D3.8 'Data Governance, TRUSTS Knowledge Graph II'

Authors: Victor Mireles (**SWC)**

Additional Information: Report, Public

June 2022

This project has received funding from the European Union's Horizon 2020
research and innovation programme under grant agreement No 871481

TRUSTS Trusted Secure Data Sharing Space

D3.8 ‘Data Governance, TRUSTS Knowledge Graph II’

Document Summary Information

Grant Agreement No	871481	**Acronym**	TRUSTS
Full Title	TRUSTS Trusted Secure Data Sharing Space		
Start Date	01/01/2020	**Duration**	36 months
Project website	https://trusts-data.eu/		
Deliverable	D3.8 ‘Data Governance, TRUSTS Knowledge Graph II’		
Work Package	WP3 ‘TRUSTS platform Implementation’		
Contractual due date	30/06/2022	**Actual submission date**	30/06/2022
Nature	Report	**Dissemination Level**	Public
Lead Beneficiary	Semantic Web Company (SWC)		
Responsible Author	Victor Mireles (SWC)		
Contributions from	Stefan Gindl, Michael Boch (Research Studios Austria), Sotiris Karampatakis, Martin Kaltenböck, Robert David (Semantic Web Company), Nikos Fourlataras (Relational), Diether Thelier (KNOW Center), Gianna Avgousti (EBOS)		

Revision history (including peer reviewing & quality control)

Version	Issue Date	% Complete	Changes	Contributor(s)
v0.1	02.05.2022	1%	Initial Deliverable Structure	Victor Mireles (SWC)
v0.2	14.06.2022	90%	Deliverable ready for review	Stefan Gindl (RSA) Sotiris Karampatakis, Victor Mireles, Robert David (SWC) Nikos Fourlataras (REL) Diether Thelier (KNOW)
v0.3	28.06.2022	95%	Addressing comments from Peer Review	Victor Mireles (SWC)
v1.0	29.06.2022	100%	Final submission version	Martin Kaltenböck (SWC)

Disclaimer

Copyright message

Table of Contents

List of Figures

List of Tables

Glossary of terms and abbreviations used

Abbreviation / Term	Description
DSC	Dataspace Connector
EOSC	European Open Science Cloud
OWL	Web Ontology Language
RAM	Reference Architecture Model
RoD	Registry of Datamarkets
RDF	Resource Description Framework
KG	Knowledge Graph
IDS	International Data Spaces
IM	Information Model
DMA	Data Market Austria
TRUSTS	Trusted Secure Data Sharing Space
GA	Grant Agreement
GDPR	General Data Protection Regulation
CKAN	Comprehensive Knowledge Archive Network data portal

1 Executive Summary

The Trusted Secure Data Sharing Space (TRUSTS) project aims to develop a platform for trading data and data services in a trustworthy and reliable manner, which will enable a data economy in which privacy and security are at the forefront. This platform will consist of a set of nodes, each operated by a different organization which will remain in full control of their data assets. The nodes, in turn, will be executing a set of common components which will allow for transactions that will strictly adhere to any contractual agreements between trading parties, will be fully auditable, and will enable a set of innovative privacy-preserving and data-ownership respecting business models. It will result in enhancements to the European data economy, by enabling new types of transactions to take place, and open the door for restricted and private data to be monetized with strict adherence to GDPR and other relevant regulations.

The architectural design of the TRUSTS platform requires the orchestration of different components, which in turn necessitates the exchange of information in an unambiguous and consistent manner. This is especially important since the TRUSTS platform has as its objectives the interoperability of different existing data infrastructures, some of which are operated by the project partners or their customers, and some of which are operated by third parties. In particular, the metadata about assets, computing resources, participants and policies has to be exchanged for the platform to satisfy its functional requirements. The collection of this metadata will be termed the TRUSTS Knowledge Graph (KG), and the specific organization of said graph is termed the TRUSTS Information Model (IM).

This document is the successor of Deliverable 3.7 and builds upon it. Here, we present the concrete implementation of the TRUSTS KGh. In particular, the different sources of metadata, and the different ways in which it is leveraged are explained. A special emphasis is placed on the different components that have been developed to satisfy the metadata flows: two extensions for CKAN and a platform client, as well as the connections to other pre-existing components such as the IDS Dataspace Connector or the Vocabulary Management Service inherited from the DMA.

This deliverable also contains a detailed description of the different vocabularies that were selected or developed for the project, in particular to satisfy the needs of interoperability with existing third-party initiatives. These vocabularies, which are still under active development, along with the adaptations done to the IDS IM, will constitute one of the legacies of the TRUSTS project.

It is envisioned that the public nature of this deliverable, along with the proper dissemination activities, will help communities and projects facing similar challenges to reuse the solutions proposed so far.

2 Introduction

The TRUSTS Knowledge Graph (KG) is the name given to the collection of metadata about different resources in the platform, and to the technical and organizational mechanisms for its maintenance, consumption and governance. This metadata is expressed as a collection of statements about entities such as assets, participants, nodes, and themes. These statements, when read by components of the platform, empower the different functionalities specified in the functional requirements to the platform.

In Deliverable 3.7 the general organization of the KG was outlined. The base Ontology (IDS-IM) was presented, along with the modifications that it has undergone during the TRUSTS project. Likewise, the properties of entities that take values from controlled vocabularies were listed. This organization is motivated by the functional requirements identified in the different documents produced by the project’s Work Package (WP) 2. In particular, the need for interoperability with external marketplaces, the possibility of having a variety of business models for asset providers, and the need to handle access control in accordance to machine-readable policies imposes several restrictions on the TRUSTS KG. Some of these restrictions manifest themselves as changes to ontologies or choice of controlled vocabularies. Others become patents in the software components in charge of manipulating, querying, and propagating metadata. Both of these have been maturing through the project, and Deliverable 3.8 is meant to report on them.

This document is an update on the design and use of this graph, with respect to Deliverable 3.7. Some of the points which were left outstanding (such as the concrete list of vocabularies to be used, or the mechanism for the platform-wide distribution of vocabularies) are resolved, and modifications to the interpretation of different statements, due to the technical evolution of the platform, are reported upon. In particular, the use of the Dataspace connector has introduced minor modifications to the interpretation of several statements. Furthermore, the inclusion in the platform of metadata coming from external data sources in order to test the interoperability of TRUSTS has also induced changes into the use and interpretation of metadata.

2.1 Definitions

TRUSTS platform

A set of interconnected nodes, and the components running within them, that support the functional requirements of the TRUSTS project.

TRUSTS resources

A set of entities whose description is relevant for the operation of the platform. In particular, this includes assets (datasets, applications, services), nodes, deployed components, organizations. The phrase “TRUSTS resource” refers to the actual, concrete entity.

Metadata

Any description about resources in the TRUSTS platform. This document is devoted to describing which of these descriptions are relevant and how they are organized and transmitted.

Metadata schema

A specification of how metadata for one or more classes of resources is to be recorded. It enumerates the list of metadata for a given resource and the type that said metadata should have (e.g. string, integer, controlled vocabulary).

Controlled vocabulary

An organized set of concepts with fixed identifiers, each of which can have one or more labels for human consumption. In this document, a controlled vocabulary is assumed to be conformant to the SKOS[1] specification.

Information Model

A specification of the different classes of TRUSTS resources that are to be considered, the metadata schemata that are to be adopted for each of them, and the relations that can hold among them. An information model specifies a set of valid resources, statements about said resources and an interpretation of said statements that can be operationalized. In this document, we consider an information model to be described using the Ontology Modeling Language OWL[2] alongside a natural language description that is sufficient for interpreting, constructing, and processing statements that conform to this specification.

Knowledge Graph

A graph that contains nodes corresponding to TRUSTS resources and that

- i) conforms to a given Information Model,
- ii) represents the state of a set of resources,
- iii) can be stored and queried according to well-defined methods, and
- iv) can be linked with other such graphs in order to enrich the meaning of the statements encoded in its edges.

IDS Connector

An IDS Connector is the core of the data space. It is the gateway to connect existing systems and their data to an IDS ecosystem. Its architecture and functionalities are defined by the IDS Reference Architecture Model (RAM) and specified by the certification criteria. The IDS Connector allows to exchange data and enrich it with metadata. An important aspect of this are usage conditions, which can be defined, administered, and implemented by the Connector. The metadata is described by the ontology of the IDS Information Model.

Dataspace Connector

[1] https://www.w3.org/TR/skos-reference/ Last accessed June 22, 2021
[2] https://www.w3.org/TR/2012/REC-owl2-syntax-20121211/ Last accessed June 22, 2021

The Dataspace Connector is an IDS connector that is being developed at Fraunhofer ISST. With the help of the Dataspace Connector, existing software can easily be extended by IDS Connector functionalities in order to integrate them into an IDS data ecosystem. Furthermore, it is possible to use the Dataspace Connector as a basis for the development of tailor-made software that is to be connected to an IDS data ecosystem.

TRUSTS Broker

An IDS Metadata Broker, which is a registry for IDS Connector self-description documents.

2.2 Mapping Projects' Outputs

Purpose of this section is to map TRUSTS Grant Agreement (GA) commitments, both within the formal Deliverable and Task description, against the project's respective outputs and work performed.

Table 1: Adherence to TRUSTS GA Deliverable & Tasks Descriptions

TRUSTS Task		Respective Document Chapter(s)	Justification
T3.4 *Data Governance : Metadata, Lineage and Semantic Layer*	*This task provides one of the backbones of TRUSTS to ensure a clear data governance model in the form of a TRUSTS Knowledge Graph that includes models (taxonomies, ontologies), metadata of all TRUSTS objects (data, services, tools, users, etc.), and lineage information (the information about provenance as well as the lifecycle of a dataset, service or software tool et al.) that can be used for interoperability (T3.3), Smart Contracts (T3.2), Search and Brokerage (T3.5 and 3.6) and above. This Knowledge Graph will be realised in the form of a*	*Whole Deliverable*	*This document is one of the main outputs of T3.4, as it is here where the metadata layer is formally specified.*

	semantic layer for TRUSTS that connects all objects in the system, and provides context and meaning for TRUSTS mechanisms and features.		
T3.3 *Data marketplaces interoperability solutions*	*Based on the findings of D2.1: Definition and analysis of the EU and worldwide data market trends and industrial needs for growth, and by analysing existing interfaces and standards, and even developing new relevant standards (see T7.4 Standardisation), the interoperability solution for TRUSTS will be designed in this task. This means the definition of interfaces to ensure interoperability with other industrial data marketplaces. In addition, interoperability solutions with the European Open Science Cloud (EOSC) will be evaluated and implemented where possible.*	*Chapter 3, Section 8. Chapter 5, Sections: 1 and 6*	*Semantic Interoperability is an important aspect of interoperability in general. This document contains the solutions, from the metadata point of view, that TRUSTS proposes for interoperability with external sources. The research into the requirements of interoperability undertaken in T3.3 have greatly informed this document.*
T3.5 *Platform Development & Integration*	*Based on the outcomes of T2.4, this task focuses on the implementation, testing and deployment of the TRUSTS platform components. Prior to release of D2.4A, this task is expected to collaborate with T2.1, 2.2 and 2.3 in order to prepare a smooth start of development in M6. The task makes use of infrastructure provided by T3.1. Assets from existing platforms (IDS, DMA) will be reused, enhanced and adapted to cover the specifications of T2.4. This*	*Chapter 3, Sections 1-7*	*Metadata exchange is necessary for the integration of the different components that constitute the platform. Furthermore, the TRUSTS-IM proposed in this document is greatly informed by the implementation activities taken so far in the project.*

	gives the task a head start by building on established and proven technologies. While, from an implementation point of view, this task covers general functionality (e.g., dataset and participant registrations), T3.2, 3.3 and 3.4 extend this functionality by providing specific state-of-the-art implementations that address the TRUSTS objectives. To that end		
TRUSTS Deliverable			
D3.8 ‘Data Governance, TRUSTS Knowledge Graph II’ *The deliverable contains the definition and specification of the Semantic layer, its utilized taxonomies, ontologies and metadata schemata. In addition, it elaborates on how the semantic layer supports the functionality of the TRUSTS platform. D3.4B is a revised and updated version of D3.4A, covering the final state of semantic technologies in TRUSTS (including all related software components).*			

2.3 Deliverable Overview and Report Structure

In the 12 months since the previous deliverable concerning the TRUSTS KG, several changes have been introduced to its definition and interpretation. This deliverable is organized in a way that these changes and updates can be understood, while at the same time providing a standalone description of the use of metadata in the TRUSTS platform.

Chapter 3 recapitulates the different uses of the TRUSTS KG in the platform, with an emphasis on a description of the software components that power said uses.

Chapter 4 goes into the details of the metadata lifecycle as of the current implementation of the platform. This implementation, because of the different components developed and re-used, requires a series of translations and transformations from one data model to another. This document serves as a documentation of said operations.

