Ref. Ares(2019)2288438 - 31/03/2019

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

ICT-14-2016-2017: Big Data PPP: cross-sectorial and cross-lingual data integration and experimentation

ICARUS:

"Aviation-driven Data Value Chain for Diversified Global and Local Operations"

D3.1 – ICARUS Architecture, APIs Specifications and Technical and User Requirements

Workpackage:	WP3 – ICARUS Platform Design		
Authors:	UBITECH, SUITE5, SILO, UCY, ENG, PACE, AIA, CELLOCK, ISI		
Status:	Final	**Classification:**	Public
Date:	31/03/2019	**Version:**	1.0

ICARUS Project Profile

Grant Agreement No.:	780792
Acronym:	ICARUS
Title:	Aviation-driven Data Value Chain for Diversified Global and Local Operations
URL:	http://www.icarus-h2020.eu
Start Date:	01/01/2018
Duration:	36 months

Partners

UBITECH	UBITECH (UBITECH)	Greece
ENGINEERING	ENGINEERING - INGEGNERIA INFORMATICA SPA (ENG)	Italy
PACE	PACE Aerospace Engineering and Information Technology GmbH (PACE)	Germany
Suite5	SUITE5 DATA INTELLIGENCE SOLUTIONS LIMITED (SUITE5)	Cyprus
University of Cyprus	UNIVERSITY OF CYPRUS (UCY)	Cyprus
CINECA	CINECA CONSORZIO INTERUNIVERSITARIO (CINECA)	Italy
OAG	OAG aviation Worldwide LTD (OAG)	United Kingdom
SingularLogic	SingularLOGIC S.A. (SILO)	Greece
	ISTITUTO PER L'INTERSCAMBIO SCIENTIFICO (ISI)	Italy
CELLOCK	CELLOCK LTD (CELLOCK)	Cyprus
ATHENS INTERNATIONAL AIRPORT	ATHENS INTERNATIONAL AIRPORT S.A (AIA).	Greece

Document History

Version	Date	Author (Partner)	Remarks
0.10	21/09/2018	Dimitrios Miltiadou (UBITECH)	Initial draft of the Table of Contents (ToC)
0.20	10/10/2018	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Initial outline of Sections 2-6
0.25	17/10/2018	Ralph Schultz, Henry Puls (PACE), Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Initial contribution to section 2
0.30	25/10/2018	Ralph Schultz, Henry Puls (PACE), Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Updated contribution to section 2
0.35	07/11/2018	Ralph Schultz, Henry Puls (PACE), Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri(Suite5)	Updated contribution to section 2, Initial contribution to section 3
0.40	16/11/2018	Ralph Schultz, Henry Puls (PACE), Corrado Gioannini, Luca Rossi (ISI), Haris Zacharatos, Neofytos Vlotomas (CELLOCK), Nikolaos Papagiannopoulos (AIA), Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Updated contribution to section 3
0.45	26/11/2018	Ralph Schultz, Henry Puls (PACE), Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Updated contribution to section 3
0.50	03/12/2018	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Initial contribution to section 4
0.55	14/12/2018	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Updated contribution to section 4
0.60	21/12/2018	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Updated contribution to section 4, Initial contribution to section 5
0.65	14/01/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Marios Zacharias, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contributions to sections: 5.3.1, 5.3.8, 5.3.11 by SILO, 5.3.2, 5.3.5, 5.3.6, 5.3.7, 5.3.10, 5.3.12, 5.3.13, 5.3.18 by UBITECH, 5.3.3, 5.3.4, 5.3.9, 5.3.14 by Suite5, 5.3.16, 5.3.17, 5.3.19 by ENG, 5.3.15, 5.3.20, 5.3.21 by UCY
0.70	30/01/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Marios Zacharias, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contributions to sections: 5.3.1, 5.3.8, 5.3.11 by SILO, 5.3.2, 5.3.5, 5.3.6, 5.3.7, 5.3.10, 5.3.12, 5.3.13, 5.3.18 by UBITECH, 5.3.3, 5.3.4, 5.3.9, 5.3.14 by Suite5, 5.3.16, 5.3.17, 5.3.19 by ENG, 5.3.15, 5.3.20, 5.3.21 by UCY
0.75	13/02/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia	Updated contributions to sections: 5.3.1, 5.3.8, 5.3.11 by SILO,

Version	Date	Author (Partner)	Remarks
		Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Marios Zacharias, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	5.3.2, 5.3.5, 5.3.6, 5.3.7, 5.3.10, 5.3.12, 5.3.13, 5.3.18 by UBITECH, 5.3.3, 5.3.4, 5.3.9, 5.3.14 by Suite5, 5.3.16, 5.3.17, 5.3.19 by ENG, 5.3.15, 5.3.20, 5.3.21 by UCY
0.80	28/02/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Marios Zacharias, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contributions to sections: 5.3.1, 5.3.8, 5.3.11 by SILO, 5.3.2, 5.3.5, 5.3.6, 5.3.7, 5.3.10, 5.3.12, 5.3.13, 5.3.18 by UBITECH, 5.3.3, 5.3.4, 5.3.9, 5.3.14 by Suite5, 5.3.16, 5.3.17, 5.3.19 by ENG, 5.3.15, 5.3.20, 5.3.21 by UCY
0.85	14/03/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Marios Zacharias, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contributions to sections: 5.3.1, 5.3.8, 5.3.11 by SILO, 5.3.2, 5.3.5, 5.3.6, 5.3.7, 5.3.10, 5.3.12, 5.3.13, 5.3.18 by UBITECH, 5.3.3, 5.3.4, 5.3.9, 5.3.14 by Suite5, 5.3.16, 5.3.17, 5.3.19 by ENG, 5.3.15, 5.3.20, 5.3.21 by UCY
0.90	23/03/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Updated full draft circulated for internal review
0.90_PACE	26/03/2019	Ralph Schultz (PACE)	Internal review
0.90_ISI	26/03/2019	Corrado Gioannini (ISI)	Internal review
0.90_S5	26/03/2019	Fenareti Lampathaki (Suite5)	Internal review
0.95	29/03/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Updated version addressing comments received during the internal review process
0.99	31/03/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Final version for submission to the EC
1.0	31/03/2019	Dimitrios Alexandrou (UBITECH)	Submission to the EC

Executive Summary

The document at hand, entitled "ICARUS Architecture, APIs Specifications and Technical and User Requirements" constitutes a report of the preliminary efforts and the produced results of Tasks T3.1 "Technology Requirements", T3.2 "Demonstrator User Requirements "and T3.3 "Platform Architecture Design and APIs Specifications". The purpose of this deliverable is to deliver the user requirements and the technical requirements of the ICARUS platform, as well as to deliver the first version of the conceptual architecture of the ICARUS platform. Hence, the scope of the current report can be described in the following axes:

- To define the ICARUS agile development methodology, that is adopted in order to facilitate the execution of all the project's development activities in a solid and organised manner. This methodology is a tailored to the needs of the ICARUS project agile development methodology. Within this methodology, the agile processes, instruments, roles and methods that are adopted in all the phases of the development of the ICARUS platform are defined. As part of this methodology, the methods for defining and collecting the User Stories in a structured way are documented. Furthermore, the requirements engineering process that is used in the requirements elicitation is documented, the requirements' key characteristics and classification is presented, and the ICARUS stakeholders and their interactions with the ICARUS platform is clearly defined.
- To present the collected User Stories that are stemming directly from the demonstrator partners of the ICARUS project. The User Stories contain a high-level description of the excepted behaviour of all sub-systems of the platform from the end-user perspective. They have been prepared following the predefined structure "*As a <user-type (stakeholder)>, I want to <user-requirement> so that <reason>".* Furthermore, for each User Story a set of additional management information, such as a unique identity, their category, their priority and their acceptance criteria, is collected to facilitate their analysis in the next phases of the methodology. The User Stories were collected through internal focus groups in the demonstrators and after iterations between the demonstrators and the technical partners 57 User Stories were collected.
- To document the user requirements of the ICARUS platform that are the results of the analysis of the collected User Stories. A comprehensive set of user requirements were extracted following the requirements extraction techniques of the agile software development. These elicited user requirements were further processed in order to adhere the key characteristics, as defined in the methodology, and to ensure their usefulness in the design of the ICARUS platform. Moreover, each user requirement has been assigned to a relevant feature category and were classified into the functional and the non-functional requirements of the ICARUS platform. From the analysis of the User Stories, 85 functional requirements were consolidated in ICARUS.

- To document the technical requirements of the ICARUS platform that derived from the comprehensive analysis of the extracted user requirements. As with the user requirements, the technical requirements also comply with the key characteristics defined in the methodology in order to be leveraged in the design and the specification definition of the components that will be integrated in the ICARUS platform. Hence, the user requirements were further elaborated in 72 technical requirements, that span the different methodology phases defined in D1.2, are based on the MVP features and take into consideration the additional requirements that were extracted from the theoretical approaches of WP2 and from the feedback received during the external validation of the ICARUS MVP. The list of concrete and solid technical requirements constitutes the complete requirements backlog of the ICARUS platform that will be maintained during the project implementation in order to guide all development tasks.
- To deliver the first version of the conceptual architecture of the integrated ICARUS platform that will drive the implementation activities of the ICARUS platform. The ICARUS architecture is designed in a modular manner and is composed by a set of 21 key components with distinct roles and scope towards the aim of providing the envisioned platform features. The technical requirements were thoroughly analysed and the results were utilised in the design of the necessary components which will address the ICARUS stakeholders' needs as expressed into these requirements. For each component, a comprehensive description of the design and functionalities is documented ensuring that it addresses a specific set of technical requirements from the list of the ICARUS technical requirements.

The current deliverable presents the first version of the ICARUS conceptual architecture, as well as the user and technical requirements. However, the design of the architecture is a living process that will last until M32 as per the ICARUS Description of Action. Thus, D3.1 constitutes a living document that will include the updates that will be based on further identified functional requirements translated into technical requirements, originating mainly from the evaluation and feedback received from the demonstrator partners, and that will introduce updates and refinements in the ICARUS architecture and will be presented in the upcoming versions of this deliverable.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The scope of D3.1 is to document the preliminary efforts undertaken within the context of Tasks T3.1 "Technology Requirements", T3.2 "Demonstrator User Requirements" and T3.3 "Platform Architecture Design and APIs Specifications". The deliverable D3.1 is prepared in accordance with the ICARUS Description of Action and constitutes the first iteration of the work performed under these tasks and will be provide the first version of the conceptual architecture of the integrated ICARUS platform based on the extracted user requirements and identified technical requirements.

