This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 871754

Intelligent, Interoperable, Integrative and deployable open source MARKETplace

Shared Semantic Models specification V1

Authors

Name	Partner	e-mail
Achille Zappa	01 NUIG	achille.zappa@insight-centre.org

Key data

Keywords	Open platforms, semantic interoperability, Metadata Management, Semantic Engine, Semantic Model
Lead Editor	Name: Achille Zappa Partner: NUIG

Abstract

Due to its diverse set of stakeholders (and use cases), i3-Market project has to cover a wide range of marketplaces and domain models. The project has a certain number of goals:

- specify how basic data assets and datasets are represented via metadata descriptions.
- define how to describe concepts for data marketplaces and data spaces.
- identify data models for Automotive, Wellbeing and Industry applications, e.g. for smart transport systems, smart manufacturing.
- specify metadata for the i3-Market Backplane system and its components, e.g. to contracts, pricing models, operations information.

This modeling introduces the mechanisms for enabling the interoperability among data marketplaces and data spaces and backplane operations at the semantic level. A semantic model is introduced to facilitate the formal semantic metadata descriptions of concepts and properties used in the context of the i3-Market Backplane. A central role plays the concept of "Data Offering", which represents a set of data assets resources offered by a data marketplace or data space. The defined concepts and properties are utilized to uniquely identify and semantically annotate meaningful relationships and contexts of data assets as well as to specify semantically defined values for relevant operations. The interoperability of data represented in this fashion is instrumented by standardized machine-readable formats whereby consumers of these resources can autonomously interpret the meaning of the annotated data. The needed vocabulary management includes features such as the registration, retrieval, update, and removal of semantic descriptions, as well as the alignment between predefined semantic descriptions.

Table of contents

1 About This Document

This is the first version of the specifications related to the Task 4.2, Shared Semantic Data Model Repository specification V1 reported for deliverable D4.3.

This deliverable introduces the mechanisms for enabling the interoperability for smart object platforms and services at the semantic level. A semantic model is introduced to facilitate the formal semantic descriptions of concepts and properties used in the context of the i3-Market Backplane and API. A central role plays the concept of “Data Offering”, which represents a set of data assets resources offered by a data marketplace or data space. The defined concepts and properties are utilized to uniquely identify and semantically annotate meaningful relationships and contexts of data assets as well as to specify semantically defined values for relevant operations. The interoperability of data represented in this fashion is instrumented by standardized machine-readable formats whereby consumers of these resources can autonomously interpret the meaning of the annotated data. The needed vocabulary management, that is operated via the Semantic Engine components, includes features such as the registration, discovery, retrieval, update, and removal of semantic descriptions, as well as the alignment between predefined semantic descriptions.

To simplify both semantic model development and reuse, a modular design is beneficial. Based on the project specification and the requirements in Deliverable D2.3 and D2.6, the semantic model can be modularized according to their scope.

As we introduced the concept of "data offering" that allows to describe the capabilities and interfaces of an i3-Market Provider and how this information can be used for discovery and access purpose, the specifications for the semantic description of resources are going to be described. The specific description is called the Data Offering description and the Semantic Engine and Framework were created to manipulate and manage the metadata.

The first prototype of the Semantic Engine and Framework solution is available and integrated into the i3-Market backplane. Another concept is the Metadata semantic Registry stored in Triple Stores. With this feature, the backplane can rely on the metadata registry storage capacity to collect the semantic information that can be queried.

This deliverable addresses the following audiences:

- **Researchers, developers and integrators within the i3-Market consortium**, which will use this deliverable and the therein defined models as shared conceptualisation, i.e. shared vocabulary and taxonomy of the i3-Market domain.
- **Data Marketplace owners** who wish to join i3-Market will be able to use the data offering description to annotate their offerings. These descriptions should comply with the semantic model proposed within i3-Market.
- **Members of other Data space** communities and projects (such as projects of the BDVA cluster) can take this document as an initial reference or inspiration to design and implement their own marketplace interfaces to the backplane that also stores resources that are semantically annotated.
- **Open-source communities** will be able to have a better understanding of the technical details needed for them to join the i3-Market ecosystem. The semantic models and engine are also part of the assets of open-source solutions released by i3-Market.

- **Standardization bodies**. As a public document, this deliverable will be accessible by any groups listed above and including standardization bodies.

2 Main technical Motivations and Contributions

The i3-MARKET project addresses the growing demand for a single European Data Market Economy by innovating marketplace platforms, demonstrating with industrial implementations that the data economy growth is possible. The i3-MARKET consortium works towards providing technologies for trustworthy (secure and reliable), data-driven collaboration and federation of existing and new future marketplace platforms, with special attention on industrial data and particularly on sensitive commercial data assets from both SMEs to large industrial corporations.

The i3-MARKET Backplane innovates industry solutions by developing lacking building blocks to overcome the barriers on interoperable and integrative data marketplaces. Architecturally speaking, i3-MARKET uses trusted, federated and decentralised software components, enabling the integration of other marketplaces. The i3-MARKET architecture is designed to enable secure and privacy preserving data sharing across data spaces and marketplaces, through the deployment of a backplane across operational data marketplaces.

In i3-Market we are not trying to create another new Marketplace but we are implementing a backplane solutions that allow other Data Marketplace and Data Space to expand their market, facilitate the registration and discovery of data assets, facilitate the trading and sharing of data assets among providers, consumers and owners and provide tools to add functionalities they lack for a better data sharing and trading processes.

Taking into consideration the nature of the project we worked in task 4.2 on a) the definition, creation and collection of Semantic Data Models that allow to share a common description of the data assets (as per the case of common Data Offering Descriptions, Section 8 for more details), operations, services, data details, credentials, contracts, pricing, actors; b) development and implementation of semantic engine system and storage to manage the management of such information, creation of Data Offering Description, management of controlled registries, mapping of information, interfaces among components, links of data and actors, discovery and retrieve of necessary information, compiling of smart contracts and other operations; c) share the semantic models with the community to make use of those and work with people in improving and maintain the models for the present and future.

2.1 View on the status of Data Marketplaces and Semantic Models

We took a look at the state for understanding Data Marketplaces solutions.

We started with the consortium Partners solutions in SIEMENS, ATOS, IBM like Agora, Mindsphere and ACTIVAGE Suite and we looked at different reportings like:

Data-Driven Economy: Challenges and opportunities

https://www.intereconomics.eu/contents/year/2019/number/4/article/data-driven-economy-challenges-and-opportunities.html

Data Marketplaces: Trends and Monetisation of Data Goods

https://www.intereconomics.eu/contents/year/2019/number/4/article/data-marketplaces-trends-and-monetisation-of-data-goods.html

Trends and Exploitation of Data Goods

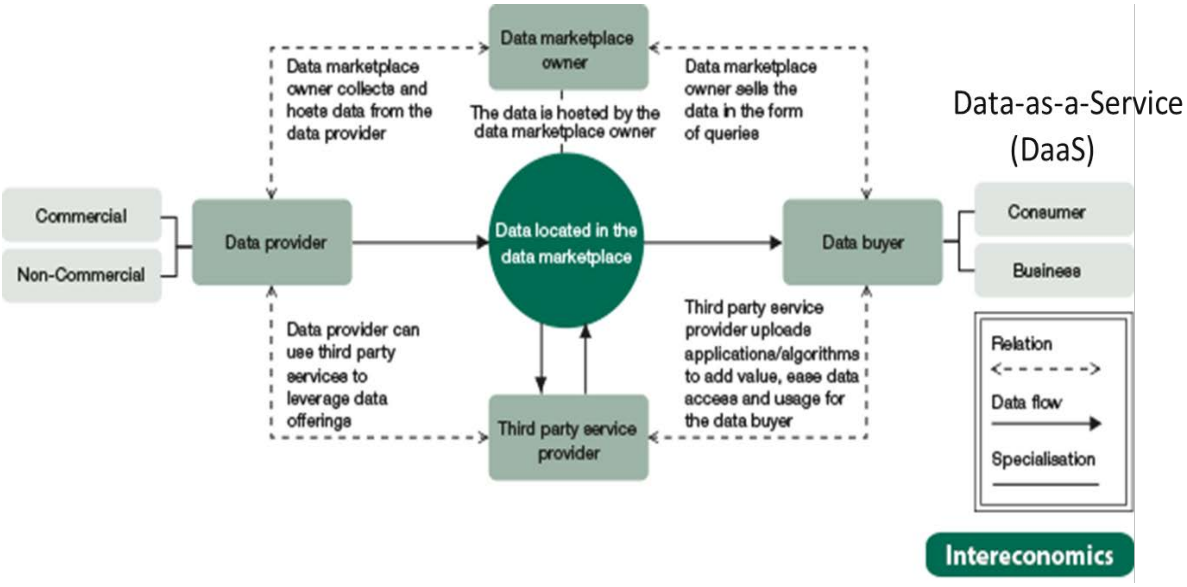


Figure 1: Trends and Exploitations of Data Goods

Having a general view on the State of Data Marketplaces:

Classification of Data Marketplaces

Data marketplace	Value proposition	Market positioning	Market access	Integration	Data transformation	Platform architecture	Price model	Revenue model
Dawex	Transaction	Neutral	Hybrid	Unspecific	Raw Data	Centralised	Fixed-price	Freemium
IOTA	Transaction	Neutral	Hybrid	Specific	Raw Data	Decentralised	Progressive	Transaction Fee
Databroker DAO	Transaction	Neutral	Hybrid	Specific	Raw Data	Decentralised	Progressive	n/a
Streamr	Transaction	Neutral	Hybrid	Unspecific	Aggregation	Decentralised	Progressive	n/a
Data Intelligence Hub	Data	Neutral	Hybrid	Unspecific	Raw Data	Centralised	Multiple	Transaction Fee
Advaneo	Data	Neutral	Hybrid	Unspecific	Raw Data	Centralised	Fixed-price	Transaction Fee
Otonomo	Data	Neutral	Hybrid	Specific	Aggregation	Centralised	Fixed-price	Transaction Fee
Datafairplay	Transaction	Neutral	Hybrid	Specific	Normalisation	Centralised	Progressive	Transaction Fee
InfoChimps	Transaction	Neutral	Hybrid	Unspecific	Raw Data	Centralised	Fixed-price	Transaction Fee
Qlik	Data	Provider	Hybrid	Unspecific	Raw Data	Centralised	Package	Freemium
xDayta	Transaction	Neutral	Open	Unspecific	Raw Data	Centralised	Fixed-price	n/a
Kasabi	Transaction	Neutral	Open	Unspecific	Normalisation	Centralised	Fixed-price	Freemium
Here OLP	Data	Provider	Hybrid	Specific	Aggregation	Centralised	Multiple	Freemium
Azure Data Marketplace	Transaction	Neutral	Hybrid	Unspecific	Raw Data	Centralised	Fixed-price	Transaction Fee
International Data Spaces	Data	Neutral	Hybrid	Unspecific	Raw Data	Decentralised	Multiple	Transaction Fee
Caruso Dataplace	Data	Neutral	Hybrid	Specific	Aggregation	Centralised	Multiple	Membership Fee

Figure 2: Classification of Data Marketplaces

We examined the information derived from Data-Driven Economy: Challenges and opportunities, Data Marketplaces: Trends and Monetisation of Data Goods, EU Data Landscape and our Industrial Partners

There is also a variety of software solutions for operating self-hosted data management platforms, such as DSpace, CKAN, Figshare, Zenodo, ePrints, EUDAT.

From the DM list included in [6], we examined the DMs listed in Table 6. Finding DMs with appropriate mechanisms for developing an interoperability solution is a challenging task. Some of them do not provide APIs. Other DMs mention the availability of an API on their website, but upon closer inspection it turned out that there is nothing implemented. Some DMs do not use REST APIs but cryptocurrency-based solutions. Most of them have no semantic Models and the ones that use some Semantic information Models don't cover the entire requirements for the i3-Market needs. Also important to highlight the lack of openly available Semantic Engine components that can be used to manage and manipulate the semantic metadata and operations that are necessary for the i3-MARKET Semantic Models & Data Lifecycle Processes.

For the planning on the development of the i3-Market Semantic Models and O-Casus packages we studied a List of relevant standards

The main ontologies, vocabularies and metadata schemas that have been studied and compared to check their applicability in i3-Market range from general-purpose, linked data- or open data-specific metadata standards to identity, data assets market focused metadata standards based on their alignment to the i3-Market scope as follows:

- Dublin Core Metadata Initiative (DCMI), a domain-agnostic and widely-used metadata standard that can be put into use to describe any web resource. The Dublin Core Metadata Element Set (Version 1.1) is practically a vocabulary of 15 core properties while the full specification of all metadata terms maintained by the Dublin Core Metadata Initiative, including properties, vocabulary encoding schemes, syntax encoding schemes, and classes, expands to the subsequent fields.
- schema.org, https://schema.org/
- INSPIRE Data Specifications, https://joinup.ec.europa.eu/collection/inspire/solution/inspiredata-specifications/about and the related directive: https://eur-lex.europa.eu/legalcontent/EN/ALL/?uri=CELEX%3A32007L0002
- FAIR principles, https://www.go-fair.org/fair-principles/
- RDA FAIR data maturity model, https://joinup.ec.europa.eu/collection/semanticinteroperability-community-semic/solution/rda-fair
- DIN SPEC 27070 "Requirements and reference architecture of a security gateway for the exchange of industry data and services", https://www.beuth.de/de/technische-regel/din-spec27070/319111044 and https://www.internationaldataspaces.org/ids-

officially-a-standard-dinspec-27070-is-published/

- W3C Web Content Accessibility Guidelines, https://www.w3.org/WAI/standardsguidelines/wcag
- Data Catalog Vocabulary (DCAT), a RDF-based vocabulary designed to facilitate interoperability between data catalogs published on the Web. On the basis of DCAT, the DCAT Application profile for data portals in Europe (DCAT-AP) has been proposed for describing public sector datasets in Europe in order to enable a cross-data portal search. Another profile of DCAT that has been proposed is the Asset Description Metadata Schema (ADMS) which is focused on the description of semantic assets.
- Vocabulary of Interlinked Datasets (VoID), an RDF Schema vocabulary that specializes on metadata for linked RDF datasets for cataloguing and archiving purposes. VoiD provisions for: (a) General metadata in compliance with the Dublin Core metadata schema; (b) Access metadata regarding how access to data can be ensured using various protocols; (c) Structural metadata related to the structure and the schema of datasets for querying, and (d) Description of links between datasets.
- DataCite Metadata Schema is a list of core metadata properties chosen for an accurate and consistent identification of a resource for citation and retrieval purposes, along with recommended use instructions.
- CKAN Domain Model that contains metadata for datasets and their associated 'resources' (i.e. files, APIs, etc.). Each dataset needs to have certain "core" metadata attributes, but may be complemented by arbitrary additional metadata in the form of "extra" key/value associations, and explicit relationships between datasets (such as depends on, child of, derived from, etc.).
- ISO 19115 (Geographic information -- Metadata) which defines the metadata schema required for describing geographic information and services. The metadata included in ISO 19115 address the identification, the extent, the quality, the spatial and temporal aspects, the content, the spatial reference, the portrayal, distribution, and other properties of digital geographic data and services.
- W3c Verifiable Credentials. This specification provides a standard way to express credentials on the Web in a way that is cryptographically secure, privacy respecting, and machine-verifiable.
- IT Service Ontology. OWL Ontology for IT Services delivered via the Cloud.
- ETSI European Telecommunications Standards Institute
- PROV-O The Provenance Ontology
- QUDT Quantities, Units, Dimensions, and Types Ontology
- SAREF Smart Appliances REFerence (SAREF) ontology
- SKOS Simple Knowledge Organization System
- SKOS-XL Simple Knowledge Organization System eXtension for Labels
- WSDL Web Services Description Language