Chapter 5 touches briefly upon the organizational needs of the TRUSTS KG, in particular the governance of the TRUSTS-IM.

3 The use of the Knowledge Graph in the TRUSTS platform

There is a diverse set of entities involved in the TRUSTS platform. Users belong to organizations; each organization has a Node in the platform. Assets (Services, Datasets, and Applications) are exchanged among organizations using contracts, and the exchange results in new access possibilities to users. All of these entities, their attributes and the relations among them, are encoded in the TRUSTS KG: a flexible and extensible representation of metadata, which is used by a variety of services and components to satisfy the different functionalities of the platform. This KG can be thought of as a database, with the added benefit that it can be seamlessly combined with other sources of knowledge, such as other KGs, databases, or metadata catalogs, to increase the functionalities of the platform.

From a formal point of view, KG consists of two main components: an ontology and a series of statements respecting this ontology. The ontology is the abstract description of the domain (in this case, the TRUSTS platform and all of the entities involved), which serves both as a schema specifying the way data is represented, and as a data dictionary specifying the way data is to be interpreted and acted upon. Importantly, ontologies are machine readable and can thus be used to auto-configure services, user interfaces, and communication protocols. Ontologies define a set of classes of entities that are present in the domain, and a set of properties, or predicates, that can be used to state relationships between entities, as well as their properties. Statements contained in a KG encode knowledge about entities and their relationships in the form of subject-predicate-object triples, where subjects are entities, objects can be entities or literals (strings, dates, etc.), and predicates are specified in the ontology. They are called KGs because entities can be thought of as nodes in a graph, joined together according to edges, one per statement.

In TRUSTS, the KG is stored in a triple store, a special type of database that supports efficient and flexible querying of graph patterns. This allows us to ask, for example, questions like "what are all the resources that belong to a theme which is more specific than industry and that are published by an organization that is educational?". This triple store is wrapped by the component known as Broker which communicates with other components via messages as specified in the IDS Information Model. These messages constitute a standardized way to read and write statements in the KG, that is supported by other IDSA components, and form the basis of interoperability within TRUSTS.

The TRUSTSKG has a variety of uses, which will be described in the following.

3.1 Search

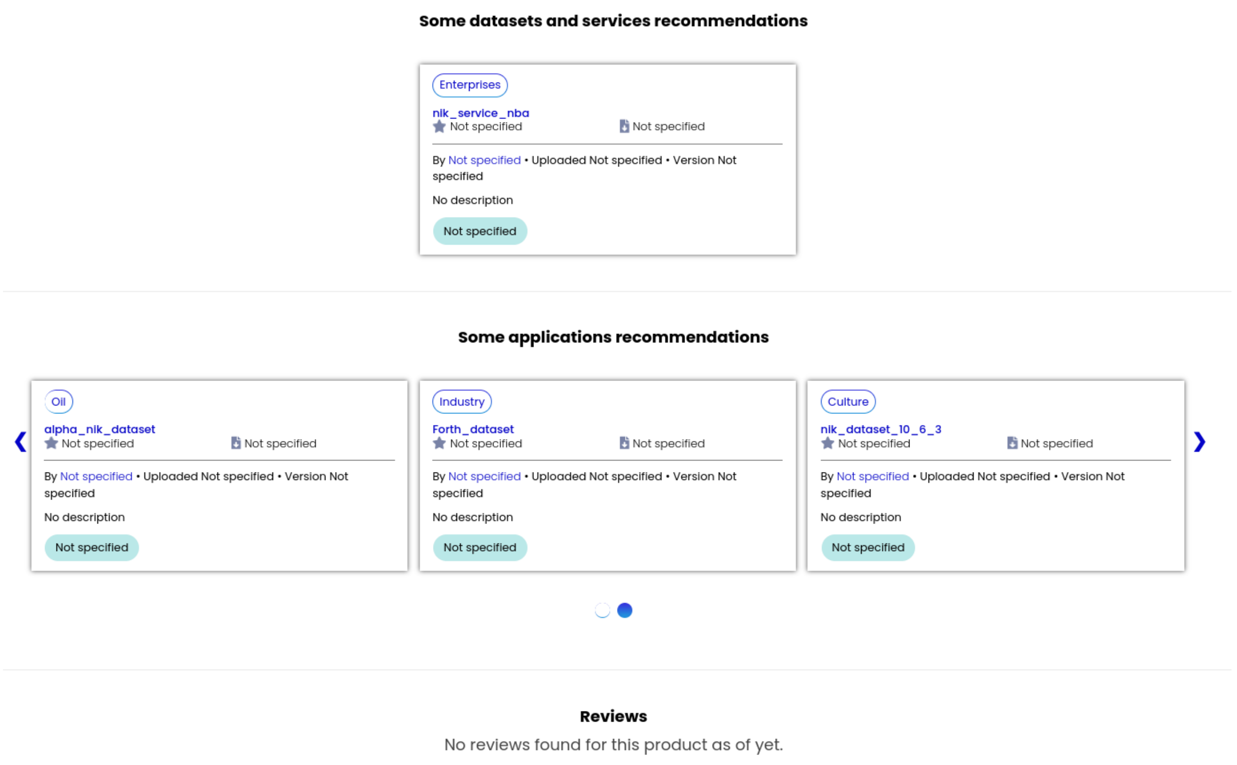
When a user wishes to acquire an asset through the TRUSTS platform, they must enter one of the user interfaces provided and browse the catalog of assets. This catalog is part of the KG and consists of a series of nodes, each representing an asset, with properties about them. Assets can have titles, descriptions, keywords and so on, as described in the TRUSTS-IM contained in Deliverable 3.7. These properties are used in a variety of ways during search.

Textual properties, such as titles, descriptions and so on can be used for full text search. In TRUSTS, the user interface is powered by CKAN, which already includes input text search. The data input by the user into these fields is then used in two parallel ways. On the one hand, it is forwarded to the local CKAN instance search mechanism which is based on the document indexer Apache Solr. On the other hand, it is forwarded to the local Dataspace Connector instance which, in turns, crafts an IDS message out of it and sends it to the broker instance. The query itself is then resolved by a triple store attached to the broker, which returns a set of resources in the form of subgraphs from the Knowledge Graph. Each of these subgraphs arrives at the local CKAN where it is converted into the data model of CKAN by the extension. The details of these mappings are provided in Chapter 5 of this document.

Properties which come from controlled vocabularies are used for the faceting of search results, that is, the selection of subsets of results according to whether or not they have a certain property. The faceting mechanism is implemented as SPARQL queries sent to the Broker. It is in this process of creation of SPARQL queries where the extra knowledge (the parts of the graph which are not part of the catalog) can be exploited. For example, for fields whose values come from a controlled vocabulary with a structure (taxonomy), the query can make use of RDF property paths to bring back results that contain either the values selected by the user, or values which are narrower according to the taxonomy.

3.2 Recommendation

The TRUSTS platform includes recommendation services that help users discover new and suitable assets. The Recommender Service queries the KG on a regular basis to receive updates on existing assets as well as new assets in the form of Resources specified by the IDS Information Model. Subsequently, a subset of available attributes found for assets is stored within the Recommender Service's data layer. To be able to differentiate between different types of assets, i.e., Dataset, Service, and Application, the Recommender uses the ***Asset_type*** attribute on Resources from the TRUSTS Ontology. Further, ***OfferId*** is used to receive the URI at which the respective asset can be contracted. In turn, the ***OfferId*** attribute is used by the Recommender Service to identify resources/assets. For time-based updates, the **Created** and **Modified** attributes of the IDS IMl are queried and stored. As the Recommender Service also incorporates content-based features to recommend assets, the Title, Description and Theme are fetched in addition.


Some datasets and services recommendations

Enterprises
nlk_service_nba
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Some applications recommendations

Oil
alpha_nlk_dataset
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Industry
Forth_dataset
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Culture
nlk_dataset_10_6_3
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Reviews
No reviews found for this product as of yet.

Figure 1: View of the recommendations on datasets

3.3 Contracting

As stated in the documentation of the IDS-IM[3], an offered resource is only complete if it contains at least one contract offer with at least one rule, along with other required objects like a well-defined representation and an artifact. An offered resource without a valid contract is available only to the user that created the resource. In order to be actually offered either directly from the provider’s Dataspace Connector, or through a Broker, a valid contract should be created and associated with the offered resource. Each asset can have multiple associated contracts offers within which a consumer node may choose to negotiate. A contract is always time constrained, meaning that it is valid only within a predefined period. Additionally, to this a contract might be constrained to a number of usage policies. The Dataspace Connector supports usage policies written in the IDS Usage Control Language based on ODRL. As it is referred on the IDSA position paper[4]:“*An IDS Contract is implicitly divided into 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*

[3] https://international-data-spaces-association.github.io/InformationModel/docs/index.html
[4]https://internationaldataspaces.org//wp-content/uploads/dlm_uploads/IDSA-Position-Paper-Usage-Control-in-the-IDS-V3..pdf

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.”

Table 2 presents the usage policies that are available on the Dataspace Connector. A provider may select a number of those when creating a new contract offer using the TRUSTS platform’s user interface.

Table 2: POLICIES. Usage policies available in the Dataspace Connector and their configuration

pattern name	policy	description	parameters
PROVIDE_ACCESS	Allow the Usage of the Data	This policy simply grants access to the resource.	-
PROHIBIT_ACCESS	-	This policy denies the access to the resource. A resource can’t be shared if it is annotated with this policy.	-
N_TIMES_USAGE	Restricted Number of Usages	This policy counts the access number of the resource and denies access if the access number is greater than the maximum number of accesses.	a maximum number of accesses
USAGE_DURING_INTERVAL	Interval-restricted Data Usage	Checks if the data access time is between the start and end time defined.	start and end time (of type *ZonedDateTime*, a representation of an instant in the universal timeline)
DURATION_USAGE	Duration-restricted Data Usage	This policy starts a duration for a resource: the resource can only be accessed in the	a duration, as specified by the Duration Java class[5]. (Example: “PT10H” stands for 10

[5] https://docs.oracle.com/javase/8/docs/api/java/time/Duration.html#abs--

		specified period. The duration starts counting from the artifacts' creation time. If the consumer tries to access the resource and the current time is over the allowed period access is denied.	hours)
USAGE_UNTIL_DELETION	Use Data and Delete it After	Similar to the Interval-restricted Data Usage policy, this one checks if the data access time is between the start and end time defined with an additional postDuty field with a DELETE action. Technically, this deletes the data after the interval has passed. The Usage Until Deletion policy should be used if it is desired that the resource be deleted after the time interval.	start and end time (of type *ZonedDateTime*, a representation of an instant in the universal timeline)
USAGE_LOGGING	Local Logging	This defines the policy to send usage logs to the clearing house. The clearing house has to be defined in the connectors' configuration. The logs are sent as IDS Messages to the clearing house[6].	clearing house url in connector configuration

[6]https://international-data-spaces-association.github.io/DataspaceConnector/CommunicationGuide/v6/IdsEcosystem/ClearingHouse#ids-clearing-house

USAGE_NOTIFICATION	Remote Notifications	This policy is similar to Usage Logging but is not restricted to sending messages to a clearing house. In the post duty field of the rule, an url can be defined within a constraint to which the DSC will send usage notifications to. The payload of the logs sent contain target, issuer connector and access time. If the message could not be sent, the access will still be granted.	URL to which usage notifications should be sent to (not limited to clearing house)
CONNECTOR_RESTRICTED_USAGE	Connector-restricted Data Usage	This policy checks if the issuer connector is equal to a specified connector.	allowed connector URI defined in a rule
SECURITY_PROFILE_RESTRICTED_USAGE	Security Level Restricted Policy	This policy checks if the connector has a specific security profile. This is verified by analyzing the DAT claims of the message received.	required connector security profile (BASE_SECURITY_PROFILE, TRUST_SECURITY_PROFILE and TRUST_PLUS_SECURITY_PROFILE)

3.4 Access to assets

Access to assets is also heavily dependent on the metadata stored about them. Since the TRUSTS platform is now based on a series of Dataspace Connector (DSC) instances, several changes to the access mechanisms have occurred with respect to what is described in Deliverable 3.7. In particular, since the DSC has built-in management of Apache Camel routes for a variety of scenarios, it is no longer necessary to manually create xml files defining routes using the metadata. The data model of the DSC allows for an asset to have an *access_url* attribute, which determines where to access an asset. In particular, when resources are hosted using the platform interfaces (CKAN), these urls point to the CKAN-provided API endpoints for accessing assets. In the case where resources are hosted by other components, as is the case of services or assets harvested from third party initiatives, this attribute points to the hosting component.