Towards this end, the scope of the current deliverable is:

- To present the ICARUS agile development methodology that is adopted in order to facilitate the successful execution of each phase and activity of the project. This methodology is based in the standard agile methodology with several adaptions to the needs of the ICARUS project.
- To collect the User Stories that are coming directly from the demonstrator partners of the ICARUS project. The User Stories are defined following the template specified in the methodology that ensures that the end-user's perspective for the platform's expected behaviour is captured.
- To extract and document the user requirements from the collected User Stories taking into account the MVP features (defined in D1.2). The elicited user requirements are classified into the functional and the non-functional requirements of the ICARUS platform.
- To document the technical requirements of the ICARUS platform as the outcome of the in-depth analysis of the functional and non-functional user requirements. The list of technical requirements will be maintained during the platform implementation as a requirements backlog. In addition to the technical requirements from the collected User Stories, a set of additional technical requirements, as result of the feedback received from the external ICARUS MVP validation, are documented.
- To design and document the first version of the conceptual architecture of the integrated ICARUS platform. The architecture will be composed by a set of modular components that will address all the needs of all the different stakeholders of the platform. To ensure this, the technical requirements are translated into specific platform features that that are grouped under the modular components of the platform. Especially, for the case of technical requirements originating from the external ICARUS MVP validation (e.g. end-to-end encryption) a series of changes in the initial design were introduced in order to ensure that it brings added value to the aviation data value chain stakeholders. Such changes required internal discussions and brainstorming how they could be effectively addressed in the architecture. Even if they imposed significant deviations from the initial design, the ICARUS consortium decided to properly address them to assuage the security-related concerns of the external aviation stakeholders with whom the consortium interacted. The architecture is

designed in a modular manner that will enable the desired scalability, interoperability and extensibility.

It should be noted that the identification and analysis of the functional and non-functional requirements, as well as their translation into technical requirements is a living process that will last until M32 according to the ICARUS Description of Action and as the project evolves it will be constantly updated and documented in the upcoming versions of this deliverable.

1.2 Document Approach

In the current deliverable a systematic and comprehensive approach is followed in order to deliver the outcomes of T3.1, T3.2 and T3.3 as depicted in the Figure 1-1.

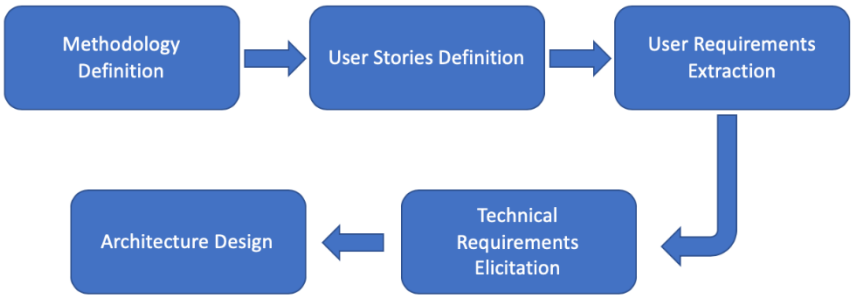

Figure 1-1: Requirements Identification and Elicitation Methodology

At first, the requirements identification and elicitation methodology is defined. In the ICARUS project the agile development processes, instruments, roles and methods are adopted in order to support in a clear manner each phase and activity of the project. In the methodology definition the outcomes of the ICARUS Methodology, which includes various phases, with each phase having its own specific steps, and the ICARUS MVP as derived from D1.2 are taken as input. The methodology provides the guidance for the user story collection, as well as the requirement definition in terms of key characteristics and requirements classification. Moreover, the ICARUS stakeholders and their interactions with the ICARUS platform are identified.

Following the methodology definition, the user stories that are stemming directly from the demonstrator partners of the ICARUS project are collected. These User Stories are analysed towards the aim of extracting the user requirements that fully comply with the requirements characteristics defined in the methodology. From the comprehensive set of user requirements that is derived from the User Stories analysis, the requirements are further classified into functional, platform and demonstrator, and non-functional requirements. Furthermore, the user requirements of each type are also categorised based on the ICARUS methodology phases and the mapping between the User Stories and the derived MVP features from D1.2 is documented.

From the elicited user requirements, the technical requirements are extracted. For each technical requirement, a high-level categorisation based on the phases of the ICARUS methodology is provided,

as well as the mapping between this technical requirement and the relevant user requirements. Moreover, in this step the mapping between the extracted technical requirements and the MVP features from D1.2 is provided as one more assurance of the requirements' validity.

Following the technical requirements elicitation, the design of the conceptual architecture of the integrated ICARUS platform is documented. In this step, the extensive analysis of the technical requirements provides the design and definition of the components of the architecture that will be integrated in order to provide the envisioned platform features that will address the stakeholders' needs based on the input provided from all previous steps. The ICARUS architecture is modular and for each component of the architecture the design and specifications are provided, along with the list of addressed requirements.

1.3 Relationship with other ICARUS Results

Deliverable D3.1 is released in the scope of WP3 "ICARUS Platform Design" activities and reports the preliminary efforts undertaken within the context of Tasks T3.1 "Technology Requirements", T3.2 "Demonstrator User Requirements " and T3.3 "Platform Architecture Design and APIs Specifications". Moreover, as depicted in Figure 1-2 , the outcomes of T3.1 and T3.2 are provided as input to T3.3 in order to formulate the initial version of the integrated ICARUS platform. As the project evolves, the updated outcomes of T3.1 and T3,2 will be also provided as input to T3.3, T3.4 an T3.5 in order to formulate the designs of the Core Data Service Bundles and the Added Value Services that will drive the upcoming versions of the integrated ICARUS platform.

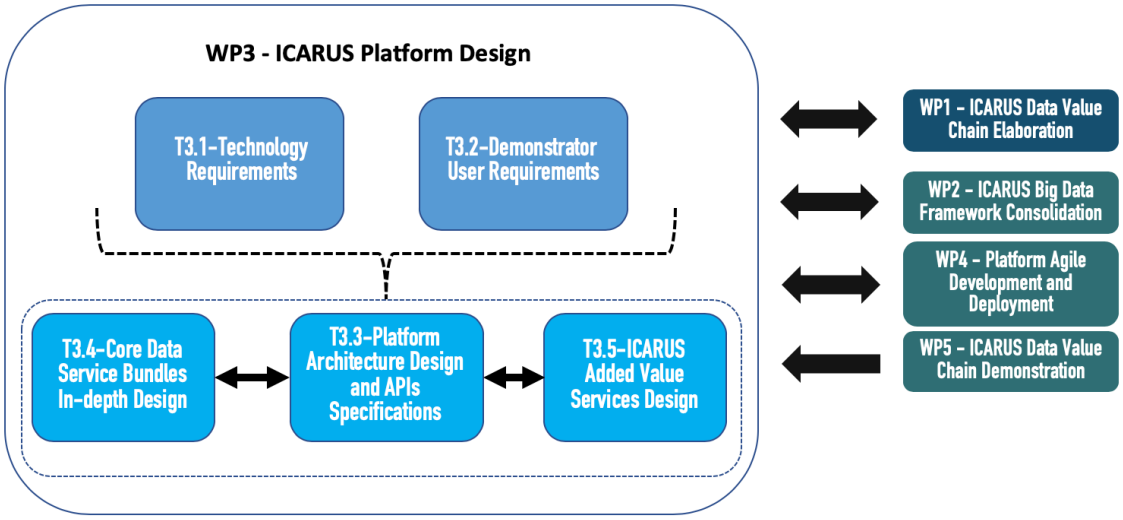

Figure 1-2: Relation to other ICARUS Work Packages

D3.1 and WP3, are directly related to the outcomes of WP1 "ICARUS Data Value Chain Elaboration" with regard to ICARUS methodology, the ICARUS Minimum Viable Product (MVP) and WP2 "ICARUS Big Data Framework Consolidation" with regard to the Big Data Framework methods for data collection, data provenance, data safeguarding, data curation, data linking, data analytics and data sharing.

The external ICARUS MVP validation activities that are performed within the context of WP1, that resulted in new additional requirements, also affected the progress of the tasks of WP3. In order to ensure proper alignment with such additional requirements across the ICARUS results, the deliverables of WP2 and WP3 proceeded almost in parallel with D3.1.

D3.1 provides the necessary design and the specifications of the components, as well as the overall platform design, to WP4 that will deliver the implementation of these components following the approach formulated in the WP3 activities. Finally, the feedback that will be collected from the continuous evaluation of the platform as a result of the WP5 activities will be fed in WP3 and will drive the updates and adjustments in the design and specifications of the components of the platform, as well as the overall integrated ICARUS platform.

1.4 Structure

The structure of the document is as follows:

- In Section 2, the requirements identification and elicitation methodology is defined. At first, the ICARUS Agile development methodology is presented, providing an overview of the adopted development processes, instruments, roles and methods. Moreover, in this section the User Stories definition process is presented, as well as the requirements definition process.
- In Section 3, the elicited user requirements are documented. The user requirements are classified into platform and demonstrator functional requirements and non-functional requirements. For each requirement, the relevant category based on the ICARUS methodology is documented along with the related User Stories and MVP features.
- In Section 4, the extracted technical requirements are presented. The technical requirements are grouped logically based on the platform features and for each requirement the relevant phase of the ICARUS methodology and the relevant user requirements are documented. Furthermore, the mapping between the technical requirements and the MVP features from D1.2 is presented.
- In Section 5, the first version of the conceptual architecture of the integrated ICARUS platform is presented. Additionally, the mapping between the technical requirements and the designed components is presented. For each component of the architecture, the design and functionalities overview are presented along with the list of addressed requirements.
- Section 6 concludes the deliverable. It outlines the main findings of the deliverable which will guide the future research and technological efforts of the consortium.
- Annex I lists the references included in the present deliverable.
- Annex II documents the collected User Stories.
- Annex III presents the elicited user requirements.
- Annex IV documents the technical requirements backlog.

2 Requirements Identification and Elicitation Methodology

A major benefit of research projects is the different expert skills brought by each partner into the consortium. In order to ensure that the collaboration of all partners in ICARUS leads to excellent results, development processes have been defined in a clear manner for each phase and activity of the project.

The following paragraphs describe the adaption of the agile development processes, instruments, roles and methods that will support the ICARUS project activities in the best way. The common understanding of what has to be done and how it has to be done, enables all members of the project to fulfil their assignments and to develop their parts being ensured, that there will be a harmonised and controlled interaction with the other project partners.

An important step in the agile development is the definition of the requirements elicitation process, which is done in the second part of this chapter. The ICARUS requirements elicitation process starts with the collection of user requirements for the defined user stories, brought by partners who are implementing a demonstrator in the project and ends with the derivation of technical requirements out of these user requirements, by technical partners who will perform the platform implementation.

As one decisive result of deliverable *D1.2: The ICARUS Methodology and MVP*, the breakdown of the ICARUS methodology will be taken as input. Precisely, the ICARUS methodology is divided into:

- **Phase I** - Data Collection
- **Phase II** – Data Enrichment
- **Phase III** – Asset Storage
- **Phase IV** – Asset Exploration and Extraction
- **Phase V** – Data Analytics
- **Phase VI** – Added Value Services
- **Phase VII** – Service Collection.

