INTERNATIONAL DATA SPACES ASSOCIATION

Position Paper | Version 3.0 | March 2021

Usage Control in the International Data Spaces

● Position Paper of members of the IDS Association
○ Position Paper of bodies of the IDS Association
○ Position Paper of the IDS Association
○ White Paper of the IDS Association

Fraunhofer

Publisher

International Data Spaces Association
Anna-Louisa-Karsch-Str. 2
10178 Berlin
Germany

Copyright

International Data Spaces Association,
Dortmund 2021

Cite as

Steinbuss S. et al. (2021): Usage Control in the International Data Spaces. International Data Spaces Association.
https://doi.org/10.5281/zenodo.5675884

Editor

Sebastian Steinbuß
International Data Spaces Association

Authors & Contributors

Fraunhofer IESE	Andreas Eitel Christian Jung Robin Brandstädter Arghavan Hosseinzadeh
Fraunhofer IAIS	Sebastian Bader
Fraunhofer IOSB	Christian Kühnle Pascal Birnstill
Fraunhofer AISEC	Gerd Brost Mark Gall
Fraunhofer ISST	Fabian Bruckner Norbert Weißenberg
Fraunhofer IML	Benjamin Korth

Contributing Projects

Industrial Data Space

Industrial Data Space+

www.fraunhofer.de/en/research/lighthouse-projects-fraunhofer-initiatives/industrial-data-space.html

Research Center Data spaces

www.cit.fraunhofer.de

Abstract

In the age of Industry 4.0, data exchange between different organizations is an essential prerequisite to add more value to data and to develop modern business models. However, we have to solve several challenges to facilitate a secure and trustworthy data exchange between different organizations. Data sovereignty is a key success factor for data-driven business models. In the International Data Spaces (IDS), we provide solutions to realize a secure and trustworthy data exchange as well as data sovereignty.

In this report, we focus on data usage control and data provenance that are conceptual and technological solutions to cope with data sovereignty challenges. We introduce a common scenario for the Industry 4.0 age, in which a supplier and an original equipment manufacturer (OEM) are exchanging data to mitigate risks in the supply chain management. We describe the difference between access control and usage control, the usage control concepts and related concepts such as digital rights management or user managed access.

We present the implementation of data usage control in the IDS. In doing so, we present possible integration layers and its integration with the IDS Reference Architecture Model. In addition, we also present the IDS Usage Control Object as a way to transfer usage control metadata between connectors.

We explain the topic of policy specification in the IDS by presenting the information model and policy language, the IDS policy classes and the IDS policy editor, which supports data owners to expressing usage restrictions. Furthermore, we show how we handle policy transformation to machine-readable policies as well as policy handshake and negotiation in the IDS.

As there are different ways to implement data usage control, we present three approaches researched and developed within Fraunhofer: The MYDATA Control Technologies, the Logic-based Usage Control and Degree. Every technology is presented in detail including its integration concepts. Finally, we compare these technologies and discuss them. We address data provenance as additional concept to data usage control to cope with transparency and accountability.

We conclude the document by elaborating on the current state and presenting future work, like the role of usage control in the App Store and automated IDS contract negotiation.

Keywords: MYDATA, IND²UCE, LUCON, Degree, Data Usage Control, Data Provenance, Information Model, International Data Spaces, IDS, FDS, CCIT

Table of Contents

1. What's new?

In the International Data Spaces (IDS) we are constantly refining and improving our data sovereignty methods and technologies. The last release of our Position Paper was already more than one year ago – a long time in field that is the focal point of many discussions. Since our last release, we refined many aspects in the field of data sovereignty.

In this chapter, we will provide you with a short overview of what we changed throughout the document so that you are able to quickly navigate through the new content in this document.

While updating this document, we noticed that several sections grew substantially and thus decided to slightly change the structure of the document. Besides this chapter, we introduce an own chapter for the field of policy specification (Chapter 5, page 30). Some topics from the future work subsection have evolved and are now ready to be used, so we moved them into their respective chapters and added new topics to the future work. Besides these changes, the overall structure stays the same.

In Section 3.6 (page 17) we provide two new sections where we discuss other related concepts like the Solid Access Control Model and the Open Policy Agent (OPA) and how they relate to our data sovereignty approach in the IDS.

In Chapter 4 (page 22) we explain Usage Control specific terminology used in IDS Contracts (Section 4.1) and introduce the IDS Usage Control Object (Section 4.4). We updated all sections that describe the integration with the Reference Architecture Model (Section 4.5).

Chapter 5 (page 30) is, as already stated above, a new chapter that contains old topics, that have been in Chapter 0 before. All sections have been updated with new information. The descriptions of the IDS policy classes were moved into the Appendix (Section A.1). We added Section 5.5 to introduce the topic policy handshake and negotiation. Automated policy negotiation, that may be introduced to the IDS in future, is briefly described in the future work subsection 8.2.2.

In Chapter 6 (page 40) almost all sections are updated. Subsection 6.1.1 introduces the MYDATA Usage Control App. While D° (Section 6.3) received major updates, LUCON (Section 6.2) received minor updates.

Chapter 7 (page 67) received minor updates and introduces two new Sections about provenance tracking with MYDATA and the transformation of IDS Usage Control policies.

Throughout the document, we clarified the usage of IDS/ODRL Policy Language, IDS Contracts, Usage Policy, IDS Usage Control Language.

2. Introduction

The International Data Spaces are about to create data spaces where businesses can exchange and exploit data in a secure manner. For the IDS as well as other data-driven businesses, data sovereignty is a key success factor. Data sovereignty has the goal to provide a Data Owner [1] with full control over her data. This includes being able to control the usage of her data by the Data Consumer.

In this document, we present data usage control and data provenance as conceptual and technical solution to cope with data sovereignty.

2.1 Motivation and Problem

Nowadays, business is spurred by continuously exchanging information between business partners. However, data is typically secured by access control mechanisms only. After access to data has been granted by these mechanisms, data can be arbitrarily altered, copied and disseminated by the recipient. Data usage control offers possibilities to control future data usages beyond the initial access (also known as obligations).

In the age of Industry 4.0, there is more critical and sensitive data exchanged between business partners (see Figure 1). In general, companies have intrinsic and extrinsic motivations to apply usage control: On the one hand, companies may use usage control to prevent misuse of their own data, to protect their intellectual property, and to preserve the data value (intrinsic motivation). On the other hand, companies have to comply with legal obligations such as the European Union General Data Protection Regulation EU-GDPR (extrinsic motivation). Hence, companies have to prevent misuse of other persons or companies' data.

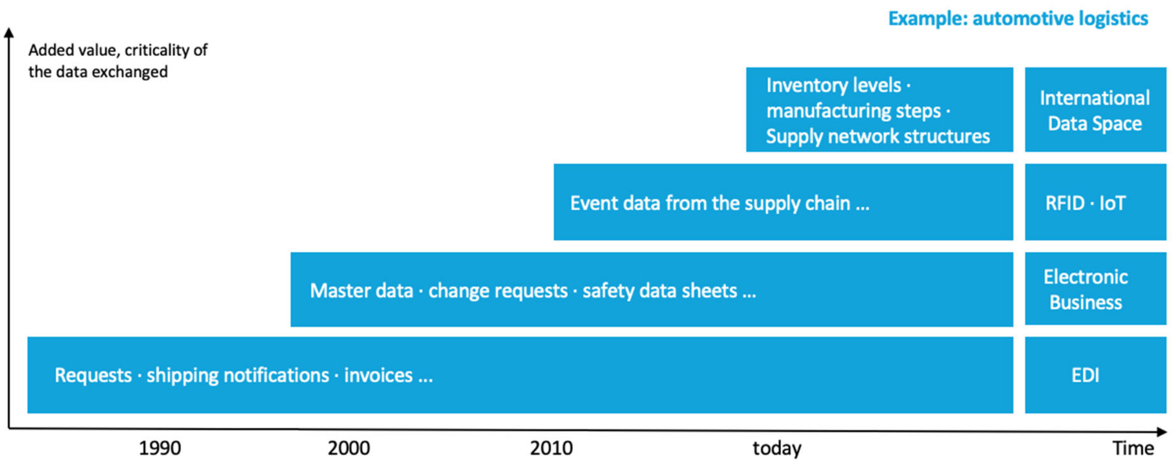

Example: automotive logistics
Added value, criticality of the data exchanged
Inventory levels · manufacturing steps · Supply network structures
International Data Space
Event data from the supply chain ...
RFID · IoT
Master data · change requests · safety data sheets ...
Electronic Business
Requests · shipping notifications · invoices ...
EDI
1990
2000
2010
today
Time

Figure 1: Data exchange in the age of industry 4.0

2.2 Accompanying Scenario: Supply Chain Risk Management

The following subsection presents our application scenario for data usage control. In the age of globalization and high-cost pressure, supply networks of automotive original equipment manufacturers (OEM) are complex and interference-prone for risks (e.g., earthquake, fire, war). For that reason, supply chain risk management (SCRM) becomes more and more important for a high supply reliability.

Figure 2 illustrates the data exchange between a supplier and the OEM in a collaborative SCRM scenario. On the one hand, there is data flowing from the suppliers to the OEMs such as affected parts and sub-supplier information, which the OEMs use in their supplier management system. On the other hand, the OEMs send data such as part demands or inventory range to the suppliers, which the suppliers process in their risk management system.

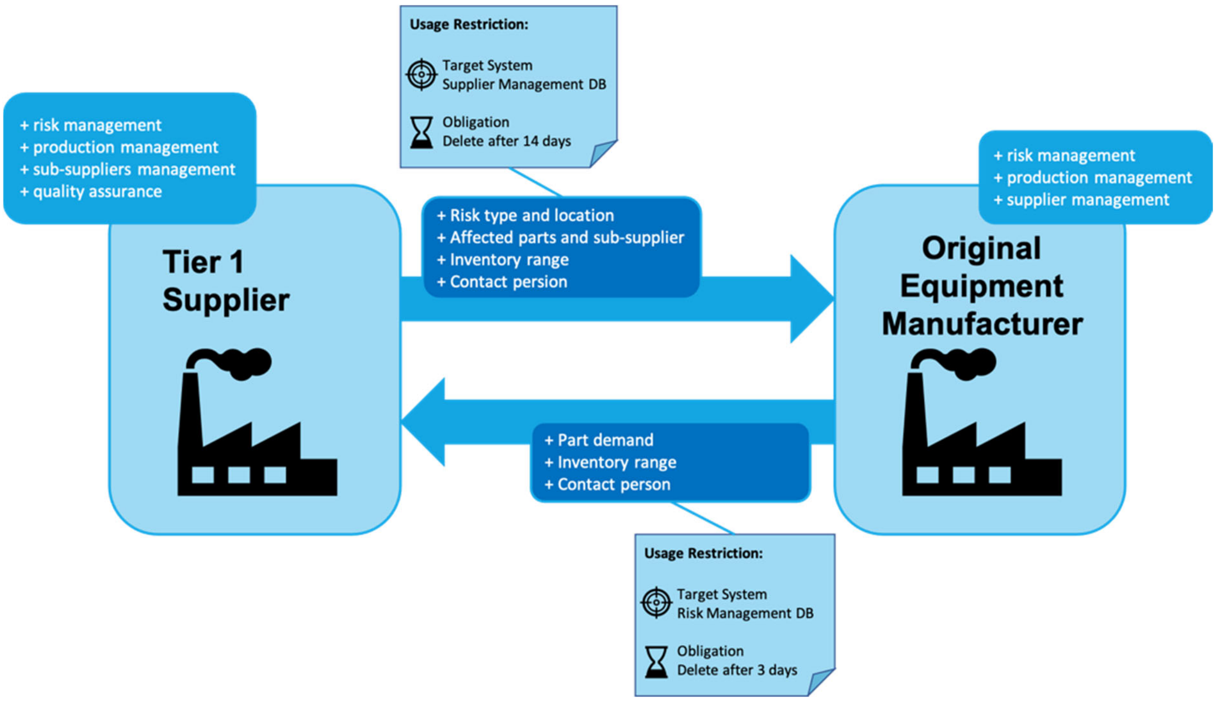


Figure 2: Supply Chain Risk Management illustration

Nowadays in the SCRM processes, most of the communication between the OEMs and the suppliers is done via phone, email, or web conferences. Table 1 shows the attributes of the data exchange from the supplier and OEM perspective (supplier as data provider and OEM as data provider).

Table 1: Attributes of the data exchange from the supplier and OEM perspective

From/To	Supplier	OEM
Supplier		• Risk type and location • Affected parts and sub-supplier • Inventory range • Contact person
OEM	• Part demand • Inventory range • Contact person	

In the process, there is sensitive and valuable data provided by the supplier as well as by the OEM: For example, data about the sub-supplier is very sensitive for the supplier. With such data, the OEM could skip the supplier and purchase directly form the sub-supplier. The part demand and inventory range are sensitive data for the OEM, because they make the production volume and warehouse transparent.

An automation of the data exchange in the SCRM process would lead to time and money savings for suppliers and OEM. In this case, the systems must ensure that the exchanged data is compliant with the company policies. This is where usage control can be used as technical extension to a contract to technically enforce the policies of the respective data provider. In fact, usage control improves security by controlling the data usage on the target system. Examples for appropriate policies in natural language are:

- The OEM can only use supplier data for risk or bottleneck management, but not for purchasing or sales purposes.
- The OEM has to delete all exchanged data in the SCRM process after 14 days.
- The supplier has to delete all exchanged data in the SCRM process after three days.
- The supplier can only import data from the OEM into the system "risk management".

2.3 Document Structure

We structure the remainder of the document as follows:

Chapter 0 addresses the difference between access control and Usage Control as well as related concepts such as Digital Rights Management and Windows Information Protection. In addition, we introduce the basic concepts of usage control comprising the technical enforcement, decision making and information retrieval and policy specification, management and negotiation.

In Chapter 0, we address the general implementation of Usage Control in the IDS. We discuss the different stages of Usage Control in the Usage Control Onion. In addition, we describe the Usage Control Object. Finally, we describe the integration with the Reference Architecture Model.

Chapter 0 is about policy specification. We introduce the information model and the relation to the Usage Control policy language. Hence, we present the policy specification and the derived policy classes within the IDS as well as the policy transformation, that handshake and how to negotiate policies.

Chapter 0 is dedicated to the different Usage Control technologies that offer technical solutions to enforce our usage restrictions. We present the MYDATA Control Technologies (MYDATA), the Logic-based Usage Control (LUCON) and Degree (D°). In more detail, we present communication flow, the integration concept and the transformation of IDS usage control policies to the technology-dependent policies. Finally, we compare these technologies and conclude with a discussion.

We present Data Provenance in Chapter 0. Data Provenance is a complementary concept to the enforcement technologies to cope with transparency and accountability of data usages. Hence, we present the relation between Data Provenance and Usage Control, the Data Provenance principle, its architecture and how data provenance is reflected in the IDS Contracts.

In Chapter 0, we discuss our work by presenting capabilities, current limitations and implications of Usage Control. We end with future work such as the parameters of the policy negotiation process, next activities for Data Provenance and our implementation.

3. Usage Control Concepts

Usage control is an extension to traditional access control (see Figure 3). It is about the specification and enforcement of restrictions regulating what must (not) happen to data. Thus, usage control is concerned with requirements that pertain to data processing (obligations), rather than data access (provisions). Usage control is relevant in the context of intellectual property protection, compliance with regulations, and digital rights management.

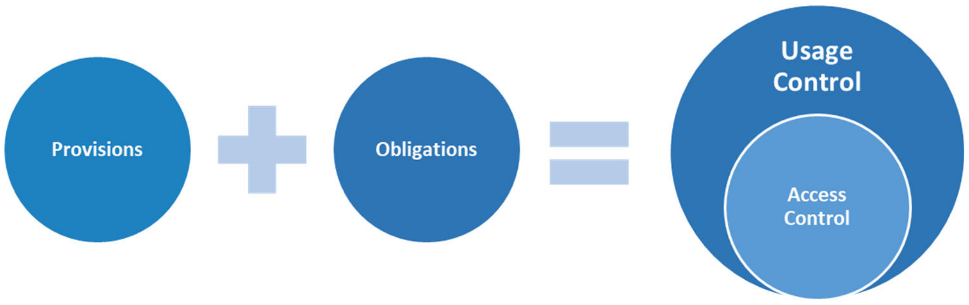


Figure 3: Usage Control consists of provisions and obligations

3.1 Access Control

In information security, access control restricts access to resources. The term authorization is the process of granting permission to resources. Several access control models exist, such as Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role-based Access Control (RBAC), Attribute-based Access Control (ABAC), etc. Although such a plethora of access control models exists, RBAC and ABAC are most commonly used.

We will use the XACML (eXtensible Access Control Markup Language) Standard [2] to introduce commonly used terms in the field of access control. XACML is a policy language to express ABAC rules. The main building blocks of the language are subject, action, resource and environment. The subject describes who is accessing a data asset (e.g., a user). The action describes what the subject wants to perform on the data asset (e.g., read, write). The resource describes the data asset. Finally, the environment specifies the context (e.g., time, location). Figure 4 illustrates the data-flow model of XACML and the main actors or components to implement it: Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Information Point (PIP), and Policy Administration Point (PAP).

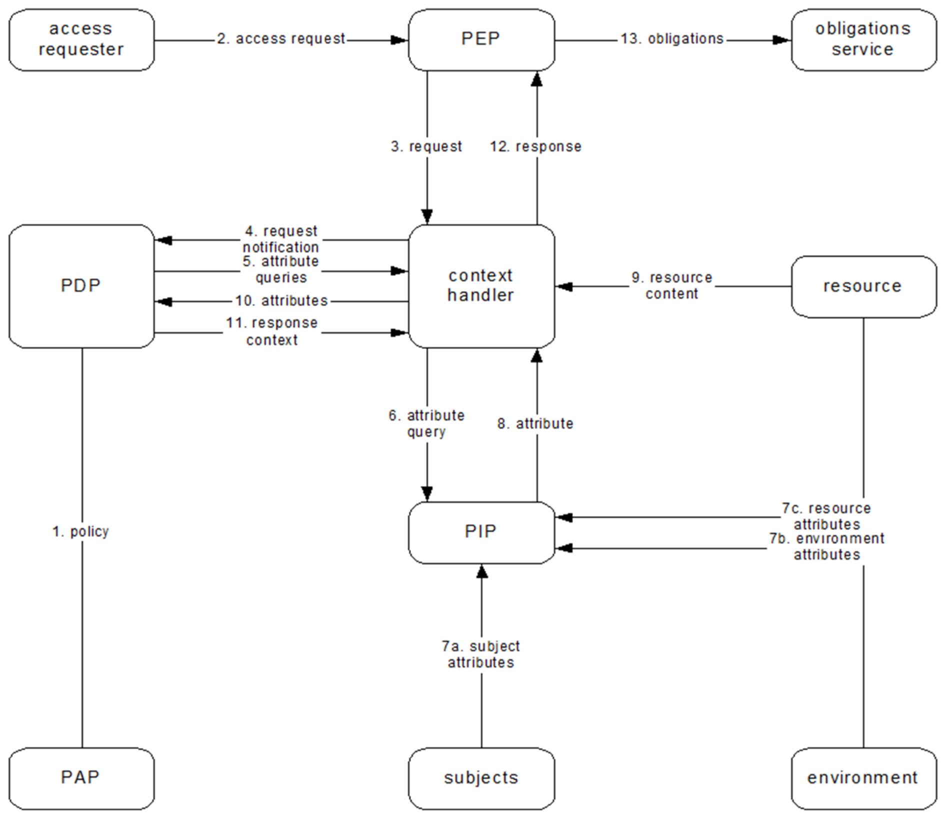


Figure 4: XACML data flow illustration

In general, attributes can describe anything and anyone, but tend to split into four categories:

- **Subject attributes:** Attributes that describe the user by e.g., age, role or clearance.
- **Action attributes:** Attributes that describe the action attempted e.g., read, delete or view.
- **Resource (or object) attributes:** Attributes that describe the resource itself e.g., object type, location or classification.
- **Contextual (environment) attributes:** Attributes that address time, location or other dynamic aspects.

Access control in the IDS is a resource-centric regulation of access requests from subjects (i.e., IDS participants) to resources (i.e., data services). Resource owners define attribute-based access control policies for their endpoints and define the attribute values a subject must attest in order to grant access to the resource. These attributes may include:

- Specific identity of connector(s) (only access requests from a specific connector or specific connectors will be granted)
- Connector attributes (only access requests from a connector that possesses specific attributes will be granted)

- Security profile requirements (only access requests from a connector that fulfils specific security feature requirements will be granted, e.g., having a TPM >= 1.2 and doing application isolation)

The actual access control decision has to be taken within the connector and can be realized using technologies such as XACML or JAAS, depending on the implementation of the connector. The IDS security architecture does not dictate a specific access control enforcement language or implementation. A discussion paper about access control for industry 4.0 can be found here [3].

3.2 Usage Control

In contrast to access control, where access to specific resources (e.g., a service or a file) is restricted, the IDS architecture additionally supports data-centric usage control. In general, the overall goal is to enforce usage restrictions for data after access has been granted. Therefore, the purpose of usage control is to bind policies to data being exchanged and to continuously control the way how messages may be processed, aggregated, or forwarded to other endpoints. This data-centric perspective allows the user to continuously control *data flows*, rather than *accesses to services*. At configuration time, these policies support developers and administrators in setting up correct data flows.

At runtime, the usage control enforcement prevents IDS connectors from treating data in an undesired way, for example by forwarding personal data to public endpoints. Thus, usage control is both a tool for system integrators to ensure they are not building an architecture that violates security requirements, and an audit mechanism, which creates evidence of a compliant data usage.

The following examples illustrate security requirements that cannot be achieved using traditional access control, but rather require data-centric usage control:

- **Secrecy:** Classified data must not be forwarded to nodes which do not have the respective clearance.
- **Integrity:** Critical data must not be modified by untrusted nodes as otherwise their integrity cannot be guaranteed anymore.
- **Time to live:** A prerequisite for persisting data is that it must be deleted from storage after a given period of time.
- **Anonymization by aggregation:** Personal data must only be used as aggregates by untrusted parties. A sufficient number of distinct records must be aggregated in order to prevent deanonymization of individual records.
- **Anonymization by replacement:** Data which allows a personal identification (e.g., faces in camera images) must be replaced by an adequate substitute (e.g., blurred) in order to guarantee that individuals cannot be deanonymized from the data.
- **Separation of duty:** Two data sets from competitive entities (e.g., two automotive OEMs) must never be aggregated or processed by the same service.
- **Usage scope:** Data may only serve as input for data pipes within the connector but must never leave the connector to an external endpoint.

It is important to note that the purpose of usage control is to allow the specification of such constraints and enforcing them in the running system. It is a prerequisite to usage control that the enforcement mechanism itself is trusted, i.e. usage control itself does not establish trust in an endpoint. It rather builds upon an existing trust relationship and facilitates the enforcement of legal or technical requirements such as service level agreements (SLA) or data privacy regulations. Thus, users must be aware that usage control will only provide certain enforcement guarantees if applied on highly trusted platforms, such as Trusted Connectors in the IDS (see [4]).

3.3 Enforcement

For enforcing usage restrictions, data flows need to be monitored and potentially intercepted by control points (i.e., PEPs). These intercepted data flows are given to the decision engine (i.e., the PDP) for requesting permission or denial of the data flow. In addition to just allowing or denying the data flow, the decision can also require a modification of data. A PEP component encapsulates the enforcement.

Regarding our accompanying scenario, OEM and supplier demand the deletion of data after a certain time or that only a limited audience can access the sensitive data. Hence, we have to intercept the data flow and check which audience (i.e., processing system) is using the data. For example, the supplier demands the OEM that only the supplier management system can use the data.

3.4 Decision and Information

The enforcement relies on a decision. A Policy Decision Point (PDP) takes the responsibility to answer incoming requests (i.e., data flows) from a PEP with a decision (see Figure 5). The decision-making based on usage restrictions is also called (policy) evaluation. There are several evaluation possibilities such as event- (see Section 6.1), or flow-based approaches (see Section 6.2 and 6.3).

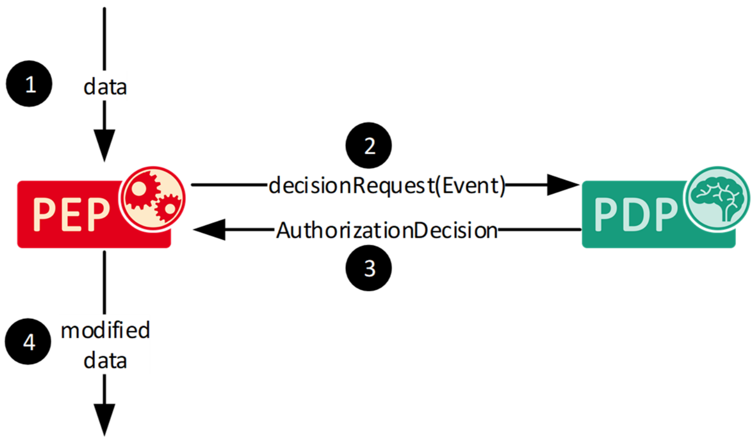


Figure 5: Illustration of a PEP intercepting data with decision making (PDP)

For event-based systems, data usage occurrences are represented as events including attributes to characterize the data usage. The event processing can be differentiated in simple processing (e.g., event-condition-action paradigm) and stream processing (e.g., sliding window) of events. The terms “event stream processing” and “complex event processing” are often used interchangeably.

