Ref. Ares(2021)7998500 - 29/12/2021

H2020 - INDUSTRIAL LEADERSHIP - Information and Communication Technologies (ICT)

ICT-38-2020: Artificial intelligence for manufacturing

Grant Agreement: 957362

Explainable Manufacturing Artificial Intelligence

WP2: Industrial Asset Management and Secure Asset Sharing Bundles

D2.1: Asset Management Bundles Methods and System Designs

Deliverable Leader: FRAUNHOFER

Due Date: 30.10.2021

Dissemination Level: Public

Version: D1.0

Short Abstract

Deliverable 2.1 deals with the architectural design for the asset management layer of the overall XMANAI Platform. This layer has the central task of importing/extracting data from external data sources (i.e. legacy and operational manufacturing systems) and ensuring data explainability in order to make them available for running AI pipelines.

In order to specify the asset management-related services, a detailed state-of-the-art analysis was performed. On the one hand, this includes all necessary methods for the execution of all asset management and sharing processes. And on the other hand, current technologies for fulfilling the XMANAI requirements were examined. Based on these results, a detailed architecture for the management of industrial assets and AI models was designed and accompanied by the selection of technologies and the elaboration of mock-ups to demonstrate the expected user interactions.

Disclaimer. The views represented in this document only reflect the views of the authors and not the views of the European Union. The European Union is not liable for any use that may be made of the information contained in this document. Furthermore, the information is provided “as is” and no guarantee or warranty is given that the information is fit for any particular purpose. The user of the information uses it at its sole risk and liability.

Document Log

Contributors	**FRAUNHOFER**, ATHENA, POLIMI, SUITE5, TXT, TYRIS, UBITECH
Internal Reviewer 1	SUITE5
Internal Reviewer 2	UNIMORE
Type	Report

History

Versions	Description
D0.1	State-of-the-art analysis for methods
D0.2	Extension of sota analysis for technologies
D0.3	Design of architecture and components
D0.4	XMANAI methods on asset management, sharing, provenance and security assurance
R0.1	Revision of internal reviewer 1 (SUITE5)
R0.2	Revision of internal reviewer 2 (UNIMORE)
D0.5	Updated version addressing comments received during the internal review
F1.0	Final version submitted to the EC

Executive Summary

Deliverable D2.1 "Asset Management Bundles Methods and System Design" provides the methods and the designs for developing the components responsible for data collection, management, provenance and sharing as well as for overall security of the XMANAI platform . In addition, it provides the first low fidelity mockups for an initial assessment of the intended functionalities and user experience. The deliverable refines these elements of the overall XMANAI architecture defined in the deliverable D5.1 “System Architecture, Bundles Placement Plan and APIs Design”. The results from this deliverable will be used for further development to prototype the XMANAI asset management services bundles in the first iteration (that is expected on M18).

This deliverable first provides a state-of-the-art analysis. It highlights the existing methods for asset management for industrial data and their suitability for XMANAI. Once the study of the methods is defined, the relevant technologies are researched and evaluated. The goal is to select technologies that can support or even fully implement the assessed methods to provide the required functionalities expected from the components. The XMANAI deliverable D1.2 “XMANAI Concept Detailing, Initial Requirements, Usage Scenarios and Draft MVP” is used as a reference here.

The challenge is to design and later develop an all-inclusive data management solution for assets and models for industrial data. Data need to be pulled, imported, managed, shared and tracked. A high level of security must be provided. It needs to be noted that the AI-related services, e.g. data pre-processing and AI pipeline management processes, are part of the deliverable D3.1, "Core Artificial Intelligence Bundles for Algorithm Lifecycle Management".

In the progress of the deliverable, the following points are presented:

- Current **methods** used and proven in science or industry. Six important topics of methods were addressed. These are data management methods for industrial data, for artificial intelligence (AI) analyses models and other assets. The analyses of methods is sorted into six topics. The first topic, *Industrial Data Ingestion* deals with data integration. Data must be imported and stored in XMANAI from different data sources and in various formats. The second topic, *Security, Privacy* , examines methods and technologies for data security and data protection. The central aspects are the data access control, secure data transfer and data anonymization. The third topic, *Trust Considerations* is dedicated to identity management, authentication and accountability management. *Industrial Asset Sharing* is the fourth topic and deals with data sharing. It examines how industrial data and models can be distributed. This includes the case of "data marketplaces" and smart contracts, which can be used to implement sharing without a middleman, for example. The fith topic, *Missing Data Acquisition* deals with the data enrichment and the acquisition of missing information for improving data quality. It focuses on data exploration, discovery and on syntethic data generation. *IPR Handing and Industrial Assets Provenance* is the last topic. Every access, every modification of data or other important assets should be tracked.
- Proven and modern **technologies** that can be used to implement the evaluated methods. These 47 selected technologies grouped in 14 categories are examined in a discussion for different purposes and to what extent they are suitable for XMANAI. In doing so, the solution must be practicable and applicable, as with other technologies, and must be closely interchangeable. Thus, compatibility is always taken into account.
  The development of the **XMANAI assests management architecture** is the most important outcome of this document. After the methods and technologies have been examined for their suitability for XMANAI, the components functional design is created and arranged into the architecture for the asset management services bundles and accomplished with initial mockups that serve as a basis for discussion and further design and development steps. The

architecture consist of five main components built from 16 subcomponents. These five components are a part of the overall XMANAI architecture presented in D5.1:

- Data Handler responsible for data and metadata collection and management of data and other relevant for XMANAI assets.
- XAI Marketplace responsible for sharing of data and management the metadata in XMANAI.
- Provenance Engine managing the provenance information about assets in XMANAI.
- Access Manager enabling secure and trusted access, management and communication of assets in XMANAI.
- The XMANAI Data Storage Services storing assets and information about them.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 XMANAI Project Overview

Despite the indisputable benefits that Artificial Intelligence (AI) can bring to society and in any industrial activity, humans typically have little insight about AI itself and even less concerning the knowledge of how AI systems make decisions or predictions due to the so-called “black-box effect”. Many machine learning/deep learning algorithms are opaque and not possible to be examined after their execution to understand how and why a decision has been made. In this context, to increase trust in AI systems, XMANAI aims at rendering humans (especially business experts from the manufacturing domain) capable of fully understanding how decisions have been reached and what has influenced them.

Building on the latest AI advancements and technological breakthroughs, XMANAI shall focus its research activities on Explainable AI (XAI) to make the AI models, step-by-step understandable and actionable at multiple layers (data-model-results). The project will deliver "glass box" AI models that are explainable to a "human-in-the-loop" without greatly sacrificing AI performance. With appropriate methods and techniques to overcome data scientists’ pains, such as lifecycle management, security and trusted sharing of complex AI assets (including data and AI models), XMANAI provides the tools to navigate the AI’s “transparency paradox” and therefore:

(a) accelerates business adoption addressing the problem that “if manufacturers do not understand why/how a decision/prediction is reached, they will not adopt or enforce it”, and
(b) fosters improved human/machine intelligence collaboration in manufacturing decision making while ensuring regulatory compliance.

XMANAI aims to design, develop and deploy a **novel Explainable AI Platform** powered by explainable AI models that inspire trust, augment human cognition and solve concrete manufacturing problems with value-based explanations. Adopting the mentality that “AI systems should think like humans, act like humans, think rationally, and act rationally”, a catalogue of **hybrid and graph AI models** is built, fine-tuned and validated in XMANAI at two levels: (i) baseline AI models that will be reusable to address any manufacturing problem, and (ii) trained AI models that have been fine-tuned for the different problems that the XMANAI demonstrators’ target. A bundle of **innovative manufacturing applications and services** is also built on the XMANAI Explainable AI Platform, leveraging the XMANAI catalogue of baseline and trained AI models.

XMANAI will validate its AI platform, its catalogue of hybrid and graph AI models and its manufacturing apps in **4 realistic, exemplary manufacturing demonstrators** with high impact in (a) optimising performance and manufacturing products’ and processes’ quality, (b) accurately forecasting product demand, (c) production optimisation and predictive maintenance, and (d) enabling agile planning processes. Through a scalable approach towards Explainable and Trustful AI as dictated and supported in XMANAI, manufacturers will be able to develop a robust AI capability that is less artificial and more intelligent at human and corporate levels in a win-win manner.

1.2 Deliverable Purpose and Scope

This deliverable on "Asset Management Bundles Methods and System Designs" is the first deliverable from WP2. It describes an approach to secure collection, management and sharing of assets in XMANAI.

For this purpose, an analysis of existing technologies is essential, which is the first central result documented in this deliverable. Therefore, state-of-the-art analysis of all relevant technologies and methods is included. This evaluation aims to determine an appropriate data management for XMANAI.

The second major part takes a closer look at the necessary components leveraging the draft XMANAI architecture (reported in the XMANAI Deliverable D5.1). In this context, a component is primarily a system consisting of different technologies and functions that encapsulate a feature as a whole. These are components for industrial assets collection, management and sharing. The assets handling in XMANAI should be trusted and secure.

Finally, references are made to the state-of-the-art analysis and additionally to Deliverable 5.1 "System Architecture, Bundles Placement Plan and APIs Design" to ensure alignment of the individual asset management-related components in the overall XMANAI project.

1.3 Impact and Target Audiences

The deliverable is intended to technical audiences within the XMANAI consortium in order to get more details about the adopted approach and early design for asset management in XMANAI. The reader should already have a basic understanding of the fundamentals of data management. While some basic methods are addressed in Chapter 2, knowing standard data management processes and techniques is helpful. This includes processes for collecting and storing data - interfaces / APIs and how they work are not discussed; different data management solutions (e.g. relational, graph database systems or triplestores) should be known.

This deliverable D2.1 contains a detailed state-of-the-art analysis of current methods and technologies. The reader subsequently receives an excellent overview of recent developments in the field of data management. In addition a classification of the XMANAI platform and descriptions of individual components are presented to provide a viable architectural approach that addresses the existing challenges and requirements of such a platform. Besides the traditional collection and provision of data, approaches to identity management, data anonymisation, modification tracking and even data marketplaces with smart contracts are described.

1.4 Deliverable Methodology

Six core topics were identified within this comprehensive deliverable: Industrial Data Ingestion and Storage Methods", "Security and Privacy", "Trust Considerations", "Industrial Asset Management and Asset Sharing", "Missing Data Acquisition" and "Industrial Assets Provenance and IPR Handling". For each topic, desk research on methods and technologies was made. All results were collected in appropriate tables together with a description, references and a consideration for use in XMANAI. An evaluation was designed to identify potential candidates and to establish additional cross-connections.

With the assistance of an iterative analysis, the use of each method and technology for XMANAI was examined. As a result, the selected and tested methods can be found in this document. Following this a list of all technologies for the corresponding purpose is shown. Each purpose can be assigned according to the six topics, described above. The decision which functionalities to specify for the emerging XMANAI platform is based on the associated MVP features listed in Deliverable D5.1, "System Architecture, Bundles Placement Plan and APIs Design" and elaborated in Deliverable D1.2 "XMANAI Concept Detailing, Initial Requirements, Usage Scenarios and Draft MVP".

The architecture design and the division of the components took place in internal workshops with all partners involved, in which the technical requirements were compared with the features of a component. In the end, an architectural design for the overall asset management for XMANAI was created.

1.5 Dependencies in XMANAI and Supporting Documents

Deliverable D2.1 "Asset Management Bundles Methods and System Design" was developed closely with Work Packages WP3 and WP5 and on the basis of the results reported under Work Package 1. While Deliverable D5.1 "System Architecture, Bundles Placement Plan and APIs Design" focuses on

the system architecture as a whole, Deliverable D3.1 "AI Bundles Methods and System Design" covers AI methods and technologies for working with data and addressing AI explainability aspects. The requirements come from Deliverable D1.2, "XMANAI Concept Detailing, Initial Requirements, Usage Scenarios and Draft MVP". This describes the technical specifications and features on which the work of Deliverable D2.1 is based.

The current document (D2.1) and its content must fit the holistic concept of the XMANAI platform, which was summarised in Deliverable D5.1. For this reason, there is a strong dependency on deliverable D5.1.

Deliverable D3.1 links directly to the components of work package 2 in terms of technology and methodology. The exact coordination of the individual functions and interfaces was therefore essential.

1.6 Document Structure

Section 1 relates the present document to the XMANAI platform and the work results of other work packages. The reader has learned what dependencies there are and how they are related to each other. A first overview and description of the content of Deliverable D2.1 can be found here.

The following section 2 deals with the state of the art analysis. Starting with methods for "Industrial Data Management", particular emphasis is placed on security and trust. The topic of data sharing and marketplaces plays just as important a role here as IPR handling and data provenance.

A state-of-the-art analysis of all relevant technologies follows the methods. Sorted by topic, all relevant technologies have been listed in a table, including an assessment of their use for XMANAI. Below each listing is a discussion of the potential use of each technology for XMANAI.

Section 3 identifies the components to be used in the XMANAI platform. Each component consists of a set of technologies and interfaces that encapsulate a feature or function. In the beginning, the functions of each component are described in more detail. Then, the respective technologies that are necessary to fulfil the function are explained.

Finally, in section 4, the work results are discussed, and the next steps are derived from them.

1.7 Ethics

Modern factories nowadays produce lots of data. In some cases they may include some personal data. With the enactment of the General Data Protection Regulation (GDPR), the European Union has created a framework to protect natural persons with regard to the processing of their personal data. For this reason, the integration of such data is a critical operation. For this purpose, methods and technologies were investigated to enable data providers to anonymise their data sets. This was done in chapters 2.1.2.4 and 2.2.5.

2 Industrial Asset Management and Sharing Landscape Analysis

2.1 Methods

2.1.1 Industrial Data Ingestion

This section focuses on methods around industrial data ingestion to give the reader an insight into the different ways in which data integration works.

2.1.1.1 Background

In general, "industrial data ingestion" refers to the collection and import of industrial data for immediate use. “Immediate use" can only mean persisting in a database or storing files in a file system. A general distinction is made between streamed data and batch processing. Streaming data is minute-by-minute data that is immediately available and can be evaluated. Batch data is a combination of data over a more extended period of time. A mix of the two methods is also possible, in which case it is a so-called lambda architecture. More on this issue can be found in the section "Streaming & Batch Data Processing".

Integrating data into a system is a standard procedure in data analysis. It involves retrieving data from remote sources and copying it into the system. This ensures that the data is not lost (e.g., the data source is no longer available at a certain point in time). In addition, it is more efficient to have the data already available on the system when carrying out transformations or analyses. Querying the information again for each analysis reduces performance and may require new data pre-processing. The data sources are usually external systems and must be connected to the in-house system first.

The ETL principle (Theodorou, Abelló, Thiele, & Lehner, 2017) is the best-known procedure. It stands for extract, transform and load. Data is gathered from one or more sources (extraction), transformed for individual purposes and made available for further use in a data store (loading).

Figure 2-1 shows the ETL process in a simplified representation. The individual components are shown in colour and work independently of each other. The arrows here represent the data flow. In the end, there is a database with the pre-processed data that can be used for analysis or AI application. In addition, there may well be other databases at the individual components in the ETL process to process more significant amounts of data better or to store them temporarily.

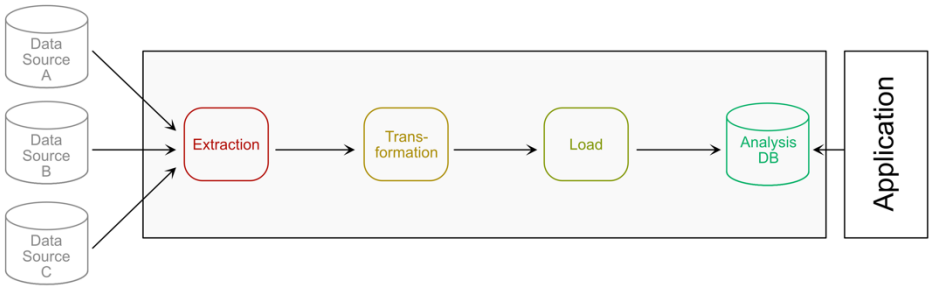


Figure 2-1 Graphical representation of ETL pipeline example

The problem with the classical approach (such as ETL) is that the data volume has multiplied over the last few years, and classical techniques are no longer powerful enough to work efficiently. Furthermore, there is much more heterogeneity due to the data volume, making data cleansing more complex.

Industrial data ingestion thus describes the process of ingesting data from external sources in the most performant and correct way possible. New approaches are used to address the growing volume of data.

2.1.1.2 Data formats

To work with data, two things are needed - the data itself and metadata that describes the actual data. Several formats and standards exist for both.

A few standard file formats have become a standard in industrial manufacturing. JSON and CSV are the most prominent representatives. XML is also used somewhat less frequently. While JSON / XML tend to be provided via APIs, CSV is usually available as a file.

The *CSV* format (Comma-separated values) is a standard file format for the exchange of tabular data. Individual data is written in a row, while a column represents the same type of information. A comma or semicolon separates data values. The TSV format rarely exists, where the data is separated by an indentation (the tabulator). In this way, it is easy for machines to interpret the information quickly or to insert it into databases.

There is no general standard, but the format is described in RFC 4180 (Shafranovich, 2005). A character encoding is not specified, which must always be maintained during processing, as otherwise, incorrect information will find its way into the system. If more complex data structures have to be transmitted - such as nested data - several CSV files can be linked together. However, JSON or XML are better suited for this case.

The structure of CSV is straightforward and, therefore, very understandable. A line break is used to separate data records, and individual data fields (columns) are separated by a comma, semicolon, tab, double point or space. Decimal numbers are marked with a dot (.). If a single value needs spaces, it can be placed in quotation marks (") so that programmes interpret it correctly.

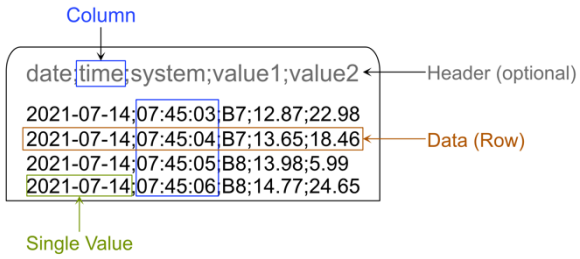


Figure 2-2 Structure of CSV

Figure 2-2 shows the structure of a CSV file schematically. There is also a header that contains the titles for the individual columns. The header is optional but should always be present to put the data into context.

This format is handy for time series and is frequently used in industrial production. Many programming languages have corresponding libraries and APIs to transfer CSV data into other data structures or quickly interpret/save it as a file and validate CSV. Furthermore, it should be mentioned that the W3C has been working on a CSV Schema Language[1] but has not published an official standard yet.

Unlike CSV, *JSON* does not have a direct tabular data structure. The basic structure always has key-value pairs. However, JSON has a unique feature in its definition (Bray, 2017) that the value is not simply a string or numerical value. The value in the key-value pairing can take on several data types, which makes JSON more flexible than CSV.

[1] http://digital-preservation.github.io/csv-schema/csv-schema-1.2.html

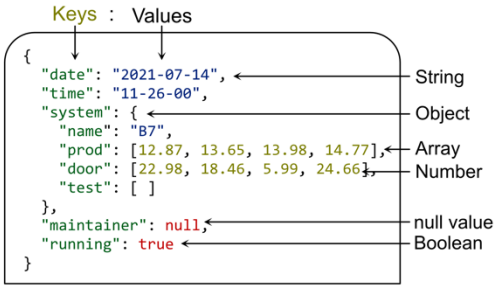


Figure 2-3 Structure of JSON

Figure 2-3 shows an example of the structure of a JSON file. A value can be a string or a number. If several key-value pairs are contained, they are separated by a comma. The value is automatically interpreted as a decimal number with a dot, otherwise as an integer. Date formats are always entered as strings and should be formatted according to a standardised format like ISO-8601[2].

It is possible to specify lists (arrays) of values by enumerating them in square brackets ([]). The array can, in turn, contain different values of different data types. Nesting is implemented by making a value itself a new object (JSON) - this is indicated by curly brackets ({}). This object can then again contain any number of key-value pairs and also new objects. JSON objects are constructed following this scheme[3]. In addition, it should be mentioned that Boolean can simply be entered as "true" or "false".

Asset Monitoring

Working with data needs two things - the data itself and metadata that describes the actual data. For example, in manufacturing, data is usually produced automatically by systems, mostly every second. This can be data about production, machine failures, product sales, product quality measurements, or something else that is important in industrial manufacturing.

It is essential that the data can be provided and copied from another system - typically via standardised interfaces.

An analysis of the data enables the establishment of monitoring processes and interventions in case of problems. In this way, other errors can be avoided, and delays in the following steps do not happen.

These data is often not in a standardised format because the purpose for further use was often unclear at the time of installation. However, the integration of the data then goes hand in hand with the transformation into a standardised schema, such as the XMANAI Data Model and state-of-the art manufacturing data models described in the XMANAI Deliverable D3.1.

Asset Metadata

Metadata is information about the nature of the industrial manufacturing process it describes, e.g. products, manufacturers, process flows, location.

Descriptive metadata enables the identification, location and retrieval of resources. A controlled vocabulary is another advantage for classifying and indexing data. This is described in more detail in the Data Linking section. Technical metadata provides information about the technical process used to create the data presented. Administrative metadata documents the version of the data set at hand. Temporal factors such as the creation or modification date are stored. Finally, usage metadata records access and calculates checksums to confirm correctness.

Some metadata standards exist that also represent a defined vocabulary for the metadata. In this way, the integration process is simplified because the values of the metadata are acceptable. Well-known

[2] https://datatracker.ietf.org/doc/html/rfc3339
[3] https://www.json.org/json-en.html

representatives are Dublin Core[4] and DCAT[5]. However, these two are very generally defined and may need to be extended for use in XMANAI.

2.1.1.3 Harvesting Process

Specific data is collected from an online resource in order to process them for analysis. In this process, the data sets mentioned in the beginning and their metadata are copied between two or more data platforms in order to make them usable.

In general, harvesting refers to accessing an online resource. This can be a website, an HTTP interface (REST), or files on file servers that are opened to the public. The process runs entirely automatically and is executed at specified time intervals. The goal is to import the data as accurately as possible. As a rule, two stages are executed, the harvesting itself and optional post-processing of the data.

This post-processing can be combined with schema mapping to corresponding to a data model or a metadata vocabulary.

2.1.1.4 Streaming & Batch Data Processing

As already mentioned in the "Background" section, there are two ways to integrate data into the own system.

Batch processing combines data into blocks, and these are made available as files or other package formats via the internet. This aggregation of data creates large amounts of data. As a result, processing takes more time, but the algorithms can be applied very efficiently. The best known are the MapReduce algorithms, with which these batch data can be processed very well.

Streaming data, on the other hand, is useful when analysis results are to be in real-time. The data is generated by a system and integrated within a few seconds using appropriate technologies. Well-known tools are Apache Kafka or Apache Flink, which offer, for example, an SQL variant to query data from a stream.

The difference to batch processing is the timeliness of the data, but usually, only snapshots can be considered in further analyses. However, if a larger view of the data is required for a more extended period, batch processing is more suitable.

Errors or patterns can be identified more quickly with stream processing, but larger patterns or behaviours are better identifiable in batch data because they cover an extensive period at a glance, and the analyses are more efficient.

A combination of the two approaches to integrating data is possible in any case, and ultimately the choice depends on the application purpose for which the data is to be used.

2.1.1.5 Linking

Concerning the use of knowledge graphs or graph structures in general, the principles for Linked Data of the Semantic Web suggest themselves. Tim Berners-Lee set out four principles for Linked Data, which are best practices.

Datasets should be given a Uniform Resource Identifier (URI) to identify in a uniform and unique way. HTTP URIs are suitable for this purpose in order to make them additionally dereferenceable.

The Resource Description Framework (RDF) is suitable for the design of graphs. RDF is a simple and graph-based data model for statements about resources in the form of triples (subject, predicate, object). Subject and object are related to each other, which is described by the predicate. Combining these triples results in large semantic networks that can be used for analyses and AI models.

[4] https://www.dublincore.org/specifications/dublin-core/dcmi-terms/
[5] https://www.w3.org/TR/vocab-dcat-2/

The introduction of controlled vocabularies (ontologies) can also be accomplished with the Resource Description Framework Schema (RDFS). Schemas already exist for unique data models and the metadata standards mentioned above.

The individual triples should be linked to each other by relating two resources with a different triple. For this purpose, own predicates exist to describe links among the data.

The query language SPARQL makes it possible to formulate complex queries about data sets in Linked Data format. The generation of such a graph would be part of the data transformation in order to transfer harvested data into the RDF structured format.

2.1.1.6 Key Challenges & Considerations

Harvesting and processing data is a major challenge because the data must address different requirements. Any adaptation of the data model, schema, transformation algorithms or other steps must always be aligned with the expected goal of the analysis. Each later analysis within the XMANAI platform has different requirements for the data. This is why the ingestion system must be flexible enough to integrate heterogeneous data.

AI explainability is the main topic of the project. It requires a high level of understaning of all steps in the analytical process and all used artefacts. The understandability or explainability of data is one of key requirements here. Data engineers and data analysts have to understand the semantic and structure of the data at hand. This make important the availability of high quality of metadata providing all required information about the data. Achieving high data quality is also associated with effort because harmonisation and cleansing are made more difficult by large data heterogeneity.

2.1.2 Security, Privacy

2.1.2.1 Background

The rise of technology platforms which are handling the tremendous amount of information that is generated from the increasing number of data sources has created the emerging need for security and privacy technologies and solutions. In addition to the rise of technology platforms, the evolution of the cloud and computing technologies has resulted into the introduction of several offerings which are leveraged by the corresponding stakeholders to effectively store and process their data. Towards this end, the privacy and security have been very important topics in the age of big data and the extended research on this topic resulted in a constantly evolving large list of state-of-the-art security and privacy technologies whose combination is attempting to provide solutions for a secure and privacy preserving environment. Nevertheless, the security and privacy research area is too broad, hence the most relevant topics which are directly related to the XMANAI project's activities are the *security of the data in motion*, the *data access control* and the *data anonymisation* topics.

2.1.2.2 Secure Transfer

The digital world is on the verge of an era where the platforms and applications are often transferring large volumes of data between resources that are usually residing across different network locations. In this new era, the need for high performance data transfer rises in conjunction with the hard requirement for secure data transfer and data sharing which are considered fundamental aspects of the designed data intensive applications. To this end, large resource efforts have been performed towards the design and implementation of protocols that can effectively and efficiently support these operations with the main characteristics of high throughput, fairness and stability in various layers of the TCP/IP model.

The main protocols for secure and efficient data transfer are as follows:

- *IPSec:* This is a well-established network protocol that has been defined in RFC 4301 for the establishment of a secure channel of the network. IPSec is compatible with both IPv4 and IPv6 networks and enables the required security, confidentiality, integrity at the Internet Layer of

the TCP/IP model. It provides the required security on the Transport Layer in two phases. The first phase is related to the authentication and the negotiation steps for the secure channel establishment and the second phase is related to the secret and parameters exchange.

- *TLS:* This is the most widely adopted protocol for the secure communication over networks assuring privacy and data security. Its main scope is to encrypt the exchanged information between the communicating parties. TLS is Internet Engineering Task Force (IETF) standard and its most current version (v1.3) was published in 2018. TLS is considered the successor of the previously established Secure Socket Layer (SSL). Its utilisation in conjunction with the Secure HTTP protocol (HTTPS) is the de-facto choice for secure Internet communication.
- *GridFTP* (Suresh, Srinivasan, & Damodaram, 2010)*:* It was established as a secure data transfer protocol, extending the FTP protocol and RFC 959, RFC 2228 and RFC 2389. It is primarily used for the efficient and effective secure transfer of large amounts of data utilised in grid computing. Its main features include the parallel data transfer which is realised through multiple TCP connections towards the increase of the throughput. It supports the TCP connections establishment and the automatic negotiation of the TCP socket buffer size. It is characterized by high performance, security, reliability.
- *GridCopy* (Kettimuthu, Allcock, Liming, Navarro, & Foster, 2007)*:* It is a secure data transfer protocol that is designed for optimal performance. It differentiates from other protocols by utilising the scp-style source and destination specifications. It builds directly on top of GridFTP in order to provide provides a simple user interface to this enhanced functionality, while also taking care of all tuning operations that are required to get optimal performance for data transfers.
- *UDT* (Lu & Chengrong, 2018)*:* It is a secure data transfer protocol operating on the application layer of the TCP/IP model. It is an end-to-end, unicast and reliable connection-oriented data transport protocol. Its main differentiation is that it utilises UDP as the basis instead of TCP which is only used for the control of the established channel. It provides several functionalities, such as packet loss check, high speed bulk data transfer and a sophisticated congestion control mechanism.

2.1.2.3 Data Access Control

Data access control refers to the methodologies applied that guarantee the selective restriction of access to any critical or valuable resources towards the assurance of data confidentiality and data integrity. To eliminate unauthorised data access or data misuse, data access control covers all the authentication, the authorisation, access regulation and access auditing aspects. It regulates the access and legitimate actions that the subjects can perform over the objects, which refers to the system or the resources (data, services, network, etc..) of the system. In general, access to resources refers to discovering, reading, creating, editing, deleting, reserving and executing resources (Hu, et al., 2013).

In order to perform access control a variety of Access Control Mechanisms (ACM) are available with the aim of the realization of the various logical access control models that exist. Each model proposes a security framework along with a set of conditions that define how the objects, subjects, operations and rules can be combined in order to form the access control decision that will either grant or deny the access to the requestor. The most widely-used ACMs at the moment are the following:

- *Access Control List (ACL):* ACL constitutes the most basic ACM where access to a system resource is regulated through a list of permissions that bind a system resource with a user or a system process in order to define the allowed operations on this specific system resource. ACM is considered the basis of most widely-used ACMs that further enhance and optimise this approach to introduce more fine-grained solutions. Different implementations of ACL can

be found on multiple resource types such as local and network filesystems, Active Directory and networking resources.

- *Discretionary Access Control (DAC)*: DAC is providing access to the subject on the basis of subject identity and/or groups to which the subject belongs to. Within the DAC policies, the access to the object for subjects is defined along with the authorisation that specifies the permitted access method. As such DAC consists of access rules and access attributes. Through the access attributes several distinct levels of authorisation are defined and by utilising the access rules every individual request to access the object is validated. With DAC, controlled sharing of objects among various subjects is enabled. In DAC, the controls are discretionary, as the owner of an object or a subject with certain permissions on the object is capable of passing (perhaps indirectly) that permission to the other subjects.
- *Mandatory Access Control (MAC)*: MAC refers to a type of access control by which the operating system constrains the ability of a subject or initiator to access or generally perform some sort of operation on an object or target. Following a hierarchical approach, access to all objects is controlled by settings defined by the administrator of the operating system. As a result, all access to objects is controlled by the operating system and the subjects cannot change the access control of an object. Particularly, subjects and objects are assigned with security attributes. For every request of a subject to access an object, an authorisation rule enforced by the operating system kernel examines the security attributes and grants or denies the access. As such, all operations from subjects are validated against a set of authorisation rules (policies). In MAC, subjects have no authority on overriding or modifying a policy at any circumstances.
- *Identity Based Access Control (IBAC)*: In IBAC access control mechanisms such as the access control lists (ACLs) are employed in order to capture the identities of the subjects that are allowed to access the object. The identity of the subject must be included in the ACL in order to grant access to the object. For each subject, the object owner is granting individual privileges to perform operations on the object. As such, each object has its own ACL and set of privileges assigned to each subject. The authorization decisions in the IBAC model are based on the ACL and are made statically. If the requesting subject is in the ACL prior to making the request, then access is granted.
- *Role Based Access Control (RBAC)*: RBAC is an approach restricting access to objects based on the role of the subject. In RBAC, a set of pre-defined roles that hold a set of privileges is employed and the subjects are assigned to these roles. Subjects assigned to different roles have access to different sets of objects. The access is based on the person that assigns the roles to the subjects and the object owners that determine the privileges associated with a role for their objects. The access control mechanism evaluates a request based on the role assigned in the subject performing the request and the privileges of this role authorised to perform on the object in order to permit or deny the access.
- *Attributed Based Access Control (ABAC)*: ABAC model defines an access control mechanism in which access rights are granted to users through the use of policies in which attributes are combined together. IBAC and RBAC are special cases of ABAC. IBAC works on the attribute of "identity" with ACLs, while RBAC works on the attribute of "role". The differentiation of the ABAC is the concept of policies in which multiple different attributes are evaluated through a complex Boolean rule set. As such, the model supports Boolean logic, in which rules contain "IF, THEN" statements about who is making the request (subject), the resource (object) and the action (operation). ABAC is considered a "next generation" authorisation model, although it has existed for many years, because it is enabling a dynamic, context-aware and risk-intelligent access control to resources. For the ABAC policies different languages have emerged including the eXtensible Access Control Markup Language (XACML), the Next

Generation Access Control (NGAC), the Security Assertion Markup Language (SAML) and the Enterprise Privacy Authorization Language (EPAL).

2.1.2.4 Data Anonymization

In the big data era, data is collected in enormous volumes from a large variety of data sources. However, a significant part of the collected data is also incorporating personal and sensitive data in various forms, such as personal information, financial information and personal activity information. The collection, processing and sharing of personal and sensitive data however is now strictly regulated by various national and international data protection laws, such as the European General Data Protection Regulation (GDPR), US Health Insurance Portability and Accountability Act (HIPAA) and the California Consumer Privacy Act (CCPA). Hence, the challenging task of data anonymisation arises that will ensure both the compliance to the underlying regulations, as well as to ensure the privacy preservation.

Data Anonymisation is the process where the information included on a dataset is transformed in a way that the privacy of the individuals is preserved, as well as the privacy risks are reduced while the reduction of risks is balanced against a reduction of data utility (Fung, Wang, Fu, & Philip, 2010). At its core, a data anonymisation process is providing the means to safeguard the privacy preservation when the data is processed, shared or published, ensuring that malicious activity or process that would attempt to extract any information that would trace back to any specific individual will not succeed. During the data anonymisation, the variables included in the dataset are classified as *insensitive variables* which are kept unchanged, *direct identifiers* that should be removed, *quasi-identifying (QID) variables* that when combined may be used to identify an individual that should be transformed and *sensitive variables* that should be protected. In this sense, the data anonymisation process provides the means to preserve the privacy by applying several data anonymisation techniques that will effectively generalize, conceal or mask the personal information.

The main data anonymisation techniques can be grouped as follows (Jayabalan & Rana, 2018):

- *Generalization techniques*: The generalisation techniques are focusing on masking the information of the QID variables by applying a general value that complies to a particular classification in the case of categorical variables and an internal value in the case of numerical values. Four different variations of generalisation techniques are found in literature: a) *full domain generalisation* where the QID variables are generalized at the same level in a given tree structure, b) *subtree generalization* where the generalisation of node to its parent node imposes the generalisation of its child nodes also, c) *unrestricted subtree generalization* that is a variation of subtree generalisation with increase granularity level on the generalisation of of the child nodes, d) *cell generalisation* that focuses on the generalisation of a single record and e) *multidimensional generalization* where all QID variables are generalised based on a combination of all the values of the QID variables.
- *Suppression techniques*: The suppression techniques are focusing on the replacement of a specific value or part of the value of a variable with a special character such as the asterisk or hash character.
- *Bucketization techniques*: The bucketization techniques are focusing on the re-arrangement of sensitive variables while keeping the QID variables unchanged. In particular, the identifiers are eliminated and the remaining tuples are partitioned into buckets. Afterwards, the sensitive variable values are separated from the QID variable values and are randomly rearranged into each bucket. As a result, the anonymised data is composed by buckets containing rearranged sensitive variable values.
- *Slicing techniques*: The slicing techniques are breaking the associations between different columns while preserving the associations within the same column. The data is either partitioned vertically

or horizontally on the associations. Then a variation of the bucketization technique is utilised to generate the anonymised data.

- *Randomization*: The randomization techniques are focusing on the introduction of randomly masked data for both categorical and numerical variables. In particular, these techniques are introducing noise in the data by either additive perturbation randomized noise that introduced to specific records or by multiplicative perturbation that disrupts the records with arbitrary rotation and projection techniques.

2.1.2.5 Key Challenges & Considerations

While in the security and privacy research significant advancements have been made in the last decade, there are still several key challenges that are not completely solved and a number of considerations that should be carefully taken care of before a specific technology is embraced and the proper solution is designed. The following list describes these key challenges and considerations:

- The evolution of data policies should be carefully managed. As the needs for data access control are dynamic and might evolve during the lifetime of the service or platform, an effective and efficient process must be designed and implemented to successfully handle the complete data access policies lifecycle.
- In data anonymisation it is crucial to achieve the balance in the "privacy vs usefulness/quality" trade-off. While the current regulations impose significant restrictions in the collection, processing and sharing of data that contain personal information, on the other hand the data anonymisation process might lower the quality and usefulness of the underlying data if not utilised in the proper manner.

2.1.3 Trust Considerations

2.1.3.1 Background

The increase of trust of the users in the technology platforms and services is considered among the most important topics in the new digital era and as a consequence extended research has been performed on this topic which has resulted in a list of emerging state-of-the-art technologies which can be leveraged. To this end, a number of candidate solutions are available in the areas of identity management and accountability management. in this new era the need for efficient and effective identity management arises, that covers the holistic user account management lifecycle that spans from the creation and maintenance to the de-provision of user accounts following a set of administrative standards. On the other hand, the need for robust and solid accountability management that effectively cover the aspects of how data is used and under which circumstances as it is also evident from the multiple data breaches that the technology platforms and services have suffered from in the last decade.

2.1.3.2 Identity Management / Authentication

In the new digital world, the need for efficient and effective identity management arises, that covers the holistic user account management lifecycle that spans from the creation and maintenance to the de-provision of user accounts. This is referenced as Identity Management and it incorporates all the processes and policies that are related to the management of digital identities. As the number of services and offerings in this digital era are exponentially growing, the identity management research area has evolved significantly over the past year and a vast amount of effort has been performed towards more innovative and novel identity management models. The core parts of all models include the following three core entities:

- The User which is the individual that utilises his/her digitally identity to perform an action;

- The Identity Provider (IdP) that is responsible for the complete identity management lifecycle (generation, update, maintenance and delete of the digital identity) as well as the execution of the authentication process.
- The Service Provider (SP) that is responsible for providing the underlying service(s) which the user is willing to utilise and is relying on the IdP for the identity verification process.