Each phase consists of one or more step and illustrates how the different ICARUS stakeholders interact with ICARUS. These phases or steps will be used to cluster the user requirements and the User Stories to functional sets, which will be the input for the next step, the technical requirements definition.

Furthermore, User Stories are the instrument, which bridges the technical requirements and the agile development process. Thus, this section describes, among others, how a set of one or more requirements can be translated into a concrete software feature. Additionally, development related information like priority and test case definition is completing a user story. Assigning a user story to a coming release defines the envisaged functionality of a product increment.

2.1 ICARUS Agile development methodology

This paragraph describes the methods and instruments of the standard agile methodology and their adaptions to the needs of the development of the ICARUS platform.

One characteristic of the agile development is the assignment of activities to precise specified roles (Cohn, 2010):

- **Product Owner**: The viewpoint of the product owner is the perspective of the customer. The product owner translates the product related interest of the customer into a functional description – so called User Stories - which will feed into the development process. By assigning a priority to a functionality, the product owner determines the sequence of functions development. Finally, the product owner checks during an acceptation if the functional requirement has been implemented completely and correctly.
- **System Architect:** A System Architect is a developer which special skills. A system architect designs the system architecture to ensure among other things reliability, availability and maintainability of the product during the development process.
- **Developer:** A developer executes development task to realise the functional request defined by the product owner.
- **Quality Assurance:** The Quality Assurance ensures during the development that newly implemented functions are working as specified and that the rest of the product works still faultless.

Figure 2-1 describes the core parts of all agile development processes in which the recurring execution of a defined sequence of process phases is presented. The following table describes the typical activities of each phase for the software development process.

Table 2-1: Phases, Roles and Activities in an Agile Software Development Process

Phases	Participants	Actions Taken
Plan	Product Owner System Architect Developer Quality Assurance	Define and specify user features for the next product increment. Features are described by User Stories. To prepare the acceptance of a User Story by the Product Owner, special test criteria have to be defined and specified by the whole team. Only if all of these criteria are fulfilled, the user story will be accepted by the Product Owner.
Design	System Architect Developer	As a preparation for the development, the system architecture will be designed by the system architects and developers.
Build	Developer Quality Assurance	The software will be developed by consideration of predefined test criteria
Test	Quality Assurance	The predefined test criteria will be translated into test cases. The test cases will be applied to the product increment.
Review	Customer Product Owner System Architect Developer Quality Assurance	The development team presents the new features of the product increment to the customer. The customer checks if the requirements are fulfilled completely. In case of required changes new User Stories will be defined

These phases can vary depending on certain requirements of the project. Nevertheless, each iteration creates a product increment with available features defined in the planning, designed, built and verified in the following phases. The product increment is the input for the next iteration.

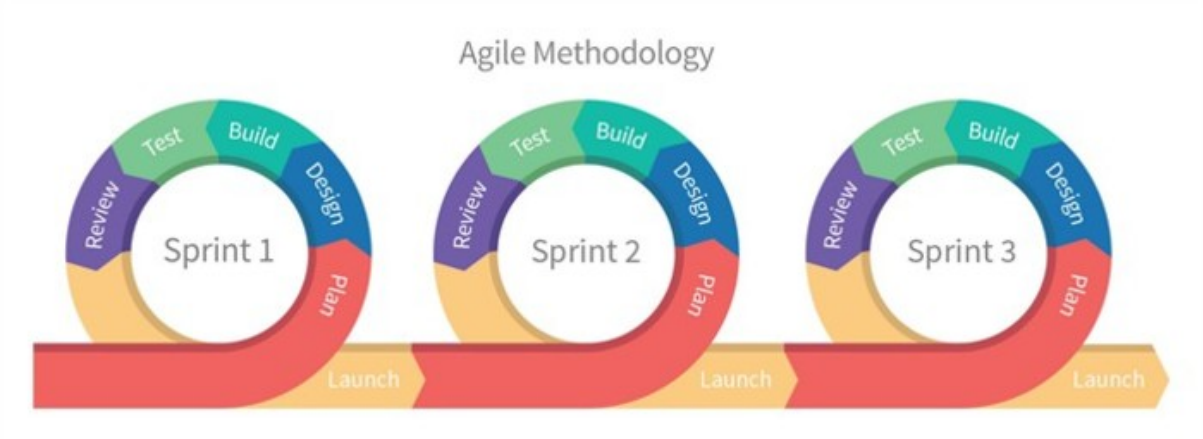

Figure 2-1: Sprint Based Agile Methodology (Source: http://www.illuminationworksllc.com/agile-enablement/)

The following picture shows the agile development process in a more general manner. User requests for a new feature or the enhancement of an existing feature of the product are managed in a so called backlog. For each iteration a set of these requests will be determined and brought into the development. The resulting release will be delivered to the stakeholders of the project, who will evaluate it and provide feedback. The ICARUS development process will embrace this process with the necessary adjustments to fit the needs of the project.

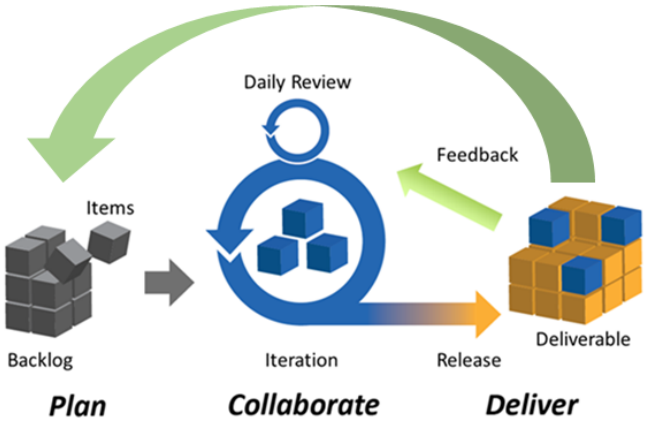

Figure 2-2: General Agile Development Process (Source: http://empireone.com.au/agile-iterative-lean-development-what-does-it-all-mean)

Backlog:

The backlog manages user requested product features. In ICARUS we will start with the collection of product features and translate them into the so-called User Stories. Each User Story is constructed in a predefined schema, which is explained in section 2.2. A User Story contains a clearly separated

product function that enables developers to give a rather good workload estimation. Finally, a priority is assigned to the User Stories. It will be used in the selection of User Stories for the next iteration and by the developer for the decision in which sequence the User Stories will be implemented.

Iteration:

The iteration is the software development process for a predefined set of backlog items. Due to the earlier estimation and the knowledge about the capacity of the development team, a set of User Stories will be selected which fits into an iteration.

Each User Story will run through the phases plan, design, build and test. Of particular importance during the development is the collaboration of all stakeholders in the process. Frequent meetings between the technical and demonstrator partners of ICARUS constitute the chosen methodology.

Deliverables

At the end of each iteration an increment of the product is available for verification and validation conducted by the ICARUS demonstrators and external stakeholders in the aviation industry. This process step will create a first assessment of available product features which allow the acceptation or lead to a rejection or change request of the requirements as a first feedback to the development team.

According to the implementation efforts some changes can be implemented immediately, and others have to be translated to new User Stories and stored in the backlog again, waiting to be selected for a certain iteration.

The ICARUS agile method and the usage of User Stories, Requirements and Backlog allows a requirement engineering process as described in Figure 2-3.

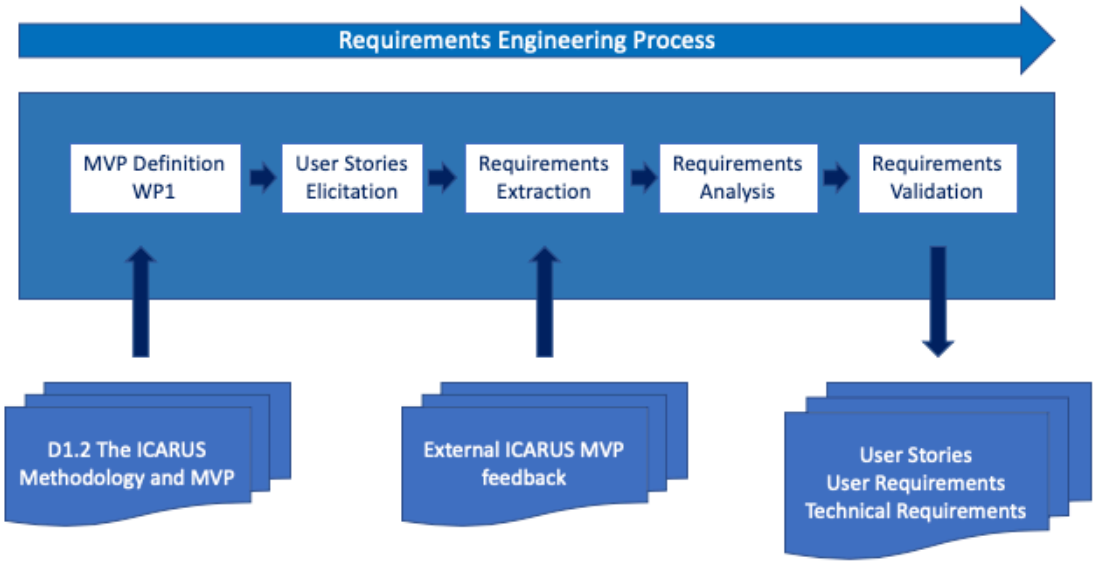

Figure 2-3 Requirements Engineering in ICARUS

In the first step, the state of research of the project topic has been analysed and the first high-level versions of the ICARUS demonstrators have been defined. Out of this knowledge a set of User Stories were defined related to the platform itself or to a certain demonstrator.

Then, the requirements derived from the User Stories in a work package overlapping collaboration. Furthermore, additional requirements are derived from the external ICARUS MVP validation. During the analysis the requirements were checked in order to ensure that the requirements fulfil the characteristics explained in section 2.3.1, e.g. redundancies and inconsistencies were eliminated, the level of detail has been harmonised and so on. Finally, the requirements were validated and accepted or rejected or adapted.

The following table assigns the agile instruments to the ICARUS adapted instruments, lists the activities and the responsible work packages and partners.