- EU Vocabularies Frequency Named Authority List http://publications.europa.eu/resource/authority/frequency .
- EU Vocabularies File Type Named Authority List http://publications.europa.eu/resource/authority/file-type .
- EU Vocabularies Languages Named Authority List http://publications.europa.eu/resource/authority/language .
- EU Vocabularies Continents Named Authority List [10], EU Vocabularies Countries Named Authority List [11], EU Vocabularies Places Named Authority List, Geonames http://publications.europa.eu/resource/authority/continent/ , http://publications.europa.eu/resource/authority/country , http://publications.europa.eu/resource/authority/place/, http://sws.geonames.org/
- ADMS licence type vocabulary http://purl.org/adms/licencetype/
- Distribution availability vocabulary http://data.europa.eu/r5r/availability/
- The International Data Space (IDS)

The use of standardized semantic meta-data models and interaction patterns is important to enabling interoperability between nodes, user-friendly services, exchangeability of Data Assets, representation of actors (Marketplaces, Providers, Consumers, Owners) and data exchange between different instances in the Infrastructure Ecosystem. A variety of standards already exist for sub-specific topics and domains, the most suitable ones to set up a common information model are selected and integrated into a high-level collection of vocabularies and ontologies. ON top of these models, we created i3-Market core models to define the missing parts and for the main operational interactions and links among entities. Within i3-Market Backplane, the Information Ecosystem and the Infrastructure Ecosystem have to be combined to enable a seamless exchange of information and operations in a federated distributed architecture.

More Information on the Background solutions & Technologies introduced in the project are presented in the Section 3.

2.2 Main initial Contributions

From a meta-modelling perspective, the i3-Market has raised certain requirements that go beyond the simple main description of data sets, adding information models to define other entities, operations actors, sharing agreements, data details. While the existing semantic models covers only partially the requirements for the backplane scopes we imported, linked and in case extended common vocabularies and created the i3-Market Semantic Core Model, pricing model, contractual model for Data Sharing Agreements and service agreements for contracts to compile a collection of semantic information models in O-CASUS library to cover the needs.

IDS Information Model is a straightforward RDFS ontology with limited use of OWL features, uses SHACL for validation purposes and makes use of SPARQL queries to retrieve self-descriptions, e.g., from a Broker, GAIA-X aims at a hierarchical organization of information, e.g., that one Node represents “a pan-European Node Provider that is structured into country regions, which are themselves structured into data centre locations, racks and individual

servers, which themselves are exposed as GAIA-X Nodes." It is a subject of ongoing discussions whether or how, e.g., redundant storage and synchronization problems in such a hierarchy can be avoided by an inheritance mechanism that propagates properties of nodes through the hierarchy.

From an operational perspective, i3-Market envisages Semantic Engine components (eg. SEED) to manage query mechanisms on top of the Registries Graphs, including complex discovery and retrieve checks that make sure, e.g., that the necessary information are retrieved by the actors and services. Very important also the functionalities related to the creation and registration of the Data Offering Descriptions and the management of local and federated Registries.

It is a subject of ongoing discussions whether or how, e.g., redundant storage and synchronization problems in such a hierarchy can be avoided by an inheritance mechanism that propagates properties of nodes through the federation Indexing and querying.

Online i3-Market Semantic Model repository and community management

The resulting i3-Market Semantic Models will be shared not only with Consortium Partners but also with stakeholders and community. As part of open-source assets, the data models, documentations and files used in the i3-Market project are made available, like:

- The i3-Market data Pack is the set of files
- The i3-Market Semantic Model documentation.
- The Models Files will be shared publicly in Git repository.

3 Background & Technologies

3.1 Semantic Models and Ontologies in i3-Market

A data marketplace is an online transactional location or store that facilitates the buying and selling of data. As many businesses seek to augment or enrich internal data sets with external data, cloud-based data marketplaces are appearing at a growing rate to match data consumers with the right data sellers.

Typical data types for sale in a data marketplace can range from business intelligence and research, demographic, firmographic, and market data to business intelligence and public data. A data marketplace is a more public (and sometimes commercial or monetized) form of data sharing. Data sharing has a long history in academic, research, and public policy circles but in recent years has made enormous inroads into private enterprises, from big business to analyst, consulting, and market intelligence firms. Data consumers include government, analysts, big business, and market intelligence firms. As data volumes continue to explode and machine learning and AI become more important in decision-making, data marketplaces are helping organizations reduce the effort and cost involved in locating required data sets and helping data providers extend their market reach.

Big Data is supported by continuous however, heterogeneity of underlying data sources (,e.g. in IoT spaces), devices and communication technologies and interoperability in different layers, from communication and seamless integration of platforms to interoperability of data to a global scale.

In a white paper on interoperability [1] it is discussed that many layers of interoperability exist:

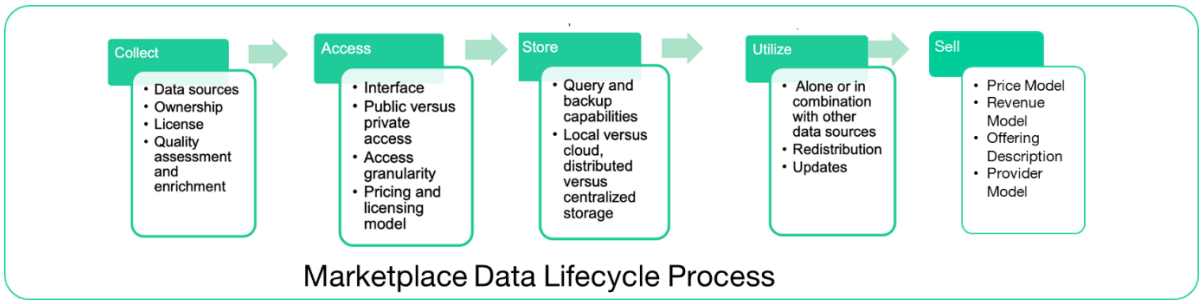
- Technical Interoperability
- Syntactical Interoperability
- Semantic Interoperability
- Organizational Interoperability
- Dynamic interoperability
- Static interoperability

Discovery, understanding, and collaboration at this level require more than just an ability to interface and to exchange data. Whereas interoperability is “the ability of two or more systems or components to exchange data and use Information" [2] , semantic interoperability “means enabling different agents, services, and applications to exchange information, data and knowledge in a meaningful way, on and off the Web” [2] [3].

Semantic interoperability is achieved when interacting systems attribute the same meaning to an exchanged piece of data, ensuring consistency of the data across systems regardless of individual data information. This consistency of meaning can be derived from pre-existing standards or agreements on the description and meaning of data or it can be derived in a dynamic way using shared vocabularies either in a schema form or in an ontology-driven approach.

In i3-Market we are aiming at an innovative approach for Semantic Data, Metadata and Modelling Activities.

i3-MARKET Data Model & Data Lifecycle Process



To lead to concept of O-CASUS, which is an idea based on the Data Lifecycle Process where we:

- Compile Vocabularies and taxonomies in relation to Marketplaces Metadata, Operation and Management;
- Formalise the state of current marketplaces by using best practices and standards;
- Compile an Ontology for Collecting, Accessing, Store, Utilise and Sell Data.

In Section 2.2, we present various of these semantic interoperability challenges to overcome to deliver the potential of innovative services that open data market is looking for.

3.2 Challenges for Semantic Interoperability

3.2.1 Semantic Interoperability

The overall challenge in interoperability is first to ensure technical interoperability from technologies to deliver a mass of information and then complementary challenges are for the information to be understood and processed. The tables below present a summary of the challenges for technical and semantic interoperability, as reported by the European Research Cluster [4].

Table 1 IoT Semantic Interoperability Challenges/Requirements

Requirement(s)	Rationale
Best practices Avoid spreading effort in addressing interoperability for internationally adopted protocols.	Use clear models development and testing methodologies leading to improve quality while reducing time and costs in a full chain optimized development cycle. Define, if needed, specifications to improve interoperability.
Validation of specifications	Specifications development time could be too long.

Reduce ambiguities in specifications and development time.	Ambiguities in specifications could lead to major non-interoperability issues. Quality, time and cost factors lead to the needs of models and automation.
Test specifications Provide market accepted test specifications ensuring minimum accepted level of interoperability.	The absence of test specifications leads inevitably to different specifications implementation and interoperability issues. Development of test specifications is often too expensive for a limited set of stakeholders and effort should be collectively shared.
Tools and validation programs Develop market accepted and affordable test tools used in market accepted validation programs.	Development of test tools is expensive. Available test tools may not be sufficient and not optimized for all tests needs. The full chain of specifications to tool development not considered. Providing final confidence to end users with consistent tests not always considered.
Integration Support multiple resources (sensors, actuators) and relevant types of data sources (independently of vendor and resource location).	Enable scalable sharing and integration of distributed data sources. All IoT applications involve multiple heterogeneous devices. Orchestrate resources in order to automatically formulate composite workflows as required by end-user applications.
Annotation Enable the (automated) linking of relevant data sources.	Linking of data sources facilitates application integration and reuse of data. Enable interactions between sensors and between services. Built on the standards.
Management Enable the creation and management of virtual sensors and virtual resources based on the composition and fusion of multiple data sources.	Application development and integration involve multiple distributed and heterogeneous data sources to be processed in parallel.
Discovery Provide the means for discovering and selecting resources and data sources pertaining to the application.	End users need a high-level interface to be accessed. Provide the means for describing/formulating IoT services and applications according to high-level descriptions.
Analysis and Reasoning Provide analytical and reasoning tools on top of semantic level capabilities.	IoT addresses large-scale environments with numerous resources featuring different functionalities and capabilities.
Visualisation Optimize usage of info across multiple users sharing these resources.	For a better understanding and reporting of resource interactions or interactions between services.

3.2.2 Why using Semantics?

Many of the problems present in current technologies are going to be mainly generated by interoperability problems, thus there are three persistent problems:

1. Users are offered relatively small numbers of Internet services, which they cannot personalize to meet their evolving needs; communities of users cannot tailor services to help create, improve and sustain their social interactions;
2. The Internet services that are offered are typically technology-driven and static, designed to maximize usage of capabilities of underlying network technologies and not to satisfy user requirements per se, and thus cannot be readily adapted to their changing operational context;
3. Network operators cannot configure their networks to operate effectively in the face of changing service usage patterns and rapid networking technology deployment; networks can only be optimized, on an individual basis, to meet specific low-level objectives, often resulting in sub-optimal operation in comparison to the more important business and service user objectives.

3.2.3 Data Catalog Vocabulary (DCAT) - Version 3

W3C (World Wide Web Consortium) Recommendation

DCAT is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web. This document defines the schema and provides examples for its use.

DCAT enables a publisher to describe datasets and data services in a catalog using a standard model and vocabulary that facilitates the consumption and aggregation of metadata from multiple catalogs. This can increase the discoverability of datasets and data services. It also makes it possible to have a decentralized approach to publishing data catalogs and makes federated search for datasets across catalogs in multiple sites possible using the same query mechanism and structure.https://www.w3.org/TR/vocab-dcat-3/

Also, its extension DCAT-AP

The DCAT Application Profile for data portals in Europe (DCAT-AP) is a specification based on the Data Catalogue Vocabulary (DCAT) developed by W3C.

This application profile is a specification for metadata records to meet the specific application needs of **data portals in Europe** while providing semantic interoperability with other applications on the basis of reuse of established controlled vocabularies (e.g. EuroVoc) and mappings to existing metadata vocabularies (e.g. Dublin Core, SDMX, INSPIRE metadata, etc.).

DCAT-AP provides a common specification for describing public sector datasets in Europe to enable the exchange of descriptions of datasets among data portals. DCAT-AP allows:

- **Data catalogues** to describe their dataset collections using a standardised description, while keeping their own system for documenting and storing them.
- **Content aggregators**, such as the European Data Portal, to aggregate such descriptions into a single point of access.
- **Data consumers** to more easily find datasets through a single point of access.

DCAT-AP has an extension GeoDCAT-AP for describing geospatial datasets, dataset series and services. Another extension, StatDCAT-AP, provides specifications and tools that enhance interoperability between descriptions of statistical data sets within the statistical domain and between statistical data and open data portals.

https://joinup.ec.europa.eu/solution/dcat-application-profile-data-portals-europe/release/200

3.2.4 RDF Store

As part of the Marketplace persistence framework back-end layer, we will need to use and deploy a database that will be able to store our semantic (meta)data in the best way. This database will represent the main registry and repository where all the semantically annotated (meta)data will be uploaded and saved.

In the persistent database we will need to store all the (meta)data Descriptions created and collected by marketplace stakeholders, e.g. with the information about Providers, Consumers, Offering Descriptions and Recipes. In our research for a semantic interoperability in i3-Market we decided to model our Providers, Consumers, Data Offering Descriptions and Parameters following an RDF schema model, annotated with our **i3-Market Semantic Core Model** and represent and exchange data in Json serialization format.

So due to the nature of such kind of (meta) data we need to choose the best solution for storing, managing, accessing and retrieving information.

The semantic descriptions are generated following the **i3-Market Core Model**, annotated with the **i3-Market Domain Models** and mapped with the **i3-Market Application Model** vocabularies and then loaded into an RDF Triple-Store.

RDF triple-store is a type of graph database that stores semantic facts. Being a graph database, triple-store stores data as a network of objects with materialized links between them. This makes RDF triple-store a preferred choice for managing highly interconnected data. Triple-stores are more flexible and less costly than a relational database, for example.

The RDF database, often called a semantic graph database, is also capable of handling powerful semantic queries and of using inference for uncovering new information out of the existing relations. In contrast to other types of graph databases, RDF triple-store engines support the concurrent storage of data, meta-data and schema models (e.g. the so called ontologies). Models/Ontologies allow for the formal description of the data. They specify both object classes and relationship properties, and their hierarchical order as we use our i3-Market models to describe our resources.