In our accompanying scenario, we can model the transition of data as event with attributes about the data itself and the recipient. The attributes contain metadata and the target system (e.g., supplier management system). Taking our example from the previous section, the decision engine would draw a deny decision if the target system does not correspond to the expected supplier management system.

The policy decision may also depend on additional information that is not present in the intercepted data flow itself. This includes information about contextual information such as previous data usages or the geographical location of an entity. There is also the possibility for pre- or post-conditions that have to hold before (e.g., integrity check of the environment) and after (e.g., data item is deleted after usage) the decision-making. In addition, there is the possibility to define on-conditions that have to hold during usage (e.g., only during business hours). These conditions usually specify constraints and permissions that have to be fulfilled before, during, and after using the data (see Figure 6).

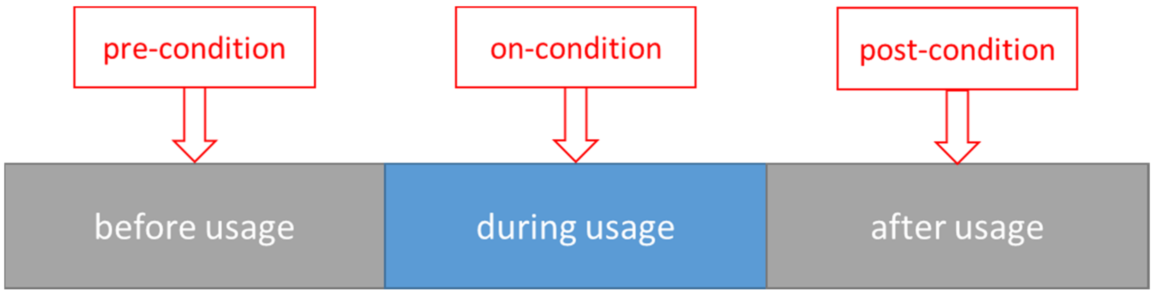


Figure 6: Types of conditions and when they are enforced

A Policy Information Point (PIP) provides missing information for the decision-making. In addition, we can use such a component to get contextual information for or about the intercepted system action (e.g., data flow information, geolocation of the requesting device).

Regarding our accompanying scenario, we may transform the D-U-N-S number [5] of a supplier to a concrete supplier name and address information. For example, if we want to limit the use of data depending on the geolocation of the supplier, a PIP can resolve the D-U-N-S number to a postal address and finally the postal address to GPS coordinates. Supplier and OEM are usually using different part numbers. Therefore, another example for a PIP is the translation of supplier part number to OEM part number and vice versa.

Finally, there is the concept of a Policy Execution Point (PXP). A PXP is used to perform additional actions based on the policy rules, such as sending an email when data is used or writing to a specific log system. Figure 7 illustrates an exemplary sequence of all processing steps to enforce usage control restrictions on a data flow:

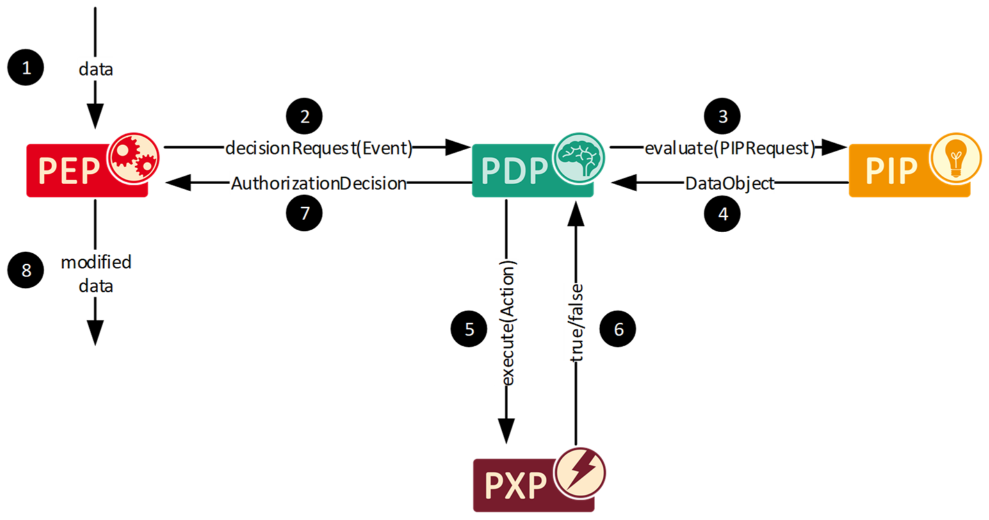


Figure 7: Full illustration of a usage-controlled data flow

- 1. PEP intercepts the data flow
- 2. PEP transforms the data flow to a decision request and sends that decision request to PDP
- 3. PDP starts the policy evaluation and invokes a PIP to retrieve additional information
- 4. PIP responds with the requested data to the PDP
- 5. PDP triggers an additional action at a PXP
- 6. PXP confirms that the action succeeded to the PDP
- 7. PDP sends authorization decision to the PEP
- 8. PEP enforces the decision on the intercepted data flow

3.5 Specification, Management, and Negotiation

Another important aspect of usage control is the specification and management of usage restrictions. Data providers have to express their restrictions on their data in a more or less formal way. For a technical enforcement, the specification must produce a machine-readable output. The Policy Administration Point (PAP) is the entry point for specification of usage policies, often via a user-friendly graphical interface.

In our accompanying scenario, the Collaborative Supply Chain Risk Management (CSCRM) App takes the role of the PAP. There is a version for the supplier and a version for the OEM to specify their data usage restrictions.

A Policy Management Point (PMP) administers the usage restrictions. Hence, the component is concerned with the policy life cycle. This includes the instantiation, negotiation, deployment, and revocation of usage restrictions, as well as conflict detection and resolution.

There are two ways where usage restrictions are placed. First, usage restrictions can be adhered to the data, which is also called sticky policy [6]. Sticky policies are one way to cope with the distribution of the usage restrictions. In this approach, machine-readable usage restrictions (policies) stick to data when it is exchanged. There exist different realization possibilities. Usually, data is encrypted and can only be decrypted when the adherence to the usage restrictions are guaranteed. Second, policies can be stored independently from the data, for instance, in a central component (i.e., a PMP/PRP). In this case, the management component has to take responsibility to exchange the usage restrictions between different systems.

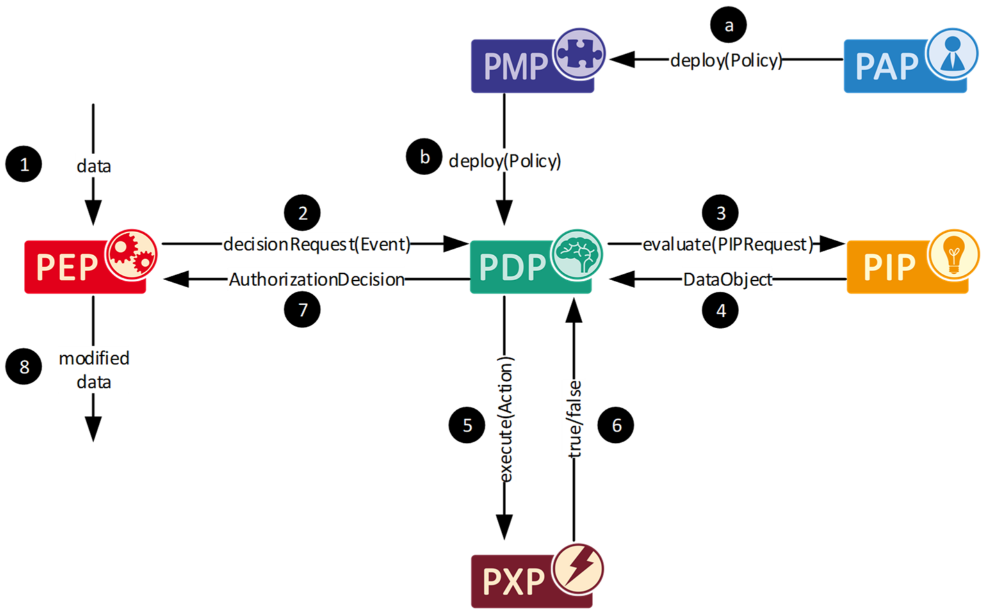


Figure 8: Usage Control illustration with PMP and PAP

In Figure 8, the PAP and PMP interactions are represented in the steps a) and b) to illustrate the deployment as policies as an example sequence.

The management of usage policies becomes especially important when exchanging data across system boundaries. Every time data crosses system boundaries, the target system must be prepared for the protection of incoming data, that is, the corresponding policies need to be deployed. The resulting negotiation of policies is also part of the policy management. As enforcement mechanisms can work differently (e.g., work on different system actions) on different systems or technologies, abstract policies can have different instantiations. Hence, usage policies must be instantiated on the target system.

3.6 Related Concepts

There are related concepts to cope with data sovereignty challenges. We present them in the following subsections.

3.6.1 Access Control

In general, data sovereignty challenges at the providing endpoint may be solved by access control technologies. As its name already implies, access control is suitable to handle the access to data but has drawbacks in terms of data usages (as described in

Section 3.2). However, a limited set of usage restrictions can also be handled by access control technologies.

3.6.2 Data Leak/Loss Prevention

Data Leak/Loss Prevention (DLP) technologies detects and prevents potential data breaches by monitoring sensitive data. Commonly used are Endpoint DLP solutions that run on the client's operating system (e.g., as extension or feature of a security suite). In addition, there are also DLP solutions available that are monitoring the network or access to central storage devices.

3.6.3 Digital Rights Management

The term Digital Rights Management (DRM) is frequently used in the area of protecting digital content from unintended use, modification, and distribution. Different DRM technologies exist to protect multimedia content such as movies (e.g., DVD, Blu-ray), music (e.g., Audio CDs, Internet music), television, or E-books. In addition, there exist DRM technologies to protect digital documents (e.g., MS Word, PDF) within enterprises. This kind of DRM is also known as enterprise rights management (ERM) or information rights management (IRM) and aims to control of access and use of corporate documents.

3.6.4 User Managed Access

The purpose of User Managed Access (UMA) is to empower the resource owner to control the authorization of data sharing. It is often used to protect resources between online services on behalf of the owner. OAuth-based access management systems are representations of UMA. Several open-source implementations exist that follow the UMA core protocol.

3.6.5 Windows Information Protection

Microsoft introduced several technologies to establish a comprehensive information protection in their operating system and software such as Microsoft Office (e.g., BitLocker, Windows Information Protection (WIP), Office 365 and Azure Information Protection) [7]. WIP, for instance, is an integral part of Windows 10. Goals of the WIP are to protect data on own devices, to separate private and business data (data separation), to prevent unauthorized access and use (data leakage protection), and to protect data when shared. WIP-protected documents can only be used in WIP-compliant apps. For example, WIP prevents pasting sensitive information (e.g., by using ctrl+c and ctrl+v) to non WIP-compliant apps.

3.6.6 Solid Access Control Model vs. IDS Usage Control Language

The IDS Usage Control language from the IDS Information Model contains the necessary concepts and attributes to describe rich usage permissions and how to enforce them. These concepts follow the very holistic scope of an overall usage control perspective. Solid, the approach for a truly decentralized social network of self-controlled data pods aims at giving human users the sovereignty of their personal data. The Solid promoted data protection solution – the Web Access Control language (WAC, [8]) - differs slightly from the IDS concepts. First, its scope is restricted to pure access control. The expressiveness of the protection rules, an Access Control List (ACL) or `acl:Authorization`, ends as soon as the data resource is requested by the consumer. Following usage activities or even the further distribution of the resource is not reflected.

Nevertheless, both approaches follow the same identification pattern. The data resources and the Authorization (Usage Contract) related to them follow the URI schema, usually as http URIs. While ACLs in general used also in many other environments and are open to any - also non-URI – identifier scheme, the special usage of Web Access Control language in Solid requires the compliance towards the Linked Data Platform (LDP) specification. `acl:Authorizations` determine the permitted RESTful operations on the (LDP) target resource, and can also be accessed as own LDP Resources. As such, one can even think about `acl:Authorizations` managing the access to other `acl:Authorizations`, something which is not possible for IDS Usage Contracts.

The WAC language does not allow the specification of a resource owner. This is the same approach as in the IDS model. The authorized entity that has the technical ability to control the resource itself and to define its usage permissions is implicitly assumed to be the sovereign of this resource. This procedure somehow solves the challenge of ownership by relying on the technical capabilities of users or user agents. One has to note that such indirect proceedings are necessary as an ownership concepts - as for physical goods - is not possible or applicable for digital data. The thereby created implication is that the Data Sovereign is the one who has the technical capability and permission to manage the access to a source, and that the access permissions and control rights are granted to the Data Sovereign. Obviously, this logical circle requires more in-depth examination.

One further challenge is the definition of user groups. The range starts with a group of the size one, which is equivalent to an individual user. In an WAC statement, this user is identified by its WebID. On the other end of the possible descriptions is the group including everyone, which is expressed by the class `foaf:Agent`. As everyone, either human or non-human, organization or individual, is by definition an instance of `foaf:Agent`, a permission assigned to this class applies also to everyone.

Between these somehow trivial cases are the more relevant defined groups, for instance the members of a certain organization or company. The recommended encoded for such groups is the use of URLs. The Web resource of the group URL shall provide a list of its members. The access management service therefore needs to dereference the group identifier, and thereby receives all required information to accept or decline the access request. The IDS handles such groupings completely different. Its business to business focus leads to the view of all users or organizations as instances of the generic `ids:Participant` class. Recursive declarations then allow the descriptions of memberships. A department is modeled as a Participant which is part of the bigger company, also an IDS Participant.

The core challenge for both approaches is the clarification whether or not a user is a valid member of a group or organization. In the terminology of XACML, a Policy Information Point for this type of information is needed. The WAC specifications solves it by using the identifier also as the reference to the PIP interface - the group URI is also the pointer to the Web resource containing its members. That is not possible in the IDS where membership information requires the highest protection level and must not be open to arbitrary Web requests. The respective PIPs in an IDS ecosystem are therefore globally known components like the Participant Information Service ParIS, or company-specific like an LDAP system.

Defined by the scope of their language, the WebAccessControl language can only describe the limited amount of access operations - read, write, append, and control. The IDS Usage Control language must also be able to express activities after the granted request, for instance to not distribute, log, notify about usages, or even to delete the resource from the receiving system. Obviously, such obligations cannot be ensured by the providing server but require a trusted system at the receiving party. While the IDS includes specifications and guarantees to achieve this, it is not in the scope of Solid or the WAC language.

An additional difference can be found in the inheritance regime of the WAC and the IDS Usage Control language. Solid ACL files are interpreted relative to their location in LDP Containers. That means, that the same permissions apply for contained container and resources. As the IDS does not follow a strict container model, such passing of rules is not possible. Each `ids:Resource` needs its own assigned usage policy. Nevertheless, the IDS follows a similar concept when it comes to the appearances of a data asset. By default, policies specified for the `ids:Resource` are propagated through the related `ids:Representations` to the final `ids:Artifacts`. Solid misses this differentiation, and consequently has only one way to describe the target data asset.

Another significant difference lies in the embedding into HTTP headers for the runtime discovery of ACL files. As the IDS supports several protocol bindings, the limitations to RESTful discovery operations (GET, HEAD) are not sufficient. Potential consumers therefore need to use infrastructure services, in particular the IDS Metadata Broker, or directly the resource self-description at the hosting connector to find the applying Usage Policies. As the Solid interactions are only enabled through the LDP operations, the discovery mechanism using Link headers are sufficient and no third-party component is required.

3.6.7 Open Policy Agent

Open Policy Agent (OPA) is a relatively novel approach of the Cloud Native Computing Foundation (CNCF), supporting authorization mechanisms for infrastructure services such as Kubernetes. It is already used in production at over 20 organizations. OPA internally relies on a PEP/PDP based structure and thus is structurally compatible with the approach described in this document, besides some exceptions: OPA does not support standardized external Policy Information Points and relies on external data being pushed into the OPA server. Furthermore, OPA does not support Usage Control but is based on authorizations. The policy language supported by OPA is called REGO. Neither REGO nor OPA are standardized yet, although REGO is inspired by Datalog, a well understood, old query language. IDS concepts are moving conceptually into the direction of cloud services (e.g., based on Kubernetes). Accompanying this development could be an integration of the IDS concepts into OPA. However, some challenges remain:

- OPA and REGO are not standardized and thus remain moving targets for implementation.
- OPA does not support usage control and data lifecycles, so these concepts would have to be implemented.
- REGO would have to be replaced or totally modified to support the usage policy classes defined for IDS interactions.

A policy engine such as OPA is well suited for Authorization management in a Kubernetes based environment, but several aspects are missing for Usage Control as discussed here.

4. Implementation of Usage Control in the IDS

The chapter describes the implementation of data usage control in the IDS in general, starting with an explanation of important Usage Control terms in IDS Contracts, followed by a brief overview about the two general activity streams.

4.1 Usage Control Terms in IDS Contracts

Throughout the document, we use several Usage Control specific terms related to IDS Contracts. In Figure 9, we show how these terms fit together.

An IDS Contract is *implicitly* divided to two main sections: the contract specific metadata and the *IDS Usage Control Policy* of the contract.

The contract specific information (e.g., date when the contract has been issued or references to the sensitive information about the involved parties) has no effect on the enforcement. However, the *IDS Usage Control Policy* is the key motive of organizational and technical Usage Control enforcement.

Furthermore, an IDS Usage Control Policy contains several Data Usage Control statements (e.g., permissions, prohibitions and obligations) called *IDS Rules* and is specified in the *IDS Usage Control Language* which is a technology independent language. The technically enforceable rules shall be transformed to a technology dependent policy (e.g., MYDATA) to facilitate the Usage Control enforcement of data sovereignty. Listing 1 in Section 5.5 shows an example of an IDS Contract.

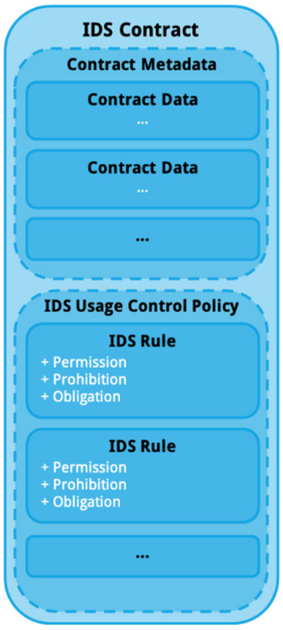


Figure 9: Usage Control terms in IDS Contracts

4.2 Overview

There are two main activity streams within the IDS to implement data usage control:

First, a policy language to express data usage restrictions is developed. The policy language is descriptive, technology-independent and based on the Open Digital Rights Language (ODRL) and further detailed in the form of IDS contracts. To express usage restrictions within the IDS, there are several predefined classes that express the most commonly data usage restrictions. This activity stream is addressed in chapter 0.

Second, usage control technologies are developed to enforce these usage restrictions at technical level. We differentiate between proactive and retrospective technologies. The proactive technologies enforce provisions and obligations across system boundaries during runtime. It controls the data usages and is called preventive enforcement. The retrospective technologies are rather monitoring and recording technologies, but do not prevent or actively control any data usages. Therefore, it does not prevent undesired data usages and is called detective enforcement. The usage control technologies are described in chapter 0.

The following subsections addresses the different expansion stages of usage control enforcement: the usage control onion and the IDS Usage Control Object. The chapter ends with the integration of usage control with the Reference Architecture Model. The policy language is not addressed in this document. We refer the interested reader to [9] and to [10] for more details on the ODRL language and to [11] for more details on IDS Information Model.

4.3 The Usage Control Onion

We can characterize and implement the enforcement of data usage restrictions in different shapes. Organizational rules or legal contracts can be substituted or at least accompanied by technical solutions, which introduce a new level of security. Vice versa, technical solutions can be accompanied by organizational rules or legal contracts to support the overall goal achievement (e.g., to compensate missing capabilities of the technical solution).

Although it is a commonly used solution to solve usage control restrictions with organizational rules, we will focus on the technical enforcement in this document.

4.3.1 Technical enforcement, organizational rules and legal contracts

Usage control can be implemented in different ways. The solutions range from organizational rules or legal contracts to complete technical enforcement of usage restrictions. Intermediate levels may contain parts of both enforcement manifestations. We will describe a transition of enforcing usage restrictions from organizational rules/legal contracts to a complete technical enforcement that we align to our accompanying application scenario.

Usage control should be seen as a machine-readable contract, which is expected to be fulfilled by a party. It is a way to track and trace data as it is used within different systems and to collect evidence of the violation of agreed usage constraints. With that in mind, solutions range from organizational rules or legal contracts to a completely technical enforcement of usage restrictions. For example, an organizational rule could state that the company rules forbid using removable storages such as USB sticks. Similarly, a technical enforcement such as group policies by the windows operating system can prevent the employees from using removable storage media. In some scenarios, we can interchangeably use organizational rules/legal contracts and technical enforcement. In other scenarios, we can use both enforcement forms to complete each other. In the long term, we assume a substitution of organizational rules/legal contracts by technical enforcement (as illustrated in Figure 10).

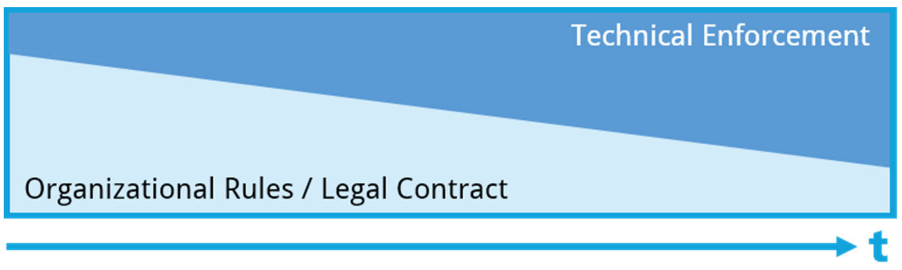


Figure 10: Technical Enforcement vs. Organizational Rules

Although it is not in the focus of this document and for the sake of completeness, we can express usage restrictions as organizational rules or as part of a legal contract between two companies. In this case, we have no technical measure, but may enforce some violations by disciplinary penalty or lawsuit. Regarding our accompanying scenarios, there is a legal contract between the two companies stating that the exchanged data can only be used in the specific target systems (i.e., supplier management, risk management) or for the purpose of predictive maintenance. Furthermore, the contract may state that data must be deleted after a certain time. For violations, the contract imposes fines that are a multiple of the total contract value. In this case, organizational measures are applied to enforce usage restrictions.

The following presents the different stages of usage control that we name the usage control onion, starting from the inner part of the onion, which is the IDS connector, and ending in the outer onion shells with external systems.

4.3.2 Usage Control within the IDS Connector

The inner part of the usage control onion is the IDS connector. Depending on the usage restrictions, they are applied at the data provider connector or at the data consumer connector.

At the data provider connector, usage control enforces policies such as how often data can be accessed, at what times (e.g., only within business hours), or that data must be filtered or masked (e.g., anonymized) before leaving the company. The usage restrictions at data provider connector are usually provisions that are technically handled by a PEP.

At the data consumer connector, usage control enforces policies that are usually obligations for the data consumer such as "data can only be used for fourteen days" or "data can only be used for the purpose of predictive maintenance". The technical enforcement is handled by a PEP or PXP, depending on the usage restriction. Limiting data flowing to a specific target system to ensure the correct usage purpose is handled by a PEP, the deletion of data in storage infrastructure outside the connector is handled by a PXP that performs the delete operation.

4.3.3 Usage Control within the Storage Infrastructure

The usage control enforcement can also be implemented at the storage infrastructure. Storage infrastructure may be any kind of storage to persist data such as a file system or a database. In general, there are two possibilities for usage control capabilities at the storage infrastructure.

First, the storage infrastructure is used without modification. In this situation, usage control is implemented by encrypting the data within the connector before transferring the data to the storage infrastructure. Using the data is only possible by using the IDS connector to decrypt the data. Hence, every usage is controlled by the IDS connector. In such cases, usage restrictions such as data lifetime or time constraints can be enforced by deleting the cryptographic key material. There are no changes at the storage infrastructure needed, it can be used as is. However, every usage of the data must be handled by the IDS connector, which could lead to a bottleneck.