Table 2-2: ICARUS agile process responsibilities

Agile Instrument	Representation in ICARUS	Represented in WP by:
Product Owner	Represented in ICARUS project by the partners who are implementing demonstrators and using the platform features. Their input corresponds to the user requirements.	WP3, WP5: AIA, CELLOCK, ISI, OAG and PACE.
System Architect, Developer	Represented in ICARUS by the partners who are designing and implementing the ICARUS product. Their activities are the derivation of technical requirements out of user requirements, the definition of User Stories and implementation of the product increments alpha, beta, V1.0, V1.5 and V2.0.	WP2, WP3, WP4: UBITECH, Suite5, ENG, SILO, UCY
Quality Assurance	Represented in ICARUS by the partners who are defining the validation and evaluation framework, which ensures a comparable assessment of each ICARUS product increment combined with the application of the related demonstrator version.	WP5: CINECA, SILO, AIA, CELLOCK, ISI, OAG and PACE.
Iterations	ICARUS product increments are planned for - Beta version: M18 - Release 1.00: M24 - Release 1.50: M30 - Release 2.00: M36 After release, the demonstrators will be applied to verify and validate the product increments by following the steps defined in the framework. Assigned User Stories will be checked, and new or adapted requirements will be created and reported to WP3 and WP5, to be considered for the next product increment.	WP2, WP3, WP4: UBITECH, Suite5, ENG, SILO, UCY
User Stories	Defined by the partners the partners who are implementing demonstrators and using the platform features with the help of the partners who are responsible for the implementation of all features and layers of the ICARUS product.	WP3, WP4, WP5: AIA, CELLOCK, ISI, OAG, PACE (demonstrator partners) UBITECH, Suite5, ENG, SILO, UCY (technical partners)

2.2 User Stories definition

A User Story is an instrument used in Agile software development to capture a description of a software feature from an end-user perspective. The User Story describes the type of user, what they want and why. A User Story is very high level and helps to create a simplified description of a requirement.

Usually a User Story provides in one sentence enough information related to the described product feature, for which the development team can conduct a reasonable work load estimation. Furthermore, the User Story is used in planning meetings to enable the development team to design and implement the product features.

A User Story typically has a predefined structure:

- *As a <user-type (stakeholder)>, I want to <user-requirement> so that <reason>*

As mentioned earlier, the agile software development method allows the definition and adaption of User Stories during the whole project life cycle and the selection of them for implementation in the upcoming iteration.

Table 2-3 presents four examples of User Stories from the ICARUS demonstrators with the additional management information:

- **ID**: a unique identifier of the User Story composed by the demonstrator partner name and a number;
- **Category**: a generic term for grouping the required functionality. Categories have been defined in *D1.2: The ICARUS Methodology and MVP*. Categories are linked either to a complete phase of the ICARUS methodology or to a concrete step of a phase:
    - Collection
    - Curation
    - Enrichment
    - Linking
    - Exploration
    - Analytics
    - Notification
    - Recommendation
    - Sharing
    - Service Collection

  The list of categories constitutes only an initial guideline. Demonstrators may define additional categories if required.
- **User Story:**
    - As a <user-type (stakeholder)>: This is the type of the stakeholder of the story writing the user story; the roles are those defined in *D1.2: The ICARUS Methodology and MVP* and described in Table 2-6.

- I want to <user-requirement>: the requested feature or functionality that will be included in the ICARUS platform.
- So that I can <reason>: a description of the benefit or the added value of the requested feature or functionality.

- **PRIORITY**: The priority (high, medium or low) defines the importance of the user story and usually how soon should this user story be included in the upcoming development iteration.
- **VALUE**: The value (high, medium or low) defines the level of benefit or added value of the addition of the described feature or functionality to the ICARUS platform from the point of view of the stakeholder.
- **ACCEPTANCE**: The acceptance defines the criteria for the successful implementation and the successful evaluation of the implementation of the user story.

Table 2-3: Example for User Story Definition and Addition of Management Information

ID	Category	User Story			Priority	Value	Acceptance
		As a <user-type>	I want to <user requirement>	So that <reason>			
PACE_001	Collection	Data consumer	retrieve periodically the updates automatically from data providers	it is ensured that data are always up-to-date	Medium	Medium	ICARUS provides any available updates on a dataset from a data provider automatically by performing periodic checks
AIA_002	Analytics	Data consumer	login to a secure space	I can analyse my confidential data	High	High	A user should be able to analyse its confidential data in secure space
CEL_009	Analytics	Data Scientist	use a Dashboard to combine and run multiple reports	perform an analysis of the results	High	High	A user should be able to use visualisation tools for statistical analysis
ISI_006	Notification	Data consumer	Be informed about any update or modification of the license/terms of usage of datasets I am	I can take immediate action if needed and get new relevant data when possible	Low	Medium	The platform notifies the users about any update on data terms of usage

			using or interested into				

All partners in ICARUS and especially the ones that are implementing a demonstrator (AIA, CELLOCK, ISI and PACE), have been involved in the definition of the MVP (Minimum Viable Product) in WP1 by evaluating a set of potential platform features. Furthermore, all demonstrators' execution scenarios have been detailed more and more in the course of the project. The User Stories were collected through workshops between the demonstrator partners and the technical partners of the project as a continuation of the definition of the MVP. These User Stories were used in the elicitation of the functional and non-functional user requirements that are documented in section 3. All the collected User Stories are provided as an annex in Annex II.

2.3 Requirements Definition

In this section the requirements definition is elaborated, describing the definition of instruments to be used in the requirements elicitation process. In particular, the characteristics of the requirements are presented, followed by the classification of the elicited requirements into categories.

2.3.1 Requirements Characteristics

User Stories are a useful instrument to get well defined portions of system requirements in a language that users without technical background are able to use. User Stories are the input for the next action, where the technical partners are translating "user" requirements to more implementation-oriented requirements, which can be interpreted by system architects or developers.

Requirements do not have such a strong definition schema like User Stories but need to own a set of characteristics in order to be useful as an input for the design and development process (Ericson, 2015). These characteristics are listed in Table 2-4.

Table 2-4: Requirements characteristics

Characteristic	Explanation
Unitary (Cohesive)	The requirement addresses one and only one thing.
Complete	The requirement is fully stated in one place with no missing information.
Consistent	The requirement does not contradict any other requirement and is fully consistent with all authoritative external documentation.
Non-Conjugated (Atomic)	The requirement is atomic, i.e., it does not contain conjunctions. E.g., "The postal code field must validate American and Canadian postal codes" should be written as two separate requirements: (1) "The postal code field must validate American postal codes" and (2) "The postal code field must validate Canadian postal codes".
Traceable	The requirement meets all or part of a business need as stated by stakeholders and authoritatively documented.
Current	The requirement has not been made obsolete by the passage of time.
Unambiguous	The requirement is concisely stated without recourse to technical jargon, acronyms (unless defined elsewhere in the Requirements document), or other esoteric verbiage. It expresses objective facts, not subjective opinions. It is subject to one and only one interpretation.

	Vague subjects, adjectives, prepositions, verbs and subjective phrases are avoided. Negative statements and compound statements are avoided.
Specify Importance	Many requirements represent a stakeholder-defined characteristic the absence of which will result in a major or even fatal deficiency. Others represent features that may be implemented if time and budget permits. The requirement must specify a level of importance.
Verifiable	The implementation of the requirement can be determined through basic possible methods: inspection, demonstration, test (instrumented) or analysis (to include validated modelling & simulation).

2.3.2 Requirements Classification

The requirements can be classified in the two following main categories:

Functional:

Functional requirement is the declaration of the intended functionality of a system and its components as reported by a hypothetical non-technical observer. The functional requirement is facilitating the development team to determine the expected behaviour or output of the system in the case of a certain input and in which a technical problem is addressed. Additionally, within the functional requirements the outputs of the envisioned product increment when receiving the described input is described. Depending on the degree of efficiency they can be split into "platform" and "demonstrator" oriented functional requirements.

Non-functional:

As per ISO/IEC 25010:2011, the Non-functional requirements define system attributes such as security, reliability, performance, maintainability, scalability and usability. Also known as system qualities, they are just as critical as functional requirement as they safeguard the usability and effectiveness of the entire system. Failing to meet any of them can result in systems, that fail to satisfy business or markets or user needs (ScaledAgileFramework, 2018).

In some cases, non-functional requirements cannot be resolved to one function or component or layer in the architecture. And in other cases, they cannot be tested directly. Nevertheless, they have to be kept in mind for choosing the right design and implementation of a system.

Table 2-5: Requirements Definition

Type		Description
Functional	Platform	Specifies the intended functionalities of the ICARUS platform that will be used from all users of the platform (stakeholders and demonstrators).
	Demonstrator	Specifies the intended functionalities of the ICARUS platform that are relevant to the specific demonstrators' needs and that will support the applications that will be developed by each demonstrator.
Non-functional		Specify additional system properties like performance, quality, robustness and many other more.

2.4 ICARUS Stakeholders and Interactions

In the deliverable *D1.2: The ICARUS Methodology and MVP* a set of general workflow descriptions, in another word scenarios, are documented. Each of these scenarios describes on a high level, by whom and how the ICARUS platform is used. The identified stakeholders of these scenarios documented in D1.2 are utilised in the definition of the User Stories in order to present their view point. Table 2-6 lists all identified stakeholders and describes their role using the ICARUS platform.

Table 2-6: ICARUS Stakeholders

Stakeholder	Description
Data Provider	The user's main objective is to share his/her data for processing or consuming in ICARUS. *(Anyone that provides data to the platform can undertake the role of data provider, such as data scientist, data analyst, researcher, non-expert user)*
Data Consumer	The user's main objective is to consume and/or process data offered through ICARUS. *(Anyone that consumes data from the platform can undertake the role of data consumer, such as data scientist, data analyst, researcher, non-expert user)*
Service asset provider	The user's main objective is to create a service (e.g. custom-made algorithm) on top of ICARUS data value chain and make it available in ICARUS. *(Usually this role is undertaken by data scientists or data analysts and software developers)*
Service asset consumer	The user's main objective is to consume a service asset offered through ICARUS. *(Anyone that consumes the services of the platform can undertake the role of data consumer, such as data scientist, data analyst, researcher, non-expert user, software developer)*
Administrator	Installs, checks and maintains the system.

3 ICARUS User Requirements

In section 2.1 the ICARUS agile development methodology was presented. Within this methodology, all partners of the ICARUS project, who are implementing a demonstrator which uses the platform, were involved in the User Stories definition following the instructions described in the previous section. In the next crucial step of the methodology, a comprehensive set of user requirements has been derived out of these User Stories.

Initially, the elicited user requirements were checked and analysed in order to assure that each requirement fulfils the characteristics explained in 2.3.1. For a better handling during the platform implementation, each requirement has been assigned to a category and a requirement type. The list of categories derived from the phases or concrete step of a phase of the ICARUS methodology, in the same manner as it was utilised in the User Stories definition. For the identification of the category of each requirement, each phase of the ICARUS methodology has been checked for a functional mapping with the requirement. For the identification of the requirement type, a mapping has been created based on the requirements classification that was described in section 2.3.2.

Hence, the following subsections present the derived user requirements, in the form of tables, classified into the following types:

- Platform functional requirements,
- Demonstrator functional requirements,
- Non-functional requirements.