This allows creating a unified knowledge base grounded in common semantic models that allow combining all meta-data coming from different sources making them semantically interoperable to

- create coherent queries independently from the source, format, date, time, provider, etc.
- enable the implementation of more efficient semantic querying features
- enrich the data, make it more complete, more reliable, more accessible
- enable to perform inference as triple materialization from some of the relations

In the following paragraphs, we are going to give some more information and examples about the semantic data formalization, query interface and the interface of the semantic framework backend layer within the Backplane.

3.2.5 RDF: Data Model & Serialisation Formats

Linked Data is based around describing real world things using the Resource Description Framework (RDF). The following paragraphs introduce the basic data model, and then outline existing formats to serialize data model in RDF.

3.2.5.1.1 RDF Data Model

RDF is a very simple, flexible, and schema-less to express and process a series of simple assertions. Consider the following example: “Sensor A measures 21C.” Each statement, i.e. piece of information, is represented in the form of **triples** (RDF triples) that link a **subject** (“Sensor A”), a **predicate** (“measures”), and an **object** (“21C”). The subject is the thing that is described, i.e. the resource in question. The predicate is a term used to describe or modify some aspect of the subject. It is used to denote relationships between the subject and the object. The object is, in RDF, the “target” or “value” of the triple. It can be another resource or just a literal value such as a number or word.

In RDF, resources are represented by Uniform Resource Identifiers (URIs). The subject of RDF triples must always be a resource. The typical way to represent an RDF triple is a graph, with the subject and object being nodes and the predicate a directed edge from the subject to the object. So the above example statement could be turned into an RDF triple illustrated in the figure below:

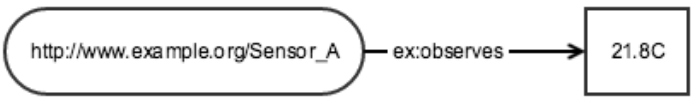


Figure 3: RDF triple in graph representation describing “Sensor A measures 21.8°C.”

Since objects can also be a resource with predicates and objects on their own, single triples are connected to a so-called RDF graph. In terms of graph theory, the RDF graph is a labeled and directed graph. As the illustration we extend the previous example, replacing the literal “21.8C” by a resource “Measurement” for the object in the RDF triple in the following figure. The resource itself has two predicates assigning a unit and the actual value to the measurement. The unit is again represented by a resource and the value is numerical literal. The resulting RDF graph looks as follows:

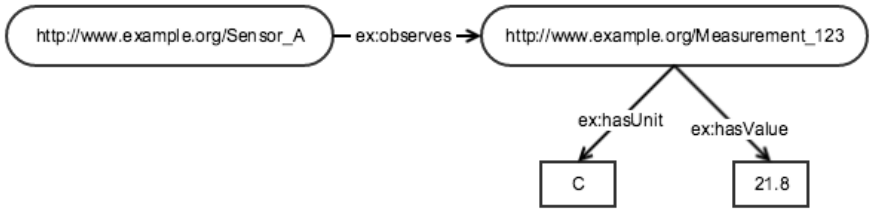


Figure 4 *Simple RDF graph including the example RDF triple*

3.2.5.1.2 Serialisation Formats

The RDF data model itself does not describe the format in which the data, i.e. the RDF graph structure, is stored, processed, or transferred. Several formats exist that serialize RDF data; the following overview lists the most popular formats, including a short description of their main characteristics and examples. Figure 2 shows a simple RDF graph to serve as the basis.

RDF/XML: The RDF/XML syntax is standardized by the W3C and is widely used to publish Linked Data on the Web. On the downside, however, the XML syntax is also viewed as difficult for humans to read and write. This recommends consideration of

a) other serialization formats in data management and control workflows that involve human intervention and
b) the provision of alternative serializations for consumers who may wish to examine the raw RDF data.

The RDF/XML syntax is described in detail as part of the W3C RDF Primer. The MIME type that should be used for RDF/XML within HTTP content negotiation is application/rdf+xml. The listing below shows the RDF/XML serialization for the RDF graph.

RDF/XML Serialisation Example

```
<?xml version="1.0"?>
<rdf:RDF xmlns:ex="http://www.example.org/"
<rdf:Description rdf:about=" http://www.example.org/Sensor_A">
        <ex:title>21.8°C</ex:title>
</rdf:Description>
</rdf:RDF>
```

Turtle: Turtle (Terse RDF Triple Language) is a plain text format for serializing RDF data. It has support for namespace prefixes and other shorthands, making Turtle typically the serialization format of choice for reading RDF triples or writing them by hand. A detailed introduction to Turtle is given in the W3C Team Submission document Turtle. It was accepted as a first working draft by the World Wide Web Consortium (W3C) RDF Working Group in August 2011, and parsing and serializing RDF data is supported by a large number of RDF toolkits. The following listing shows the serialization listing for the example RDF graph in Turtle syntax.

Turtle Serialisation Example

```
@prefix : <http://www.example.org/> .
:Sensor_A :measures “21.8°C”
```

N-Triples: The N-Triples syntax is a subset of Turtle, excluding features such as namespace prefixes and shorthands. Since all URIs must be specified in full in each triple, this serialization format involves a lot of redundancy, typically resulting in large N-Triples particularly compared to Turtle but also to RDF/XML. This redundancy, however, enables N-Triples files to be parsed one line at a time, benefitting the loading and processing of large data files that will not fit into main memory. The redundancy also allows compressing N-Triples files with a high compression ratio, thus reducing network traffic when exchanging files. These two factors make N-Triples the de facto standard for exchanging large dumps of Linked Data. The complete definition of the N-Triples syntax is given as part of the W3C RDF Test Cases recommendation. The following listing in Table 3 represents the N-Triples serialization of the example RDF graph.

N-Triples Serialisation Example

```
<http://www.example.org/Sensor_A> <http://www.example.org/measures> “21.8°C”@en-UK
.
```

JSON-LD: Many developers have little or no experience with Linked Data, RDF or common RDF serialization formats such as N-Triples and Turtle. This produces extra overhead in the form of a steeper learning curve when integrating new systems to consume linked data. To counter this, the project consortium decided to use a format based on a common serialization format such as XML or JSON. Thus, the two remaining options are RDF/XML and JSON-LD. JSON-LD was chosen over RDF/XML as the data format for all Linked Data items in BigIoT. JSON-LD is a JSON- based serialization for Linked Data with the following design goals:

- **Simplicity**: There is no need for extra processors or software libraries, just the knowledge of some basic keywords.
- **Compatibility**: JSON-LD documents are always valid JSON documents, so the standard libraries from JSON can be used.
- **Expressiveness**: Real-world data models can be expressed because the syntax serializes a directed graph.
- **Terseness**: The syntax is readable for humans and developers need little effort to use it.
- **Zero Edits**: Most of the time JSON-LD can be devolved easily from JSON- based systems.
- **Usable as RDF**: JSON-LD can be mapped to / from RDF and can be used as RDF without having any knowledge about RDF.

From the above, terseness and simplicity are the main reasons why JSON-LD was chosen over RDF/XML. JSON-LD also allows for referencing external files to provide context. This means contextual information can be requested on-demand and makes JSON-LD better suited to situations with high response times or low bandwidth usage requirements. More information can be found in http://json-ld.org/.

The data model underlying JSON-LD is a labeled, directed graph. There are a few important keywords, such as @context, @id, @value, and @type. These keywords are the core part of JSON-LD. Four basic concepts should be considered:

- **Context**: A context in JSON-LD allows using shortcut terms to make the JSON-LD file shorter and easier to read (as well as increasing its resemblance with pure JSON). The context maps terms to IRIs. A context can also be externalized and reused for multiple JSON-LD files by referencing its URI.

- **IRIs**: Internationalised Resource Identifiers (IRIs) are used to identify nodes and properties in Linked Data. In JSON-LD two kinds of IRIs are used: absolute IRIs and relative IRIs. JSON-LD also allows defining a common prefix for relative IRIs using the keyword @vocab.
- **Node Identifiers**: Node identifiers (using the keyword @id) reference nodes externally. As a result of using @id, any RDF triples produced for this node would use the given IRI as their subject. If an application follows this IRI it should be able to find some more information about the node. If no node identifier is specified, the RDF mapping will use blank nodes.
- **Specifying the Type**: It is possible to specify the type of a distinct node with the keyword @type. When mapping to RDF, this creates a new triple with the node as the subject, a property rdf:type and the given type as the object (given as an IRI).

JSON-LD Example

```
[{"@id":"http://www.example.org/Sensor_A","http://www.example.org/measures":[{"@val
ue":"21.8C"}]}]
```

3.2.6 SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) is the most popular query language to retrieve and manipulate data stored in RDF, and became an official W3C Recommendation in 2008. Depending on the purpose, SPARQL distinguishes the following for query variations:

• SELECT query: extraction of (raw) information from the data

• CONSTRUCT query: extraction of information and transformation into RDF

• ASK query: extraction of information resulting a True/False answer

• DESCRIBE query: extraction of RDF graph that describes the resources found

Given that RDF forms a directed, labeled graph for representing information, the most basic construct of a SPARQL query is a so-called **basic graph pattern**. Such a pattern is very similar to an RDF triple with the exception that the subject, predicate or object may be a variable. A basic graph pattern matches a subgraph of the RDF data when RDF terms from that subgraph may be substituted for the variables and the result is RDF graph equivalent to the subgraph. Using the same identifier for variables also allow combining multiple graph patterns. Besides aforementioned graph patterns, the SPARQL 1.1 standard also supports the sorting (ORDER BY), and the limitation of result sets (LIMIT, OFFSET), the elimination of duplicates (DISTINCT), the formulation of conditions over the value of variables (FILTER), and the possibility to declare a constraint as OPTIONAL. The SPARQL 1.1 standard significantly extended the expressiveness of SPARQL. In more detail the new features include:

- Grouping (GROUP BY), and conditions on groups (HAVING)
- Aggregates (CONT, SUM, MIN, MAX, AVG, etc.)
- Subqueries to embed SPARQL queries directly within other queries
- Negation to, e.g., check for the absence of data triples
- Project expression, e.g., to use numerical result values in the SELECT clause within mathematical formulas and assign new variable names to the result
- Update statements to add, change, or delete statements

- Variable assignments to bind expressions to variables in a graph pattern
- New built-in functions and operators, including string functions (e.g., CONCAT, CONTAINS), string digest functions (e.g., MD5, SHA1), numeric functions (e.g., ABS, ROUND), or date/time functions (e.g., NOW, DAY, HOURS)

As mentioned previously that RDF graph data is represented as triples i.e., “subject”, “predicate”, “object”. A very basic SPARQL, which brings back 100 triples from the RDF graph, can be written as:

SPARQL Example

```
SELECT * WHERE {?s ?p ?o} LIMIT 100
```

For the Semantic subsystems in charge of dealing with “Semantic Data Management” we can highlight parts like:

- Semantic Data Storage: This component on receiving semantically annotated data push data into triple-store, for example, virtuoso. A Triple-store is database and responsible to store semantic data and process the SPARQL queries.
- Semantic Mapping: This component does semantic mappings and transform data (JSON to RDF) received from API endpoints.
- Vocabulary Management: This environment keeps and manages all of the vocabularies, defined as i3-Market Semantic model, used in different operations of the semantic engine.
- Offering Registration: This component is basically REST APIs exposed as endpoints. Semantic engine exposes different endpoints for offering registration. Examples are:
    - register data provider,
    - register data offering of a data provider,
    - update data offerings,
    - deleting a data offering
    - query existing offerings, etc.
- Offering Discovery: This component is basically REST APIs exposed as endpoints. Semantic engine exposes different endpoints for offering discoveries and retrieve. Examples are:
    - Retrieve list of Data Offerings,
    - discover data offerings by Providers,
    - discover data offerings by parameters,
    - search for particular metadata, etc.

The following picture shows a detailed landscape of the current set of microservices (cubes), API’s (little yellow rectangles), components (blue rectangles), and storages (white rectangles) on i3-MARKET.

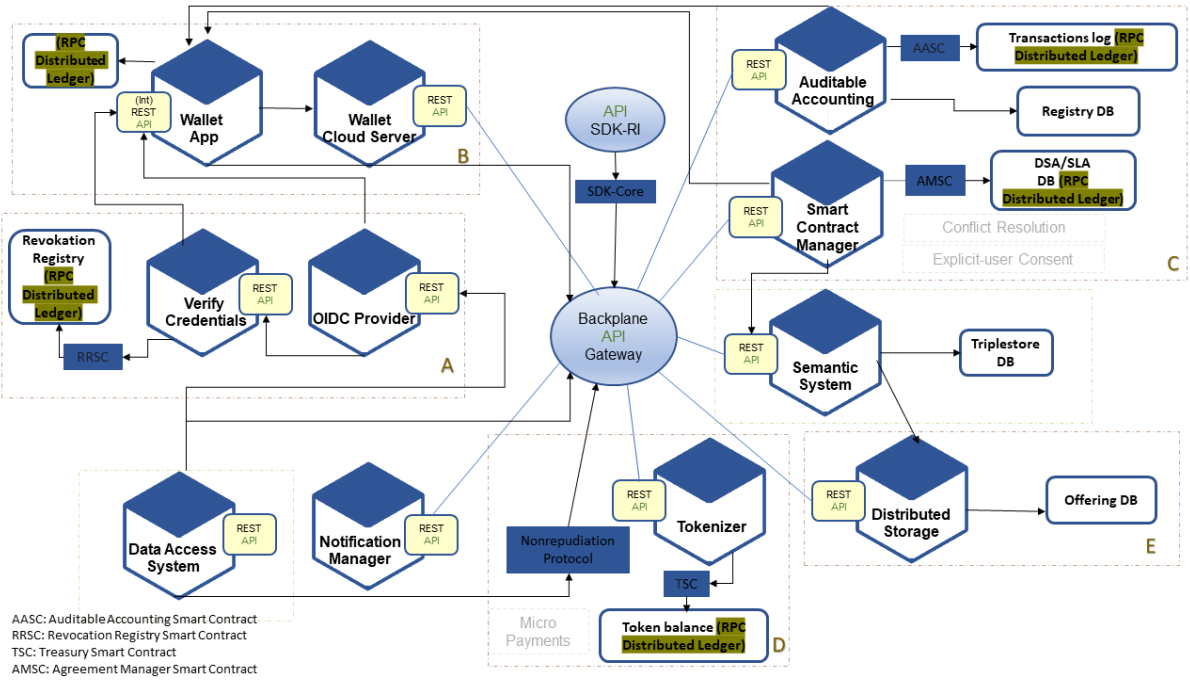


Figure 5: i3-MARKET services layout (D2.6)

How the above components interact with each other can be explained with the use of a scenario, for instance, registration of data provider. To complete this task, a data provider fills a template provided by offering registration component. The template as a JSON request is Semantically mapped by the Semantic Mapping component and is forwarded towards the Semantic Data Storage component where this annotated data is pushed into the triple-store.