Second, the storage infrastructure is enriched with usage control enforcement component (i.e., PEP) that controls the usage of the data. It offers all usage control enforcement capabilities offered by PEPs but demands changing or adapting the implementation of the storage infrastructure. However, such an adaptation may not be possible without support of the storage infrastructure vendor.

The two possible solutions are presented in the following Figure 11. Encryption and decryption are handled by the PEP in the connector (illustration on the left) or the usage of the data is controlled by the PEP inside the storage infrastructure (illustration on the right).

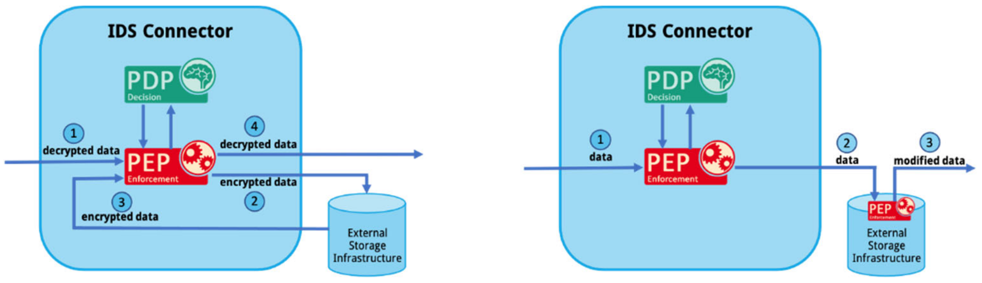


Figure 11: Usage Control within the Storage Infrastructure

4.3.4 Usage Control within the Application

The next onion shell is the application that uses the data. Hence, the data flow within the application has to be controlled and adhere to the usage restrictions. Similar to the storage infrastructure, a PEP can be integrated into the application that controls the data flows. In addition, the application is developed by using technologies such as D° to directly integrate usage control capabilities at compile time. Both approaches address usage control during development and therefore need support by the software vendor (as long as the software is not developed by the company itself or it is open-source software that the company is willing to adapt).

4.3.5 Usage Control within the Clients

However, there are still possibilities how data may be used without adhering to the usage restrictions. For example, the user may print the data or take a screenshot. To tackle this issue, the client operating system such as Windows, Linux, Apple iOS, or Android has to be adapted. The integration of usage control capabilities is still possible but demands deep knowledge about the operating system. In addition, not every operating system is open source. However, some vendors already started to implement security measures such as Windows Information Protection or the Android Enterprise (previously known as Android Device Administrator) that offer usage control capabilities (e.g., prevent printing, prevent screenshots, or prevent screen casting).

Finally, when data is flowing out of the Usage Control ecosystem, there are still possibilities how data may leak. For example, instead of making a screenshot, the users could take a picture of the screen by using a mobile device (i.e., external system or media disruption). Hence, we cannot achieve a perfect and comprehensive protection of data, but we can put controls to the system to reduce the possibilities for potential misuses.

To conclude, the enforcement of data usage restrictions can be characterized and implemented in different shapes. Organizational rules or legal contracts should be accompanied by technical solutions; and vice versa, technical solutions should be accompanied by organizational rules or legal contracts to support the overall goal achievement. In order to implement comprehensive usage control, we have to integrate control points into different systems and abstraction layers (see Figure 12) that are working together to achieve the overall goal of data sovereignty.

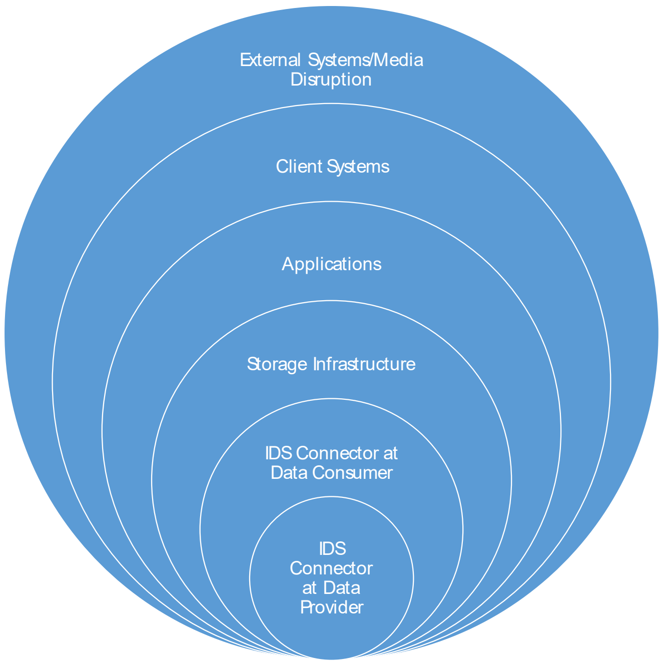

External Systems/Media Disruption
Client Systems
Applications
Storage Infrastructure
IDS Connector at Data Consumer
IDS Connector at Data Provider

Figure 12: Illustration of the Usage Control Onion

4.4 IDS Usage Control Object

A Connector needs to identify the data it transfers together with the rules it has to apply. In order to enable a connector to identify the data it transfers we introduce the IDS Usage Control Object. It contains metadata, including an identifier (such as a UUID) and the data as payload. The Usage Control technologies are using this identifier during the enforcement process to check, if there are policies registered for this unique identifier and applies them on the payload of the Usage Control Object.

In general, it is possible to allow or deny the entire object. If the object is allowed, policies can demand the modification of the payload.

When a Data Provider sends data to another connector, it stores that payload in a Usage Control Object, which also contains a unique identifier from the Data Provider. Moreover, additional (contextual) information like encryption state (and many more) is stored within the meta data description. A first version of the Usage Control Object is available in the information model.

4.5 Integration with the Reference Architecture Model

The subsection addresses the integration of data usage control with the reference architecture model [1]. In more detail, it describes the relation to other core components such as Identity Provider, Clearing House [12] and Metadata Broker [13]. In addition, roles such as data provider, data owner and data consumer are mentioned as well as context sources such as DAPS, ParIS and PIPs.

4.5.1 Identity Provider

IDS identities for both connectors and participants are created, maintained and revoked through Identity Providers. These components serve as the interface from the preceding certification processes to the technical onboarding in any data space. By supplying X.509 certificates, the Identity Providers serve the fundamental proofs for identity claims. Any interaction with connectors providing incorrect, expired or otherwise compromised identity tokens must be terminated instantly. Therefore, the Identity Providers supply the most basic protection against misuse and are the most basic trust provider. One part of the Identity Provider is the Dynamic Attribute Provisioning Service (DAPS). In addition to the long-term identity certificates, DAPS tokens constitute short-term proofs of several security and usage control related features. For instance, the compliance to certain security requirements is encoded in its attributes. In case these characteristics change at a connector, for instance because a new update supplies higher security features, an updated DAPS token reflects this development. The valuable X.509 certificate can stay untouched.

4.5.2 Clearing House

The Clearing House as a component in the IDS is a trusted third party, which can protocol interactions and events in the IDS. The goal is to log negotiation results, access requests or usage events. For the interaction with the Clearing House IDS connectors need a valid DAPS token. Currently, the data stored in the clearing house is encrypted using attribute-based encryption, which is basically a form of cryptographically enforced access control. It allows the encryption of data such that only someone with the correct attributes is able to decrypt the data. The policies in this access control mechanism are defined locally within the Clearing House. The integration of this rudimentary approach with the IDS usage control framework is planned for the future. More information about the IDS Clearing House is available in [12].

4.5.3 Metadata Broker

The IDS Metadata Broker is a purely informative component, intended to provide registry and discovery functionality to the IDS participants and components. The main task of the Metadata Broker component is the provisioning of search functionalities for data offerings. As such, a certain openness of the metadata provisioning is necessary. However, in some scenarios the knowledge of the existence of resources might already impose a threat to the Data Sovereign. Therefore, certain Metadata Broker implementations might be able to regard usage restrictions also for the received metadata. For instance, a Metadata Broker could hide the existence of a company's connector to its direct competitor - even though it provides this information to the company's own suppliers and customers. In contrast to Identity Providers or the Clearing House, no data protection mechanism can be expected by default. Nevertheless, some Metadata Broker may indicate such behavior through certified self-descriptions and signed attributes. More information about the IDS Metadata Broker is available in [13].

4.5.4 Data Provider, Data Owner and Data Consumer

According to the information model, a Data Owner (Data Sovereign) is a core participant of IDS who has complete control over the data and makes it available in the IDS and defines the terms and conditions of use of the data. A Data Provider is another Core participant of IDS who exposes the Data Sources via a Connector. A Data Provider may be an enterprise or other organization, a data marketplace, an individual, or a “smart thing”.

Moreover, a Data Consumer requests and uses the data provided by a Data Provider and a Data User is an IDS participant that has the legal right to use the data of a Data Sovereign as specified by the usage policy.

4.5.5 Context Sources

The provisioning of context information is essential in order to integrate external events or states into the usage control process. The IDS Reference Architecture Model does not explicitly define a general PIP component. However, certain security-related and participant-related features are supplied by the Identity Provider, more specific through the Participant Information Service (ParIS) and the Dynamic Attribute Provisioning Service (DAPS).

The DAPS combines identity proofs of IDS components with additional information about their characteristics and abilities. As one of the mandatory components in any IDS data exchange, the DAPS is crucial to inform each party about its communication partner and its most relevant features. In order to do so, the DAPS issues a specific form of JSON Web Tokens, so-called Dynamic Attribute Tokens (DAT). Each IDS interaction requires the exchange of the respective DAT of each actor, thereby giving the opposite party the possibility to verify each other. The DATs are immutable, as only the DAPS is allowed to sign the tokens, and each connector can easily verify the validity of the provided DAT signature.

The ParIS further extends the provisioned features contained in a DAT with more information about the corresponding IDS participant. Legal information like the registration number, the VAT ID, or according contact details. The ParIS is usually equipped through a trustworthy entity, for instance the operator of an Identity Provider or any other party responsible to onboard IDS participants.

Some usage control frameworks are offering context information by default, like date and time. If there are standardized Policy Information Points (PIP), it enables a connector to process various context information. However, there is the open question, how trustworthy this information is, and how it could be made trustworthy. An example would be a connector app with a Policy Information Point that offers the purpose of this app. This information could be used to apply a policy, which only allows the use of data for a specific purpose (e.g., transfer the data only, if the purpose of the app fits). PIPs can connect an internal or an external information source, but the interfaces have to be defined to be reusable.

For the most part of the policy classes the IDS defines also PIP interfaces. This means that required request parameters and response data are defined. We are currently researching the usage of ontologies to describe PIP definitions, but this will be not complete and domain specific in the first step.

Another source of context information is the Usage Control Object, which has been described in Section 4.4.

5. Policy Specification

The IDS includes several use cases such as IDS Connector, Digital Supply Chain, Smart Urban Mobility, Intelligent Sensor, Interconnected ESN and so on in which the data is exchange between the IDS parties. These use cases lead to variety of Data Usage Control policies. An important step towards controlling the usage of the data is the policy specification. The Data Providers of the IDS need to specify their Data Usage Control policies, although, they are from different technical backgrounds. A policy specification dashboard can support the customers in the process of policy specification.

Figure 13 illustrates the process of policy specification in IDS. The IDS Data Providers and Data Consumers shall use the IDS policy editors (i.e., Policy Administration Points) to specify their Data Usage Control policies and consequently, create their IDS Contracts. The outcome of the IDS policy Editor is a machine-readable contract specified in IDS policy language. The Data Usage Control statements of the IDS Contract shall be negotiated on a semi-automated policy negotiation platform. There, the involved partied agree on a contract that address their demands and benefits. After all, the Data Usage Control statements of the agreement contract shall be transformed to a technology-dependent language (e.g., MYDATA, LUCON, etc.) which is interpretable by a Data Usage Control technology and can be enforced to the systems.

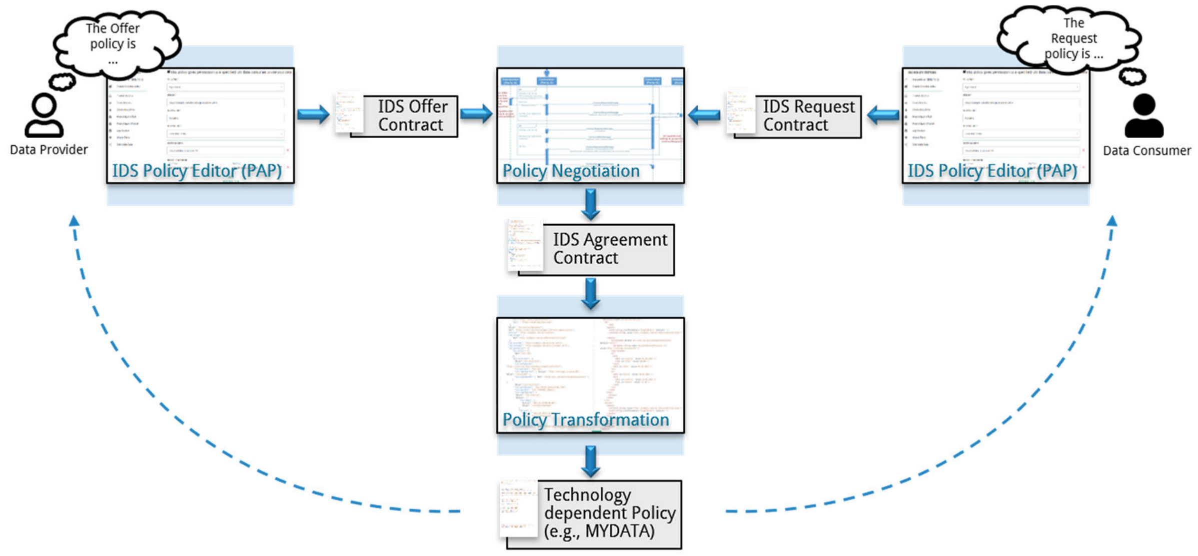


Figure 13: Policy Specification

5.1 Information Model and Policy Language

Usage restrictions can be exchanged in an infinite number of forms and models. From a legal point of view, verbally communicated agreements are perfectly fine and can be regarded as valid contracts. However, as proving the details of such an agreement is a challenge, usually textual contracts are of course preferred. Until now, this procedure has been sufficient and created the foundation for any business process. In a more and more digitalized world however, humans are not the only players anymore. As machines -- connectors in terms of the IDS -- take part in the processes, higher degrees of formalizations are necessary. As Figure 14 depicts, several stages are possible, and all can provide value in different use cases. For instance, machine-readable policies reduce the degree of fuzziness of natural language texts and can be parsed and exchanged between actors, as e.g., XML or JSON.

Enforceable policies further specify distinct and deterministic criteria for each clause and can be transformed to executable code. As such, enforceable policies must connect the description of usage restrictions with infrastructure components and existing endpoints and unambiguous instructions.

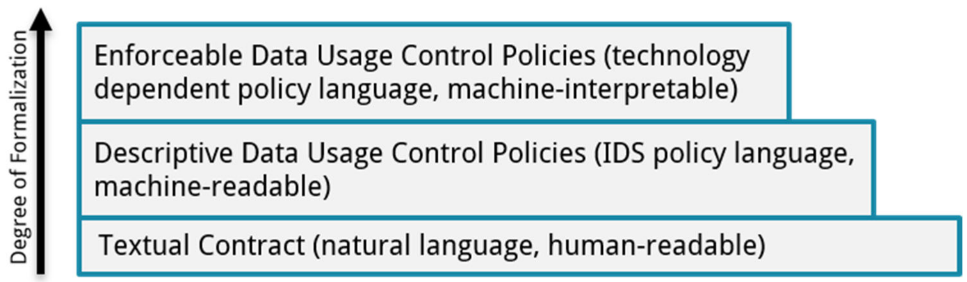

Degree of Formalization
Enforceable Data Usage Control Policies (technology dependent policy language, machine-interpretable)
Descriptive Data Usage Control Policies (IDS policy language, machine-readable)
Textual Contract (natural language, human-readable)

Figure 14: Level of formalization of policies

The selection of the appropriate format depends on the use case. Scenarios with high uncertainty, variety and valuable assets involved require more manual influence and should be located at the lower layers. Less heterogeneous use cases with high interaction rates can pay off the higher formalization effort through an increased automation. Furthermore, several usage control scenarios require machine-readable policies to independently evaluate whether a usage request is appropriate or not. For instance, in the SCRM example, the automated deletion after 14 days must not depend on a human action but needs to be executed automatically.

Exchanging any usage control policy starting from the descriptive layer requires an explicit understanding of both, the form and the content of the policies. The syntax of the IDS Usage Control language is therefore relying on the Resource Description Framework RDF (see [14], [15] for details). In the same way that RDF has several serialization formats, e.g., XML or JSON, any IDS Usage Policy can be transformed into any RDF compliant serialization and vice versa without loss of information. Regarding IDS messages, the determined serialization is JSON-LD. Still, the comprehensive modelling of the intended content of usage policies is challenging. In order to achieve a common understanding, the IDS promotes its Information Model (see Figure 15) as the core vocabulary and structure to encode semantic meaning. As such, the IDS Information Model specifies the shared terms, the supported schema and the relations between allowed patterns in the Commodity concern. More precise, this concern presents a profile of the Open Digital Rights Language version 2.2 (see [16], [10] for details). ODRL is a W3C recommendation and specifies a vocabulary and data model for the description of digital and machine-readable contracts. The IDS further extends ODRL towards usage control descriptions and enforcement, provides explanations regarding the compliant interpretation of constructs and defines implications for real-world implementations. This is accomplished in the form of IDS subclass constructs to the according ODRL classes. The design preserves the structure of the introduced terminology, resulting in the compliance of every IDS Usage Policy with ODRL recommendations. Nevertheless, not all ODRL terms are part of the IDS Usage Control language as the requirements slightly differ. Even though this might change in future versions, it should not be expected implicitly.

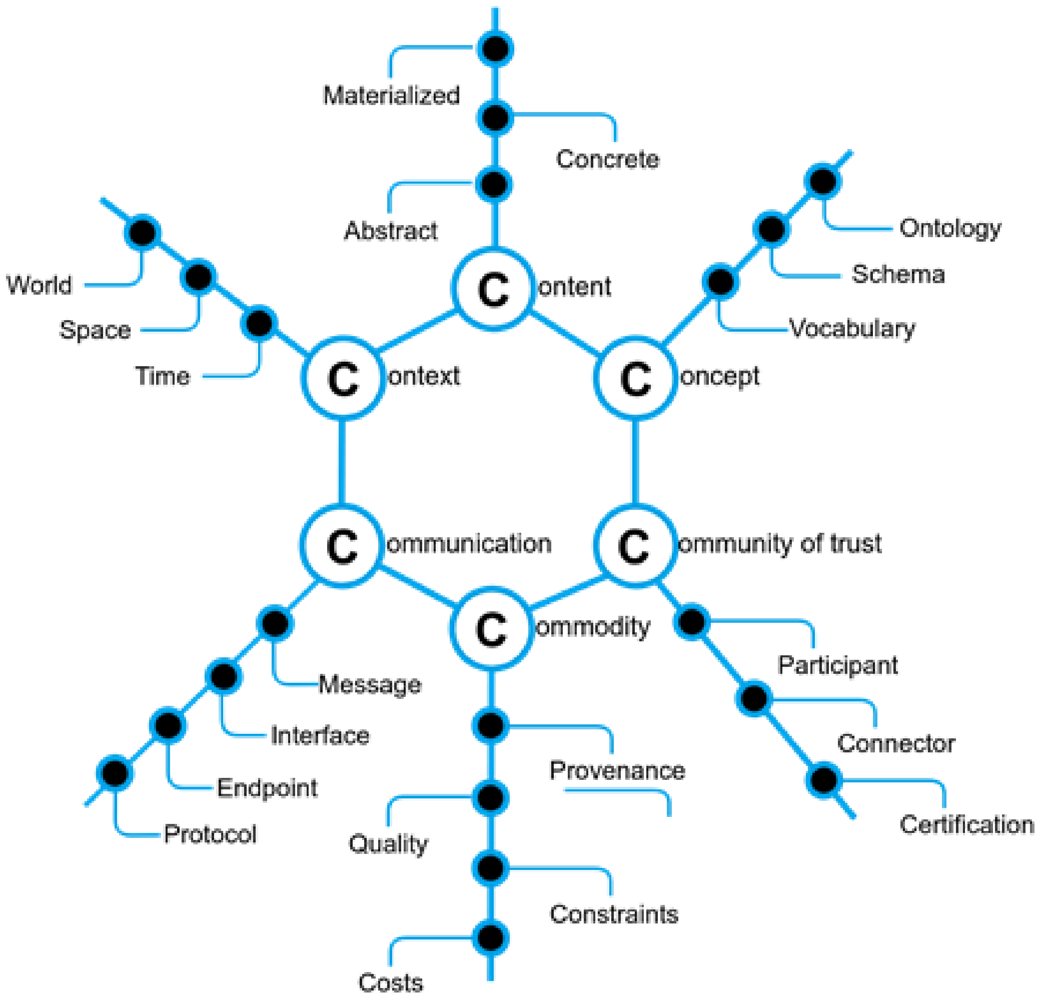


Figure 15: Concerns of the IDS Information Model. The concepts of the IDS Usage Control Language are defined as part of the IDS ontology. It expresses the constraints in the commodity category.

The IDS Usage Control Language does not support the full expressiveness of all possible ODRL compliant policies. The reason for this design decision is based on the nearly infinite number of possible combinations of ODRL instances. As the IDS needs to automatically evaluate restrictions in many cases, more fine-grained definitions are necessary. One example is the limitation to only two actors, one provider of any possible target resource and one consumer, which usually reimburses the provider for its effort. In order to formally encode the difference, the IDS Information Model mirrors the supported ODRL classes by own subclasses with further annotations and requirements. No IDS compliant connector can be required to understand plain ODRL constructs or terms but only their IDS counterparts. This is important to ensure a proper and unambiguous understanding between different IDS participants.

The IDS Contracts come in three different realizations: *Contract Requests*, *Contract Offers*, and *Contract Agreements* (see Figure 16). While all share a similar syntax, their interpretation is slightly different. Requests indicate a desire to achieve a certain contract. Therefore, they must be regarded as a suggestion or a query, usually coming from a potential Data User (IDS Data Consumer). Requests have no binding character and are used before any actual usage control system is considered. Similar to Contract Requests, Contract Offers have only an informative meaning. Contract Offers present a potential willingness to interact under the specified conditions. Usually, a Data Provider or Service Provider publishes Offers to signal its willingness to exchange data or services as outlined in Contract Offers. However, neither Requests nor Offers obligate the participants to any later commitments.

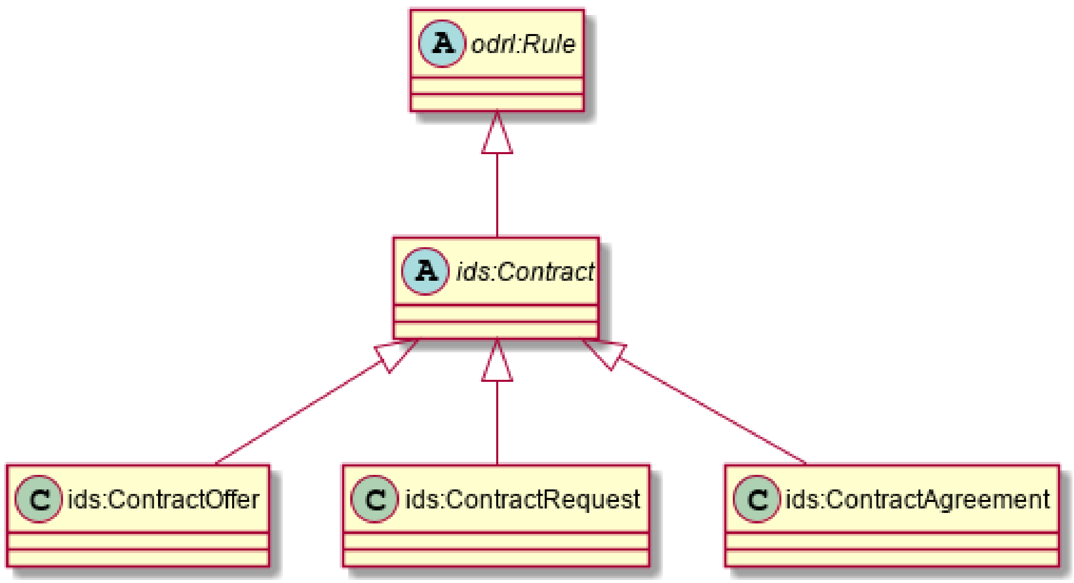


Figure 16: The IDS Information Model defines offers, requests and agreements as subclasses of the abstract contracts.