Setting the *access_url* attribute results in the automatic configuration of a camel-route in the background, and the creation of a DSC-served access endpoint which will forward requests to the specified *access_url*. This forwarding is done only when access to the resource is allowed according to the DSC's access control mechanism (rules specified via the IDS Usage Control Language). Furthermore, the automatically created access endpoint is only accepting IDS messages, which in the case of TRUSTS are emitted by another DSC, in the asset-consuming node. This connector, in turn, creates its own resource access URL which accepts HTTP requests from any client application (or a browser) and converts them into the appropriate IDS messages, after validating access control rules.

As of writing, the metadata used for this access control is solely the asset's *access_url* which includes its port. However, the TRUSTS-IM allows for further specification of access to the providing component, including authentication method and credentials. These can be programmatically provided to the DSC using its own REST API, in which case additional metadata properties will be exploited. Work is in progress to include these more complicated access mechanisms and exploit the full potential of the DSC. This would allow for complex back-ends serving the services offered to be placed outside the node’s internal network, for example. However, for the market-place use case, and using the architecture specified for the TRUSTS platform, these extra mechanisms are not essential, as it is assumed that the providing component is network-isolated from other nodes.

The steps for access control are outlined in the Figure2.

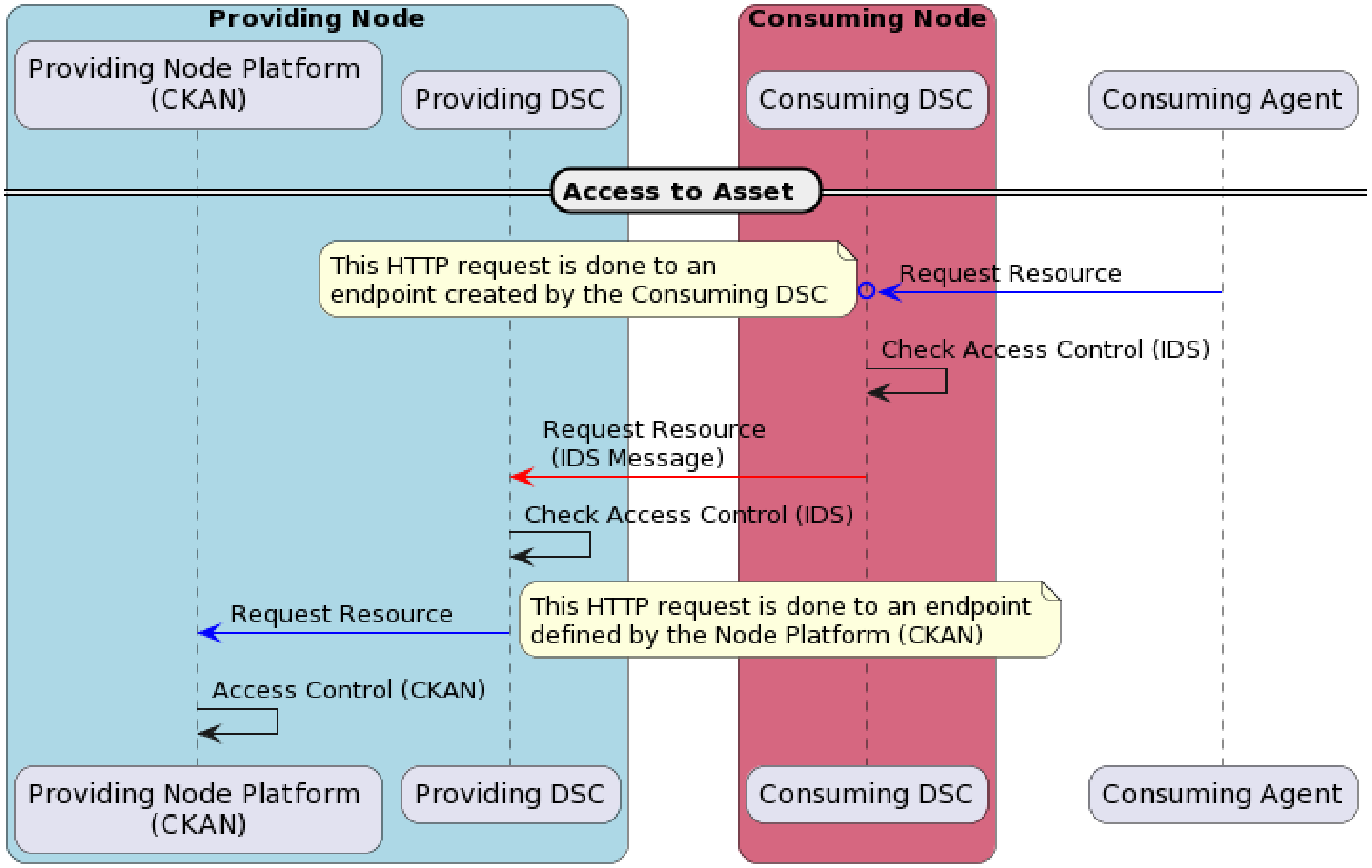

Providing Node
Consuming Node
Providing Node Platform (CKAN)
Providing DSC
Consuming DSC
Consuming Agent
Access to Asset
This HTTP request is done to an endpoint created by the Consuming DSC
Request Resource
Check Access Control (IDS)
Request Resource (IDS Message)
Check Access Control (IDS)
Request Resource
This HTTP request is done to an endpoint defined by the Node Platform (CKAN)
Access Control (CKAN)
Providing Node Platform (CKAN)
Providing DSC
Consuming DSC
Consuming Agent

Figure 2: ACCESS: Sequence Diagram of Access to Resources

3.5 Notification of asset changes

The TRUSTS notification services, based on the DSC subscription functionalities, allow for users to receive email notifications when assets they have acquired are subject to change. The configuration of subscriptions necessitates the use of metadata for pointing to users, assets and connectors. To activate the subscription to an artifact, the user should take the following steps:

a. The consumer should create a contract with the Provider for using the corresponding artifact.
b. The consumer should subscribe through the existing DSC functionality (IDS subscription) to remote Consumer artifact using an HTTP POST with a json payload like the following:

```
{
  "title": "IDS subscription",
  "description": "Notify on update",
  “target": "https://provider:8282/api/artifacts/artifactID",
  "location": "https://consumer:8282/api/ids/data",
  "subscriber": "https://consumer:8282",
  "pushData": false
}
```

where target is the URI of the corresponding artifact, location is the URL where those changes should be sent and subscriber is the endpoint that has subscribed to those changes.

c. The consumer should subscribe using Non IDS subscription to their local artifact using an HTTP POST request with a json as payload like the following:

```
{
  "title": "Non IDS subscription",contract
  "description": "emailto@dnsname.com",
  “target": "https://consumer:8282/api/artifacts/artifactID",
  "location": "http://note-service:5055/notify",
  "subscriber": "http://note-service:8282",
  "pushData": false
}
```

Where **description** is an email where to send notifications when the provider artifact changes, **target** is URI of the local artifact corresponding with remote artifact, **location** is the endpoint where the DSC will send POST HTTP requests to notify. These requests will carry three headers: **ids-event** containing the fixed text “update”, **ids-toemail,** containing the email address from the **description** fiend, and **ids-target** containing the URI from the **target** field.

d. The notification service developed int he project, called Note-service will then be responsible to send emails to inform user according to the “**ids-toemai**l” attribute.

4 Sources from which the Knowledge Graph is created

The statements which comprise the KG have a variety of origins, as illustrated in Figure 3. The provenance of these statements is not kept at the statement-level (as per limitations of the RDF model), but instead, some of the statements undergo a reification process which results in extra statements denoting their origin. In the following, we detail the different sources of these statements.

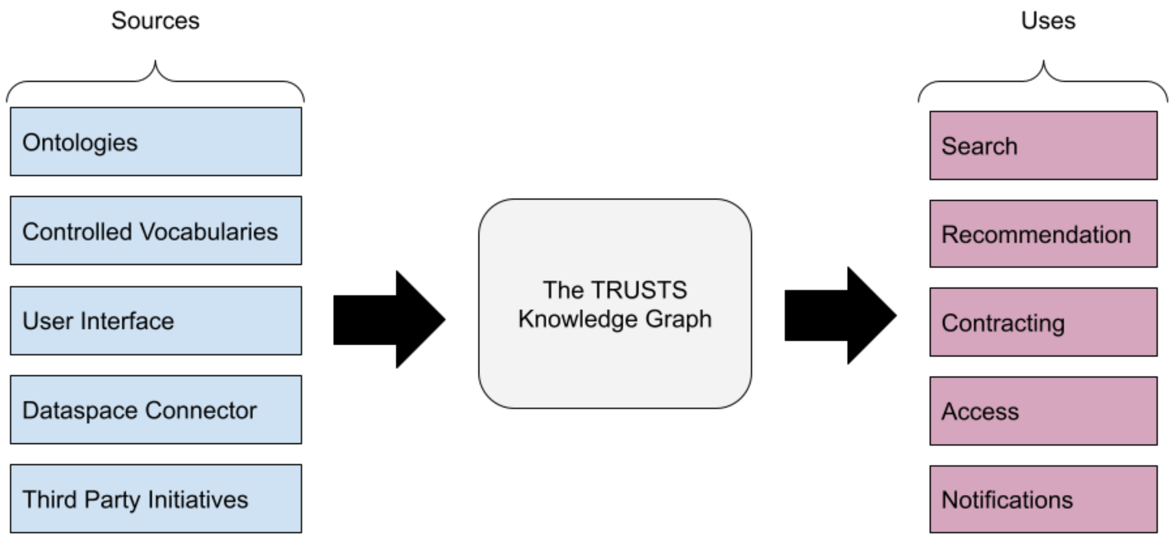


Figure 3: SOURCESKG: Origin and use of statements from the TRUSTS KG

4.1 Ontologies

The ontologies that comprise the TRUSTS-IM are detailed in Deliverable 3.7. The summarizing table from said deliverable is reproduced here and expanded with a description of their respective uses in the TRUSTS platform. These ontologies have been complemented, as described in said document and detailed below, with a series of predicates that enhance the compatibility of the IDS-IM with the metadata schema of the EOSC, and which further allow for better configuration of the access to assets as per the design of the TRUSTS platform architecture.

Table 3: Ontos: Ontologies that constitute the IDS-IM

Ontology	Number of Linked Classes	What is it used for in the TRUSTS platform
DCAT	11	Cataloging of assets
ODRL	26	Description of access policies
Data Cube	15	Description of structured datasets
ProvO	31	Description of Provenance of assets
vcard	63	Description of individual users
OWL-Time	20	Description of time points and intervals
Organization Ontology	10	Description of organization
Foaf	13	Description of individual users
TRUSTS	-	Extra properties to accommodate resource descriptions compatible with external data sources and access properties specific of TRUSTS

This document, and the TRUSTS project in general, applies the name TRUSTS-IM to the collection of these ontologies.

4.2 Controlled Vocabularies

Several of the data properties that are specified in the above ontologies have as a range concepts coming from a controlled vocabulary. The use of these concepts, which are usually but not always organized into a hierarchy, improves the usability of interfaces in two ways. First, it allows for a hierarchical organization of input and filtering components (e.g. pull down menus and checkboxes), which greatly reduces the strain on users. Second, it incorporates the knowledge encoded in the hierarchical structure in order to satisfy queries for search. Thus, a dataset tagged with, for example, the theme "Universities" is also returned for a search of the tag "Education". The list of vocabularies used in

TRUSTS is definitive, although the actual concepts belonging to the TRUSTS-developed vocabularies is still ongoing work.