4 i3-Market Semantic Models

4.1 i3-Market Models Specifications

In the following sections, we will propose the i3-Market Semantic Core Model and the Semantic Models imported and extended that create the collection of O-CASUS ontologies based on the terminologies, definitions and vocabularies needed to represent the i3-Market domain entities and operations. These concepts and their relationships are explained in more detail, including additional sub-concepts.

The O-CASUS Semantic Models comprise a collection of ontologies and vocabularies to cover the concepts used in the Backplane to define the

- i3-Market Semantic Core Model
- W3c Data Catalog Vocabulary (DCAT and DCAT-AP)
- W3c Vocabulary of Interlinked Datasets (VoID),
- W3c Verifiable Credentials and DID
- SKOS Simple Knowledge Organization System
- IT Service Ontology
- EU Vocabularies Frequency Named Authority List
- EU Vocabularies File Type Named Authority List
- EU Vocabularies Languages Named Authority List
- EU Vocabularies Continents Named Authority List
- ADMS licence type vocabulary
- Distribution availability vocabulary
- Domain Annotations

4.2 i3-Market Semantic Core Model

4.2.1 Schema Namespaces

This Section will list the main potential namespaces that will be used in i3-Market Semantic Core Model.

Table 2. Schema Namespaces

Prefix	Ontology/Language	Namespace
DataCategories	i3-Market Categories	http://i3.market.eu/auth/dataCatagory
core	i3-Market Core Models	http://i3-market.eu/backplane/core/
pricingmodel	i3-Market Pricing Model	http://i3-market.eu/backplane/pricingmode/
dct	Dc terms	http://purl.org/dc/terms/
dcat	Dcat Vocabulary	http://www.w3.org/ns/dcat#
schema	Schema.org ontology	http://schema.org/
td	Things description	http://w3c.github.io/wot/wot.owl#
xsd	XML schema Definition	http://www.w3.org/2001/XMLSchema#
rdf	RDF Concepts Vocabulary	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	RDF Schema ontology	http://www.w3.org/2000/01/rdf-schema#

4.2.2 Modularization of the i3-Market Semantic Core Model

One of the key aspects when designing a semantic model is reusing of knowledge. Once a semantic model is created for a domain, it should be (at least to some degree) reusable for other applications in the same domain. To simplify both semantic model development and reuse, a modular design is beneficial. Based on the project specification and the domain environment, the semantic models can be modularized according to their scope, as follow:

- Organization module
- Market Module
- Provider module
- Consumer module

- Owner Model
- Query module
- Data Offering module
- Contractual Parameters Module
- Links to Pricing Module and the other vocabularies and ontologies to cover the various parts of the i3-Market O-CASUS Sematic Information Models.

Figure 2 illustrates the high level of i3-Market Semantic Models which include all the basic conceptual entities and their relationship of all modules. Details of each module will be presented in the next subsections.

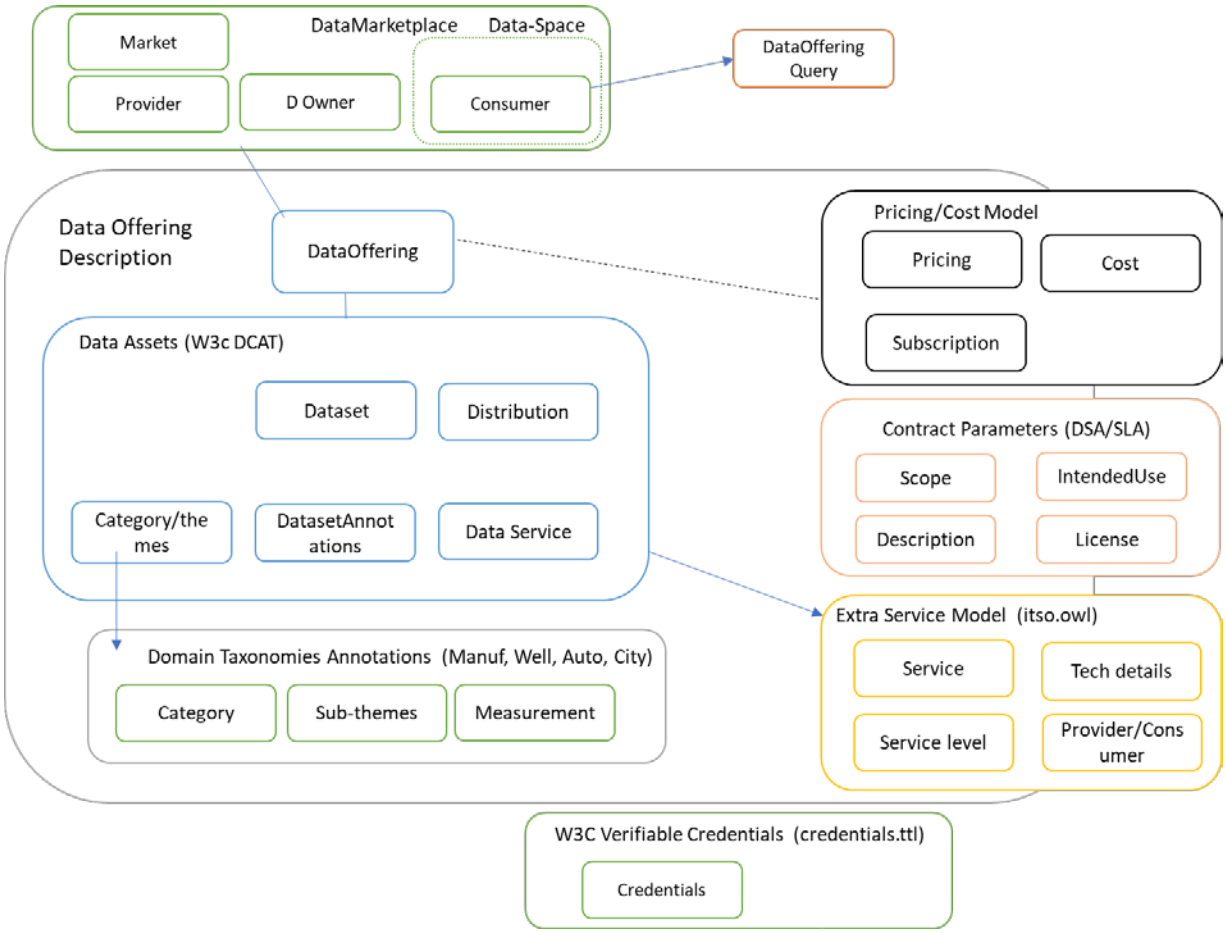


Figure 6 High Level Semantic Models Structure of main modules

One of The main contribution of the Data Models / Ontologies is the consolidation of the i3-Market Semantic Models and the integration and extensions of other common Sematic Models to enable the mapping of the metadata describing the data assets, contracts and operations, provided from i3-Market stakeholders, to the model/"ontology" concepts to capture the

structural and semantic characteristics metadata of the various entities in each data asset = Data Offering.

More specifically, the core uses of this models are for:

1) data Registration of metadata descriptions which corresponds to the data harmonization process. In this way, each provided data asset will be registered in our Registry with concepts from the i3-Market data offering model in a semi-automatic way;

2) metadata linking where any provided data asset metadata will be linked with other relevant sources (or data assets) that exist in the Backplane;

3) data Discovery (for local or federated Registries) which involves the development of algorithms and software for supporting the selection of the most appropriate metadata that best match user preferences.

4) Management of information related to Smart Contract, Data access and transfer, pricing models, identity and credentials identifications, notifications.

The i3-Market Models will be used for capturing the structural and semantic metadata characteristics of the various entities involved in the i3-Market backplane domain, whereas the underlying conceptual models facilitate the use of lightweight reasoning during the discovery and operational process e.g. for contracts and service/agreements, data access/transfer operations.

Model Coding: The representation of the ontology capture was transformed into a

formal (ontology) language (e.g. OWL, Turtle).

- Top-level Ontology: Create a top-level ontology that describes the main high level concepts and links for the i3-Market backplane domain

- Integrating Existing sub topic Ontologies: Use existing related domains ontologies and incorporate them under the top-level ontology.

- Expanding the Domain-level use-cases pilots categories/annotations: Expand (Build-on-top) the domain-level conceptSkema categorization taxonomies and include the new concepts, data fields and relationships between the concepts that were extracted from the demonstrators' domains.

i3-Market High Level view of Building Models

4.3 Data Marketplaces and Data Spaces Actors and

Provider module

A provider can be Marketplace , Data Space or.. service instance that offers available DataOfferings. A provider is described through **core:Provider** class. At this stage, each provider will have a name (schema:name) and id (core:providerId) and its organization as shown in Table 1. More information about the provider can be added in the future.

Property Name	Data Types	Description

core:providerId	string	Provider id
providerDescription	string	A description of the provider
providerName	string	Name of the provider
:sourceOrganization	core:Organization	A provider's organization

Table 3- Provider properties

- **Organization module**

A provider may also describe its organization. The provider's organization will be an instance of the **schema or org (Organization ontology):Organization** class. Connection between the Provider and the Organization is the :sourceOrganization property. Table 2 below presents some basic properties of the organization, e.g. at the moment example taken from the :Organization).

Property Name	Data Types	Description
core:organizationId	string	Organization id
organizationName	string	Name of the organization
:address	string	Physical address of the organization
contactPoint	schema:ContactPoint	A contact point for organization
organizationDescription	string	A description of the organization

Table 4- Organization properties

- **Consumer Module**

A consumer can be either an entity, application or service instance that requires access to data resources in order to implement an intended service or function. In the consumer model, we create the **core:Consumer** class that represents the i3-Market Consumers. Same as the provider, the consumer is also linked to the Organisation. The Table 3 below presents some basic properties of the core:Consumer.

Property Name	Data Types	Description
core:consumerId	string	Consumer id
core:dataOfferingQuery	core:DataOfferingQuery	Query to i3-Market of consumer

consumerDescription	string	A description of the consumer
consumerName	string	Name of the consumer
:sourceOrganization	schema:Organization	A consumer's organization
core:subscribedTo	core:DataOffering	Data Offering IDs the consumer subscribes to

Table 5- Consumer properties

- **Owner module**

The actual Owner of the Data sources provided by Marketplace , Data Space or.. service instance that offers available DataOfferings. An owner is described through **core:Owner** class. At this stage, each owner will have a name (schema:name) and id (core:ownerID) as shown in Table 4. More information about the provider can be added in the future.

Property Name	Data Types	Description
core:ownerId	string	Owner id
ownerDescription	string	A description of the provider
ownerName	string	Name of the provider
:sourceOrganization	schema:Organization	A provider's organization

Table 6 - Owner properties

- **DataMarket Module**

Information on the connected Data Marketplace

Property Name	Data Types	Description
core:dataMarketId	string	Data Market id
dataMarketDescription	string	A description of the Data Marketplace
dataMarketName	string	Name of the Data Marketplace

dataMarketNode	string	Info of Data Market Node

Table 7 – Market Module

- **DataOfferingQuery Module**

The query module presents an abstraction for a query for discovery DataOfferings. It is used to describes the properties of Offerings a consumer is interested in (Offering type, category of data, price, license, ...). Through the backplane, the consumer discovers the offerings of interest by providing a (dataOffering) query. In this schema, the query information can be described through the **core:DataOfferingQuery** class. For example, a consumer can register a query by providing a description of the desired resources and also define the maximum price, the desired license types, time and extensions, etc. All properties of core:DataOffering may also apply to core:DataOfferingQuery.

The properties specific to Query are shown in the Table 5 below.

Property Name	Data Types	Description
queryName	string	Name of the query
core:queryId	string	Query id
queryDescription	string	A description of the query
core:hasFilters	.core:QueryFilters	Gives a detailed description of the transported payload data between consumer and provider for the offerings that the user wants to query.

Table 8- Query properties

- **Subscription**

Agreement to access the **Resource(s)** of a single Data**Offering**. This comprises:

- a **Consumer**'s willingness to access the Data**Offering** (he checked **License**, service level, rating, description, ...)
- the **Consumer's** consent to pay for the access to the **Resources** (according to the specified **Price**), if applicable

4.4 Data Offering module

Data-Offering

i3-market enables Providers to offer or trade access to datasets via the backplane. A Data-Offering is defined by a Data-Offering description, which describes via metadata a set of Resources offered via the i3-market backplane. It typically encompasses a set of related Information. A Data-Offering description provides a semantic description of the datasets provided to a Consumer once the Data-Offering is registered. The description also entails context and meta information about the Distribution, including information to, the Pricing for accessing the Resource(s), the License of the Information provided, Contractual Parameters and service description as URL for data access).

As illustrated in Figure a, the Data Offering module represents the initial conceptualisation which is built around the DataOffering and its metadata. All the core concepts of this module are defined as follows:

A provider registers its offerings on the marketplace by providing an offering description. An offering description is an instance of the Data Offering class, itself a subclass of schema:Offer. It contains the information about the Data Assets, data service, categories of data assets, sub classes components of catalogues and resources, data services, categories of the offering (:category). All relevant communication metadata are provided on how the offering can be accessed through the data service and service extension descriptions.

Details of all classes and their properties in the Offering module are presented in Sections Below, DataOffering Description.

4.4.1 "Data Offering Description"

To Describe the Data Assets, Contractual Parameters, Rights, Licenses, Pricing Models, Data Service, Endpoints, Format of data, domain annotations, related Actors, and other information that describe the datasets we defined shared "Data Offering Descriptions".

We use W3c Data Catalog Vocabulary (DCAT) - Version 2 vocabulary related to parts as: Dataset , Distribution , DataService used in data offering description

https://w3c.github.io/dxwg/dcat/

(Check the description and specifications of DCAT for the all information related to part of vocabulary related to parts : Dataset , Distribution , DataService used in data offering description ; in progress at w3c v3 https://www.w3.org/TR/vocab-dcat-3/)

DCAT enables a publisher to describe datasets and data services in a catalog using a standard model and vocabulary that facilitates the consumption and aggregation of metadata from multiple catalogs. This can increase the discoverability of datasets and data services. It also makes it possible to have a decentralized approach to publishing data catalogs and makes federated search for datasets across catalogs in multiple sites possible using the same query mechanism and structure.