The following list presents the dominant identity management models:

- *Isolated (or Silo) Model* (Laurent & Bouzefrane, 2015)*:* The specific model is considered the most basic model. The core concept of this model is that it merges the role of the identity provider with the service provider. To this end, the service provider is providing both the identity verifications and the authentication operations. However, all identity management operations such as the creation, modification or deletion of the user's identity is performed by the service provider and it is not shared among different service providers. This implies, that the user can only access and utilise the services of the specific service provider, while also that the service provider should create, store and maintain all the identity information of the users.
- *Central Model* (Zwattendorfer, Zefferer, & Stranacher, 2014)*:* The specific model is based on the concept of centralised identity management. In this sense, the identity management is performed by a single IdP that undertakes the responsibility to perform the complete identity management lifecycle for several SPs. In this manner, the users can have a single digital identity and the model enables the Single Sign On (SSO) where the same identity can be used by the user across different SPs. Furthermore, the identities of the users are stored and maintained by the single IdP and the SPs do not have to store and maintain them in their own repositories.
- *User-Centric Model* (El Jaouhari, Bouabdallah, & Bonnin, 2017)*:* The specific model is providing the opportunity to the users to store and maintain their identity into their own repository. In this manner, the user has full control of their identity information and upon their consent the user can explicitly approve the use of their identity from an IdP to the selected SP. This approach significantly increases the privacy of the user's identity.
- *Federated Model* (Zwattendorfer, Zefferer, & Stranacher, 2014)*:* The specific model introduces the concept of a federation in which several IdPs and SPs are participation in the trusted manner. Hence, the identity is stored and distributed across the different IdPs and SPs of the federation and no single entity is in control of the identity. The specific model enables the identification and authentication across different domains supporting also the SSO concept.

The most dominant technologies used for the realisation of the identity management, as well as its authentication mechanism, are as follows:

- *OAuth 2.0*: It is the most common authorization standard and is proposed and maintained by the IETF through a series of RFCs such as the RFC 6749, the RFC 6750 and the RFC 6819. While being an authorization standard, it is also used as an authentication process. It is designed to support not only web applications and services but also mobile applications. It supports four different authorization cases (referred as Grant Types). The different clients are classified based on their ability to securely authenticate with the authorisation server.
- *OpenID Connect*: It is widely used authentication protocol issued by the OpenID Foundation. Its current version is based on a set of specifications composed by core and additional services. OpenID Connect is based on the OAuth 2.0 protocol which was expanded by resolving several open items that were not covered in the OAuth 2.0 protocol.
- *SAML 2.0*: Security Assertion Markup Language is an XML-based standard which was developed by the OASIS consortium. The standard defines a clear hierarchy of basic concepts such as assertions, protocol, bindings and profiles which enable the transmission of the authorization

credentials from the IdP to SP. SAML bridges the gap between the authentication of user's identity and the authorization to use a specific service.

2.1.3.3 Accountability Management

The emergence of ICT platforms that usually operate on cloud environments raised the requirement for the accountability of the data provenance and data management lifecycle as several stakeholders, such as the service providers, the data consumers and data providers, are involved. To this end, it is imperative that a clear definition of how information is managed, how any action is verified and how any discrepancies between the occurred actions and the expected actions are remedied with proper explanation and verification. In this sense, accountability management is considered an important prerequisite for the increase of trust of any ICT platform.

To achieve this, several aspects of the ICT platforms should be taken into consideration. At first the security and privacy threats of the underlying system should be taken into account when the overall security approach is designed. Furthermore, the limitations of the utilised technologies shall be taken into consideration together with their trustworthiness as the utilisation of cloud offerings introduces requirements in trust and security.

Accountability in ICT platforms and cloud offerings consists of accepting responsibility for data with which it is entrusted in a cloud environment, for its use of the data from the time it is collected until when the data is destroyed (including onward transfer to and from third parties) (Jaatun, et al., 2018). Another research proposed (Zou & Pavlovski, 2007), the modelling of the accountability in IT services as the obligation that several persons, groups, or organizations assume for the execution and fulfilment of a service with the following obligations: a) answering, providing an explanation or justification, for the execution of that authority and/or fulfilment of that responsibility, b) full disclosure on the results of that execution and/or fulfilment, c) undeniable liability for those results (non-repudiation) and d) obtaining trusted agreement of accountability from all entities involved in the service who in turn are bound to the obligations set out above.

The complete accountability lifecycle, namely the Cloud Accountability Life Cycle (CALC) as referenced in literature (Ko, Lee, & Person, owards Achieving Accountability, Auditability and Trust in Cloud Computing, 2011), is consisting of seven distinct phases which are the following (Ko, et al., 2011):

- *Policy Planning*: This phase includes the decision of what information should be logged and which events should be logged on-the-fly. Depending on the nature of the data, different approaches should be followed however the main information that should be logged should include: a) events describing the activities performed, b) the stakeholder that triggered the activity, c) the exact time and date of the performed activity, d) the details of the location (virtual or physical) that the activity took place.
- *Sense and Trace*: This phase handles the triggering of the logging whenever a new event occurs or it is expected to occur. Monitoring and triggering should include all activities from the lower layers (such as read/write operations) to the highest layers of the architecture such as the triggering of services or workflows.
- *Logging*: This phase handles the complete logging lifecycle. It takes into consideration various parameters, such as the lifespan of logs, the level of detail, the storage location and rotation policy of the collected logs, as well as the need for data anonymisation on some sensitive information.
- *Safe-keeping of Logs*: This phase is responsible for the security and privacy preservation of the collected logs. It should handle the proper access to the logs and their integrity by applying access control mechanisms, encryption and a backup utility to prevent loss or corruption of the collected logs.

- *Reporting and Replaying*: The specific phase handles the generation of the proper reports for inspection providing information such as audit trails, access history of files and the complete lifecycle of the files. It should also handle the detection of irregularities and suspicious events.
- *Auditing:* The specific phase handles the detection of irregularities and suspicious events which are highlighted to the administrators of the system. In the case where this is performed automatically, the administrators are able to quickly react on any incident.
- *Optimising and Rectifying*: The specific phase acts as the retrospective phase of all other faces. It detects any problematic areas or security loopholes and constantly enhances the whole process.

2.1.3.4 Key Challenges & Considerations

While the technologies and solutions for both identity management as well as accountability management are constantly evolving, it is evident that several key challenges remain still open despite the advancements that have been made in these research areas. The following list describes these key challenges and considerations:

- Nowadays many applications have adopted the hybrid architecture approach of having both on premises and cloud application's version. The effective and efficient access on both the application's versions can be rather challenging with the existing identity management approaches.
- As the applications are rapidly evolving nowadays and new versions are produced in small period, the integration with any identity management solution should also remain also up-to-date which can be rather challenging in the case where connectors are utilised.
- The accountability management for services and applications operating over a cloud infrastructure that scales elastically increase the need for efficient logging techniques with the proper scope and scale in the level of logging. Moreover, these can operate over different operating systems. Additionally, the level of detail includes the decision to log operations and actions on the system level, on the file level or even both which increases the complexity of the process but also the need for scaling.

2.1.4 Industrial Asset Sharing

2.1.4.1 Background

Since the beginning of mankind, communities have traditionally come together to share or trade resources and work collaboratively for a common cause. Hence, asset sharing is not a new term. In recent years, however, digital technologies and the increasing volume of generated data have transformed the economy, as well as the society, affecting people, their activities and everyday life in unprecedented ways. The global proliferation of big data—and the ways in which organisations can analyse it—is shaping healthcare, education as well as industry. Data have become the new precious commodity that can be accessed and processed to obtain knowledge, used in decision making. While the mainstream research in big data focuses on developing algorithms of knowledge extraction and resource management, there is also an economic perspective that has drawn subtle attention (Spiekermann, 2019). Data marketplaces have emerged as a new business model where data from various sources can be collected, aggregated, processed, enriched, bought, and sold. At the same time, the provision of value-adding data-related services beyond the core functions of a data marketplace, like data analytics and AI models, produce even more complex assets for sharing and trading.

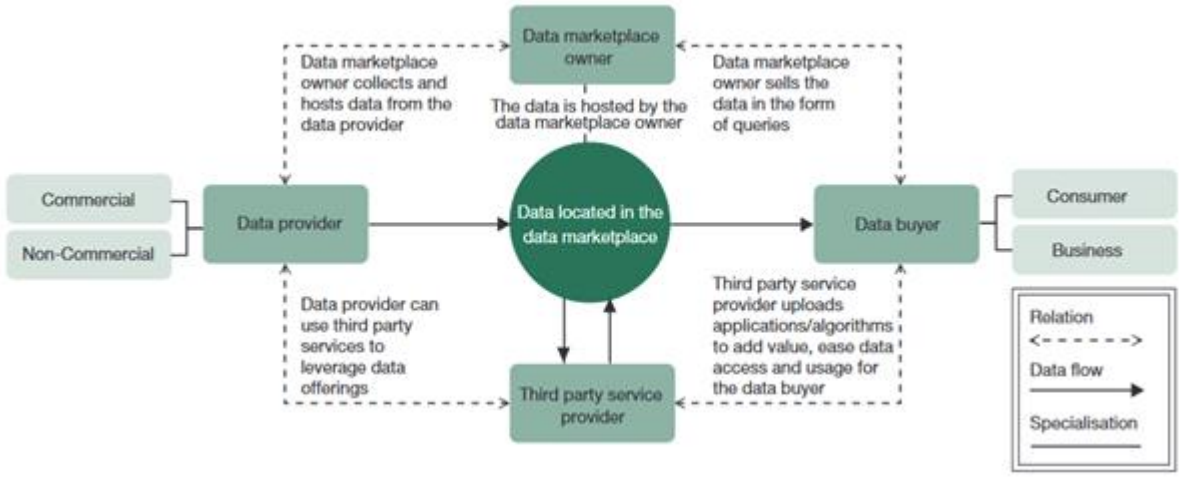


Figure 2-4 Data Marketplace according to (Spiekermann, 2019)

In the XMANAI context, an industrial asset is considered to be any digital asset related to industry data or *derived* from it, such as processed data, trained AI models, experimental results or analytics reports. These "data products", called AI Assets in this section, can also be shared or traded through a proper infrastructure, standardised licensing models, as well as regulations regarding data access and usage.

Looking at the European industry and market, the European Commission has taken a number of steps in the last decade (e.g. the EU Open Data Portal in 2012) and has proposed a number of policy and legislative initiatives to unlock the re-use potential of different types of data and create a common European data space. But it was only recently, in February 2020, when the European Commission announced a comprehensive European Digital Strategy[6] for data under the "Europe Fit for the Digital Age" vision[7], accompanied by a white paper outlining the EU intentions for Artificial Intelligence (AI) regulation, especially in high-risk areas (e.g. health care), providing also ethical guidelines for building trustworthy AI systems (European Commision, On artificial intelligence-A European approach to excellence and trust, 2020). In the same direction, during April 2021, the Commision published a proposed legal framework on AI (AI regulation) in order to address in a future-proof manner the specific challenges and risks related to AI systems and applications (European Commision, Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL LAYING DOWN HARMONISED RULES ON ARTIFICIAL INTELLIGENCE (ARTIFICIAL INTELLIGENCE ACT) AND AMENDING CERTAIN UNION LEGISLATIVE ACTS, 2021).

In this vision, which aspires to enable EU to become the most attractive, most secure and most dynamic data-agile economy in the world, Technological Sovereignty, Artificial Intelligence and Data Economy are three of the fundamental keywords. Infrastructure and Data are at the centre of the proposed transformation, since the growth of the amount of technologically mediated information has been quantified in different ways, including society's technological capacity to store information, to share and exchange information, and to process information.

To this end, a single, EU law abiding, data space is envisioned for the industry and the public sector. A genuine single market for data, open to everyone where personal as well as non-personal data, including sensitive business data, are secure and businesses also have easy access to an almost infinite amount of high-quality industrial data, boosting growth and creating value, while minimising the human carbon and environmental footprint. Additionally, Europe's data strategy relies on a healthy ecosystem of private actors and data markets, planning to fund the establishment of EU-wide common, interoperable data spaces in strategic sectors. Such spaces should focus on overcoming legal

[6] https://ec.europa.eu/digital-single-market/en/content/european-digital-strategy
[7] https://ec.europa.eu/info/strategy/priorities-2019-2024/europe-fit-digital-age_en#latest

and technical barriers to data sharing across organisations, by combining the necessary tools and infrastructures and addressing issues of trust, for example by way of common rules developed.

In this direction, the GAIA-X initiative[8] aims to develop the foundations and set the standards of a federated, secure, open data infrastructure that strengthens the ability to both access and share data securely and confidently. An open digital ecosystem for companies and individuals that will allow both the digital sovereignty of cloud services users and the scalability of European cloud providers. The project is supported by representatives of business, science and administration from France and Germany, together with other European partners and is in line with the "data-as-infrastructure" approach to the Data Economy, which aims at a negotiated, common data governance setting between the EU and like-minded partners. A central element of the GAIA-X architecture is the International Data Spaces (IDS) standard, which enables open, transparent and self-determined data exchange.

The IDS (also known as Industrial Data Spaces) was proposed in 2019 by the IDS Association[9]. Its aim is to enable a "network of trusted data" built on the core principles of sovereignty, protection of confidence, decentralization and security. IDS leverages existing standards and technologies, as well as accepted governance models, to facilitate the secure exchange and easy linkage of data in a trusted business environment. Inspired by the reference architecture for Industry 4.0 (RAMI), the Industrial Data Space reference architecture model[10] (IDS-RAM) conceptually supports the establishment of secure data supply chains from the lowest layer (i.e. the data source) to the highest layer (i.e. data use), and thus, sets the standard for building data-driven ecosystems, products and services.

An evolving example of an actual Data Space based on IDs and GAIA-X, related to the European industry, is the Catena-X Automotive Network[11] founded by leading German enterprises like BMW, Deutsche Telekom, Robert Bosch, SAP, Siemens and ZF Friedrichshafen. The companies involved want to increase the automotive industry's competitiveness, improve efficiency through industry-specific cooperation and accelerate company processes through standardization and access to useful information.

A recently released position paper by the International Data Spaces Association compares GAIA-X and IDS finding that GAIA-X infrastructure is not yet as advanced as the International Data Spaces (IDS) initiative. Yet in terms of proliferating data sovereignty, both initiatives have the same perspective. The same goes for the vision for creating an ecosystem of trust for data sharing. The two differ though in concepts that incorporate data storage and cloud-related elements. Currently, those are part only of GAIA-X but could serve to complement the IDS architecture in the future.

As one can easily observe, the keywords "trust", "security" and "data sovereignty" appear in many of the aforementioned descriptions. Indeed, they can be regarded as prime elements of a modern data sharing environment. All of these words are also encountered in texts concerning the distributed ledger technology (DLT), a database infrastructure based on a computer peer-to-peer network that can record data and transactions and replicate them as identical copies in multiple independent devices (nodes). This technology is now being leveraged in the design of the decentralised trading platforms as it offers numerous advantages compared to traditional systems. Among others, it addresses the need to enforce transparency and data democratisation, while at the same time it eliminates the single-point of failure and single target for hackers that exists with centralised systems (Özyılmaz, Doğan, and Yurdakul 2018). Various types of distributed ledger technology are currently in use, with Blockchain being the most widely known. Blockchain, which bundles transactions into blocks that are chained together, and then broadcasts them to the nodes in the network, allows and supports the usage of cryptocurrency models and smart contracts (Daniel & Luca, 2019; Golosova & Andrejs,

[8] https://www.data-infrastructure.eu/GAIAX/Navigation/EN/Home/home.html
[9] https://www.internationaldataspaces.org/
[10] https://internationaldataspaces.org/use/reference-architecture/
[11] https://catena-x.net/en/

2018). Hence, it promotes a high level of transaction integrity, security and data sovereignty, as well as flexible economic models, making it a suitable match for data and AI asset marketplaces.

2.1.4.2 Data and AI Assets Sharing

This section reviews and investigates the state-of-play in the field of data and AI asset sharing. But since this field is too broad to attempt an exhaustive analysis, the work conducted is more focused on B2B big data marketplaces, as well as on best practises concerning sharing agreements and data contracts. Moreover, a thorough analysis will be provided concerning the manufacturing domain with its unique implications, requirements and challenges.

In a recent article (Jones, 2021), Lydia Clougherty Jones, Senior Director Analyst in Gartner claims that "Data sharing is the way to optimize higher-relevant data, generating more robust data and analytics to solve business challenges and meet enterprise goals". According to the same source, organisations that promote the sharing of data and "data products" (AI assets) will outperform their peers on most business value metrics, by 2023. Yet, at the same time Gartner predicts that through 2022, less than 5% of data-sharing programs will correctly identify trusted data and locate trusted data sources.

Indeed, the growth of the digital economy has made the sharing of data between various stakeholders very crucial for generating value in terms of intra-organisational efficiency, inter-organisational practices, and even solving problems and improving living standards for the general public. The open data paradigm has also motivated new ideas and initiatives for sharing closed data. In this context, big data and AI marketplaces have received a lot of attention as of late in the research community (Koutroumpis, Leiponen, & Thomas, The (unfulfilled) potential of data marketplaces, 2017).

In general, sharing data exhibits similar characteristics to those for sharing patents and ideas, in a sense that they are not rivalrous in use, and a single idea or datum may be usable by many individuals and replicated at low or zero marginal cost (Koutroumpis, Leiponen, & Thomas, Markets for data, 2020). On the other hand, AI assets, like trained AI models, can be rivalrous (in terms of performance), but also require some form of maintenance, in contrast to data where little or no maintenance is needed after the transfer to a consumer (Kumar, Finley, Braud, Tarkoma, & Hui, Marketplace for ai models, 2020).

In the lines to follow, we will delve deeper into the field of data sharing and data contracts, while AI marketplaces are discussed in more detail in section 2.1.4.3.

Big data sharing

The current big data sharing landscape includes generic purpose platforms provided by large cloud computing companies (e.g. Microsoft Azure and Amazon), open data platforms with market features, and vertical marketplaces, i.e. within-industry targeted to specific types of data or domains of application. The abundance of data (especially with the advent of IoT sources), combined with the integration of cloud-based data analytics and visualisations in a secure environment have strengthened these marketplaces as big data sharing enablers that enhance business operations across a variety of domains and applications. As such, the problem of big data sharing cannot be seen independently of data marketplaces.

Typically, these platforms offer an infrastructure for data exchange by acting as intermediaries that create a link between data providers and data consumers. Consumers are encouraged to share their data needs because the market will solve the *discovery* and *integration* problem for them, often in exchange for some form of currency.

Classifying data marketplaces has been the subject for many research papers in the last few years (Stahl, Schomm, Vossen, & Vomfell, 2016; Koutroumpis, Leiponen, & Thomas, The (unfulfilled) potential of data marketplaces, 2017; Spiekermann, 2019; Fruhwirth, Rachinger, & Prlja, 2020). The most recent work, that of Fruhwirth et al., reviewed twenty operational marketplaces and identified

a large number of common characteristics, which eventually defined a business model taxonomy with four data marketplace archetypes:

- *Centralized data marketplace:* This type of data marketplace has similar characteristics to conventional online marketplaces, offering possibilities to exchange data in a simple, yet efficient, manner. Data marketplaces of this archetype do not rely on a specified data origin, data domain, data output type, or pricing model.
- *Centralized data marketplace with smart contract:* This type supports straightforward trading of data, while addressing security and legal issues that can occur in centralized data marketplaces.
- *Decentralized data marketplace:* This archetype relies on decentralized infrastructure typical for smart contracting. Marketplaces of this archetype guarantee data quality and usually offer self-generated, dynamic data (e.g. from IoT sensors).
- *Personal data marketplace:* This archetype has a consumer-to-business characteristic and operates with user-generated personal data. Data trading is performed through use of simple, specialized software.

According to (Koutroumpis, Leiponen, & Thomas, The (unfulfilled) potential of data marketplaces, 2017) there are four main types of data marketplaces based on how consumers and providers are matched:

- *One-to-One*: This type of matching expects a bilateral relationship, common in, which is typically how data brokers operate and is characterised by negotiated terms of exchange. Markets of this type have low transparency and can be rather inefficient since the secrecy involved in these transactions hinders the finding of possible trading partners. On the other hand, data traded in these marketplaces are of high value and high confidentiality, which is accompanied by high transaction costs due to costs of search, negotiation, and ongoing relationship management.
- *One-to-Many*: This type is encountered when a single provider transacts with many consumers for the same data, e.g. when data is distributed through APIs to many interested consumers, like the Twitter API. In this dispersal marketplace the terms of exchange are often standardised, since the costs to individually negotiate each exchange could be very high. Therefore, transaction costs tend to be low, while provenance and boundaries are unclear. Data is usually of low value and confidentiality.
- *Many-to-One*: These marketplaces are characterised by the “harvesting” of data of many users by one service provider. This provider usually offers a free service in exchange for data (e.g. Google search, Facebook). The underlying rules are weak, with minimal monitoring, and transaction costs can be very low. This marketplace type is vulnerable to repugnance concerns such as norms related to privacy. As an example, the General Data Protection Directive (GDPR) gives users the “right to be forgotten”, which, if mass exercised, could inflate the costs associated to monetising user data.
- *Many-to-Many*: This type of markets refers to multilateral sharing platforms upon which anybody (or at least a large number of users) can upload and maintain datasets. They can be centralised or decentralised and there are varying licensing models – standardized or negotiated – to regulate data trading. These marketplaces often emphasise on data discoverability and other facilitation activities, including online payment. Transaction costs, as well as data confidentiality and value vary. In their most common form, these platforms do not have ownership of the data, but only act as intermediaries that facilitate transactions.

Table 2-1 Types of marketplaces based on matching mechanism (Koutroumpis, Leiponen, & Thomas, The (unfulfilled) potential of data marketplaces, 2017)

Matching	Marketplace design	Terms of Exchange	Examples
One-to-one	Bilateral	Negotiated	Data brokers
One-to-many	Dispersal	Standardized	Twitter API
Many-to-one	Harvest	Implicit Barter	Google Services
Many-to-many	Multilateral	Standardized or negotiated	Azure Marketplace

Multilateral markets may provide several desirable features over other matching models, and, potentially enable economies of scale and scope, innovation, transaction and search. Following the work of (Thomas & Leiponen, 2016), such features can be the following:

(a) clearly defined boundaries that enables the identification of a legitimate user;

(b) rules regarding how data resources can be utilized off-line or outside of the transaction;

(c) opportunities for contributors to participate in the development or improvement of the platform;

(d) effective monitoring by a group of core users or a third party accountable to the core users;

(e) rules that define how the resources are to be used and the penalties for not doing so;

This methodological framework may enable value creation and sharing due to the public-good properties of big data, as well as high flexibility in terms of involved stakeholders and data trading options. In this context, there is an emerging need to develop and maintain contract engines that provide querying and validation mechanisms for access to and usage rights of data and AI assets, as well as the status of the agreements being performed.

Data contracts and sharing agreements

In order to facilitate data sharing in environments where the data is not necessarily open, the support of some form of contract or agreement for data sharing is crucial, especially when dealing with sensitive information. A Data Sharing Agreement (DSA) is a human-readable, yet machine-processable contract, that sets out the purpose of the data sharing, sets standards to meet the requirements of the data protection principles and covers what happens to the data at each stage, helping all the parties involved to understand their role and responsibilities (Caimi, Gambardella, Manea, Petrocchi, & Stella, 2015).

A systematic review of the landscape reveals a lack of standardization, as well as an increased confusion regarding intellectual property and ethics related to sensitive data (Grabus & Greenberg, The Landscape of Rights and Licensing Initiatives for Data Sharing, 2019). Nevertheless, there is a substantial number of publications that address certain aspects and specific challenges related to one type of data and/or one particular domain. For example, (Balint & Truong, 2017) introduced a new extensible platform for enabling contract-aware IoT dataspace services, which supports data contract specification and IoT data flow monitoring based on established data contracts, whereas (Sakr, 2018) studied the challenges related to spatial data sharing and proposed a model and query algorithms for this type of platform.

In addition, various research projects related to data management and decision making have studied and investigated the issue of data sharing agreements. The EU project AEGIS, for instance, has defined a conceptual policy and brokerage framework, which covers several aspects related to data assets rights, data quality, policies and pricing models (AEGIS Deliverable D2.1 "Semantic Representations

and Data Policy and Business Mediator Conventions", 2017). The presented framework aims to cover numerous trading workflows for various virtual assets, including datasets and data-as-a-service, but also spans to microservices, algorithms and analytics reports, hence it is inherently more generic.

On the other hand, the NSF Spoke project "A Licensing Model and Ecosystem for Data Sharing"[12] explored and tried to address various data sharing challenges, aiming to provide a data sharing framework with the following features:

(1) A licensing model to facilitate data sharing between different organisations.

(2) A prototype data sharing software platform (ShareDB).

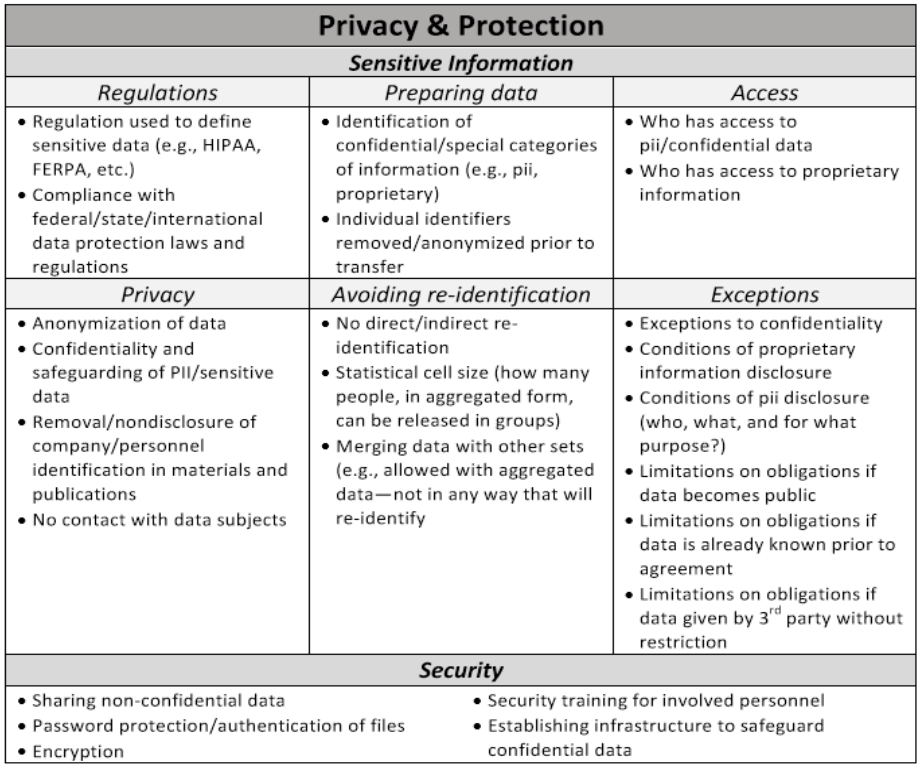
(3) Relevant metadata that will accompany the datasets shared under the different licenses, making them easily searchable and interpretable.

During the course of this project, the authors of (Grabus & Greenberg, Toward a metadata framework for sharing sensitive and closed data: an analysis of data sharing agreement attributes, 2017) performed an analysis on 26 data sharing agreements collected from industry, academia and government in order to reduce the complexity of all the involved regulations and policies into a few sets of attributes. They identified six high-level, partially overlapping, metadata categories that can support the development of data sharing agreements:

- *General*: attributes related to the project and the agreement itself, e.g. description of data, definition of terms
- *Privacy & Protection*: the protection of sensitive information and security
- *Access*: the definition of who can access the data and how this can be done, including approved software and hardware
- *Responsibility*: legal, financial, ownership and rights management pertaining to the data, e.g. indemnity clause and establishment of data ownership
- *Compliance*: ensuring fulfilment of agreement terms, e.g. third-party compliance with contract
- *Data Handling*: specifics of permissible interactions with the data

It should be clarified that the term data license in this context is not limited to predefined data licenses, such as Creative Commons, CDLA and Open Data Commons and refers to a broader concept. For each of these aspects, a set of more fine-grained attributes is also collected and explained. Indicatively, for the "Privacy & Protection" category, the attributes depicted in the following figure were identified.

[12] https://cci.drexel.edu/mrc/research/a-licensing-model-and-ecosystem-for-data-sharing/



Privacy & Protection		
Sensitive Information		
Regulations	*Preparing data*	*Access*
• Regulation used to define sensitive data (e.g., HIPAA, FERPA, etc.) • Compliance with federal/state/international data protection laws and regulations	• Identification of confidential/special categories of information (e.g., pii, proprietary) • Individual identifiers removed/anonymized prior to transfer	• Who has access to pii/confidential data • Who has access to proprietary information
Privacy	*Avoiding re-identification*	*Exceptions*
• Anonymization of data • Confidentiality and safeguarding of PII/sensitive data • Removal/nondisclosure of company/personnel identification in materials and publications • No contact with data subjects	• No direct/indirect re-identification • Statistical cell size (how many people, in aggregated form, can be released in groups) • Merging data with other sets (e.g., allowed with aggregated data—not in any way that will re-identify	• Exceptions to confidentiality • Conditions of proprietary information disclosure • Conditions of pii disclosure (who, what, and for what purpose?) • Limitations on obligations if data becomes public • Limitations on obligations if data is already known prior to agreement • Limitations on obligations if data given by 3rd party without restriction
Security		
• Sharing non-confidential data • Password protection/authentication of files • Encryption	• Security training for involved personnel • Establishing infrastructure to safeguard confidential data	

Figure 2-5 Privacy & Protection Attributes of Data Sharing Agreements (Grabus & Greenberg, Toward a metadata framework for sharing sensitive and closed data: an analysis of data sharing agreement attributes, 2017)

The aforementioned six categories appear more or less in various other studies in literature (Truong, Comerio, De Paoli, Gangadharan, & Dustdar, 2012; Van Panhuis, et al., 2014), with each one focusing on the metadata attributes most important for the specific underlying data sharing problem. The outcome of these research studies is that the number of metadata properties that will be required and their level of sophistication depends on the overall scope of the data sharing system to be developed. When metadata properties can be organised in a set of predefined value groups, efficient and easier to implement solutions can be designed.

Data Sharing in manufacturing

The Manufacturing domain, and industry in general, is certainly impacted by the great potential of data exploitation, that is currently facilitated by a larger availability of data (deriving for instance from Industry4.0 paradigm with sensors deployment), but that can be boosted also by data sharing among enterprises, that put at disposal an even larger amount of data.

Having in mind the “smart factories” that generate and collect information in each production step (from planning to product development, thanks to the increasing number of sensors available in the plant), it is easily understandable why nowadays there is abundancy of data in industries. What should be carefully considered is the use of such data by the companies. Actually, it happens that different departments of a plant are siloed and data is not exchanged between them. As a result the data can lose its potential.

This situation makes even more difficult sharing data with external partners. However, leveraging not only on own data, but also on information from other companies, is fundamental to unlock new data analytics opportunities from which the company itself can benefit, since it allows to have at disposal a larger amount of data of different nature.

Overcoming the first issue of data siloed inside the company, in a process of data sharing, the next step is to create trust between the data owner (sender) and data receiver, guaranteeing the sovereignty of the former. The solution that is taking shape is to use the same technology or platform

both for sender and receiver, where the data owner can specify rules on how the data may be used by the data receiver – and even enforce compliance with these rules on a technical level.

This concept stands at the base of Data Space implementation.

Aware of the expected economic impact deriving from data exploitation, the European Commission launched a series of guidelines to support European private and public sectors, including manufacturing, toward data sharing. In February 2020, the Commission announced the adoption of a European strategy for data[13], which aims at creating a single market for data to ensure Europe's global competitiveness. Several industrial sectors are directly mentioned in the paper, among those manufacturing, for which the EC is boosting the rollout of dedicated European Data Spaces.

On 23th November 2020, in the webinar "Data Space for manufacturing – current state of play"[14], the European Commission invited experts, companies and business associations to discuss about the current situation of Data Spaces in the manufacturing domain, identifying the obstacles and challenges to be overcome. Actually, in manufacturing, as well as in other domains, achieving a full data exploitation is not straightforward and requires to address several issues: economic support to finance the digital transformation, mindset change in companies' management to make the transformation start and in workers to make accept it; in addition, a number of technological barriers are emerging, to be overcome with new investments in research and development. Presenting the Commission strategy for boosting a European data economy, the DG CNECT put in light which are the main problems currently affecting Europe:

- Absence of comprehensive data governance approaches;
- Lack of European data processing and storage solutions;
- Lack of the required digital skills;
- Poor data reusability, due to low data availability from the public sector, to scarce confidence in sharing data with third parties and lack of secure infrastructure to do it.

Last bullet point is strongly related to the concept of Data Space: regarding manufacturing domain, as of now the main objective is to set up and deploy several operational data spaces for specific value chains, which enables companies in different user roles (supplier, client, service provider, etc) to interact with large amounts of manufacturing data. In particular, the deployment of a European Data Space for manufacturing is planned for the end of the year (2021) and it will focus mainly on two use cases: Supply Chain Management and Predictive Maintenance. In addition, various research projects funded by the European Commission deal with private data spaces and data sharing for manufacturing.

The European Industrial Data Space (EIDS) is based on the International Data Spaces Reference Architecture, which was developed by the International Data Spaces Association (IDSA). IDSA is "a coalition of more than 130 member companies that share a vision of a world where all companies self-determine usage rules and realize the full value of their data in secure, trusted, equal partnerships". The association is not specifically focused on manufacturing domain, however, due to the generic definition of the reference architecture model (RAM), manufacturing can benefit from it.

Unlike with other platforms, users of EIDS do not need to create an account to be granted access. Instead, each user must install the IDS Connector, and all technical components are certified by IDS and respect IDS standards. Each participant has a "trust level" and it is possible to established a set of rules governing the exchange of information according to the trustworthiness of the receiver but also on the confidentiality of data shared.

[13]https://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1593073685620&uri=CELEX:52020DC0066
[14] https://digital-strategy.ec.europa.eu/en/events/data-spaces-manufacturing-current-state-play

In addition, the IDS Reference Architecture requires no central data storage entity; instead, the data physically remains with the respective data owner.

According to the "Reference Architecture Model" paper[15] published on April 2019, the proposed RAM is based on five layers: at the top, the business layer, where the participants' roles are defined, the functional layer, that specifies the functional requirements of a Data Space, the process layer, to define interactions among participants, the information layer, to define the conceptual model, and at the bottom the system layer, that considers aspects such as integration, configuration, deployment, and extensibility of these components. As the picture below shows, the five layers are developed taking into account the security, certification and governance point of view.

The OPEN DEI[16] initiative has gathered the accumulated knowledge in the paper 'Design Principles for Data Spaces' that represents a milestone in the evolution of Europe's data economy. This is about how to build data spaces and make them work collaboratively across multiple industries and includes the perspectives of all the key stakeholders.

Three main layers are identified:

- The software infrastructure, at the lowest level, conceived to be domain independent and agnostic.
- The Manufacturing data space, in the middle.
- The Manufacturing ecosystem, at the top.

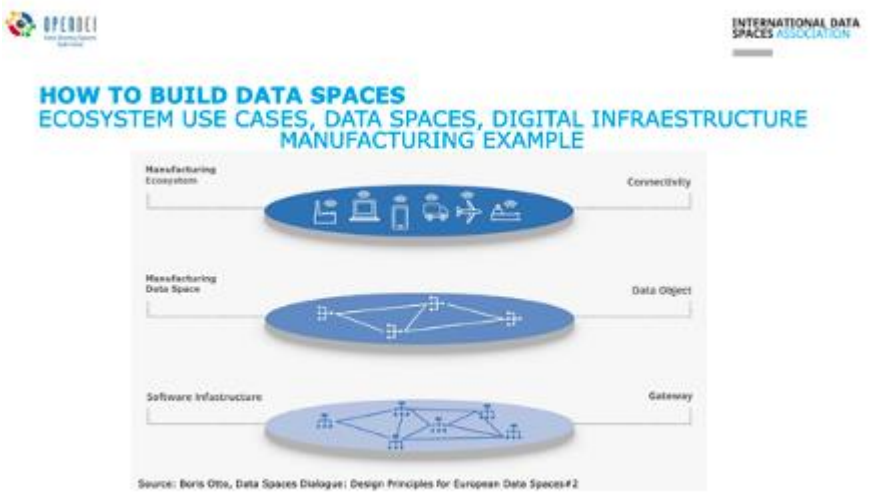


Figure 2-6 Manufacturing Data Space (three levels)

2.1.4.3 Data and AI Assets Trading

As already discussed, data and AI assets can be treated as resources that can be shared in a market, mainly for academic and research purposes. However, as a strategic resource for enterprises, data and AI assets can be considered a tradable commodity. A commodity that has an economic value and generates costs due to its management. This has fuelled the emergence of a digital asset trading industry in recent years whose primary business model is comprised of the trading of raw and processed data and the provision of AI-related services and products.

[15] https://internationaldataspaces.org/publications/ids-ram/
[16] https://www.opendei.eu/projects/

Although there is an abundance of information regarding the aspect of data trading and pricing, the same is not true, yet, for AI related products, even though the need for AI asset sharing is constantly increasing. The concept of transfer learning has played a vital role in this. Transfer learning is a machine learning method where a model trained for a task is reused as the basis (pre-trained model) to be re-trained and further refined on a different but related task (Lu, et al., 2015). This method is used frequently in natural processing language problems, but has been adopted also by other domains, such as the manufacturing domain (Ferguson, Ronay, Lee, & Law, 2018; Wen, Gao, & Li, 2017), especially when the availability of useful data is scarce. Using pre-trained models for commercial purposes is one of the primary motives behind AI Marketplaces.