Each table contains the following information for each of the identified user requirements:

- A unique identifier of the user requirement;
- A high-level description of the requirement;
- A high-level categorisation based on the phases of the ICARUS methodology presented in D1.2, where applicable;
- The list of the related User Stories from which the requirement was elicited;
- The list of the related MVP features that were reported in D1.2;

The complete list of the ICARUS user requirements is provided also in Annex III.

3.1 Functional Requirements

3.1.1 Platform Functional Requirements

The following table presents the elicited platform functional requirements with the additional management information:

Table 3-1: Functional Requirements

ID	Description of the requirement	Category	Related User Stories	Feature ID
Req_001	ICARUS should inform users for updates on datasets.	Notification	PACE_001 AIA_007 CEL_003 CEL_010 ISI_004	PLATF_F_01 PLATF_F_02 PLATF_F_48
Req_002	ICARUS should support connections to various APIs for data exchange (import/export)	Collection	PACE_001 PACE_014 CEL_008 CEL_002	PLATF_F_01 PLATF_F_02
Req_003	ICARUS should provide functions to create and manage shortcuts and workflows related to the user's recent actions or workflows.	Analytics	PACE_002 AIA_001	PLATF_F_30 PLATF_F_51 PLATF_F_52 PLATF_F_53
Req_004	ICARUS should offer a public and a proprietary and confidential working space.	Analytics	PACE_023 AIA_002	PLATF_F_46 PLATF_F_47
Req_005	ICARUS should support tags for datasets in addition to categories.	Enrichment Linking	PACE_005 PACE_008 PACE_009 CEL_001 CEL_002 CEL_005	PLATF_F_16 PLATF_F_20
Req_006	ICARUS should support filters for tagged datasets (i.e. real-time data, historical, proprietary, public, demo/preview, etc).	Linking	PACE_005 PACE_008 PACE_009 CEL_001 CEL_002 CEL_005	PLATF_F_18
Req_007	ICARUS analytics should work with a mixture of confidential and public data.	Collection, Exploration Analytics	PACE_003 AIA_003 AIA_004 AIA_005 AIA_012	PLATF_F_04 PLATF_F_11 PLATF_F_12 PLATF_F_13 PLATF_F_26 PLATF_F_29 PLATF_F_30 PLATF_F_32 PLATF_F_40
Req_009	ICARUS should support to search for datasets by type	Exploration	PACE_005 PACE_008 PACE_009 PACE_010 CEL_001 CEL_002 CEL_003 CEL_004 CEL_005	PLATF_F_22 PLATF_F_23
Req_010	ICARUS should support to search for datasets by keywords	Exploration	PACE_005	PLATF_F_22 PLATF_F_23
Req_011	ICARUS should provide data sets which are relevant to my search.	Linking, Recommendation	PACE_005 PACE_015	PLATF_F_18 PLATF_F_51

Req_012	ICARUS should support to search for datasets by date and time.	Exploration	PACE_005	PLATF_F_22 PLATF_F_23
Req_016	ICARUS platform should ensure that the users can query data that they are authorised to access.	Exploration Analytics	PACE_006 ISI_001 ISI_002	PLATF_F_26
Req_017	ICARUS platform should offer different level of confidentiality for the datasets.	Collection, Exploration Analytics	PACE_003 PACE_021 AIA_002	PLATF_F_04 PLATF_F_26
Req_018	ICARUS platform should provide a set of advanced analytics algorithms.	Analytics	PACE_007 AIA_003 AIA_004 AIA_005 CEL_009 ISI_005	PLATF_F_33 PLATF_F_34
Req_019	ICARUS platform should provide features to either customise the defined analytics algorithms or define custom analytics algorithms.	Analytics	PACE_007 AIA_003 AIA_004 AIA_005 CEL_009 ISI_005	PLATF_F_29
Req_020	ICARUS platform should provide private space where I can store the data obtained through a query.	Analytics	PACE_023 AIA_002 AIA_003	PLATF_F_43 PLATF_F_46 PLATF_F_47
Req_022	ICARUS should support file upload and download services for common text formats such as ASCII, CSV, XML, YAML, JSON	Collection, Exploration	PACE_011 PACE_014 CEL_008	PLATF_F_41 PLATF_F_42 PLATF_F_43
Req_023	ICARUS platform should allow the user to select a file format for download if a conversion is feasible.	Exploration	PACE_011 CEL_007	PLATF_F_27
Req_024	ICARUS platform should suggest available data from other sources related to my queries.	Recommend ation	PACE_012 AIA_008 ISI_004	PLAT_F_23
Req_027	ICARUS platform should be able to combine data and provide the means to obtain them.	Collection, Exploration	PACE_013 CEL_007 ISI_001 ISI_002	PLATF_F_42
Req_028	ICARUS should support the upload of external data sets	Collection	PACE_014 PACE_021 CEL_008	PLATF_F_03
Req_029	ICARUS should provide data conversion for uploaded file format into ICARUS platform internal data format.	Curation	PACE_014 PACE_021	PLATF_F_11 PLATF_F_12 PLATF_F_13 PLATF_F_27
Req_030	ICARUS platform should provide features for adding additional (semantic) information to data assets	Linking	PACE_015	PLATF_F_16 PLATF_F_17
Req_031	The user should be able to explore the ICARUS data model and can provide suggestions to the data administrator.	Curation	PACE_015 PACE_016 CEL_008	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_032	ICARUS platform should provide features for updating my datasets	Curation	PACE_016	PLATF_F_14
Req_033	ICARUS platform should inform a user if data, for which the user owns a license to use, have been updated or deleted.	Notification	AIA_007 CEL_11 ISI_004	PlATF_F_48 PlATF_F_49

Req_034	ICARUS platform should provide features for defining and modifying the license model of my data.	Sharing	PACE_017 AIA_009 AIA_010 AIA_011 ISI_002 ISI_006 ISI_007	PLATF_F_56 PLATF_F_57 PLATF_F_58 PLATF_F_59
Req_035	ICARUS should provide indicative usage analytics on the datasets usage within the platform	Analytics	PACE_017	PLATF_F_45
Req_036	ICARUS platform should support the negotiations between data provider and data consumer until a data sharing agreement has been signed.	Sharing	PACE_017 PACE_019 AIA_010 CEL_008 ISI_007	PLATF_F_60 PLATF_F_61 PLATF_F_62
Req_037	ICARUS platform should provide the means to improve the quality level of the user's data	Collection	PACE_018 ISI_003	PLATF_F_06 PLATF_F_07
Req_038	ICARUS platform should provide mechanisms to define the licensing requirements and privacy restrictions (DSGVO/GDPR compliance) for a dataset	Collection	PACE_019 PACE_020	PLATF_F_08 PLATF_F_09 PLATF_F_10
Req_039	ICARUS platform should provide a tool for data anonymisation.	Collection	PACE_020	PLATF_F_08
Req_042	ICARUS should support a simplified data upload process, if these data will be used only by myself.	Collection	PACE_014 PACE_021 CEL_008	PLAT_F_05
Req_043	ICARUS should be able to link different datasets	Curation	PACE_015 PACE_022 AIA_001 AIA_003 AIA_004 AIA_005	PLATF_F_16
Req_045	ICARUS platform should provide features for transferring data from my confidential space to the ICARUS platform	Collection, Analytics	PACE_023	PLATF_F_43 PLATF_F_46 PLATF_F_47
Req_046	ICARUS platform should provide the functionality to manage (semantic) links between data assets.	Linking	PACE_015	PLATF_F_16
Req_047	ICARUS platform should be able to suggest integrated data sets in the context of queries and data uploads.	Linking	PACE_012 AIA_008	PLATF_F_17
Req_049	ICARUS should provide business intelligence tools that enable the automated generation of event driven alerts and customised reports and notify user about results.	Analytics, Notification	AIA_002 AIA_006	PLATF_F_37 PLATF_F_38 PLATF_F_41
Req_050	ICARUS should be able to integrate structured and unstructured data.	Collection	AIA_002	PLATF_F_14
Req_051	ICARUS should be able to report and visualise analysis results.	Analytics	AIA_004 AIA_012 AIA_015	PLATF_F_37 PLATF_F_38
Req_053	ICARUS should provide comprehensive means to visualise and to compare results (graphical, tabular, …)	Analytics	AIA_006 AIA_015 AIA_016 AIA_017	PLATF_F_37 PLATF_F_38 PLATF_F_39 PLATF_F_53

			CEL_009	
Req_054	ICARUS should be able to execute analytics and workflows automatically (through pre-scheduled jobs).	Analytics	AIA_005	PLATF_F_29 PLATF_F_30 PLATF_F_40
Req_055	ICARUS should provide dashboards and help the user compare the results with minimum number of interactions.	Analytics	AIA_006 AIA_015 AIA_016 AIA_017 CEL_009	PLATF_F_39
Req_056	ICARUS platform should support connections to web services and provide API for the upload and download of data from/to other data sources and sinks.	Collection	AIA_007	PLATF_F_01 PLATF_F_02 PLATF_F_48
Req_057	ICARUS platform should provide a listing with all the available data sources and related information and among others the terms of use for each one of them.	Collection Recommendation	AIA_008	PLATF_F_03 PLATF_F_51
Req_058	ICARUS platform should provide the monitoring, logging and auditing mechanisms in order for the stakeholders to be able to audit data usage and resolve any disputes	Sharing	AIA_014	PLATF_F_45
Req_059	ICARUS should provide a guideline or a guidance to create an appropriate license definition and agreement.	Sharing	AIA_010 CEL_008 ISI_006 ISI_007	PLATF_F_55 PLATF_F_56 PLATF_F_57
Req_060	ICARUS should provide the features for the management and update of data licenses.	Sharing	AIA_011 ISI_006	PLATF_F_58 PLATF_F_59
Req_061	ICARUS should support notifications regarding the result of the execution of scheduled analytics	Analytics	PACE_003 AIA_003 AIA_004 AIA_005	PLATF_F_50
Req_062	ICARUS should provide a GUI where the progress of processes and workflows can be monitored.	Analytics	PACE_003 AIA_003 AIA_004 AIA_005	PLATF_F_50
Req_063	ICARUS should have the ability to create edit and run what if scenarios.	Analytics	AIA_013 AIA_016	PLATF_F_29 PLATF_F_30 PLATF_F_40
Req_064	ICARUS platform should provide information about my data usage: which datasets, which algorithms, which reports.	Analytics	AIA_014	PLATF_F_45
Req_066	ICARUS should provide features for statistical analyses.	Analytics	AIA_015 AIA_016 AIA_017 CEL_003 CEL_005 CEL_009	PLATF_F_29 PLATF_F_30 PLATF_F_40
Req_072	ICARUS should provide the last modification time of each dataset.	Collection	CEL_010	PLATF_F_01 PLATF_F_02 PLATF_F_48
Req_075	ICARUS should be able to anonymise data so that legal regulations can be considered.	Collection	CEL_007 ISI_001	PLATF_F_08 PLATF_F_09 PLATF_F_10