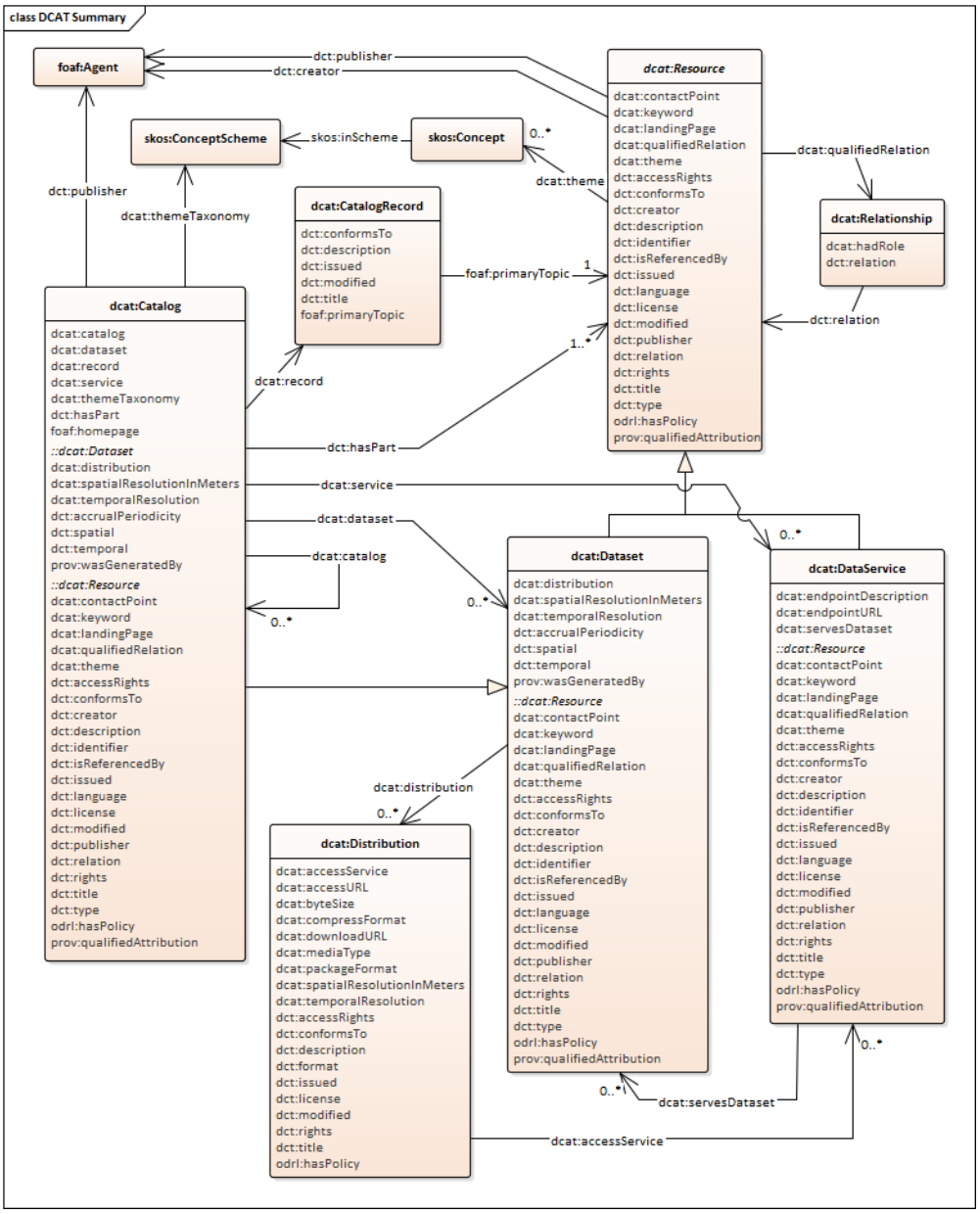


Figure 7 Original DCAT Diagram Overview

- **DataOffering Class**

Definition: High Level Class in i3-market core model that introduce the Data Offering description of Datasets resources

Property	Data Types	Description
hasDataset	Dataset	links the Data Offering in core to a DCAT (-AP) Dataset

description	String	contains a free-text account of the **DataOffering**
provider	Provider	refers to an entity (organisation) responsible for making the Data Offering available
owner	Owner	refers to an entity that have source ownership of the Data
title	String	contains a name given to the Catalogue
license	LicenseDocument	This property refers to the license under which the Catalogue can be used or reused.
category	ConceptScheme	refers to a knowledge organization system used to classify the Data Offering categories for Datasets. The high-level Categories Terms URI to use are defined in scheme file DataOfferingCategory.ttl
themeTaxonomy	skos:ConceptScheme	This property refers to a knowledge organization system used to classify the DataOffering's Datasets.
isActive	string	flag to set if the DataOffering is activated/available by the provider to be check/search by e.g. Consumer
core:hasPricingModel	pricingmodel:PricingModel	the pricing model for the Data Offering
core:contractParameters	core:ContractParameters	Some Specific Contract Parameters related to Data Offering

Table 9 – Data Offering Propreties

- **(DCAT) Dataset Class**

Definition: A collection of data, published or curated by a single agent, and available for access or download in one or more representations.

Property	Data Types	Description
description	String	contains a free-text account of the dataset
title	String	contains a name given to the dataset
keyword	String	contains a keyword or tag describing the Dataset
core:datasetInformation	core:DatasetInformation	Some specific information annotations of datasets metadata information types which represent attributes of observations, measurements , fields, etc.. in the dataset.
core:datasetRecord	core:DatasetRecord	In case data records types which represent attributes of fields,.. in the dataset.
datasetDistribution	Distribution	links the Dataset to an available Distribution
geographicalCoverage	Location	refers to a geographical area covered by the Dataset.
temporalCoverage	PeriodOfTime	refers to a temporal period that the Dataset covers
category	Concept	refers to a category of the Dataset. A Dataset may be associated with multiple categories.
accessRights	RightsStatement	refers to information that indicates whether the Dataset is open data, has access restrictions or is not public.
Frequency	Frequency	refers to the frequency at which the Dataset is updated
Documentation	Documentation	refers to a page or document about this Dataset
hasVersion	Dataset	refers to a related Dataset that is a version, edition, or adaptation of the described Dataset
creator	Agent	refers to the entity primarily responsible for producing the dataset
dcat:theme	skos:Concept	This property refers to a category of the Dataset. A Dataset may be associated with multiple themes.

Table 10 DCAT Dataset main Properties

- **DatasetInformation Class**

Definition: extended specific annotations to add extra information related to a dataset. This information is used to give the possibility to providers to describe with more granularity the source and types of data in datasets and annotations related to specific domains.

Property	Data Types	Description
core:measurementType	String	The data types which represent attributes of observations, measurements, in the dataset.
core:measurementChannelType	String	The data measurement Channel types in the dataset.
core:sensorID	String	Sensor ID
core:deviceID	String	Device ID
core:cppType	String	Cyber Physical Systems cpp Type
core:sensorType	String	Sensor Type

Table 11 DatasetInformation class main properties

- **(DCAT) Distribution Class**

Definition: A specific representation of a dataset

Property	Data Types	Description
description	String	contains a free-text account of the distribution
accessURL	Resource	contains a URL that gives access to a Distribution of the Dataset
availability	Concept	indicates how long it is planned to keep the Distribution of the Dataset available
format	MediaTypeOrExtent	refers to to the file format of the Distribution.
downloadType	String	Download Type (if means Frequency as 'Stream' or 'Bulk' dataset download)
license	LicenseDocument	Refer to the licence under which the Distribution is made available.
accessService	DataService	refers to a data service that gives access to the distribution of the dataset
byteSize	Double	size of a Distribution in bytes.
Documentation	Documentation	refers to a page or document about this distribution

downloadURL	Resource	URL that is a direct link to a downloadable file in a given format.
releaseDate	DateTime	contains the date of formal issuance (e.g., publication) of the Distribution.

Table 12 DCAT Distribution main properties

- **(DCAT) DataService Class**

Definition: A collection of operations that provides access to one or more datasets or data processing functions.

Property	Data Types	Description
description	String	contains a free-text account of the data service
endpointURL	Resource	The root location or primary endpoint of the service (an IRI).
title	String	contains a name given to the data service
servesDataset	Dataset	refers to a collection of data that this data service can distribute.
license	LicenseDocument	Refer to the licence under which the Data Service is made available.
accessRights	RightsStatement	include information regarding access or restrictions based on privacy, security, or other policies.
serviceID	String	Service ID
serviceSpecs	ServiceSpecs	Service Specification reference to ITSO extra service model specifications

Table 13 – DCAT DataService main Properties

- **ContractParameters Class**

Definition: A collection of parameters that provides information about use and scope of the DataOffering/Dataset.

Property Name	Data Types	Description

core:interestOfProvider	Literal	This property is used to identify the interest of the data owner. The following possibilities exist: - Free Sharing -Quotation -Selling of data (e.g. just earning money by selling the data, no specific feedback on these data by a data consumer expected)
core:interestDescription	Literal	Data provider can specify which sort of quotation he wants exactly, e.g., quotation for maintenance service or quotation for optimization of production
core:hasGoverningJurisdiction	Literal	Jurisdiction
core:purpose	Literal	Purpose for the use of the Dataset
core:purposeDescription	Literal	Description of the Purpose for the use of the Dataset
core:hasIntendedUse	core:IntendedUse	To Intended Use class/properties
core:hasLicenseGrant	core:LicenseGrant	To License Grant class/properties

Table 14- ContractParameters properties

- **LicenseGrant Class**

Definition: Definition of type of license is associated with the Data asset

Property Name	Data/Object Types	Description
core:copyData	Literal	License to copy data (True or False)
core:transferable	Literal	Transferable (True or False)
core:exclusiveness	Literal	License of exclusiveness (True or False)

core:revocable	Literal	License revocable (True or False)

Table 15- LicenseGrant properties

- **IntendedUse Class**

Definition: what the data provider allows the consumer to be the intended use of the data assets

Property Name	Data/Object Types	Description
core:processData	Literal	Process Data (True or False)
core:shareDataWithThirdParty	Literal	Share Data with Third Party (True or False)
core:editData	Literal	Edit Data (True or False)

Table 16 - IntendedUse properties

Table 17 -Preliminary example for a Metadata Descriptions

```
##
PREFIX rdfs: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdf: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX dct:  <http://purl.org/dc/terms/>
```

```
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX pricingmodel: <http://i3-market.eu/backplane/pricingmode/>
PREFIX core: <http://i3-market.eu/backplane/core/>
PREFIX : <http://i3-market.org/resource/>

:Mindsphere
  a core:Provider ;
  rdfs:label "Mindsphere"@en ;
  core:dataOffering ex:DataOffering-1  .
```

Preliminary example for a DataOffering Description

```
##
PREFIX rdfs: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdf: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX dct:  <http://purl.org/dc/terms/>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX pricingmodel: <http://i3-market.eu/backplane/pricingmode/>
PREFIX core: <http://i3-market.eu/backplane/core/>
PREFIX : <http://i3-market.org/resource/>

## DataOffering Description example

:DataOffering-1 a core:DataOffering ;
core:dataOfferingId "DataOffering-1" ;
core:provider ex:Mindsphere ;
core:dataOfferingDescription "This is the description of the Data Offering"@en ;
core:dataOfferingTitle "Manufacturing maintenance dataset"@en ;
core:category http://i3.market.eu/auth/dataCatagory/Manufacturing ;
core:isActive "Yes" ;
```

```
## Pricing Model Description example
core:hasPricingModel [ :Pricing-1   a pricingmodel:PricingModel
  pricingmodel:currency "euroToken" ;
  pricingmodel:hasPaymentType [
    a pricingmodel:PaymentOnSubscription ;
    pricingmodel:timeDuration “2 weeks”
    pricingmodel:repeat pricingmodel:WEEK ;
    pricingmodel:hasSubscriptionPrice "2 euro" ;
    pricingmodel:fromValue "2011-12-05"^^xsd:date
    pricingmodel:toValue "2011-12-19"^^xsd:date
  ] ;
] ;
.

## Contract Parameters Description example
core:contractParameters [   a core:ContractParameters
  core:interestOfProvider "Quotation" ;
  core:interestDescription "Request of Quotation for maintenance" ;
  core:hasGoverningJurisdiction "EU JURISDICTION" ;
  core:purpose "yyyyyyy" ;
  core:purposeDescription "yyyyyyy" ;
  core:hasIntendedUse  [
    a core:IntendedUse ;
      core:processData "TRUE" ;
      core:shareDataWithThirdParty "TRUE" ;
      core:editData "TRUE" ;.
  ] ;
  core:hasLicenseGrant  [
    a core:LicenseGrant  ;
      core:copyData "TRUE" ;
      core:transferable "TRUE" ;
```

```
      core:exclusiveness "TRUE" ;
      core:revocable "TRUE" ;.
  ] ;
] ;
.

## Data assets Description example
core:hasDataset [ :MindsphereDataset a dcat:Dataset ;
    dct:description "This is the description of the datasets offered"@en ;
    dct:title "Manufacturing maintenance dataset"@en ;
    dcat:keyword "Manufacturing"@en, "Maintenance"@en, "Machinery"@en ;
    dct:creator :Manufacturing-employee-001 ;
    dct:issued "2011-12-05"^^xsd:date ;
    dct:modified "2011-12-15"^^xsd:date ;
    dcat:theme skos:Air_conditioning_equipment  ;
    dcat:theme skos:Maintenance-Machine  ;
    core:datasetInformation [
       core:measurementType :Sleep_count_micro_awakenings ;
       core:measurementChannelType  "stream-series"
       core:sensorID  "000001" ;
       core:deviceID  "00000A" ;
       core:cppType   "building"  ;
       core:sensorType  "Monitor" .
     ];

    core:datasetRecord [
        ] ;
    dct:temporal <http://reference.data.gov.uk/id/quarter/2006-Q1> ;
    dcat:temporalResolution "P1D"^^xsd:duration ;
    dct:spatial <http://sws.geonames.org/6695072/> ;
    dct:language <http://id.loc.gov/vocabulary/iso639-1/en>  ;
```

```
    dct:accrualPeriodicity <http://purl.org/linked-data/sdmx/2009/code#freq-
W>  ;
    dcat:distribution [
      rdf:type dcat:Distribution ;
      dct:description "This is the description of the specific distribution
of data that can be downloaded"@en ;
      dct:license <http://id.loc.gov/vocabulary/iso639-1> ;
      dct:conformsTo <https://www.w3.org/TR/xml-schema/> ;
      dct:title "xml representation of the data"@en ;
      dcat:mediaType application/json ;
      dcat:packageFormat jsonfile ;
      dcat:accessService [
        rdf:type dcat:DataService ;
        dct:conformsTo <http://example.org/apidef/ > ;
        dcat:endpointDescription  <http://example.org/api/figure-006/params>
;
        dcat:endpointURL <http://example.org/api/figure-006> ;
        dcat:servesDataset :dataset-001 ;
        core:serviceSpecs :Spec-1. ] ;
    ] ;
 ] ;
.
```

.