Table 4: Properties of the TRUST-IM which use Controlled Vocabularies and their description.

property	vocabulary	description	statistics
dcat:theme	Theme	This is a custom developed vocabulary in the context of TRUSTS. It captures the domain of which a particular asset might be relevant to. In the current state, it is composed of 6 top level concepts, namely Agriculture, Culture, Education, Health, Industry, Science, and Services. It is offered as a multilingual resource, currently available in English, Spanish and Greek. It will be extended in the course of the project.	concepts: 33 languages: en, es, el broader/narrower relationships: 27
dct:format	File type	The File type authority table is a controlled vocabulary listing the different types of (digital) files (e.g. PDF, XML, DOC, ...). The File type authority table is under governance of the Interinstitutional Metadata Maintenance Committee (IMMC) and maintained by the Publications Office of the European Union on the EU Vocabularies website. In the context of TRUSTS, this vocabulary and its bound property are used to define the file type of a dataset.	concepts: 196 languages: en,et broader/narrower relationships: 0
rod:timeframe	Time Frames	This is a custom developed vocabulary in the context of TRUSTS. It captures the time frames in which a particular resource is updated. In the current state it is composed of 2 top-level concepts, namely the Fixed and the Continuous Update.	concepts: 9 languages: en, es, el broader/narrower relationships: 7
dcat:theme	EuroSciVoc	European Science Vocabulary (EuroSciVoc) is the taxonomy of fields	concepts: 1013 languages: de, en,

		of science based on OECD's 2015 Frascati Manual taxonomy. It was extended with fields of science categories extracted from CORDIS content through a semi-automatic process developed with Natural Language Processing (NLP) techniques. It is used in TRUSTS to finely categorize the theme in the case of science-related assets, particularly those being harvested from the EOSC.	es, fr, it, pl broader/narrower relationships: 1007
tdm:revenueModel	Revenue Model	This is a custom developed vocabulary in the context of TRUSTS. It captures the revenue models which a particular third-party initiative, such as a data market, might employ.	concepts: 17 languages: en broader/narrower relationships: 0
rod:dataAccessModel	Data Access	This is a custom developed vocabulary in the context of TRUSTS. It captures the data access method which might be used to access resources in third-party initiatives such as data markets.	concepts: 6 languages: en broader/narrower relationships: 0

In particular, the vocabularies used to describe third party initiatives are detailed below:

Revenue Model

Table 5: Revenue Model

Asset sale	Ownership rights are sold (Bergman, 2020).
Free	Free of charge (Muschalle et al., 2012; Becker et al. ,2014)
Flat rate	Customers pay a fixed amount irrespective of the use of the data/service for a limited time (Becker et al., 2014).
Transaction fee/ commission	Charge clients per transaction (Bergman, 2020; Spiekermann, 2019).
Usage fee	The customer pays for the use of an individual service or dataset (Bergman, 2020).

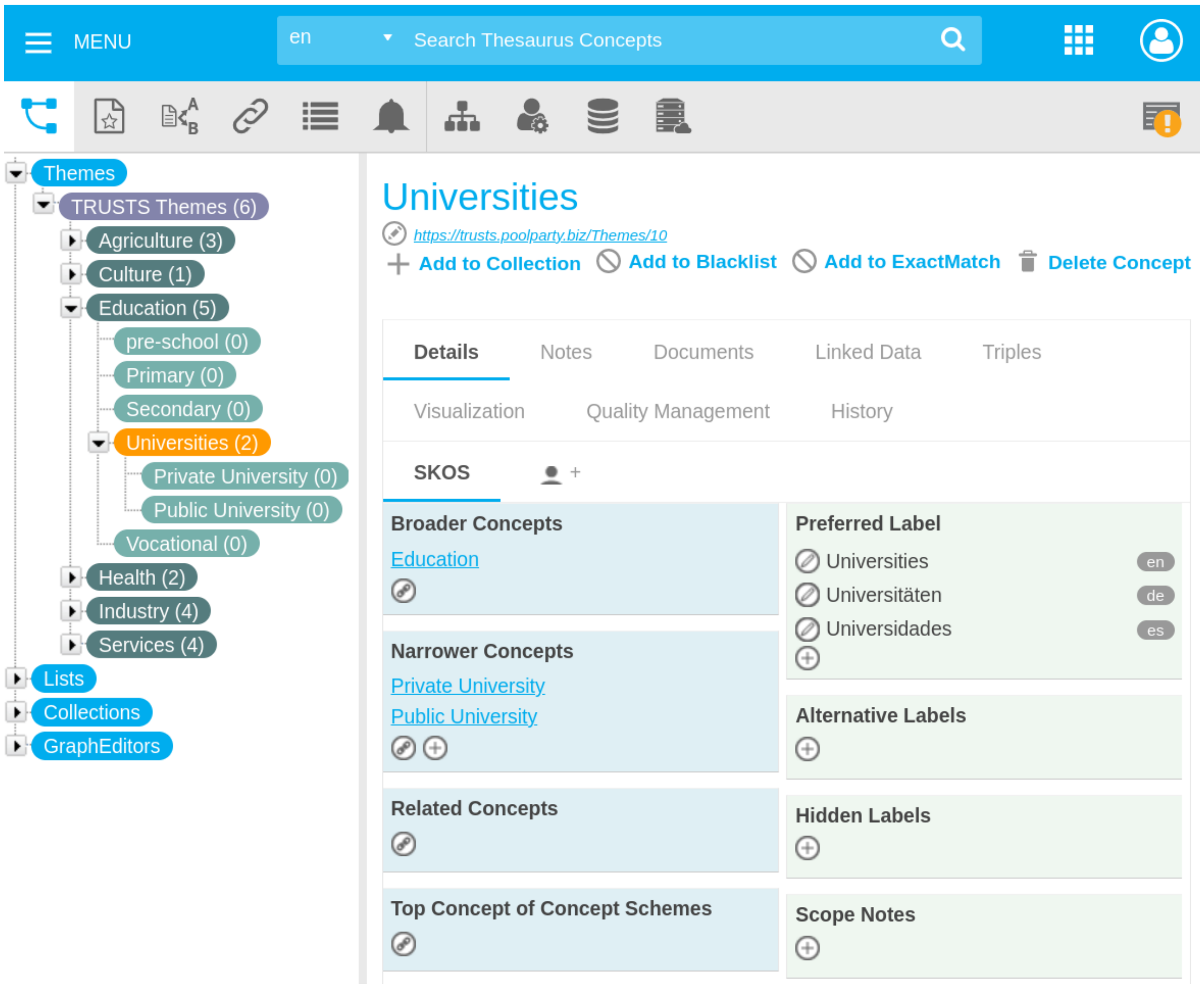
Freemium	Users can use the data and services free of charge, but for the full range of functions they must pay a fee (Stahl, Löser & Vossen, 2015).
Pay-per-use	This pricing model sets a price for each unit consumed and measures the final price based on the units consumed (Stahl, Löser & Vossen, 2015).
Subscription fees	Access to the data marketplace paid with periodic (e.g. monthly) fees (Bergman, 2020).
Licensing	Customers can buy licenses to use the data (Bergman, 2020).
Brokerage fee	Data marketplace acts as an intermediary between two or more parties and receives a share (Bergman, 2020).
Donation	Voluntary donation for the use of the data set (Bergman, 2020).
Gain sharing	Provider and owner share profits (Schüritz, Seebacher & Dorner, 2017).
Pay-with-data	A data trader provides data free of charge and in return receives the enriched data back (Stahl, Löser & Vossen, 2015).
Buy-and-sell data	The customer can sell and buy data on the platform (Schüritz, Seebacher & Dorner, 2017).
Progressive price	The price of datasets is based on the time of purchase. The price increases the more customers acquire a license to use the dataset (Stahl, Löser & Vossen, 2015).
Package pricing / Bundling	Data and services are packed together and sold together at a fixed price (Muschalle et al., 2012)
Other	Other options

Data Access

Table 6: Data Access

API	Use of an interface to gain access to the data/service (Stahl, Schomm & Vossen, 2014; Fricker & Maksimov, 2017; Fruhwirth, Rachinger & Prlja, 2020; van de Ven, 2020).
Download	Data access through file download (Stahl, Schomm & Vossen, 2014; Fricker & Maksimov, 2017; Fruhwirth, Rachinger & Prlja, 2020; van de Ven, 2020).
Specialized Software	Designated software of the data market is used to access data/service (Stahl, Schomm & Vossen, 2014; Fruhwirth, Rachinger & Prlja, 2020; van de Ven, 2020).
API and Download	Both options, API and download, are available (Fruhwirth, Rachinger & Prlja, 2020).
(Customised) web interface	Data is accessed via (customised) web interface (Stahl, Schomm & Vossen, 2014; Fricker & Maksimov, 2017).
Multiple Options	Combination of different options (Fricker & Maksimov, 2017; Fruhwirth, Rachinger & Prlja, 2020; van de Ven, 2020).

The vocabularies are maintained in an instance of PoolParty thesaurus manager. This provides a user-friendly user interface (Figure 4), a SPARQL endpoint (which is leveraged by the vocabularies extension developed for CKAN), as well as a REST API.



- Themes
  - TRUSTS Themes (6)
    - Agriculture (3)
    - Culture (1)
    - Education (5)
      - pre-school (0)
      - Primary (0)
      - Secondary (0)
      - Universities (2)
        - Private University (0)
        - Public University (0)
      - Vocational (0)
    - Health (2)
    - Industry (4)
    - Services (4)
- Lists
- Collections
- GraphEditors

Universities

https://trusts.poolparty.biz/Themes/10

Broader Concepts	Preferred Label
Education	Universities (en); Universitäten (de); Universidades (es)

Narrower Concepts	Alternative Labels
Private University; Public University	

Related Concepts	Hidden Labels

Top Concept of Concept Schemes	Scope Notes

Figure 4: PoolParty View of one of TRUSTS vocabularies

4.3 Metadata Ingested from the User Interface

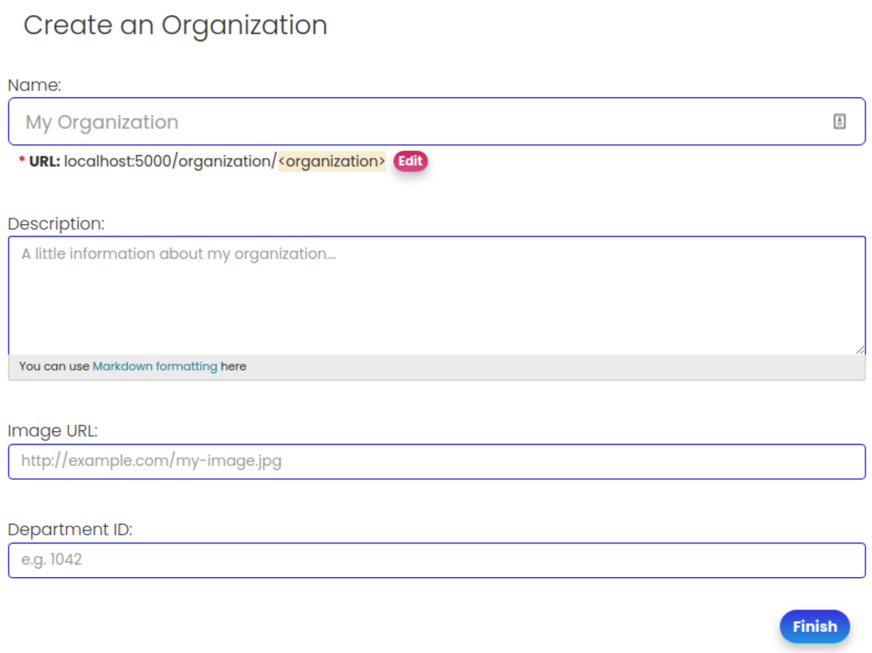
Providers and consumers can use the user interface (UI), implemented as a customised CKAN instance, in order to make available datasets, services, or applications on the TRUSTS platform. This functionality is provided by a CKAN extension, namely the ckanext-ids extension, which adds:

- Support for the mapping of the TRUSTS IM to the CKAN schema.
- Registration of an asset to a local Dataspace Connector
- Support to create a new Contract Offer and publish the offer on the global TRUSTS Broker

- Searching of assets through the common TRUSTS Broker
- UI and backend logic for the acceptance of a Contract Offer from a consumer node.
- UI and backend logic for the consumption of an asset
- Integration of the Recommender Service
- Tracking of important actions on the platform (creating a new dataset, a new contract offer etc.) through the activities stream that CKAN offers

Brief overview of the User Interface

In order to start creating datasets, services, or applications in the UI, a user needs to create an organization (see Figure 5). Organizations are the primary way to control who has access and permission to view, create, and update assets in CKAN. Each asset can belong to a single organization, and each organization controls access to its assets.


Create an Organization

Name: My Organization

* URL: localhost:5000/organization/<organization> Edit

Description: A little information about my organization...

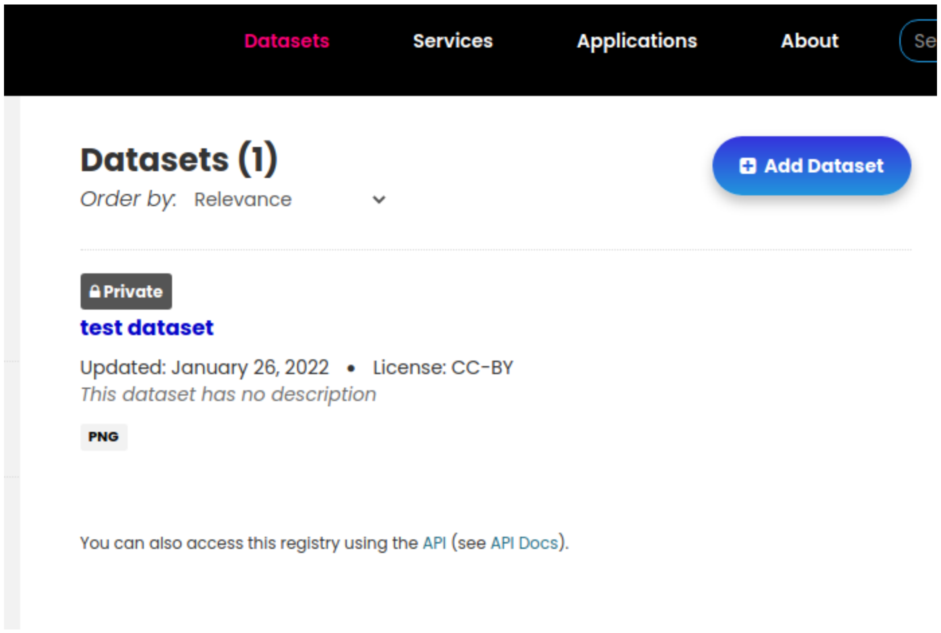
You can use Markdown formatting here

Image URL: http://example.com/my-image.jpg

Department ID: e.g. 1042

Figure 5: Creation of an organization

In order to create a dataset, service or application, a user can click on the respective element on the top navigation bar (see Figure 6).