AI Marketplaces

The research on online marketplaces for AI models, AI results and insights, and other data related products is still in its infancy. Very few papers tackle this issue and most of them is from a theoretical point of view. For example, (Kumar, Finley, Braud, Tarkoma, & Hui, Sketching an AI Marketplace: Tech, Economic, and Regulatory Aspects, 2021) envision an AI marketplace as a place that facilitates the buying and selling of AI models among different actors, such as AI developers and AI customers.

Table 2-2 Key roles in an AI marketplace as envisioned in (Kumar et al. 2021)

Role	Description
Developer/Scientist	An entity which sells pre-trained AI models or develops customized AI models
Customer	An entity which purchases AI models, or uses services of Developer to get customized AI models
Data Owner	An entity which sells their data, or training data updates for training AI model
Auditor	An entity which verifies the correctness of AI model developed by the Developer
Regulator	An entity which ensures marketplace guidelines/regulations are being respected

Following their previous work in (Kumar, Finley, Braud, Tarkoma, & Hui, Marketplace for ai models, 2020), the authors discuss technical, economic, as well as regulatory aspects surrounding the development of an AI marketplace. The key points researched, as well as the recommended methods, can be summarized as follows:

(1) *On **Technical** aspects*. The most important technical aspect is how to maintain Data Privacy in an AI marketplace context. Several major strategies can ensure data privacy for the involved actors with the most prominent being:
   i) *Federated/ Peer-to-Peer learning as learning paradigm*, by leaving data on the data owners' devices and partially train AI model on each device.
   ii) *Using contextual integrity as a design principle for data sharing*, by allowing users to maintain control of their data even after trading it.
   iii) *Zero knowledge as a design principle*, by making mandatory the expiration and deletion of datasets used for training after a set date.

(2) *On **Economic** aspects*. The most important issues identified are:
   i) *Network effect*. This is a phenomenon where the value of a platform or service is proportional to the number of participants and is often the most important success indicator. In order to be able to compete against well established companies, a new AI marketplace needs to support interoperability between datasets and models so that developers can efficiently aggregate enough smaller training datasets, federated training users, or even models (into an ensemble) to compete or at least reach a minimum performance threshold.

ii) *Disintermediation***.** This is a tactic where a buyer and a seller agree to conduct subsequent transactions outside the platform they first met. The risk of disintermediation could be minimized in an AI marketplace if the platform could offer additional added-value services like AI auditing and traceability.

iii) *Information asymmetry***.** It is a phenomenon where a consumer may feel uncertain over the quality of the sellers/developers. A common solution to this problem is a trust or reputation mechanism to help facilitate transactions between strangers.

(3) *On **Regulation** aspects***.** The laws and regulations on AI and data marketplaces are still evolving and developing in most parts of the western world. Of course, there are historical precedents regarding general online marketplaces or sensitive data, but these do not cover clearly AI specific issues, like liability (i.e. when an AI model causes harm, who is to blame?).

Relying on these principles, an H2020 project called "Bonseyes" developed a cloud-based marketplace for AI products. Their research work on privacy and trust resulted in the concept of the "Virtual Premise" (Mehri & Tutschku, 2017). As stated in their publication a Virtual Premise (VP) provides an area where privacy measures are enforced in a cloud platform, i.e., where controlled access and monitoring of data usage is executed. The VP aims to extend local computation (at premise) when this is not sufficient, and data needs to be transferred securely to a remote location.

Making knowledge a tradable asset is also discussed in (Lin, Li, Wu, Liang, & Yang, 2019), where the analysis focuses on edge-AI assets involving IoT data. A knowledge consortium blockchain for secure and efficient knowledge management and trading for the market is developed, which includes a new cryptographic currency, smart contracts, and a new consensus mechanism named 'proof of trading'.

Other interesting approaches in this field, include the Genesis AI platform, which introduced a machine learning protocol, on top of which a decentralized marketplace for AI products and services is built (Cheishvili & Fan, 2018), and the singularityNet (Goertzel, Giacomelli, Hanson, Pennachin, & Argentieri, 2017) which facilitates market interactions with AI and ML tools. Both are, by origin, based on the Ethereum blockchain.

Data and AI asset pricing models

Solving the problem of pricing lies at the core of the data and AI asset trading research field. It is true that data or AI products exhibit different characteristics compared to tangible goods, a fact that prevents the direct transfer of established processes and rules of trading products, especially in terms of pricing mechanisms. One such characteristic is that data is often rather costly to initially collect (and AI models are, likewise, costly to develop and train), and very cheap to copy or disseminate.

In data and AI asset trading, the willingness to pay is lower. For example, it is difficult to convince potential buyers of the value of data items without actually showing the data prior to purchase (known as the 'Arrow paradox') (Koutroumpis & Leiponen, Pantelis, Koutroumpis, and Leiponen Aija. "Understanding the value of (big) data, 2013; Stahl, Schomm, Vomfell, & Vossen, 2015).

Usually, considering costs is the only rule for pricing a commodity, especially for digital commodities. In fact, only considering cost is a common defect, and should be just one factor of reasonable pricing. Thus, the factors of commercial price measurement for a digital commodity are developing cost, collocation or analytics cost, and maintenance cost.

Regarding data pricing there are many recommendations and practices. Indicatively, a set of fixed payment plans that could be used as pricing models were identified by (Cao, et al., 2016):

- Payment on package delivering (API handle): data is split into separated packages (e.g., messages or images) and consumers are charged each time they successfully receive a number of packages from the marketplace.
- Payment on data size: consumers are charged based on the size of the received data, e.g. per MB, GB, etc.

- Payment on time of subscription: This model is appropriate when the data (usually streaming data) is generated on a duration of time (e.g. every day between 1pm and 3pm) and each unit of time has a predefined cost. So, consumers are charged based on the total time they spent subscribed to the specific time unit.
- Payment on data unit: providers split their data in different data units and set up the basic unit charge fee for each. Consumers will pay one time and get the data until reaching the limitation of unit.
- Payment on plan (fixed payment on a period): consumers subscribe to use data in a subscription period (e.g., a week or a month) and only pay one time for this period with or without maximum limitation of received data.
- Free usage: consumers can use these services at no charge from providers.

The authors in (Fricker & Maksimov, 2017) reviewed a number of related papers on data products pricing and identified 3 key quality attributes when determining the price:

- Time, in terms of a dataset being up-to-date
- Accuracy, in respect to how much the dataset captures the problem and is free from noise.
- Completeness, in respect to missing parts or erroneous values

In the same literature review paper, it was revealed that price determination could also be derived from equations or algorithms (e.g. binary tree sampling), with most of the papers suggesting that pricing is an NP-complete problem in general. For instance, (Heckman, Boehmer, Peters, Davaloo, & Kurup, 2015) defined a set of attributes based on which they constructed ML regression models that help establish a correlation between data attributes and the price of a given dataset. These attributes, taken directly from this work, are as follows:

- Value-based parameters (value of data to the consumer):
  - The value of the data in terms of saving in time, effort, or money
  - The ROI for the customer
  - Risk exposure, i.e. inducing higher costs for data cleansed of personally identifiable information and privacy violations
  - Data exclusivity
  - Level of ownership, ranging from transfer of ownership to allowing use for a fixed time to allowing limited use for a specific purpose
- Qualitative parameters (attributes or meta-attributes of the dataset):
  - Age of the data
  - Credibility of the data
  - Accuracy of the data elements
  - Quality of the data
  - Format and level of structure of the data
- Fixed and marginal cost parameters (directly measurable cost):
  - Cost of collecting the data
  - Cost of data storage, bandwidth, and other operational costs
  - Cost of data-as-a-service offerings – add-on services to process the data, computing resources for the data, analytic reports, or aggregation on the data
  - Delivery cadence – one-time, batch, or continuous basis

From this perspective, several attributes examined to define the appropriate pricing strategy depend on the other term categories, such as the data rights regime and the data quality.

Another methodology for estimating the value of data, especially when updating incrementally already acquired data, is the subject of study in (Dalessandro, Perlich, & Raeder, 2014). The conclusion is that any decision to invest in acquiring and using data with ROI as the primary criterion, should be

endorsed by mathematical and scientific means. Finding the optimal pricing scheme is also investigated in (Niyato, Alsheikh, Wang, Kim, & Han, 2016) in the context of big data and IoT. Finally, in (Agarwal, Dahleh, & Sarkar, 2019) the authors recommend market mechanisms able to price training data and match buyers to sellers at the same time, while addressing the associated computational complexity.

Regarding AI asset pricing, very little information exists in literature. One example is (Chen, Koutris, & Kumar, 2019), where the authors propose a model-based pricing framework which directly prices ML model instances, instead of pricing the data. Probably, the concept of buying and selling AI products is still at its very early stages or the complexity of the IPRs involved are severely hindering any progress. One other possible reason could be that the research institutes propose always newer and better algorithms and architectures, ahead of enterprises, and prefer to share their work instead of commericializing it.

2.1.4.4 Key Challenges & Considerations

Data marketplaces are becoming increasingly popular in theory and practice. Nevertheless, a number of trust related and technical challenges remain for the industrial sector, as summarized in Table 2-3. The fear of what happens to the data after being shared or the possibility of disclosing valuable or sensitive business information, and thus losing a competitive advantage, are the main considerations for potential data providers according to WEF (World Economic Forum, 2020). On the other hand, the costs related to adopting new technologies and the fear of a security breach are the most prominent obstacles from a technical perspective.

Table 2-3 Barriers to data sharing in industry according to recent whitepaper (World Economic Forum, 2020)

Trust related barriers	Fear of unintentionally giving away valuable or sensitive data about the business
	Fear of losing negotiation power or a competitive advantage
	Lack of visibility into data usage and analysis once shared
Technical barriers	Risk of data breaches and losses
	Accessibility and interoperability issues that arise from combining data
	Different digital maturity levels among participants in the same solution
	Costs of switching technologies (or fear of technological lock-in)

Regarding AI products sharing and trading, Kumar et al has highlighted a few key considerations for establishing a successful AI marketplace:

- Developing AI models often requires the use of data that may be proprietary and or sensitive. Therefore, an AI marketplace should have a mechanism that ensures that model developers use that data only for training purposes and alleviate any privacy concerns.
- Carefull consideration should be given on cases where the datasets that trained an AI model cannot be provided for privacy or other reasons, as the traceability and auditability of models from these sources becomes cumbersome.
- An AI marketplace also requires a mechanism that can determine the quality of a shared AI model. Conventionally, accuracy has been the primary metric. However, alternative metrics that capture reliability, robustness, and fairness are also now considered important.
- Since AI models are practically pieces of software, they may also require maintenance over time for various reasons e.g. to use updated versions of libraries or be compatible with new privacy regulations. So, an AI marketplace needs standard guidelines that AI developers should follow while developing models for the marketplace, so that maintenance by other AI developers can be much easier.

- Developing a comprehensive reputation system to rate AI models and/or users is also recommended, in order for a platform to maintain a high level of quality products and protect from malicious AI scripts.

The goal for the XMANAI project is to examine the current landscape and address these challenges in the most efficient way in order to build a successful AI and data marketplace.

2.1.5 Missing Data Acquisition

2.1.5.1 Background

Data quality is one of the most crucial problems in the world of data management (Fan & Geerts, 2012). In order to have and maintain high quality datasets, a number of processes are required to run on a frequent basis. Correcting errors in the data or cleaning duplicates, inconsistent or inaccurate values are among the common tasks. Nevertheless, improved data quality comes also from data enrichment and the acquisition of missing information, usually from external sources, as well as the proper handling of missing values.

Data enrichment is the process of combining and merging multiple different data sources together. These data sources can be first party data from an internal software system, disperse data from other internal systems or third-party data from external sources. The outcome of data enrichment is a unified source that is more valuable and useful than its constituents and thus, it may become a valuable asset for any organization. In the world of Big Data, data acquisition and enrichment are accomplished by integrating taxonomies, ontologies, and third-party libraries as a part of the data processing architecture (Krishnan, 2013). For this to happen, a process for finding and collecting appropriate data from various sources by detecting patterns and excluding irrelevant information is required. This process is usually called Data exploration (when you search at random) or Data discovery (when you know what you are looking for).

Missing information could also appear in the form of missing or incomplete observations in a given dataset. Missing data may occur in a variety of domains, for several different reasons, and each domain may handle this implication in a domain-specific way as deemed appropriate. The two most common strategies for managing missing data are to delete the incomplete observations, or replace the missing values by inferring them from the existing part of the data, also known as data imputation. The former risks losing valuable information, while the latter may cause bias (Cismondi, et al., 2013). In order for imputation to be less biased, several techniques have been introduced in the literature that employ AI algorithms in order to synthetically generate the missing values and emulate certain key information found in the actual data while providing the ability to draw valid statistical inferences. These synthetic datasets are an attractive framework to afford widespread access to data for analysis while mitigating privacy and confidentiality concerns (Raghunathan, 2021). Recent publications in the field of data synthesis aim to address these challenges and increase the performance of imputation algorithms (Santos, et al., 2019).

A special challenge under this notion, is data labelling. Usually done by human experts (or not so experts), data labelling is the process of assigning a predefined label to an observation by identifying the meaning of it, for example when tagging whether an image is showing a cat or a dog. This “class” assignment is particularly important for classification problems where an AI model has to be trained on past labelled data.

2.1.5.2 Data Exploration & Discovery

The first step towards data enrichment is to explore the available data sources within the company, comprehend their structure and assess both their quality as well as their relevance to specific business objectives. Data exploration refers to understanding the data and creating insights in order to generate/ test hypothesis, justify the application of statistical methods and set the basis for further

data acquisition. This process includes, among others, the identification of data types and schema, the visualization of value distributions and summary statistics, the detection of redundant or corrupt data and the identification of outliers. Dimensionality reduction techniques such as principal component analysis or embeddings are often applied in high-dimensional data, allowing to uncover the most informative components that suffice to explain most of the variance in the data. An important aspect in data exploration is to search for patterns in the data and interrelations between variables or entities, with the use of univariate, bivariate or multivariate analysis, pattern recognition and ML techniques. Clustering is a common technique applied to this end (Ianni, Masciari, Mazzeo, M, & Zaniolo, 2020). Another important aspect is the localization and quantification of missing data. Pattern mining is performed on the missing data as well, in order to define sources of missingness, that is, whether data is missing at random or in an interdependent manner.

Having analyzed both the existing as well as the missing data, the data scientist can decide on the best strategy to handle the missing information and fill gaps in the dataset. It is also possible to query and discover internal or external data sources with similar characteristics, to complement or augment the dataset under study. Common strategies to handle missing values in the data include:

- The deletion of entries with missing values, in case their number is relatively small as compared to the size of the dataset. Attributes/variables with a high portion of missing values may also be discarded.
- The imputation of missing values with the mean/mode/median.
- The imputation of missing values with values predicted by ML algorithms or extracted from similar entries based on similarity measures.
- The missing values can be acquired from discovered datasets with matching schemas, by querying on the existing values. In case more than one such dataset exists, conflicts between different possible values must be resolved, or a “majority voting” strategy can be adopted.

In order to enable data fusion from multiple sources, entity recognition and disambiguation are necessary, along with the identification of data schemas, types and vocabularies. Data may also need to be transformed to a common format. In the case of exploring raw or unstructured data, unsupervised deep representation learning can be applied to infer the underlying structure (Ansari & Soh, 2019). Once this is accomplished, different data sources can be integrated through record linkage (Gu, Baxter, Vickers, & Rainsford, 2003) or statistical matching (Gessendorfer, Beste, Drechsler, & Sakshaug, 2018). This also allows to augment data horizontally (adding new attributes), vertically (adding new entries) or hierarchically (stacking data sources). Analyzed data can be further enriched with semantic metadata, a process resulting in added value to the data and facilitating subsequent asset sharing and discovery. When integrating data from multiple sources to model a complex or multistage process, it is particularly useful to model data and domain knowledge in Knowledge Graphs, where nodes represent different data sources and domain knowledge is often imposed in the form of logical rules or constraints (Hogan, et al., 2021).

In the case of industrial Big Data sources, it is computationally inefficient and often infeasible to explore the entire dataset so that data sampling is necessary. Although sampling at random is a common strategy, sampling algorithms from active learning are also employed, such as density sampling, uncertainty sampling and query by committee. It appears that data scientists have the opportunity to create different insights on a given dataset, by using multiple or coupled sampling techniques, enriching thus their understanding of the data and the modelled industrial process (Rojas, Kery, Rosenthal, & Dey, 2017), (Liu & Zhang, 2020).

Data exploration and discovery is a dynamic procedure, since the first analysis results provide insights on how to proceed further, may also reveal additional dimensions to the analysis. In addition, it is essential to constantly monitor the enriched dataset to detect data drifting. For these reasons, it is argued that despite the usefulness of automated tools, it is rather preferable to employ interactive

methods to achieve a thorough and flexible data exploration and discovery (Puolamäki, Oikarinen, Kang, Lijffijt, & Bie, 2020), (Huang, et al., 2018).

2.1.5.3 Synthetic Data Generation

Industrial AI has shown tremendous potential in a wide array of manufacturing applications but still, data availability remains as a major challenge to overcome in order to go beyond the pilot stage, as a considerable volume of research work has been based on the assumption that sufficient and adequate data is available to successfully train and validate models.

ML and Deep Learning in particular require a very large amounts of (mostly labelled) data to achieve proper generalization and avoid overfitting. However, in real manufacturing environments data from different settings, conditions and configurations is often scarce (e.g. different failures, defects, energy consumption), given that these typically represent undesired states of the system and acquiring said data with currently adopted practices tends to be unfeasible from both an economic and operational standpoint. Moreover, labelling raw data is a time consuming and costly endeavour which in this context frequently requires expertise and domain knowledge.

Data synthesis emerges as a viable venue that addresses this challenge. To that end, this field focuses on producing artificially generated data which closely resembles data from real operational environments and thus allows the generalization of Industrial AI solutions based on it to real scenarios. This approach has been successfully employed in research studies related to Industry 4.0 during the last years.

Regarding classification problems, traditional approaches to handle class imbalances due to data availability issues usually involve artificially re-sampling the data set. Typically, this was achieved by under-sampling the majority class (Liu, Wu, & Zhou, 2008), resulting in a balanced distribution. However, with data being the most valuable asset in industry, this method implies that some bundles of data with relevant information can potentially be lost, thus diminishing the predictive power of any ML model. This fact becomes particularly critical for small datasets, where pruning data may not be feasible. Through synthetic data generation, another way to solve this problem is to oversample the minority class. There are many approaches to achieve this, such as the randomized replication of instances of the minority class (Sáez, Krawczyk, & Woźniak, 2016), that achieves a more balanced distribution at the risk of potential overfitting.

Another popular approach is the synthetic minority oversampling technique (SMOTE) (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). The goal of SMOTE is to create synthetic samples to balance an unbalanced dataset. To achieve this, SMOTE oversamples the samples belonging to a minority class without increasing the number of samples from the majority classes. The way this oversampling is done is by interpolating the samples of the classes in their feature space. Thus, the features of the new samples consist of a combination of features from the existing samples. Although SMOTE and its extensions have achieved considerable results in recent years (Fernández, Garcia, Herrera, & Chawla, 2018), the new synthetic samples are very feature space dependent and studies have shown that for high dimensionality problems, their effects are underwhelming (Lusa, 2012).

With the popularization of GANs (Goodfellow, et al., 2014), new opportunities for the generation of reliable synthetic data in manufacturing have arisen. Such architectures are based on the idea of generating fake samples that are not able to be detected as false by a discriminator. Thus, during the training stage, there are two components in the architecture: the generative and the discriminator. Through an iterative process, the generative part will aim to learn the distributions of the original data to generate new false samples and the discriminator, which will be static and will not vary throughout the iterations, will try to be able to detect them. The more fake samples it is unable to detect, the better the generative model will be. Some promising examples of this can be found in the literature, addressing the class imbalance problem in applications in which the data of faulty operational

conditions is scarce and difficult to obtain (Cabrera, et al., 2019) (Mao, Liu, Ding, & Li, 2019) (Zhou, Yang, Fujita, Chen, & Wen, 2020).

Another kind of architecture that has been used for synthetic data generation is the Variational Autoencoder (VAE) model. These structures leverage Autoencoder architecture to generate new samples without the needed of labelling samples, that is unfeasible or too costly in industrial context. The aim of VAEs is to learn a latent distribution from an original complex distribution (given by the available samples in a dataset) through an encoder stage and transform this latent distribution in the original space by means of a decoder phase. These reconstructed samples are the synthetic generated data. The measure employed to control accurate reconstructions is a quantifiable reconstruction error. The lower this reconstruction error is, the better the new sample is.

Recent efforts have also shown promising empirical results on semi-supervised learning (Dai, Yang, Yang, Cohen, & Salakhutdinov, 2017) (Di, Ke, Peng, & Dongdong, 2019) for cases in which labelling the entire dataset is unfeasible or too costly.

2.1.5.4 Key Challenges & Considerations

The acquisition of missing data, either as values or complete sets, aims to enhance the overall analytical power of data. In general, a successful approach on selecting the best strategy for data discovery or for handling missing values relies on having a clear goal of what you want to accomplish and a clear understanding of the data at hand.

Considering data exploration and discovery, it is equally important to provide the right means and meta-data for data fusion and unification across multiple sources of both internal and external nature (Diez-Olivan, Del Ser, Galar, & Sierra, 2019). Sharing data that is easy to discover and integrate is hard because data owners usually lack information (who needs what data) and they do not have incentives to prepare the data in a way that is easy to consume by others (Fernandez, Subramaniam, & Franklin, 2020). Providing the right incentives and the tools to determine relationships between data objects and models in structured, unstructured and semi-structured data is a key challenge for XMANAI, as well as the matching of temporal and spatial characteristics. Knowing these various properties is essential for understanding a data asset and provides greater accessibility to the end user.

Regarding missing data values, there seems to be no perfect way to handle it. One imputation strategy can perform better on particular datasets, and much worse on other types of data. In some cases with plenty of data and few missing values, removing instances might be the best way to go. On the other hand, when dealing with nominal or categorical missing values, traditional techniques offer little or no assistance (Raghunathan, 2021). At this point, it should be mentioned that handling missing data is also part of the data manipulation process, which is performed by WP3 and documented more extensively in deliverable D3.1.

Synthetic data generation, as already discussed, can offer a viable solution to this and at the same time provide a privacy protective mechanism for data usage and sharing (El Emam & Hoptroff, 2019). The manufacturing domain, however, still poses a few challenges to be addressed. For instance, the subject of recent research studies is the semantic data enrichment of IoT streams at the edge layer (Xhafa, Kilic, & Krause, 2020; Gomes, da Rosa Righi, da Costa, & Griebler, 2021). On the same page is also the labelling of noisy, unsegmented time series data. Identifying causes or events on machine monitoring sensor streams can be a time-consuming and costly process. As such, the design of tools that can help researchers select and apply appropriate labelling techniques, even in real-time, is currently being investigated in the research community (Woodward, Kanjo, Oikonomou, & Chamberlain, 2020). Another possible solution would be to use pre-trained generic models that have learnt to discern events of the same nature (Malhotra, TV, Vig, Agarwal, & Shroff, 2017).

2.1.6 IPR Handing and Industrial Assets Provenance

2.1.6.1 Background

The provenance of data is of high relevance due to the massive increase of data generated these days. Provenance not only plays a proper role for the web and its data, but the topic has also arrived at the analogue world. In detail, the concept of Industry 4.0, which aims to establish production chains and manufacturing processes increasingly without human involvement, has relative references to data provenance. Many production and logistics processes are to be automated by machines and require reliable data along the way.

Data provenance records are nothing more than metadata that describe the actual industrial data.

The basic idea is formulated through five questions in order to achieve the mentioned level of trust:

- Why was the data produced?
- How was the data produced?
- Where was the data produced?
- When was the data produced?
- By whom was the data produced?

The W3C PROV standard was developed by the World Wide Web Consortiums (W3C). The recommendation provides a specification to answer the above questions. In addition, existing and standardised interchange formats, such as XML and RDF, are used to achieve interoperability and compatibility of metadata with provenance information.

The core of this recommendation consists of a data model (PROV-DM) that defines a common vocabulary for data provenance. The W3C intentionally defined the model very generically to adapt it to as many use cases as possible.

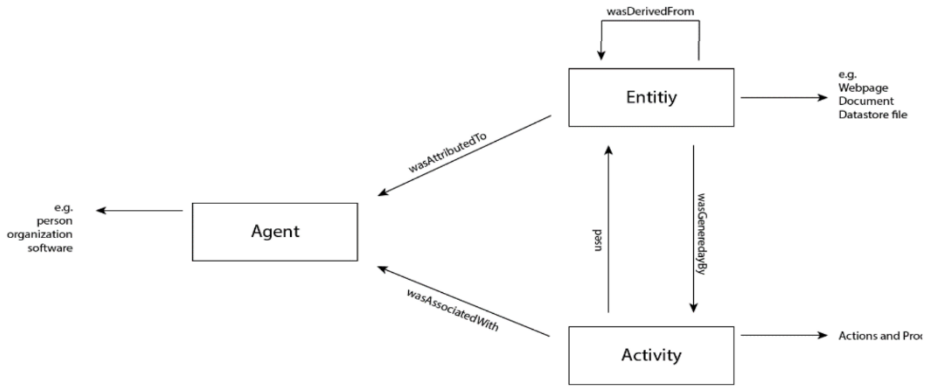


Figure 2-7 Data model of W3C PROV

Figure 2-7 shows a high-level overview of the W3C PROV data model. There are three central entities: Entities, Activities and Agents. Entities define the origin; they stand for documents, a diagram, or a piece of software. They can also refer to other entities with which they are related. When content or features are transferred from one entity to another, they are derivatives of each other.

Activities represent the dynamic processes and actions that help entities change over time. For example, updates to existing records may be published, or translations of texts may be written. So new entities are constantly being created by the activities.

Agents are the roles within activities. They can be natural persons, but also a computer programme, a whole organisation or any other object. They are always bound to activities; they have a corresponding responsibility. The same applies to entities and agents. A role in this context is a description of the person or function.

One of the most important factors is time cause data can be changed over time. Changes are addressed in great detail in the W3C PROV Recommendation and are recorded separately for each entity.

2.1.6.2 Identifiers for data

The use of a persistent data identifier is one possible solution to track the origin of data. *Persistent identifiers* are typically unique strings, which can be an unimportant string or usually a dereferenceable URI. One approach to ID generation is the Digital Object Identifier[17] (DOI). Initially intended for online articles of scientific journals, the publisher's inclusion indicates the origin of a document.

Another method more suitable for XMANAI is the use of URI. Each record is assigned an URI that is unique on the web. On the web, URI / URL must be unique so that resources can be found without mistakes. However, a system should be available that can check the URI of a record to ensure correctness.

Data versioning

A Provenance Engine controls the tracking of data. This is a holistic system that distributes functions across several components. Provenance data (or often called data lineage) is primarily metadata linked to the records to track changes or the progression over time.

The scheme used to create the metadata is described in more detail in the subchapter above. The recommendation of the World Wide Web Consortium is called W3C PROV and specifies “a record that describes entities and processes involved in producing and delivering or otherwise influencing that resource. Provenance provides a critical foundation for assessing authenticity, enabling trust, and allowing reproducibility. Provenance assertions are a form of contextual metadata and can themselves become important records with their own provenance.”

Multiple operations often change data within a data science pipeline and then no longer match the original state. For example, data scientists often perform these steps to prepare the data for the models used. In addition, data is aggregated, changed in format or removed because it is not useful.

If errors occur later in the evaluations, time and resources are needed to find the causes. It follows that in order to ensure the most accurate traceability of research results, it is important to implement a provenance engine that generates and stores the metadata for the records accordingly. Not only should the changes be tracked via the data sets, but also the programmes or scripts. It is also essential to track the programmes or scripts that process the data - which generate it or change the values. Figure 2-8 Figure 2-8: Separation of Store for a provenance engineshows an initial overview of the distribution of additionally saved data.

[17] https://www.iso.org/standard/43506.html

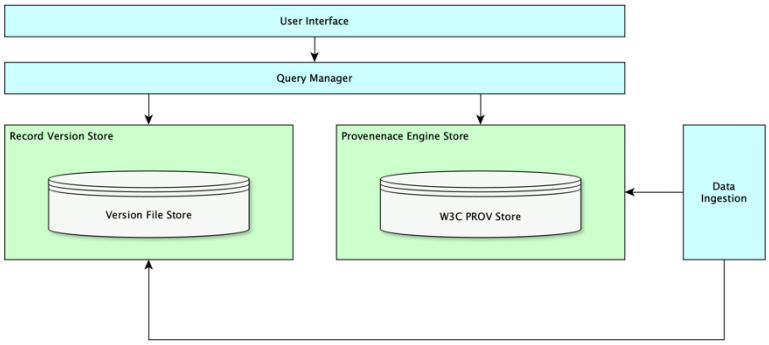


Figure 2-8: Separation of Store for a provenance engine

The changes to a record are saved by a version management system. The metadata for the changes according to the W3C PROV recommendation is held in a different store, as shown in Figure 2-8. This can be a graph database or a triplestore. The standard W3C PROV is available as a graph structure. A suitable store should be chosen. If changes are then made, such as adding new data, deleting tuples or manipulating values within the records, the changes should be versioned with a VCS store. At the same time, the PROV store should be expanded with metadata.

Users should be involved in this process with minimal effort. Change tracking will be user-friendly if it runs automatically in the background. Intervention by a user can nevertheless be helpful if some changes are not to be applied. For this purpose, the Provenance Engine should provide a suitable interface, ideally with an additional user interface component. However, operation via the console should also be possible in this case.

This way, it can be ensured that changes to text-based file formats can be executed efficiently and quickly without losing information. Changes to data already stored in database systems (DBS), for example, can also be exported and versioned using the mechanisms of a DBS.

Distributed Ledger and Blockchain

Blockchain and the distributed ledger are technologies based on a peer-to-peer network and generally store processes as transactions in a series of blocks. These blocks are not easily exchangeable because they build on each other. The terms blockchain and distributed ledger are often used as synonyms. The blockchain is a special case of the distributed ledger. Further differentiation is not intended at this point for the application in the XMANAI project.

Blockchain technology contains a list of blocks, whereby each block consists of a finite number of transactions. Each block receives a digital fingerprint based on a cryptographic hash function. The fingerprint of the preliminary block is always included in the computation of the digital fingerprint. In this way, a chain of blocks is created. If changes were made in the middle of the chain, this would affect the fingerprints of the block under examination and all subsequent blocks. This system ensures that only new blocks can and should be appended.

The use of blockchain makes sense when you want to share data and need a security aspect regarding the trust of data. As this technology is designed for decentralised infrastructures, it is recommended on this. Once established, a blockchain system is a fault-tolerant registry that is tamper-proof and transparent.

A blockchain can be used to secure provenance information of data. For this purpose, a transaction is created for a block, and this block contains one identifier, a hash of the related data and the position inside the blockchain. In addition, the signature of the data provider and source information about the data are also added. If data should be traced, a hash of the data is calculated and compared with that of the blockchain. If they do not match, it can be assumed that they have been manipulated.

With this approach, data authenticity and provenance can be verified, but recovery is not provided for in the first instance. XMANAI does not offer a decentralised infrastructure. This makes this approach not most suitable.

2.1.6.3 IPR Management

It is widely acknowledged that big data is reshaping many industrial sectors such as the manufacturing sector. One core aspect of big data is that usually data is more valuable when combined than it is in individual form. The combination of multiple data usually leads to the generation of new valuable information and knowledge when properly combined and aggregated. In addition to this, when it comes to machine learning (ML) or deep learning (DL) algorithms and their training, it is significantly increased when the amount of data utilised is also increased (Atkinson, 2019).

Nevertheless, while the landscape of big data is still evolving, the lack of suitable principles that will be effectively governing the access to big data from the intellectual property rights perspective still exists (Andanda, 2019). The growth of big data has revealed the need for rights related to both intellectual property and privacy of the data. Intellectual property rights (IPRs) are defined in order to safeguard the rights of the data owners over their content. Big data and IPRs are tightly interconnected because IPRs impact and interfere with the processes of acquisition, collection and storage followed in a big data ecosystem, as well as with the utilisation and the analysis of big data and the outcomes of these processes. In particular, during AI processing highly valuable outcomes might be produced in the form of new data or information, analyses, reports, correlations which are leveraged in decision making and knowledge extraction processes.

License is the legal instrument that specifies a standard set of terms and conditions regarding the reuse and sharing of the owner's data. Intellectual property rights are typically expressed in the form of licenses that drive the formulation of the decisions related to data access and data sharing of any typical big data ecosystem. To this end, it is instrumental that any big data ecosystem provides the means to owners of the data to define the proper licensing information of their data as well as to provide the mechanisms that utilise these licenses as a core part of its data access and data sharing mechanisms. During the data reuse or sharing it is imperative that all legal, ethical and professional obligations that have been set by the asset owner are met.

Licenses can be formulated and customised by the legal department of the organisation that owns the data according to the organisation's needs (usually referred as data use agreement – DUA). In case of open data, they are ussually based on a variety of standard licenses which are prepared by international organisations with aim of promoting and facilitating the effective and legally compliant data sharing and are commonly used licenses. Independently if the data license is based on a custom template or a proposed standard that are three common conditions that are explicitly defined on any open data license:

- *Attribution*: The specific condition defines that the data owner must be given the appropriate credit, a link to the license must be provided and in the case where changes were introduced they have to be indicated.
- *Copyleft*: The specific condition dictates that any outcome that derives from the utilisation of the specific data must be distributed under the same license as the original.
- *Non-Commercial*: The specific condition dictates that the data cannot be used for commercial purposes under any circumstances.

With respect to the standard open data licenses, the following table presents the most commonly used:

Table 2-4 Most common open data license types

License	Description	Variations
Public Domain	The specific license constitutes a relinguishing of all rights over the data to the public domain. It allows all data to be used freely, waiving all copyright interests and rights.	N/A
Creative Commons[18]	The specific group of licenses enables control over how data may used instead of declaring it public domain or reserving all rights with multiple variations. They provide the means to control whether the work can be used for commercial purposes or not, and if the work can be altered for other uses. In addition to this, they provide the Attribution condition.	• Attribution (CC BY) • Attribution-ShareAlike (CC BY-SA) • Attribution-NoDerivs (CC BY-ND); • Attribution-NonCommercial (CC BY-NC) • Attribution-Non-Commercial-ShareAlike (CC BY-NC-SA) • Attribution-Non-Commercial-NoDerivs (CC BY-NC-ND)
Open Data Commons[19]	The specific group of licenses as similar to the Creative Commons license however they focus on the licensed reuse of databases and similar datasets. It includes a variations that require Attribution, Copyleft and more.	• Open Data Commons Open Database License (ODC-ODbL) • Open Data Commons Attribution License (ODC-BY) • Open Data Commons Public Domain Dedication and License (PDDL)
Community Data License Agreement (CDLA)[20]	The specific group of licenses was introduced and sponsored by the Linux Foundation with the aim to promote free exchange of data. It allows users to use, modify and adapt the licensed dataset and the data within it, and to share it, as well as Copyleft and Attribution variation.	• Community Data License Agreement – CDLA Permissive 2.0 • Community Data License Agreement – CDLA Sharing-1.0

2.1.6.4 Key Challenges & Considerations

Data provenance will be a central component of the XMANAI platform to enable the traceability of data accesses and modifications for all users. However, the approach of implementing a blockchain does not meet the requirements and does not lead to the desired results. A blockchain is particularly suitable for a decentralised network of participants, which is not envisaged here. For this reason, the focus is on developing an appropriate and efficient solution that is less resource-consuming.

On the one hand, the application must store and make all datasets available; different versions should also be stored. In addition, many accesses and changes must be stored synchronously in a separate metadata catalogue and made available again for different use cases. While text-based data can be managed with a version management system quite easily, binary data is not versionable. This also counts, for example, for a relational database system.

Data IPR can be rather challenging in terms of IPR handing and IPR conflict resolution. As the analysis of big data usually involved the combination and aggregation of multiple datasets which are licensed with different licenses containing different terms and conditions, the challenge of IPR conflict resolution on the produced outcomes and results raises. The cross-check and evaluation of the

18 https://creativecommons.org/licenses/?lang=en
19 https://opendatacommons.org/licenses/
20 https://cdla.dev/

compatibility of the different licenses of the involved datasets needs a sophisticated process to be established that respect the terms and conditions of each license and will effectively resolve all conflicts.

2.2 Technologies

2.2.1 Industrial Data Ingestion

The section presents an overview of the technologies for industrial data ingestion, which will be considered for the use in XMANAI. The initial consideration about their suitability is given in the following subsection.