Req_076	ICARUS should be able to assign costs to data assets.	Sharing	CEL_008	PLATF_F_56 PLATF_F_57
Req_077	ICARUS should provide features for different forms of payments.	Sharing	CEL_008	PLATF_F_56 PLATF_F_57
Req_078	ICARUS should provide the functionality to save and restore user defined configurations for data analysis.	Analytics	PACE_002 CEL_009	PLATF_F_43
Req_080	ICARUS platform should be able to perform aggregations on data sets.	Collection	ISI_001	PLATF_F_14
Req_082	ICARUS platform should be able to check that the data usage and delivery is compliant to the defined data access rights.	Analytics	ISI_002	PLATF_F_32
Req_083	ICARUS platform should provide data cleaning mechanisms.	Collection	PACE_018 ISI_003	PLATF_F_15
Req_084	ICARUS platform should provide mechanisms for anonymisation and data cleaning	Collection	PACE_018 ISI_003	PLATF_F_10 PLATF_F_15
Req_085	The platform should be able to integrate data sets based on common fields.	Linking	ISI_001	PLATF_F_18
Req_086	ICARUS should support API for data export	Collection	PACE_001 CEL_003	PLATF_F_01 PLATF_F_02
Req_087	ICARUS should support the provision of data updates.	Collection	PACE_016	PLATF_F_01 PLATF_F_02

3.1.2 Demonstrator Functional Requirements

As explained in section 2.3.2, the demonstrator functional requirements are depicting the expected functionalities of the ICARUS platform from the demonstrator partners in order to facilitate the execution of the applications that will be implemented by each demonstrator. It should be noted that these requirements are more data related in this version of the deliverable. However, as the project evolves and the demonstrators' execution scenarios will be further detailed and clearly formed, these requirements will evolve also.

The following table presents the elicited demonstrator functional requirements with the additional management information:

Table 3-2: Demonstrator Functional Requirements

ID	Description of the requirement	Category	Related User Stories	Feature ID
Req_013	ICARUS should be able to support search for historical flight information	Exploration	PACE_005 PACE_009	PLATF_F_22 PLATF_F_23
Req_014	ICARUS should be able to integrate flight information data with flight number, airline, date and time of departure/arrival.	Curation	PACE_009	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_015	ICARUS should be able to integrate airport weather data by period with airport identifier (IATA/ICAO code, airport/city name).	Curation	PACE_010 CEL_003	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_021	ICARUS should be able to integrate airport data with IATA/ICAO code, airport or city name	Curation	PACE_008	PLATF_F_11 PLATF_F_12 PLATF_F_13

ID	Description of the requirement	Category	Related User Stories	Feature ID
Req_025	ICARUS should be able to integrate obstacle data with runways and runway data with airports	Curation	PACE_013	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_026	ICARUS should be able to combine airport data, runway data and obstacle data and download them as a file	Curation	PACE_013	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_040	ICARUS should be able to support the upload of external airport, runway and obstacle data	Collection	PACE_021	PLATF_F_01 PLATF_F_02
Req_041	ICARUS should be able to support to upload external airport weather data	Collection	PACE_021	PLATF_F_01 PLATF_F_02
Req_048	ICARUS should be able to integrate flight information data with internal AIA airport data	Curation	AIA_003	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_052	ICARUS should allow to use look up tables (SXF is an IATA code for Berlin Schoenefeld Airport)	Analytics	AIA_012	PLATF_F_16
Req_065	ICARUS should be able to integrate aircraft on ground data with ICAO category	Analytics	AIA_015	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_067	ICARUS should provide means retrieve data from Amadeus Airport Operational Database (AODB)	Analytics	AIA_017	PLATF_F_01
Req_068	ICARUS should be able to integrate flight number with airport data like check in counter, luggage information	Collection	CEL_001	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_069	ICARUS should be able to integrate flight number with security process information like passport control or security scan.	Collection	CEL_002	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_070	IARUS should be able to integrate airport locations with weather data, current weather as well as statistical weather	Collection	CEL_003	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_071	ICARUS should be able to integrate flight number with airport information and flight plan data (e. g. delays)	Collection	CEL_004	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_073	ICARUS should be able to integrate flight data with connection flight information.	Collection	CEL_005	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_074	ICARUS platform should be able to integrate personal booking data with other data like fight information data, airport data, etc..	Collection	CEL_007	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_079	ICARUS platform should be able to integrate route information with passenger information.	Collection	ISI_001	PLATF_F_11 PLATF_F_12 PLATF_F_13
Req_081	ICARUS platform should be able to integrate passenger data with booking data	Collection	ISI_002	PLATF_F_11 PLATF_F_12 PLATF_F_13

3.2 Non-functional Requirements

The following table presents the elicited non-functional requirements with the additional management information:

Table 3-3: Non-functional Requirements

ID	Description of the requirement	Category	Related User Stories	Feature ID
Req_008	ICARUS platform should have high availability	Reliability	PACE_004	
Req_044	ICARUS should support know-your-customer practices, with organization registration and user login with credentials	Security	PACE_023 AIA_002	PLATF_F_46
Req_088	ICARUS should support an extended list of algorithms on a mixture of confidential and public data in order to perform big data analytics	Functional Suitability	PACE_006 PACE_020 PACE_023 AIA_002 ISI_002	PLATF_F_28 PLATF_F_31 PLATF_F_32 PLATF_F_36
Req_089	ICARUS should be able to execute big data analytics in a timely and efficient manner	Performance efficiency	PACE_004 PACE_010 AIA_017 ISI_005	PLATF_F_35
Req_090	ICARUS should guarantee the efficient and effective resource allocation for the success analytics jobs execution	Performance efficiency	AIA_017 CEL_004	PLATF_F_35
Req_091	ICARUS should be able to handle and store large datasets	Performance efficiency	PACE_005 PACE_010 ISI_002	
Req_092	ICARUS should enable the interconnection and exchange of information with other platforms or devices with appropriate secure mechanisms (e.g. REST API)	Compatibility	PACE_001 PACE_011 PACE_017 PACE_023 AIA_002	PLATF_F_01 PLATF_F_02
Req_093	ICARUS should be able to support the functional and flexible operation in a distributed cloud infrastructure	Compatibility		
Req_094	ICARUS should be able to consume and handle different datasets in various formats (e.g. CSV, JSON, XML files)	Compatibility	PACE_011 PACE_014 PACE_021 CEL_008	PLATF_F_02 PLATF_F_14 PLATF_F_41 PLATF_F_42
Req_095	ICARUS should provide an easy-to-use and user-friendly interface in which the analytics and visualisation processes are supported by guides and manuals	Usability	AIA_006 AIA_015 AIA_016 AIA_017 CEL_009 ISI_007	PLATF_F_07 PLATF_F_56
Req_096	ICARUS should provide a user interface that supports straightforward task accomplishment	Usability	PACE_002	PLATF_F_29 PLATF_F_30
Req_097	ICARUS should provide easy navigation through the platform features with support of dashboards or wizard/guide	Usability	PACE_002 AIA_010 AIA_013 ISI_007	PLATF_F_31 PLATF_F_32

ID	Description of the requirement	Category	Related User Stories	Feature ID
Req_098	ICARUS should provide the suitable error protection methods for all input fields	Usability	PACE_018 ISI_003	
Req_099	ICARUS should enable the secure storage of assets (datasets, reports, etc.)	Reliability	PACE_020	PLATF_F_46 PLATF_F_47
Req_100	ICARUS should be able to handle simultaneous requests on a timely and efficient manner	Reliability	PACE_004 CEL_004	
Req_101	ICARUS should provide the mechanisms to recover after system failure conditions	Reliability		
Req_102	ICARUS should be able to handle software errors without affecting the platform overall functionality	Reliability		
Req_103	ICARUS should ensure different authorisation access to different datasets	Security	PACE_003 PACE_006 PACE_017 AIA_002 ISI_001	PLATF_F_04 PLATF_F_36 PLATF_F_47
Req_104	ICARUS should provide the appropriate logging mechanisms for all operations	Security	PACE_017 AIA_014	PLATF_F_26
Req_105	ICARUS should be able to verify the identity of the user/subject performing any operation	Security	PACE_017 AIA_014	
Req_106	ICARUS should be able to trace all user/subject operations	Security	PACE_017 AIA_014	
Req_107	ICARUS should be composed by components that are operating independently	Maintainability		
Req_108	ICARUS should provide the tools that support enhanced system monitoring and debugging	Maintainability		
Req_109	ICARUS should provide a sophisticated alarm mechanism to identify failures or deficiencies	Maintainability	PACE_018 ISI_003	
Req_110	ICARUS should provide the proper mechanisms for system upgrade with minimum downtime	Maintainability	PACE_004	
Req_111	ICARUS should offer easy installation process in a timely manner	Portability		
Req_112	ICARUS should support deployment on various Linux distributions	Portability		
Req_113	ICARUS should be composed by independent components that are replaceable with minimum impact and effort	Portability		

4 ICARUS Technical Requirements

The previous section provided detailed descriptions of the functional, both platform- and demonstrator-specific, and non-functional requirements that emerged during the requirement elicitation process. As a next step, technical requirements have been extracted from each of the aforementioned requirements individually and have subsequently been grouped into 72 requirements, as shown in section 4.1. These technical requirements will be leveraged in order to draft the first conceptual ICARUS architecture, which will be presented in section 5.

4.1 List of technical requirements

For each of the identified technical requirements, the following table provides:

- A high-level categorisation based on the phases of the ICARUS methodology presented in D1.2, where applicable
- A more fine-grained categorisation based on the requirement categories provided in Section 3
- A unique ID
- A description
- The relevant requirements from which it was extracted or more generally its origin, in case this is not limited to the reported requirements in the previous section.

For the description, the following guidelines proposed in the ISO/IEC/IEEE 29148:2011 are adapted and followed:

- Requirements that constitute mandatory binding provisions use ‘shall’.
- Statements of fact, futurity, or a declaration of purpose are non-mandatory, non-binding provisions and use ‘will’. ‘Will’ can also be used to establish context or limitations of use. However, ‘will’ can be construed as legally binding, so it is best to avoid using it for requirements.
- Preferences or goals are desired, non-mandatory, non-binding provisions and use 'should’.
- Suggestions or allowances are non-mandatory, non-binding provisions and use 'may'.
- Non-requirements, such as descriptive text, use verbs such as ‘are’, ‘is’ and ‘was’. It is best to avoid using the term ‘must', due to potential misinterpretation as a requirement.
- Positive statements are to be used, whereas negative requirements (such as ‘shall not’) are to be avoided.
- Usage of active voice is preferred, whereas passive voice is to be avoided.