In contrast, the exchange of valid Contract Agreements represents a binding and final consent to the stated constraints and requirements. An Agreement is the IDS terminology for a valid contract, which both sides accepted and therefore is binding as far as the IDS is related. As a result, only Contract Agreements must be regarded by any IDS usage control system. In Table 2, we listed all contract types together with their implications and descriptions.

Table 2: Contract types with their implications and descriptions

Contract Type	Implication	Description and Interpretation
Contract Offer	The usage of a certain data resource might be possible regarding the stated constraints.	An offer is purely informative and voluntary metadata by the data provider. They give a rough idea on the usage restrictions and shall improve the discovery and selection process for Data Users. The Data Sovereign benefits by reaching a better visibility of its preferences.
Contract Request	The usage of a certain data resource is desired under the stated constraints.	The Data User indicates its interest, and may create the request relative to a previously exchanged Data Offer. The Data Sovereign gets to know acceptable constraints of the Data User while the later can further detail a contract.
Contract Agreement	Both participants conclude a contract and agree to the stated constraints. A later adjustment is not possible without a mutual consent.	The constraints have been fixed and accepted by both participants. The usage control systems import the agreement and enable or prevent access and usage accordingly.

5.2 Policy Classes

A Data Usage Control policy, in general, may provide permission to an IDS Data Consumer to operate specified action(s) over a Data Asset or prohibit the operation of that specified action(s). Providing permission or prohibition of an operation is extended to variety of actions. A policy can be specified to provide permission to use the data. The action of using the data covers various operations over that piece of data such as displaying it, printing it, making calculation over it, and so on. In addition, a policy may address only a particular fine-grained action. For example, a policy that permits reading data, allows the act of obtaining the Data Asset from the data source without further restrictions, however, the action of printing data is not permitted.

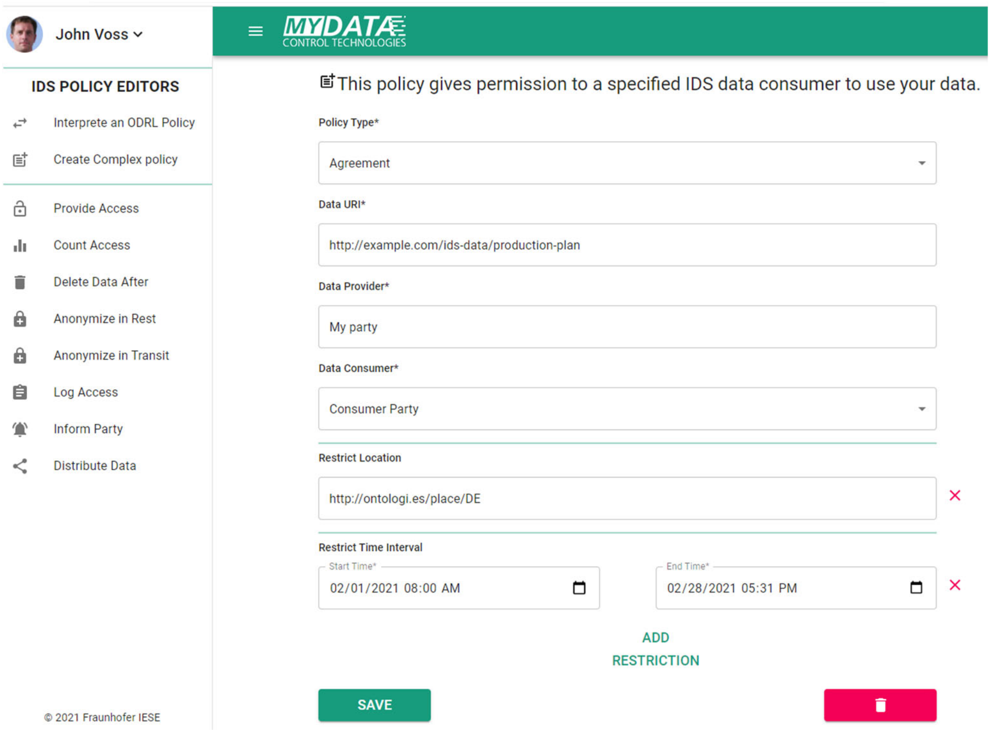
The Data Usage Control technologies in IDS context support the whitelisting approach to protect the data. It means that the access to the non-public data is prohibited by default. The studies on the requirements and use cases of the IDS projects shows that several restrictions shall apply when data is used. For example, an IDS Data Consumer may request to use the data in a specific time interval, or an IDS Data Consumer may restrict the usage of the data to a specific location. We have categorized these restrictions into 21 atomic templates called policy classes that are explained in Appendix A.1 in details.

Eventually, a Data Usage Control policy is a combination of one or more instances of these policy classes that is identified and is referring to a specific piece of data. Furthermore, the policy classes may evolve over the time in the context of IDS, depending on the stakeholders' demands as well as public rules and regulations.

5.3 IDS Policy Editor

A policy editor or in XACML terminology a Policy Administration Point (PAP) supports data owners and data providers in specifying their usage restrictions. Policy editors usually comprise a Graphical User Interface and offer different levels of guidance to the user, depending on knowledge and skill level. The IDS Lab offers an IDS Policy editor to express the aforementioned policy classes within the IDS. Interested reader can access the web user interface via the following link: https://odrl-pap.ids.isst.fraunhofer.de/

The policy editor (see Figure 17) is regularly updated. At the moment, it supports the users in creating their IDS Contracts for the above-mentioned policy classes that can be used within the information model.


This policy gives permission to a specified IDS data consumer to use your data.

Policy Type*: Agreement

Data URI*: http://example.com/ids-data/production-plan

Data Provider*: My party

Data Consumer*: Consumer Party

Restrict Location: http://ontologi.es/place/DE

Restrict Time Interval

Start Time*: 02/01/2021 08:00 AM

End Time*: 02/28/2021 05:31 PM

Figure 17: Screenshot of the policy editor

5.4 Policy Transformation

In general, we can differentiate the degree of formalization in the process of specifying Data Usage Control policies. For example, a contract can be specified using natural language (e.g., “use my data only for billing and delete after fourteen days”). In order to enforce these contracts to the systems, the Data Usage Control technologies must be able to transform the IDS Contracts from the specification level policies (SLP) to their Implementation Level Policies (ILP). Therefore, ILPs have a direct mapping to technical enforcement. The IDS context promotes several different languages for the ILPs (e.g., MYDATA or LUCON Policy Language). These technology-dependent policy languages are further explained in this document.

In order to support the Data Usage Control technologies, the policy transformation service is additionally added to the IDS policy editor. Currently, it supports the transformation to enforceable policies for the MYDATA Control Technology. The support for other technologies and further extension will follow.

5.5 Policy Handshake and Negotiation

A negotiation process takes care of the potential bargaining of the usage conditions. When the usage restrictions are specified, the requirements and preferences of the data user are usually unknown. Following a simple accept or reject pattern drastically reduces the number of potential users and thereby reduces business opportunities. In addition, fixing obligations without knowing the context and implementation details of the potential usage is not reasonable as the information gap between specification and implementation time leads to unforeseen mismatches and conflicts. Therefore, an interested data user should be enabled to respond to a usage offering with a slightly adjusted counteroffer. Still, it must be always in the authority of the data sovereign to accept or reject the request, or even make an additional offer regarding the details of the received counteroffer.

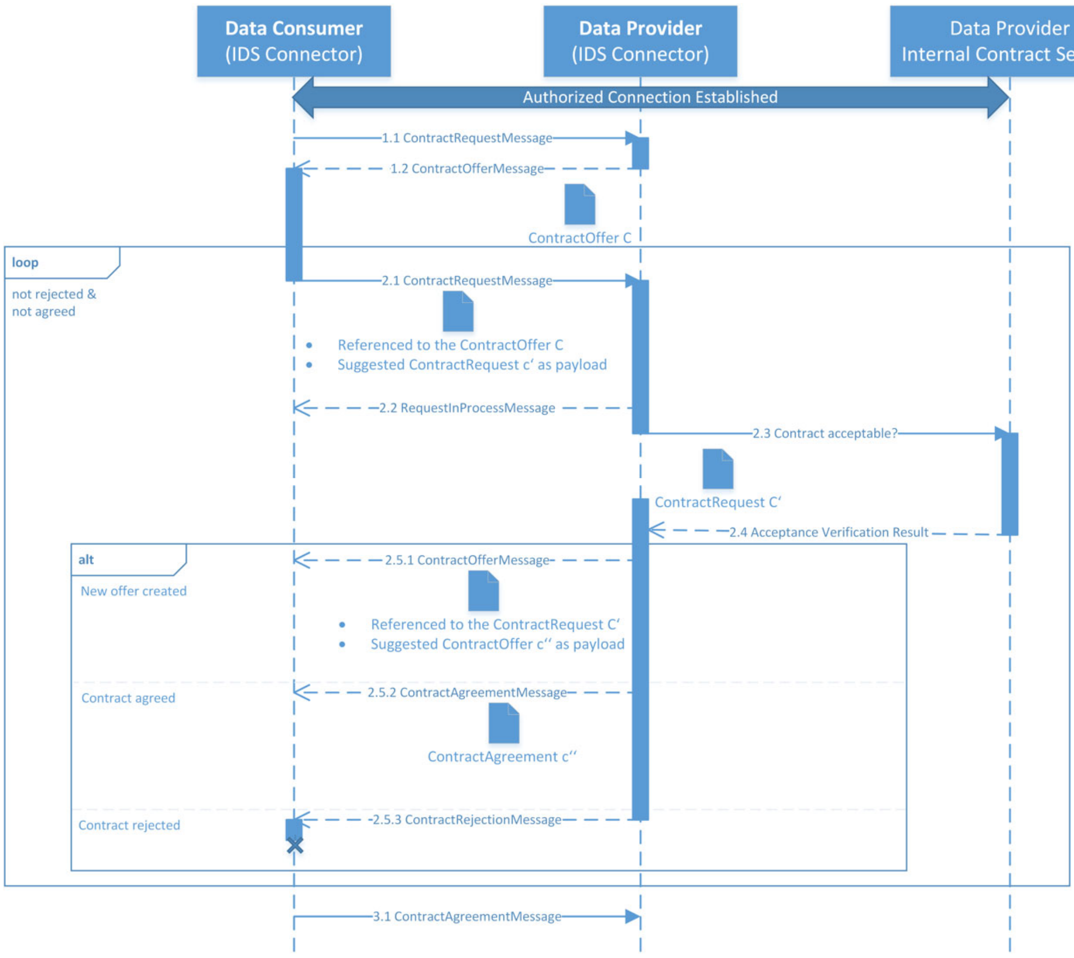

Data Consumer (IDS Connector)
Data Provider (IDS Connector)
Data Provider Internal Contract Service
Authorized Connection Established
1.1 ContractRequestMessage
1.2 ContractOfferMessage
ContractOffer C
loop
not rejected & not agreed
2.1 ContractRequestMessage
Referenced to the ContractOffer C
Suggested ContractRequest c' as payload
2.2 RequestInProcessMessage
2.3 Contract acceptable?
ContractRequest C'
2.4 Acceptance Verification Result
alt
New offer created
2.5.1 ContractOfferMessage
Referenced to the ContractRequest C'
Suggested ContractOffer c'' as payload
Contract agreed
2.5.2 ContractAgreementMessage
ContractAgreement c''
Contract rejected
2.5.3 ContractRejectionMessage
3.1 ContractAgreementMessage

Figure 18: Contract negotiation between a data consumer and a data provider

A negotiation procedure is depicted in Figure 18. The Data Consumer, in the SCRM example the part supplier, would like to use a dataset that is accessible in the IDS. The OEM, here in the role of a Data Provider, restricts the usage to a certain amount of time. In this scenario, the Data Consumer announces its interest by creating a Contract Request as part of an IDS ContractRequestMessage and sends it to the providers IDS connector (1.1). The provider answers the request by sending a first ContractOfferMessage (1.2). In a second iteration, the Data Consumer reflects his gained knowledge of the previous Contract Offer and transforms it into a new proposal, which better fits its preferences (2.1). For instance, one can easily imagine that the OEM wants to minimize the usage time in order to protect its data and therefore asks to only use the dataset for one week. The consumer, on the other hand, would like to gain more flexibility, and asks for a usage permission for at least one year.

Consequently, the Data Provider must decide whether it:

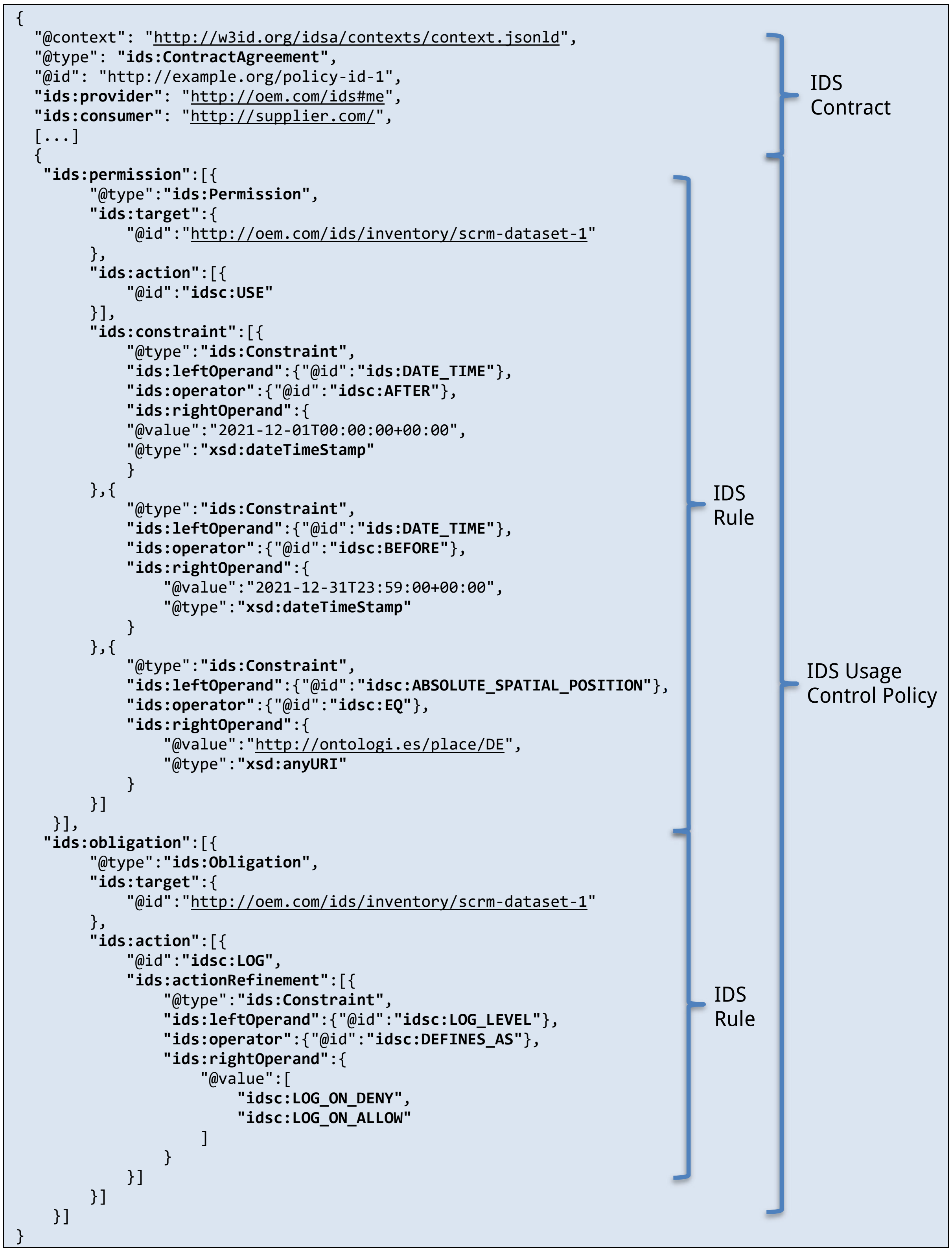
- responds with an adjusted Contract Offer (2.5.1),
- accepts it and reacts with a Contract Agreement (2.5.2), or
- rejects the request and terminates the interaction (2.5.3).

As this decision usually requires extensive knowledge about the context of the scenario and deep insights into the business logic, it is very challenging to automate. In most scenarios, the provider connector therefore acknowledges the second Contract Request Message by stating that some time is needed and forwards the request to a human operator. That operator, however, needs to interpret the IDS contract and its legal implications before confirming an agreement. Nevertheless, in case a proper business logic is in place, this task may also be completed by an autonomous agent or software module. After a decision has been made, the respective response message is sent (2.5). The IDS ensures the transparent linkage between the single messages through certain correlation message attributes in the message header but also references in the following contracts. Each message, contract, or any other resource and information entity has a unique URI and is therefore identifiable throughout the whole negotiation process. This fact is also illustrated by the time-restricted Contract Agreement shown below (Listing 1). It reflects the constraint to only use the targeted dataset for only one month (from 01.12.2021 to 31.12.2021). Last, the consumer must acknowledge the received contract agreement (3.1).

At this stage of the interaction process, it is important to note that even though the provider already signaled its general willing to allow a usage through its offers, it has still every right to withdraw its offers at any point. The IDS regards it as a successful interaction only after both parties have exchanged Contract Agreement Messages with exactly the same contract payload. The Contract Agreement C'' with the unique URI 'http://example.org/policy-id-1', represents the result of the described negotiation. In general, the iterations in such negotiations are not limited. Still, both connectors can terminate the interaction at any step and decide to reject the potential data exchange.

After a successful negotiation process, the Data Provider prepares the data transfer and sends a download link to the Data Consumer. The Data Consumer is now able to download the data (pull). As an alternative, the Data Provider is also able to send the data directly to the consumer (push). In case of multiple data requests from the Data Consumer, it is not necessary to repeat the negotiation process as every request via the prepared Data Provider endpoint is linked to the same contract.

Another way towards an automated IDS Contract Negotiation is shown in the Future Work Subsection 8.2.2.



```
{
  "@context": "http://w3id.org/idsa/contexts/context.jsonld",
  "@type": "ids:ContractAgreement",
  "@id": "http://example.org/policy-id-1",
  "ids:provider": "http://oem.com/ids#me",
  "ids:consumer": "http://supplier.com/",
  [...]
  {
   "ids:permission":[{
        "@type":"ids:Permission",
        "ids:target":{
            "@id":"http://oem.com/ids/inventory/scrm-dataset-1"
        },
        "ids:action":[{
            "@id":"idsc:USE"
        }],
        "ids:constraint":[{
            "@type":"ids:Constraint",
            "ids:leftOperand":{"@id":"ids:DATE_TIME"},
            "ids:operator":{"@id":"idsc:AFTER"},
            "ids:rightOperand":{
            "@value":"2021-12-01T00:00:00+00:00",
            "@type":"xsd:dateTimeStamp"
            }
        },{
            "@type":"ids:Constraint",
            "ids:leftOperand":{"@id":"ids:DATE_TIME"},
            "ids:operator":{"@id":"idsc:BEFORE"},
            "ids:rightOperand":{
                "@value":"2021-12-31T23:59:00+00:00",
                "@type":"xsd:dateTimeStamp"
            }
        },{
            "@type":"ids:Constraint",
            "ids:leftOperand":{"@id":"idsc:ABSOLUTE_SPATIAL_POSITION"},
            "ids:operator":{"@id":"idsc:EQ"},
            "ids:rightOperand":{
                "@value":"http://ontologi.es/place/DE",
                "@type":"xsd:anyURI"
            }
        }]
    }],
   "ids:obligation":[{
        "@type":"ids:Obligation",
        "ids:target":{
            "@id":"http://oem.com/ids/inventory/scrm-dataset-1"
        },
        "ids:action":[{
            "@id":"idsc:LOG",
            "ids:actionRefinement":[{
                "@type":"ids:Constraint",
                "ids:leftOperand":{"@id":"idsc:LOG_LEVEL"},
                "ids:operator":{"@id":"idsc:DEFINES_AS"},
                "ids:rightOperand":{
                    "@value":[
                        "idsc:LOG_ON_DENY",
                        "idsc:LOG_ON_ALLOW"
                    ]
                }
            }]
        }]
    }]
}
```

Listing 1: Example of a time-restricted IDS Contract Agreement

6. Usage Control Technologies

In the first part of this section each Usage Control technology available in the IDS is presented in detail. Every technology section covers the topics of communication flows, integration concepts including their characteristics and how the necessary Usage Control permissions and obligations are extracted out of the IDS Contracts. At the end of this section, all IDS Usage Control technologies are compared with respect to their general characteristics, their capabilities and policy languages as well as their location of implementation. A discussion closes the chapter.

6.1 MYDATA Control Technologies

MYDATA Control Technologies (MYDATA for short) is a technical implementation of data sovereignty, which represents an essential component for informational self-determination. It is based on the IND^{2}UCE framework for data usage control developed at Fraunhofer IESE.

In general, MYDATA implements data sovereignty by monitoring or intercepting security-relevant data flows. This enables fine-grained masking and filtering of data flows in order to make them anonymous, for example. Compared to classical access control systems, MYDATA can enforce partial filtering and masking of data, context and situation restrictions as well as restrictions on the purpose of use.

6.1.1 Usage Control App

MYDATA provides developers with a Usage Control App (UC App) to enable an easier integration within the connector. It is able to translate IDS Usage Control Policies into MYDATA policies. The MYDATA Library inside offers a Policy Management Point (PMP) to manage and deploy the MYDATA Policies at the Policy Decision Point (PDP). In addition, it also has an integrated Endpoint for the Policy Enforcement Point (PEP), which works together with the integrated PDP to enforce active policies. Finally, the UC App also supports live context evaluation via MYDATA Policy Information Points (PIPs) and the execution of commands via MYDATA Policy Execution Points (PXPs).

In summary, the UC App packages the main components of MYDATA in one app and thus enables an easy integration in IDS connectors.

6.1.2 Communication Flow and Integration Concepts

The overall communication flow from a Data Source at the Data Provider to a Data Sink at the Data Consumer is illustrated in an overall flow in Figure 19. There are several data sources at the Data Provider side that can be connected to the IDS Connector (e.g., database, file system, application). The dataflow starts at the Data Source and is handled by the core container in the IDS Connector of the Data Provider. As part of the routing and before data is flowing to the Data Consumer, the Usage Control App is invoked and Data Provider specific Usage Control rules are applied to the data (e.g., remove person-related information). Data is then sent to the

IDS Connector of the Data Consumer. At the Data Consumer side, the same procedure is happening within the connector, hence, the Usage Control App is invoked as part of the data routing and applies the Data Consumer specific Usage Control rules (e.g., data is only allowed to be used by a specific target system). Data Sources and Data Sinks can be located inside the IDS Connector (e.g., data apps) or outside the IDS Connector (as depicted in Figure 19). If data resists within the IDS Connector, Usage Control can be easily achieved.

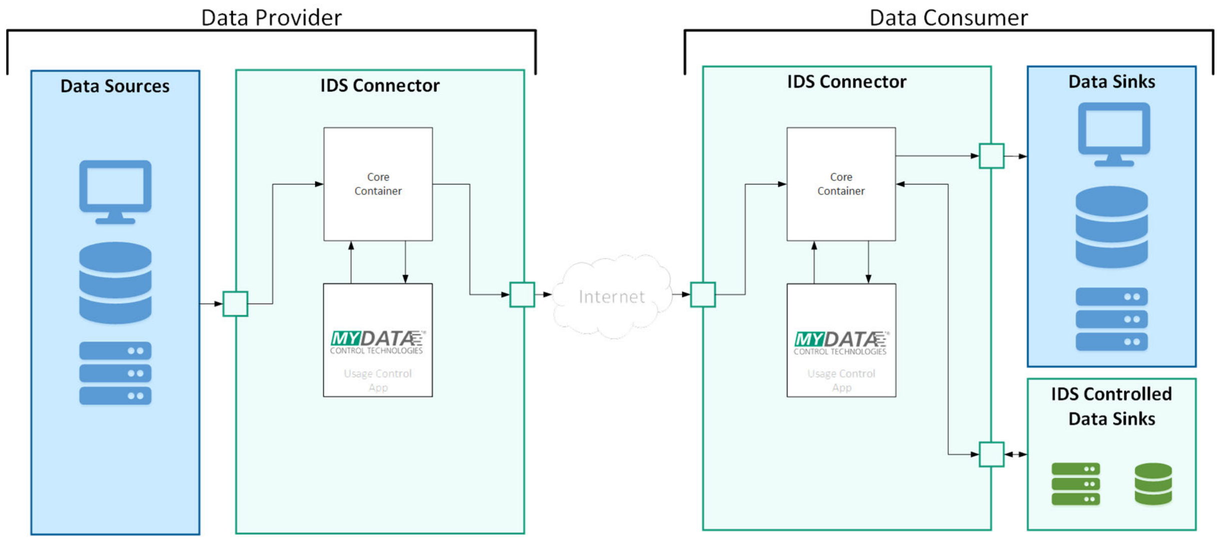


Figure 19: MYDATA Communication Flow and Integration Concepts

Hence, Usage Control can be separated into Usage Control at the Data Provider Side and Usage Control at the Data Consumer Side. Basically, MYDATA can be integrated as part of the message routing or as interceptor pattern. In the following, we will differentiate these two approaches and explain them in more detail.