Table 2-5: Overview of relevant industrial Data Ingestion Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Piveau Consus	Consus is responsible for the data acquisition from various sources and data providers. This includes scheduling, transformation and harmonization. Service to import, process and store data or metadata in any datastore from any source www.piveau.de	Apache 2.0	• (Fast) Importer for metadata and data • Adaptable modules • Scripting component for different metadata transformation • Modern Pipe-concept • Monitoring about CRUD operations	• Transformation errors not logged • Some paragraphs in documentation in progress; for installation and administration complete
IDS Trusted Connector	The IDS Trusted Connector is an open IoT edge gateway platform and an implementation of the Trusted Connector in the Industrial Data Space Reference Architecture. It connects sensors to cloud services and other connectors, using a wide range of protocol adapters. The Trusted Connector is open source and based on open standards to avoid vendor lock-in. https://internationaldataspaces.org/; https://github.com/industrial-data-space/trusted-connector	Apache 2.0	• Adaptable modules	Only first research prototypes are avaialble
Kepware	Kepware's software solutions for the Industrial Automation Industry bridge the communication gap between diverse hardware and software applications, https://www.kepware.com/en-us/	Proprietary Business License	• High compatibility with control system drivers of the main industrial manufacturers. • Make available the work with several communication protocols and industrial buses • Data Integration parameters tested in many of industrial environments	• It is paid software depending of the kind os drivers implemented • Must renew some licences every several years • Proprietary software without the possibility of adapting functionalities
Apache Kafka	Broker system for distributed publish-subscribe messaging operations that receives data from	Apache 2.0	• Low Latency (up to 10ms), which is relevant in industrial environments.	• Does not contain a complete set of monitoring and managing tools

Technology Name	Short Description, URL	License	Pros	Cons
	multiple origins and makes the data available for subscriptors https://kafka.apache.org/		This leads to high throughput • Fault tolerance (resistance to machine failure within the cluster) • Its distributed architecture makes it easily scalable	• Lacks some message paradigms such as point-to-point queues, request/reply
Spark Streaming	Extension of the core Spark API that provides scalable, high-throughtput and robust stream processing of live data streams https://spark.apache.org/docs/latest/streaming-programming-guide.html	Apache 2.0	• Faster than other frameworks as it works in cluster mode and uses distributed processing • Robust and fault tolerant, with fast fault recovery. • Unified engine which backs both batch and stream processing workload	• It is developer-focused, not business user-focused • Does not provide event level granularity (works in batches) • Limited windowing support
Apache Camel	Framework that offers a message-oriented middleware that provides rule-based routing and mediation engine. https://camel.apache.org	Apache 2.0	• Open source • Easy to create clear directives to route and filter the messages • Supported by large community	• Medium management of heavy datasets • Very dependent on Spring • Lack of in-depth documentation

2.2.1.1 Architectural Considerations

Piveau Consus is suitable for an architecture consisting of various available data sources and a central data lake. It is using a pull approach to gather data from the sources. Piveau Consus is the driving factor in such architecture. Data source endpoints need to be available for Consus to harvest and newly created endpoints need to be registered with Consus.

The *IDS Trusted Connector* is suitable for an architecture in which the data is stored decentrally. It acts as a gateway to the data source. In this scenario each data consumer / publisher operates an instance of the connector and performs data exchange on demand. There exist no central data lake. Trust considerations are implemented in the connector. The data is not leaving the connector if the requesting connector is not considered trustworthy. Only early research prototypes are available at the moment.

Kepware simplifies the gap between hardware and software applications. Instead of communicating with industrial devices directly (adding inefficiencies and operative and security problems), Kepware acts an intermediate between industrial components, such as PLCs, RTUs or databases, and software applications. Thus, these systems recover different information through Kepware by means of the well-known standard protocols OPC DA and OPC UA.

Apache Kafka is a platform that provides 4 APIs with different features (Producer, Consumer, Streams and Connector). It works as a simple distributed commit log system. All messages are ordered and this sequence is immutable, avoiding packet losses. Apache Kafka's architecture allows horizontal scaling. This means that it works properly with both a small and a large number of connected devices.

Traditional stream processing systems work with one stream of record data at a time in a continuous flow. This strategy can lead to packet loss or slow performance, among other problems. In contrast,

Spark Streaming works with discretized streams that can process data in parallel, making the system fault-tolerant and optimized in terms of latency.

Apache Camel consists on 2 main components: Camel Components and Camel Endpoints. Camel Components provide a uniform Endpoint Interface and act as connectors to all other systems while Camel Endpoints can send messages to Camel Components or receive messages to them. It acts as a routing system where a message is sent from one sending application and received by another, allowing communication between different platforms.

2.2.2 Data Transformation / Curation

The section presents an overview of the technologies for data transformation and curation, which will be considered for use in XMANAI. As Data Transformation and Curation we describe the subpart of the Data Wrangling process that is responsible for structuring, cleaning, enriching and validating the raw data. There is a wide range of tools (free and commercial) that could be used for XMANAI needs. The initial consideration of their suitability is given in the following subsection.

Table 2-6: Overview of relevant industrial Data Transformation & Curation Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Pandas	pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language. https://pandas.pydata.org/	BSD 3-Clause	• Largest collection of dataFrame manipulation tools • Perfect documentation • Huge community • Easy debugging • Many ready-to-use tools for data cleaning, harmonization, canonicalization etc	• Does not scale to big data • Slow execution on big data. Needs to run sequentially in a batched fashion • Manually write code for batch processing of the dataset
Dask	Dask provides advanced parallelism for analytics, enabling performance at scale for the tools you use https://dask.org/	BSD 3-Clause	• Supports almost all functionalities of Numpy, Pandas, Scikit-Learn • Supports build-in task scheduling with Task Graphs • Can be integrated to distributed computing environments e.g. Kubernetes • Lazy loading and execution • Supports end-to-end pipelines • Massive community • Open Source	• No direct connection to databases (though there are some plugins e.g. for SQL) • Needs some expertise to avoid bugs
Apache Spark	Apache Spark is a unified engine for large scale data processing. It enables processing data in batches or in real-time streaming and executing fast, distributed ANSI SQL queries. https://spark.apache.org/	Apache 2.0	• Works in distributed environments (YARN, MESOS, Kubernetes etc.) • Support distributed data datasets of tabular data • Integrated SQL • In-memory data processing • Immutable data	• In-memory processing can be expensive • Spark job requires manual optimization • Back pressure handling • No file management system

Technology Name	Short Description, URL	License	Pros	Cons
			• Lazy loading and execution • Supports processing on streaming data • Most commands are the same with Pandas • Many built-in functions for column manipulation • Supports Python UDFs (User Defined Functions) • Has its own ML lib • Supports integrated end-to-end pipelines • Task visualization • Part of Apache family • Massive community • Open Source	• Supports only near real-time processing (micro-batches) of live data • No built-in scheduler for automating jobs
Optimus	Optimus is a library to easily load, process, plot and create ML models that run over pandas, Dask, cuDF, dask-cuDF, Vaex or Spark. https://hi-optimus.com/	Apache 2.0	• Supports many backends – Pandas, Dask, CuDF, Dask_Cudf • Visualization of data with Bumblebee • Fast execution & easy to implement • Clean API with many handy functions for data curation • Open Source	• Supports only tabular data • Does not support lazy-loading and lazy-execution so far • Cannot load data from a database – needs intermediate step • Does not support scheduling
AWS – Data Wrangler	AWS – Data Wrangler can be described in a phrase as the Pandas library on AWS. It is a tool for loading and processing pandas dataframes when developing an application in the AWS ecosystem. Tt integrates with most databases like Athena, PostgreSQL, MySQL etc. https://aws-data-wrangler.readthedocs.io/en/stable/index.html	Apache 2.0	• Very simple API • Integrates perfectly with Pandas	• Restricted to AWS ecosystem. Useful only in case we host all our services in AWS

Technology Name	Short Description, URL	License	Pros	Cons
Trifacta	Trifacta is an interactive cloud platform (service) for data engineers and analysts to prepare data for analytics and machine learning https://www.trifacta.com/	Proprietary	• Easy-to-use User Interface • Supports many input sources • Many smart tools for Data Quality • Suggest transformations automatically • High level of interactivity • Cloud native, supports all well-known cloud platforms (e.g. AWS, google cloud etc)	• Assumes tabular data • Standalone (cannot be embedded) • Doesn't define explicitly how many records it can handle

2.2.2.1 Architectural Considerations

According to the classification published by (Alfredo Nazabal, 2020), data transformation and curation are part of the data organization and quality assurance steps. For a tool to be a legit candidate, it should fulfil, among others, the following requirements; it should (a) have an extensive collection of handy data transformation functionalities; (b) have a friendly interface for monitoring/debugging/scheduling the data transformation process; and (c) be able to handle large amounts of data. These are the main criteria that we should evaluate to decide the perfect tool for the XMANAI purposes.

Data Transformation tools fall into two categories based on the type of computational environments they support. Some tools can be utilized in all types of computational environments (cloud, local, on-premise), e.g. the Apache Spark and some others are connected to a specific service, e.g. AWS-Data Wrangler.

The prominent representatives of the first category are Pandas, Dask and Apache Spark. Pandas is the fundamental tool for handling DataFrames. It has a vast community and supports an extensive collection of ready-to-use functions. The consideration regarding Pandas is that it cannot operate efficiently on large datasets. Hence, moving to Dask or Spark is the only option when the data source reaches some gigabytes. Dask, in one phrase, can be characterized as distributed Pandas, since it exposes most of the Pandas' functionalities with a similar API. Behind the scenes, it splits the input source into smaller parts (batches), applying the transformation pipeline in a parallel or distributed manner. It can also be used from a single machine up to large clusters of computers. The main disadvantages of Dask, are some robustness problems that have been reported and the difficulty to debug it. Apace Spark is the de-facto choice for big data processing. It is part of the Apache ecosystem with many data handling tools that work seamlessly together. Apache Spark is a framework combining distributed computing, SQL queries, machine learning, and more that runs on the JVM and is commonly co-deployed with other Big Data frameworks like Hadoop. Finally, we should also consider Optimus, a library built on top of Spark/Dask/Pandas (depending on the configuration); hence it can work seamlessly with all the tools above. Optimus offers some valuable tools for data cleansing and transformation.

The second category contains data transformation tools dedicated to a specific cloud service, e.g. AWS, Google Cloud etc. For example, AWS – Data Wrangler is a package offered by AWS-Amazon for data frame processing, and it works only on data stored in AWS storing services such as the S3 bucket. Hence, these tools restrict us to a specific provider and should be considered a choice only if we want to tie all our services to the particular provider.

In conclusion, we propose Spark as our default solution for general purpose data transformation. In cases where we want to apply some light transformations, Dask can be considered an alternative. Pandas is an option only in cases of small dataset cases.

2.2.3 Data Storage

The section presents an overview of the technologies for data storage, which will be considered for use in XMANAI. The initial consideration of their suitability is given in the following subsection.

Table 2-7: Overview of relevant industrial Data Storage Technologies

Technology Name	Short Description, URL	License	Pros	Cons
HDFS	Hadoop Distributed File System (component of Apache Hadoop) is a collection of Open Source software utilities, able to manage massive amounts of data and computations. As the name suggests storage is distributed across several nodes (and MapReduce computation is supported) https://hadoop.apache.org https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html	Apache 2.0	• High availability • Replication is enabled by default which helps us prevent data loss • Easy to scale by adding more nodes • High performance with big data volume, mainly big single files • Reliable and failure resistant thanks to replication factor	• Can be hard to maintain as we keep adding more Hadoop ecosystem tools • Prior dominant open-source distribution of Hadoop provided by Hortonworks (HDP) was acquired by Cloudera and it is only available as commercial solution • Small files and impact performance • No security by default. Should enable authentication and authorization with Kerberos and Ranger respectively.
MinIO	Kubernetes-native object storage is a High Performance Object Storage able to handle unstructured data, such as photos, videos, etc https://min.io/	GNU AGPL 3.0	• Lightweight and high performant • Easy to deploy in both standalone and distributed mode • Supports both BareMetal and Docker/Kubernetes deployments • Clients are available in many languages, as well as a console client • S3 compatible • Able to handle unstructured data	• Many of the functionalities that are available in console client, are not available in other clients (e.g., share a file)
MongoDB	MongoDB is a non-relational database, conceived as a document database, to store mainly JSON-like files https://www.mongodb.com/	GNU AGPL 3.0	• High availability • Data structure to be easily changed, if varies over time • Easy to scale (horizontal scalability) • Distributed computation • Wide number of programming languages supported	• A proper indexing of data is required to have optimal performance • Transactions not allowed • Only CSV and JSON formats are allowed

Technology Name	Short Description, URL	License	Pros	Cons
			• High-performance query execution • No SQL required	
Neo4j	Graph database management system, ACID compliant transactional database with native graph storage and processing https://neo4j.com/	AGPL v3	• Data are stored in form of graph • Efficient in terms of performance and hardware usage (if compared with others) • Easy to scale (horizontal scalability)	• Low number of programming languages supported (if compared with MongoDB)
Virtuoso	A multi-purpose and multi-protocol (Hybrid) Data Server that supports management of data represented as relational tables and/or property graphs https://virtuoso.openlinksw.com/	GNU GPL	• It allows to manage data both in form of graph and table •Efficient in terms of performance • Easy to scale (horizontal scalability) • SQL supported • MapReduce supported • Wide applicability for Enterprise Knowledge Graphs (and Explainable AI)	• Low number of programming language supported (if compared with MongoDB)
PostgreSQL	A free and open-source object-relational database management system, SQL compliant https://www.postgresql.org/	PostgreSQL Licence (v14.1)	• Written in C • SQL standard compliant • complete ACID compliant • Able to support complex queries • Very flexible in term of scalability • Including built-in binary replication • Able to support Matlab/R analysis • Among the others, it is able to support JSON, XML documents • Wide number of programming languages supported • Flexible full text search • It is the default database for most cloud platforms	• Documentation only available in English • Not high performance in terms of speed • Content migration hard to be performed • Not trivial the version update

Technology Name	Short Description, URL	License	Pros	Cons
Amazon S3	Amazon Simple Storage Service is a storage service offered on the scalable infrastructure of Amazon https://aws.amazon.com/it/s3/	Proprietary	• Can be used to store any kind of object. • Easy to scale • Data are equipped with ID and metadata • Data storage granted at 99.99%	• AWS S3 offers simply the cloud storage service.
AWS Neptune	A managed Graph Database published by Amazon (part of Amazon Web Services), supporting popular graph models https://aws.amazon.com/neptune/?nc1=h_ls	Proprietary	• Very efficient in terms of performance • Easy to scale • Based on common standards	• Based on AWS S3 cloud storage service • No open source licence
MySQL	Relational database management system (RDBMS), developed by Oracle Corporation https://www.mysql.com	GNU GPL (and commercial licence)	• High availability and large spread in industry domain • SQL supported • Supported by the most common programming language (Java, ODBC, php, Python)	• Only structured data (stored in table format) is allowed • Predefined schema required • It is difficult to change the data structure changing over time • Vertical scalability
Cassandra	NoSQL distributed database, based on column storage (component of Apache Hadoop) https://cassandra.apache.org/_/index.html	Apache 2.0	• High availability • High performance with big data volume, mainly several small files • SQL supported but not required • Easy to scale (horizontal scalability) • Distributed computation • Reliable and failure resistant thanks to replication factor • The architecture is based on peer-to-peer node, overcoming the issue of HDFS Master Node	• Data are organized in table

2.2.3.1 Architectural Considerations

Due to the large availability of different systems for data storage, it is not easy to draw up an exhaustive list without applying any preliminary filter on the choice of solutions to be presented. To compile the table above the consortium tried to include the most common representatives of different data storage solutions types.

Opensource solutions are investigated mainly to follow the European Commission Open Source Sofware strategy[21] in terms of Digital Transformation, that encourage the innovative and collaborative power of open source and its principles with the objective of improving the sharing and reusing of low-cost software solutions. In the table above, *Amazon Neptune* (database for graph storage) is mentioned only for completeness, since it is the graph database leveraging on one of the most popular and well-known cloud services, AWS S3 (listed in the table as well, as example of simple storage service). However, far from being an opensource solution, it won't represent the first option in the choice of the most suitable data storage system for XAMANAI.

Then, the (expected) nature of data to be managed and use cases plays an important role toward the choice of the right database and data storage system in general. Firstly, it is important to take into account if data can be easily store in table shape (maybe according to a relational database) or it is more heterogenous and requires a different type of infrastructure behind. As of now, the known four pilots' datasets have tabular structure. However, other datasources of other data types can be introduced later.

Relational (SQL) databases are suitable to manage "static" data with low complexity in data structure. If on one side an SQL DB requires a fixed data schema that must be defined in advance, on the other side it facilitates some common operations, like filtering and aggregation. Choosing it, you must have clear in mind the structure of data to be stored and the queries to be addressed. As a representative of family of such databases, we included in the list above *MySQL* which is one of the most popular relational databases (with large applicability also in industry domain). It provides great advantages in using a very common and well-known system, where data structure is predefined (speeding up a number of operations and queries) and can be a good choice at least for the first versions of the XMANAI prototype.

PostgreSQL is another example of relational SQL database, allowing the data binary replication. For XMANAI purposes, considering that most of the data can be provided in the shape of harvested as or transformed into structured data, a relational database is an appropriate component for an asset store. Nevertheless, it is thinkable to store binary files in a file system.

Among the several noSQL options, that allow to manage data in a more flexible way, it is important to take into account the type of data to be stored and the final use. For instance, *Cassandra* is a wide-column storage system where datais still represented by models in table shape but it can be easily altered; *MongoDB* is a document storage system where documents (with different structure) are stored equipped with a key and it is possible to modify their content.

When relationship among entities of the same type is a key value of the information stored, it's worth to consider databases that allow graph data storage, that is, data shaped according to an entity (the node) and a number of relationship (the edges).

Once the right class of databases and storage systems has been identified, other important parameters must be taken into account to choose the best solution. Actually, filling the table, we consider other features such as: scalability (dealing with large amount of data which size may increase), security (to protect mainly sensitive and personal data but also to avoid data loss) and performance (to query information spending the lowest amount of time). Documentation availability and deployment easiness are other key features to take care about.

For instance, focusing on graph databases, Neo4j and Virtuoso are two possible options. Of course, the two solutions are not equivalent and it is important to compare their features carefully to identify the most suitable one. Even if both are graph database management systems, the former is mainly dedicated to graphs, while the latter accepts also data in format of document and tables, hence it's fundamental to have clear in mind what is the purpose of the choice. Neo4j is quite new if compared

[21] https://ec.europa.eu/info/departments/informatics/open-source-software-strategy_en

with Virtuoso and it is implemented in Java/Scala language, while the second one is implemented in C programming language.

2.2.4 Metadata Management

The section presents an overview of the technologies for metadata management, which will be considered for use in XMANAI. The initial consideration of their suitability is given in the following subsection.

Table 2-8: Overview of relevant industrial Metadata Management Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Piveau Hub	Service to store metadata based on Linked Data. www.piveau.de	Apache 2.0	• Linked Data Support • Native DCAT-AP Support • Native StatDCAT-AP & GeoDCAT-AP Support • Extensible vocabulary • Scalable • Container ready for CI / CD • Indexing for Searching / Filtering • Triple store configurable • Supports Keycloak • Adaptable UI component • Microservice approach with HTTP REST API	• Use of SPARQL Language • Documentation in progress (not complete yet) • Needs effort to use another Vocabulary
IDS Metadata Broker	IDS Metadata Registry to store information about IDS participants and their data offers. http://internationaldataspaces.org	Apache 2.0	• Metadata Repository of the IDS architecture • Works with other IDS components • Light-weight specification. Message component can be integrated into other solutions, such as piveau-hub • IDS: Industry concept/standard to be adapted and most likely integrated into GaiaX	• Works best in combination with other IDS components. Data sharing would call for extensive use of IDS standard
Apache Atlas	Data Governance and Metadata Framework for Hadoop https://atlas.apache.org/#/	Apache 2.0	• Displays lineage of data as it moves through various processes • Extensive list of metadata types and ability to create new ones • Functionalities available through UI and REST API	• Only available in the Hadoop Ecosystem
CKAN	Data management solution for building on-premise data repositories and mostly a standard for publishing in public sector (open government data). The Open Knowledge Foundation is	AGPL	• Open Source • Huge development community • Detailed documentation • Fokus on metadata, also on data • Data storage feature for binary and tabular data	• Maybe not usable licence for XMANAI • Realtime data only supported through extensions

Technology Name	Short Description, URL	License	Pros	Cons
	supporting the maintaining process. https://ckan.org/government			

2.2.4.1 Architectural Considerations

A metadata manager has the task of providing the metadata for the different data assets within the XMANAI project and also making it searchable. This also includes the filtering of data sets. In addition, there is the provision of further metadata so that information about the status and content of the data is available.

Piveau Hub already addresses these requirements and goes one step further. Firstly, the established metadata standard DCAT-AP is used, which precisely specifies the description of the data. Associated with this is also a user interface that is aligned with this standard. Secondly, the use of graph-based data structures is already integrated, including suitable interfaces (APIs).

The alternative solution would be CKAN, which already offers a similar scope for metadata management, but can convince with a variety of plug-ins for a customised solution. In this case, however, adjustments would have to be made to the data management and user interface.

Both solutions mentioned so far have a standardised solution in relation to Big Data applications. Although both solutions can also manage and provide numerous data sets (volume), there are only few interfaces to other Big Data solutions such as Hadoop, Hive or Falcon.

2.2.5 Data Anonymization

The section presents an overview of the technologies for data anonymisation, which will be considered for use in XMANAI. The initial consideration of their suitability is given in the following subsection.

Table 2-9: Overview of relevant industrial Data Anonymization Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Piveau Incognito	Service to anonymize selected parts of data or metadata. It includes a rich set of features such as the recognition of anonymisation needs, the execution of anonymisation on a local environment and the use of state-of-the-art anonymisation algorithms www.piveau.de	Apache 2.0	• Checks anonymization needs • Use of state-of-the-art algorithms for anonymization • Configurable to anonymize desired parts of the data • Works well for structured data • Non-structured data can be transformed into table data • Is built on ARX	• Works not for image data or binary data files • Excluding Differential Privacy for big volume data • Documentation only for important things
ARX	A state-of-the-art open source software for data anonymization. It supports an extensive variety of privacy and risk assessment	Apache 2.0	• Probably the most extensive open-source anonymization tool with available anonymization techniques	• Not applicable in distributed environment • Significant resources needed

Technology Name	Short Description, URL	License	Pros	Cons
	models while also offering a large variety transformation methods. https://arx.deidentifier.org/		• Available both as a desktop tool and Java library	for complex operations
UTD Anonymisation Toolbox	UTD Anonymisation Toolbox is offering the implementation of a large number of anonymisation algorithms and methods over different privacy definitions. http://www.cs.utdallas.edu/dspl/cgi-bin/toolbox/index.php	GNU General Public License	• Easy integration Available as a standalone tool or library	• Limited support for anonymisation methods
Amnesia	A tool for performing research and sharing your results that satisfy GDPR guidelines by using data anonymization algorithms. https://amnesia.openaire.eu/	GNU General Public License	• Easy integration • Nice and easy User Interface	• Not applicable in distributed environment

2.2.5.1 Architectural Considerations

As explained in section 2.1.2.4, data anonymisation constitutes a major challenge nowadays due to the strict regulations which are imposed by the European Commission and national legislations. In addition to this, the nature of this task imposes several challenges which may vary depending on the nature of the data that should be anonymised. The tools described in the previous sections constitute candidate solutions that can address these challenges, nevertheless despite the advancements that have been made through the intensive research each tool has its benefits and drawbacks.

In particular, ARX is suitable for an architecture where the tool can run in a centralised deployment as there is no possibility to scale in a distributed environment. Ideally, it should be hosted on a server with significant resources which are needed in complex anonymisation jobs. Hence, the architectural decision to utilise ARX as the technology to address the anonymisation challenge should take this into consideration. On the other hand, Piveau Incognito integrates ARX as a library. The advantage is the use of simplified configuration options for the anonymisation of structured data and as continuous text. The component can be used as a stand-alone service and provides an interface for communication. However, it lacks of support for big volume data and of proper documentation that would facilitate the easy and smooth integration. Amnesia is a user friendly tool well suited for centralised environments only. Finally, the UTD Anonymisation Toolbox is considered as suitable candidate for the anonymisation process as it can be easily integrated with the rest of the services of the platform. However, its main drawback is the limited support for anonymisation methods which can be rather restrictive.

2.2.6 Identity Management

The section presents an overview of the technologies related to identity management, which will be considered for use in XMANAI. The initial consideration about their suitability is given in the following subsection.

Table 2-10: Overview of relevant industrial Identity Management Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Keycloak	Service for identity and access management that supports different access methods. https://www.keycloak.org	Apache 2.0	• Very easy to deploy either bare-metal or with docker • Extensive list of authorization techniques • Can be easily integrated in available services	• Many of the available features are not available in the REST API, or are at an experimental level • REST API doc is not very good and it is not 100% synced with the actual implementation
Okta	Cloud-based identity management tool. It offers other functionalities as well like SSO. https://www.okta.com/	Proprietary business license	• SSO Support • SDKs available in different languages • Login integration support for various services such as Google, Facebook, etc.	• Available only through Okta cloud, can't be installed locally • Developer Plan (free edition) offers less than other plans
Janssen (gluu)	Janssen is offering a scalable centralized authentication and authorization service. The components of the project include implementations of the OAuth, OpenID Connect and FIDO standards.	Apache 2.0	• Support for OAuth, OpenID Connect and FIDO standards • High availability, • High concurrency, • Highly flexible	• Community edition offers less than commercial version
OpenAM	OpenAM is an open-source access management, entitlements and federation server platform. It features password management, provisioning, audit and dompliance, self-service and delegated admin. It provides RBAC, XACML, Federation and SSO, Web Access Control and SOA Security. https://www.openidentityplatform.org	Proprietary business license and community edition version	• SSO Support • API integration, • Two-factor authentication • Multiple-factor authentication • RBAC support	• Community requires registration • Community edition offers less than commercial version
Hashicorp Vault	Provides functionalities for Secrets Management, Data Encryption, Identity Management. https://www.vaultproject.io/	Mozilla Public License 2.0	• Credentials generation • Secrets management • Ideal for low trust public clouds	• Setup during startup is not that intuitive, significant learning curve

2.2.6.1 Architectural Considerations

The need for an effective, efficient and flexible identity management mechanism has been addressed with different frameworks that address this need following a variety of approaches and offering different toolsuites as described in the previous section. As these frameworks vary in terms of processes and services offered, the decision to adopt one of them merely depends entirely on the needs of each project.

In particular, Keycloak is suitable for managing the authorization aspects of an architecture. It is well-equipped with multiple functionalities, enabling the establishment of robust authorization and easy integration with the rest of the components of the architecture. In addition to this, it offers the holistic user account management lifecycle required by any project. On the other hand, Okta can only be integrated as a 3rd party tool, hence it poses significant restrictions on the architecture which should be designed in a way to establish communication with the Okta cloud. This imposes also a strong dependency with this 3rd party tool that can very restrictive. Janseen provides a scalable centralized authentication and authorization service that supports several identity management models and technologies however the open source version is lacking of valuable features. OpenAM provides support for enhanced identity management and easy integration via the provided APIs, however the open-source community edition has limited offerings in comparison with the proprietary licensed version which can be also rather restrictive. Vault is a well-established solution that enables secure storage and management of secrets and credentials as tokens, passwords, certificates, encryption keys for protecting secrets and other sensitive data using a UI, CLI, or HTTP API. It can be easily integrated and provides support for multiple data storage solutions as well as with trusted identity providers. With regards to the identity management aspects, Vault is mostly focused on the successful integration of third-party identity providers while it lacks in the implementation of its own identity management mechanism, as the tool is mostly focused on the management of secrets and credentials.

2.2.7 Access logs

The section presents an overview of the technologies related to accessing and managing access logs, which will be considered for the use in XMANAI. The initial consideration about their suitability is given in the following subsection.

Table 2-11: Overview of relevant industrial Access Logs Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Sentry	Logs and application monitoring. https://sentry.io/welcome/	Proprietary Business License	• Available both for on-premise installation, or as a cloud service. Developer (free plan) available • Ability to investigate which commit "broke" the implementation and is responsible for an error • Available SDKs for many languages	• Proprietary Business License
Logstash	ETL tool for managing log files produced from various services. https://www.elastic.co/logstash	Elastic License 2.0	• Many transformations available, so that you can modify your log entries as you desire • SDKs available in many languages • Mature and highly adopted • Extensive documentation	• Requires enough memory (at least 3GB of RAM) to properly work • Difficult to start for inexperienced users
Graylog	Central logging system with the flexibility to adapt warnings and dashboards for different use cases. https://www.graylog.org/products/open-source	SSPL	• Open Source • Adaptable for different log formats • Dashboard support • Open plug-in architecture	• Licence maybe not fit to XMANAI requirements (like GPL)

Technology Name	Short Description, URL	License	Pros	Cons
Octopussy	Easy and not overloaded logging system with an octopus. https://github.com/Octopussy-Project/Octopussy	GPL	• Very fast • Easy to use with some configurations • Open source on GitHub	• Only some statistics available • No user interface • GPL licence

2.2.7.1 Architectural Considerations

When using a logging system, it is more a matter of integration into the existing system. If Elastic Stack technology is already used, Logstash would probably be more suitable than one of the other logging systems. The integration is already very good with regard to ElasticSearch.

Octopussy impresses with its simplicity and comes without many features for analysis and visualisation. This can be an advantage, but also a disadvantage. In such a case, it should be easier to integrate and adapt to the XMANAI platform, and features that are not needed are installed as well. Otherwise, however, it can happen that some features are missed and the only way to meet the requirements is to switch to another logging system.

With Graylog and Octopussy, the licences used also have an impact. These are GPL and a variant (SSPL) of it. These require that the developed software is also placed under an open source licence. The extent to which it can only be used for platform integration must be checked separately. However, technical extensions are not recommended.

2.2.8 Policy Enforcement

The section presents an overview of the technologies for policy enforcement, which will be considered for use in XMANAI. The initial consideration of their suitability is given in the following subsection.

Table 2-12: Overview of relevant industrial Policy Enforcement Technologies

Technology Name	Short Description, URL	License	Pros	Cons
IDS Connector	IDS Component to enforce usage restrictions and access control. http://internationaldataspaces.org	N/A	• IDS: Industry concept/standard to be adapted and most likely integrated into GaiaX • Component for data providers and consumers to connect their data sources/sinks to the Web.	• Policy enforcement needs to be integrated into the connector • Works best in combination with other IDS components. Data sharing would call for extensive use of IDS standard
myData (Ind2uce)	Technology providing means to control and restrict usage of data. It enables fine-grained masking and filtering of data flows at interface level. https://www.mydata-control.de/de/ Based on the Ind2uce framework. A technology	Proprietary	• Partial filtering and masking of data, context, situation restrictions and restrictions on the purpose of use • Central services for managing and controlling data flows at runtime • Easy integration into existing systems	• Proprietary solution • seems that the project is not actively maintained, last version release was in 2019

Technology Name	Short Description, URL	License	Pros	Cons
	providing means to control and restrict usage of data. https://www.iese.fraunhofer.de/en/competencies/security/ind2uce-framework.html		• Flexible set of rules for mapping data sovereignty requirements	
Casbin	An authorization library that supports access control models like ACL, RBAC, ABAC https://casbin.org/	Apache 2.0	• SDKs available in many languages • Very easy to create policies • Policies can be stored in various databases, such as PostgreSQL, MongoDB, etc • Supports various Access Control Models, such as RBAC, ABAC, ACL	Does not deal with authentication at all (no user tables)
Apache Ranger	Apache Ranger is a framework to enable, monitor and manage comprehensive data security across the Hadoop platform. https://ranger.apache.org/	Apache 2.0	• Fine-grained authorization • RBAC and ABAC supported • Can be used to secure several resources, such as HDFS, HIVE, Spark, etc • Support for auditing	• Available only for the Hadoop ecosystem • Difficult to set up from scratch

2.2.8.1 Architectural Considerations

Policy enforcement constitutes a core component of every platform architecture as it ensures the security, trust and integrity of valuable resources by regulating the access to the resources only to legitimate requestors. To achieve this, it usually interacts with most of the other components in the architecture to guarantee that the designed logical access is honoured. To this end, the most crucial aspects are a list of supported access control mechanisms as well as their flexibility and the ease of their integration into the designed solution.

The IDS Connector is based on the IDS standard in order to enable efficient access control and usage restriction features. While the IDS standard is considered a trustful industry standard, it requires additional effort for the integration of policy enforcement and the adoption of the proposed standard will propagate requirements in the data sharing framework. MyData is based on the Ind2uce framework and facilitates the access control and restricted usage of data. Partial filtering and masking of data flows is possible based on context, situation restrictions and restrictions on the purpose of use. Unfortunately, the software is proprietary and no evaluation based on example can be found. From the products internet presence it should easily integrable into existing systems and offers a flexible set of rules for mapping data sovereignty requirements.Another solution that needs to be mentioned is Casbin. It is a well-established framework offering high flexibility and an extensive toolset of features and integrations and it is suitable for policy management and enforcement, based on various access control models. One of its core advantages, is that it is easily integrated with a variety of applications since it is offering SDKs for many programming languages. Finally, Apache Ranger is a very powerful authorisation tool that is directly connected with HDFS deployments. Nevertheless, it is only suitable for HDFS-enabled deployments while requiring significant effort to setup.

2.2.9 Data marketplace

A data marketplace can be defined as an online transactional platform that facilitates the buying and selling of data and, more often than not, data products or data assets. Since many businesses seek to augment or enrich their internal information with external data in order to improve their analytics and create better insights, these data marketplaces are appearing at an increasing rate to match data consumers with data sellers in a secure and reliable environment.

An overview of relevant industrial Data Marketplace technologies is presented in the following table.

Table 2-13: Overview of relevant industrial Data Marketplace Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Dawex Data Exchange Platform and Global Marketplace	Dawex Data Exchange Platform is a fully customized white-label SaaS platform to distribute, source, commercialize and/or orchestrate data ecosystems. https://www.dawex.com/en/product/	Proprietary Business License	• Provides a forum for matching offer and demand. • Takes advantage of blockchain to allow exchanges without intermediary. • High trustability, traceability and transparency	• Fully proprietary. • Targets mainly on advisory services i.e. how to get value of your data. • Difficult to test the provided features - does not provide access to the application.
Datum	Datum network allows anyone to store structured data securely in a decentralized way on a smart contract blockchain, optionally enabling selling and buying data using the DAT smart token. Based on Ethereum. https://datum.org/	LGPL-3.0	• Highly secure because uploaded data are encrypted. Claims to be fully GDPR compliant. • Provides simple and easy-to-use API for smooth integration. • Data Storage is charged only for what you use.	• Allows for data sharing (selling/buying) only. • More suitable for personal use. • Data storage is not free. • Must pay in DAT TOKENS.
Streamr	Streamr offers a marketplace for real time data, leveraging blockchain and Ethereum-based smart contracts for security critical operations like data transfers https://streamr.network/discover/marketplace	AGPL 3.0	• Open Source publish/subscribe platform. • Low latency compared to other P2P networks. • Delivery latency can be predicted. • Less vulnerable to attacks. • Core web app also offers Canvases for data processing and analytics. • Integrates with Apache Spark.	• Not yet feature complete. • Not yet fully decentralized. • Work in progress.
OceanMarket	OceanMarket is an open-source community marketplace for data, built on the foundation of datatokens. https://oceanprotocol.com	Apache Licence 2.0	• Open source B2B data asset trading platform based on contracts run on Ethereum Mainnet. • Enables monetization of private data while preserving privacy. • Allows algorithms and models to run on on-premise data. • Supports automatic determination of price.	• Must use OCEAN tokens. • OceanMarket app is still in Beta version. • Stakers have the risk of losing funds.

Technology Name	Short Description, URL	License	Pros	Cons
			• Allows earnings by staking/curating on data • Provides tools for developers to build their apps • Is part of the OceanProtocol ecosystem	
GAIA-X	GAIA-X is a project for the development of an efficient and competitive, secure and trustworthy federation of data infrastructure and service providers for Europe, which is supported by representatives of business, science and administration https://www.data-infrastructure.eu/GAIAX/Navigation/EN/Home/home.html	Apache Licence 2.0 (not confirmed yet)	• Decentralised: GAIA-X connects centralised and decentralised infrastructures within a homogeneous system. • Secure, trustworthy and confidently: The goal is to create a modular, secure, trustworthy and user-friendly system, that brings together existing cloud providers and their services and in which data and applications can be handled in a way, that ensures full control over these. • Transparent: Small and medium-sized enterprises in particular, benefit from transparent markets, broad access to customised services and the options becoming available as a result. • Open: The data infrastructure will be based on the open source principle.	• Still at the stage of prototype implementation. • Work in progress.
S5 Share Platform	S5 Share offers a data marketplace to navigate and search over datasets based on their metadata and allows different stakeholders to express their interest over a data asset and acquire it on the basis of a smart contract mechanism that is powered by the Ethereum blockchain.	Proprietary Business License	• Inclusion of licenses metadata for defining and enforcing IPR restrictions • Easy negotiation over smart contracts • Enforcement of smart contracts for their duration • Support for sharing and trading between different stakeholders	• Not tracking/ enforcing legal terms expressed in natural language • Lack of contract extension functionalities • Not open source

2.2.9.1 Architectural Considerations

Since Data Sharing and Trading is an emerging trend and it is based on emerging technologies like blockchain, many of the available tools and platforms lack in maturity and completeness. Nevertheless, they can be the basis or the live paradigm for any architecture that aims to implement a modern data marketplace.

Dawex platform, in particular, is suitable for an architecture where the tool can run in a centralised deployment as there is no possibility to scale on distributed environment. Ideally it should be hosted on a server with significant resources.

Datum is better suitable for structured data sharing, while Streamr is more suitable for real time, streaming data sharing with a notable advantage that it integrates with Apache Spark. OceanMarket,

on the other hand, is a very promising initiative that, among others, addresses the challenges of data privacy and data pricing using state-of-the-art techniques.

GAIA-X is a project initiated by Germany and France and supported by other EU memebers that aims to create an open digital ecosystem and a European data infrastructure. In XMANAI, as the GAIA-X maturity increases (since it's still in its infancy), the alignment with the various technical concepts and architectural decisions introduced by GAIA-X will be particularly pursued.

Lastly, the S5 Share Platform is relevant for the asset sharing scope of XMANAI since it provides a metadata catalogue for datasets as well as a flexible blockchain-based mechanism that allows the involved stakeholders to reach an agreement on the sharing terms. It allows users to easily register their datasets and enforces the contract terms in terms of who has access, under what conditions and for how long to a specific dataset.

2.2.10Smart contracts

Smart contracts are computer programs or transaction protocols intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract, where the terms of the agreement between buyer and seller being directly written into lines of code.

An overview of relevant Smart Contract technologies is presented in the following table.

