A concept for building standardized interoperable digital twins for industrial assets.
IEC 61406-1, -2 Identification Link	Part 1: General Requirements Part 2: Types/Models, Lots/Batches, Items and Characteristics
AASSpec-P2-V3	IDTA e.V., Asset Administration Shell Specification - Part 2: API, Version 3.0

LIST OF AUTHORS
Kai Garrels (ABB STOTZ-KONTAKT GmbH), Michael Jochem (Robert Bosch GmbH), Dr. Lutz Jänicke (Phoenix Contact GmbH & Co. KG)

THIS PUBLICATION IS A RESULT OF THE “TASK FORCE SICHERE VERWALTUNGSSCHALE” OF PLATTFORM INDUSTRIE 4.0 (JOINT EFFORT BETWEEN AG1 AND AG3).