Datasets (1)

Order by: Relevance

test dataset

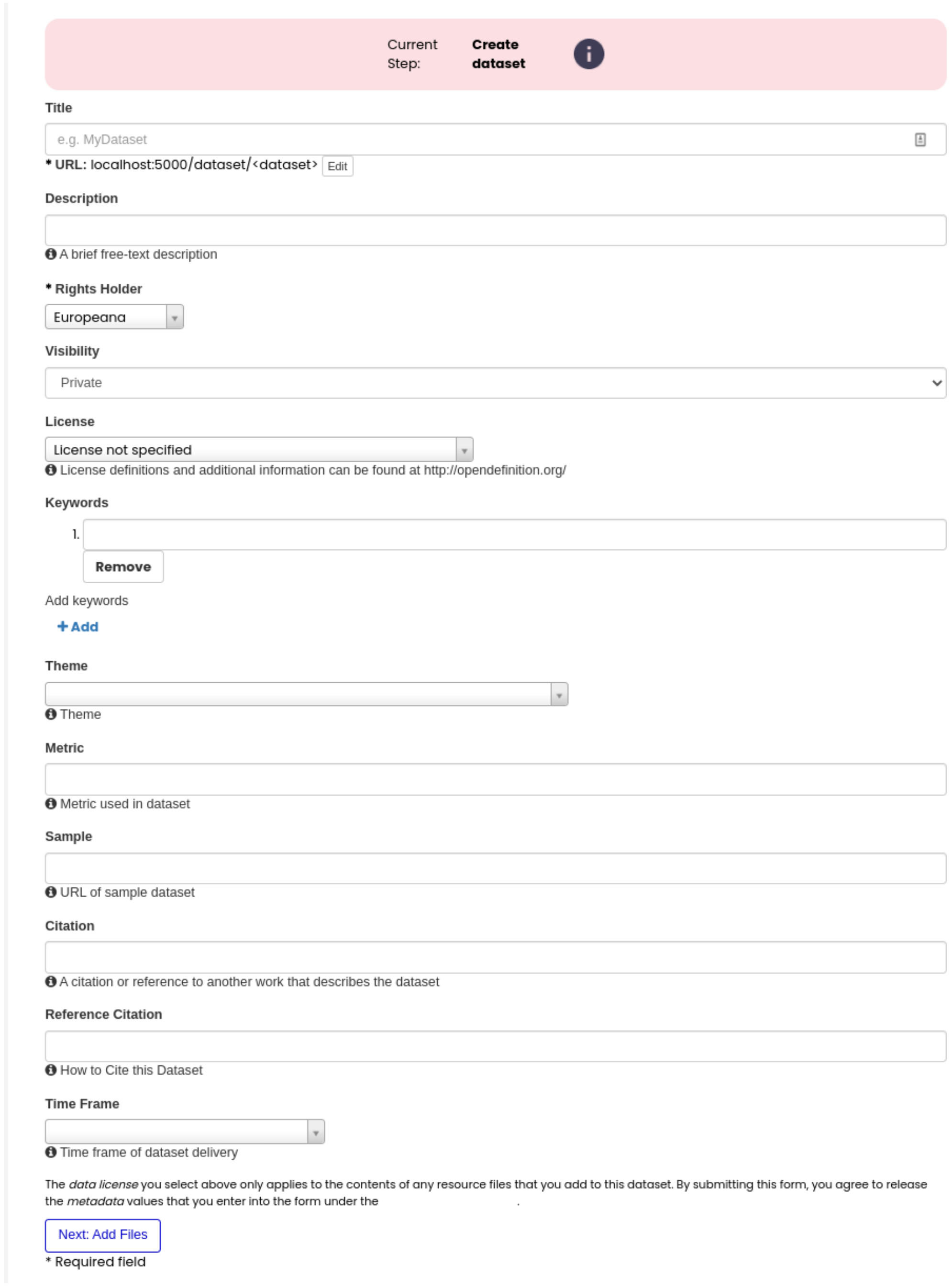
Updated: January 26, 2022 • License: CC-BY

This dataset has no description

You can also access this registry using the API (see API Docs).

Figure 6: The Datasets view

Then, clicking on the "Add Dataset" button will present the respective form to create an asset (see Figure 7). Through this view, a provider user can define the required metadata to register a new asset on the local Dataspace Connector. These metadata fields correspond to those defined in Deliverable 3.7, and their mapping to the UI is specified below in Table 7.


Current Step: **Create dataset**

Title

e.g. MyDataset

* **URL:** localhost:5000/dataset/<dataset> Edit

Description

A brief free-text description

* **Rights Holder**

Europeana

Visibility

Private

License

License not specified

License definitions and additional information can be found at http://opendefinition.org/

Keywords

1\.

Remove

Add keywords

+Add

Theme

Theme

Metric

Metric used in dataset

Sample

URL of sample dataset

Citation

A citation or reference to another work that describes the dataset

Reference Citation

How to Cite this Dataset

Time Frame

Time frame of dataset delivery

The *data license* you select above only applies to the contents of any resource files that you add to this dataset. By submitting this form, you agree to release the *metadata* values that you enter into the form under the .

Next: Add Files

* Required field

Figure 7: Creating a new dataset.

After filling in the required metadata fields for the asset, clicking the "Next: Add Files" will present the create resources view (See Figure 8). There the resources (artifacts) of the asset should be uploaded. A user can either upload a local file or provide a remote link where the resource exists. Depending on the type of the asset, additional files or metadata might be required.

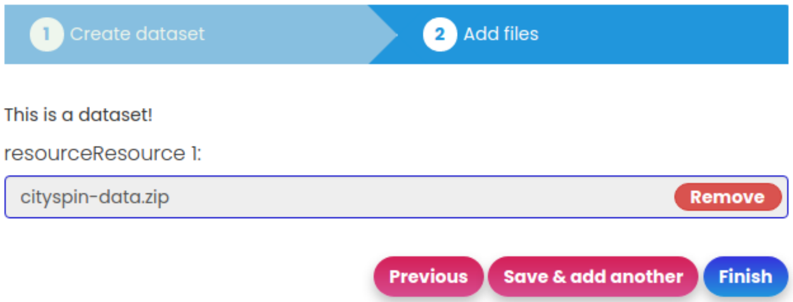


Figure 8: Uploading a file on CKAN

Clicking on the "Finish" button will create the asset as a CKAN asset. There are two additional steps needed in order to publish them as offers on the Dataspace Connector and the respective TRUSTS Broker.

First, the provider has to push the metadata of the asset to the Dataspace Connector. This is accomplished by clicking the "Push to Dataspace Connector" button on the asset view (see Figure 9). A notification is received to inform the provider about the status of the operation. In case of success, the provider now has to create a new Contract Offer.

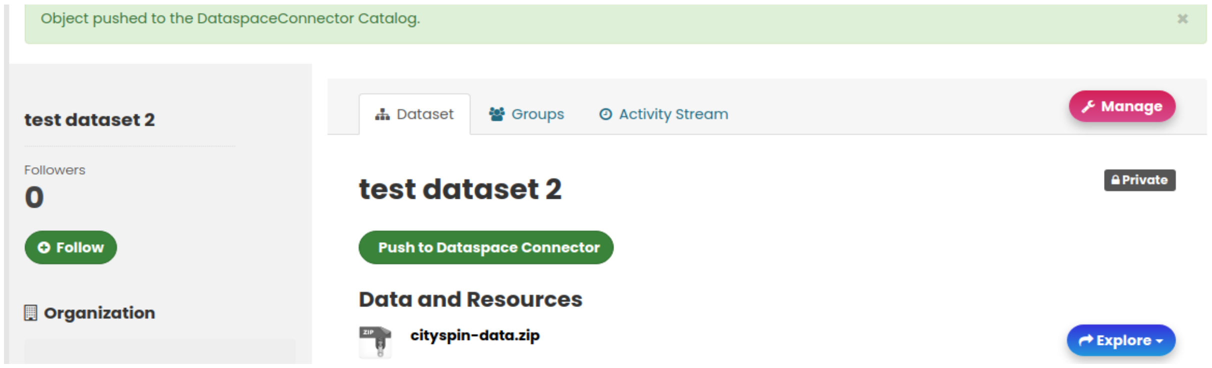
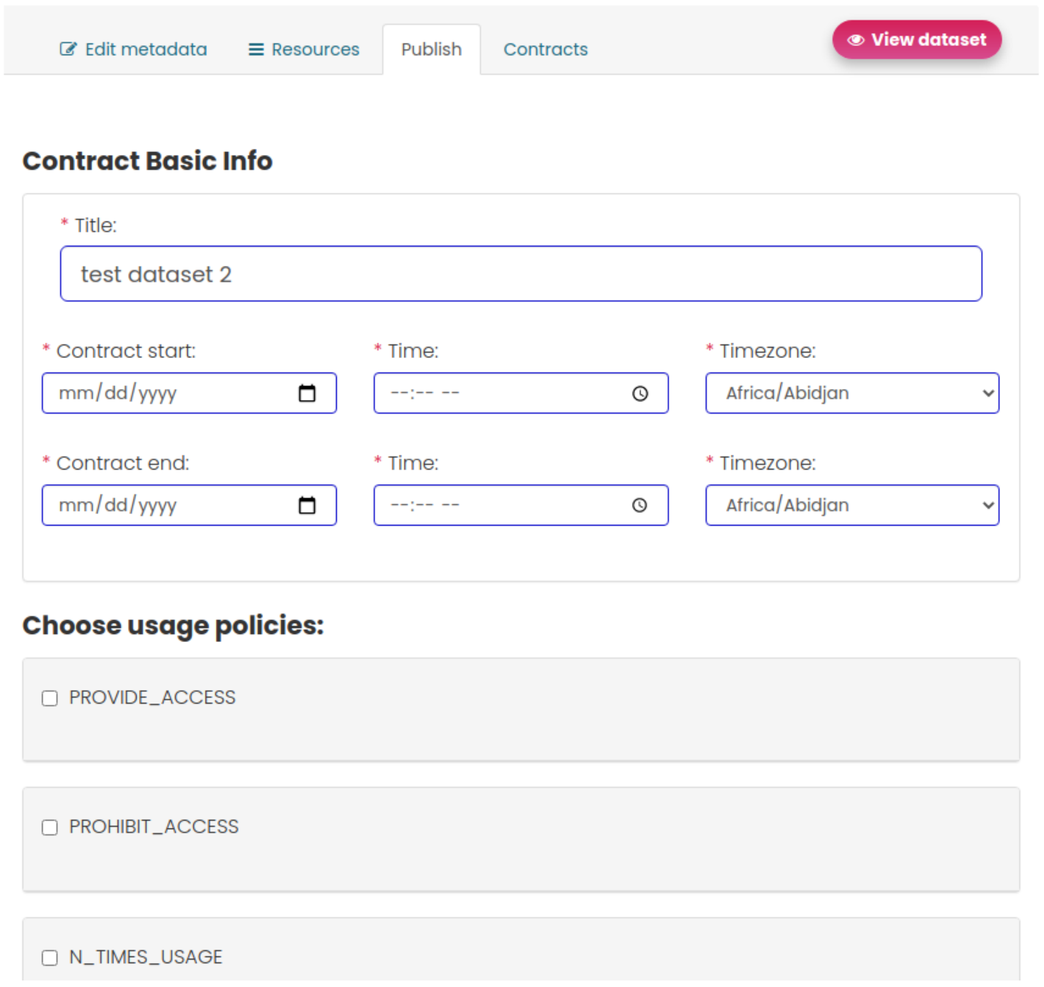

Object pushed to the DataspaceConnector Catalog.

Figure 9: Confirmation of registering an asset on the local Dataspace Connector

A provider can click on the "Manage" button. From this menu the provider can modify the metadata of a package, create or review available contracts and remove the asset completely. Clicking on the "Publish" tab will render the "Create Contract" view of the asset (see Figure 10).