6.1.2.1 Usage Control at the Data Provider Side

The Usage Control at Data Provider Side is applied whenever data is processed by the IDS Connector. The core container is responsible to handle the data processing. We use Apache Camel as message-oriented middleware that handles the message router in the following. To integrate Data Usage Control into the data processing of the IDS Connector, we enforce that all messages are routed to the Usage Control App. In doing so, we are able to control any data that is processed by the IDS Connector. To ensure full data control, the Usage Control App has to be invoked last before data is leaving the IDS Connector at Data Provider Side. However, the Usage Control App may additionally be invoked between other processing steps of the IDS Connector. For example, before or after data is processed by Apps that are running within the IDS Connector. For the sake of simplicity, we will not address this kind of Usage Control enforcement at this point.

6.1.2.2 Usage Control at the Data Consumer Side

If an IDS Connector is consuming data, the simplest case is the forwarding of the received data to a Data Sink at the infrastructure of the Data Consumer. Similar to the Data Sources, the Data Sinks can be any kind of system (e.g., application, database). Contrary to the Usage Control at the Data Provider Side, we have chosen to implement the Usage Control at the Data Consumer Side as interceptor pattern. In this way, we can hook into every message routing step and apply the Usage Control rules to the data flow.

In summary, Usage Control at the Data Provider Side as handled by message routing as part of the route configuration, Usage Control at the Data Consumer Side is handled by message interception that is part of the IDS Connector implementation. The two integration concepts are depicted in Figure 20.

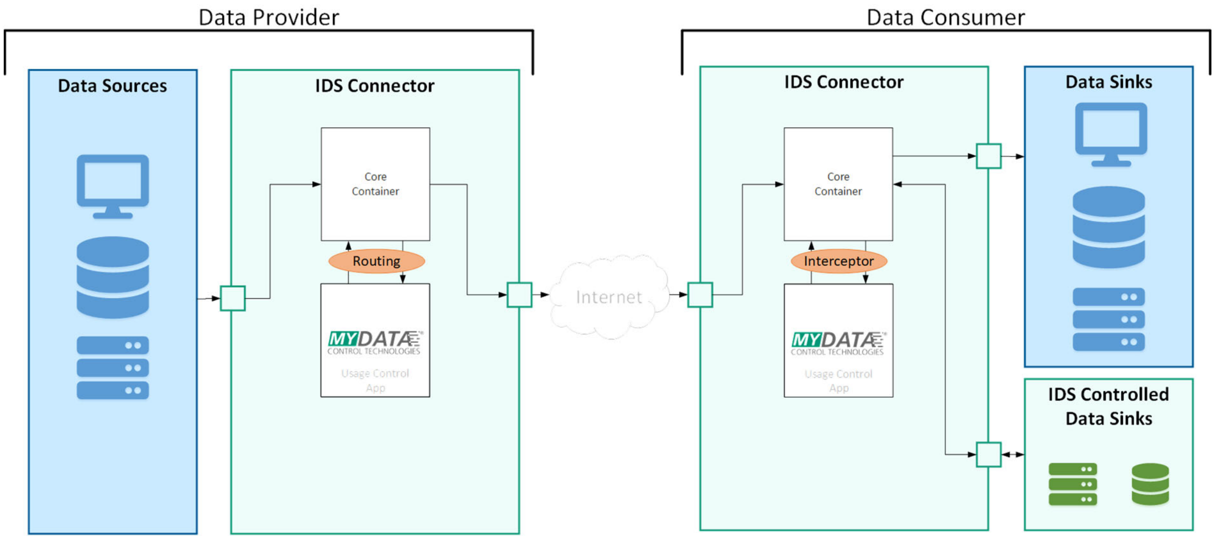


Figure 20: Routing and Interceptor Approach

In Section 4.3.3, we presented two ways of storing data at storage infrastructure at the Data Consumer. However, there is also the possibility to store the data within the IDS Connector. In this case, the communication to and from the internal storage is handled by the Core Container, which uses the interceptor pattern to control the data. Hence, the Usage Control enforcement is handled analogously.

6.1.3 Detailed Integration Concepts

We presented the two integration concepts in the previous sections. We will have a closer look to them in the following. However, before we start, we will explain some more details.

First, we will explain the Usage Control App. The Usage Control App contains at least a PEP, a PDP and a PMP. It offers interfaces for calling the PEP interfaces and the PMP interfaces. Invoking the Usage Control App, as used in the previous sections, results in invoking the PEP within the app (follow the processing explained in Figure 21). In addition, there is the possibility to deploy machine-readable policies by calling the PMP interfaces.

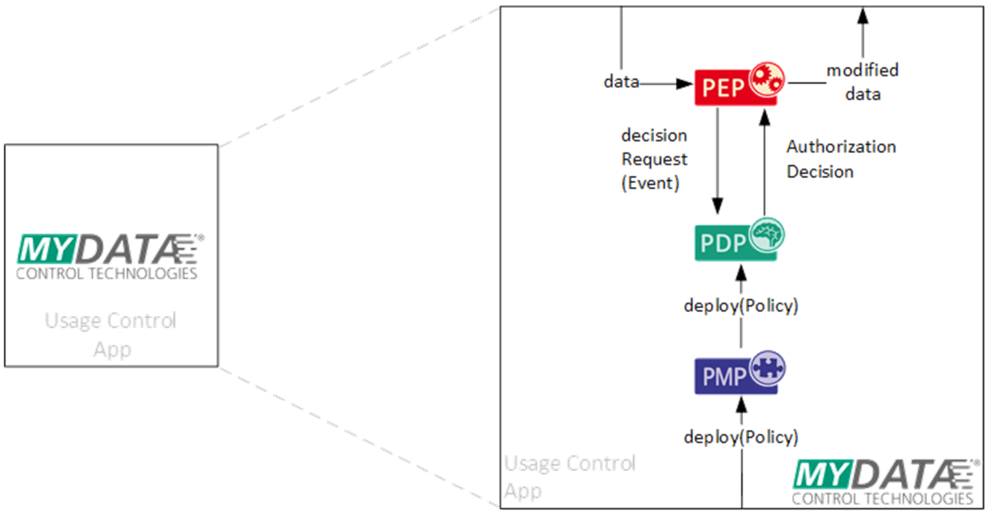


Figure 21: MYDATA PEP invocation and processing by the PDP

Second, we explain the message routing of a message-oriented middleware such as Apache Camel. The middleware coordinates the data flow between different systems and applications. From a technical point of view, Apache Camel provides this by a routing data between different nodes. In a nutshell, it passes the output from one node and puts it as input to the next node. The routing may comprise several nodes as depicted in Figure 22. Data Source and Data Sink are named Endpoints in Apache Camel terminology, every node in such a route is named a processor.

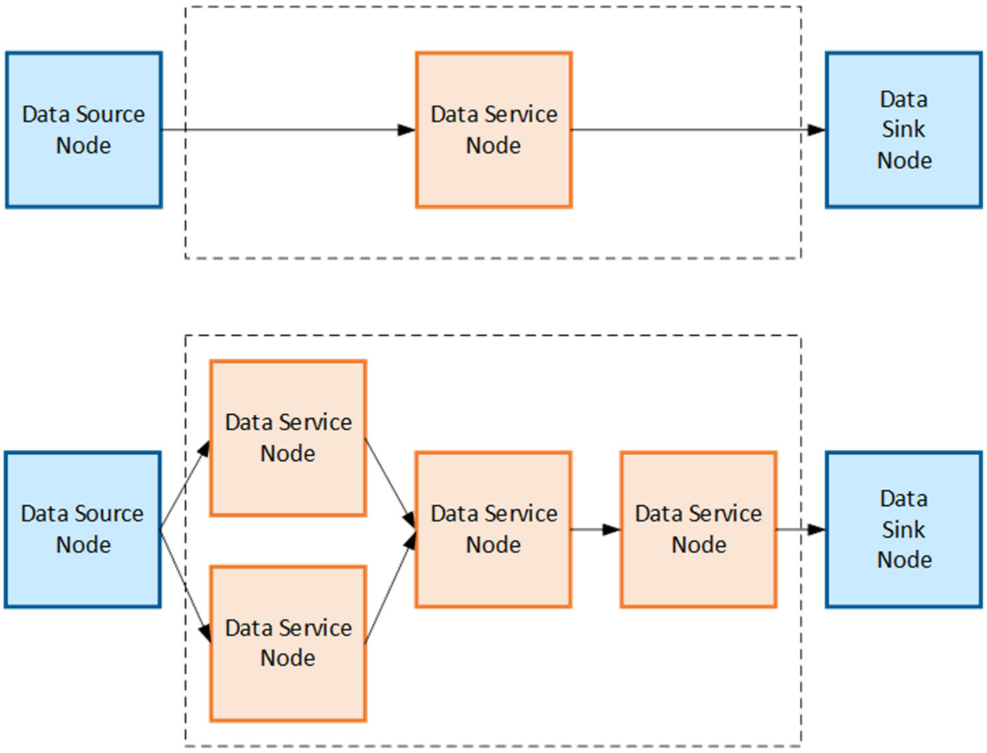


Figure 22: Routing example without interceptor. Above we show route 1 and in the bottom of the picture we show route 2.

Third, the interceptor pattern is used to invoke or handle some processing of the output or the input data before and after each node. The interceptor is implicitly called contrary to the message routing, which has to be explicitly configured.

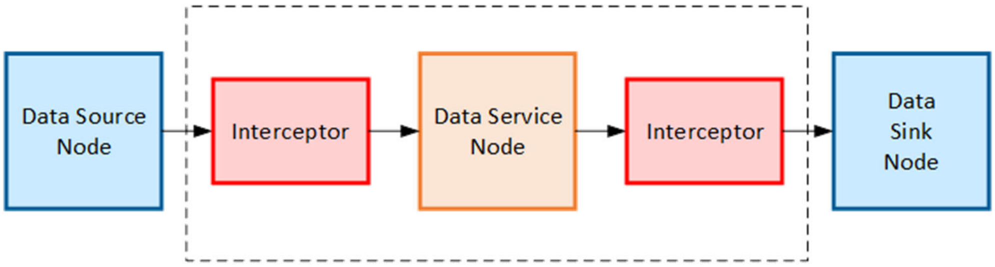
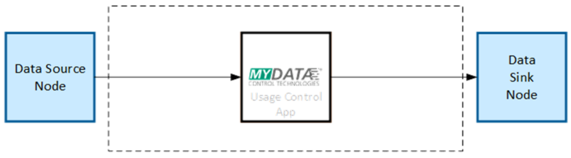
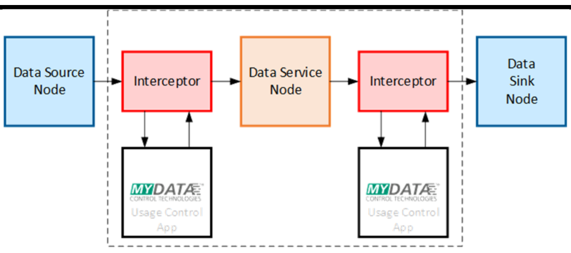


Figure 23: Routing example of route 1 (from Figure 22) with interceptor

We use "Route 1" for explaining the different approaches to integrate Usage Control with MYDATA (see illustrations in Figure 22 and Figure 23). See also

Table 3 for details.

Table 3: Comparison of Usage Control by Routing and Usage Control by Interceptor

Usage Control by Routing	Usage Control by Interceptor
 Data Source Node MYDATA CONTROL TECHNOLOGIES Usage Control App Data Sink Node	 Data Source Node Interceptor Data Service Node Interceptor Data Sink Node MYDATA CONTROL TECHNOLOGIES Usage Control App MYDATA CONTROL TECHNOLOGIES Usage Control App
If we integrate the Usage Control by Routing concept to control the usage of data, the Usage Control App has to be explicitly configured as a processor in the route. When using this approach, it is important to consider where the processor is located within the route. For example, in case of counting it should be the last processor in the route before data is leaving the organization. Contrary, if you want to prevent the processing of person-related data, it should be the first processor in the route. In case an Apache Camel Processor is used, the Processor delegates its call to the Usage Control App.	If we integrate the Usage Control by Interceptor concept to control the usage of data, the Usage Control App is implicitly called between every processing step. Apache Camel offers the possibility to integrate interceptors that it executes every time before and after a processor is working. In this approach, MYDATA is completely integrated into the Camel Interceptor and forwards every interception as a decision request to the PDP. In case an Apache Camel Interceptor is used, the Interceptor internally delegates its call to the Usage Control App.

6.1.4 Comparison and Discussion of the Integration Concepts

The decision which approach to use depends on the requirements to be fulfilled by Usage Control. For example, integrating the Apache Camel Interceptor approach leads to a higher performance impact than implementing as Apache Camel Processor (due to the (probably) fewer number of interceptions). On the other side, implementing the Usage Control App only at one place in the route lowers the security as the messages are only intercepted at that one point in the route. This kind of approach may be suitable for the Data Provider Side, but probably not for the Data Consumer Side, because data entering the route can be processed by apps without being checked by the Usage Control App. At the Data Consumer Side, a Data Provider probably wants the Usage Control App to check the data usage before and after every app processor (e.g., data apps). When implementing the Usage Control by Routing, the developer has to select suitable location(s) so that the usage Control App is able to enforce all required Usage Control policies. Table 4 summarizes the comparison in a short form.

Table 4: Comparison of integration concepts

	Independent Development	**Additional Component**	**Performance Impact**	**Suitable for Consumer/ Provider**
Usage Control by Interceptor	partly (Usage Control App independently; integration with Camel needed for the interceptor)	yes	high	Consumer/ Provider
Usage Control by Routing	yes	yes	low	If Provider (preferred), Consumer (partly, in exceptions)

6.1.5 Transformation of IDS Usage Policies

The focus of this chapter is on MYDATA policy language and the transformation of an IDS Usage Policy to a MYDATA Policy. An IDS Contract is specified with a unique identifier and expresses the Usage Control Policy and various further contract related information. An IDS Usage Policy consists of at least one Rule. The IDS Target, IDS Action, IDS Assigner and IDS Assignee are the main elements of a Rule. The IDS Target addresses the data asset to be protected. The IDS Action is the type of access to the data asset. Example of actions are read, display, delete, use, etc. The IDS Assigner and the IDS assignee are the parties that issue the policy rule and receive it, respectively. An IDS Policy Rule is the permission, prohibition or obligation of operating one or more IDS Actions over a specified IDS Target. Moreover, The Constraints are Boolean or logical expressions that refine the semantics of the actions, parties and assets, however, it might declare the conditions applicable to the rules.

Likewise, a MYDATA Policy consists of one or more mechanisms and it is specified with a policy identifier. The MYDATA Mechanisms follow the if-then-else schema. The MYDATA Event, MYDATA Solution, MYDATA Decision, PIP, PXP and MYDATA Modify are the main elements of a MYDATA Mechanism. A MYDATA Event is an attribute of a MYDATA Mechanism. An Event is a hooking point of a system in which data is used. Additionally, it is a point of a system in which data must be anonymized, deleted and so on (i.e., protected). It contains a name, the time that it has occurred and a list of key-value parameters.

A MYDATA Solution is a site in which the MYDATA technology is used to protect their data. A site can be a company, an organization, an end user device or an IDS Connector. The MYDATA Solution identifier represents the site's name. A MYDATA Decision is an authorization decision that is specified in the then-block of a MYDATA Mechanism. It can be specified as "allow" or as "inhibit" in order to permit or prohibit the occurrence of the corresponding event, respectively. Moreover, a MYDATA Decision can be an execution of a PXP or an application of a MYDATA Modify. A PXP executes a requested action on the occurrence of an event. A MYDATA Modify modifies the flowing data according to the user's preferences. A MYDATA Modifier is mostly used to filter or anonymize the data. The data is addressed as an Event parameter.

A PIP returns a requested value from an external information resource. It is mostly used to indicate and evaluate a condition. A MYDATA Condition is specified in the if-block and leads to a specified MYDATA Decision, when it matches.

The MYDATA Policy and MYDATA Mechanism are the equivalent concepts for the IDS Usage Policy and IDS Rule, respectively. Table 5 gives an overview about comparable elements in IDS and MYDATA policy languages.

Table 5: IDS language equivalents in the MYDATA language

IDS	MYDATA	Description
id	Policy pid	A policy must have a unique identifier in ODRL and MYDATA policy languages.
Rule	Mechanism (Decision)	An IDS usage policy consists of one or more rules. Similarly, a MYDATA policy consists of one or more mechanisms. The IDS rules and MYDATA mechanisms reflect a decision for specific usage of data.
Rule Permission	Decision Allow	A permission allows access and/or usage of a data asset with further specifications in the

		form of constraints or connected duties. In order to allow the usage of data, a corresponding permission rule must exist.
Rule Prohibition	Decision Inhibit	A MYDATA inhibit decision represents an IDS prohibition rule.
Rule Obligation	Execute (PXP)	A MYDATA execute action (PXP) represents an IDS obligation or duty.
Assigner	Solution	An assigner is a party that issues a rule. In a provider-side policy, the assigner exposes the enforcement site.
Assignee	Solution	An assignee is a party that receives a rule. In a consumer-side policy, the assignee exposes the enforcement site.
Target	Event Parameter, PIP condition	An IDS target represents a data asset. In MYDATA, the data can be addressed as an event parameter or can be examined using a PIP.
Action	Event	A MYDATA event is a representation for an IDS action which defines an operation on a data asset and also, clarifies where the policy must be enforced.
Duty	Execute (PXP)	Duties frame any type of obligations connected to the usage of a data asset. When the operation of a duty is about modifying the data, a MYDATA modifier can be used instead of a PXP. It is because MYDATA has specific tag for modifying data in transit. The ODRL language allows the declaration of consequences for not fulfilling an ODRL duty. Currently, the relevant concepts to the ODRL consequences are not defined in the IDS context.
Left Operand Delay Period	Timer, PIP condition	The MYDATA timer can be used to represent a delay period. For example, when a delay period attribute is set to 5 days, we translate it to a MYDATA timer that sends a corresponding event daily. In addition, we need a PIP that on each day, evaluates whether 5 days has passed since the starting point.

Left Operand System	PIP condition	Most of the IDS constraint left operands such as system, purpose, etc., can be represented as a MYDATA PIP condition.
Left Operand Purpose	PIP condition	Most of the IDS constraint left operands such as system, purpose, etc., can be represented as a MYDATA PIP condition.
Left operand Date Time	Date and Time condition	In MYDATA, the date and time functions can be used to specify a condition that represents an IDS constraint with a date time left operand.
Left Operand Count	Count condition	A MYDATA count function can be used to check if a specified MYDATA event (IDS action) happened at least once (in this hour).

Furthermore, in order to transform an IDS Rule to a MYDATA Mechanism, one can consider the MYDATA policy as a template, and fill the relevant elements in it one by one starting from the authorization decision.

We assume that the IDS Assignee exposes the enforcement site when the policy has to enforce on the Data Consumer side and therefore it reveals the MYDATA Solution. It must be considered that the IDS Rules with different enforcement sites can be collected to one single IDS Policy. However, it is not possible to collect MYDATA Mechanisms with different solutions into one single MYDATA Policy and therefore, we need to split them up into more than one MYDATA Policy.

For more information, refer to the document [9].

6.2 Logic based Usage Control (LUCON)

LUCON (Logic based Usage CONtrol) is a policy language for controlling data flows between endpoints. The Trusted Connector uses Apache Camel to route messages between services (such as MQTT, REST, or OPC-UA endpoints). The ways how messages may be processed and passed around between services is controlled by LUCON, a simple policy language for message labeling and taint tracking. LUCON now fully supports the IDS policy language and can perform remote attestation after successful policy negotiation.
Security goals (c.f. Section 3.2) LUCON can help to achieve are:

- **Secrecy:** By defining a policy that disallows forwarding of labelled data to nodes that are explicitly marked as trusted in that context.
- **Integrity:** Integrity can be monitored by comparing in- and output of a node.
- **Time to live:** A state monitoring number of usages in a workflow can be defined to track allowed number of usages.
- **Anonymization by aggregation:** A policy can be defined that specifies how many samples need to be aggregated before an aggregation may be forwarded. This

functionality can be wrapped inside a micro service (e.g., aggregating five samples each) and this service then gains the according property.
- **Anonymization by replacement:** This is a task that needs to be wrapped inside a microservice. This service container could then even be certified to attest its functionality. The service then gains the property to satisfy the anonymization policy.
- **Separation of Duty:** A policy can be specified that disallows conflicting labels.
- **Usage scope:** A policy can be defined that disallows data with a specific label to leave the connector.

Two examples illustrate how this can be applied to real world problems:

Example 1: Automotive Supplier and OEM must restrict forwarding of data and limit data processing

An automotive supplier is providing valuable data about its current production capacities to an OEM. This information is extremely helpful for the OEM as it allows real-time and precise prediction of logistic chains, alignment of purchase strategies, and automated planning of production capacities at the OEM's side. However, for the supplier this information is highly critical and while it is willing to share it with the OEM in the context of a bilateral agreement, it needs to ensure that the data is only used for the agreed analytics and never shared with any competitor. Access control cannot solve this problem as it is only able to decide whether the OEM should be granted access to the data or not. With usage control on trusted endpoints, however, the supplier can state the following requirements and have them enforced at the OEM's side:

- Data must only flow into the known (and trusted) analytics applications.
- If data flows from the Trusted Connector at the OEM's side into any other (untrusted) endpoint, the supplier wants to be informed about this event so evidence of the violation is created, and legal actions can be taken.

Example 2: Enforcing Data Protection Regulations in Health Care Applications

A company is processing patient records for the sake of accounting and billing as a service to doctors and insurances. In addition, it is running data analytics as a service to the healthcare industry to assess drug sales in certain regions and support planning of drug productions and logistics. Strict legal regulations on personal identifiable information (PII) such as the German Bundesdatenschutzgesetz and the EU-GDPR require that PII must only be used for the purpose the user consented to. It is thus in the interest of the company to ensure (and potentially prove in an auditable way) that it complies to those regulations and is not leaking PII to the health care industry in the context of its analytics services. With usage control, the company can express the following requirements and have them automatically enforced within the Trusted Connector:

- Incoming raw data from patient records is marked as PII.
- PII must not leak to any endpoint except those leading to certified applications which are running containerized and are guaranteed to comply with GDPR rules.

- These applications are identified by a manifest containing a signed hash value.

This way, the data may only be processed in containers identified and certified.

6.2.1 Communication Flows and Integration Concepts

The policy set (contract) is translated into a rule set that specifies how data is handled while being kept in the consumer connector. LUCON is integrated into the "Trusted Connector". It could be used in other Camel based implementations as well. However, without accompanying security means, Usage Control with technical enforcement is hardly sensible.

The Trusted Connector is based on Apache Camel as a routing engine. It is possible to define routes between data sources and data sinks. A route can contain multiple steps and even deviations, forks and aggregations. In this example (see Figure 24) we illustrate a simple route. Data is pulled from a data source and forwarded to "Data Service Node 1". The output is forwarded to "Data Service Node 2". That output is finally sent to a data sink (e.g., another connector).

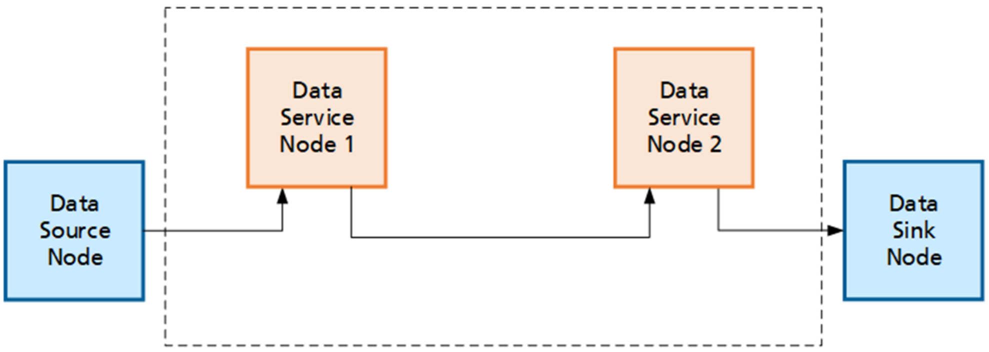


Figure 24: LUCON Camel route without interceptor

Apache Camel provides an Interceptor. This means, that a PEP is called whenever an Exchange object (the Camel wrapper for a message) leaves or enters a node. This is why the Interceptor is called twice between "Data Service Node 1" and Data Service Node 2" (see Figure 25).