Following the above conventions, the phrasing of each requirement also serves as an implicit prioritisation. The selection of the most appropriate phrasing for each requirement incorporates both the importance of the relevant functional and non-functional requirements (as outlined during the requirement elicitation stage), as well as an early technical feasibility assessment. However, as can be

seen in the following table, the majority of identified technical requirements is in the form of mandatory binding provisions.

Table 4-1: Technical requirements

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
Data Collection			
Import & Export	TR_001	The ICARUS platform shall allow data to be imported from external sources.	Req_002, Req_028, Req_032, Req_055, Req_056, Req_067
	TR_002	The ICARUS platform shall allow the user to upload and download files.	Req_022, Req_026, Req_040, Req_041
	TR_003	The ICARUS platform should offer a simplified data check-in process for data that the providers intend to keep for personal usage only.	Req_042
	TR_004	The ICARUS platform should allow the user to save datasets that are currently in a private analytics space on the central platform storage.	Req_045
	TR_005	The ICARUS platform shall offer a well-defined API for data export.	Req_086
	TR_006	The ICARUS platform shall support updating and maintaining uploaded datasets.	Req_087
	TR_007	The ICARUS platform should allow the user to choose in which format to download data, when a transformation service is available.	Req_024, Req_094
	TR_008	The ICARUS platform should provide a service that transforms data from a format to another for selected predefined data formats.	Req_023, Req_094
	TR_009	The import and export mechanisms of the ICARUS platform should support large files.	Req_091
	TR_010	The ICARUS platform should be able to consume data from external RESTful APIs.	Req_056, Req_092
	TR_011	The ICARUS platform should support end-to-end data encryption.	From external stakeholders during MVP validation interviews
	TR_012	The ICARUS platform should support all data types described in the data requirements reported in D1.1	D1.1 (also relevant to Req_013, Req_014, Req_015, Req_021, Req_025, Req_048, Req_065,

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
			Req_068, Req_069, Req_070, Req_071, Req_073, Req_074, Req_079, Req_081)
	TR_013	The ICARUS platform should allow users to choose which field types in their datasets will be encrypted.	Stemming from TR_011
Data Cleansing	TR_014	The ICARUS platform should provide data cleansing functionalities.	Req_037, Req_083, Req_084
Data Anonymisation	TR_015	The ICARUS platform should provide a data anonymisation tool / service.	Req_039, Req_075, Req_084
Data Enrichment			
Data Representation, Semantics & Metadata	TR_016	The ICARUS platform shall comply with a common underlying metadata schema	Requirement coming from and clarified in D2.1
	TR_017	The ICARUS platform shall comply with a common underlying data model	Req_002, Req_028, Req_029, Req_032, Req_055
	TR_018	The ICARUS platform shall ensure that external data being imported in ICARUS are mapped to the ICARUS data model (in a semi-automatic manner).	Req_002, Req_028, Req_029, Req_032, Req_055
	TR_019	The ICARUS platform should provide the ability to data providers to assign predefined and/or custom tags (keywords) to their datasets.	Req_005, Req_010
	TR_020	The ICARUS platform shall offer a service that enriches uploaded data based on information from certain predefined controlled vocabularies (e.g. airport codes).	Req_030, Req_052
	TR_021	The ICARUS platform shall enable the users to assign IPR related attributes to the datasets.	Req_038
	TR_022	The ICARUS platform should provide predefined data license templates	Req_034
	TR_023	The ICARUS platform should allow data providers to customise the provided data license templates.	Req_034, Req_060
	TR_024	The ICARUS platform shall allow the user to define and configure a custom data license.	Req_034, Req_060
	TR_025	The ICARUS platform should store and show in an intuitive manner provenance-related information, e.g. when a dataset was last modified.	Req_072
	TR_026	The ICARUS platform shall offer an interactive UI to let the user browse the ICARUS data model.	Req_031, Req_046, Req_047

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
	TR_027	The ICARUS platform should support a model lifecycle management service that enables the user to recommend extensions to the data model.	Req_031, Req_046, Req_047
	TR_028	The ICARUS platform should support a process / service to enable the ICARUS administrator to review the data model recommendations and approve or decline them.	Req_031, Req_046, Req_047
Asset Exploration and Extraction			
Search	TR_029	The ICARUS platform shall support search functionality over the datasets to allow the user to find datasets by type, keyword, date, time.	Req_006, Req_009, Req_010, Req_011, Req_012
	TR_030	The ICARUS platform should save the query history of the user and allow the user to review it.	Req_020
	TR_031	The ICARUS platform shall retrieve and show the datasets that are relevant to a dataset that is returned as a query result.	Req_024, Req_047
	TR_032	The ICARUS platform should provide a mechanism for identifying connections among datasets based on their mapping to the common underlying data schema/model.	Req_043
	TR_033	The ICARUS platform should allow for spatiotemporal information to be un-encrypted in the datasets so that search queries can be performed on it.	Req_012
Data Sharing	TR_034	The ICARUS platform shall provide an information catalogue about all datasets that are open or available for sharing (by their respective data providers).	Req_057
	TR_035	The ICARUS platform shall enable the creation of data sharing contracts with detailed terms in an immutable manner.	Req_059
	TR_036	The ICARUS shall provide walkthroughs and guidelines regarding the creation of data sharing contracts.	Req_059
	TR_037	The ICARUS platform shall allow users to set pricing terms for their datasets.	Req_076
	TR_038	The ICARUS platform should support various payment methods.	Req_077
	TR_039	The ICARUS platform should provide a mechanism for data providers and data consumers to negotiate prior to signing the data sharing contract.	Req_036
	TR_040	The ICARUS platform may allow existing, active data contracts (traditionally signed by a data provider) to be facilitated / executed by the platform.	From external stakeholders during MVP validation interviews
	TR_041	The ICARUS platform shall allow users to request to purchase and to access datasets not owned by them	Requirement coming from and clarified in D2.2
	TR_042	The ICARUS platform shall allow users to accept or deny requests for access on their datasets made by other users	Requirement coming from and clarified in D2.2

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
	TR_043	The ICARUS platform shall store the data sharing contracts in a DLT-based repository for non-repudiation purposes.	From external stakeholders during MVP validation interviews
Data Analysis & Visualisation			
Analysis & Visualisation	TR_044	The ICARUS platform should provide a UI that allows the user to define, configure, review and manage data analysis jobs and save configurations for later re-usage.	Req_003, Req_055, Req_078
	TR_045	The ICARUS platform shall enable the integration and combined analysis over multiple datasets.	Req_007
	TR_046	The ICARUS platform should allow the easy configuration and application of advanced data analysis algorithms.	Req_018, Req_019, Req_088
	TR_047	The ICARUS platform shall enable the application of predefined data analysis algorithms on datasets.	Req_019
	TR_048	The ICARUS platform should support the combination (merging) of datasets based on common fields into one dataset.	Req_027, Req_026, Req_085
	TR_049	The ICARUS platform should provide a monitoring UI for the progress and status of data analysis jobs.	Req_062
	TR_050	The ICARUS platform should provide tools/services to define and execute what-if scenarios on the datasets.	Req_063
	TR_051	The ICARUS platform shall provide tools and services to apply machine learning algorithms	Requirement coming from and clarified in D2.2
	TR_052	The ICARUS platform should provide tools and services to apply deep learning algorithms	Requirement coming from and clarified in D2.2
	TR_053	The ICARUS platform shall provide tools and services to apply basic analytics	Requirement coming from and clarified in D2.2
	TR_054	The ICARUS platform should provide tools and services that enable users to perform statistical analysis over datasets	Req_066
	TR_055	The ICARUS platform should offer data management methods and algorithms that handle both structured and unstructured data.	Req_050
	TR_056	The ICARUS platform shall offer data visualisation tools/ functionalities.	Req_051, Req_053
	TR_057	The ICARUS platform shall enable the users to define and schedule data analysis jobs.	Req_054
	TR_058	The ICARUS platform should enable the users to define, configure and schedule data management and processing recipes	From MVP
	TR_059	The ICARUS platform should allow a user to easily perform aggregations on a dataset.	Req_080
	TR_060	The ICARUS platform shall support analytics jobs in a scalable and reliable manner	Req_089

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
	TR_061	The ICARUS platform should provide tools and services to perform resource allocation for data analysis purposes.	Req_090
Added value services and platform features			
Notifications	TR_062	The ICARUS platform shall allow users to manage their notification preferences.	Req_001, Req_033, Req_049
	TR_063	The ICARUS platform should inform users with active contracts on a dataset that the dataset has been updated.	Req_001, Req_033
	TR_064	The ICARUS platform should provide notifications to inform users when their scheduled analytics jobs finish.	Req_049, Req_061
Usage Analytics	TR_065	The ICARUS platform should provide data usage analytics to the users for the datasets they own.	Req_035, Req_064
Security and Privacy	TR_066	The ICARUS platform shall provide public, private and confidential working spaces.	Req_004
	TR_067	The ICARUS platform shall ensure that access control over datasets is applied according to the data provider's policies and the terms of relevant active valid data sharing contracts.	Req_016, Req_082
	TR_068	The ICARUS platform shall forbid unauthorised user access to the platform and the datasets.	Req_044
	TR_069	The ICARUS platform storage shall be secure.	Req_099
	TR_070	The ICARUS platform should ensure different authorisation levels for accessing datasets.	Req_017, Req_088, Req_103
	TR_071	The ICARUS platform should be able to verify the identity of the user/subject performing any operation in the platform.	Req_105
	TR_072	The ICARUS platform shall provide a secure and controlled registration process for new users	Requirement coming from and clarified in D2.2

4.2 Mapping of technical requirements and ICARUS stakeholders

The work reported in the current deliverable regarding requirements elicitation cannot be viewed independently from the extracted MVP features that were reported in D1.2. In fact, there are strong links between the extracted technical requirements described above and the identified MVP features. This was both expected and desired, as it constitutes a confirmation of the requirements' validity. The following table shows how the MVP features map to the technical requirements.