Contract Basic Info

* Title: test dataset 2		
* Contract start: mm/dd/yyyy	* Time: --:-- --	* Timezone: Africa/Abidjan
* Contract end: mm/dd/yyyy	* Time: --:-- --	* Timezone: Africa/Abidjan

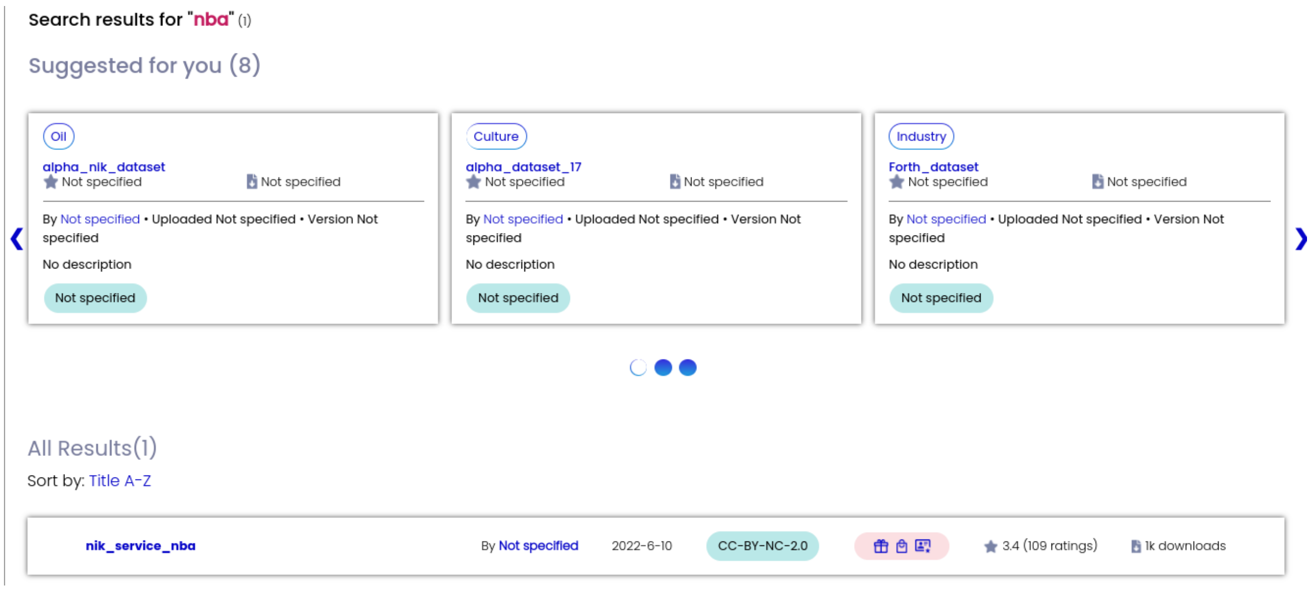
Choose usage policies:

- [ ] PROVIDE_ACCESS
- [ ] PROHIBIT_ACCESS
- [ ] N_TIMES_USAGE

Figure 10: The "Create Contract Offer" view of a resource

After filling in the required metadata and choosing a usage policy, the provider user can click on the "Publish" button, at the bottom of the page (not shown). This will create the Contract Offer on the local Dataspace Connector and register the asset and the offer on the TRUSTS Broker. The asset is now available for consumption by a consumer node.

On the search page, a user can browse and search through the available assets. Currently free text search is available on the titles of assets. Faceted search across all available metadata will be available on the final version of the extension.


Search results for "nba" (1)

Suggested for you (8)

Oil	Culture	Industry
alpha_nik_dataset	alpha_dataset_17	Forth_dataset
Not specified · Not specified	Not specified · Not specified	Not specified · Not specified
By Not specified • Uploaded Not specified • Version Not specified	By Not specified • Uploaded Not specified • Version Not specified	By Not specified • Uploaded Not specified • Version Not specified
No description	No description	No description
Not specified	Not specified	Not specified

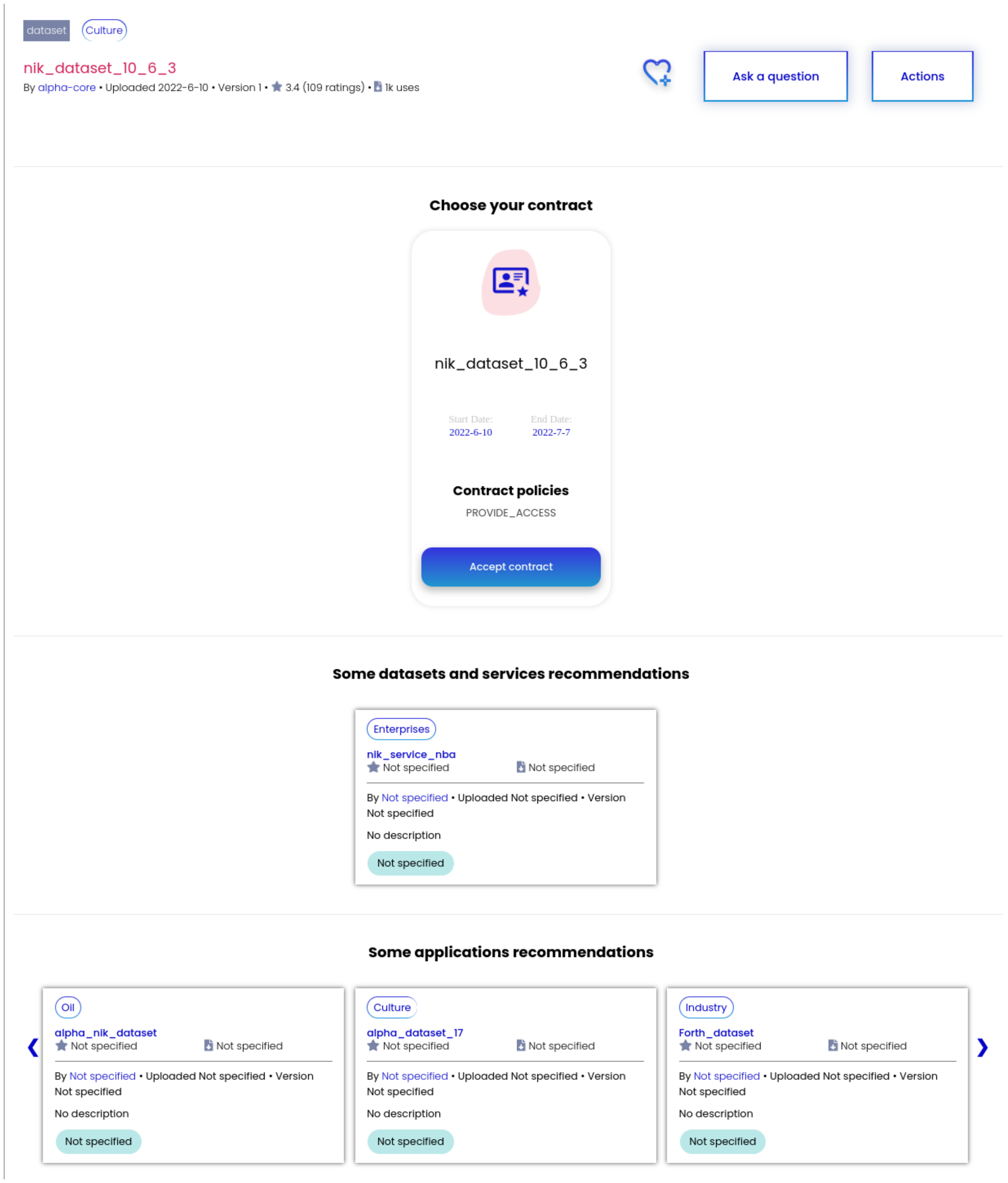
All Results(1)

Sort by: Title A-Z

nik_service_nba	By Not specified	2022-6-10	CC-BY-NC-2.0	3.4 (109 ratings)	1k downloads

Figure 11: The search results page

After clicking on an offered resource, the user can view its metadata (see Figure 12). Additionally, a list of the available contract offers is provided. A user can use the “Accept Contract” button in order to initiate the contract negotiation process. Contract negotiation is handled automatically by a series of actions on the backend. After the acceptance of a contract offer, the consumer node is able to consume the related artifacts.



nik_dataset_10_6_3

By alpha-core • Uploaded 2022-6-10 • Version 1 • 3.4 (109 ratings) • 1k uses

Choose your contract

nik_dataset_10_6_3

Start Date: 2022-6-10 End Date: 2022-7-7

Contract policies

PROVIDE_ACCESS

Some datasets and services recommendations

Enterprises
nik_service_nba
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Some applications recommendations

Oil
alpha_nik_dataset
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Culture
alpha_dataset_17
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Industry
Forth_dataset
Not specified Not specified
By Not specified • Uploaded Not specified • Version Not specified
No description
Not specified

Figure 12: View of an offered resource on the consumer node.

Required Fields

The provider should provide a minimum set of metadata fields in order to successfully create a new asset. These are the title, name, and owner organization fields. Through the interface, only the title is requested from the provider. The name field is automatically filled, using a series of transformation functions like lower casing, replacing spaces with dash etc, in order to conform with URL specification. Additionally, the owner organization field is automatically filled with the ID of the provider’s organization. However, when an asset is created through the API, those values should be defined as input data.

In order to enhance the discoverability of an asset, providers are suggested to provide additional optional metadata as described in Chapter 3. These metadata can then be used to allow faceted search of the assets. These metadata are also part of the knowledge graph, composing the description of an asset.

Use of vocabularies

As discussed thoroughly in D3.7, controlled vocabularies can be exploited in order to enhance the search capabilities of a platform, reducing human efforts in cataloging a resource on one hand and in search on the other hand. The TRUSTS-IM defines a set of properties that use controlled vocabularies as target values. Table 4 above shows the list of vocabularies used and the properties that link them to the different entities in the TRUSTS KG.

4.4 Metadata Ingested from third party initiatives

One of the objectives of the TRUSTS project is to create the technical basis for cooperation, both technical and commercial, between different data exchange initiatives, including other existing and future datamarkets, and the European Open Science Cloud (EOSC). Since it forms the basis of many functionalities, the TRUSTS KG should contain the metadata required to support this interoperability.

Third party initiatives, such as data markets, OpenAIRE or the different participants in the EOSC, have their own catalogs of resources. These will be integrated, in part or in whole, with the TRUSTS KG in order to make the assets hosted in these initiatives accessible to the participants in the TRUSTS platform. Therefore, it will be necessary to harvest this catalographic metadata, either using open protocols such as OAI-PMH, or using data dumps. Once harvested, this metadata will be mapped into the schema of the TRUSTS-IM, as detailed in Chapter 5.

The TRUSTS KG must also keep information regarding the source of these catalographic entries. In particular, technical details of the access method, the URLs and other coordinates are necessary for TRUSTS participants to access the data hosted in third party initiatives. Likewise, in order to support commercial interoperability with putative data markets, their revenue models should be recorded, in order to guide the types of contracts that will be offered for their assets on the TRUSTS platforms. The latter, along with general metadata about said initiatives is kept as part of the Registry of Data Markets, which is the subject of Deliverable 3.5 and 3.6.

5. The metadata pipelines in TRUSTS

The general flow of metadata can be visualized in Figure 13. In brief, metadata is transmitted between nodes using the IDS messaging protocol. The messages for this traffic are composed by the different Dataspace Connectors (DSC) installed in the nodes. Since the DSC has its own REST API corresponding to its own data model, it means that any input and output of metadata from the TRUSTS KG for the purposes of asset search and access (shown in Figure 2) must be done in accordance with the API defined by the DSC.

The architecture of the TRUSTS platform thus necessitates several steps of metadata mapping.

5.1 Mapping UI-generated metadata to the TRUSTS-IM

In Table 7 below, we present the different entities of the TRUSTS-IM, their properties, and their corresponding properties from the CKAN data model. This CKAN data model is specified using the popular ckanext-scheming extension[7], which provides a convenient, yaml-based, way to generate new database schemas for CKAN and their corresponding user-interface elements. CKAN organizes its data into three types of entities (relevant here): organizations, packages, and resources. One organization can publish many packages, and each package can contain one or more resources.