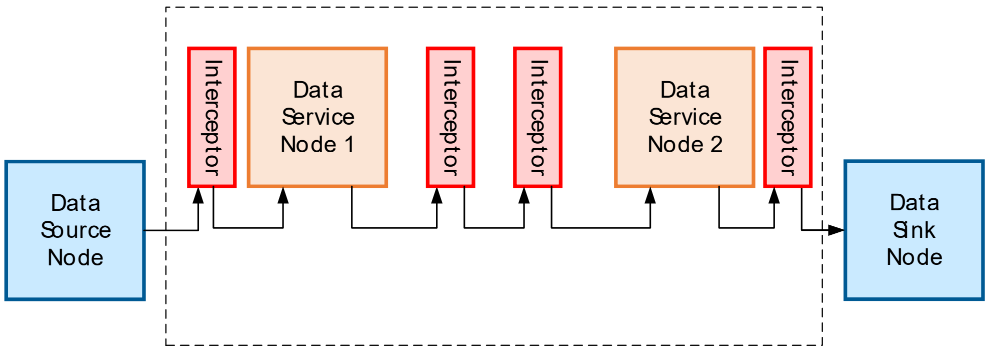


Figure 25: LUCON Camel route with interceptors

The integration is transparent for the administrator of data routes and defined policies can be seen as an allowed corridor for data routes. The overall design of communication flow is similar to that of MYDATA. The concept, however, is based on information flow control.

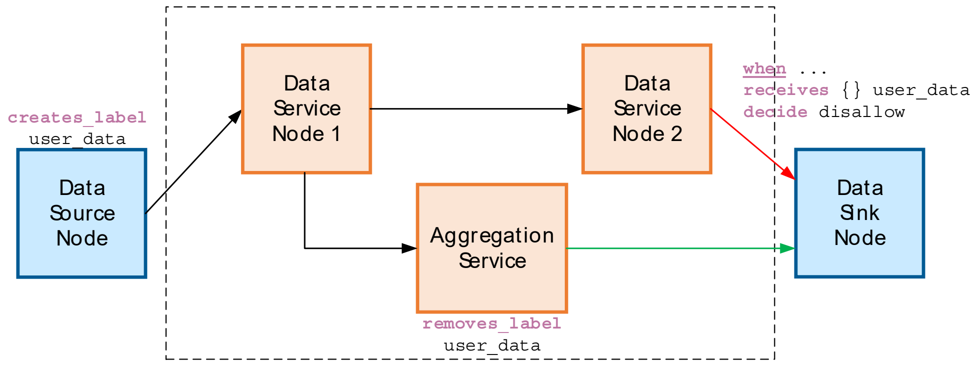


Figure 26: LUCON Camel route with aggregation service

If we take a look at an exemplary route, we can illustrate the usage of labels. For incoming data items from "Data Source Node", a label "user_data" is attached. This label is only removed if a service that provides the required anonymization or aggregation functionality (as the "Aggregation Service" does). The Data Sink is not allowed to receive data items that are labelled "user_data". Thus, no unaggregated user information may leave the connector (see Figure 26). This kind of policy set could either be deployed on data provider side or as well on data consumer side. The functionality of the app can be verified by the certification process and the application cannot be altered without altering its hash value and thus invalidating the applied signature. Another functionality is limiting data flows to dedicated containers on the data consumer side.

6.2.2 Transformation of IDS Usage Policies

LUCON is embedded into the Trusted Connector software stack. Policies can be submitted via IDS Usage Control Policy Negotiation (resp. Contract Negotiation). IDS policies need to be processed and translated into valid LUCON policies locally. LUCON supports the full IDS Contract Negotiation interaction pattern. This way, all solutions support a similar user experience.

6.3 Degree (D°)

While LUCON and MYDATA aim at providing usage control for existing applications and workflows, D° takes another approach. D° is a Domain Specific Language (DSL) for the development of data processing applications (so called Data Apps) and takes usage control into account from the beginning of the development. D° uses Java as host language. Through the use of Model Driven Software Development (MDSD) Data Apps which are developed with D° are transformed into Java applications which are finally compiled into executable applications.

It is important to notice that D° is not based on/an implementation of the XACML architecture. Nevertheless, XACML is referenced at various points in this chapter to adapt the general structure of this document, which has some kind of focus on XACML. Even though functionalities of D° are mapped to XACML constructs through this document, it is important to keep in mind that D° is a programming language which produces applications which have most of the components directly integrated. These applications do not exist separately from these components. Therefore, the described components are not separate services or even servers which is a big difference to the XACML architecture.

D° uses a special type system called nucleus. Initially, Nucleus was a module of D°, which has gained more and more functionality over time and is now available as a stand-alone project. Individual aspects of Nucleus are described later in Subsection 6.3.1.1.

Before detailed information about D° and its components is given, the following picture (see Figure 27) shows the big picture of D° with all relevant components and their relationships.

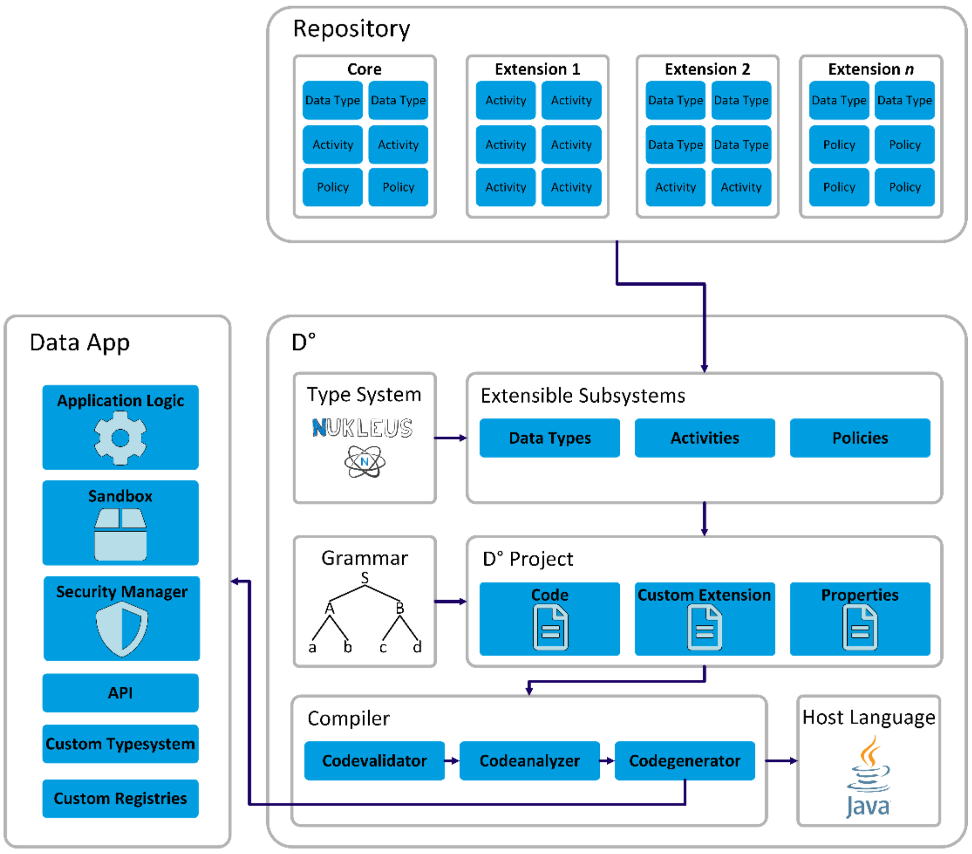


Figure 27: Big picture of D°

6.3.1 Communication Flows

Since usage control is one of the central features of D°, most relevant components are directly integrated into D° components. Therefore, there are only few communication flows required for operating a Data App. Nevertheless, a Data App can, at any time, establish a connection to external systems (e.g., DAPS or PAPs) if it is necessary for operation. Although a direct mapping of the XACML terminology to D° is not possible, the concepts can be mapped to each other with some drawbacks in accuracy.

Looking at the internal communication flows of an application developed with D°, only the data exchange between D° and Nucleus is relevant. Nucleus is responsible for the management of context information and provides an interface that can also be used in D° to access this context information.

6.3.1.1 Policy Administration Point

D° does not have a PAP as specified above, as the policy specification is done with the grammar of D° itself. The grammar of D° allows the definition of policies as well as the definition of where the policies need to be enforced. D° distinguishes between two different entities for policies: Constraints and Policies.

A Constraint is the representation of a single, simple rule which needs to be enforced (e.g., "Use data only after 00:00 01.01.2020"). Each constraint needs a piece of Java (or any compatible JVM language) code which is responsible of the enforcing. The following code block shows the textual D° definition of a constraint that ensures that a given content does not exceed a defined length.

This text will not give any detailed information about the syntax of D° since this is out of the scope of this document.

```
MaxLength := $Constraint(
    @attribute["maxLength", $Integer],
    @attribute["content", $Text]
)
```

The corresponding implementation for this constraint can be found in the next code block. It is written by using the JVM language Kotlin.

```
package de.fhg.isst.degree.policies.core.date

import de.fhg.isst.degree.policies.annotations.PolicyAnnotation
import de.fhg.isst.degree.policies.api.EmbeddedPolicyApi
import de.fhg.isst.degree.policies.execution.PolicyInputScope
import de.fhg.isst.degree.types.CompositeInstance
import de.fhg.isst.degree.types.PrimitiveInstance
import java.util.*

@PolicyAnnotation("MaxLength")
class MaxLengthConstraint : EmbeddedPolicyApi {

    override fun acceptPrecondition(input : PolicyInputScope): Boolean {
        val maxLengthInstance = input.get("maxLength")!!
        val contentInstance = input.get("content")

        val maxLengthValue = (maxLengthInstance as
                                      PrimitiveInstance).read().toIntOrNull()
        val content = (contentInstance as PrimitiveInstance).read()

        return content.length <= (maxLengthValue ?: 0)
    }

    override fun acceptSecurityManagerIntervention(input:
                                              PolicyInputScope): Boolean {
        return true
    }

    override fun acceptPostcondition(input: PolicyInputScope): Boolean {
        return true
    }

}
```

The second type of policy entities is the policy itself. It does not provide any code/logic for its enforcement. Instead, the policy is a container for an arbitrary amount of other policy entities. That way it is possible to construct complex policies by using (simple) constraints and policies. If a Data App is using a policy, during the enforcement all contained elements (policies and constraints) will be evaluated and only if none of the elements enforcement fails the execution is continued.

The following code block shows a simple policy which restricts the usage to a given time interval. To achieve this goal two constraints are placed in the policy: One for the start of the usage interval and one for the end.

```
AllowedTimeInterval := $Policy(
    @dependency["useNotBefore", $UseNotBeforeTimeStamp],
    @dependency["useNotAfter", $UseNotAfterTimeStamp]
)
```

The mapping of definitions to implementations for constraints is done by using the `@PolicyAnnotation` within the implementation. Each policy entity has an execution container which encapsulates the implementation. If there is no implementation available or needed (e.g., for policies) a special NOOP (no operation) execution container is used.

While policies only contain other policy entities, constraints contain actual attributes. For the enforcement, it is necessary to bind these attributes to actual values. If we take a look at the `MaxLength` example again, there are two attributes. The content attribute can only be resolved during runtime since it is probably that the length of some input variable should be restricted. But the maximal length should already be known during the development. To prevent later misuse (intentional or inadvertently) the known values should be set as early as possible.

To support this and also greatly increase the reusability of constraints with different attribute values, D° uses policy entity instances within Data Apps instead of policy entities. So, if within a Data App two values should not exceed a maximum length, two instances of the `MaxLength` constraint are used, each with its own set of attribute values.

The concept of instances is used for policies, too. It allows to create a policy instance which contains constraint instances and other policy instances.

If instances for constraints and policies are created, they must match to their definitions.

The following UML class diagram (see Figure 28) shows all classes and their relations which have been described in this section.

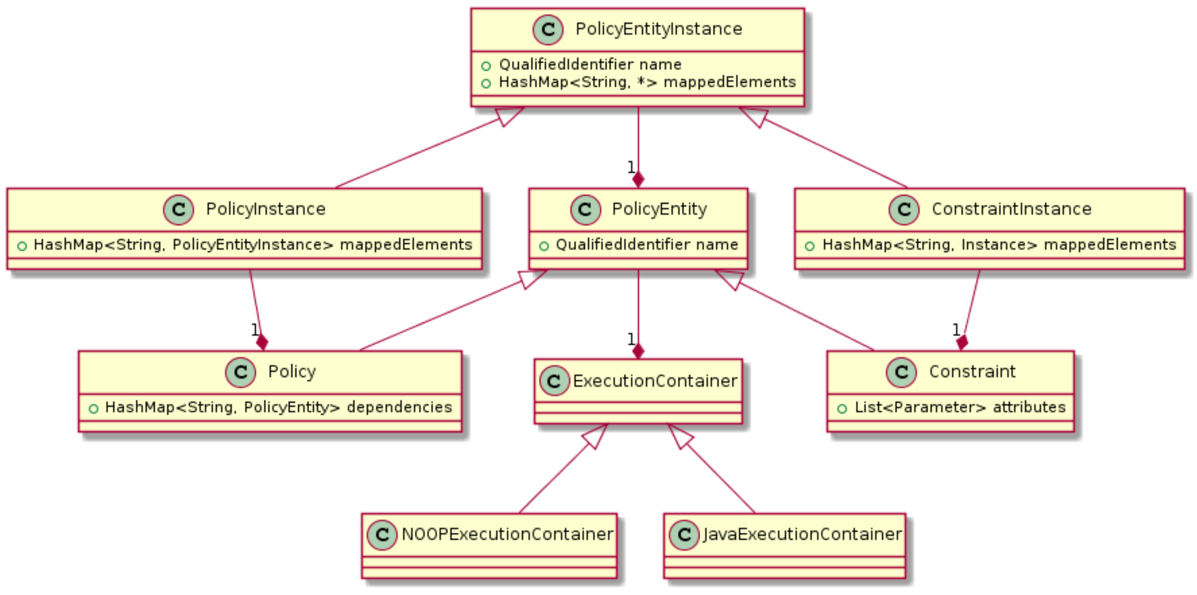


Figure 28: UML diagram describing all the PAP classes of D°

An important aspect to be addressed by D° is the simple and correct application of usage control in applications. For this purpose, the paradigm of policy-agnostic programming was implemented. This is based on the separation of application logic and usage control during development. In a later step it is necessary to connect these two components (inseparably).

For this reason, policies in D° are linked to other language constructs (for example, activities) and are not found directly in the D° code. The compiler then ensures during code generation that all linked policies are enforced in the correct places. In the context of D°, an activity is an atomic functionality, comparable to a syscall. The actual execution of the functionality is provided by an arbitrary set of code in the host language used.

Figure 29 shows an example of how the individual language constructs are linked together and then used in an application. Only the interface of the activity is visible to the developer. It does not matter to the developer whether and if so, how many policies are attached to the individual constructs.

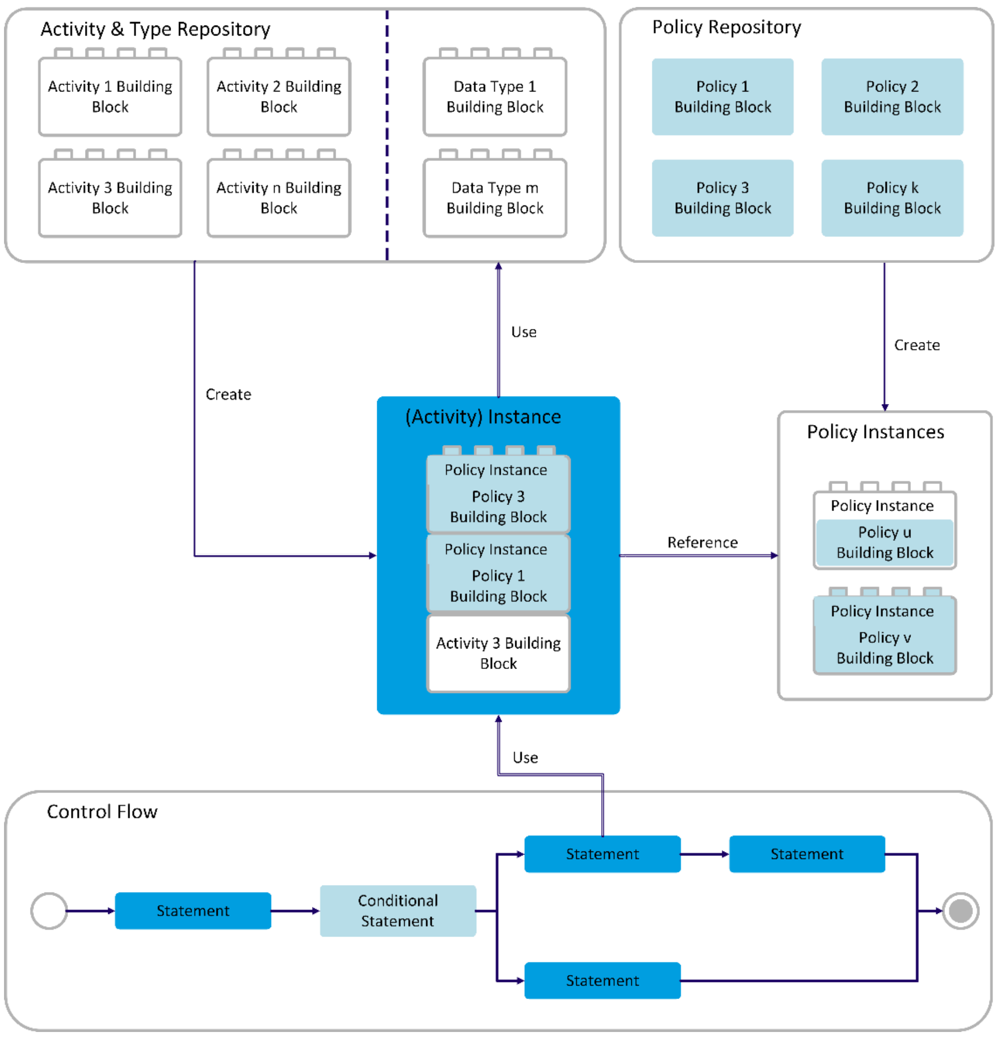


Figure 29: Example how the different components of D° are combined and used in applications

In addition, Nukleus provides a policy system, too. Based on the individual policy an enforcement in Nukleus or in D° is easier or better suited. Furthermore, not all policies can be enforced by Nukleus and D°. Therefore, the combination of the two policy systems enhances the expressiveness of D°.

Every action that is performed on a type instance emits an event to the Nukleus event bus. It is possible to define ECA-rules, which are the policies of Nukleus' policy system. These policies will be evaluated if relevant events occur in the event bus.

Figure 30 shows a simple example of the policy evaluation in Nukleus.

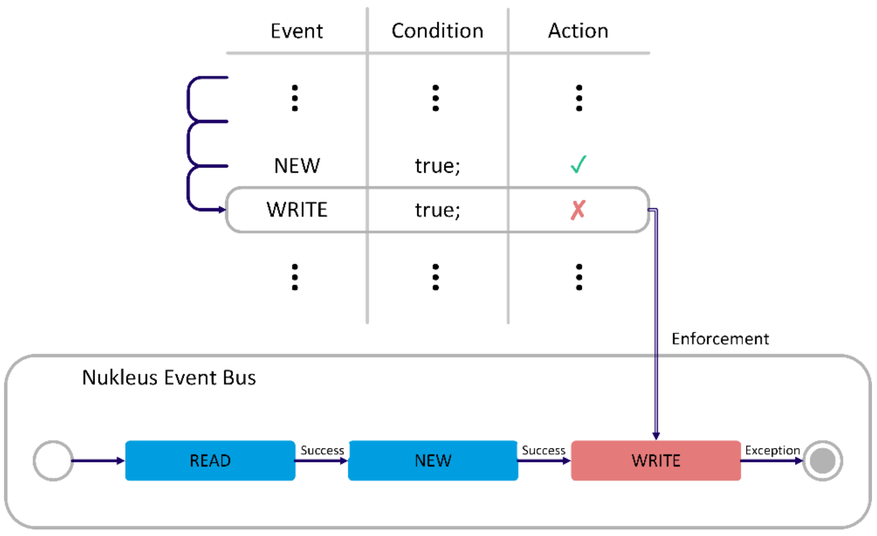


Figure 30: Illustration of the Nukleus policy system

6.3.1.2 Policy Information Point

D° allows to retrieve additional required information (e.g., contextual or historical) from different locations, depending on the nature of the required information.

The management of this information is done by the type system Nukleus. The functionality is known as the Policy Context of Nukleus. Nukleus provides an API which allows to perform CRUD-operations on the context. The context is a key-value store and supports different data types (e.g., counters). While the Nukleus Policy Context can be operated locally, it is possible to use it in conjunction with key-value databases (redis) in order to persist the context data and share the data between different instances of Nucleus and different usage control solutions.

Furthermore D° itself features a solution to store and persist context information. But this solution is deprecated because of the Nukleus functionality. It may be removed in a future version of D°.

6.3.1.3 Policy Enforcement Point

The Policy Enforcement Point (PEP) is responsible of intercepting actions and making decision request to the Policy Decision Point (PDP). This functionality is provided by two elements within Data Apps for D° itself:

On the one hand side the Data App contains a sandbox. Each called activity within a Data App is delegated to the sandbox. In the context of D° an activity is a (complex) functionality which is executed atomically within Data Apps. For every policy which must be enforced for this activity call, the sandbox checks if the precondition is fulfilled before executing the activity, and if its post condition is fulfilled after execution.

The second element is the DegreeSecurityManager. It allows to intercept the execution prior specific actions (e.g., writing a file, network communication). During these interceptions the relevant policies can perform checks and if one of them fails the desired action will not be executed.

For the policy system of Nukleus, the event bus acts like a PEP. It determines which policies are affected by each event and then evaluates the identified policies.

6.3.1.4 Policy Decision Point

The actual decision making if a policy is met or not is performed by the PDP. D° does not provide a single PDP. Instead of the implementation for each constraint is a PDP for exactly this constraint. Each constraint can perform checks prior execution, after execution, and if the security manager intercepts the execution.

The same holds true for Nukleus policies. Each policy in nucleus makes a decision on its own.

6.3.2 Integration Concept

Since D° is a programming language instead of a piece of software which is operated separately/in addition to applications, no cumbersome integration is required. As long as a system is capable of executing the D° host language or Docker, all requirements are met to execute Data Apps which have been developed with D°.

6.3.3 Transformation of IDS Usage Policies

While an automatic transformation of arbitrary IDS policies into the format D° uses is neither reasonable (- since D° has a limited set of enforceable policies just like every other solution -) nor technically feasible a manual translation is possible in many cases.

Since the IDS policies often allow arbitrary extensions (e.g., additional duties), a manual translation is often the only possibility to transform the policies into the D° format. D° contains implementations for various IDS policies which can be used in the transformation process. The transformed policies can be used like every other policy within D°.

6.4 Usage Control Technologies in Comparison

There are various technologies available in the IDS which can be used to implement Usage Control (see Section 0). In the following sub-sections, these technologies are compared based on general characteristics, their capabilities and policy languages as well as their location of implementation as described in the previous sections (see Section 4.3). All these tables also give a compact overview about all the technologies and their capabilities.

The comparison in Table 6 considers general characteristics such as purpose, documentation, license, programming language, management capabilities, graphical user interface and the technology readiness level (TRL).

Table 6: General Usage Control Technologies Comparison

	MYDATA	LUCON	Degree (D°)
Purpose	Usage Control Enforcement	Control of data flows, enforcement of obligations dependent on data flows	Development of data processing applications with integrated usage control.
Documentation	https://www.mydata-control.de/ https://developer.mydata-control.de/	https://industrial-data-space.github.io/trusted-connector-documentation/docs/usage_control/	T3 – Deliverables (available on request)
License	Open-Source SDK: Apache 2 Decision and Management Service: Proprietary License	Apache 2	Apache 2
Programming Language	Java	Java	D° Java as Host
Management	On premise hosting and Java library.	IDE & LUCON Environment	IDE & D° runtime environment
Graphical User Interface	Web UI	Web UI (IDS Policy Editor).	No
TRL	7-8	5	4

Table 7 considers the capabilities of the presented usage control technologies.