Table 2-14: Overview of relevant industrial Smart Contract Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Ethereum	Ethereum is a decentralized, open-source blockchain featuring smart contract functionality. https://ethereum.org/en/	GNU GPL	• Ethereum is very popular, mainly because it allows developers to build decentralized applications on top of it. • Robust and efficient consensus mechanism (Proof of Work). • Has simple and modular architecture. • Network has no downtime.	• Expensive for complex contracts ("gas" cost). • Security issues (e.g. Integer overflow bug). • Scalability issues. • Maintenance issues.
Hyperledger Fabric	Fabric is an open, modular, proven, enterprise-grade, distributed ledger (blockchain) platform with advanced privacy controls. It is part of the Hyperledger community that builds components for blockchain technologies. https://www.hyperledger.org/	Apache License 2.0	• Designed for B2B. • Permissioned architecture. • Highly modular. • Pluggable consensus. • Open smart contract model — flexibility to implement any desired solution model. • Low latency of finality/confirmation. • Flexible approach to data privacy. • Designed for continuous operations. • Governance and versioning of smart contracts. • Flexible endorsement model for achieving consensus across required organizations. • Queryable data (key-based queries and JSON queries). • Compatible to Hyperledger Composer tool	• No built-in cryptocurrency (you have to create your own if you need one). • Complex architecture. • Minimum APIs and SDKs. • It is not network fault-tolerant. • For smart contracts language it supports only Golang and Javascript

Technology Name	Short Description, URL	License	Pros	Cons
Hyperledger Sawtooth	Hyperledger Sawtooth offers a flexible and modular architecture separates the core system from the application domain, so smart contracts can specify the business rules for applications without needing to know the underlying design of the core system. https://www.hyperledger.org/use/sawtooth	Apache License 2.0	• Belongs also to the Hyperledger family • Supports both permissioned and permissionless scenarios. • Superior consensus mechanisms. • Superior fault tolerant support • Multi-language smart contract support: Rust, Java, Javascript, Go, Python. • Support for EVM and Solidity using Seth.	• Lacks in membership management • Requires more time to develop and deploy. • Minimum APIs and SDKs. • It is not network fault-tolerant. • Not Compatible to Hyperledger Composer tool
Cardano	Cardano is an emerging 3rd generation proof-of-stake blockchain network, being developed into a decentralized application (DApp) development platform with a multi-asset ledger and verifiable smart contracts. https://cardano.org/	Apache License 2.0	• It is driven by a scientific philosophy and as such it is based on a lot of research. • It includes leading-edge technologies, models and methodologies	• Lot of research means that it is evolving at a slow pace • Not yet fully tested
Corda	Corda is an open-source permissioned blockchain platform designed and built for the recording and automation of legal agreements on private transactions between identifiable parties. https://www.corda.net	Apache Licence 2.0	• Open source, designed for B2B. • Supports wide range of databases like postgres and oracle • enables the sharing of data in a network without the need for a central controller • Consensus is achieved at the level of individuals transacting, rather than the entire system at large • Geared towards handling complex transactions • Adheres to highest privacy and security standards • Nice documentation	• Heavily influenced by the financial sector. • Peer to peer messaging hinders global broadcasts. • Does not have a native currency.
Quorum	Quorum is an open source blockchain protocol specially designed for use in a private blockchain network, where there is only a single member owning all the nodes, or, a consortium blockchain network, where multiple members each own a portion of the network. Quorum is derived from Ethereum by modifying the Geth client. https://consensys.net/quorum/	Apache License 2.0 (GoQuorum is LGPL-3.0)	• Allows only trusted nodes to participate in the blockchain. • Allows contracts to be deployed and transactions to be sent to a subset of participating nodes in the blockchain • Proof-of-authority based consensus, which provides immediate block finality, reduced time between blocks and high data integrity and fault tolerance. • It creates blocks “on-demand,” faster block times in the order of milliseconds instead of seconds and transaction finality (absence of forking).	• Mostly targeted towards the financial service industry
IOTA	An open, feeless Data and Value transfer protocol that recently started to support smart contracts. Based on	Apache License 2.0	• IOTA Foundation (foundation under German law) is a non-profit organisation with headquarters in Berlin.	• IOTA has accepted a lot of criticism mainly due to security issues and vulnerabilities

Technology Name	Short Description, URL	License	Pros	Cons
	IOTA token and the Tangle network. https://www.iota.org/		• Each smart contract can be executed in a localized context without forcing the whole network to execute them	(https://iota-beginners-guide.com/criticism/). • Smart contracts support is currently in pre-alpha release.

2.2.10.1 Architectural Considerations

Ethereum is suitable when a simple, general-purpose, decentralized architecture is required that facilitates smart contracting on digital assets. It runs on a virtual network known as the Ethereum Virtual Machine (EVM).

Hyperledger Fabric is a more advanced, complex and flexible solution for smart contract support and is the preferred platform used in most of the enterprises today. Suitable when you want to build your own cryptocurrency or if you do not need one. Hyperledger Sawtooth is similar to Fabric, but it also supports permissionless scenarios. It is more flexible than Fabric with superior consensus mechanisms and smart contract language support, but its main drawback is the time it takes to develop and deploy. Both are part of the Hyperledger ecosystem backed by the Linux Foundation.

Corda and Quorum, are another set of blockchain integration platforms with smart contract support mainly for private B2B transactions (permissioned network). They are both influenced by the financial industry. In quorum, the available currency is Ether and programming language is Solidity, while Corda has no native currency and is written in Kotlin.

Lastly, Cardano and IOTA both seem not yet suitable for integration, due to their low maturity in different aspects related to smart contracting.

2.2.11 Data Quality Curation

Data quality curation can be defined as the service of ensuring that the processed data fulfil some predefined standards. Data outcomes derived from big data processing pipelines often suffer from poor quality of input data. It is known that all data-centric applications are as good, as the data comes in. Hence, it is crucial to define some quantitative and qualitative standards that the data should fulfil, in order to have valuable outcomes. This is the task of data quality curation. In Figure 2-9, we present the steps of quality assurance in a high-level abstraction.

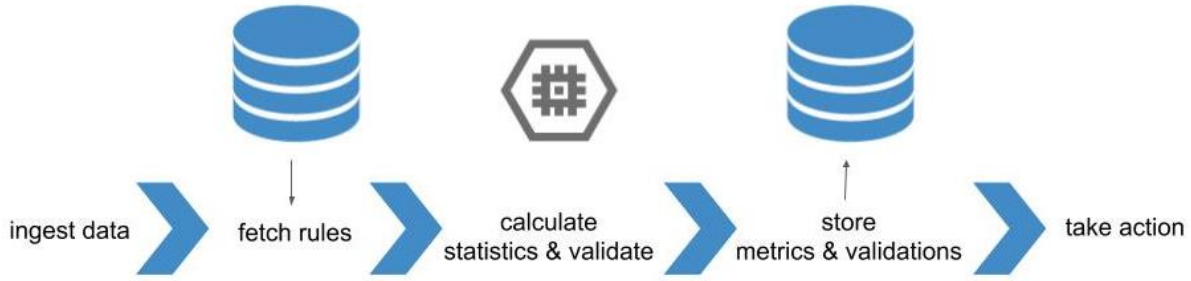


Figure 2-9 Data quality curation steps in a high-level view.

In this section we focus on open source, freely available tools for data curation and quality assurance, which are summarized in Table 2-15.

Table 2-15: Overview of relevant industrial Data Quality Curation Technologies

Technology Name	Short Description, URL	License	Pros	Cons
Great expectations	A free, open source tool for data profiling, curation and pipeline testing. https://greatexpectations.io/	Apache License 2.0	• Open source • Multiple backends (pandas, Spark, SQL) and multiple data sources (files, S3, data warehouses). • Exposes python API.	• Focuses less on data profiling (more on validation). • Docs lack in clarity and conciseness.
Apache Griffin	https://griffin.apache.org/	Apache License 2.0	• Integrates perfectly with Apache Ecosystem • Scales easily to big data • Data quality rules are defined in a configuration file	• Integrates only with Apache Spark
AWS Deequ	https://github.com/awslabs/deequ	Proprietary	• Simple interface • Integrates nicely with Spark Dataframes	• Small collection of functionalities • Supports only Spark Dataframe as data source
Tensorflow Data Validation	https://www.tensorflow.org/tfx/data_validation/get_started	Apache License 2.0	• Integrates well with TFRecords • Nice data profilin	• Doesn't scale to big data
OpenRefine	A free, open source, Java-based tool for cleaning, transforming and extending datasets. https://openrefine.org/	BSD-3-Clause	Open source, supports various file formats, but you can also import data from other sources (like database, web link, google sheets, etc)	• It runs locally, and due to this, it does not scale (limited to the computer's RAM). • Long learning curve as it uses its own javascript-like expression language (GREL).
Talend	Talend open studio for data quality is a free, open source, Java-based data quality tool based on Eclipse IDE environment. https://www.talend.com/products/data-quality/data-quality-open-studio/	Apache License 2.0	• Open source, versatile and easy to use tool with pre-built widgets. • Has a modular approach (suite of tools) and integrates with various databases, web services (and FTP).	• It is the free version of Talend's commercial set of tools and as such has limited functionalities. • Not easy to setup, bad support, many bugs.

2.2.11.1 Architectural Considerations

Great Expectation is an open-source tool that integrates nicely with all critical data sources and data transformation tools, e.g. Pandas, Spark, SQL. It provides a collection of data quality functions that, when violated, automatic warnings or errors are raised. Great Expectation can be integrated into the application code or configured externally through a Command-line interface (CLI). Apache Griffin is a tool for data quality assurance belonging to the well-known Apache ecosystem. It integrates only with Apache Spark and supports data assurance tools in a distributed cluster of computers. Deequ is a much simpler tool than Great Expectation. It is built on top of Apache Spark, and it only requires a Spark dataframe as input. TensorFlow Data Validation is a tool that integrates nicely with the Tensorflow ecosystem. Its primary drawback is that it supports only the TFRecords input format, and it doesn't

scale to big data. Finally Talend is a proprietary platfom for building ETL pipelines. As part of its services it supports data quality assurance.

In conclusion, Great Expectations, Apache Griffin and AWS Deequ are all worth-noticing alternatives for XMANAI's data quality curation part. Great Expectations is a complete tool but also demanding in the configuration part. It can be the main workhorse for building a comprehensive quality assurance platform. Apache Griffin and AWS Deequ – especially Deequ - are lighter alternatives, only working with Apache Spark Dataframes. If our data comes only in this format, they are both reliable alternatives.

OpenRefine and Talend, despite their beneficial features, are both software tools that have been designed to run locally on a computer, and therefore cannot scale.

Great expectations, on the other hand, is a scalable Python-based library for data ingestion, validation, and quality assurance with many modern services and conveniences.

2.2.12 Provenance Engine

The section presents an overview of the technologies for a provenance engine, which will be considered for use in XMANAI. The initial consideration of their suitability is given in the following subsection.

Table 2-16: Overview of relevant Technologies for a provenance engine implementation

Technology Name	Short Description, URL	License	Pros	Cons
IDS Clearing House	IDS clearing house provides decentralized and auditable traceability of all transactions. http://internationaldataspaces.org	N/A	• Central Logging Component of the IDS architecture • Works with other IDS components • Audit logging for legal questions and transaction logging for provenance information • Provenance Dashboard in development • IDS: Industry concept/standard to be adapted and most likely integrated into GaiaX	• Works best in combination with other IDS components. Data sharing would call for extensive use of IDS standard • Sparce documentation • Just a prototype implementation available (https://github.com/Fraunhofer-AISEC/ids-clearing-house-service)
Apache Jena	Jena is a free and open source Java framework for building Semantic Web and Linked Data applications. One of its components, the reasoner, allows to enable derivation logging	Apache License 2.0	• Simple API • Works with different types of reasoners • Already implemented and open source	• Requires the use of the whole Jena software • Uses Java as programming language • Inference is based on the original graph and not on an embedding

Technology Name	Short Description, URL	License	Pros	Cons
Prov-package (python)	This Python package is a full implementation of the W3C PROV recommendation, which is an ontology to trace the provenance of data (see https://www.w3.org/TR/prov-overview/) https://pypi.org/project/prov	MIT License	• Simple API • Easy integration in the platform, since it's a python package • Ontology developed specifically for industry	• Not ready to use, requires effort • No UI
Git	Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. https://git-scm.com/	GPLv2	• Efficient version control based on stored differences from one version to a following one • distributed and collaborative • branching and merging allows parallel states of versioned files and multiple workflows	• Not efficient for binary data – tracking of databases that store data in binaries would not make sense • GPL license
Liquidbase	Liquibase enables track, version, and deploy database schema changes. This is achieved by control database schema changes for specific versions, automatically order scripts for deployment and easily rollback of changes. https://github.com/liquibase/liquibase	Apache-2.0 License	• Repeatable migrations (Perform rerunnable and non-rerunnable changes) like storage procedures • works with a wide range of databases • enable rollbacks to undo changes based on date, operation counts in history, tags • succeeding DB states can be recalculated after changes in a previous state by applying repeatable migrations in combination with rollsbacks	• differentiate between commercial and free features (for future development)
Sqitch	Sqitch is a database change management application. https://sqitch.org/	MIT License	• changes of the database can be reverted based on changeID • dependencies between changes can be declared • works with a wide range of databases	• does not provide support for NoSQL or GraphDBs
Metagration	Metagration enables creating and managing migrations as stored procedures. https://github.com/michelp/metagration	MIT License	• migrations can be stored and run as procedures • restoring previous DB states is possible • stored procedures can be imported and exported from sql files	• is bound to PostgreSQL

2.2.12.1 Architectural Considerations

Data provenance engines provide features to trace data manipulation and its utilization starting from the creation to record a lineage. This allows particular entities to comprehend the provenance of given data.

One solution or technology that covers all needed features does not exist – to our best knowledge.

To enable the tracing of data manipulation and its utilization, these activities can be logged by transforming information about respective events into semantic triples/RDF triples and added to a triplestore. There is not much documentation on the web about derivation logging and no comprehensive studies about implementations and technologies can be found. However, many of the sources with regard to derivation logging point to Apache Jena. This framework is based on an RDF graph and enables management of metadata in the form of triples. Furthermore a lot of documentation and many tutorials for different use cases are provided. Therefore Apache Jena can be a component of a suitable solution.

As an alternative the python library PROV also targets the management and processing of metadata. It is an implementation of the W3C PROV Data Model[22]. The library's documentation is unfortunately incomplete and the development state seems to be ongoing, since it is still a prototype by all appearances. Therefore an implementation[23] based on that library seems to involve a lot of unplanable effort, even there already exist an implementation. It is a public repository for documents, which is the first to completely implement the W3C PROV recommendation.

IDS Clearing House is a framework that enables the logging of data transactions between entities. It does unfortunately not provide enough flexibility to integrate it as a component of a suitable solution to trace data manipulation and utilization and combine that with respective data versions.

The tracing of information about data utilization and mutation events is one part of problem that needs to be solved. The other part targets versioning of the data and the association of theses data versions with the corresponding metadata events. Liquidbase is a tool that enables version control of a database. It is a popular solution that is widely used and has a big community. The recovering of previous database versions is possible. In addition migrations as stored procedures can be rerunned to recalculate a defined state after the underlying dataset has changed. This tool fulfils the requirements and can be a component of a suitable solution.

Git as widely known solution for version control is really efficient and allows parallel workflows. It is unfortunately published with a GPL license and cannot be efficiently applied when a relational database stores data as binaries on the disc. Therefore Git is not a suitable component.

Sqitch as a database change management application offers similar functionalities in comparison with Liquidbase when it comes to rollbacks of database versions. It also provides a comprehensive documentation and many tutorials. Furthermore the developer do not offer commercial features or services for their software. Therefore all available features can be used for an implementation in XMANAI context.

Metagration as a migration tool written in PostgreSQL. The area of application is limited on PostgreSQL. It enables restoring of previous database versions, but in comparison with other presented versioning frameworks the features are limited. In addition, the software is maintained by a way smaller team. As the result the integration of Metagration as a component would not fulfil the requirements.

[22] http://www.w3.org/TR/prov-dm/
[23] https://openprovenance.org/store/

3 Industrial Asset Management and Sharing in XMANAI

3.1 View as a whole

On the basis of the landscape analysis presented in the Section 2 of this document this section describes the approach of the industrial assets management and sharing functionalities in XMANAI by making use of orchestrated components belonging to respective service bundles highlighted in the XMANAI architecture diagram below that is presented in detail ni Deliverable D5.1.

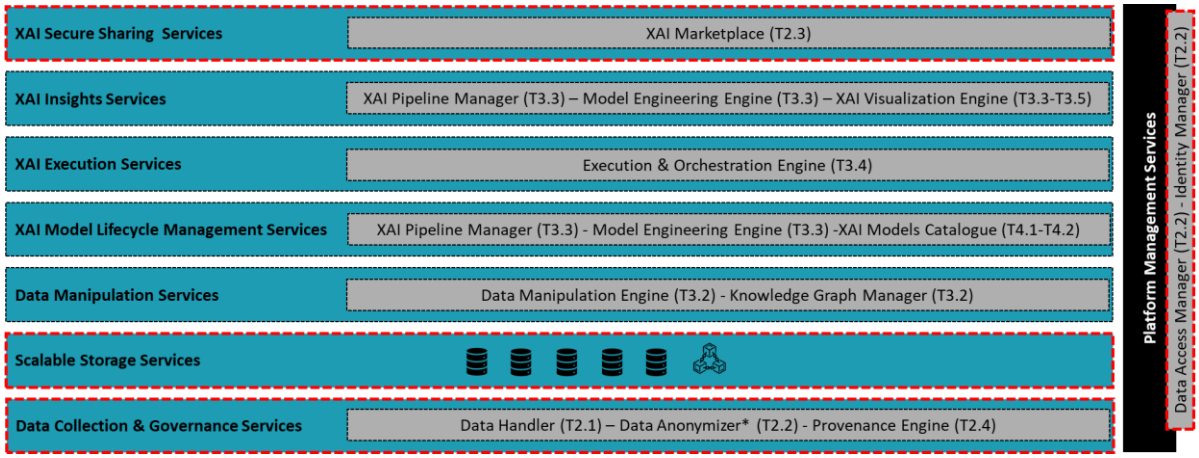


Figure 3-1 XMANAI architecture overview

The data is going to be stored and managed by the Platform Storage services. The Data Handler will enable data collection from external data sources to the Platform Storage Services, data export for a download feature from XMANAI and access to the XMANAI data for external 3rd party services through an API. The Data Handler consist of:

- API Data Harvester collecting data from external APIs.
- File Data Harvester transforming data files to the XMANAI data model and saving data as XMANAI datasets in the storage.
- Data Gateway providing an API to get and push data from/to XMANAI data store.
- Data Exporter exporting data snapshots in requested file formats.
- File/Data Manager for managing data, models and other files associated with data analytics projects in XMANAI.

The monitoring and logging of modifications and access to data will be done by the Provenance Engine. The Data Anonymizer will be realised as a standalone tool executed on the data provider side. The data will be anonymised before sharing it with the XMANAI platform. The security mechanisms in the platform will be realised with help of Identity and Access Manager, Policy Engine and Policy Editor. The data sharing between the data provider and data consumers will implemented in the XAI Marketplace component consisting of the Contract Manager and Registry/Metadata Manager, whose mission in the platform is the management of metadata for all data, analytical models and other assets in XMANAI. The figure below provides an overview of these components and their subcomponents implementing the industrial assets management and sharing functionalities.

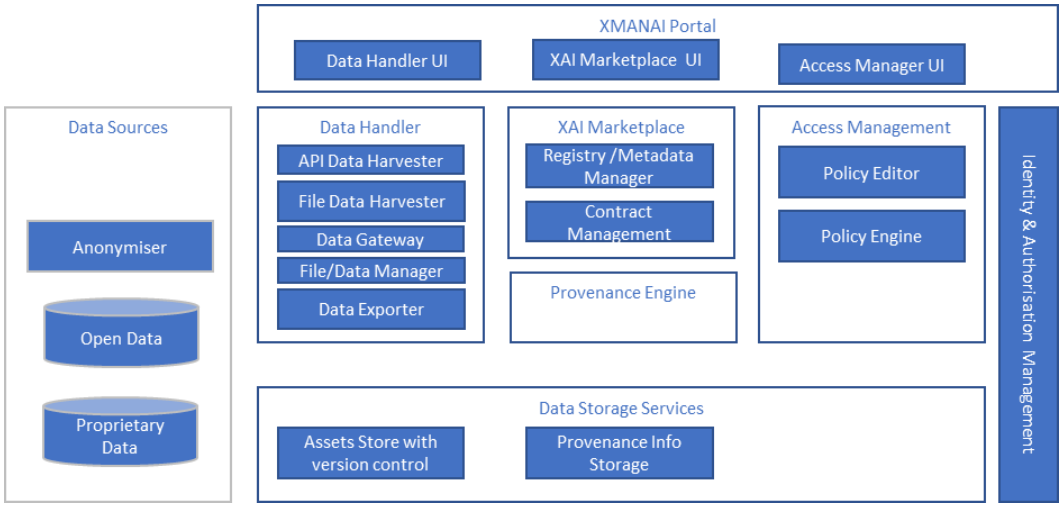


Figure 3-2 Overview of components providing industrial assets management and sharing functionalities

The following chapters present the components in detail including descriptions of their implementations on technical level and mock-ups of their user interface. These components implement relevant presumed methods to provide industrial assets management and sharing functionalities, and meet the technical requirements and MVP features specified in the Deliverable D1.2. With regard to the component descriptions, the following remarks are relevant.

Asset Management in XMANAI.

The management of assets in XMANAI should meet a number of critical requirements. One of them is the explainability of data, since Explainable AI is the main objective in the project. Data engineers and data analysts working in XMANAI should be able to understand the data, its structure and its semantics. This is an important requirement, not just for working with the data in XMANAI, but for sharing high quality data with third parties as well. Another important requirement is that the users of the XMANAI platform should be able to query data. Therefore, data management components should provide an API for that. The main challenge though is in the heterogeneity of data. The XMANAI platform is not restricted to support a limited set of data analytics scenarios or the data which is collected and handled by the platform. It can vary referring to structure, semantic, volume and dynamics. Apart from data, XMANAI should be able to manage other types of assets as well.

The asset management definition takes into account a number of key decisions that defines the design of the data related components of the platform:

- **Types of assets in XMANAI**. The XMANAI asset management has to support different types of assets including data files, data collected from external APIs, data pushed in XMANAI through its API, data analytics scripts, metadata and more. For managing the assets we will distinguish three types of them: files (of any type), structured data and metadata according to the XMANAI data model. The ability to collect and manage any files enables the required flexibility in addressing the heterogeneity of assets. The conformance of data and metadata to the XMANAI data model enables the provision of an advanced query interface.
- **Datasets and role of metadata**. Sharing of data and provision of transparency about the data transactions, data structure and semantics demand the availability of high-quality metadata in XMANAI. The information stored in metadata will be used to provide the knowledge about the data to XMANAI users. A set of data and its metadata is called a dataset.
- **Data structures in XMANAI.** The heterogeneity of data and the need for an advanced query interface requires a special approach when it comes to data structures handling in XMANAI. The main idea of our approach is to define the most suitable data structure for storing and managing data in each particular case. The definition of this internal XMANAI representation

for storing the data should be done based on the data structure and semantics provided by the users who register the data in the XMANAI platform. This registration includes a description of data that is done by the allocation of data fields to elements of the XMANAI data model. In particular, for the first versions of the XMANAI platform the consortium decided to focus on tabular structures as the internal representation of data. The decision is motivated by the assumption that the data emerges in the XMANAI demonstrators have tabular structure (taking into consideration the early data profiling documented in Deliverable D1.2). Another important reason is the simplicity of working with tabular data. The process of creating a data structure for the internal representation of data in XMANAI includes the following steps:

- The user describes the data structure and semantics of a concrete data file(s) or data stream using the elements from the XMANAI data model. This information, provided by the user, is saved as part of metadata for the data in the XMANAI metadata registry.
- On the basis of this information about the structure and semantics of data, the XMANAI platform creates a table with a suitable structure in a relational (or potentially other type of) database and adds the respective data to it.
- The information about the created table is added to the corresponding metadata in the metadata registry. This information can be used to query the table of the database and get, update, modify, delete data or perform more complex queries involving more than one table.

This relatively simple data handling approach in XMANAI can be extended by adding other types of databases and allowing users to select the most suitable internal representation.

Data and Asset Provenance in XMANAI

The provenance of data and assets is realised in XMANAI by several components. Therefore, it requires close interaction between stores, i.e. database systems, and management layers, which handle the metadata.

All data, added to XMANAI, will be traceable. Since many types of representations of data exist, several differentiations have to be determined. In XMANAI, we consider data in the database and as files. So there are two types of data that matter - files, which are text-based or binary, and data organised in database tables. A further differentiation targets the storage format independently of the data representation. Once data is added to XMANAI, a provenance process begins that logs each transaction with the respective data. All emerging data is versioned by a Version Control System (VCS). The traditional VCS has a focus on text-based files. Therefore the implementation of such a VCS for the existing data representation is benefitial because only changes are stored, and no exact copies are created. Binary files can also be versioned using this approach, but each version is stored as a new, modified copy of the previous one. This is not very efficient but effective.

A persistent storage has to ensure non-interrupted versioning. This can be enabled by an asset store in combination with version management. With this approach, it is possible to version data, scripts and other commands for analyses and modelling. The asset store saves all assets and makes them available. In addition all versions of all assets are also available and can be reproduced by the version management.

A relational database system saves the databases in binary files. Thus, only the database system knows the structure of the stored data. This makes versioning at the file level difficult. Nevertheless, solutions exist that enable versioning at the database level. Some suitable solutions have already been discussed in chapter 2.2.12.

Applying the above-noted methods, versioning can be implemented and completed for every data representation provided for XMANAI.

In addition to the versions that are created of the existing data, a log of emerging modifications is needed. For this purpose, the Provenance Engine exists, which consists of a metadata handler and a triplestore. Each new version of a dataset is also stored in the Provenance Store in the form of descriptive metadata. The triplestore implements the W3C PROV recommendation, which ensures that provenance information are stored according to the standardised approach. As a result inter-operable interchange of provenance information can be achieved. The metadata consists of information about dataset modification, versions, transactions and modifications that were executed. In addition information about the performing entity of the respective transaction is stored. In this case, a performing entity can be a user or a computer programme, for example, when an algorithm cleans the data.

The combination of versioning in the Asset Store and logging in to the Provenance Store ensures that all data has a history and a user can track and trace every state.

An extension of this approach is logging of data accesses. A Provenance Engine is configured for this purpose. It logs which entity accessed an existing dataset. Even if the entity does not edit the data, the access should be logged as an activity.

By logging all transactions in a Data Provenance Store and combining it with a version history of the data, all modifications and and accesses are traced. If parts of data are unintentionally removed, the engine can restore them. It is also known at which point in time and by which entity the data was manipulated in order to provide transparency and build trust in the system.

This approach means that the provenance of data and assets in XMANAI is considered from the very beginning and is taken into account at every step.

Data Asset Sharing in XMANAI

In XMANAI, data asset sharing broadly embraces the cataloguing, navigation, expression of interest, agreement and transfer of different types of data assets ranging from datasets to Explainable AI (ML/DL) models and results (as described in the XMANAI Deliverable D3.1) among the involved stakeholders. The XMANAI data asset sharing methodology is based on best practices as provided by the literature review of the industrial asset sharing domain 2.1.4. In practice, the five conceptual steps enable trustful data asset sharing among the involved parties include:

I. Preparation for data sharing: Data asset owners are able to properly describe an asset (with metadata), define its licence and intellectual property rights (IPRs), and then make it available for sharing and easily discoverable for acquiring by another user or another organisation in the manufacturing value chain.
II. Navigation to data assets: Potential data asset consumers may view the catalogue of available data assets (for which the access policies that have been set are satisfied), search for data assets of interest based on different criteria and browse their profile to identify whether they are appropriate for them.
III. Expression of interest for data assets: Potential data asset consumers may request access to a specific data asset of interest. This request is forwarded to the respective data asset owner in order to take a decision whether to proceed with satisfying it.
IV. Agreement on sharing terms: The respective data asset owner defines the sharing terms for the specific data asset in order to safeguard its organization's interests and express how the data asset should be used and under what restrictions by the data asset consumer. Since it is crucial to ensure non-repudiation and trust over such an agreement, smart contracts are generated and validated through Distributed Ledger Technologies (DLT), tracing all changes performed while the negotiation among the data asset owner and consumer are ongoing.
V. Enforcement of sharing contract: Once an agreement is in place and for the duration that it is effective, the data asset consumer may access the data asset and utilize it for their own ends as long as the terms of use (that are written in the smart contract) are respected. At any

moment, controlled access and monitoring of data asset usage (in accordance with the contract's terms) is verified by an integrated DLT mechanism.

In the context of XMANAI, sharing is enabled to facilitate different organizations and types of users (e.g. data scientists from external organizations to a manufacturer) to legitimately acquire access to data assets for a specific time frame. As a result such users can contribute to solving existing manufacturing problems. To this end, XMANAI contributes to clarifying the terms of use on a one-to-one (for simple data assets), as well as the one-to-many (for complex data assets that may have derived from other assets) basis, but does not intend to support monetary compensation for the smart contracts (as it is eventually considered as out of scope for the manufacturing problems addressed by the XMANAI demonstrators as explained in D1.2 and D6.1).

Security in XMANAI:

XMANAI adopts a holistic security approach in order to provide a secure and trusted environment that will facilitate the effective safeguarding of the assets of the platform and will increase trust of stakeholders in the XMANAI platform. This holistic security approach incorporates multiple security aspects which are applied across the XMANAI platform. It targets the complete lifecycle of the assets' exploitation, spanning from the overall access to the platform offerings that involves access and utilisation of the assets within the platform via the various operations performed them. The formulation of this holistic security approach is based on landscape analysis presented in the Section 2, and specifically the sections related to security and privacy (section 2.1.2) and trust considerations (2.1.3).

At first, access to all platform's offerings, services and assets is granted only to registered and successfully logged in users. To achieve this, two different aspects are effectively covered, the user account management lifecycle and the authentication of the identity of the users. One of the main pillars of the holistic security approach is the user account management lifecycle that covers the controlled registration of the users of the platform via a robust registration process and a regulated user invitation process. These processes provide the basis for the solid XMANAI authentication process to secure the access to the platform offerings, services and assets to only registered users. For this reason, the Identity and Authorisation Manager will act as the single core identity provider of XMANAI providing all the account management lifecycle operations and the proper authentication mechanism built on top of this identity provider.

Going one step further, the XMANAI holistic security approach, safeguards the security of the information exchanged by the various components of the platform. As the XMANAI platform architecture, as documented in deliverable D5.1, is composed of multiple components organised in different layers, it is obvious that the intercommunication of the components should be properly secured. To this end, the Identity and Authorisation Manager provides an authorisation mechanism that controls which components can communicate and exchange information during the various operations of the platform. To achieve this, authorisation rules are defined and applied when different components intercommunicate in which intercommunication the Identity and Authorisation Manager plays the role of the mediator. With regards to the security of data in transit in this intercommunication of the components, the adoption of the Transport Layer Security (TLS) for the secure and efficient data transfer of information by all components is foreseen.

While the user management and the proper authentication of the identity of the users constitute the first level of the applied security of the platform, it is also imperative to employ an effective access control mechanism within the platform that regulates the access to all available assets. Since not all users should have access to all the assets available in the platform, the requirement arises for the owners of the asset to be able to define which registered users should have access to their assets and under which conditions. Hence, two additional aspects of the holistic security approach related to authorisation are covered, the ability of the owners to define the proper authorisation rules for their

assets based on their needs and the mechanism that regulates each access request based on these authorisation rules. To this end, the Policy Editor provides the complete access policy lifecycle management that facilitates the assets' owners to define, update or even eliminate the authorisation rules in the form of access policies that will be applied on their assets. Additionally, the Policy Engine provides the effective and robust access control mechanism that intercepts all access requests in order to formulate an access control decision based on the defined access policies. In this way, all assets of the platform are protected from unauthorised access based on the needs of the assets' owners.

3.2 Data Storage Services: Assets Store with Version Control

This and the following sections present the components for industrial asset management and sharing depicted in Figure 3-2 in detail.

3.2.1 Overview

The Assets Store with Version Control is a component responsible for storing XMANAI assets. Version control as a feature will help to avoid losing data or other important information in the experimental multi-user data analytics XMANAI environment and if needed will help the user to get the required version of data or other XMANAI asset. From the storage perspective there are the following asset types to be stored in this component:

- Files of different type including data files, data analysis or processing script files, any other files. The Store has to be able to store and manage any files without awareness about their content. It should provide an endpoint for adding, updating, removing the file(s) and getting the specified version of the file(s).
- Structured Data conforming to XMANAI data model. The Assets Store has to be aware about the structure of the data and provide the suitable storage structure to store it as well as an endpoint for data querying and modification.

The heterogeneity of the present asset types makes a decision complicated to find a solution that provides optimal database management for all of the asset types. Since structured data, which comes along with a data model, can be efficiently stored and managed in a relational database. Files of different types can either be transformed into structured data by the Data Handler: File Data Harvester or have to be stored and managed as binary files. The data in those files can not be analysed and extracted from them. The management of binary files can also be done by relational database management systems, however, with additional overhead. According to that, the files are either directly stored as binary data in the database or as files in a file system that are linked in the database. Considering that most of the data can be provided as, harvested as or transformed into structured data, a relational database is an appropriate component for an asset store. Nevertheless, it is thinkable to store binary files in a file system.

With reference to the Provenance Engine that links metadata to binary files and datasets, an asset store solution has to closely work with it. Thus, the version control of the asset storage is closely intertwined with the Provenance Engine that attaches metadata to versions of the data and links them in a triple graph.

3.2.2 Technology

The Assets Store will be developed on the basis of an existing relational database management solution. PostgreSQL is a powerful and efficient DBMS. Also, binary files up to 100MB can be efficiently managed by PostgreSQL with less overhead. Larger files can be stored in the file system with a link in the particular database management system. Moreover Postgres as a RDBMS is supported by state-of-the-art version control solutions presented in 2.2.12. Such an overlaying version control component

provides features that are embedded in the Provenance Engine to link specific versions of data to metadata.

3.2.3 Mockups

The Assets Store with Version Control is a back-end component without a GUI.

3.3 Data Handler: Data Gateway

3.3.1 Overview

The Data Gateway (DG) is the central service when it comes to data handling. It is responsible for the data flow through the XMANAI Platform, as it exposes the interfaces needed to perform the data push and pull operations. According to this, the DG manages and controls data activities and the corresponding information about these events. The component will provide a set of functionalities via its API. These functionalities target updating and getting data in the XMANAI data store(s):

DG1. Add/update Data: this operation is requested by internal and external services that need to add/modify data in the XMANAI data store(s). The external contributors as well as XMANAI services interact with a publicly available interface to push data into the data store(s).

DG2. Get Data: this operation can be requested by internal or external services to get data from the XMANAI data store(s).

DG3. Delete Data: this operation can be requested by internal or external services to remove data from the data store(s).

3.3.2 Technology

The Data Gateway component

features a standalone web server that expose an HTTP REST API. A REST API is composed by an URL pointing at the service that is responsible for its handling, plus any number of parameters that are needed by the receiving service, in order to carry out the requested task. Moreover related meta data about tasks are gathered and submitted to the Provenance Engine. The technologies that can be used in order to build this component have to be picked from a set of modern technologies that support API handling and expose a web service to intercept the API requests. Among these, the choice depends on implementation details and, therefore a list of suggested technologies is depicted below:

- Python: an interpreted, object-oriented, high-level programming language that comes with a wide variety of modules to deploy web servers (e.g., Django, Flask, ...) and is plenty capable of processing API requests.
- Java: a high-level, class-based, object-oriented programming language that comes pre-packed with all the tools needed to deploy the Data Gateway and to handle all types of HTTP requests.

The technologies needed instead to interact with the Data Storage component can be selected among the previous ones (as they also support the creation and delivery of HTTP REST requests), plus the following one:

- JavaScript: the programming language that has become the de-facto standard for the web. It offers built-in tools to build, send and receive HTTP requests, including POST ones.

By offering multiple ways for external contributors to interact with the Data Gateway component, the API callers can incorporate the request invocation in their current workflows, without the need to refactor the existing services. If, instead, the request needs to be performed manually, a tool like Postman is needed in order to construct the POST requests since those (unlike the GET ones) can’t be

manually handcrafted. Postman is an API platform that comes with a wide variety of tools needed to design, build, test and maintain APIs. In the scope of the interaction with the Data Gateway component, this tool will be only used to craft HTTP POST requests that will be delivered to DG. All details about Postman can be found in the official documentation[24].

3.3.3 Mockups

This component does not provide a graphical user interface.

3.4 Data Storage Services: Provenance Information Storage

3.4.1 Overview

The Provenance Information Store is mainly responsible for managing the metadata for modifications. Based on the CRUD operations that a dataset can perform (create, read, update and delete), relevant metainformation about the dataset is stored in the Provenance Information Store. For this purpose, the World Wide Web Consortium (W3C) has developed a recommendation that includes a metadata schema. This recommendation is called W3C PROV, and at its core, it consists of three entities that interact with each other. A detailed description of this standard can be found in section 2.2.12.

The Provenance Information Store consists of a database and an additional management layer, which is dedicated to the handling of tracking modifications. For example, if modifications or access to a dataset are registered inside the XMANAI platform, a message will be sent to the Provenance Information Store. This will make a related record in the metadata store. This ensures that users can track accesses and modifications over time.

The W3C PROV recommendation has three entities for this. The first entity is the dataset itself and its metadata - called the " Entity". Next, the entity "Agent" is either the person or the computer program that makes modifications or accesses the dataset. Finally, "Activity" saves the history of the changes and accesses.

Based on these requirements, the interface that a Provenance Information Store must provide is similar to the CRUD operations for traditional database systems. For example, new metainformation is saved via this interface when changes are made (create, update or delete). However, even in the case of simple access to a dataset, according to this recommendation, it can be guaranteed which agent (person or script) has accessed a particular entity (dataset). The entity "Activity" therefore is logging read access.