4.5 Controlled vocabularies suggested to be used for particular annotations

In the table below, a number of properties are listed with controlled vocabularies that should be used for the listed properties. The declaration of the following controlled vocabularies as high recommendation (in DCAT_AP specifications are listed as mandatory) ensures a minimum level of interoperability.

Property URI	Used for Class	Vocabulary name	Vocabulary URI	Usage note
dcat:mediaType	Distribution	IANA Media Types [5]	http://www.iana.org/assignments/media-types/media-types.xhtml	

dcat:theme	Dataset	Dataset Theme Vocabulary	http://publications.europa.eu/resource/authority/data-theme	The values to be used for this property are the URIs of the concepts in the vocabulary.
dcat:themeTaxonomy	Catalogue	Dataset Theme Vocabulary	http://publications.europa.eu/resource/dataset/data-theme	The value to be used for this property is the URI of the vocabulary itself, i.e. the concept scheme, not the URIs of the concepts in the vocabulary.
dct:accrualPeriodicity	Dataset	EU Vocabularies Frequency Named Authority List [6]	http://publications.europa.eu/resource/authority/frequency	
dct:format	Distribution	EU Vocabularies File Type Named Authority List [7]	http://publications.europa.eu/resource/authority/file-type	
dct:language	Catalogue, Dataset, Catalogue Record, Distribution	EU Vocabularies Languages Named Authority List [8]	http://publications.europa.eu/resource/authority/language	
dct:publisher	Catalogue, Dataset	EU Vocabularies Corporate bodies Named Authority List [9]	http://publications.europa.eu/resource/authority/corporate-body	The Corporate bodies NAL must be used for European institutions and a small set of international organisations. In case of other types of organisations, national, regional or local vocabularies should be used.
dct:spatial	Catalogue, Dataset	EU Vocabularies Continents Named Authority List [10], EU Vocabularies Countries Named Authority List [11], EU Vocabularies Places Named Authority List[12], Geonames	http://publications.europa.eu/resource/authority/continent/, http://publications.europa.eu/resource/authority/country, http://publications.europa.eu/resource/authority/place/, http://sws.geonames.org/	The EU Vocabularies Name Authority Lists must be used for continents, countries and places that are in those lists; if a particular location is not in one of the mentioned Named Authority Lists, Geonames URIs must be used.
adms:status	Distribution	ADMS status vocabulary	http://purl.org/adms/status/	The list of terms in the ADMS status vocabulary is included in the ADMS specification [13]
dct:type	Agent	ADMS publisher type vocabulary	http://purl.org/adms/publishertype/	The list of terms in the ADMS publisher type vocabulary is included in the ADMS specification

dct:type	Licence Document	ADMS licence type vocabulary	http://purl.org/adms/licencetype/	The list of terms in the ADMS licence type vocabulary is included in the ADMS specification
dcatap:availability	Distribution	Distribution availability vocabulary	http://data.europa.eu/r5r/availability/	The list of terms for the avalability levels of a dataset distribution in the DCAT-AP specification.

4.6 Pricing/Cost Model

Here we present the initial representation of a Pricing-Model to describe the pricing information attached to the data assets related to legacy information of pricing specification in the Marketplaces.

4.6.1 Pricing Model

Pricing models associated to DataOffering Class.

- **Base Class pricingmodel:PricingModel**

Property Name	Data Types	Description
currency	xyz	currency type
:description	string	A description
:name	string	Name
:hasPaymentOnSubscription	pricingmodel:PaymentOnSubscription	PaymentOnSubscription
:hasPaymentOnAPI	pricingmodel:PaymentOnAPI	PaymentOnAPI
:hasPaymentOnPlan	pricingmodel:PaymentOnPlan	PaymentOnPlan

:hasPaymentOnUnit	pricingmodel:PaymentOnUnit	PaymentOnUnit
:hasPaymentOnSize	pricingmodel:PaymentOnSize	PaymentOnSize
:hasFreePrice	pricingmodel:FreePrice	FreePrice

Table 18- PricingModel basic properties

For Payment categories from Marketplace terms we can have like: pricingmodel:PaymentOnPlan , pricingmodel:PaymentOnAPI , pricingmodel:PaymentOnUnit , pricingmodel:PaymentOnSize , pricingmodel:PaymentOnSubscriptiOn , pricingmodel:FreePrice

- **PaymentOnPlan Class**

Payment type Class pricingmodel:PaymentOnPlan

Property Name	Data Types	Description
:hasPlanPrice	string	plan price
:description	string	A description
:name	string	Name
: :planDuration	string	Plan Duration

Table 19- PaymentOnPlan basic properties

- **PaymentOnAPI Class**

Payment type Class pricingmodel:PaymentOnAPI

Property Name	Data Types	Description
:hasAPIPrice	string	basic price
:description	string	A description

:name	string	Name
pricingmodel:numberObject		Number of Objects moved via API

Table 20 - PaymentOnAPI basic properties

- **PaymentOnUnit Class**

Payment type Class pricingmodel:PaymentOnUnit

Property Name	Data Types	Description
:hasUnitPrice	string	basic price
:description	string	A description
:name	string	Name
pricingmodel:dataUnit		Data unit type
:unitID	string	:unit ID

Table 21 - PaymentOnUnit basic properties

- **PaymentOnSize Class**

Payment type Class pricingmodel:PaymentOnSize

Property Name	Data Types	Description
:hasSizePrice	string	basic price
:description	string	A description
:name	string	Name
:dataSize		Size of Data

Table 22 - PaymentOnSize basic properties

- **PaymentOnSubscriptiOn Class**

Payment type Class pricingmodel:PaymentOnSubscriptiOn

Property Name	Data Types	Description
:hasSubscriptionPrice	string	subscription price
:description	string	A description
:name	string	Name
:timeDuration	time	Subscriptiond duration
:fromValue	date time	Subscription validity starting point.
:toValue	date time	Subscription validity ending point.
:repeat	pricingmodel:RepeatBy	In case the subscription is repeatable.

Table 23 - PaymentOnSubscription basic properties

- **FreePrice Class**

Payment type Class pricingmodel:FreePrice

Property Name	Data Types	Description
:hasFreePrice	string	Free option

Table 24- FreePrice basic properties

4.7 Domain Categorization / Taxonomies for domain specific annotations of datasets

4.7.1 Property: core: category and dcat:theme

The dcat:theme is used to give annotation , information about the domain categorization of the datasets. In i3-Market we use the themes as sub-categories to give more granularity in defining the domain annotations. In DCAT 1 the domain of **dcat:theme** was **dcat:Dataset**, which limited use of this property in other contexts. The domain has been relaxed in later revisions.

We added also a upper level property for a Data Offering to annotate directly the High Level type of Category the data offering belong to as core :category

RDF Property:	**dcat:theme**
Definition:	A category of the resource. A resource can have multiple themes.
Sub-property of:	dct:subject
Range:	skos:Concept
Usage note:	The set of skos:Concepts used to categorize the resources are organized in a skos:ConceptScheme describing all the categories and their relations in the catalog.

Class: Concept Scheme

RDF Class:	**skos:ConceptScheme**
Definition:	A knowledge organization system (KOS) used to represent themes/categories of datasets in the catalog.

Class: Concept

RDF Class:	**skos:Concept**
Definition:	A category or a theme used to describe datasets in the catalog.
Usage note:	It is recommended to use either skos:inScheme or skos:topConceptOf on every skos:Concept used to classify datasets to link it to the concept scheme it belongs to. This concept scheme is typically associated with the catalog using dcat:themeTaxonomy.

We are using **skos:ConceptScheme** via **skos:Concept** to create taxonomies to annotate high level types of annotations for domains themes/categories classifications.

Example of Categories Terms as in i3-Market DataCategory.ttl schema

```
Data Categories

<http://i3.market.eu/auth/dataCatagory/Manufacturing>
                skos:prefLabel "Manufacturing"@en.
<http://i3.market.eu/auth/dataCatagory/Automotive>
                skos:prefLabel "Automotive"@en.
<http://i3.market.eu/auth/dataCatagory/Wellbeing>
                skos:prefLabel "Wellbeing"@en.
<http://i3.market.eu/auth/dataCatagory/Agriculture>
                skos:prefLabel "Agriculture,   fisheries,   forestry   and
food"@en.
<http://i3.market.eu/auth/dataCatagory/Culture>
                skos:prefLabel "Culture and sport"@en.
<http://i3.market.eu/auth/dataCatagory/Economy>
                skos:prefLabel "Economy and finance"@en.
<http://i3.market.eu/auth/dataCatagory/Education>
                skos:prefLabel "Education"@en.
<http://i3.market.eu/auth/dataCatagory/Energy>
                skos:prefLabel "Energy"@en.
<http://i3.market.eu/auth/dataCatagory/Environment>
                skos:prefLabel "Environment"@en.
<http://i3.market.eu/auth/dataCatagory/Government>
                skos:prefLabel "Government and public sector"@en.
<http://i3.market.eu/auth/dataCatagory/Health>
                skos:prefLabel "Health"@en.
<http://i3.market.eu/auth/dataCatagory/International>
                skos:prefLabel "International issues"@en.
<http://i3.market.eu/auth/dataCatagory/Justice>
                skos:prefLabel "Justice, legal system and public safety"@en.
<http://i3.market.eu/auth/dataCatagory/Regions>
                skos:prefLabel "Regions and cities"@en.
<http://i3.market.eu/auth/dataCatagory/society>
                skos:prefLabel  "Population and society"@en.
<http://i3.market.eu/auth/dataCatagory/Science>
                skos:prefLabel "Science and technology"@en.
<http://i3.market.eu/auth/dataCatagory/Transport>
                skos:prefLabel "Transport"@en.
```

4.8 Extra IT Services Ontology vocabulary

Importing of Ontology for IT Services delivered via the Cloud

For the addition of particular terms that could be useful for extending the properties of the DataService Class and add extra concepts to describe some features not in DCAT, we are reusing some part of the OWL Ontology for IT Services delivered via the Cloud. Developed by Karuna P Joshi, CSEE Department, UMBC. https://ebiquity.umbc.edu/ontologies/itso/1.0/itso.owl.

https://ebiquity.umbc.edu/resource/html/id/325/IT-Service-Ontology

An overview of the Classes in itso.owl that are available for extra data service metadata description are like:

- Compatible_Applications
- Consumer_Negotiation
- Contract_Negotiation
- Dependent_Service
- Functional_Specs
- Human_Agent_Specs
- Latency
- Operating_System
- OWL:S-Composite Service
- Performance
- Provider_Negotiation
- Quality_of_Service
- Reliability
- Request_For_Service
- Resolution_Time
- Response_Time
- Security_Policy
- Security_policy
- Service Contract
- Service_Certification
- Service_Compliance
- Service_Level_Agreement
- Service_Search_Engine
- Software_Component
- Specification
- Technical_Specs
- Throughput

4.9 W3c Verifiable Credentials Data Model

For representing the verifiable credentials the backplane follows the W3c Verifiable Credentials Data Model 1.0.

Credentials are a part of our daily lives; driver's licenses are used to assert that we are capable of operating a motor vehicle, university degrees can be used to assert our level of education, and government-issued passports enable us to travel between countries. These credentials provide benefits to us when used in the physical world, but their use on the Web continues to be elusive.

Currently it is difficult to express education qualifications, healthcare data, financial account details, and other sorts of third-party verified machine-readable personal information on the Web. The difficulty of expressing digital credentials on the Web makes it challenging to receive the same benefits through the Web that physical credentials provide us in the physical world.

This specification provides a standard way to express credentials on the Web in a way that is cryptographically secure, privacy respecting, and machine-verifiable.

Also in i3-Market the SSI& IAM Subsystems use DIDs that follow the W3c Decentralized Identifiers (DIDs) v1.0 specifications.

Decentralized identifiers (DIDs) are a new type of identifier that enables verifiable, decentralized digital identity. A DID refers to any subject (e.g., a person, organization, thing, data model, abstract entity, etc.) as determined by the controller of the DID. In contrast to typical, federated identifiers, DIDs have been designed so that they may be decoupled from centralized registries, identity providers, and certificate authorities. Specifically, while other parties might be used to help enable the discovery of information related to a DID, the design enables the controller of a DID to prove control over it without requiring permission from any other party. DIDs are URIs that associate a DID subject with a DID document allowing trustable interactions associated with that subject.

4.10 Online i3-Market Semantic Model repository and community management

The results will be shared not only with Consortium Partners but also with stakeholders and community. As part of open-source assets, the data models, documentations and files used in the i3-Market project are made available, like:

- The i3-Market data Pack is the set of files, schemas and metadata model diagrams that represent the way the i3-Market semantics is organised and structured, it also contains the metadata in two different formats, e.g. .ttl, .**owl**.
- The i3-Market Semantic Model info is the documentation that describes in detail all the taxonomies and vocabularies from needed domains used in i3-Market and that describes and represent all the relationships between them to build the graph representation of the i3-Market semantic model.
- The Support repo is the mechanism for how the data model is maintained following the interoperability requirements in i3-Market, if you want to contribute or have any suggestion for improving the semantic models, visit this section.
- The Models Files will be shared in Git repository with releases versions where each section contains the online machine-readable files in OWL and other format for online accessibility, the files are maintained and updated regularly to keep the latest version of the models files up to date.