[7] https://github.com/ckan/ckanext-scheming

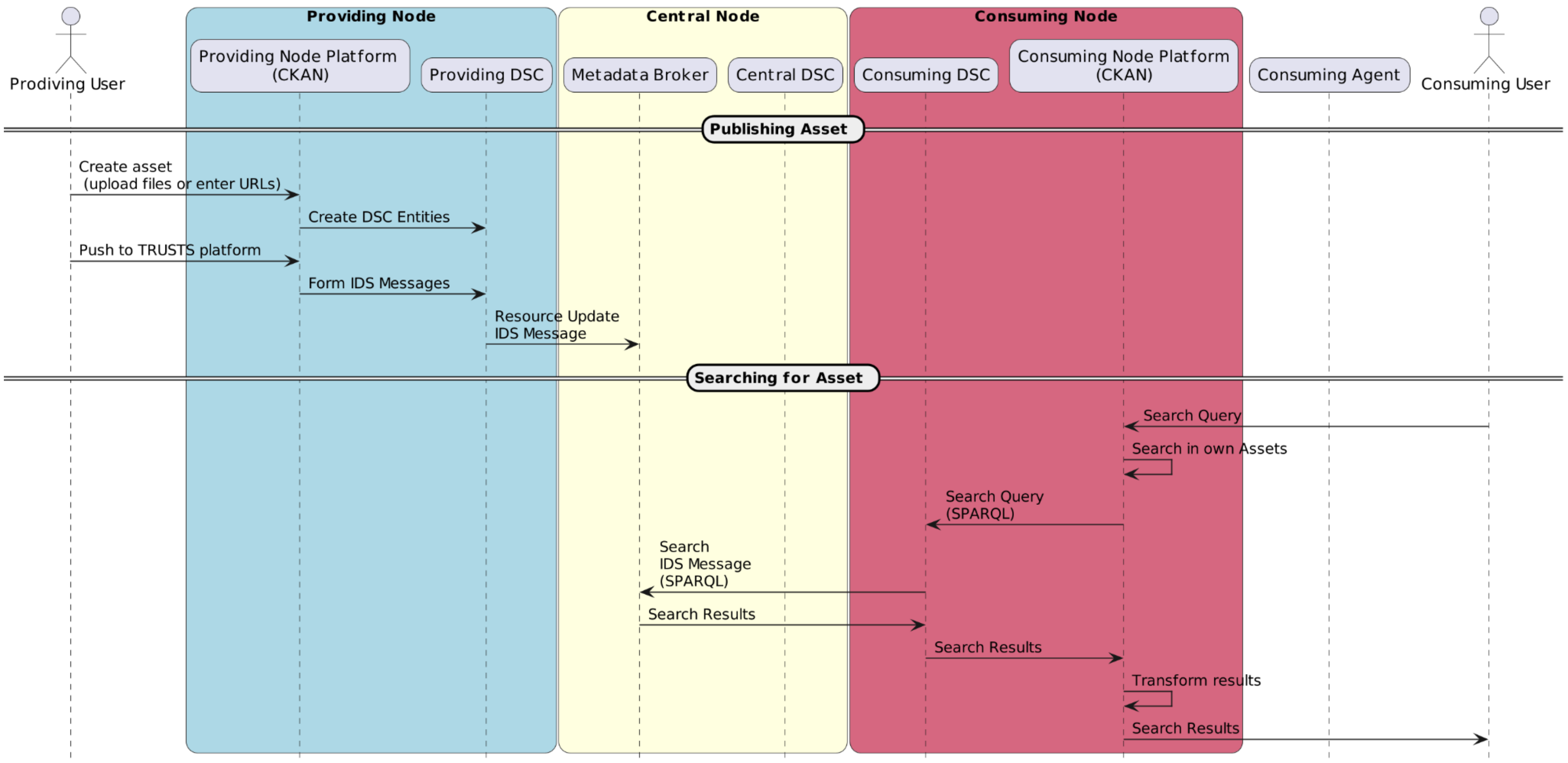


Figure 13: The flow of metadata in the TRUSTS platform

Table 7: Mapping of fields between the UI and the TRUSTS-IM

Trusts-IM Predicate	Location in CKAN	Field Name in CKAN	Type
ids:DigitalContent			
title	package	title	String
publisher	organization	name	String
creator	package	author	String
contact point	package	author_email	String
description	package	description	String
keyword	package	keywords	String, list
theme	package	theme	Controlled Vocabulary, list
metric	package	metric	String
sample	package	sample	String
citation	package	citation	String
referenceCitation	package	referenceCitation	String
timeFrame	package	TimeFrame	Controlled Vocabulary
dataAccess	resource	resource_type	Controlled Vocabulary
publication	package	Publication	String
scientificType	package	ScientificType	Controlled Vocabulary
ids:Representation			
license	package	license_id	Controlled Vocabulary
byteSize	resource	size	Integer
format	resource	mimetype	String

ids:Artifact			
accessURL	resource	url	String
title	resource	name	String
description	resource	description	String

5.2 The CKAN extension for vocabularies

In order to be able to use SKOS vocabularies as metadata fields values in an easy way, we developed a new CKAN extension, namely the ckanext-vocabularies[8]. It is based on the popular CKAN extension ckanext-scheming. The latter provides a way to configure and share metadata schemas using a YAML or JSON schema description. With ckanext-vocabularies, one can define a SKOS vocabulary as values for a metadata field through various methods:

- **Through a SPARQL Endpoint:** The vocabulary extension will query, upon startup, a fixed SPARQL endpoint and populate with it the options available in the corresponding input fields.
- **Through a local or remote file:** The vocabulary extension will access, upon startup, a file in any of the following RDF serializations: RDF/XML, TTL, N3. This file can be either in the same machine and container as the extensions (i.e. locally accessible to CKAN) or can be accessible via an HTTP GET request.
- **A resource provided as an IDSA resource:** At startup, the vocabulary extension will contact the local DSC and query it for the data of a specified dataset. This dataset is expected to contain RDF serialized as either **RDF**/XML, TTL or N3.

In all options, a SPARQL query in the context of the vocabulary triples is used to build the options list of the select element. In the case a SPARQL endpoint is defined as the source of a vocabulary, the results are fetched directly from the triple store each time a request to the asset creation page is made. In the case the vocabulary is provided as an IDSA resource, the vocabulary is retrieved once, when CKAN initiates, and stored locally. Then creates a subscription on the resource, through the local Dataspace Connector. Thus, any update on the vocabulary will automatically trigger an update on the locally stored vocabulary. The extension supports multilingual vocabularies, meaning that the values will be presented in the session defined language.

```
- field_name: theme # name of the field
  label: # labels of the field, how it is rendered on the asset forms, can support multilingual labels
    en: Theme
```

[8] https://pypi.org/project/ckanext-vocabularies/

```
  help_text: Theme # Help text to be presented next to the field
  help_inline: true
  preset: select # reusing the default select preset from ckan-scheming, leave as is
  form_snippet: autocomplete_dropdown.html # use of custom form snippet, leave as is
  choices_helper: skos_vocabulary_helper # definition of the helper to be used, leave as is
  skos_choices_sparql_endpoint: https://trusts.poolparty.biz/PoolParty/sparql/Themes # in case the
vocabulary is provided through a public SPARQL endpoint, you can define here the URL
  #skos_choices_concept_scheme: https://trusts.poolparty.biz/Themes/0 # Concept scheme filter, in
case there are more than one concept schemes available on the resource. The skos:inScheme property
should be available in all Concepts of the vocabulary.
  skos_choices_is_poolparty: true # PoolParty SPARQL Endpoint specific, leave as is if the vocabulary is
provided by a PoolParty SPARQL endpoint, otherwise switch to false
  display_property: dcat:theme # Mapped property, to the TRUSTS-IM in this case
```

Figure 14: Configuration of a ckanext-scheming field with the use of the ckanext-vocabularies plugin

5.3 Vocabulary Synchronization using the Dataspace Connector

Vocabularies (as described in Chapter 3) are stored in a centralized Vocabulary Management Service which, in the TRUSTS platform, is implemented as a PoolParty instance. These vocabularies are used, as described above, in the different metadata fields when assets are created. Since many of these assets will be created using the platform interface (CKAN), and since each node in the platform is running its own platform interface, it is necessary for all nodes in the platform to keep an up-to-date version of the vocabularies, so that the CKAN Extension for Vocabularies can properly display them.

In TRUSTS, this synchronization is done using the notion of Subscription as provided by the IDS. This means that a publisher-subscriber model can be used to communicate with the vocabulary provider (PoolParty) with all the platform nodes. This is schematized in Figure 15 where the general sequence is described. The summary of this synchronization mechanism is the following:

1. A vocabulary is created in the Vocabulary Management component (PoolParty).
2. A new vocabulary asset is created in the DSC which is present in the platform's central node (Central DSC), which contains a dump of the vocabulary in one of the different RDF serialization formats (e.g. Turtle).
3. A Camel route in the Central DSC is created. This route specifies that, upon reception of a PUT request, a new request should be done on the Vocabulary Service's API to obtain a new dump of the vocabulary, and then it should be uploaded as a new version of the asset created in point 2.
4. An IDS subscription is generated in each of the node's DSC to the Central DSC for the asset created in point 2. This is done, using the node's DSC API, by the CKAN vocabulary extension running in the node.

5. A non-IDS subscription is created in the node's DSC, specifying that every change of the state in the subscription of point 4 above, should trigger a POST call to the vocabulary extension which should then fetch the newest version of the asset through an IDS message.

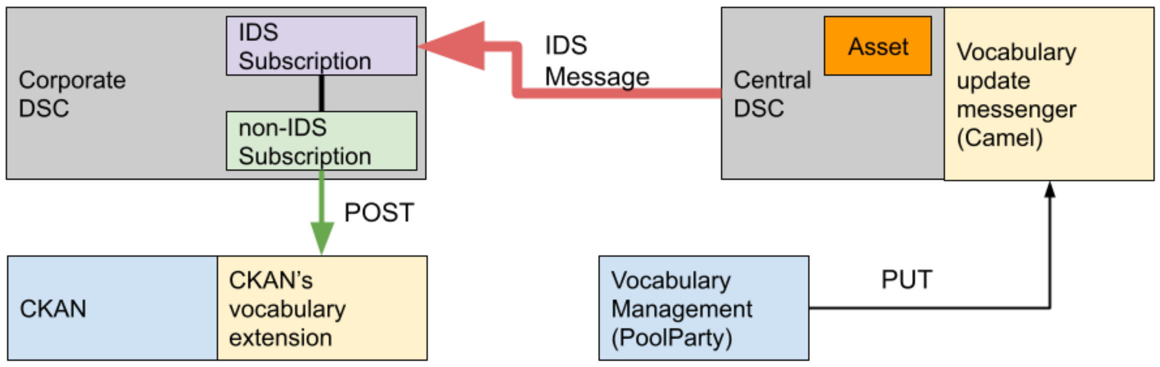

Corporate DSC
IDS Subscription
non-IDS Subscription
IDS Message
Central DSC
Asset
Vocabulary update messenger (Camel)
POST
CKAN
CKAN's vocabulary extension
Vocabulary Management (PoolParty)
PUT

Figure 15: Vocabulary Synchronization in the TRUSTS platform

5.4 Mapping of Metadata from third party initiatives

T3.3 "Data marketplaces interoperability solutions" involves the accessibility of metadata of datasets in external, third-party sources from within TRUSTS. TRUSTS will have interoperability functionality with external datamarkets on the one hand and the EOSC[9] on the other hand. In the following, we describe interoperability with EOSC. EOSC has a multitude of initiatives, which renders a "universal" interoperability with all of them infeasible. For this reason, we selected two representative initiatives and created a tool to transfer their metadata into TRUSTS. The two initiatives are OpenAIRE[10] and Europeana[11].

The incorporation of metadata of datasets from external sources requires, apart from the acquisition of the actual data, the mapping of the external metadata schema into the TRUSTS schema. The entire pipeline for transferring metadata into TRUSTS is based on the ETL (Extract, Transfer, Load) process frequently used in data warehouses. It encompasses the following steps:

- **Extract:** The first step is the extraction of the relevant metadata from the remote premises, i.e. an FTP server in the case of Europeana and a download page on Zotero for OpenAIRE.

[9] EOSC: https://www.eosc.eu/, accessed June 9, 2022.
[10] OpenAIRE: https://www.openaire.eu/, accessed June 9, 2022.
[11] Europeana: https://www.europeana.eu/en, accessed June 9, 2022.

- **Transform:** In the second step, the downloaded metadata gets converted into the TRUSTS metadata schema, using a mapping for both OpenAIRE and Europeana (see Table X).
- **Load:** The converted metadata gets loaded into the TRUSTS platform.