3.4.2 Technology

Because the W3C PROV recommendation uses the Resource Description Framework format (RDF), data storage in the form of a triplestore is recommended. The W3C PROV recommendation includes an ontology and a data model, which can be adapted and implemented for this purpose. Apache Jena and the TDB-Triplestore and Fuseki have proven to be very reliable and performant for this task. The triplestore saves the actual data as a graph in the form of triples, while Fuseki provides a SPARQL endpoint through which the management component can interact with the triplestore. One of the advantages would be additional use cases that the maintainer can flexibly implement via the SPARQL interface. Storing provenance data in triples in a Triplestore is especially useful, as users can insert later modifications quickly and without significant developments.

3.4.3 Mockups

This component will not have a user interface. Interaction is only provided via a console.

[24] https://www.postman.com/product/tools/

3.5 Data Handler: API Data Harvester

3.5.1 Overview

The API Data Harvester is a component responsible for collecting data from external APIs, transforming them according to the XMANAI data model and submitting the data to the XMANAI Assets Store. The component has to be extendable to support any type of the data source APIs. It has to provide a possibility to define the rules on how often and when the data has to be collected or updated.

The API Data Harvester consists of and orchestrates the following sub-components:

- **Scheduler** is used to plan when and how often the data has to be imported in the XMANAI platform from the original source. The user can define the dates and the intervals for the data import/update.
- **Importer** get the data from an external API. Importer works in combination with a **Connector**, which is specific for an API of the data source. Every API requires a dedicated Connector. If some data sources provide the same API then the Connector can be used for them with the API specific parameters.
- **Transformer** can be used to transform the data from the original format to the XMANAI specific format, which is supported by the integrated in XMANAI tools.
- **Exporter** is responsible for writing the imported data in the XMANAI data store.

The main functionalities provided by the API Data Harvester to the user are:

- **ADH.1: Creation and management of the data harvesting pipelines**. A pipeline defines the sequential execution of the mentioned above components (Scheduler, Importer, Transformer, Exporter) and their parameters for collecting data from a specific data source. The list of the orchestrated components can be specific for any data source and may include additional components not mentioned above, for example for anonymization, translation or cleansing of data.
- **ADH.2: Configuration of the data harvesting parameters** like:
  - Parameters of the connectors to access the external APIs.
  - Time plan for collecting the data.
  - Rules for transforming (and cleansing) of data from the original data model to the XMANAI data model.
  - Data storing parameters defining where data have to be saved in XMANAI.
- **ADH.3: Execution of the data harvesting pipelines** according the the defined time plan.
- **ADH.4: Monitoring the data collection and providing analytical reports**.

3.5.2 Technology

The API Data Harvester will be developed on the basis of the Fraunhofer's Piveau Consus software. Piveau Consus is a light, flexible and high performant open source back-end software for data and metadata harvesting. It is characterised by:

- high scalability thanks to microservices & container based architecture;
- high reliability thanks to use of independent stateless services;
- high extendibility & flexibility thanks to well defined & generic Interfaces;
- high maintainability thanks to use of DevOps.

3.5.3 Mockups

The GUI mockups of the component are presented in the figures below. They present the main interaction dialogues with the user. The user can create a new pipeline or monitor the status of existing data collection pipelines. On the right side of the screen the user can press one of the control buttons for a chosen pipeline to start/stop its execution, define a scheduler for it, see the pipeline details or to delete it from the list.

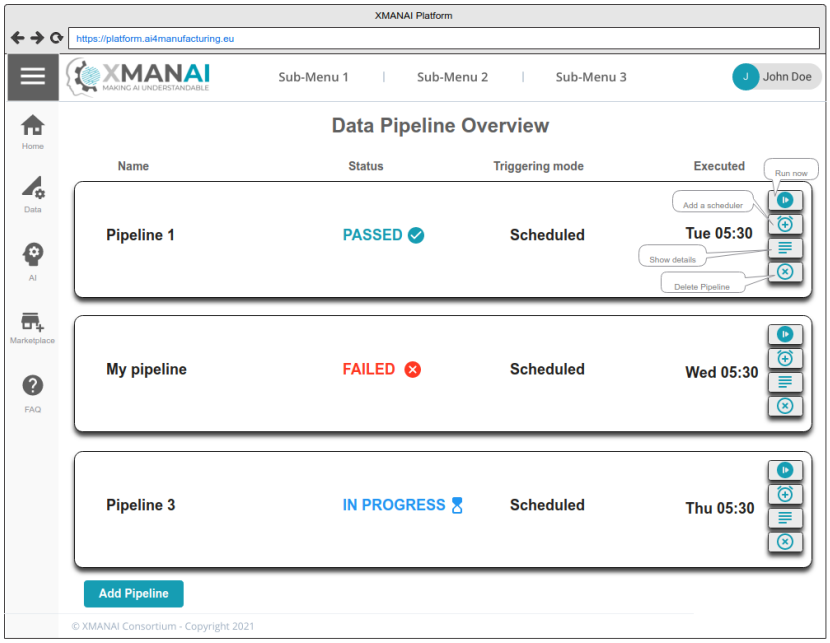


Figure 3-3 Overview of data harvesting pipelines

The figure below presents the dialogue for defining the scheduler for a data collection pipeline. The user has a possibility to create one or several regular trigger events for execution of the pipeline.

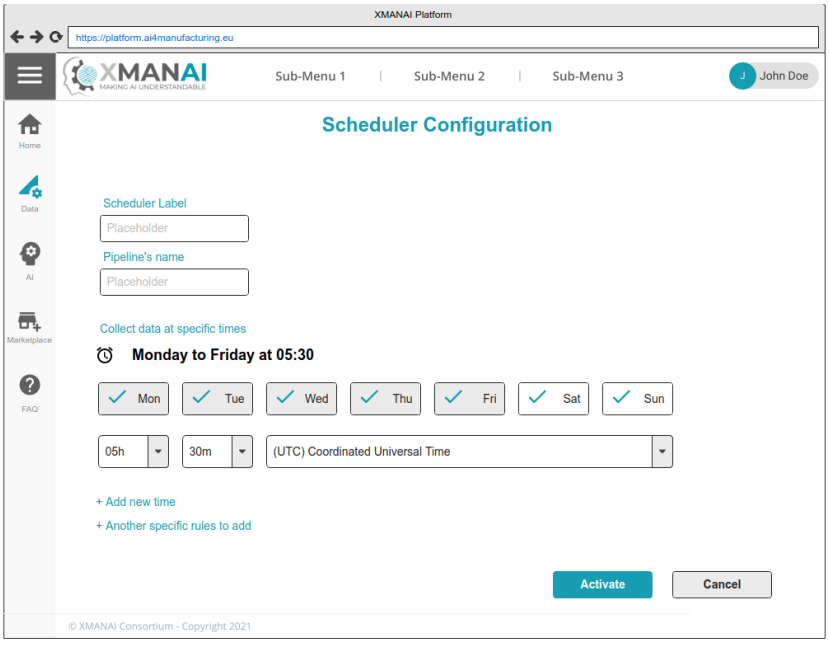


Figure 3-4 Scheduler configuration

The figure below presents the first tab of the data pipeline details dialogue dedicated to pipeline specification. This specification is provided as a json script specifying the connection details to the data source API and (optionally) the mapping to the XMANAI data model for incoming data.

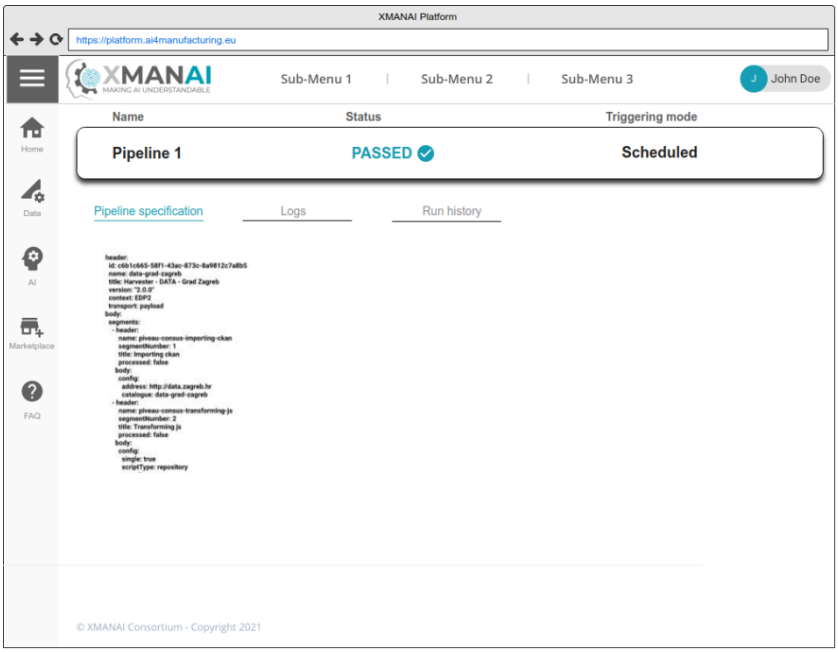


Figure 3-5 Data harvesting pipeline configuration

The second tab of the pipeline details presented below shows the harvester execution logs.

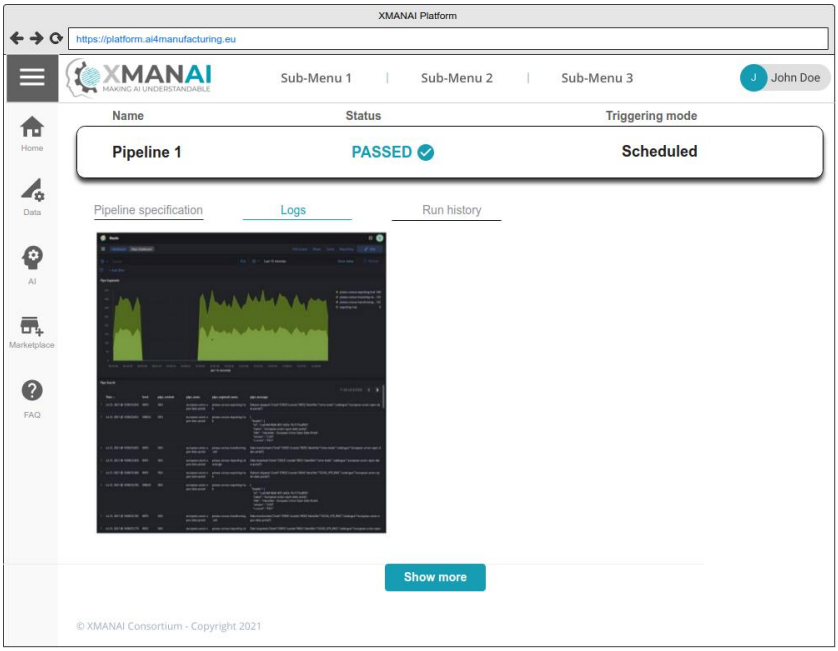


Figure 3-6 Data harvesting logs

The third tab of the pipeline details dialogue shown below presents the execution history of the data collection pipeline. The user can see when the pipeline was executed and what was the execution result.

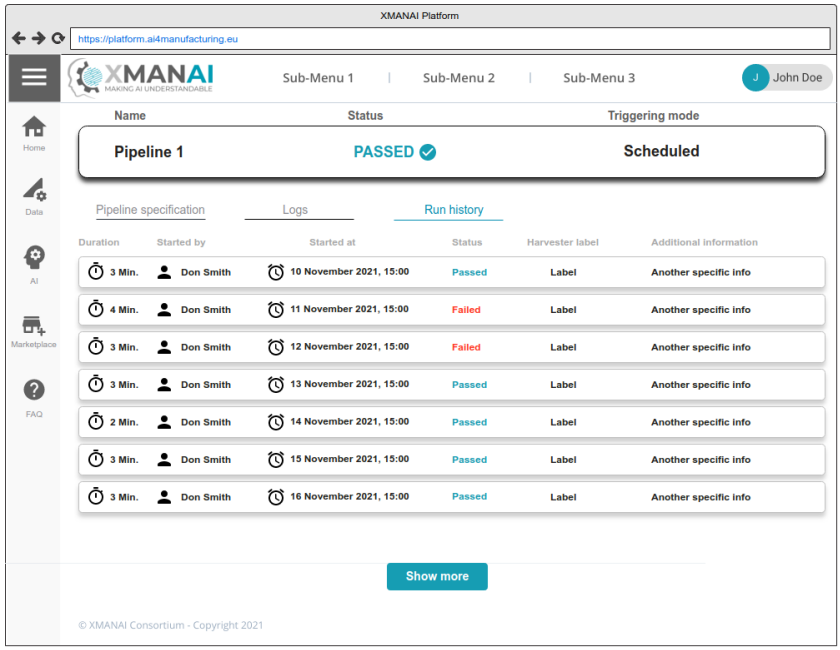


Figure 3-7 Data harvesting pipeline execution history

3.6 Data Handler: File Data Harvester

3.6.1 Overview

The File Data Harvester is a component responsible for transforming data from file(s) to a structured form according to the XMANAI data model and submitting it for storing to the XMANAI Assets Storage. The component has to provide a possibility for user to describe the semantics and structure of the data file(s) and to provide additional metadata. Using this information, the File Data Harvester will initiate the generation of the suitable data structure in the XMANAI Assets Storage and will submit the data to it. The provided description of the data (metadata) will be submitted by the component to the Metadata Store. The result of these procedures is a dataset. It consists of the metadata describing data including the URLs pointing to the data in the storage and the data conformed to the XMANAI data model kept in the XMANAI storage.

The component will provide the following key functionalities to the user:

- **FDH.1: Preview of the data**, so the user could understand the structure of the data and its semantics.
- **FDH.2: Ability to describe the type of the data using the XMANAI data model as a reference**. In the first versions, XMANAI will support only the tabular data or data, which can be represented in the tabular form. Therefore, for describing the data type the user will have to specify the types of data in the columns using the data types from the XMANAI data model.
- **FDH.3: Ability to provide the basic metadata (name, description, categories, etc.) for the dataset**, which should be created on the basis of the data file(s). This functionality will be developed in alignment with the Registry/Metadata Manager.
- **FDH.4: Save the provided metadata in the metadata registry and create a suitable data structure to store data in the datastore** on the basis of the information provided by the user about structure and semantics of the data file(s).
- **FDH.5: Ability to add data** from a file(s) to an existing dataset.

3.6.2 Technology

The component is going to be developed using Java for the back-end and JavaScript Vue.js framework for the front-end.

3.6.3 Mockups

The GUI mockups of the component are presented in the figures below. They present the main interaction dialogues with the user. The first mockup shown at the figure below presents a dialogue for registering a new dataset in XMANAI from file(s) already available in the platform . The user needs to select the files, provide some basic metadata about the dataset and select a data analytics project in XMANAI to which the dataset has to belong.

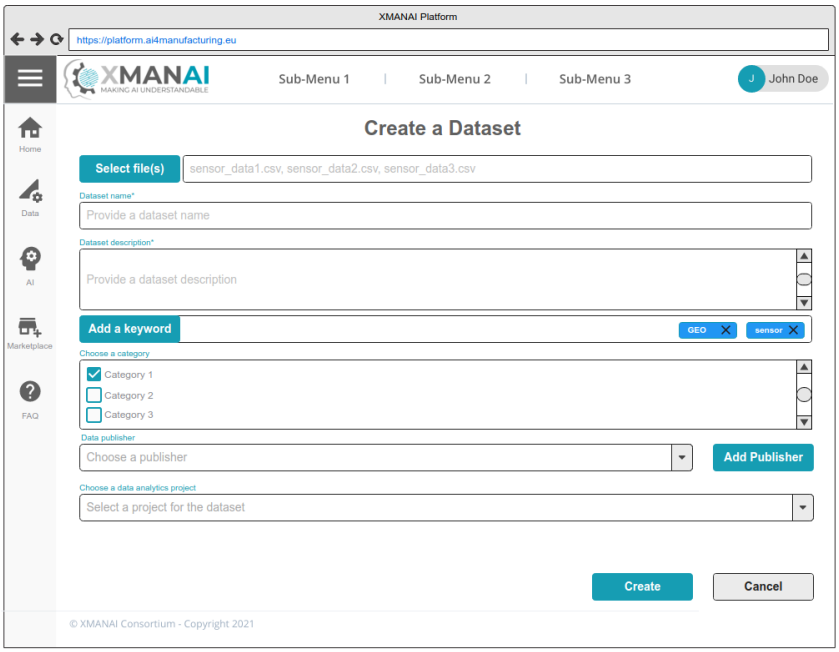


Figure 3-8 Create a dataset from a file(s) dialogue

After providing the basic information about the new dataset the user has to map the data types from the files to the XMANAI data model. The platform provides a preview of the data snapshot and the user has to specify for each column (in case of tabular data) the data type from the XMANAI data model.

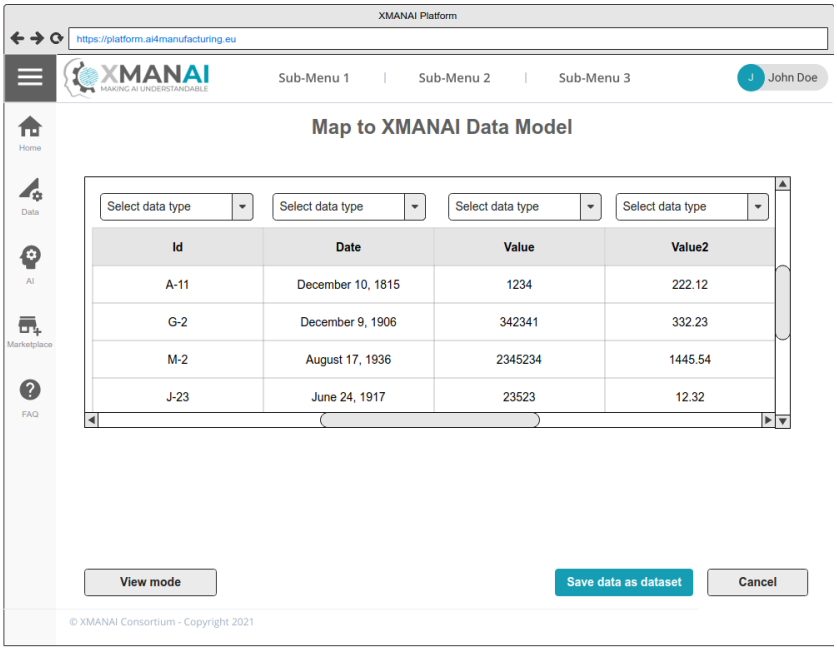


Figure 3-9 Map datatypes from .csv file(s) to XMANAI data model (simple mode)

Alternatively, the user can press the button "View mode" to change to the data types mapping mode presented in the figure below. In this view the data model presented as a graph in which the user can select the data types for describing the data structure of the data files.

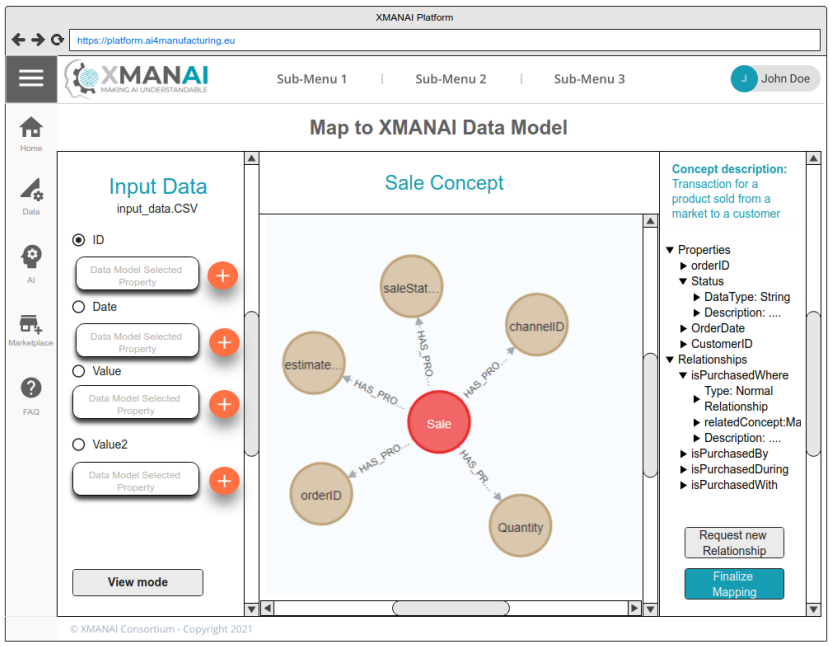


Figure 3-10 Map datatypes from .csv file(s) to XMANAI data model (graph mode)

3.7 Data Handler: File/Data Manager

3.7.1 Overview

The Data Manager is responsible for providing the appropriate interface for the user to interact with the dataset stored at the XMANAI assets store. After the user enters the XMANAI application, they

should be able to view their available projects. The overview will include a friendly interface from which the user may select/edit/create or delete their available projects.

The component will be an intermediate between the user and the XMANAI Assets Store. For fetching information about the available data, the Data Manager will interact via Data Gateway with the Assets Store with Version Control and with the Metadata Store. For identifying the current user and filtering which part of the data store should be accessible to them, the component will interact with the Identity & Authorisation Management component. These are the main inputs for the Data Manager.

The Data Manager will be responsible for collecting the aforementioned information and present it to the user. It will then provide a set of possible actions to the projects that are available to the user. Although we cannot prescribe the list of available actions at a full extent, the following capabilities should be included:

- **FDM.1: Overview of the datasets** that are accessible to the current user
- **FDM.2: Search in the list of the available datasets**. The component will provide the functionality for the user to search among the list of his available datasets.
- **FDM.3: Select an available dataset and edit it**. The user should select a dataset that he has access to it and then edit it. As editing, we describe the act of altering the actual data or the metadata of a dataset.
- **FDM.4: Create a new dataset**. The user will be able to create a new dataset and then insert information that describes it.
- **FDM.5: Delete an available dataset**.

The Data Manager is mainly responsible for receiving actions from the user and then interact with the appropriate component so that the action will take place.

3.7.2 Technology

Considering the XMANAI requirements mainly related to the interaction with the user, the File/Data Manager Component will be principally developed using an appropriate Javascript framework. Among the available options, the ReactJS framework is a suitable solution.

React is a declarative, component-based framework for building interactive user interfaces (UI). It is considered as an optimal choice for creating interactive UIs, while efficiently updating and rendering just the right components when your data changes. Since it is a low-level option for developing a user interface, it provides the appropriate freedom to design all features without restrictions. Apart from that, it is also a rapidly growing framework with an engaged community and many helpful addons.

3.7.3 Mockups

The mockups below provide an overview of the interface the File/Data Manager system will provide to the XMANAI user. In the File/Data Manager the user will see the the list of her/his projects and for the selected project the list of the belonging to the project assets as it is shown in the figure below.

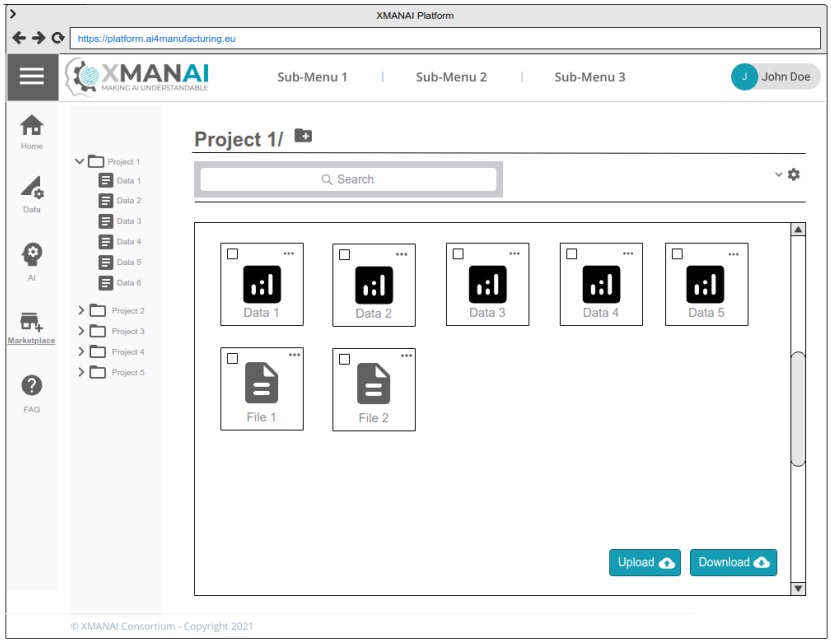


Figure 3-11 Overview of the data available to a specific user

Using the context menu, the user will be able to do some basic operations with the project assests, like moving, coping, renaming, deleting them or adding them to a dataset. The next three mockups provide examples of such interactions.

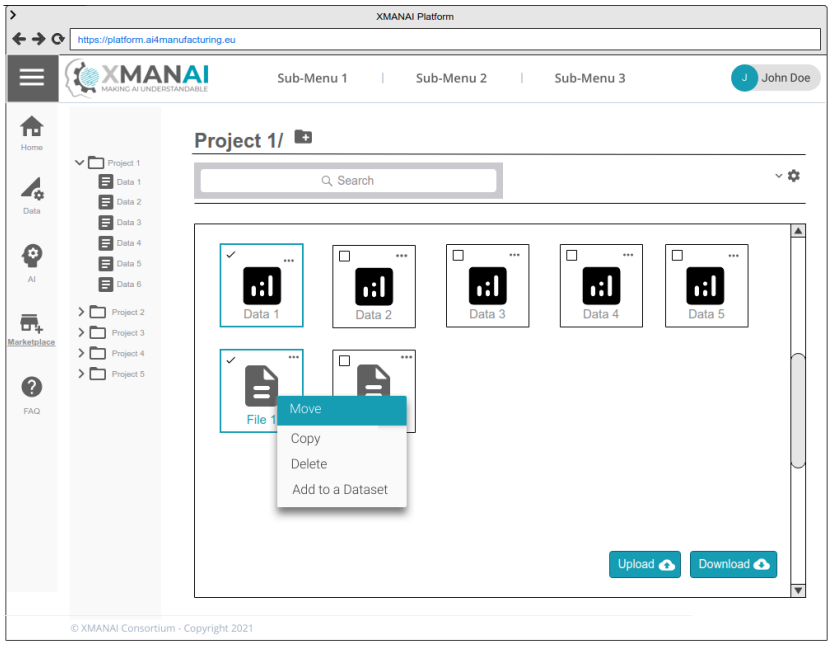


Figure 3-12 The menu of available actions for selected multiple files

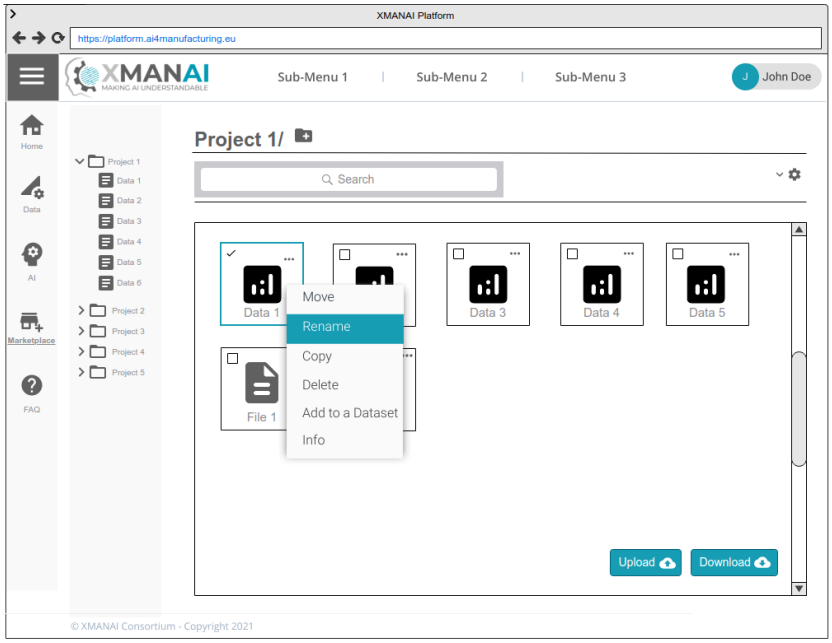


Figure 3-13 The menu of available actions for a selected file

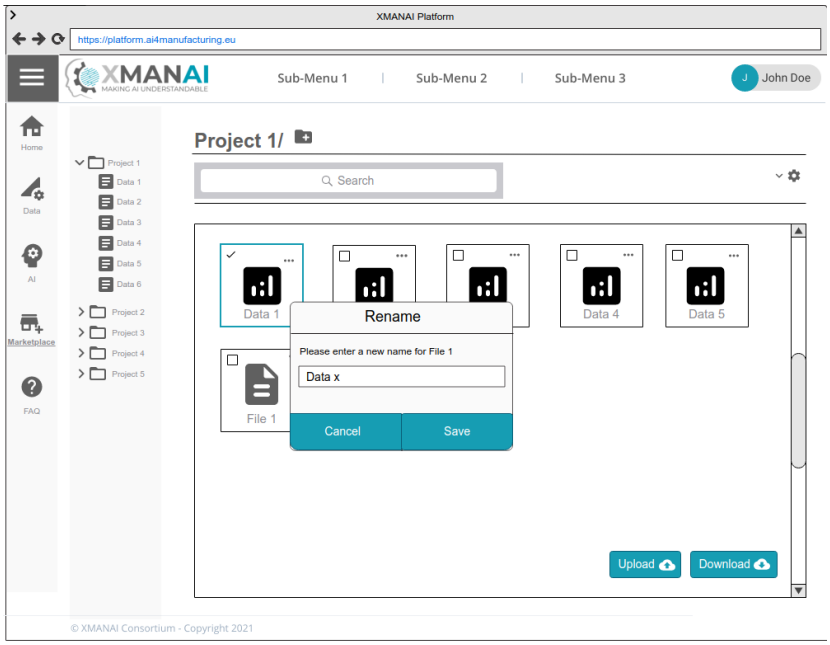


Figure 3-14 The interface of renaming a selected file

3.8 Data Handler: Data Exporter

3.8.1 Overview

The Data Exporter is the component responsible for all the data local saving functionalities. It mainly creates slices of the data that is gathered into the XMANAI platform and saves file instances of them in different accessible formats (e.g. textual, tabular or binary). The data exporter will cover different XMANAI layers of internal data sources, which include datasets.

The component is driven by the user who directly makes an inquiry for a specific type of data (and with specific context parameters), which is translated into a database query to retrieve it. Regardless of the data origin, it is parsed into a common internal format, a dataframe (which sorts the data in a tabular way), to convert it afterwards into a standard file (e.g. such as txt, csv, excel or binary file). The resulting file is then saved locally.

The list of the main actions to be performed using the Data Exporter is the following:

- Select a slice of a dataset (with regard to a specific data range) or the results of a prediction made with a pipeline within XMANAI.
- Select an output file type: csv, json, excel.
- Save the exported file into a local path (including i/o features such as path management, creation of folders and user permissions check).

The Data Explorer is expected to be used by all XMANAI user roles: business users, mainly to retrieve predictions and explanations, as well as data scientists and engineers, mainly to manage slices of the datasets. Thus, the complete list of functionalities of this component will be continuously revised and updated along the project to cover all users needs. Next, we list the main Data Explorer functionalities:

- **DE.1: Select whether to access the dataset of the project or a set of prediction results (along with their explanations) previously generated**. This operation will be performed manually directly from the main page of a project, considering that the user has access rights.
- **DE.2: Generate dataset slices by selecting a specific data range (i.e. initial and end dates) and a concrete version of the data from the version control**. This will generate a query that will obtain as a result an internal mirror of the selected data using a tabular dataframe (i.e. a Python Pandas format). To this end, the Data Exporter will use a parser to extract from the data model or the results database the variables, their values and their timestamps.
- **DE.3: Convert the queried data into a local file**. To that end, the user will manually select the type of output file (csv, json, excel, pickle, hdf5) and through GUI will select the destination route. The internal dataframe will be automatically converted using Pandas methods into the specified file type and saved in the selected path.

The final result of the operations will be the output file (whether in textual or binary format) with the queried data in a structured way, so that it would be ready to be consumed directly by the users or even third-party applications.

The main dependency of the Data Explorer is:

- Assets Store with Version Control: to access the datasets

3.8.2 Technology

The main technologies of the Data Exporter component are the following:

- Python: mainly to parse the data, store it using Pandas, and save it into a file using Pandas internal methods (to convert a dataframe to: excel, csv, hdf, pickle, json)
- Vue.js: To provide all GUI functionalities

3.8.3 Mockups

The GUI mockup for exporting a data snapshot from a dataset is presented in the figure below. The user will be able to select a dataset from a list of datasets to which he has access, filter the start and end dates of the data to be downloaded and select the output format of the data between csv, json and excel.

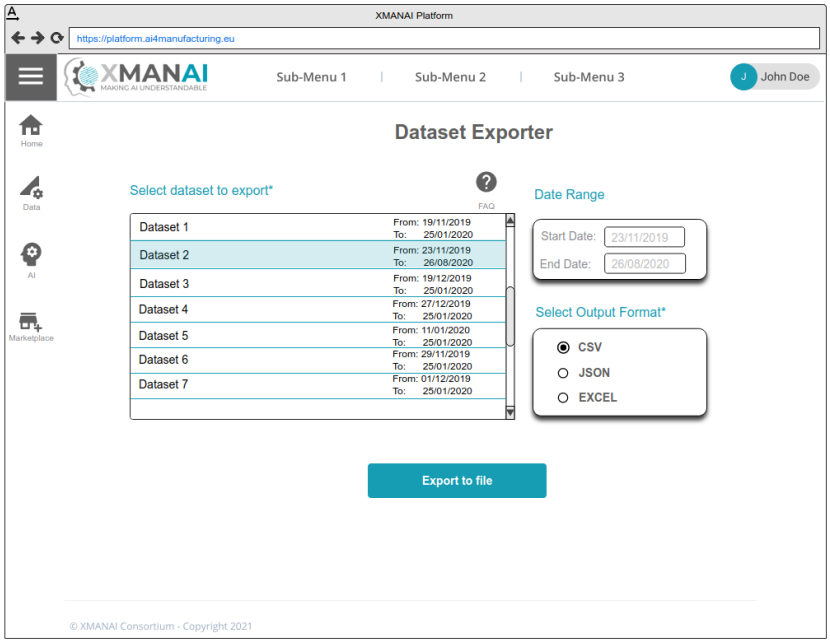


Figure 3-15 Export of data snapshot from a dataset

3.9 XAI Marketplace: Registry/Metadata Manager

3.9.1 Overview

The Registry/Metadata Manager is part of the XAI Marketplace and is responsible for the clear definition, refinement and storage of metadata information that accompanies (and better describes) every dataset and AI model and makes it easier to find, use and manage such AI artifacts. This additional information is provided in collaboration with different components, e.g. the Data Harvester and XAI Pipeline Manager. Most of the metadata are directly provided by the users, but a specific set of a few metadata, like the length (rows) of a dataset, or the size of a model file, can be computed automatically. This set of information to be kept and maintained, follows the XMANAI metadata schema and other metadata standards and is stored in a Metadata Store, which communicates directly with the Metadata Manager.

Different categories of information will be stored based on state-of-the-art metadata standards, such as DCMI and DCAT, and indicatively include:

- Basic asset details, like name, short description, related domain/subdomain, etc.
- Spatial and temporal coverage (as in the city, region, country or time period the data refers to)
- Licensing details, like licence type, IPR owner, etc.

The Metadata Manager provides the appropriate interface for data asset providers to update or extend the relevant information of their assets in order to facilitate their everyday data-relevant activities, spanning from internally managing them to sharing them with other departments within or beyond their organization. To this end, the Registry Manager allows the user to browse the available assets, search for specific asset-related information or filter out items of little interest.

To summarize, the list of functionalities provided by the Registry Manager are the following:

- **RMM.1: Data asset metadata management,** which involves the retrieval, clear definition, enrichment or update of all metadata stored in the Metadata Store.

- **RMM.2: Data asset catalogue browsing,** which provides an interface for the potential data asset consumers to investigate and examine in detail the various datasets and AI assets, as part of a catalogue or as the result of a search query. The data assets presented to the user are automatically filtered based on the respective access policies and IPRs, which are communicated by the Policy Engine and the Provenance Engine. The eligible assets can be further examined separately, exposing information that depends again on the access level of the viewer. For example, if the data asset belongs to the same organisation as the user, the view will be much more informative, whereas an external (to the department/organisation) user will view only basic details.
- **RMM.3: Data asset search and discovery,** which allows a data asset consumer to locate assets of interest or search with very specific properties, such as through a smart text search functionality, filtering and sorting.

3.9.2 Technology

The following technologies are being considered for the development of the Registry/Metadata Manager component:

- S5 Share Platform
- Vue.js[25] will be used to implement the user interface (UI)

3.9.3 Mockups

The GUI mockups of the component are presented in the figures below. They present the main interaction dialogues with the user.

In particular, as depicted in Figure 3-16, the user is able to see the main interface of the Explainable AI Catalogue where he/she may search for data assets for interest in free text and/or providing filters (such as the manufacturing problem of interest, the asset type and the provider type). For each query for data assets, the user gets a lists of results along with a quick overview of each asset (i.e. its title, type, description and provider). The applied filters are also visible in order to allow the user to refine the query whenever needed.