5 Definitions for Semantic description of Data Offerings in relation to Creation Template

DataOffering

{

"provider":

RDF Property:	core:provider
Definition:	Provider of the DataOffering
Range:	Provider Identifier: xsd:string
Usage note:	Should be the identifier of the Provider in i3-Market system Verification should be done with registered providers. All other providers shall be rejected. Return an error message in case an unregistered provider is specified.
See also:	Maybe connected with the IDs in Identity manager. **As the actual registration is by the Marketplaces/DataSpaces they have the knowledge and responsability to have the name/identity of the Providers (that have knowledge of the Owners) they would know who are the providers..**

"owner":

RDF Property:	core:owner
Definition:	Owner of the DataOffering
Range:	Owner Identifier: xsd:string
Usage note:	Should be the identifier of the Owner in i3-Market system. Owners are not registered in i3-MARKET. Optional parameter. Not to be verified.
See also:	Maybe connected with the IDs in Identity manager

"dataOfferingTitle":

RDF Property:	core:dataOfferingTitle
Definition:	The title of the DataOffering
Range:	xsd:string
Usage note:	A name to identify the dataoffering. A few words only, that summarize the offering.
See also:	

"dataOfferingDescription":

RDF Property:	core:dataOfferingDescription
Definition:	A description of the DataOffering
Range:	xsd:string
Usage note:	Used to have descrition text to describe what the data offering is about. This can be a long block of text. At least 1000 chars shall be reserved for this.
See also:	

"category":

RDF Property:	core:category
Definition:	A category to have high level classification of domain for the Data Offering
Range:	xsd:anyURI
Usage note:	Use the Categories naming schemaa defined for high level categories as URIs: Categories should only be added through extending the categories list. This is done by the community. The parameter should be checked against this list. If it does not match, return an error.
See also:	Categories in Table below Data Categories (as per definitions in gitlab file: https://gitlab.com/i3-market/code/data-models/-/blob/master/Version-1/DataOfferingCategory.ttl)

	<http://i3.market.eu/auth/dataCatagory/Manufacturing> <http://i3.market.eu/auth/dataCatagory/Automotive> <http://i3.market.eu/auth/dataCatagory/Wellbeing> <http://i3.market.eu/auth/dataCatagory/Agriculture> <http://i3.market.eu/auth/dataCatagory/Culture> <http://i3.market.eu/auth/dataCatagory/Economy> <http://i3.market.eu/auth/dataCatagory/Education> <http://i3.market.eu/auth/dataCatagory/Energy> <http://i3.market.eu/auth/dataCatagory/Environment> <http://i3.market.eu/auth/dataCatagory/Government> <http://i3.market.eu/auth/dataCatagory/Health> <http://i3.market.eu/auth/dataCatagory/International> <http://i3.market.eu/auth/dataCatagory/Justice> <http://i3.market.eu/auth/dataCatagory/Regions> <http://i3.market.eu/auth/dataCatagory/Society> <http://i3.market.eu/auth/dataCatagory/Science> <http://i3.market.eu/auth/dataCatagory/Transport> see also file DataOfferingCategory.ttl

"status":

RDF Property:	core:status
Definition:	To define if the dataoffering is “active” and so can be displayed and shared for the use of consumers (e.g.if not this is private only for providers/admin view and not on available to other views)

Range:	xsd:string
Usage note:	Possible values: “Active”: The offer is active and visible to everyone. “Inactive”: The offer is not visible, but still exists and can be activated again. “ToBeDeleted”: Data offer is still available and visible, but will be deleted once the last contract on this offer expired. No new purchases allowed on it. “Deleted”: The offer is not visible and cannot be activated again. No longer available for consumers or providers.
Note:	Rename this field to “**Status**”. Possible values: “Active”: The offer is active and visible to everyone. “Inactive”: The offer is not visible, but still exists and can be activated again. “ToBeDeleted”: Data offer is still available and visible, but will be deleted once the last contract on this offer expired. No new purchases allowed on it. “Deleted”: The offer is not visible and cannot be activated again. No longer available for consumers or providers.

"dataOfferingExpirationTime":

RDF Property:	core:dataOfferingExpirationTime
Definition:	Expiration Time of dataOffering in case
Range:	Can be: xsd:dateTime
Usage note:	The dateTime data type is used to specify a date and a time. The dateTime is specified in the following form "YYYY-MM-DDThh:mm:ss" where: • YYYY indicates the year • MM indicates the month • DD indicates the day • T indicates the start of the required time section • hh indicates the hour • mm indicates the minute • ss indicates the second **Note:** All components are required! The following is an example of a dateTime declaration in a schema: “2002-05-30T09:00:00”
See also:	

"versionNotes":

RDF Property:	`adms:versionNotes`
Definition:	A description of changes between this version and the previous version of the resource [*VOCAB-ADMS*].
Range:	xsd:string
Usage note:	In case of backward compatibility issues with the previous version of the resource, a textual description of them *SHOULD* be specified by using this property.
See also:	§ 6.4.26 Property: current version, § 6.4.24 Property: has version, § 6.4.28 Property: is replaced by, § 6.4.25 Property: is version of, § 6.4.23 Property: previous version, § 6.4.7 Property: release date, § 6.4.27 Property: replaces, § 6.4.31 Property: status, § 6.4.30 Property: version notes.

"previousVersion":

RDF Property:	`dcat:previousVersion`
Definition:	The previous version of a resource in a lineage [*PAV*].
Range:	xsd:anyURI
Usage note:	This property is meant to be used to specify a version chain, consisting of snapshots of a resource. The notion of version used by this property is limited to versions resulting from revisions occurring to a resource as part of its life-cycle. One of the typical cases here is representing the history of the versions of a dataset that have been released over time.
See also:	§ 6.4.26 Property: current version, § 6.4.24 Property: has version, § 6.4.28 Property: is replaced by, § 6.4.25 Property: is version of, § 6.4.23 Property: previous version, § 6.4.7 Property: release date, § 6.4.27 Property: replaces, § 6.4.31 Property: status, § 6.4.30 Property: version notes.

"replaces":

RDF Property:	`dcterms:replaces`
Definition:	A related resource that is supplanted, displaced, or superseded by the described resource [*DCTERMS*].

Range:	xsd:anyURI
Usage note:	resource replaced
See also:	§ 6.4.26 Property: current version, § 6.4.24 Property: has version, § 6.4.28 Property: is replaced by, § 6.4.25 Property: is version of, § 6.4.23 Property: previous version, § 6.4.7 Property: release date, § 6.4.27 Property: replaces, § 6.4.31 Property: status, § 6.4.30 Property: version notes.

"previousVersion":

RDF Property:	`dcat:previousVersion`
Definition:	The previous version of a resource in a lineage [*PAV*].
Range:	xsd:anyURI
Usage note:	This property is meant to be used to specify a version chain, consisting of snapshots of a resource. The notion of version used by this property is limited to versions resulting from revisions occurring to a resource as part of its life-cycle. One of the typical cases here is representing the history of the versions of a dataset that have been released over time.
See also:	§ 6.4.26 Property: current version, § 6.4.24 Property: has version, § 6.4.28 Property: is replaced by, § 6.4.25 Property: is version of, § 6.4.23 Property: previous version, § 6.4.7 Property: release date, § 6.4.27 Property: replaces, § 6.4.31 Property: status, § 6.4.30 Property: version notes.

"contractParameters": [

{

"interestOfProvider":

RDF Property:	core:interestOfProvider

Definition:	This property is used to identify the interest of the data owner/provider related to the trading/sharing of their data assets. The following possibilities exist: • Free Sharing • Quotation • Selling of data (e.g. just earning money by selling the data, no specific feedback on these data by a data consumer expected
Range:	xsd:string
Usage note:	It could be simple notations like: Free Sharing -Quotation -Selling of data ; or we can decide to have specific definitions for our system
See also:	

"interestDescription":

RDF Property:	core:interestDescription
Definition:	Data provider can specify which sort of quotation he wants exactly, e.g., quotation for maintenance service or quotation for optimization of production
Range:	xsd:string
Usage note:	More text description of the interest of the data owner/provider related to the trading/sharing of their data assets. Example: “This data is shared only for the purpose of creating a quotation for maintenance for the production machines described in the data set. Any other use of this data is not permitted.”
Note:	

"hasGoverningJurisdiction":

RDF Property:	core:hasGoverningJurisdiction
Definition:	The file format of the distribution.
Range:	xsd:string (or xsd:anyURI)
Usage note:	Can be string naming like: GLOBAL US JURISDICTION EU JURISDICTION

	(or we use URIs to define the specific terms for jurisdictions ToDo: Define a list of jurisdictions, which are valid here.
See also:	

"purpose":

RDF Property:	core:purpose
Definition:	Purpose of the Agreement
Range:	xsd:string
Usage note:	Short label for the purpose In case we could have specific terminology for define list of @purpose@ terms
Note:	This parameter is part of the contractual parameters. Ask contract partners, what this is for (Susanne).

"purposeDescription":

RDF Property:	core:purposeDescription
Definition:	In case full text description of describing the reasons behind the creation of the Agreement
Range:	xsd:string
Usage note:	text description
Note:	This parameter is part of the contractual parameters. Ask contract partners, what this is for (Susanne)

"hasIntendedUse": [

{

"processData": "true OR false",

RDF Property:	core:processData
Definition:	If consumer allowed to process data
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”
Note:	Part of contractual parameters. Ask contract partners, what this is for. Make this parameter to type Boolean.

"shareDataWithThirdParty": "true OR false",

RDF Property:	core:shareDataWithThirdParty
Definition:	If consumer allowed to share data with third parties
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”
Note:	Part of contractual parameters. Ask contract partners, what this is for. Make this parameter to type Boolean.

"editData": "true OR false"

RDF Property:	core:editData
Definition:	If consumer allowed to edit the Data
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”
Note:	Part of contractual parameters. Ask contract partners, what this is for. Make this parameter to type Boolean.

}],

"hasLicenseGrant": [

{

"copyData": "true OR false",

RDF Property:	core:copyData
Definition:	If licence grant to copy data
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”
Note:	Part of contractual parameters. Ask contract partners, what this is for.

"transferable": "true OR false",

RDF Property:	core:transferable
Definition:	If licence is transferable
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”
See also:	

"exclusiveness": "true OR false",

RDF Property:	core:exclusiveness
Definition:	If licence grant exclusiveness
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”
See also:	

"revocable": "true OR false"

RDF Property:	core:revocable
Definition:	If licence is revocable
Range:	xsd:boolean
Usage note:	The value space of xsd:boolean is true and false. Its lexical space accepts true, false, “TRUE” or “FALSE”

See also:	

}] }],

"hasPricingModel": [

{

"pricingModelName":

RDF Property:	pricingmodel:pricingModelName
Definition:	The name to define the legacy , by Marketplace, pricing model related to the data offering
Range:	xsd:string
Usage note:	Princing models are individually defined by marketplaces. No common pricing model will be defined for i3-MARKET. Maybe try to generalize existing pricing models.
See also:	

"basicPrice":

RDF Property:	pricingmodel:basicPrice
Definition:	The generic basic price for the traded data for basic cost of trade
Range:	xsd:double
Usage note:	Number related to price
See also:	

"currency":

RDF Property:	pricingmodel:currency
Definition:	The file format of the distribution.
Range:	xsd:string
Usage note:	Using ISO 4215 Currency Terminology
See also:	lis-ISO-4217-Currencyt_one.xml

	See XML file for 3 letter abbreviations. lis-ISO-4217-Currencyt_one.xml

"hasPaymentOnSubscription": [

{

"timeDuration":

RDF Property:	pricingmodel:timeDuration
Definition:	Time duration of subscription
Range:	xsd:anyURI
Usage note:	Or generic xsd:string text with labels for duration vocabulary or URIs with vocabulary like: e.g. "http://reference.data.gov.uk/def/intervals/Day" “http://reference.data.gov.uk/def/intervals/Hour" "http://reference.data.gov.uk/def/intervals/Minute" "http://reference.data.gov.uk/def/intervals/Month" "http://reference.data.gov.uk/def/intervals/Quarter" "http://reference.data.gov.uk/def/intervals/Second" Price is per timeDuration. E.g. if parameter is “Second” here, then the specified price is per second (€ / s)
See also:	Terms in intervals.rdf

"description":

RDF Property:	dcterms:description
Definition:	The description of payment on subscription
Range:	xsd:string

Usage note:	Text description
See also:	

"repeat":

RDF Property:	pricingmodel:repeat
Definition:	If subscription can be repeated define the frequency, e.g. Daily, Monthly,....
Range:	xsd:anyURI
Usage note:	We can use specific vocabulary e.g. in freq.ttl definitions like: http://purl.org/cld/freq/daily freq:monthly freq:weekly …….
See also:	See also freq.ttl or frequency.ttl.txt

"hasSubscriptionPrice":

RDF Property:	pricingmodel:hasSubscriptionPrice
Definition:	Price allocated to subscription payment type
Range:	xsd:double
Usage note:	Price
See also:	

}],

"hasPaymentOnPlan": [

{

There may be things like Basic Plan, Premium Plans, … Gives access to certain types of data. Difficult to implement in i3-MARKET.

Example for other usage: Deliver data only once a month or once every x period.
Optional parameter, does not have to be used.

Recommendation: Do NOT implement plans for now. Keep these parameters out, remove them.

"description":

RDF Property:	pricingmodel:planDescription
Definition:	The text description of plan
Range:	Xsd:string
Usage note:	Description text
See also:	

"planDuration":

RDF Property:	pricingmodel:planDuration
Definition:	The duration of the Plan
Range:	xsd:anyURI
Usage note:	Or generic xsd:string text with labels for duration vocabulary or URIs with vocabulary like: e.g. "http://reference.data.gov.uk/def/intervals/Day" “http://reference.data.gov.uk/def/intervals/Hour" "http://reference.data.gov.uk/def/intervals/Minute" "http://reference.data.gov.uk/def/intervals/Month" "http://reference.data.gov.uk/def/intervals/Quarter" "http://reference.data.gov.uk/def/intervals/Second"
See also:	Terms in intervals.rdf

"hasPlanPrice": "string"

RDF Property:	pricingmodel:hasPlanPrice
Definition:	The price of the Plan
Range:	xsd:double
Usage note:	Price
See also:	

}],

"hasPaymentOnAPI": [

{

"description":

RDF Property:	Dcterms:description
Definition:	The text description of payment type
Range:	Xsd:string
Usage note:	Description text
Note:	Optional. Useful for Agora.

"numberOfObject":

RDF Property:	pricingmodel:numberObject
Definition:	number of Objects for API Handle payments
Range:	Xsd:double
Usage note:	
Note:	Optional. Useful for Agora.

"hasAPIPrice": "string"

RDF Property:	pricingmodel:hasAPIPrice

Definition:	The price of the API payment type
Range:	xsd:double
Usage note:	Price
Note:	Optional. Useful for Agora.

}],

"hasPaymentOnUnit": [

{

"description":

RDF Property:	Dcterms:description
Definition:	The text description of payment type
Range:	Xsd:string
Usage note:	Description text Purchase a cluster of data. Sets of data. One cluster is a group of data sets.
See also:	

"dataUnit":

RDF Property:	pricingmodel:dataUnit
Definition:	Data Unit type handle by service
Range:	Xsd:string
Usage note:	Define what the unit resembles. Example: A predefined data set. A "Unit" of transaction as indicated in specification of the service method of exchange .
See also:	

"hasUnitPrice": "string"

RDF Property:	pricingmodel:hasUnitPrice
Definition:	The price of the by Unit payment type

Range:	xsd:double
Usage note:	Price per data unit
See also:	

}],

"hasPaymentOnSize": [

{

"description":

RDF Property:	Dcterms:description
Definition:	The text description of payment type
Range:	Xsd:string
Usage note:	Description text
See also:	

"dataSize":

RDF Property:	pricingmodel:dataSize
Definition:	The size of data exchanged for payment
Range:	typically typed as xsd:nonNegativeInteger.
Usage note:	The size in bytes can be approximated (as a non-negative integer) when the precise size is not known. While it is recommended that the size be given as an integer, alternative literals such as '1.5 MB' are sometimes used.
See also:	We can decide to use a specific vocabulary

"hasSizePrice": "string"

RDF Property:	pricingmodel:hasSizetPrice
Definition:	The price of the by Unit payment type
Range:	xsd:double
Usage note:	Price E.g. pay per Megabyte of data.