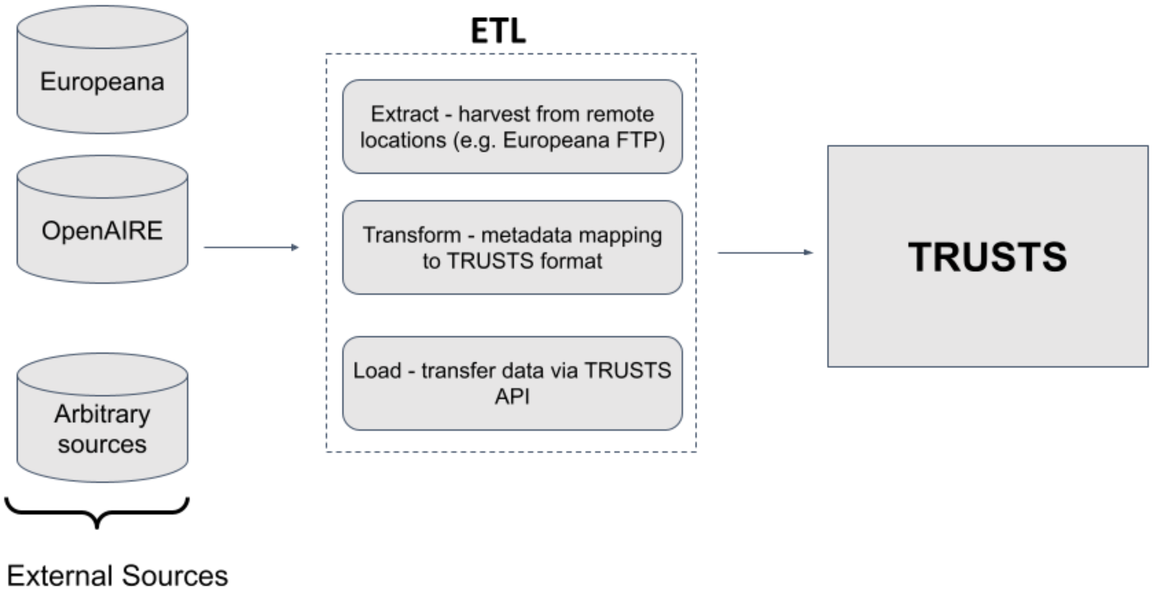

Europeana
OpenAIRE
Arbitrary sources
External Sources
ETL
Extract - harvest from remote locations (e.g. Europeana FTP)
Transform - metadata mapping to TRUSTS format
Load - transfer data via TRUSTS API
TRUSTS

Figure 16: ETL process to load metadata from external sources into TRUSTS.

The interoperability component developed in T3.3 accomplishes the first and second step, i.e. *Extract* and *Transform*. It therefore downloads the metadata from the sources and unpacks it in the *Extract* step. Subsequently, the component transforms the metadata from the external sources into the TRUSTS metadata schema. In the case of OpenAIRE the metadata comes in .json format, which makes the mapping operation a simple key replacement in a Python dictionary. The mapping operation from Europeana involves a preceding extraction step, since Europeana's metadata comes in .xml format. Consequently, relevant tags have to be extracted from the XML and converted into a dictionary first. We used the expression language XPath[12] to extract relevant parts from the XML and stored the resulting metadata in the TRUSTS format. The mapping between Europeana and TRUSTS as well as between OpenAIRE and TRUSTS was established in a previous manual step. Table 8 shows the mapping, i.e. the TRUSTS properties and their equivalents on both external sources.

A dedicated software component, the "trusts-platform-client" loaded the transformed metadata into the TRUSTS platform in the last step.

[12] XPath by W3C: https://www.w3.org/TR/1999/REC-xpath-19991116/, accessed June 9, 2022.

Table 8: The mapping of the metadata schemas of Europeana and OpenAIRE with the TRUSTS schema.

TRUSTS property	Europeana equivalent (XPath notation)	OpenAIRE equivalent
name	string(edm:EuropeanaAggregation/edm:datasetName)	maintitle
title	string(edm:EuropeanaAggregation/edm:datasetName)	maintitle
notes	string(ore:Proxy/dc:description)	description
owner_org	Pre-set: Europeana	publisher
resources::rights	string(ore:Aggregation/edm:rights)	bestaccessright::code
resources::url	string(edm:WebResource/@rdf:about)	url
resources::name	string(edm:EuropeanaAggregation/edm:datasetName)	maintitle
resources::dataProvider	string(ore:Aggregation/edm:rights)	publisher
resources::created	string(dqv:QualityAnnotation/dcterms:created)	publicationdate
resources::remoteId	string(ore:Proxy/dc:identifier)	id

The interoperability with external datamarkets is work under progress but will be inspired by the experience gained from EOSC interoperability. Interested datamarkets will be able to connect to a dedicated TRUSTS API, which they can use to upload their own data into TRUSTS. It is currently planned to have a special component, a metadata mapper, in the interoperability component, which facilitates metadata mapping and helps external datamarkets to align their schemas with the TRUSTS schema.

5.5 The TRUSTS platform client

The purpose of the TRUSTS platform client is to expose an interface to external parties that allows them (i) to access and load data from the platform on the one hand, and (ii) to transfer their own data into it on the other hand. For this purpose, the platform client communicates with the TRUSTS endpoint, i.e. REST API endpoints made available by CKAN[13], the backbone of TRUSTS. This results in CKAN transferring the metadata of external datasets into TRUSTS data storages, i.e. its database, and the search engine SolR[14]. Finally, the Client uses the API endpoints created by the IDS extension of CKAN to push the metadata to the Broker which, in turns, stores it in the triple store Fuseki[15]. Apart from these storages,

[13] CKAN: https://ckan.org/, accessed June 10, 2022.
[14] SolR: https://solr.apache.org/, accessed June 10, 2022.
[15] Fuseki: https://jena.apache.org/documentation/fuseki2/, accessed June 10, 2022.

the platform client also communicates with TRUSTS smart contract component. The information about newly integrated metadata gets communicated to this component and is stored in its blockchain (see Figure 17 for details on the platform client's architecture).

In its current state, the platform client has the following functionality:

- **Create datasets**
- **Create resources**
- **Push metadata to the Dataspace Connector**
- **Push metadata to the central node**

The platform client first creates an entry for the new metadata in TRUSTS. Subsequently, it adds resources to the metadata, which is an informational entry specifying details about the dataset, e.g. its web address, etc. After all required data for the new entry is created, the platform client first pushes the dataset to the Dataspace Connector and subsequently to the central node. After this step, the transfer of the dataset's metadata is completed.

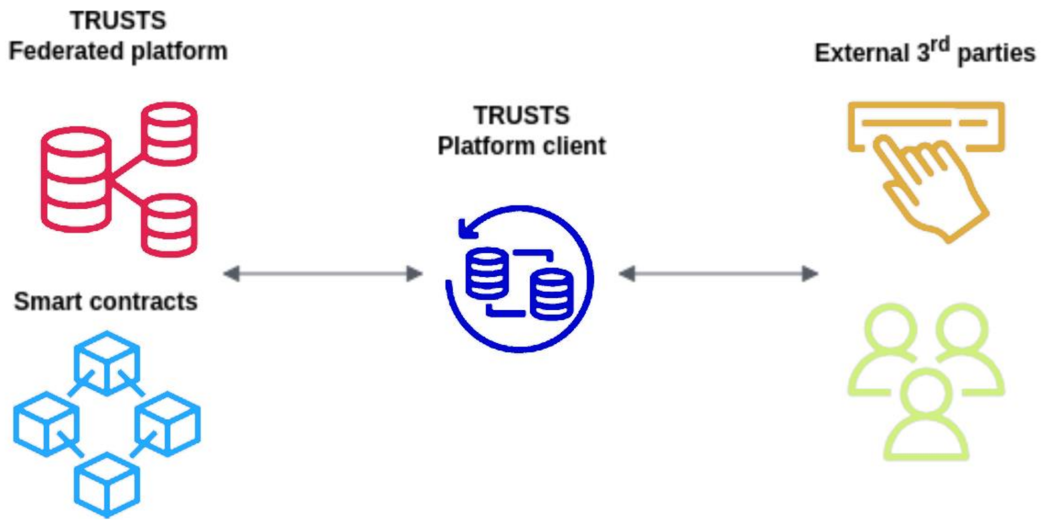


Figure 17: The architecture of the TRUSTS platform client.

6 Governance of the TRUSTS-IM

The vocabularies and ontologies that form the TRUSTS-IM, both those inherited from the IDS-IM and those developed as part of this project, are subject to change over time. Therefore, we need to account for (i) changes regarding updates in the model definitions themselves as well as (ii) updates in the parts of the system that apply them. For (i), we can make use of versioning methods which can represent different releases of vocabularies and ontologies. With versioning, we can also determine changes between two versions of the same vocabulary or ontology as the difference on an RDF level. For (ii), we need the system to manage the changes in the parts of the system that use these vocabularies and ontologies to describe metadata. This can be done by several approaches regarding the software architecture, from pushing changes when they happen, to robust components that can deal with inconsistencies regarding older versions. In general, a decentralized approach is preferable for better scalability, where the responsibility for updating the model version lies with the components themselves. We can take advantage for architectures like publish/subscribe, so that changes are pushed to a messaging system and the receiving components can join to be notified and also leave again without the need for the sender to know about. Such a decoupled architecture supports pushing changes to the system, while also providing scalability for the TRUSTS-IM governance.

7 Conclusions and next steps

This document has summarized the current state of the TRUSTS Knowledge Graph, the central repository of metadata that powers many of the platform’s functionalities. In particular, the metadata flows that feed this KG are described, as well as the necessary mappings between different schemas involved. The general architecture of the platform is very much related to these flows and mappings, both shaping them and being shaped by them.

In the current state of the platform, the TRUSTS-KG is integrated with other components for the purposes of asset onboarding, search, recommendation and contracting. Each of these integrations consists of a series of schema mappings. While the majority of these mappings are informed by the semantics defined the TRUSTS-IM, a fair amount of ad-hoc code is in place to perform them. Therefore, a common, platform wide, mapping mechanism is required to ensure the maintainability of the code, and to accommodate for subsequent versions of the TRUSTS-IM. This is a particularly challenging task since the different components which interact with the KG are written in different languages and maintained by different project partners. However, given the well-defined semantics and the collection of RDF files in which they are contained, it can be possible in the future to have the different components auto-configure themselves based on the ontology of the KG.

As was discussed in D3.7, detailed, column-specific access to data sets and services is still not accurately described using the ontology. While this is an open problem being tackled by different research groups, recent advances such as the Data Use Ontology [7] (for the biomedical domain) of the SemTab system or its derivatives [3], can be incorporated in the mid-term into the TRUSTS platform to allow for fine-grained description of assets, increased interoperability and function-aware recommendations.

In its current status, the TRUST-KG is a good basis for the operations of the platform and for its interoperability with external initiatives, from which several metadata properties were incorporated. Furthermore, the experiences in mapping schemas and developing adaptable UIs (in particular the two CKAN Extensions) can serve similar future endeavors by informing architectural decisions. Likewise, the open availability of the TRUSTS-IM and its constituent ontologies will support the operation of future dataspaces.

References

[1] Hogan, A., et al. (2020). Knowledge graphs. *arXiv preprint arXiv:2003.02320*.

[2] Bader, S., et. al (2020, November). The International Data Spaces Information Model–An Ontology for Sovereign Exchange of Digital Content. In *International Semantic Web Conference* (pp. 176-192). Springer, Cham.

[3] Chen, S., Karaoglu, A., Negreanu, C., Ma, T., Yao, J. G., Williams, J., ... & Lin, C. Y. (2022). LinkingPark: An automatic semantic table interpretation system. Journal of Web Semantics, 100733.

[4] Ben Ellefi, M., Bellahsene, Z., Breslin, J. G., Demidova, E., Dietze, S., Szymański, J., & Todorov, K. (2018). RDF dataset profiling–a survey of features, methods, vocabularies and applications. Semantic Web, 9(5), 677-705.

[5] Ivanschitz, B. P., Lampoltshammer, T. J., Mireles, V., Revenko, A., Schlarb, S., & Thurnay, L. (2018). A data market with decentralized repositories.

[6] Lagoze, C., & Van de Sompel, H. (2001, January). The Open Archives Initiative: Building a low-barrier interoperability framework. In *Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries* (pp. 54-62).

[7]Lawson, J., et al. (2021). The Data Use Ontology to streamline responsible access to human biomedical datasets. Cell genomics, 1(2), None. https://doi.org/10.1016/j.xgen.2021.100028

[8] Schütte, J., Brost, G. & Wessel, S. (2018). Datensouveränität im Internet der Dinge – Der Trusted Connector im Industrial Data Space. Whitepaper, Fraunhofer AISEC. (https://arxiv.org/pdf/1804.09442.pdf)

[9] van de Ven, M.R. (2020). Creating a Taxonomy of Business Models for Data Marketplaces. Master Thesis, TU Delft Technology, Policy and Management.

[10] Chekry, A., ORICHE, A., KHALDI, M., & ELKADIRI, K. (2011). Semantic web technologies for the reuse and adaptation of educational documents in e-learning. In *The 2nd International Conference on Multimedia Computing and Systems*.

[11] Hasnain, A., & Rebholz-Schuhmann, D. (2018, June). Assessing FAIR data principles against the 5-star open data principles. In European Semantic Web Conference (pp. 469-477). Springer, Cham.