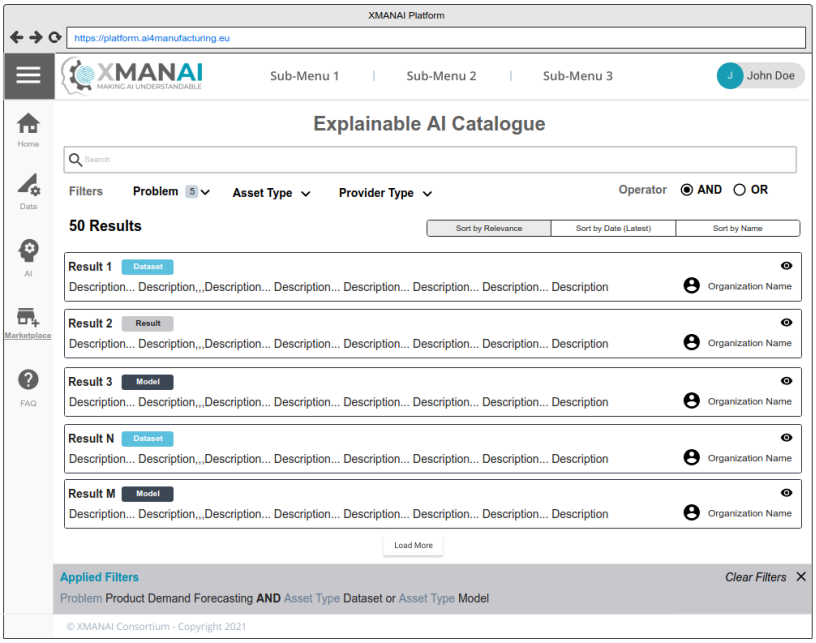


Figure 3-16 XMANAI Marketplace/Catalogue

[25] https://vuejs.org/

If the user needs a more thorough view of the data asset details, he/she may visit the detailed profile of the data asset. In the case of datasets (as depicted in Figure 3-17), such a detailed profile includes the core dataset information (e.g. description, provider, keywords, category, temporal and geographical coverage), the data explainability aspects (with the structure and semantics of the dataset as aligned to the XMANAI data model) and the dataset distribution and licensing details. If the user belongs to the organization that has provided the specific dataset, he/she is able to edit its metadata or export it with the help of the Data Exporter functionalities (described in Section 3.8). If the user is an external to the organization that owns the specific dataset, he/she has the option of acquiring it (with the help of the Contract Manager described in Section 3.10).

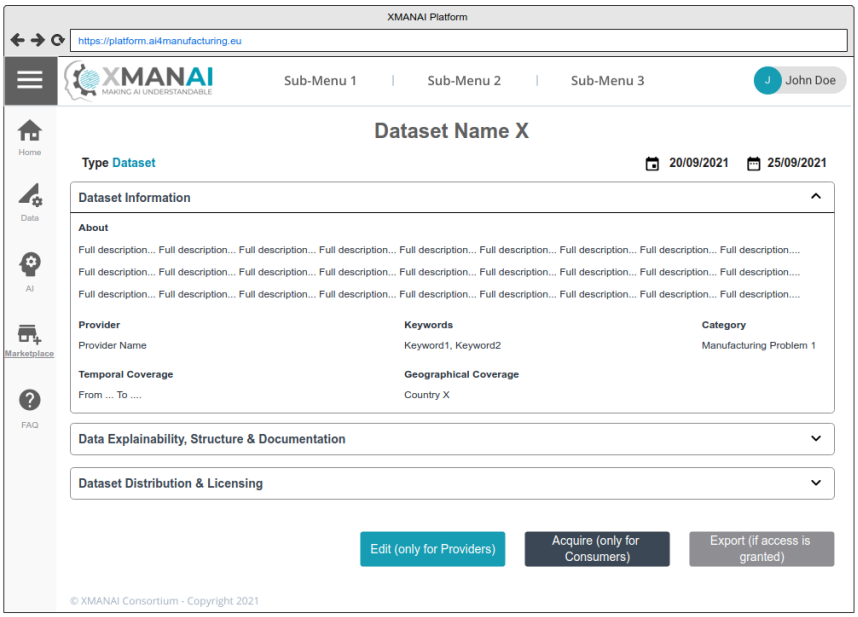


Figure 3-17 Dataset details

In the case of AI models (as depicted in Figure 3-18), such a detailed profile includes the core model information (e.g. description, provider, keywords, category, library, purpose), the model explainability aspects (with the associated XAI techniques that have been used) and the model distribution and licensing details. If the user belongs to the organization that has provided the specific model, he/she is able to edit its metadata. If the user is an external to the organization that owns the specific dataset, he/she has the option of acquiring it (with the help of the Contract Manager described in Section 3.10).

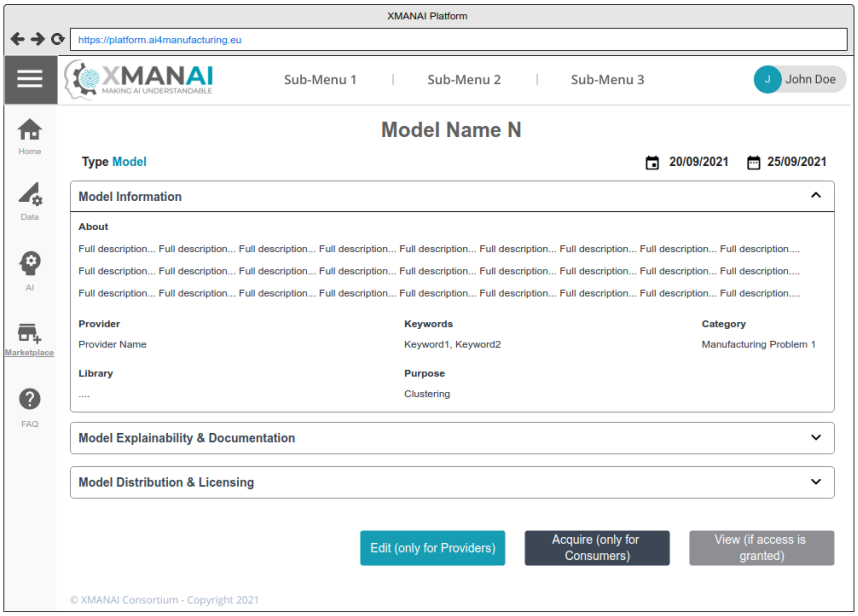


Figure 3-18 Model details

3.10XAI Marketplace: Contract Manager

3.10.1Overview

The Contract Manager can be regarded as the core component of the XAI Marketplace, as it facilitates asset sharing between different organizations, their departments and their users. It is the component that handles all operations regarding smart contracts for sharing data assets, like datasets, trained models or XAI pipelines. These data products can be shared among different business entities, like the departments of the same organisation that work under the same project or between an organisation and an external data scientist. At the same time, the Contract Manager supports asset sharing among organisations, in one-to-one agreements (for simple data assets), as well as in one-to-many (for complex data assets).

In this context, the Contract Manager shall provide the required functionalities for the creation or modification of a contract and for monitoring its current status since several factors can affect it. More specifically, a contract may be set to be expired after a set period of time or be renewed, if all parties are in agreement. Furthermore, a contract may have a number of negotiation rounds or it can be cancelled if the involved parties reach no agreement. All of these activities should occur in a semi-automatic manner, so that the final decision making is performed by the involved humans.

The functionalities supported by the Contract Manager can be summarized as follows:

- **SCM.1: Data asset sharing through smart contracts:** When an acquisition request is made from a data asset consumer, a draft smart contract is automatically produced on behalf of the data asset provider using as a basis terms that derive from the asset's metadata. Of course, the data asset provider is able to modify it and prior to sending it to the data consumer. Then, a phase of negotiation commences that can be resolved outright, or take some time if several changes on the proposed terms are progressively requested by either party. The process continues until all involved parties agree on all terms or either party decides to cancel the contract. In the case of a positive outcome, the involved parties digitally sign the contract and activate it (taking into consideration that payments are not foreseen in XMANAI). In each step

of the process, the Contract Manager communicates with the XMANAI distributed ledger to securely store the smart contract details.

- **SCM.2: Smart contract enforcement:** This functionality ensures that no violation of the contract terms occur at any given moment from the involved parties in the XMANAI Platform. To achieve this, the Contract Manager assesses all actions on the asset(s) involved and protects against any processing or manipulation against the agreed terms. The Contract Manager also communicates with the blockchain in order to verify whether a specific contract is: a) valid and b) active. This is a necessary step when a user attempts to use an acquired dataset, since it must be ensured that an active asset contract is in place and at the same time, access is still granted and has not expired.
- **SCM.3: Smart contract export:** All involved parties should be able to retrieve (i.e. download) the agreed terms and conditions as a file for archiving purposes but also to better study the contract's content and receive approval from their legal department.

3.10.2 Technology

The Contract Manager will be based on the S5 Share Platform and will heavily rely on state-of-the-art technologies related to smart contracts and blockchain. More specifically, it will utilize the Ethereum distributed platform for the blockchain layer, along with its built-in smart contract functionalities, as described in section 2.2.10. For the back-end layer, the web framework that will be used is the Nest (NodeJS) and for the front-end layer the VueJS framework.

3.10.3 Mockups

The GUI mockups of the component are presented in the figures below. They present the main interaction dialogues with the user.

In particular, as depicted in Figure 3-19, the users are able to view a list of the contracts in which their organization is involved along with the contract status (e.g. under negotiation, effective, rejected), and the available actions they need to perform (depending on the status). The may also filter the contracts by status or asset type and sort the contracts by date or name.

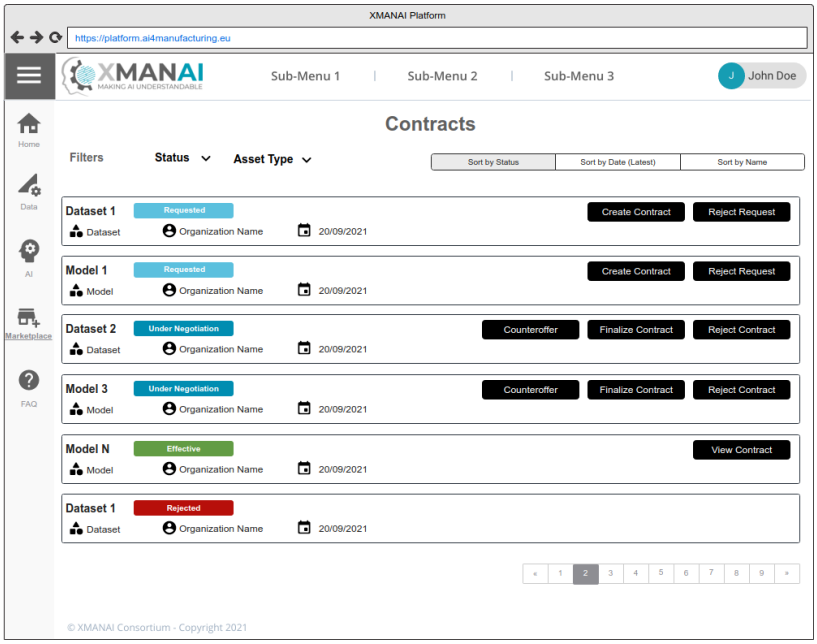


Figure 3-19 Overview of contracts

When data asset providers have received an expression of interest by a data asset consumer and have decided to share the involved data asset, they need to create the respective contract in which they

need to set the terms of the contract (in text form) and the duration of the contract, and indicate whether use outside the XMANAI framework and derivation are allowed. They may save as draft the contract and come back to it later or proceed to signing it (which will result into the contract being written in the blockchain).

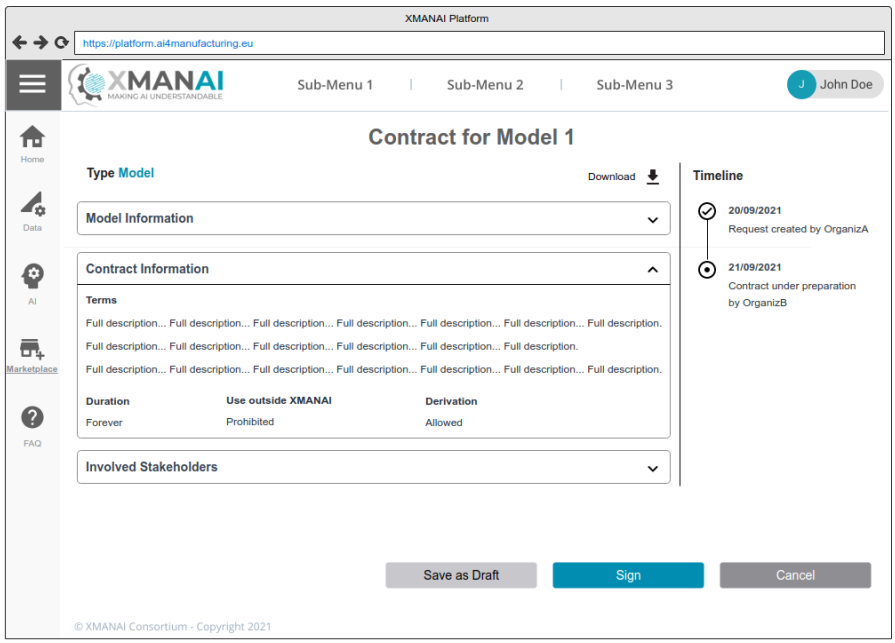


Figure 3-20 Create a contract

When the data asset consumers have received a draft contract, they may review its contents and decide whether to accept it, proceed to a counteroffer or reject it, as depicted in Figure 3-21.

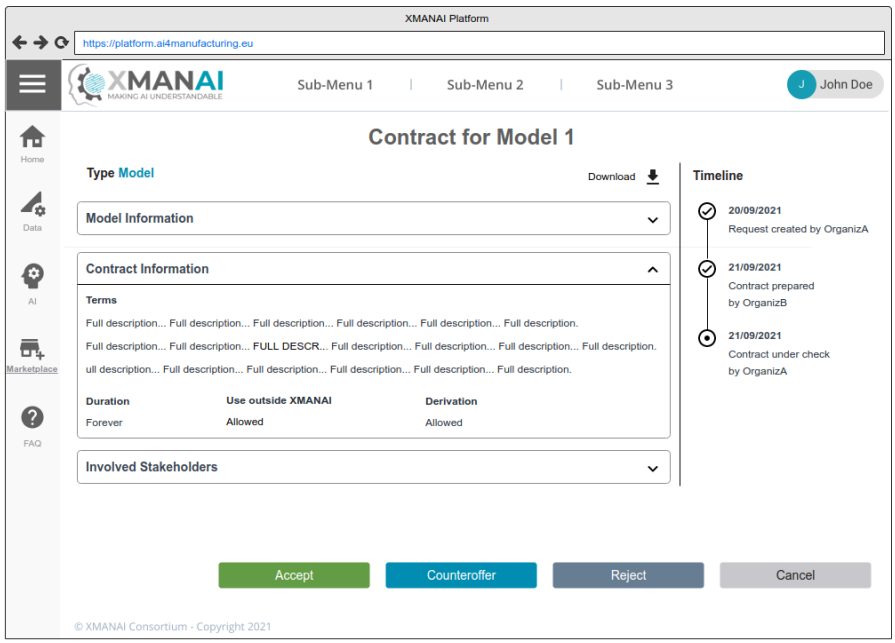


Figure 3-21 Negotiate a contract

If the data asset consumers have provided a counteroffer, the data asset providers may track the exact changes that have been introduced and the timeline of activities. They may decide to accept, counteroffer or reject the updated contract as depicted in Figure 3-22. The process practically ends when either party has accepted or rejected the terms which result into an effective contract or rejected contract, respectively.

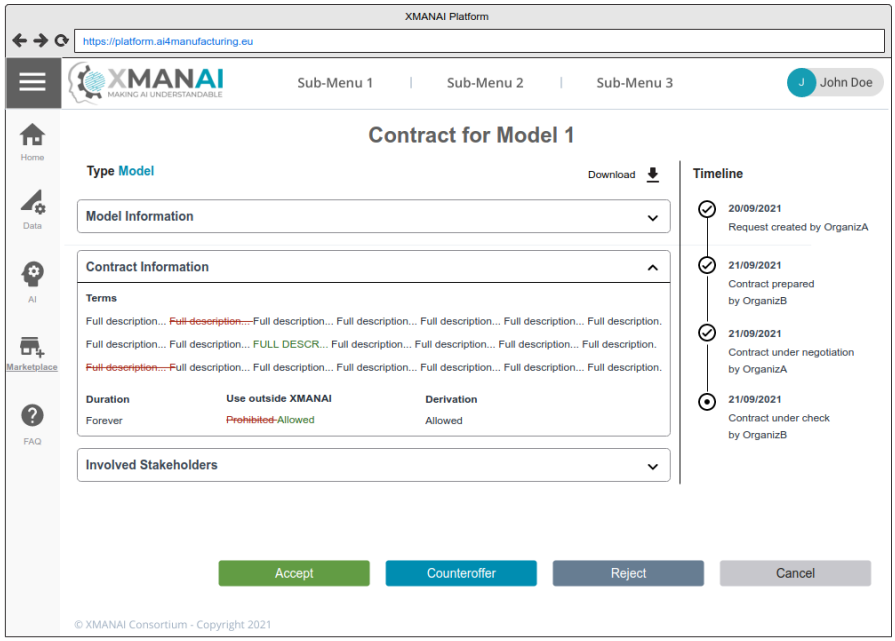


Figure 3-22 Finalize a contract

3.11 Provenance Engine

3.11.1 Overview

The provenance engine enables the determination of the complete lineage back to the creation of the data. In addition, the transformation or integration of original data into processed data or new datasets should be tracked to create transparency with respect to data usage, data manipulation, underlying manipulation methods and access privilege. This transparency allows particular entities to comprehend the provenance of given data.

Two main functionalities need to be provided by the provenance engine:

- **PRE.1: Tracing of data events** that can emerge when data is created, used or modified. This is done by generating metadata about respective activities and storing this metadata securely. Every data creation and modification entails a specific version in the lineage of a particular dataset.
- **PRE.2: Connecting the metadata of particular data versions and the respective lineage with database states**. This can be done by a version control component for the existing asset store and the integration of descriptive metadata about the database version in the metadata storage.

State-of-the-art technologies that can be part of the provenance engine are presented in 2.2.12.

3.11.2 Technology

The Provenance Engine handles all metadata about data activities and the performing entity. According to this, information about these data activities are submitted by the Data Handler: Data Gateway Component to the Provenance Engine. In addition, the information about the version of a respective dataset is generated by the version control component of the Data Storage Services: Assets Store with Version Control and processed by the Data Handler: Data Gateway Component. Such version information are attached to the submitted information about the data activity. Assuming that entities need to be authorized by the Identity & Authorisation Management to perform a data activity, the information about the performing entity is also attached to the submitted information about the

activity. The informational flow between the Provenance Engine and related components is illustrated in Figure 3-23.

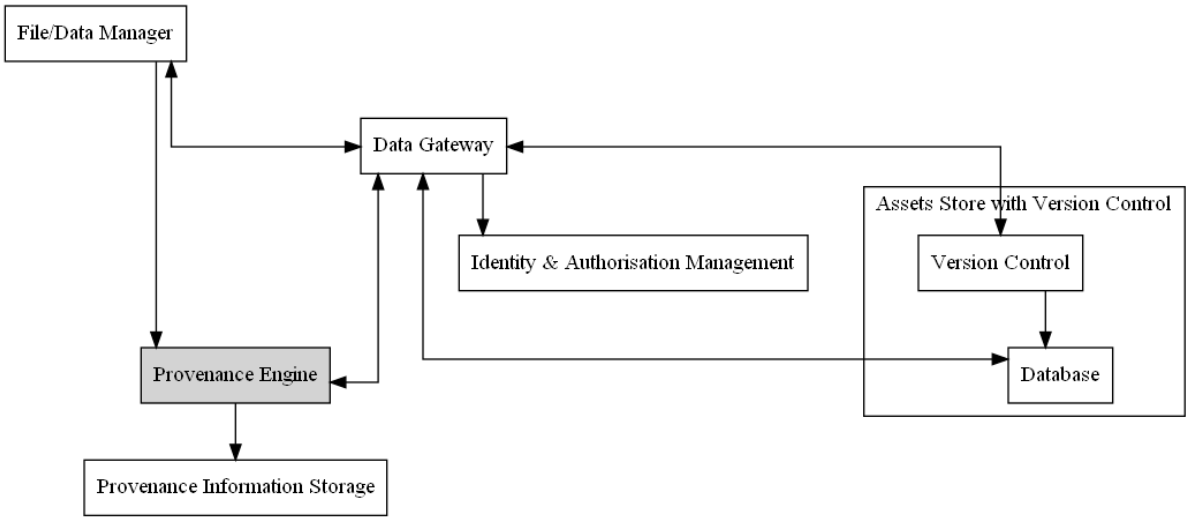


Figure 3-23 Informational flow between the Provenance Engine and related components

The underlying metadata flow in Figure 3-24 visualizes the sources of metadata on the XMANAI platform. With reference to the informational flow illustration in Figure 3-23, the meta is not send directly to the Provenance Engine, but handled by the Data Gateway.

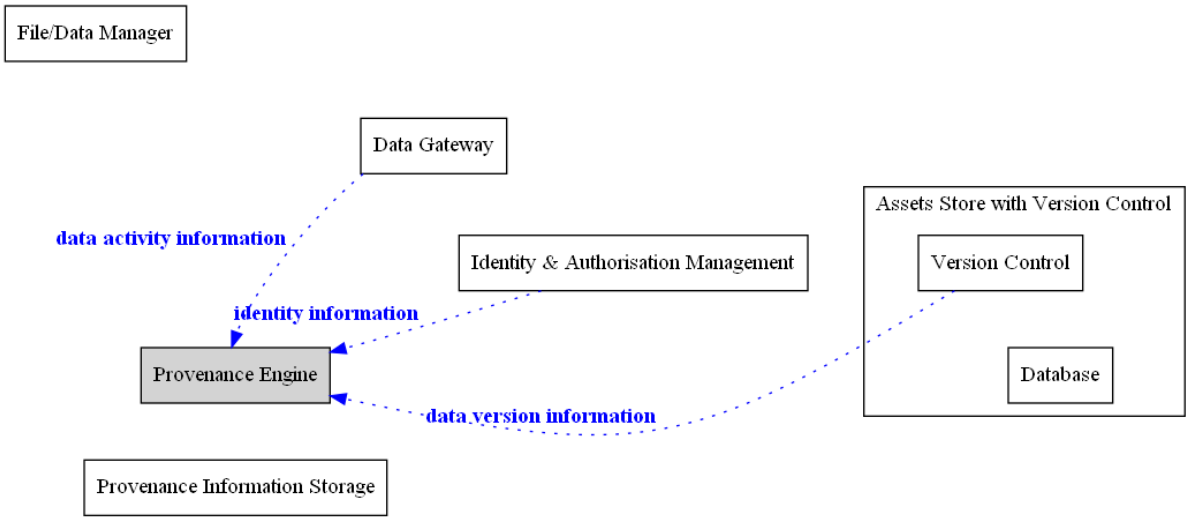


Figure 3-24 Metadata flow between metadata sources and the Provenance Enginets

The obtained metadata about a particular data activity is stored as a semantic triple in the Data Storage Services: Provenance Information Storage . The semantic triples for a respective resource are stored in form of a directed graph. A resource in the triplestore corresponds to a dataset. It is quite important that deletion of nodes or triples in the triplestore is not permitted to provide a complete log of all activities. As a consequence every data activity is traced, including transformation and usage of data. A new resource that is based on a transformation or usage of existing data is connected to the original dataset in the triplestore to provide the complete lineage and transparency. The metadata format corresponds to the W3C PROV Recommendation presented in 2.2.12.

3.11.3Mockups

This component does not provide a graphical user interface. The provenance information is displayed to user as part of metadata in the detailed data asset profile pages provided in the Registry/Metadata Manager in Section 3.9.

3.12 Access Manager: Policy Engine

3.12.1 Overview

The Policy Engine provides the robust and solid access control mechanism that facilitates the dynamic and effective regulation of the access to the various assets of the XMANAI platform. The main objective of the Policy Engine is to formulate the appropriate access control decision for granting or denying the access to any asset by evaluating each request against the defined by the owner of the assets access rules called access policies. Hence, the Policy Engine is composed of two core parts, the definition of the access control model that defines under which conditions the various assets can be accessed and by whom and the evaluation of each access request based on this model.

As described in section 2.1.2.4, multiple logical access control models exist that enable the realisation of access control mechanisms where each model is defined with different characteristics and features depending on the needs of the assets for which access should be properly regulated. Within the context of XMANAI, a hybrid approach will be followed by designing an access control model that takes the best of the most dominant models such as ABAC, RBAC and ACL. The rationale behind this approach is based on the nature of the different assets in XMANAI as any valuable resource incorporated in the platform such as projects, datasets, trained models and analytics results are considered as assets. To this end, the model will be based on access policies that depend on the nature of the protected asset as provided by the Policy Editor and the policy enforcement mechanism takes into consideration the formulated access control model to formulate an access control decision.

In detail, the Policy Engine takes as input a request to access a protected asset by a user of the platform or a service of the platform through its provided API endpoints, evaluates it taking into consideration the provided access policies as well as the attributes of the requested asset and the requestor and finally formulates the appropriate access control decision. It should be noted that the Policy Engine is responsible for the formulation of the access control model based on the access policies that are received by the Policy Editor and for the enforcement of these policies on each access request while the access policy lifecycle management is performed by the Policy Editor. Hence, there is clear separation of concerns between the policy enforcement and the policy definition, however the Policy Engine and the Policy Editor are tightly connected.

The main features of the Policy Engine are as follows:

- **PEN.1: Formulate the access control model** based on the provided by the Police Editor access policies
- **PEN.2: Offer the instant deployment, update and enforcement of the access policies** as provided by the Policy Editor
- **PEN.3: Effectively handle the combination and enforcement of multiple policies for an asset based on logical reasoning**
- **PEN.4: Regulate the access to any asset of XMANAI** by evaluating all access requests against the defined access policies and yield an access control decision that will either grant or deny the access to the requested asset
- **PEN.5: Provide the API interfaces** that will receive and evaluate all access requests

3.12.2 Technology

Based on the state of the art analysis performed on sections 2.1.2 and 2.2.9, the dominant Casbin policy enforcement framework will be exploited in the implementation of the described functionalities of the Policy Engine. Casbin provides the functionalities required for the implementation of the various access control models such as ACL, RBAC and ABAC which can be combined effectively to formulate a robust access control mechanism. Besides the access control model definition and access control mechanism, Casbin provides the basis for the access policies definition based on the selected access control model when it dictates the input that will be received for the access control model definition. Casbin supports multiple programming languages by offering different flavors of the framework. Within the context of XMANAI, the Java-based library of Casbin, namely jCasbin, will be leveraged for the implementation of the Policy Engine. In addition to Casbin, Java 11 and the powerful and well-established Spring Boot framework will be exploited.

3.12.3 Mockups

The Policy Engine constitutes a purely backend component interacting with the rest of the components of the platform via well-defined REST interfaces, hence no user-interface is foreseen for this component.

3.13 Access Manager: Policy Editor

3.13.1 Overview

Within the context of XMANAI, access to any asset of the platform such as projects, datasets, trained models, analytics results will be regulated via authorisation rules which are defined in the form of access policies. In particular, access policies will define the exact and strict conditions under which logical access will be provided for these assets to any possible requestor based on the preferences of the owner of each asset. In this sense, the legitimate owners of the assets will be able to define access policies for their assets which will be translated into authorisation rules that will be applied in order to formulate the permission granting or denial to any access request. Moreover, the owner of the assets will be able to define and impose multiple access policies over each specific asset from whose combination the desired level of security and trust can be achieved in a fine-grained manner. In addition to this, the approach of flexible access policies will enable the owners of the assets to secure and protect their assets in a way that eliminates the need for any prior knowledge of the potential consumers of their assets.

The Policy Editor offers an intuitive and user-friendly way to define and manage the access policies in a flexible and solid manner through its novel user-interface. In particular, the Policy Editor undertakes the responsibility of implementing the complete access policy lifecycle management, spanning from the creation and storage of an access policy to the reuse, update and deletion of the access policy. Hence, the Policy Editor on the one hand offers the user-interface that is leveraged by the owner of the asset to perform the access policy lifecycle management operations and on the other hand stores the results of these operations in order to be fed to the Policy Engine. As the access policies are the main pillar of the access control model that is realised by Policy Engine, which provides the access control mechanism of XMANAI, it is obvious that the Policy Editor is tightly connected to the Policy Engine. In particular, the access policies are defined following the rules and restrictions imposed by the access control model's implementation in order to be fed and be incorporated to the access control model of the Policy Engine.

The main features of the Policy Editor are as follows:

- **PED.1: Enable the flexible definition, storage, update and deletion of access policies** for an asset of the XMANAI platform.

- **PED.2: Facilitate the reuse of previous defined access policies** on a different asset
- **PED.3: Enable the combination of the multiple access policies** using Boolean logic for a specific asset in order to effectively cover the security aspects of the asset
- **PED.4: Provide the easy-to-use and novel user-interface** from which all the access policy lifecycle management operations are performed by the owner of the asset.
- **PED.5: Enable the loading of the access policies into the Policy Engine**
- **PED.6: Ensure the immediate propagation of newly created or updated access policies** to the Policy Engine

3.13.2 Technology

Based on the state-of-the-art analysis performed on section 2.1.2 and 2.2.9, the described functionalities of the Policy Editor will be realised by leveraging the well-established access policy enforcement framework Casbin, as in the case of the Policy Engine. Casbin offers the framework that supports the easy and flexible access policy definition functionalities which will be exploited for the operations of the Policy Editor for the complete access policy lifecycle management. As the Policy Editor will be tightly connected to the Policy Engine providing the required input for the access control model definition, the same Java-based library of Casbin, namely jCasbin, will be leveraged for the implementation of the Policy Editor. In addition to Casbin, Java 11 and the powerful and well-established Spring Boot framework will be also exploited.

3.13.3 Mockups

The main user interface of the Policy Editor enables the definition and management of the access policies of an asset in an easy and flexible manner. To this end, the user upon the selection of the access policy management option of an asset is navigated to policy editing user interface. At first, the user is presented with the option to select the high-level access policy that applies a preconfigured set of conditions. By selecting the “Public Access” or the “Confidential Access” high-level access policies, the access policies are by default disabled as in the first case unrestricted access is applied while in the second case access is only granted to the members of the organisation that owns the asset. On the other hand, when the “Restrictive Access” is selected, the access is regulated by the defined access policies. In detail, the user is presented with the option to define a policy or to combine multiple policies using Boolean logic. Moreover, the user is able to view, modify and delete the existing access policies.

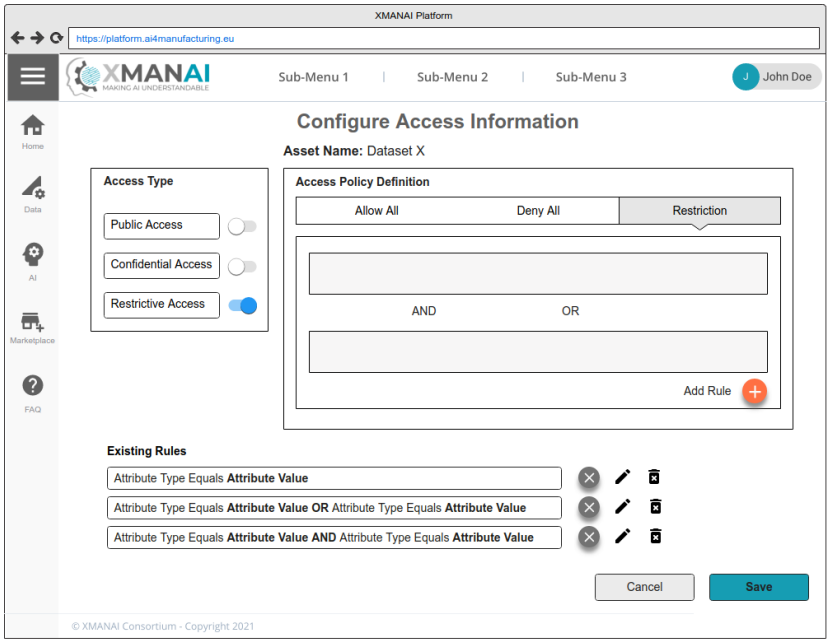


Figure 3-25 Policy editing user interface

3.14Identity & Authorisation Management

3.14.1Overview

Identity and Authorisation Manager is the component responsible for providing the holistic user account management lifecycle of the XMANAI platform as well as the authorised communication between the various components of the XMANAI platform. Hence, the Identity and Authorisation Manager is providing the single core identity provider of the platform by undertaking the operations related to the registration, verification and authentication of all the users of the platform. It classifies the users of the platform under the concept of the organisations where each user belongs to a single and unique organisation following a robust organisation registration process as well as a user invitation process.

In particular, the top-level user of the organisation initiates the organisation registration request to the XMANAI platform. Upon the approval of this request by the administrator of the platform, the top-level user of the organisation is able to invite the users of the organisation to complete their registration in the platform. Hence, the management of the users of each organisation remains in control of the top-level user of each organisation. The invited users should complete their registration to the platform before they are able to access all the platform's offerings. All users which belong to the same organisation have the same privileges on the organisation's assets while the top-level user has more escalated privileges that include the editing of the organisation's information, the invitation of the new users and the signing of smart contracts for assets purchases in the marketplace. To access the offerings of the platform, all users should successfully login and the authentication of the provided credentials is also performed by the Identity and Authorisation Manager. For the purposes of the XMANAI platform, the isolated (silo) identity management model is adopted as it is effectively covering the requirements elicited as the XMANAI platform will be the single service provider.

Besides the identity management operations, the Identity and Authorisation Manager is providing the required authorisation mechanism that regulates the intercommunication of the various layers or components of the platform. In particular, in collaboration with the Policy Engine, it ensures that the access to the set of resources of any component of the platform, such as API endpoints and exposed services, is strictly regulated. To this end, the complete intercommunication between the various

components of the platform is properly safeguarded and controlled towards the increase of integrity and trust of the exchanged data within the platform. The Identity and Authorisation Manager acts as the mediator between the various components and applies the configured authorisation permissions that allow or deny the intercommunication between the various components.

The main features of the Identity and Authorisation Manager are as follows:

- **IAM.01: Provide the complete user management lifecycle by implementing the solid organization registration**. It provides the mechanism for the registration and verification of an organization, as well as the required operations for the maintenance of the organisations' information and profile.
- **IAM.02: Enable the invitation and registration of users under a single organization**. It facilitates the invitation of new users under an organization in a regulated and organised manner. In addition to this, it offers all the required operations for the management of the organisation's users, such as their update, modification, suspension and deletion.
- **IAM.03: Provide the authentication mechanism that verifies and controls the access to the platform's services and offerings**. It offers the respective robust login mechanism via its dedicated interfaces that verifies the provided credentials before access is gained to the platform.
- **IAM.04: Regulate the intercommunication of the various components of the platform** by performing the required authorization control at a service level or an endpoint level of each component. It oversees the internal communication of the components following a set of strict authorization rules.

3.14.2 Technology

Based on the state-of-the-art analysis performed in sections 2.13 and 2.2.7 and the requirements addressed by the Identity and Authorisation Manager, the well-known identity and access management framework Keycloak will be leveraged. Keycloak provides the effective and efficient functionalities required for the realisation of the Identity and Authorisation Manager in an effortless and straightforward manner, while also offering advanced functionalities that can be leveraged to create the robust identity management, authentication and access management operations of the Identity and Authorisation Manager. In addition to Keycloak, Java 11 and the powerful and well-established Spring Boot framework will be also exploited to complement the designed solution's implementation.

3.14.3 Mockups

The user interface of the Identity and Authorisation Manager facilitates all the user management operations of the platform. As described in section 3.15.1, XMANAI follows an organisation-based user management. Hence, the top-level user of the organisation registers the organisation and upon the administrator's approval, the organisation's registration is completed as depicted in Figure 3-26.

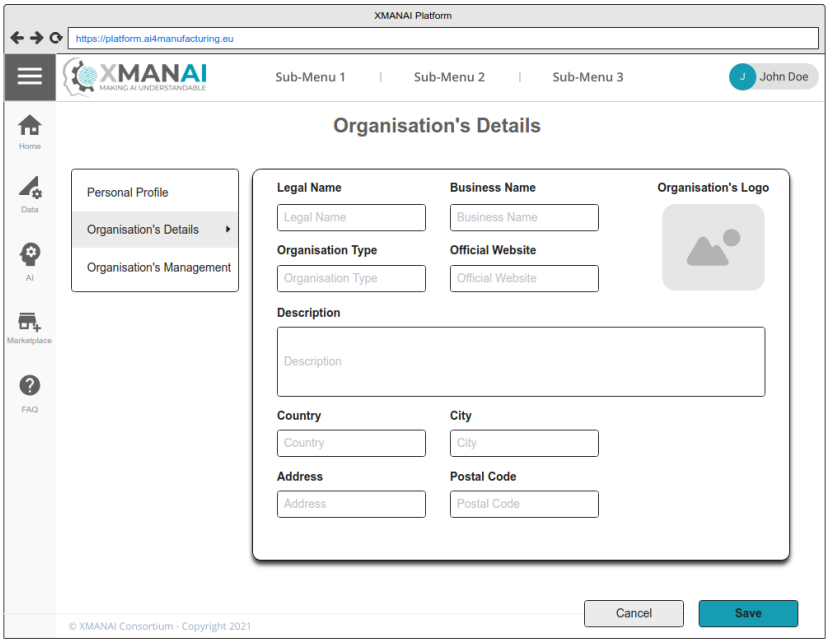


Figure 3-26 Organisation's details page

Following the successful organisation registration, the top-level user invites the users of the organisation in the platform as depicted in Figure 3-27.

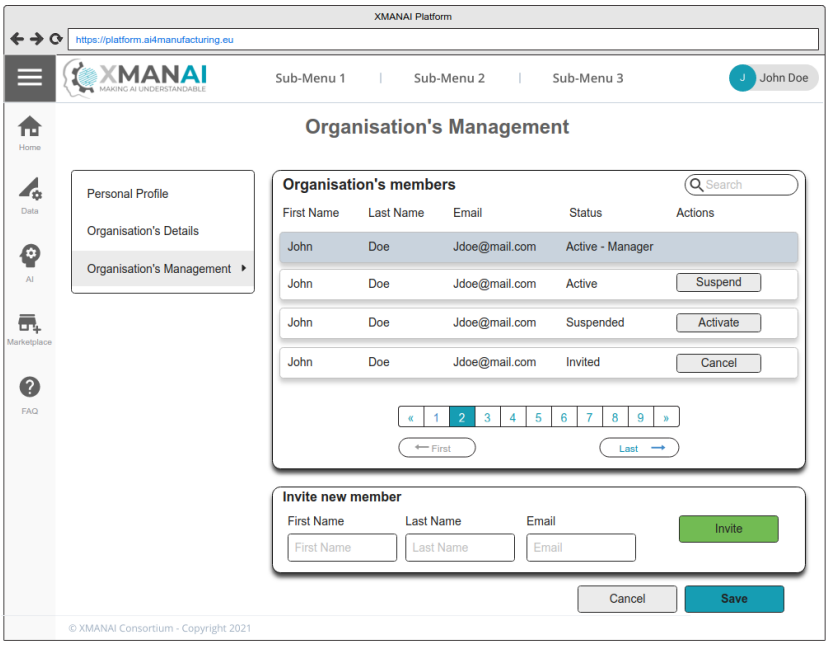


Figure 3-27 Organisation's management page

Once the users complete their registration, they are able to exploit the platform's offerings by successfully logging to the platform as shown in Figure 3-28.