See also:	

}],

"hasFreePrice": [

{

"hasPriceFree": "FREE"

RDF Property:	pricingmodel:hasPriceFree
Definition:	The data is shared for free
Range:	Xsd:string
Usage note:	“FREE”. Data is for free, no payment needed.
See also:	We might use an URI as Pricingmodel:Free as unique term

}] }],

"hasDataset": [

{ (DataSet Description)

Description of the data sets contained. Note: This is not a description of the individual data items, but an overview.

"title":

RDF Property:	dcterms:title
Definition:	A name given to the dataset.
Range:	Xsd:string (rdfs:Literal)
Usage note:	Title
See also:	

"keyword":

RDF Property:	dcat:keyword
Definition:	A keyword or tag describing the resource.

Range:	Xsd:string (rdfs:Literal)
Usage note:	Text keywords, (in case we can decide to have a selection of terminologies to set as kaywords). One or more keywords describing the data.
See also:	To have multiple keywords You can have multiple instances of the property "keyword"

"description":

RDF Property:	dcterms:description
Definition:	A free-text account of the dataset.
Range:	Xsd:string (rdfs:Literal)
Usage note:	Description Text of Data Set
See also:	

~~**"creator":**~~ **Obsolete, to be deleted.**

~~RDF Property:~~ ~~dcterms:creator~~
~~Definition:~~ ~~The entity responsible for producing the resource.~~
~~Range:~~ ~~Xsd:string~~
~~Usage note:~~ ~~Name of Creator~~
~~See also:~~

"issued":

RDF property	dcterms:issued
Definition:	Date of formal issuance (e.g., publication) of the distribution.
Range:	encoded using the relevant ISO 8601 Date and Time compliant string [DATETIME] and typed using the appropriate XML Schema datatype [XMLSCHEMA11-2] (xsd:dateTime).
Usage note:	This property SHOULD be set using the first known date of issuance. The date of the initial publication of this dataset in i3-MARKET.

See also:	§ 6.4.7 Property: release date

"modified":

RDF Property:	dcterms:modified
Definition:	Most recent date on which the item was changed, updated or modified.
Range:	encoded using the relevant ISO 8601 Date and Time compliant string [DATETIME] and typed using the appropriate XML Schema datatype [XMLSCHEMA11-2] (xsd:dateTime).
Usage note:	The value of this property indicates a change to the actual item, not a change to the catalog record. An absent value MAY indicate that the item has never changed after its initial publication, or that the date of last modification is not known, or that the item is continuously updated.
See also:	§ 6.6.2 Property: frequency, § 6.5.4 Property: update/modification date and § 6.8.4 Property: update/modification date in DCAT 3 webpage

"temporal":

RDF Property:	dcterms:temporal
Definition:	The temporal period that the dataset covers.
Range:	In general used singularly can be used URIs as in intervals vocab OR dcterms:PeriodOfTime (An interval of time that is named or defined by its start and end dates)
Usage note:	In case we extend the model to serve The temporal coverage of a dataset may be encoded as an instance of dcterms:PeriodOfTime, or may be indicated using a IRI reference (link) to a resource describing a time period or interval. e.g. as (a dcterms:PeriodOfTime dcat:startDate "2016-03-04"^^xsd:dateTime ; dcat:endDate "2018-08-05"^^xsd:dateTime ;
See also:	Intervals.rdf

"language":

RDF Property:	dcterms:language
Definition:	A language of the item. This refers to the natural language used for textual metadata (i.e. titles, descriptions, etc) of a cataloged resource (i.e. dataset or service) or the textual values of a dataset distribution
Range:	Resources defined by the Library of Congress (ISO 639-1, ISO 639-2) SHOULD be used. If a ISO 639-1 (two-letter) code is defined for language, then its corresponding IRI SHOULD be used; if no ISO 639-1 code is defined, then IRI corresponding to the ISO 639-2 (three-letter) code SHOULD be used.
Usage note:	Repeat this property if the resource is available in multiple languages.
See also:	Also If representations of a dataset are available for each language separately, define an instance of dcat:Distribution for each language and describe the specific language of each distribution using dcterms:language (i.e. the dataset will have multiple dcterms:language values and each distribution will have just one as the value of its dcterms:language property).

"spatial":

RDF Property:	dcterms:spatial
Definition:	The geographical area covered by the dataset.
Range:	Xsd:anyURI to use in case using a IRI reference (link) to a resource describing a location. It is recommended that links are to entries in a well maintained gazetteer such as Geonames. Or a dcterms:Location (A spatial region or named place)
Usage note:	The spatial coverage of a dataset may be encoded as an instance of dcterms:Location, or may be indicated using a IRI reference (link) to a resource describing a location. It is recommended that links are to entries in a well maintained gazetteer such as Geonames.
See also:	e.g. for bbox dcterms:spatial [(a dcterms:Location) dcat:bbox """POLYGON((3.053 47.975 , 7.24 47.975 , 7.24 53.504 , 3.053 53.504 , 3.053 47.975))""" ;]

"accrualPeriodicity":

RDF Property:	dcterms:accrualPeriodicity
Definition:	The frequency at which dataset is published.
Range:	xsd:anyURI
Usage note:	We can use specific vocabulary e.g. in freq.ttl definitions like: http://purl.org/cld/freq/daily freq:monthly freq:weekly …….
See also:	See also freq.ttl or at frequency.ttl.txt

"temporalResolution":

RDF Property:	dcat:temporalResolution
Definition:	Minimum time period resolvable in the dataset.
Range:	xsd:duration
Usage note:	If the dataset is a time-series this should correspond to the spacing of items in the series. For other kinds of dataset, this property will usually indicate the smallest time difference between items in the dataset.
See also:	

"theme": [

RDF Property:	dcat:theme
Definition:	A (sub-)category of the resource. A resource can have multiple themes.
Range:	would be better to have xsd:anyURI with URIs that represent the various terms in a vocabulary (to be defined with Pilot partners for terms related to domains)

Usage note:	Use this for domain specific categories. E.g. subcategories like production machines, assembly lines, … To be defined by each application domain. Theme can be used multiple times to provide multiple subcategories. The set of themes used to categorize the resources are organized in a skos:ConceptScheme, skos:Collection, owl:Ontology or similar, describing all the categories and their relations in the catalog.
See also:	

],

"distribution": [(Distribution: A specific representation of a dataset. A dataset might be available in multiple serializations that may differ in various ways,
including natural language, media-type or format, schematic organization, temporal and spatial resolution, level of detail or profiles (which might specify any
or all of the above).

{

"title":

RDF Property:	dcterms:title
Definition:	A name given to the distribution
Range:	Xsd:string (rdfs:Literal)
Usage note:	Title
See also:	

"description":

RDF Property:	dcterms:description
Definition:	A free-text account of the distribution.
Range:	Xsd:string (rdfs:Literal)
Usage note:	Description Text of Data Set
See also:	

"license":

RDF Property:	dcterms:license

Definition:	A legal document under which the distribution is made available.
Range:	dcterms:LicenseDocument
Usage note:	For interoperability, it is recommended to use canonical IRIs of well-known licenses such as those defined by Creative Commons. Information about licenses and rights SHOULD be provided on the level of Distribution. Information about licenses and rights MAY be provided for a Dataset in addition to but not instead of the information provided for the Distributions of that Dataset. Providing license or rights information for a Dataset that is different from information provided for a Distribution of that Dataset SHOULD be avoided as this can create legal conflicts. See also guidance at § 9. License and rights statements.
See also:	§ 6.8.7 Property: rights § 6.4.19 Property: license **ToDo: Describe a list of possible licenses here.**

"accessRights":

RDF Property:	`dcterms:accessRights`
Definition:	Information about who can access the resource or an indication of its security status.
Range:	dcterms:LicenseDocument
Usage note:	Information about licenses and rights *MAY* be provided for the resource. See also guidance at § 8. License and rights statements. to express statements concerning only access rights (e.g., whether data can be accessed by anyone or just by authorized parties); Access rights can also be expressed as code lists / taxonomies. Examples include the access rights code list [*EUV-AR*] used in [*DCAT-AP*] and the Eprints Access Rights Vocabulary Encoding Scheme.

See also:	§ 6.4.20 Property: rights
	
	`0 dcterms:accessRights <http://publications.europa.eu/resource/authority/access-right/PUBLIC> ;`
	`1 dcterms:conformsTo <http://www.opengis.net/def/serviceType/ogc/csw> ;`

"downloadType":

RDF Property:	core:downloadType
Definition:	Information about Download Type (if means like as 'Stream' or 'Bulk' dataset download)
Range:	xsd:string
Usage note:	To use set of words like 'Stream' and 'Bulk' ..
See also:	2

"conformsTo":

RDF Property:	dcterms:conformsTo
Definition:	An established standard to which the distribution conforms. (Very OPTIONAL)
Range:	dcterms:Standard (A basis for comparison; a reference point against which other things can be evaluated.)
Usage note:	This property *SHOULD* be used to indicate the model, schema, ontology, view or profile that this representation of a dataset conforms to. This is (generally) a complementary concern to the media-type or format. This is a link to a specific file that describes the data in a domain specific format. Can also be a text in a freely definable format.
See also:	§ 6.8.17 Property: format, § 6.8.16 Property: media type also check file-type.ttl.txt

"mediaType":

RDF Property:	dcat:mediaType
Definition:	The media type of the distribution as defined by IANA [IANA-MEDIA-TYPES].
Range:	Xsd:anyURI (dcterms:MediaType)
Usage note:	dcat:mediaType *SHOULD* be used if the type of the distribution is defined by IANA [*IANA-MEDIA-TYPES*]. https://www.iana.org/assignments/media-types/ e.g. mediaType <http://www.iana.org/assignments/media-types/application/ld+json>

	E.g. a link to a XML, csv or JSON file, to describe the data format.
See also:	§ 6.8.16 Property: media type, § 6.8.15 Property: conforms to check also file-type.ttl.txt

"packageFormat":

RDF Property:	dcat:packageFormat
Definition:	The package format of the distribution in which one or more data files are grouped together, e.g. to enable a set of related files to be downloaded together.
Range:	Xsd:anyURI (dcterms:MediaType)
Usage note:	In case it is compressed, this could be .zip, .rar, … This property to be used when the files in the distribution are packaged, e.g. in a TAR file, a Frictionless Data Package or a Bagit file. The format *SHOULD* be expressed using a media type as defined by IANA [IANA-MEDIA-TYPES], if available.
See also:	§ 6.8.18 Property: compression format.

"accessService": [

{

"conformsTo":

RDF Property:	dcterms:conformsTo
Definition:	An established standard to which the distribution conforms.
Range:	dcterms:Standard (A basis for comparison; a reference point against which other things can be evaluated.)
Usage note:	This property *SHOULD* be used to indicate the model, schema, ontology, view or profile that this representation of a dataset conforms to. This is (generally) a complementary concern to the media-type or format.
See also:	§ 6.8.15 Property: conforms to

"endpointDescription":

RDF Property:	dcat:endpointDescription
Definition:	A description of the services available via the end-points, including their operations, parameters etc.
Range:	xsd:string
Usage note:	The endpoint description gives specific details of the actual endpoint instances, while dcterms:conformsTo is used to indicate the general standard or specification that the endpoints implement. An endpoint description may be expressed in a machine-readable form, such as an OpenAPI (Swagger) description [OpenAPI], an OGC GetCapabilities response [WFS], [ISO-19142], [WMS], [ISO-19128], a SPARQL Service Description [SPARQL11-SERVICE-DESCRIPTION], an [OpenSearch] or [WSDL20] document, a Hydra API description [HYDRA], else in text or some other informal mode if a formal representation is not possible.
See also:	

"endpointURL":

RDF Property:	dcat:endpointURL
Definition:	The root location or primary endpoint of the service (a Web-resolvable IRI).
Range:	xsd:anyURI
Usage note:	The URL address of the resource via service
See also:	

"servesDataset":

RDF Property:	dcat:servesDataset
Definition:	A collection of data that this data service can distribute.The Dataset ID or Title
Range:	xsd:string
Usage note:	
See also:	

"serviceSpecs": "string"

RDF Property:	core:serviceSpecs
Definition:	Description of service specification for more detail on the data service implementations
Range:	
Usage note:	To extend in case the description of data service to add more details descriptions on the system. To describe more details about the Service, e.g. QoS, …
See also:	In progress

}] }],

"datasetInformation": [

(Optional class. A description of types which represent attributes of observations , measurements , fields,.. in the dataset..)

High level description. NOT a description of each data point (that can go into conformsTo).
This is just to give an overview of the data, not a detailed description.

{

"measurementType":

RDF Property:	core:measurementType
Definition:	The data types which represent attributes of observations , measurements , in the dataset. *derived mostly from Wellbeing requirements(
Range:	xsd:anyURI
Usage note:	Simple text strings Or use of specific vocabularies collected to support domains For example like the vocab created for wellbeing (ex <http://www.i3-market.eu/wellbeing_annotations/Sleep_count_micro_awakenings> Specific types of measurements for a certain domain. Parameter can be put multiple times in the API call.
See also:	See also example for Wellbeing in DataRecords_Annotations_for_Wellbeing_datasets_measurements_02.ttl attac

	hed to this page but also in gitlab https://gitlab.com/i3-market/code/data-models/-/blob/master/Version-1/DataRecords_Annotations_for_Wellbeing_datasets_measurements_02.ttl

"measurementChannelType":

RDF Property:	core:measurementChannelType
Definition:	The data measurement Channel types in the dataset. Derived from AGORA requirements
Range:	xsd>string or xsd>anyURI
Usage note:	Simple text strings Or use of specific vocabularies collected to support domains
See also:	

"sensorID":

RDF Property:	core>sensorID
Definition:	Sensor ID
Range:	xsd>string
Usage note:	ID used to identify the sensors in original data sets source
See also:	

"deviceID":

RDF Property:	core>deviceID
Definition:	Device ID
Range:	xsd>string
Usage note:	ID used to identify the devices in original data sets source
See also:	

"cppType":

RDF Property:	core:cppType

Definition:	The cpp types in the dataset. Derived from AGORA requirements
Range:	xsd>string or xsd>anyURI
Usage note:	Simple text strings Or use of specific vocabularies collected to support domains
See also:	

"sensorType": "string"

RDF Property:	core:sensorType
Definition:	The cpp types in the dataset. Derived from Wellbeing and AGORA requirements
Range:	xsd>string or xsd>anyURI
Usage note:	Simple text strings Or use of specific vocabularies collected to support domains
See also:	

}] }]}