Table 7: Usage Control Technology Capabilities in Comparison

	MYDATA	**LUCON**	**Degree (D°)**
Support of IDS Meta Data	Yes	Yes	Yes
Policies address Message Content	Yes, for structured data	Indirectly via apps.	Yes
Support of additional Information Sources	Yes	Yes	Yes
Policies support Permissions	Yes	Yes	Yes
Policies support Obligations	Yes	Yes	Yes
IDS Contracts supported	Yes, Contracts are transformed.	Yes, Contracts are transformed.	Yes, but no automatic transformation.
Modification of Data in Transit	Yes	No	No
Extension Possibilities	Yes Own modifiers (data modification in transit), Own PIPs (information connection), Own PXP (actions)	Yes, open-source framework, customizable	Yes Own datatypes Own policies/constraint s Own activities

Table 8 presents all policy languages for the aforementioned usage control technologies. In addition, ODRL is added as technology-independent specification level policy language.

Table 8: Policy Language Comparison

	MYDATA	LUCON	Degree (D°)	ODRL
Format	XML	LUCON DSL	D° DSL	JSON-LD, XML, RDF Triple
Features	Bool Temporal Cardinal Time-Based External Sources Context-Awareness Execution Block Modify Monitor	Bool Temporal Cardinal Time-Based External Sources Context-Awareness Execution Block Modify Monitor	Bool Temporal Cardinal Time-Based External Sources Context-Awareness Execution Block Modify Monitor	Bool Temporal Cardinal Time-Based External Sources Context-Awareness Execution Block Modify Monitor
Default Policy Decision	Black- or Whitelisting (configurable)	Black- or Whitelisting	Blacklisting	Black- or Whitelisting
Standard	No	No	No	W3C
Version	Version 4.0 (04.11.2020)	Version 2.1 (31.07.2020)	Version 1.8.1 (25.11.2020)	Version 2.2

Table 9 compares the IDS Usage Control technologies with respect to their enforcement location (explained in Section 4.3). The table lists all possible enforcement locations for the respective technology. Which location is necessary depends on the respective Usage Control requirements that need to be fulfilled.

Table 9: Enforcement Locations

	MYDATA	LUCON	Degree (D°)
IDS Connector Message Bus **(intercept between Apps and within a route)**	Yes	Yes	No
Usage Control enabled IDS Apps **(enforce policies within Apps)**	Yes	Yes, via hash sum of containers.	Yes
Infrastructure **(e.g., Database, external Systems)**	Yes	No, in development	No
Client System and Services **(e.g., Operating System, Services on (external) servers)**	Yes	No	No

Table 10: Support of Policy Classes (from Section A.1) by Usage Control Technologies

	MYDATA	LUCON	Degree (D°)
1: Allow the usage of data	Yes	Yes	Yes[1]
2: Perpetual Data Sale (Payment once)	Yes	No	Yes[1]
3: Data Rental (Payment frequently)	Yes	No	No
4: Role-restricted Data Usage	Yes	No	Yes[1]
5: Connector-restricted Data Usage	Yes	Yes	No
6: System-restricted Data Usage	Yes	Yes	No

[1] No automatic transformation available. Not all cases can be covered because of arbitrary extensions, allowed in IDS Policies.

7: Purpose-restricted Data Usage Policy	Yes	Yes, via purpose bound apps.	Yes[1]
8: Event-restricted Usage Policy	Yes	No	Yes[1]
9: Interval-restricted Data Usage	Yes	Yes	Yes[1]
10: Duration-restricted Data Usage	Yes	Yes	No
11: Location Restricted Policy	Yes	No	No
12: Restricted Number of Usages	Yes	No	Yes[1]
13: Security Level Restricted Policy	Yes	Yes	No
14: Use data and delete it after	Yes	Yes, through containers	No
15: Modify data (in transit)	Yes	Via Apps	No
16: Modify data (in rest)	Yes	No	Yes[1]
17: Local Logging	Yes	Yes	Yes[1]
18: Remote Notifications	Yes	Yes	Yes[1]
19: Attach Policy when Distribute to 3rd Party	Yes	Yes	No
20: Distribute only if encrypted	Yes	No	Yes[1]

6.5 Discussion

The chapter presented three IDS Usage Control technologies for enforcing Data Usage Control. The three technologies differ in their capabilities, license and technology readiness level. The decision on what kind of technology to choose depends on the usage scenario. For example, integrating Usage Control capabilities into an application is supported by D° and MYDATA. Both technologies are integrated into the application at development time. While MYDATA needs to be integrated as Java library and used by function calls, D° is a programming language and ensures the correct integration of usage control into applications during compile time.

While D° applications can be developed by a single developer, it is intended that specialists for the development of the application logic and the policies can work separate from each other.

LUCON and MYDATA are integrated as part of the routing or as interceptor in the message routing. Hence, both integration concepts are very similar to each other. However, LUCON and D° support the use of labels, which must be specified as parameters in MYDATA. Moreover, MYDATA supports the modification of data in transit, which is not supported by LUCON. In contrast to LUCON, MYDATA is not open source, but has the highest TRL.

MYDATA offers different extension possibilities with their Open-Source SDK to develop own PEPs, PIPs and PXPs that are supported by the MYDATA policy language.

7. Data Provenance, Transparency and Accountability

Data provenance tracking is closely related, but also complementary to distributed data usage control. It has its origins in the domain of scientific computing, where it was introduced to trace the lineage of data. Data provenance tracking thereby allows finding out when, how and by whom data was modified, and which other data influenced the process of creating new data items.

This kind of traceability is similar to the data protection requirements a data controller is confronted with, so as to be able to fulfill the data subjects' right to access. It is also closely related to the question of proving compliance with contracts, agreements, or legal regulations. And data provenance tracking can be used to facilitate clearing in decentralized data ecosystems, since it is capable of providing information concerning data transactions and data usage.

7.1 Relationship between Data Provenance and Usage Control

However, while distributed data usage control is concerned with the enforcement of rights and duties when exchanging data across system boundaries, the focus of data provenance tracking is on transparency and accountability. In other words: While a Policy Enforcement Point (PEP) serving for distributed data usage control in most cases needs to be able to proactively intercept data usage actions within the control flow (i.e., preventive enforcement), a PEP for data provenance tracking only needs to passively observe, interpret and log data transactions and data usage for retrospective examination. In terms of usage control, this kind of enforcement is denoted as "detective enforcement". Despite this fact, a data provenance tracking infrastructure can be built upon the same PEPs as distributed data usage control. Furthermore, data provenance tracking does not require a policy specification language, but rather a specification of how observed actions are to be interpreted in terms of data flow or data usage (i.e., a so-called data flow semantics specification). By this means, data provenance tracking maintains a data flow model that keeps track of the particular representations of data items. This kind of information can also be leveraged for data usage control enforcement. For this a data provenance storage serves as a Policy Information Point (PIP), i.e., it is accessible via a PIP interface.

Please note that the requirements of data provenance tracking in terms of establishing trust into remote infrastructures is equivalent to the requirements of distributed usage control. If data provenance shall serve as context information for usage control enforcement, i.e., decisions by a PDP, it must have been collected by a trustworthy infrastructure. The same applies if data provenance shall be used to prove compliance with contracts or agreements.

7.2 Operating Principle and Architecture

The operating principle of data provenance tracking is very similar to the operating principle of distributed data usage control. The architecture of data provenance tracking is given in Figure 31. Data provenance tracking relies on passive monitoring technology (e.g., PEPs), which deliver events indicating data usage or data flows to be logged. For this, a PEP needs to convey a semantic description of the data usages or data flows its observed events indicate. The data provenance tracking infrastructure provides a data flow tracking component, which understands such semantics specifications. The PEP also needs to forward events together with metadata (including a unique identifier of the data's content), so that logged transactions can be attributed to data content when data provenance queried.

The PEP resides within the message routing component of the Connector (or Data App). It is registered at the data flow tracking component via a registry component (e.g., a local Policy Management Point, PMP). The same applies for the data flow tracking component. Thereby a PEP can query the registry for the communication interface of the local data flow tracking component, which is then used to deploy semantics specifications for its observed events and to forward actual events during operation.

Data provenance information is queried at a Provenance Dashboard, which is accessible via a Clearing House. The Provenance Dashboard returns a provenance graph for the unique identifier of a data asset. In case data provenance shall be used as contextual information for usage control, a provenance architecture with a single centralized data provenance storage component per usage control domain has the advantage that for queries for data provenance concerning a specific data item provenance must not be aggregated from several data provenance storage components so that events intercepted for usage control are not blocked for an arbitrarily long time.

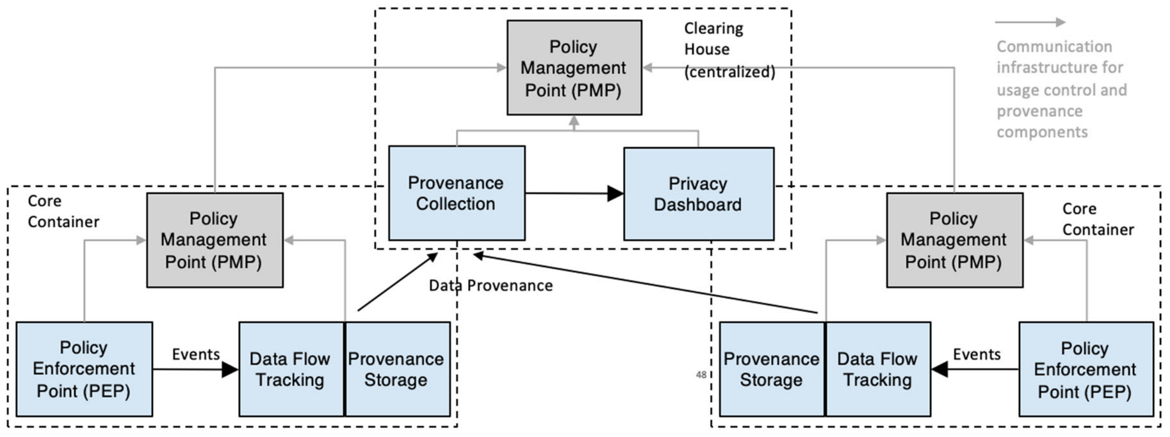

Policy Management Point (PMP)
Clearing House (centralized)
Communication infrastructure for usage control and provenance components
Provenance Collection
Privacy Dashboard
Core Container
Policy Management Point (PMP)
Core Container
Policy Management Point (PMP)
Data Provenance
Policy Enforcement Point (PEP)
Events
Data Flow Tracking
Provenance Storage
Provenance Storage
Data Flow Tracking
Events
Policy Enforcement Point (PEP)

Figure 31: Distributed Data Provenance Tracking Architecture

7.3 Provenance Tracking with Usage Control Framework MYDATA

Within the usage control infrastructure MYDATA there exists a Usage Control App where all events of the underlying Trusted Connector get intercepted. At this point the local provenance components are notified about the event. The provenance infrastructure needs to know semantic rules to interpret the event and whether the event induces a data flow. These rules are described in an information flow semantics. If the information flow goes from app to app or from connector to connector in the same domain, there is no need to inform central components, it is saved locally in the provenance storage. In case of an information flow across domain borders another semantic (remote information flow semantic) must be deployed in the receiving domain (the Usage Control App at the receiving connector) so that this domain is enabled to interpret the incoming information flow given an according event. In other words, this remote information flow semantics lets the receiving domain know how to handle the incoming data flow. After this step the actual information flow takes place.

In order to find the other connectors, there must be a registration place at a central location like the Clearing House. All connectors have to be registered there. Furthermore, all data that flows across several domains must be known by the central components of the provenance tracking infrastructure. So, each time a new data item leaves a domain the Provenance Collection must be updated with the information about the data (like an identifier) and the connector it belongs to. If one wants to know at some later point in time to which locations a specific data item has been transferred to, this provenance information can be requested at the Provenance Dashboard for the data associated to the own connector. The Provenance Collection keeps track of which data belongs to which connector so that it can ask at that connector for the local provenance. If the data has been transferred to further domains as well, this kind of provenance information is stored at the local provenance storage of the starting connector. In the next step the Provenance Collection asks at local storages of the other connectors about further transfer of the respective data item. After processing all connectors involved such information flows, the provenance collection aggregates all partial chunks of provenance information to obtain a complete view of the flows of the respective data item. The flows are then displayed at the Provenance Dashboard as a provenance tree. Only the connector who is the owner of a specific data item is allowed to request the according provenance tree.

7.4 Transformation of IDS Policies

Data provenance tracking can either be instantiated to track any observable transaction of any data item exchanged over the IDS, or to be only collected for specific data items where it is explicitly required by the data provider. In the latter case this can be done using a provenance tracking policy in IDS policy language in the information model, which is translated into a MYDATA policy at the data consumer, since so far data provenance tracking builds upon MYDATA.

8. Discussion and Future Work

This section discusses what usage control in its current state can achieve in the IDS and future work.

8.1 Discussion

The subsection discusses the current state of Usage Control in the IDS. Therefore, it summarizes the capabilities and limitations already mentioned in the previous sections and its implications.

8.1.1 Capabilities

By using the current state of usage control that is implemented into the IDS it can support developers and administrators in setting up correct data pipes that comply with policies and do not leak data via side effects. For example, usage control prevents IDS connectors from treating data in an undesired way such as forwarding personal data to public endpoints. The capabilities and reliability depend on the concrete Connector implementation and its trust level. In addition, usage control in the IDS can also be used as an audit mechanism, which creates evidence of a compliant data usage. For instance, usage control mechanisms can monitor and log usages of data.

With usage control, it is possible to modify the messages exchanged between endpoints to comply with a policy. For example, personal data can be removed, or data can be aggregated. It is furthermore possible to change the route of the package or drop it completely if demanded by a policy. Moreover, apps running on the connector can implement PEPs, which connect the usage control infrastructure and further enhance the functionality by allowing a more detailed control and data flow tracking. In addition, apps may use D° to implement data usage control capabilities.

Data usage control at the data consumer side is a special topic, which is also addressed in the document. There are possibilities to interact with third party software. For example, the PXP concept offers standardized interfaces that can be used in the policies and offers a flexible way to implement additional features. In addition, there is the possibility to encrypt the data at the provider side and decrypt the data at the consumer side (e.g., modify data in transit). In doing so, only the IDS connector is capable to perform the decryption operation. Moreover, the data processing within the IDS connector must adhere to the policies deployed.

8.1.2 Limitations

Usage control does only work within its ecosystem where it has the full control over the data. Achieving full control does also mean that there are cases that expect developers to integrate usage control components (such as PEP, PIP, PXP) into their application or services to fulfill usage restrictions (e.g., to interact with third party components). In most cases, developers have to integrate at least the PEP component to control data flows.

Although usage control uses several abstraction layers, there will always be a possibility to circumvent the system. One of the best-known examples for that is media disruption. For example, a usage control system may control taking screenshots and printing, but it cannot prevent a person to take a photo from the screen displaying the sensitive data. That said, if data leaves the ecosystem, it needs additional protection (such as encryption) in order to keep control over the data.

Usage control is no hard security feature such as cryptography, which can be proved mathematically. It is rather a complementary solution to have more fine-grained control over data flowing in a system and goes well together with organizational rules (see Section 0). In addition, it is rather an extension to access control than replacing it.

Implementing a usage control technology does not automatically establish trust in an endpoint. It necessarily builds upon an existing trust relationship such as existing contracts and a secure computing environment like highly trusted platforms (such as the IDS Trusted Connector).

When physical access is granted to administrators, protection against data theft by persons with malicious intents is almost impossible to prevent. As the administrator of the data consumer will act on behalf of the data consumer organization's management, he is a reasonable attacker for usage control enforcement. It is part of future work to evaluate possible countermeasures.

8.1.3 Implications

Implementing usage control into an existing system has various implications. Creating events, the decision-making and the transfer of events between the affected components takes extra time as well as some computational power. Besides, all usage control components need memory to persist information or to perform the computation. In sum, it will reduce the performance of the overall system and demands machines with more power.

As already stated, the basic idea of usage control is to control the dataflow. In a case where a developer enhances an application with usage control technology, he needs to integrate at least one PEP. Depending on the complexity of the enforcement, he needs to integrate even more than one PEP within one or several applications. As all of those integrations also need planning and testing, it increases the development and testing time and effort in comparison to a system without usage control.

In addition to the enforcement components, the system needs policies. Therefore, a policy specification process needs to be established. During this process, the policy experts of the data owner have to collect information about how others should use the data. This process costs additional time and communication effort for the data owners and leads in the end to higher costs.

8.2 Future Work

The subsection addresses future work for usage control in the IDS. We focus on policy negotiation, provenance tracking and the usage control object, which are currently addressed in IDS-related research projects. Other topics such as policy lifecycle and evolution are neglected.

8.2.1 App Store

A Data Provider (or a Service Provider) may run various Data Apps from their own or from the App Store. In this case, the Data Provider controls the data flow and provides the data to the Data Users. Additionally, a Data Consumer may be allowed to select, host and execute third-party Data Apps as well. In any case, the data must be treated in accordance with the usage control policy agreed upon by all parties involved.

Possible solutions to ensure the data sovereignty in the Data Apps are certifying the Data Apps, filtering the data before it is processed by the Data Apps and enforcing policies via usage control technologies.

8.2.2 Automated IDS Contract Negotiation

IDS contract negotiation is a central IDS concept and important for data ecosystems. The objective of negotiations is to find agreements about the terms of data usage which reflect the needs of all partners. This is an optimization problem that needs a sophisticated approach to find the best solution.

IDS contracts are composed of a set of parameterized contract clauses. These parameter values are negotiated and a final bid results with agreed values for all parameters. The negotiation is controlled by the applied strategies of both partners. The strategies are configured by so-called profiles, expressing which parameters a partner wants to minimize or maximize, and the partner’s negotiation preferences.

The research project “Industry 4.0 Legal Testbed” deals with automated contract negotiations in the area of production and logistics. One key concept in this project is that the negotiation can be conducted automatically by distributed computer programs. In contrast to human negotiations, it is possible to consider much more solutions in less time in this process.

In a project within the Fraunhofer Cluster of Excellence Cognitive Internet Technologies (CCIT) the concept of autonomous and automated negotiations was adapted for data usage scenarios. As in the Legal Testbed, the framework GeniusWeb developed by TU Delft and used in ten yearly ANAC negotiation competitions, is applied for automated negotiations. GeniusWeb is an open architecture for negotiation via the internet and provides many reusable components, e.g., utility functions and negotiation strategies for participants. For the use in the domain of IDS, GeniusWeb was adapted so that the communication is carried out completely using IDS connectors and IDS messages.

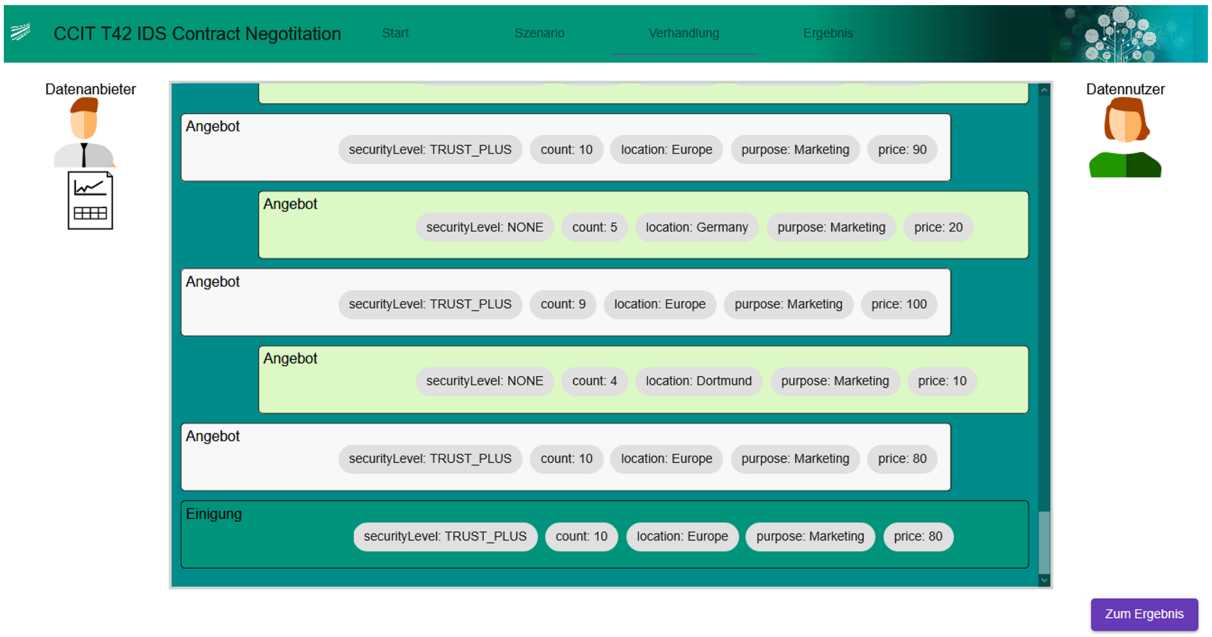
The negotiation is configured by a mapping table, which contains all negotiable parameters forming the domain for the negotiation. They are the parameters in IDS Contract JSON-LD templates for usage control clauses being also stored in the table. Furthermore, the table contains the utility function for each parameter for a participant, i.e., in how far he wants to minimize or maximize the value.

To start an IDS Contract negotiation, the data provider must be in a waiting state and the data consumer initiates the communication by triggering the negotiator component. Now a series of so-called bids are exchanged between the connectors as solution proposals and counter proposals. If a received solution is acceptable by a participant, he sends an appropriate message to end the negotiation, and the final contract is assembled from the registered clauses being filled with the negotiated parameter values.

The whole process was demonstrated by a prototype build up by four Docker containers. The first one contains the data consumer and the demonstrator web interface. The second container represents the data provider. Each of them has mapping tables for different show cases. The last two containers implement the corresponding IDS connectors.

Figure 32 shows the last few steps of a long negotiation over five contract clauses (five variables), in which solution proposals are exchanged as bids and a solution was finally accepted by both partners. The strategies used in this prototype experiment try to maximize each partners total linear-additive utility of each bid, expressed in each partner's individual utility profiles. This explains the missing obvious logic in the order of bids. However, GeniusWeb also provides more sophisticated strategies controlled by opponent models. The final IDS contract is generated by using the values of the agreed solution bid for the parameters of all IDS contract clauses applied.

Figure 33 shows the agreement bid and the generated IDS Contract.



Angebot
securityLevel: TRUST_PLUS | count: 10 | location: Europe | purpose: Marketing | price: 90

Angebot
securityLevel: NONE | count: 5 | location: Germany | purpose: Marketing | price: 20

Angebot
securityLevel: TRUST_PLUS | count: 9 | location: Europe | purpose: Marketing | price: 100

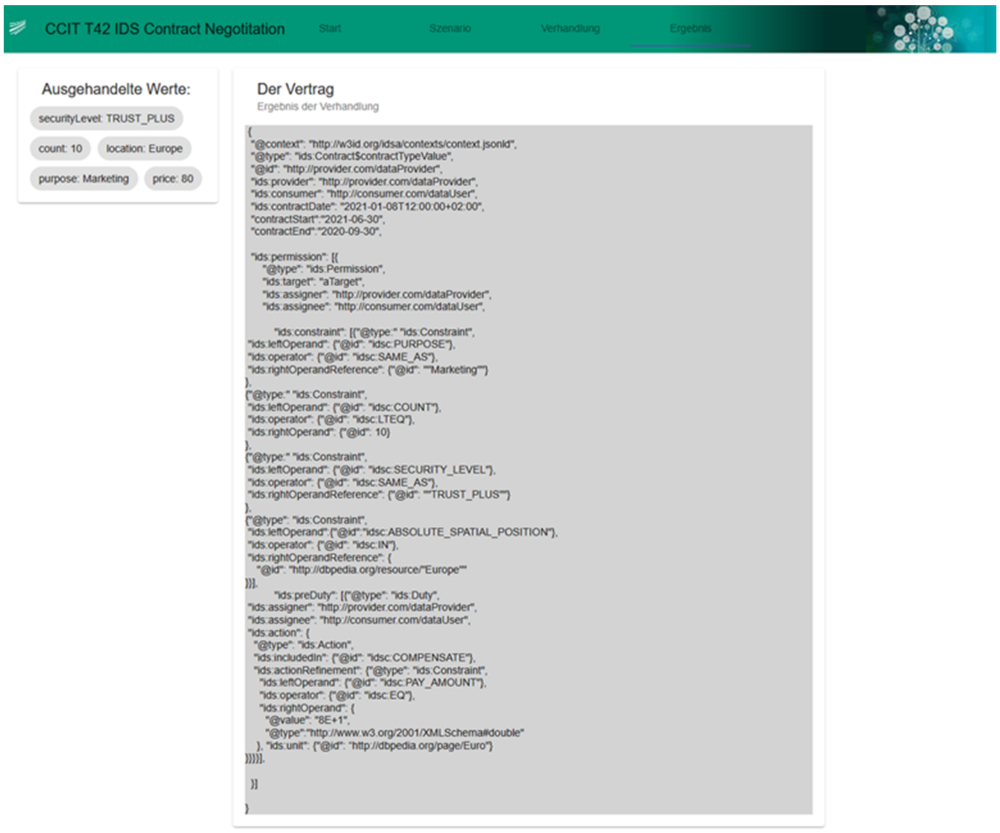
Angebot
securityLevel: NONE | count: 4 | location: Dortmund | purpose: Marketing | price: 10

Angebot
securityLevel: TRUST_PLUS | count: 10 | location: Europe | purpose: Marketing | price: 80

Einigung
securityLevel: TRUST_PLUS | count: 10 | location: Europe | purpose: Marketing | price: 80

Figure 32: Negotiation protocol

The concept of automated contract negotiation is useful when more flexibility is desired to create individual appropriate IDS data usage contracts. Its strengths become visible especially in time critical or repetitive situations. The last criterion is already fulfilled when a data resource is requested by multiple consumers or when one consumer has to choose between different data providers. Automated contract negotiation is even inevitable if there is a large solution space and the concealed space for agreements is small.