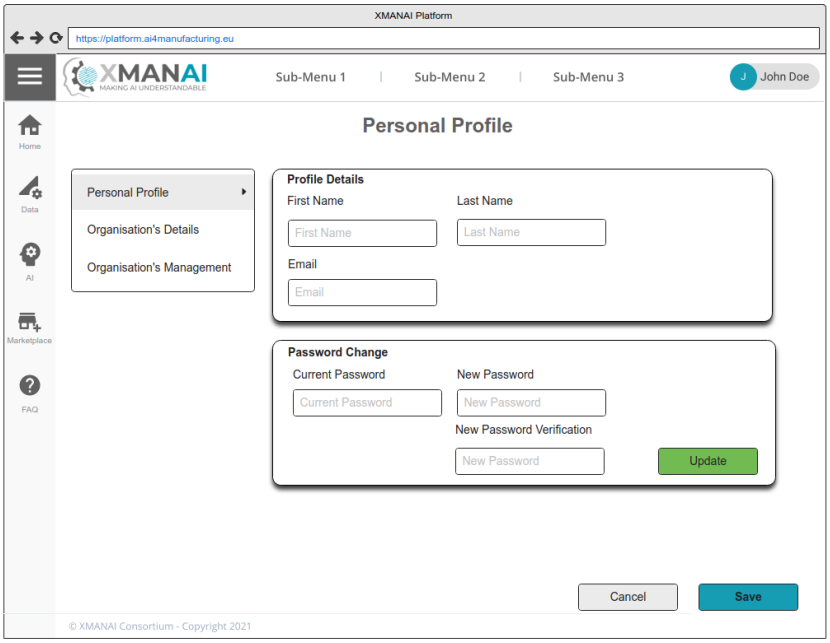


Figure 3-28 User's profile page

3.15 Anonymiser

3.15.1 Overview

Data privacy has become a significant issue in many data applications. When datasets are released to other parties for data analysis, privacy-preserving technologies are required to reduce the possibility of identifying sensitive information. The safest and most trivial solution for protecting sensitive information is to avoid any disclosure, thus losing all value in the data. The simplest solution for protecting privacy is to pseudo-anonymize the dataset by removing all direct identifiers that link a record to a person and replacing them with an arbitrary id. Although this solution reduces the risk, it does not eliminate it. Each record can be re-attributed to a person using either the arbitrary id or secondary information, termed quasi-identifiers, that still exist in the data. For example, a person might be identified by their residence zip code and their date of birth. For this reason, GDPR treats pseudo-anonymized data as personal data, and applies all common restrictions to them. Alternatively, data owners can anonymize the data, i.e., they can apply an irreversible transformation that provides a statistical guaranty that no person can be re-identified from the anonymized data. Such anonymous data is no longer treated as personal data by the GDPR; thus, its restrictions no longer hold. Data anonymization is achieved through various methods that provide statistical privacy guarantees like k-anonymity, l-diversity, differential privacy, etc. Each technique offers a different trade-off between the strength of the privacy guaranty and the quality of the anonymized data.

Data privacy is a critical requirement that should be considered in all flows described in WP2, where services responsible for the sharing, storage, and management of the data and algorithms to be used over the XMANAI infrastructure will be set up. We identify three key issues that should be taken into account in selecting the appropriate privacy-preserving technology: (a) be able to statistically guarantee that modified (anonymized) data cannot be linked to the original data, (b) reduce the need for data owners' consent by rendering the anonymous data non-personal in GDPR terms, and (c) provide user-friendly functionality and be flexible to be applied in several data usage scenarios from XMANAI pilot cases.

3.15.2 Technology

AMNESIA[26] is an open-source anonymization tool designed and developed by Athena Research Center that offers anonymization with statistical guaranties. Data anonymized with Amnesia fall outside the scope of GDPR since they are no longer considered personal. Amnesia removes direct identifiers like names, PIDs, etc., and transforms secondary (the so-called quasi-) identifiers so that the resulting data can no longer be attributed to one person. The data transformation provided by Amnesia is irreversible.

AMNESIA satisfies all requirements discussed in the previous section. It offers a range of anonymization methods, including k-anonymity and km-anonymity, and supports the user by providing automation for DICOM files, vast input datasets, etc. Furthermore, the fact that it goes beyond pseudo-anonymization helps by-passing the need for data owner's consent[27]. Finally, it provides anonymization tailored to user needs through a graphical interface, where users guide the algorithm and decide trade-offs with simple visual choices. Besides the interface, developers can incorporate the AMNESIA anonymization engine into their project through a Rest API.

3.15.3 Mockups

Amnesia is a standalone-component that provide a user interface for data anonymize on its own. It's functionality and user interface are well presented in the following webinar[28]. For showing an overview of its functionalities we present below some figures for anonymizing a dataset in the form of tabular data.

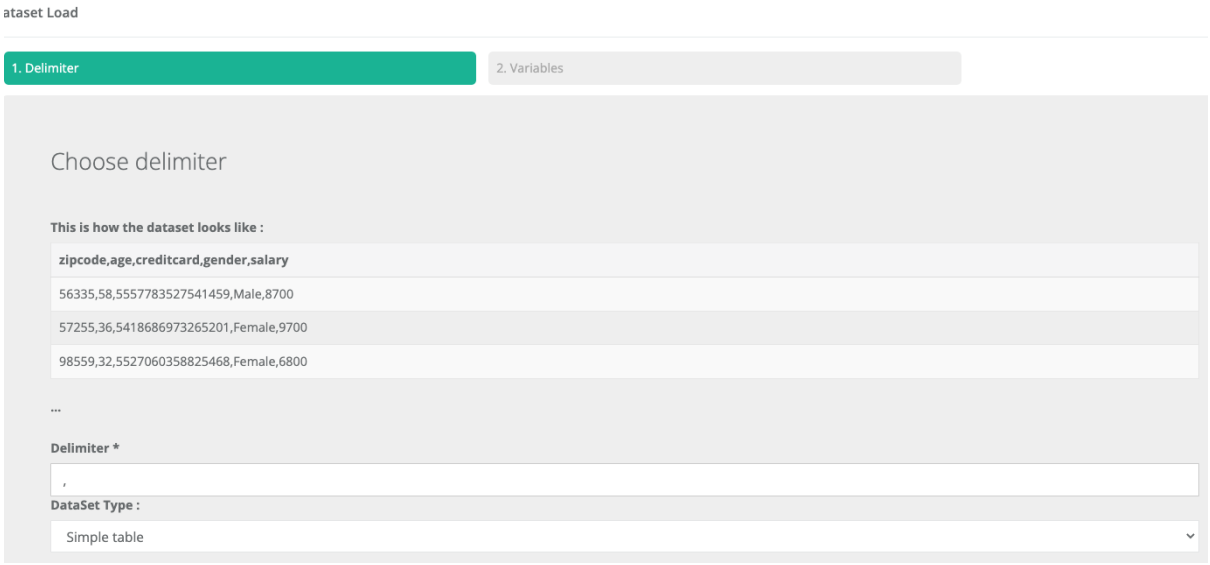
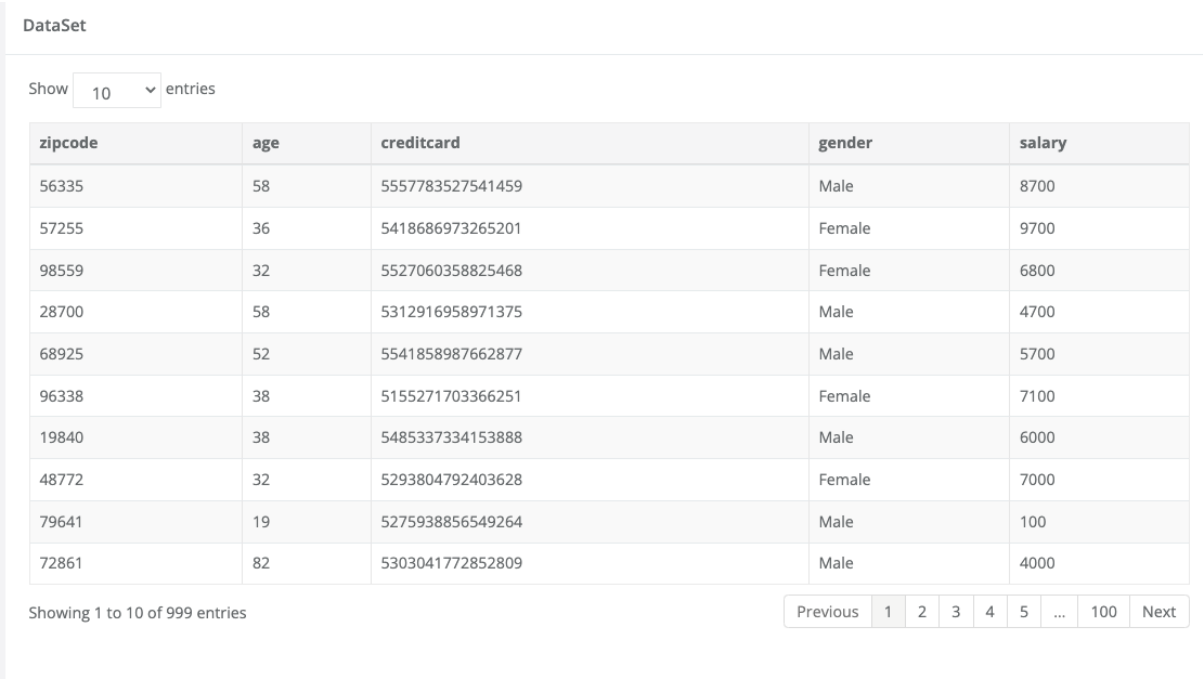


Figure 3-29 Amnesia loading sensitive data

[26] https://www.openaire.eu/item/amnesia-data-anonymization-made-easy
[27] https://gdpr.eu/recital-26-not-applicable-to-anonymous-data/).
[28] https://www.youtube.com/watch?v=_0Io6c1MPOY



DataSet

Show 10 entries

zipcode	age	creditcard	gender	salary
56335	58	5557783527541459	Male	8700
57255	36	5418686973265201	Female	9700
98559	32	5527060358825468	Female	6800
28700	58	5312916958971375	Male	4700
68925	52	5541858987662877	Male	5700
96338	38	5155271703366251	Female	7100
19840	38	5485337334153888	Male	6000
48772	32	5293804792403628	Female	7000
79641	19	5275938856549264	Male	100
72861	82	5303041772852809	Male	4000

Showing 1 to 10 of 999 entries

Previous 1 2 3 4 5 ... 100 Next

Figure 3-30 Initial Dataset

Figure 3-29 depicts the wizard that guides the user to model the data. First, the user should choose the input dataset type (simple table, sets of values, table with a set-valued attribute, disk-based simple table). Then, Amnesia parses the first lines of the dataset and presents a preview to the user guessing the attribute types as shown in Figure 3-30. The user has to confirme the guessed type and can also choose which columns will appear in the output dataset by using the check box next to each attribute. Figure 3-31 depicts the pseudo-anonymization that can be applied using the masking method. After the data loading process, the user can pseudo-anonymize data attributes by clicking on the "pseudo-anonymization" button next to every string attribute. Then, a pop-up window depicts every character of a random value of the column in a small box. The user is then asked to set the desired special character for the mask (e.g., *,&,^, etc.). and to choose which characters from the sample value will should the mask character hide. Figure 3-32 depicts an example hierarchy used to replace specific values with more general until the desired privacy guarantee is reached. The user can edit hierarchies through the depicted visual interface (by pressing the "Edit" button in the hierarchy panel). Editing includes adding, removing, renaming nodes, and moving them from one place of the tree to another. The user can get a preview of the anonymized dataset for each anonymization solution, as shown in Figure 3-33. Then, saving the anonymized dataset is possible through the results screen or the anonymized dataset screen by clicking "Save To Local" in the upper-right menu.

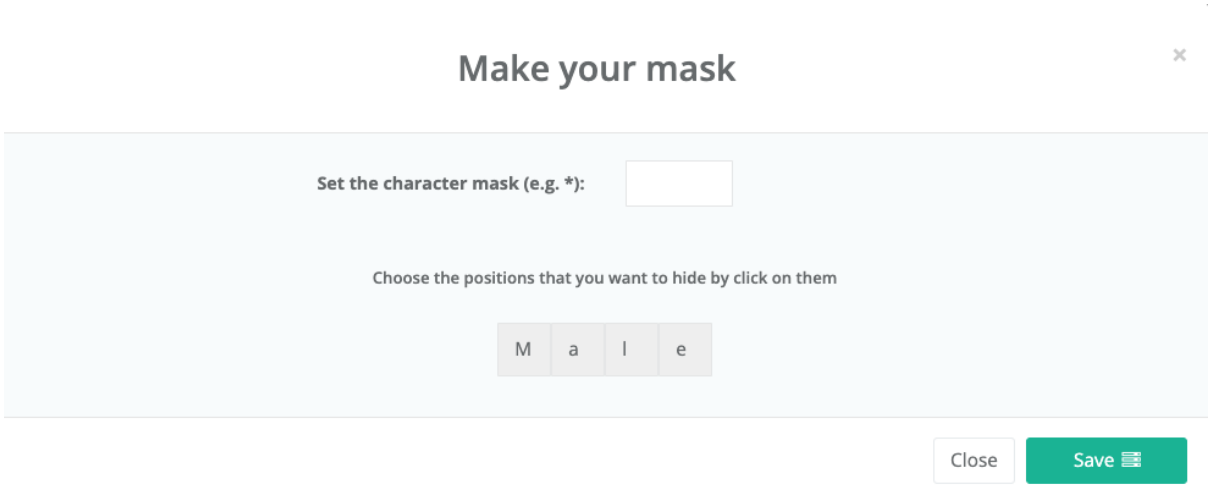


Figure 3-31 Choose the positions you want to hide

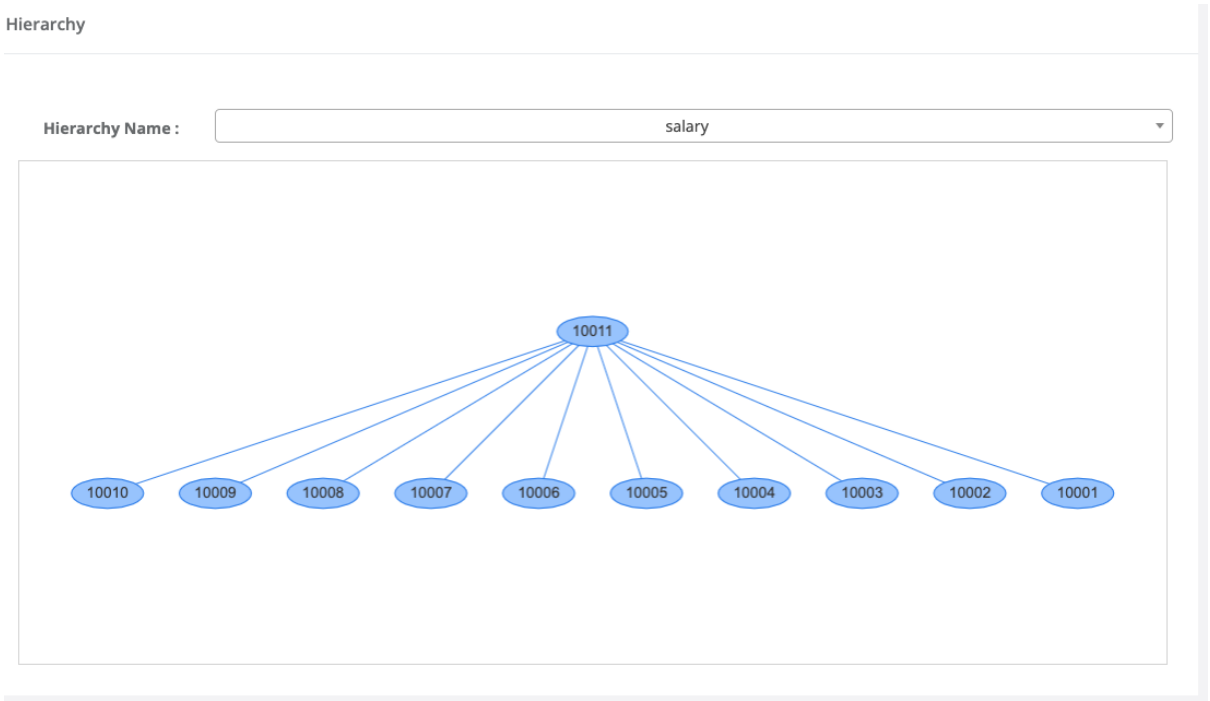


Figure 3-32 Choose the anonymization algorithm.

Anonymized DataSet

zipcode	age	creditcard	gender	salary
56335	58	2147483647	Male	10011
57255	36	2147483647	Female	10011
98559	32	2147483647	Female	10011
28700	58	2147483647	Male	10011
68925	52	2147483647	Male	10011
96338	38	2147483647	Female	10011
19840	38	2147483647	Male	10011
48772	32	2147483647	Female	10011
79641	19	2147483647	Male	10011
72861	82	2147483647	Male	10011

Figure 3-33 The final anonymized dataset

4 Conclusions and Next Steps

Deliverable 2.1 "Asset Management Bundles Methods and System Design" presents an overview of appropriate methods, technologies or solutions and provides the first version of the specifications for asset management in XMANAI including the plans for its implementation and the GUI mockups. The document is further elaborating and refining the overall XMAMAI architecture (D5.1) for the data-oriented services bundles as it is focused on the components responsible for data and other asset collection, management and sharing and on the security components in XMANAI.

The analysis of the relevant methods and technologies is presented in detail in section 2. The evaluation of relevant methods focused on industrial data ingestion, management, security, trust considerations, data quality improvement, asset sharing and provenance. This long list of topics corresponds to the functionalities expected from the assests management and sharing in the XMANAI platform. The list of the reviewed technologies includes only the most prominent examples or representatives of typical categories, which the authors of the document find relevant for the components implementation. The conclusions from the state-of-the-art analysis and research are combined here to create a solid and applicable architecture.

Section 3 presents a functional architecture and specifications for the asset management layer developed based on the outcomes of the previous section and addressing the technical requirements specified in the deliverable D1.2. It provides an overview of the architecture and highlights the most important adopted methods to provide the required functionalities. Then, the section presents the components of the architecture in detail, including the technologies considered for their implementation and the mockups of the GUI (whenever relevant/applicable). The advantage of functional encapsulation is the fact that each component can be operated independently. This increases the reliability of the overall system, and any problems that may arise can be identified more specifically in the corresponding components and then resolved. Expandability is also possible under these circumstances.

The next steps relate to definition and implementation of the technical architecture for the first working prototype for the asset management and sharing. First, interfaces (APIs) for effective data exchange must be defined for each component and the technical design of the components should be refined. Only then the individual components should be developed or/and installed and configured in an infrastructure. This work will be presented in the deliverable D2.2 documenting the first release of the XMANAI asset management bundles.

References

AEGIS Deliverable D2.1 "Semantic Representations and Data Policy and Business Mediator Conventions". (2017).

Agarwal, A., Dahleh, M., & Sarkar, T. (2019). A marketplace for data: An algorithmic solution. *Proceedings of the 2019 ACM Conference on Economics and Computation*, (pp. 701-726).

Alfredo Nazabal, C. K. (2020). Data Engineering for Data Analytics: A Classification of the Issues, and Case Studies. *IEEE Transactions on Knowledge and Data Engineering.*

Andanda, P. (2019). Towards a paradigm shift in governing data access and related intellectual property rights in big data and health-related research. *IIC-International Review of Intellectual Property and Competition Law, 50(9)*, pp. 1052-1081.

Ansari, A. F., & Soh, H. (2019). Hyperprior Induced Unsupervised Disentanglement of Latent Representations. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*, pp. 3175-3182. doi:10.1609/aaai.v33i01.33013175

Atkinson, R. (2019, 01 22). *IP protection in the data economy: Getting the balance right on 13 critical issues*. Retrieved from https://ssrn.com/abstract=3324641

Balint, F. B., & Truong, H. L. (2017). On supporting contract-aware IoT dataspace services. *2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)* (pp. 117-124). IEEE.

Bray, E. T. (2017, 10). *The JavaScript Object Notation (JSON) Data Interchange Format*. Retrieved from https://datatracker.ietf.org/doc/html/rfc8259

Cabrera, D., Sancho, F., Long, J., Sánchez, R., Zhang, S., Cerrada, M., & Li, C. (2019). Generative adversarial networks selection approach for extremely imbalanced fault diagnosis of reciprocating machinery. *IEEE Access, 7*, pp. 70643-70653.

Caimi, C., Gambardella, C., Manea, M., Petrocchi, M., & Stella, D. (2015). Legal and technical perspectives in data sharing agreements definition. *Annual Privacy Forum* (pp. 178-192). Springer, Cham.

Cao, T.-D., Pham, T.-V., Vu, Q.-H., Truong, H.-L., Le, D.-H., & Dustdar, S. (2016). MARSA: A marketplace for realtime human sensing data. *ACM Transactions on Internet Technology (TOIT), 16*(3), 1-21.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research, 16*, pp. 321-357.

Cheishvili, A., & Fan, D. (2018). *Genesis AI Protocol.* Retrieved from https://www.genesisai.io/static/whitepaper.pdf

Chen, L., Koutris, P., & Kumar, A. (2019). Towards model-based pricing for machine learning in a data marketplace. *International Conference on Management of Data* (pp. 1535-1552). Proceedings of the 2019 International Conference on Management of Data.

Cismondi, F., Fialho, A. S., Vieira, S. M., Reti, S. R., Sousa, J. M., & Finkelstein, S. N. (2013). Missing data in medical databases: Impute, delete or classify? *Artificial intelligence in medicine, 58*(1), 63-72.

Dai, Z., Yang, Z., Yang, F., Cohen, W., & Salakhutdinov, R. (2017). *Good semi-supervised learning that requires a bad gan.* arXiv:1705.09783.

Dalessandro, B., Perlich, C., & Raeder, T. (2014). Bigger is better, but at what cost? estimating the economic value of incremental data assets. *Big data, 2*(2), 87-96.

Daniel, F., & Luca, G. (2019). A service-oriented perspective on blockchain smart contracts. *IEEE Internet Computing, 23*(1), 46-53.

Di, H., Ke, X., Peng, Z., & Dongdong, Z. (2019). Surface defect classification of steels with a new semi-supervised learning method. *Optics and Lasers in Engineering, 117*, pp. 40-48.

Diez-Olivan, A., Del Ser, J., Galar, D., & Sierra, B. (2019). Data fusion and machine learning for industrial prognosis: Trends and perspectives towards Industry 4.0. *Information Fusion, 50*, 92-111.

El Emam, K., & Hoptroff, R. (2019). The synthetic data paradigm for using and sharing data. *Cutter Executive Update, 19*(6).

El Jaouhari, S., Bouabdallah, A., & Bonnin, J. (2017). Security issues of the web of things. *Managing the web of things*, pp. 389-424.

European Commision. (2020). *On artificial intelligence-A European approach to excellence and trust.* Retrieved 09 07, 2021, from https://ec.europa.eu/info/sites/default/files/commission-white-paper-artificial-intelligence-feb2020_en.pdf

European Commision. (2021, April). *Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL LAYING DOWN HARMONISED RULES ON ARTIFICIAL INTELLIGENCE (ARTIFICIAL INTELLIGENCE ACT) AND AMENDING CERTAIN UNION LEGISLATIVE ACTS.* Retrieved Decemebr 03, 2021, from https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:52021PC0206

Fan, W., & Geerts, F. (2012). Foundations of data quality management. *Synthesis Lectures on Data Management, 4*(5), 1-217.

Ferguson, M. K., Ronay, A. K., Lee, Y.-T. T., & Law, K. H. (2018). Detection and segmentation of manufacturing defects with convolutional neural networks and transfer learning. *Smart and sustainable manufacturing systems, 2*.

Fernández, A., Garcia, S., Herrera, F., & Chawla, N. (2018). SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research, 61*, pp. 863-905.

Fernandez, R. C., Subramaniam, P., & Franklin, M. J. (2020). Data market platforms: Trading data assets to solve data problems." arXiv preprint arXiv:2002.01047 (2020).

Fricker, S. A., & Maksimov, Y. V. (2017). Pricing of data products in data marketplaces. *International Conference of Software Business* (pp. 49-66). Springer, Cham.

Fruhwirth, M., Rachinger, M., & Prlja, E. (2020). Discovering business models of data marketplaces. *Proceedings of the 53rd Hawaii International Conference on System Sciences.*

Fung, B., Wang, K., Fu, A., & Philip, S. (2010). *Introduction to privacy-preserving data publishing: Concepts and techniques.* CRC Press.

Gessendorfer, J., Beste, J., Drechsler, J., & Sakshaug, J. W. (2018). Statistical Matching as a Supplement to Record Linkage: A Valuable Method to Tackle Nonconsent Bias? *Journal of Official Statistics, 34*, 909--933. doi:doi:10.2478/jos-2018-0045

Goertzel, B., Giacomelli, S., Hanson, D., Pennachin, C., & Argentieri, M. (2017). SingularityNET: A decentralized, open market and inter-network for AIs. *Thoughts, Theories Stud. Artif. Intell. Res.*

Golosova, J., & Andrejs, R. (2018). The advantages and disadvantages of the blockchain technology. *2018 IEEE 6th workshop on advances in information, electronic and electrical engineering (AIEEE)* (pp. 1-6). IEEE.

Gomes, M. M., da Rosa Righi, R., da Costa, C. A., & Griebler, D. (2021). Simplifying IoT data stream enrichment and analytics in the edge. *Computers & Electrical Engineering, 92*, 107110. doi:10.1016/j.compeleceng.2021.107110

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems, 27*.

Grabus, S., & Greenberg, J. (2017). Toward a metadata framework for sharing sensitive and closed data: an analysis of data sharing agreement attributes. *Research Conference on Metadata and Semantics Research* (pp. 300-311). Springer, Cham.

Grabus, S., & Greenberg, J. (2019). The Landscape of Rights and Licensing Initiatives for Data Sharing. *Data Science Journal, 18*(1).

Gu, L., Baxter, R., Vickers, D., & Rainsford, C. (2003). Record Linkage: Current Practice and Future Directions. *CSIRO Mathematical and Information Sciences Technical Report, 3*. Retrieved June 2003, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.14.8119&rep=rep1&type=pdf

Heckman, J. R., Boehmer, E. L., Peters, E. H., Davaloo, M., & Kurup, N. G. (2015). A pricing model for data markets. *iConference 2015 Proceedings.*

Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G. D., Gutierrez, C., . . . Neumaier, S. (2021, July). Knowledge Graphs. *ACM Computing Surveys, 54*, 1–37. doi:10.1145/3447772

Hu, V., Ferraiolo, D., Kuhn, R., Friedman, A., Cogdell, M., Schnitzer, A., . . . Scarfone, K. (2013). Guide to attribute based access control (abac) definition and considerations (draft). *NIST special publication, 800(162)*, pp. 1-54.

Huang, E., Peng, L., Palma, L. D., Abdelkafi, A., Liu, A., & Diao, Y. (2018). Optimization for Active Learning-based Interactive Database Exploration. *Proc. {VLDB} Endow., 12*(1), 71-84. doi:10.14778/3275536.3275542

Ianni, M., Masciari, E., Mazzeo, G. M., M, M., & Zaniolo, C. (2020). Fast and effective Big Data exploration by clustering. *Future Generation Computer Systems, 102*, 84-94. Retrieved from https://doi.org/10.1016/j.future.2019.07.077

Jaatun, M., Tøndel, I., Moe, N., Cruzes, D., Bernsmed, K., & Haugset, B. (2018). Accountability requirements in the cloud provider chain. *Symmetry, 10(4)*, p. 124.

Jayabalan, M., & Rana, M. (2018). Anonymizing healthcare records: a study of privacy preserving data publishing techniques. *Advanced Science Letters, 24(3)*, pp. 1694-1697.

Jones, L. C. (2021, January 17). *Data Sharing Is a Business Necessity To Accelerate Digital Business.* Retrieved 07 12, 2021, from cdotrends.com: https://www.cdotrends.com/story/15303/data-sharing-business-necessity-accelerate-digital-business

Kettimuthu, R., Allcock, W., Liming, L., Navarro, J., & Foster, I. (2007, 03). Gridcopy: Moving data fast on the grid. *IEEE International Parallel and Distributed Processing Symposium*, pp. 1-6.

Ko, R., Jagadpramana, P., Mowbray, M., Pearson, S., Kirchberg, M., Liang, Q., & Lee, B. (2011, 07). TrustCloud: A framework for accountability and trust in cloud computing. *IEEE World Congress on Services*, pp. 584-588.

Ko, R., Lee, B., & Person, S. (2011). owards Achieving Accountability, Auditability and Trust in Cloud Computing. *International workshop on Cloud Computing: Architecture, Algorithms and Applications (CloudComp2011)*, p. 5.

Koutroumpis, P., & Leiponen, A. (2013). Pantelis, Koutroumpis, and Leiponen Aija. "Understanding the value of (big) data. *2013 IEEE International Conference on Big Data* (pp. 38-42). IEEE.

Koutroumpis, P., Leiponen, A., & Thomas, L. D. (2017). *The (unfulfilled) potential of data marketplaces.* ETLA Working Papers.

Koutroumpis, P., Leiponen, A., & Thomas, L. D. (2020). Markets for data. *Industrial and Corporate Change, 29*(3), 645-660.

Krishnan, K. (2013). *Data warehousing in the age of big data.* Newnes. doi:10.1016/C2012-0-02737-8

Kumar, A., Finley, B., Braud, T., Tarkoma, S., & Hui, P. (2020). Marketplace for ai models. *arXiv e-prints*. Retrieved from https://arxiv.org/abs/2003.01593

Kumar, A., Finley, B., Braud, T., Tarkoma, S., & Hui, P. (2021). Sketching an AI Marketplace: Tech, Economic, and Regulatory Aspects. *IEEE Access, 9*, 13761-13774.

Kumar, A., Sattigeri, P., & Fletcher, T. (2017). Semi-supervised learning with gans: Manifold invariance with improved inference. *Advances in Neural Information Processing Systems, 30*.

Laurent, M., & Bouzefrane, S. (2015). *Digital identity management.* Elsevier.

Lin, X., Li, J., Wu, J., Liang, H., & Yang, W. (2019). Making knowledge tradable in edge-AI enabled IoT: A consortium blockchain-based efficient and incentive approach. *IEEE Transactions on Industrial Informatics, 15*(12), 6367-6378.

Liu, X., Wu, J., & Zhou, Z. (2008). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39(2)*, pp. 539-550.

Liu, Z., & Zhang, A. (2020). A Survey on Sampling and Profiling over Big Data (Technical Report). *ArXiv, abs/2005.05079*.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., & Zhang, G. (2015). Transfer learning using computational intelligence: A survey. *Knowledge-Based Systems, 80*, 14-23.

Lu, X., & Chengrong, W. (2018). ADVANCED UDT PROTOCOL FOR BIG DATA TRANSFER IN HIGH-SPEED NETWORK. *Computer Applications and Software*, p. 6.

Lusa, L. (2012). Evaluation of smote for high-dimensional class-imbalanced microarray data. *11th international conference on machine learning and applications, Vol. 2*, pp. 89-94.

Malhotra, P., TV, V., Vig, L., Agarwal, P., & Shroff, G. (2017). TimeNet: Pre-trained deep recurrent neural network for time series classification. *25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.* Bruges, Belgium. Retrieved from arXiv preprint arXiv:1706.08838

Mao, W., Liu, Y., Ding, L., & Li, Y. (2019). Imbalanced fault diagnosis of rolling bearing based on generative adversarial network: A comparative study. *IEEE Access, 7*, pp. 9515-9530.

Mehri, V., & Tutschku, K. (2017). Flexible privacy and high trust in the next generation internet: The use case of a cloud-based marketplace for AI. *SNCNW-Swedish National Computer Networking Workshop.* Halmstad: Halmstad university.

Niyato, D., Alsheikh, M. A., Wang, P., Kim, D. I., & Han, Z. (2016). Market model and optimal pricing scheme of big data and Internet of Things (IoT). *2016 IEEE International Conference on Communications (ICC)* (pp. 1-6). IEEE.

Puolamäki, K., Oikarinen, E., Kang, B., Lijffijt, J., & Bie, T. (2020). Interactive Visual Data Exploration with Subjective Feedback: An Information-Theoretic Approach. *Data Mining and Knowledge Discovery, 34*. doi:10.1007/s10618-019-00655-x

Raghunathan, T. E. (2021). Synthetic data. *Annual Review of Statistics and Its Application, 8*, 129-140.

Rojas, J., Kery, M., Rosenthal, S., & Dey, A. (2017). Sampling techniques to improve big data exploration. *IEEE 7th Symposium on Large Data Analysis and Visualization (LDAV)*, (pp. 26-35). doi:10.1109/LDAV.2017.8231848

Sáez, J., Krawczyk, B., & Woźniak, M. (2016). Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition, 57*, pp. 164-178.

Sakr, M. (2018). A data model and algorithms for a spatial data marketplace. *International Journal of Geographical Information Science, 32*(11), 2140-2168.

Santos, M. S., Pereira, R. C., Costa, A. F., Soares, J. P., Santos, J., & Abreu, P. H. (2019). Generating synthetic missing data: A review by missing mechanism. *IEEE Access, 7*, 11651-11667.

Shafranovich, Y. (2005). *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. Retrieved 08 25, 2021, from https://datatracker.ietf.org/doc/html/rfc4180

Spiekermann, M. (2019). Data marketplaces: Trends and monetisation of data goods. *Intereconomics, 54*(4), 208-216. doi:10.1007/s10272-019-0826-z

Stahl, F., Schomm, F., Vomfell, L., & Vossen, G. (2015). *Marketplaces for digital data: Quo vadis?* ERCIS Working paper.

Stahl, F., Schomm, F., Vossen, G., & Vomfell, L. (2016). A classification framework for data marketplaces. *Vietnam Journal of Computer Science, 3*(3), 137-143.

Suresh, J., Srinivasan, A., & Damodaram, A. (2010, 03). Performance analysis of various high speed data transfer protocols for streaming data in long fat networks. *2010 International Conference on Recent Trends in Information, Telecommunication and Computing*, pp. 234-237.

Theodorou, V., Abelló, A., Thiele, M., & Lehner, W. (2017). Frequent patterns in ETL workflows: An empirical approach. *Data & Knowledge Engineering*, 1-16.

Thomas, L. D., & Leiponen, A. (2016). Big data commercialization. *IEEE Engineering Management Review, 44*(2), 74-90.

Truong, H.-L., Comerio, M., De Paoli, F., Gangadharan, G. R., & Dustdar, S. (2012). Data contracts for cloud-based data marketplaces. *International Journal of Computational Science and Engineering, 7*(4), 280-295.

Van Panhuis, W. G., Paul, P., Emerson, C., Grefenstette, J., Wilder, R., Herbst, A. J., . . . Burke, D. S. (2014). A systematic review of barriers to data sharing in public health. *BMC public health, 14*(1), 1-9.

Wen, L., Gao, L., & Li, X. (2017). A new deep transfer learning based on sparse auto-encoder for fault diagnosis. *IEEE Transactions on systems, man, and cybernetics: systems , 49*(1), 136-144.

Woodward, K., Kanjo, E., Oikonomou, A., & Chamberlain, A. (2020). LabelSens: enabling real-time sensor data labelling at the point of collection using an artificial intelligence-based approach. *Personal and Ubiquitous Computing, 24*(5), 709-722.

World Economic Forum. (2020, January 13). *Share to Gain: Unlocking Data Value in Manufacturing.* Retrieved from weforum.org: http://www3.weforum.org/docs/WEF_Share_to_Gain_Report.pdf

Xhafa, F., Kilic, B., & Krause, P. (2020). Evaluation of IoT stream processing at edge computing layer for semantic data enrichment. *Future Generation Computer Systems, 105*, 730-736.

XMANAI Deliverable D1.2 “XMANAI Concept Detailing, Initial Requirements, Usage Scenarios and Draft MVP”. (2021).

XMANAI Deliverable D3.1 “AI Bundles Methods and System Designs”. (2021).

XMANAI Deliverable D5.1 "System Architecture, Bundles Placement Plan and APIs Design". (2021).

Zhou, F., Yang, S., Fujita, H., Chen, D., & Wen, C. (2020). Deep learning fault diagnosis method based on global optimization GAN for unbalanced data. *Knowledge-Based Systems, 187*, p. 104837.

Zou, J., & Pavlovski, C. (2007, 10 24). Towards accountable enterprise mashup services. *Proceedings of ICEBE 2007, IEEE International Conference on e-Business Engineering and the Workshops SOAIC 2007*, pp. 205-212.

Zwattendorfer, B., Zefferer, T., & Stranacher, K. (2014, 04). An Overview of Cloud Identity Management-Models. *WEBIST (1)*, pp. 82-92.

List of Acronyms/Abbreviations

Acronym/ Abbreviation	Description
AI	Artificial Intelligence
BR	Business Requirement
DoA	Description of Action
IIRA	Industrial Internet Reference Architecture
MVP	Minimum Viable Product
RAMI 4.0	Reference Architectural Model for Industrie 4.0
TR	Technical Requirement
WP	Work Package
XAI	Explainable Artificial Intelligence