Ausgehandelte Werte:

securityLevel: TRUST_PLUS
count: 10
location: Europe
purpose: Marketing
price: 80

Der Vertrag

Ergebnis der Verhandlung

```
{
 "@context": "http://w3id.org/idsa/contexts/context.jsonld",
 "@type": "ids:Contract$contractTypeValue",
 "@id": "http://provider.com/dataProvider",
 "ids:provider": "http://provider.com/dataProvider",
 "ids:consumer": "http://consumer.com/dataUser",
 "ids:contractDate": "2021-01-08T12:00:00+02:00",
 "contractStart":"2021-06-30",
 "contractEnd":"2020-09-30",

 "ids:permission": [{
    "@type": "ids:Permission",
    "ids:target": "aTarget",
    "ids:assigner": "http://provider.com/dataProvider",
    "ids:assignee": "http://consumer.com/dataUser",

       "ids:constraint": [{"@type:" "ids:Constraint",
 "ids:leftOperand": {"@id": "idsc:PURPOSE"},
 "ids:operator": {"@id": "idsc:SAME_AS"},
 "ids:rightOperandReference": {"@id": ""Marketing""}
},
{"@type:" "ids:Constraint",
 "ids:leftOperand": {"@id": "idsc:COUNT"},
 "ids:operator": {"@id": "idsc:LTEQ"},
 "ids:rightOperand": {"@id": 10}
},
{"@type:" "ids:Constraint",
 "ids:leftOperand": {"@id": "idsc:SECURITY_LEVEL"},
 "ids:operator": {"@id": "idsc:SAME_AS"},
 "ids:rightOperandReference": {"@id": ""TRUST_PLUS""}
},
{"@type": "ids:Constraint",
 "ids:leftOperand":{"@id":"idsc:ABSOLUTE_SPATIAL_POSITION"},
 "ids:operator": {"@id": "idsc:IN"},
 "ids:rightOperandReference": {
   "@id": "http://dbpedia.org/resource/"Europe""
}}],
       "ids:preDuty": [{"@type": "ids:Duty",
 "ids:assigner": "http://provider.com/dataProvider",
 "ids:assignee": "http://consumer.com/dataUser",
 "ids:action": {
  "@type": "ids:Action",
  "ids:includedIn": {"@id": "idsc:COMPENSATE"},
  "ids:actionRefinement": {"@type": "ids:Constraint",
    "ids:leftOperand": {"@id": "idsc:PAY_AMOUNT"},
    "ids:operator": {"@id": "idsc:EQ"},
    "ids:rightOperand": {
      "@value": "8E+1",
      "@type":"http://www.w3.org/2001/XMLSchema#double"
   }, "ids:unit": {"@id": "http://dbpedia.org/page/Euro"}
}}}}],

  }]

}
```

Figure 33: Assembled IDS contract with filled in parameters

8.2.3 Contract Negotiation Evaluation

The contract negotiation process is assessed with respect to its interoperability, usability and completeness. The evaluation of planning and developing this process emphasizes on the urge of considering human interaction, although, keeping it highly automated. For example, on the Data Consumer side, an authorized user shall initiate the process. Moreover, authorized users shall update their mapping tables and interrupt the process, in order to give inputs, when needed. Therefore, a Graphical User Interface might be needed to facilitate this interaction.

In addition, the completeness of the contract negotiation process is evaluated by verifying whether its negotiable parameters are paired to the usage control statements of the IDS contracts.

A. Appendix

A.1. Description of Policy Classes

The 21 policy templates introduced in Section 5.2 are explained as follows:

A.1.1. Allow or inhibit the usage of the data

This class of policy is an abstract category that either gives permission or prohibits a specified IDS Data Consumer to operate specified action(s) on the Data Asset without further restrictions. As mentioned before, the action "use" is a very generic action that is utilized to express all targeted usages and therefore, includes fine-grained actions such as "read", "distribute", "print", "delete", "display", and so on. When the permission to “use” the data is issued, the Data Consumer is allowed to operate any of the aforementioned actions on the data. In order to restrict the type of the actions that are allowed to be operated, the policy must address a particular action. For example, in a whitelisting approach, you want to allow your Data Consumer to read and display the data, therefore, you specify a policy that only permits the “read” and “display” actions.

A.1.2. Restrict the data usage to specific connectors

The context of IDS allows assigning more than one connector to a particular IDS party. Therefore, this class of policy addresses the condition of restricting the usage of data to specific connectors of the specified IDS Data Consumer.

A.1.3. Restrict the data usage to a group of systems or applications

The Data Usage Control scenarios demand further restrictions on the policies that either allow or inhibit the usage of data. In order to apply the requested restrictions such as restricting the data usage into the specific systems, the corresponding policy conditions are specified. This implies that the usage of the data is permitted or prohibited when the specified conditions are met. In a policy, the conditions are indeed the prerequisite to operate the action. For example, you can instantiate a policy of this class that allows only a specified risk management system or application to use your data. This policy class faces few limitations, i.e., in order to evaluate the conditions, it requires that the systems and the applications be certified. Thus, a Data Usage Control technology can validate the certifications and enforce the policy.

A.1.4. Restrict the data usage to a group of users

Additionally, an IDS Data Provider may demand to restrict the usage of the data to a specific group of users. This condition addresses either the membership or the role of the users. In order to enforce such a policy, a Data Usage Control technology has to check whether a user is a member of the specified organization or has a specific role from authorized resources.

A.1.5. Restrict the data usage to specific locations

This class of policy addresses the restriction on the location of the Data Consumer. This condition refines the permitted or prohibited locations of the Data Consumers by region or bounding polygons. A bounding polygon shapes an area by indicating a set of geographical points. A policy may allow a specified Data Consumer to use data only when the assigned connector is located within the permitted area.

A.1.6. Restrict the data usage for specific purposes

This category represents another highly demanded class of policy that restricts the usage of data to specific purposes. In order to formulate the purpose of usage in a policy and later on, enforce it to the system, we need to define licenses and certifications. This concept is still evolving in the context of International Data Spaces. “If the purpose is risk management, then allow the usage of data and else if the purpose is marketing, then inhibit the usage of data” is an example policy that is instantiated from this policy class.

A.1.7. Restrict the data usage when a specific event has occurred

This class of policy represents the permission or prohibition of using data under specific conditions; in the circumstances that the usage of data must be restricted due to the occurrences of specific events, a policy of this type can be constructed. Similar to the previous classes and in order to specify policies such as “if an accident occurred, provide permission to read the geographic location” or “provide permission to a Data Consumer to use the data during the exhibition”, we need to formulate the events. Therefore, a Data Provider can specify the conditions that address “when accident occurred” or “during the exhibition”. The assumption is that a set of possible events are defined in the context of International Data Spaces and are available to the ones who specify the policies. As a result, a data usage control technology is able to interpret the events and restrict the data usage accordingly.

A.1.8. Restrict the data usage to the security level of the connectors

The information model of IDS differentiates the connectors with respect to their security levels (i.e., base, trust and trust plus). This class of policy addresses the condition of restricting the usage of data to the security level of the connectors. Depending on what is specified in the condition, an assigned connector of a Data Consumer is allowed to use the data.

A.1.9. Restrict the data usage to a specific time interval

The International Data Spaces customers require further time-based constraints, i.e., allow or inhibit the usage of data in a specified time interval. A policy, for example, specifies the permission to use the data from the beginning of September 2019 to the end of November 2019. The date and time conditions can be expressed in different ways. However, it is important that the system is able to interpret the date and time conditions that are specified in the policies. For example, if “`xsd:dateTimeStamp`” is used as the data type that defines the date and time in the policy, the system must also be able to read it and understand it.

A.1.10. Restrict the data usage to a specific time duration

Another time-based constraint is to restrict the usage of data to a specific duration of time. For example, an instantiated policy from this policy class may allow a Data Consumer to use the data for a duration of three months. The permitted period may start from a given date and time. Moreover, the corresponding data type (e.g. “`xsd:duration`”) must be interpreted the same in all systems.

A.1.11. Use the data not more than N times

This class of policy demands to restrict the numeric count of executions of the action. For example, a policy specifies that the data can be printed only once or it can be displayed not more than ten times or in total, data cannot be used more than N times. We can only apply this kind of policies to the cases in which, the usage of data is countable. Therefore, a mechanism is needed that counts the usage of data and store it securely and locally, in order to enforce such a policy.

A.1.12. Use data and delete it after

This class of policy gives permission to a specified IDS Data Consumer to use the Data Asset and requires the Data Consumer to delete the data after. A policy of this type shall be refined to clarify when the data must be deleted; it shall be immediately after the usage or after a delay period or before a specified date and time.

A.1.13. Modify data (in transit)

In all aforementioned cases, the policies allow the users to use the entire data, without modifications, after the conditions are met. However, there might be cases where data must be modified or partially anonymized before it is allocated to the user. The data modification must be done before the permission to use the data is granted. This class of policy represents the Data Usage Control use cases demanding to modify the data in transit; a Data Usage Control technology intercepts the data that is transmitted and applies the modifications on them.

A.1.14. Modify data (in rest)

This class of policy demands for the data modifications or anonymizations before the permission to use the data is granted. In contrast to the previous policy class, it demands the modifications to be done when data is stored in a database. The Data Consumer is only allowed to use the data after certain modifications have been applied to the stored data.

A.1.15. Log the data usage information

The IDS Data Provider requests to log the information of transferring data from their sites to their Data Consumer sites. Although, logging the information is a part of the International Data Spaces infrastructure, a Data Usage Control technology can occasionally apply the logging policies to the systems and log the usage information locally, as well. For example, it might log the information about the data anonymizations.

A.1.16. Notify a party or a specific group of users when the data is used

The studies show that the International Data Spaces Data Providers request to be notified in a stated situation. For example, we can specify policies of this type to request to notify the Data Providers, when their data has left their sites or when it is delivered to the data consumers. The formats and possibilities of the notifications depends on which platform is used; whether it is the notification system of International Data Spaces or, for example, a mailing system.

A.1.17. Attach policy when distribute the data to a third-party

An IDS Data Provider may specify additional data usage policies to be provided to the third parties. Here, the Data Consumer is obliged to pass the specified Data Usage

Control policy to the third-party and demand for an agreement before further distributing the data.

A.1.18. Distribute the data only if it is encrypted

In most of the cases, a Data Provider specifies a policy to give permission to one or more data consumers to use the data. Although, there might be cases in which the Data Consumer requires permission to further distribute the data to other users or third parties. This class of policy exclusively addresses the state of the Data Asset in case of sharing it. For example, you can specify a policy of this type to demand your Data Consumer to share your data only if it is encrypted.

A.1.19. Perpetual data sale restrictions

The IDS platform provides the possibility to the Data Providers to sell their Data Assets. A Data Consumer has to fulfill the conditions that are specified in a data sale contract in order to buy the Data Assets. For example, a one-time payment has to be made. This class of policy addresses the conditions that are associated to a data sale contract.

A.1.20. Rental data restrictions

In contrary to the previous class of policy, this category addresses the conditions that are associated to a data rent contract. For example, a Data Usage Control technology has to check frequently whether the monthly fee which is specified in the contract is paid by the Data Consumer.

A.1.21. Restrict the data usage to specific state

This category represents a condition in which the usage of data is restricted to a specific state. This condition refers to an environment state but not the state of the Data Asset. Therefore, it is about the state of the contract and the connectors. If the contract is terminated or if the firewall is activated are examples for this restriction. The state of the Data Consumer connector and the contract must be known by the Data Usage Control technology, so the application can check whether the condition is fulfilled and issue permission to the Data Consumer to use the Data Asset.

A.2. More Information about Usage Control in the International Data Spaces

A webinar on data usage control can be found here:

- Webinar: https://www.youtube.com/watch?v=6KHqxMKOmHo

We present references to further information and the responsible Fraunhofer institute as well as the responsible contact person next.

A.3. MYDATA Control Technologies

Further information about the MYDATA Control Technologies can be found here:

- Website: https://www.mydata-control.de/
- Developer Website: https://developer.mydata-control.de/
- Source Code: https://git.iese.fraunhofer.de/ind2uce
- Binaries: https://search.maven.org/search?q=g:de.fraunhofer.iese.ind2uce

The responsible institute is Fraunhofer IESE in Kaiserslautern, contact persons: Dr.-Ing. Christian Jung and Andreas Eitel.

A.4. Logic based Usage Control

Further information about the Logic-based Usage Control can be found here:

- https://industrial-data-space.github.io/trusted-connector-documentation/docs/usage_control

The responsible institute is Fraunhofer AISEC in Garching, contact person: Gerd Brost

A.5. Degree (D°)

Further information about Degree can be found here:

- https://www.isst.fraunhofer.de

The responsible institute is Fraunhofer ISST in Dortmund, contact person: Fabian Bruckner.

A.6. Data Provenance

Further information about Data Provenance can be found here:

- https://www.iosb.fraunhofer.de/de/projekte-produkte/industrial-data-space.html
- https://www.iosb.fraunhofer.de/de/kompetenzen/bildauswertung/interaktive-analyse-diagnose/forschungsthemen/digitale-souveraenitaet/datentransparenz-privacy-insight.html

The responsible institute is Fraunhofer IOSB in Karlsruhe, contact person: Dr.-Ing. Pascal Birnstill.

A.7. GAIA-X and IDS

In the GAIA-X and IDS position paper [18] the IDSA demonstrates how the elements of the International Data Spaces Reference Architecture Model fits to the GAIA-X principles and architecture elements described in the Technical Architecture whitepaper. The topic of Usage Control starts at page 24.

The document can be found here:

- https://internationaldataspaces.org/download/19016/

Glossary

D°	Degree
DAPS	Dynamic Attribute Provision Service
DSL	Domain Specific Language
IDS	International Data Spaces (previously Industrial Data Spaces)
LUCON	Logic based Usage CONtrol
MDSD	Model Driven Software Development
MYDATA	MYDATA Control Technologies
ODRL	Open Digital Rights Language
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
ParIS	Participant Information Service
PII	Personal Identifiable Information
PIP	Policy Information Point
PIR	Participant Information Registry
PMP	Policy Management Point
PXP	Policy Execution Point
RC	Release Candidate
TRL	Technology Readiness Level
UML	Unified Modeling Language
XACML	eXtensible Access Control Markup Language
UUID	Universally Unique Identifier

References

[1] B. Otto, S. Lohmann, S. Auer, G. Brost, J. Cirullies, A. Eitel, T. Ernst, C. Haas, M. Huber, C. Jung, J. Jürjens, C. Lange, C. Mader, N. Menz, R. Nagel, H. Pettenpohl, J. Pullmann, C. Quix, J. Schon, D. Schulz, J. Schütte, M. Spiekermann and S. Wenzel, "Reference Architecture Model for the Industrial Data Space 3.0," Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V. AND Industrial Data Space e.V., München, 2019.

[2] B. Parducci, H. Lockhart und E. Rissanen, „eXtensible Access Control Markup Language (XACML) Version 3.0," OASIS Standard, 22 January 2013. [Online]. Available: https://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html. [Zugriff am 16 08 2019].

[3] P. I. 4.0, „Zugriffssteuerung für Industrie 4.0 - Komponenten zur Anwendung von Herstellern, Betreibern und Integratoren," Bundesministerium für Wirtschaft und Energie (BMWi), Berlin, 2018.

[4] A. Eitel, C. Jung, C. Haas, C. Mader, G. Brost, J. Schütte, J. Pullmann, J. Zrenner und P. Birnstill, „Usage Control in the Industrial Data Space," Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung, IESE-Report No. 056.17/E, Kaiserslautern, 2017.

[5] Wikipedia, „Data Universal Numbering System," [Online]. Available: https://en.wikipedia.org/wiki/Data_Universal_Numbering_System. [Zugriff am 16 08 2019].

[6] M. C. Mont und S. Pearson, „Sticky Policies: An Approach for Managing Privacy Across Multiple Parties," Computer, pp. 60-68, 2011.

[7] N. Mercer, „Introducing Windows Information Protection," Microsoft, 28 02 2018. [Online]. Available: https://techcommunity.microsoft.com/t5/Windows-Blog-Archive/Introducing-Windows-Information-Protection/ba-p/166564. [Zugriff am 16 08 2019].

[8] Solid Community, „Web Access Control (WAC, Version 0.5.0)," 2021. [Online]. Available: https://github.com/solid/web-access-control-spec/tree/58eae0f548a11c02c30bf2f4a1d620f5ed147490. [Zugriff am 04 02 2021].

[9] C. Jung, A. Hosseinzadeh da Silva, A. Eitel und R. Brandstädter, „Documentation of Usage Restrictions, Language Constructs and their Technical Refinement," Fraunhofer Gesellschaft zur Förderung der angewandten Forschung, IESE-Report No. 049.18/E, Kaiserslautern, 2018.

[10] W3C ODRL Community Group, „ODRL Overview," [Online]. Available: https://www.w3.org/community/odrl/. [Zugriff am 16 08 2019].

[11] IDS, „The International Data Spaces (IDS) Information Model version 4.0.0.," 11 08 2020. [Online]. Available: https://github.com/International-Data-Spaces-Association/InformationModel/tree/master. [Zugriff am 01 02 2021].

[12] H. Bastiaansen, G. Bramm, J. Ceballos, M. Gall und M. Kolenstart, „Specification: IDS Clearing House 1.0,“ International Data Spaces Association, Dortmund, 2020.

[13] S. Bader, F. Bruckner, G. Böge, D. O. Kubitza, J. Langkau, D. Murthy und R. Nagel, „Specification: IDS Meta Data Broker 1.0,“ International Data Spaces Association, Dortmund, 2020.

[14] Wikipedia, „Resource Description Framework,“ [Online]. Available: https://en.wikipedia.org/wiki/Resource_Description_Framework. [Zugriff am 16 08 2019].

[15] RDF Working Group, „Resource Description Framework (RDF),“ [Online]. Available: https://www.w3.org/RDF/. [Zugriff am 16 08 2019].

[16] Wikipedia, „ODRL,“ [Online]. Available: https://en.wikipedia.org/wiki/ODRL. [Zugriff am 16 08 2019].

[17] R. Iannella, M. Steidl, S. Myles und V. Rodríguez-Doncel, „ODRL Vocabulary & Expression 2.2,“ 15 02 2018. [Online]. Available: https://www.w3.org/TR/odrl-vocab/. [Zugriff am 16 08 2019].

[18] B. Otto, H. Pettenpohl, A. Rubina, J. Langkau, A. Eitel, J. Gelhaar, A. Teuscher, K. Mitani, A. M. Schleimer, M. Hupperz, C. Lange, M. Huber, D. Stingl, N. Jahnke und E. Loukipoudis, GAIA-X and IDS, Dortmund: International Data Spaces Association, 2021.

OUR MEMBERS

A! Aalto University	ADVANEO	agmadata	Allianz	ATLANTIS ENGINEERING	Atos
Audi	bill-X Collect. Manage. Execute.	Bitergo	Boehringer Ingelheim	Brainport Industries	bdr. BUNDESDRUCKEREI
CAICT 中国信息通信研究院	DIGITAL TECHNOLOGY SUPERCLUSTER	CDQ SHARING DATA EXCELLENCE	cea	Cefriel POLITECNICO DI MILANO	iti Information Technologies Institute
CHALMERS UNIVERSITY OF TECHNOLOGY	COSMOPlat	Cybus	CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE	DAIMLER	DATAAHEAD
DATATRONiQ	DB SCHENKER	Deutsche Bank	DGZfP DEUTSCHE GESELLSCHAFT FÜR ZERSTÖRUNGSFREIE PRÜFUNG e.V.	DHBW Duale Hochschule Baden-Württemberg	Digital Green
DIMECC	eccenca command your data!	ELDORADO	ENGIE	ENGINEERING	8760 Fastems
fir an der RWTH Aachen	FIWARE FOUNDATION	Fraunhofer	GERMAN EDGE CLOUD	GeoNet.MRN	GESIS GESELLSCHAFT FÜR INFORMATIONSSYSTEME Ein Unternehmen der Salzgitter Gruppe
Google	Haier	HITACHI	HOLONIX BRING THINGS TO LIFE	HUAWEI	i2cat
automotive engineering iav	IBM	ILVO Flanders Research Institute for Agriculture, Fisheries and Food	imec	Imperial College London	Institut Mines-Télécom
INDUSTRIE 2025 INDUSTRIA INDUSTRY	INNOPAY	innovalia ASSOCIATION	Insight	INSOLAR	Irish Manufacturing Research
ITI INVESTIGATE TO INNOVATE	Klarrio STREAMING AHEAD	KNORR-BREMSE	KOMSA DIE BESSERE VERBINDUNG	10 01 LAKES	LOSEC LEADERS IN SECURITY
Leibniz Universität Hannover	Logata Digital Solutions	Microsoft	minnosphere company of .msg	.msg	mtc Manufacturing Technology Centre

nexedi

nicos AG

NTTGroup

olmogo
data ownership solutions

orbiter

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

pi-lar GmbH

POLITECNICO
MILANO 1863

pwc

Quin Scape

realworldsystems

RI. SE

REWE GROUP

RITTAL

SALZGITTER AG
Stahl und Technologie

SAP

SCHAEFFLER

SETLOG

SICK
Sensor Intelligence.

SIEMENS
Ingenuity for life

Signal Solutions

SINTEF

SMEV AG
Smart Mobility Evolution

SOFIA UNIVERSITY
ST. KLIMENT OHRIDSKI

TARC
TUNKU ABDUL RAHMAN
UNIVERSITY COLLEGE

TATA
TATA STEEL

tecnalia
Inspiring Business

Tekniker
MEMBER OF BASQUE RESEARCH & TECHNOLOGY ALLIANCE

T · ·

thyssenkrupp

TIM

TNO

TÜV NORD GROUP

UNIKLINIK
RWTHAACHEN

UNiSCON
Sealed Cloud Technologies

UNITY
CONSULTING & INNOVATION

universität
Witten/Herdecke

UNIVERSITY OF AMSTERDAM

UNIVERSITEIT
TWENTE.

vastuu group

VDMA

vicomtech
MEMBER OF BASQUE RESEARCH & TECHNOLOGY ALLIANCE

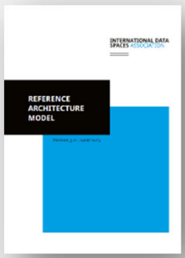
VOLKSWAGEN
AKTIENGESELLSCHAFT

VTT

WeTech
蜂巢思维

ZVEI:
Die Elektroindustrie

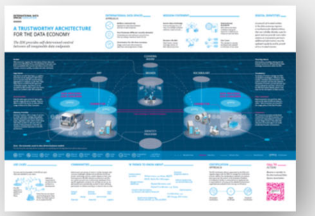
OVERVIEW PUBLICATIONS



Reference Architecture Model

Executive Summary

Image Brochure

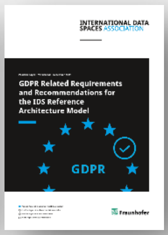


Infographic

Use Case Brochures

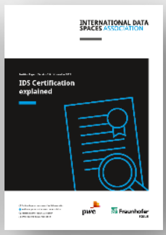
Study on Data Exchange

Position Paper Implementing the European Data Strategy



Position Paper GDPR Require-ments and Re-commendations

Position Paper Usage Control in the IDS



Position Paper IDS Certification Explained

White Paper Certification

Sharing data while keeping data ownership

Magazine Data Spaces_Now!

For these and further downloads: www.internationaldataspaces.org/info-package

Code available at: https://github.com/industrial-data-space

CONTACT

Head Office

INTERNATIONAL DATA SPACES ASSOCIATION

Emil-Figge-Str. 80
44227 Dortmund | Germany

phone: +49 231 70096 501
mail: info@internationaldataspaces.org

WWW.INTERNATIONALDATASPACES.ORG

@ids_association

international-data-spaces-association