Table 4-2: Mapping of MVP features to Technical Requirements

MVP Features	Technical Requirements
PLATF_F_01, PLATF_F_02, PLATF_F_11, PLATF_F_13, PLATF_F_14, PLATF_F_27, PLATF_F_41	TR_001, TR_002, TR_003, TR_004, TR_005, TR_006, TR_007, TR_008, TR_009, TR_010, TR_011, TR_012, TR_013
PLATF_F_15	TR_014
PLATF_F_09	TR_015

MVP Features	Technical Requirements
PLATF_F_20, PLATF_F_21, PLATF_F_56	TR_016, TR_017, TR_018, TR_019, TR_020, TR_021, TR_022, TR_023, TR_024, TR_025, TR_026, TR_027, TR_028
PLATF_F_18, PLATF_F_22, PLATF_F_25, PLATF_F_26, PLATF_F_51	TR_029, TR_030, TR_031, TR_032, TR_033
PLATF_F_57, PLATF_F_60, PLATF_F_61	TR_034, TR_035, TR_036, TR_037, TR_038, TR_039, TR_040, TR_041, TR_042, TR_043
PLATF_F_14, PLATF_F_35, PLATF_F_38, PLATF_F_43, PLATF_F_44, PLATF_F_39, PLATF_F_40	TR_044, TR_045, TR_046, TR_047, TR_048, TR_049, TR_050, TR_051, TR_052, TR_053, TR_054, TR_055, TR_056, TR_057, TR_058, TR_059, TR_060, TR_061
PLATF_F_48, PLATF_F_49	TR_062, TR_063, TR_064
PLATF_F_45	TR_065

It should be noted that there are four MVP features defined and described in D1.2 which do not appear in the previous table, as they are not covered by the extracted technical requirements. Specifically:

PLATF_F_06: (Semi-)Automatic quality check of the data and assessment of quality level.

Ensuring data quality remains a valid and also intuitive requirement for the ICARUS offering and has emerged in the context of the functional requirements reported in section 3 (e.g. Req_037). However, the way this will be addressed needs to be further examined in terms both of the algorithmic/ methodological approach that is more suitable for ICARUS and of the technical feasibility of the solution that will be selected. Hence, the relevant technical requirements will be part of the next requirements extraction phase (to be reported in D3.3).

PLATF_F_31: Automatic check whether the data asset is appropriate for a specific algorithm.

Depending on the sophistication level of the solution that will be followed in ICARUS to implement this feature, the appropriate technical requirements will be extracted in the next phase and reported in D3.3. The dependency of this feature on other data analysis related features does not allow the definition of concrete requirements, until the design (and possibly implementation) of those has progressed.

PLATF_F_32: Automatic check for data licenses compatibility to run under a specific algorithm.

The explanation provided for PLATF_F_31 above is also valid here.

PLATF_F_55: Automatic license compatibility check for data assets that build on other datasets.

The explanation provided for PLATF_F_31 and PLATF_F_32 above is also valid here. Moreover, the complexity of the underlying problem, which has been discussed in D2.2, requires further studying and exploration in order to achieve a level of specificity that will allow the extraction of technical requirements.

5 ICARUS Platform Architecture

5.1 Conceptual architecture

The conceptual architecture of ICARUS has been designed by conducting a thorough analysis of the technical requirements documented in section 4 that were later translated into technological, beyond the state of the art, software modules that will be implemented in the context of WP4. The conceptual architecture of ICARUS is supporting the smooth and effective integration of the several software modules that will be implemented with the aim of maximising the benefits of combining multiple technologies and tools from different partners and organisations. During the design process, concerns and decisions were weighted, and the stakeholder requirements were constantly validated against the design.

The ICARUS architecture is a modular architecture that provides enhanced flexibility in order to adapt and connect the various components that will be implemented as software modules. The major focus was on the functional decomposition, the strict separation of concerns, the dependencies identification and especially the data flow. As such, each component has been designed with the aim of delivering specific business services with a clear context, scope and set of features. Components were assigned to the different technical partners that were involved in the analysis of the requirements, the shaping of the conceptual architecture and the design of the individual components. The technical requirements and the functional specifications were carefully analysed and facilitated the evolution of a mature concept architecture design that is aiming to address the ambition of ICARUS to deliver a novel big data platform for the aviation data value chain.

The main challenge of the ICARUS architecture is to provide a scalable and flexible environment that will enable the interoperability of the various components that facilitate the execution of big data analytics and sharing of data through secure, transparent and advanced functionalities and features. To achieve this, all components of the ICARUS architecture will provide well-defined interfaces to ensure the seamless integration and operation of the integrated platform.

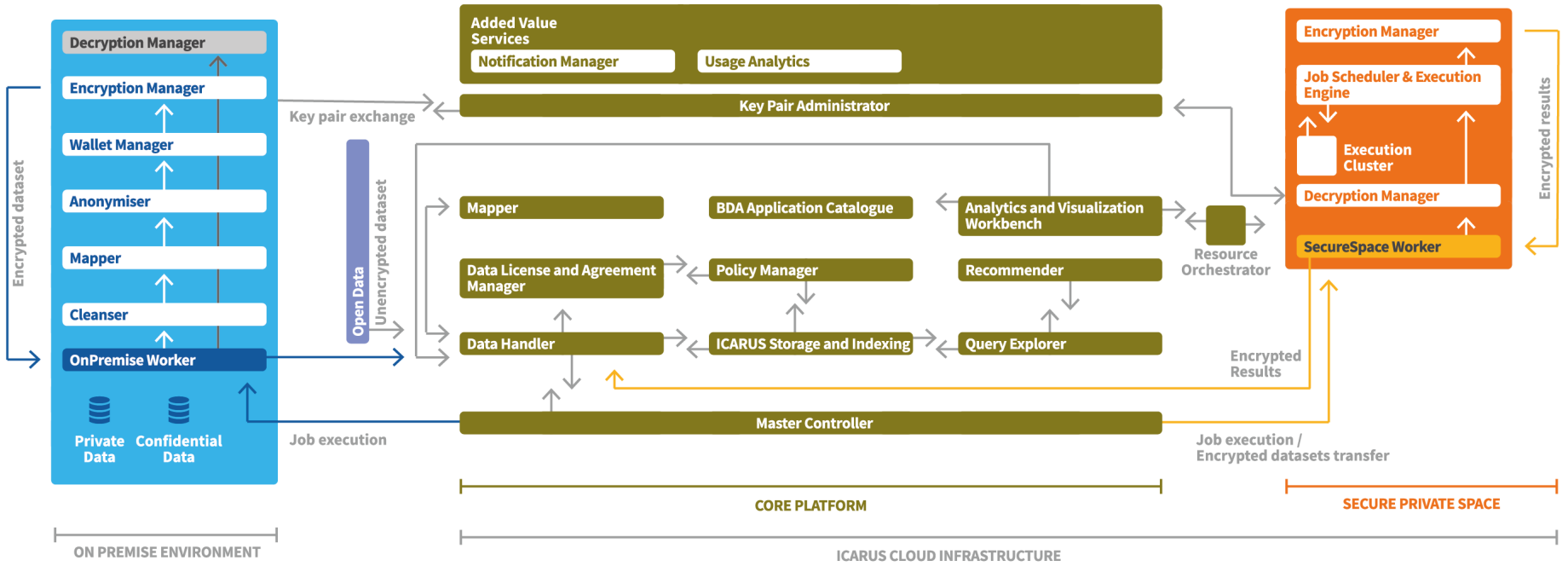

Figure 5-1: ICARUS conceptual architecture

The ICARUS architecture is conceptually divided in three main tiers, the **On Premise Environment**, the **Core ICARUS platform** and the **Secure and Private Space**. Each tier is undertaking a set of functionalities of the ICARUS platform depending on the execution environment and context. The ICARUS conceptual architecture is illustrated in Figure 5-1.

The **On Premise Environment** is composed by multiple components running on the data provider's environment with the main purpose to prepare the data provider's private or confidential datasets in order to be uploaded in the ICARUS platform. The On Premise Environment performs the tasks according to the instructions provided by the Core ICARUS platform. To achieve this, the Master / Worker paradigm is utilised. More specifically, the **OnPremise Worker** running on the On Premise Environment receives a set of instructions from the Master Controller running on the Core ICARUS platform in order to perform a set of tasks. The OnPremise Worker is responsible for completing these tasks by utilising the set of components running on the On Premise Environment for each specific task.

At first, the **Cleanser** provides the data cleansing functionalities of the platform. The Cleanser supports a set of techniques for performing simple and more advanced cleansing operations over datasets that contain erroneous or "dirty" data by detecting or correcting corrupted, incomplete, incorrect and inaccurate records from datasets with a variety of rules and constraints. The **Mapper** is responsible for the harmonisation process of the dataset by enabling the user to define the mapping of the fields of the dataset to the ICARUS common aviation model in a semi-automatic way. Moreover, the Mapper enables the exploration of the ICARUS common aviation model from the user in order to provide suggestions for possible extensions of the model. The **Anonymiser** is providing the data anonymisation functionalities in order to filter or hide the private, sensitive or personal data that cannot be disclosed outside the data provider's premises, corporate network or personal filesystem by providing the means to deal with privacy issues and protection of sensitive information with a variety of techniques such as data masking, data encryption and data scrambling.

The **Wallet Manager** is facilitating the operation of the Data License and Agreement Manager by providing a set of functionalities such as the generation and management of the blockchain account of the user and the interaction with the blockchain for the smart contract signature process. The **Encryption Manager** is undertaking all essential encryption processes with regard to encryption of the data provider's dataset. It provides the encryption cypher mechanism that generates the encryption key and the encrypted dataset, while also facilitating the dataset sharing, upon the agreement of the data provider and the data consumer, with the generation of the appropriate decryption keys, the storage and management of the generated decryption keys and the secure transmission of the corresponding decryption key from the data provider to the data consumer. The **Decryption Manager** is enabling the decryption of the dataset on the On Premise Environment when an encrypted dataset is downloaded locally, provided that a valid smart contract exists permitting the downloading of the specific dataset locally. The Decryption Manager provides the mechanisms to verify the identity of the data consumer via a certificate or public key, to request for the decryption key from the data provider

and the decryption mechanism in order to temporarily reproduce the encryption key in order to decrypt the dataset.

The **Core ICARUS platform** is composed by multiple interconnected components running on the ICARUS infrastructure. The **Master Controller** is responsible for compiling and providing a set of instructions to be executed by the OnPremise Worker and the SecureSpace Worker following the Master / Worker paradigm as explained also above. The Master Controller will submit the set of instructions in the form of jobs that will be executed by the corresponding workers running on the On Premise Environment and the Secure and Private Space and both workers will execute the requested jobs with the usage of the set of components that are running on both environments.

The **Data Handler** is responsible for receiving the incoming data provider's private or confidential dataset as produced within the On Premise Environment. Additionally, the Data Handler is facilitating the download and upload of datasets from the open data sources by providing the necessary mechanisms to connect and retrieve the corresponding datasets based on the provided configuration. For the datasets originating from open data sources the Data Handler utilises the local running instance of the **Mapper** in order to perform the harmonisation process for these datasets in the same manner as it performed on the On Premise Environment for the private and confidential datasets. The **Data License and Agreement Manager** is implementing the blockchain functionalities of the ICARUS platform which hosts a local blockchain node. Moreover, it provides the processes needed for the preparation of a smart contract that is uploaded in the blockchain and it is activated when both parties (data owner and data consumer) come to an agreement and the corresponding payment is completed.

The **Key Pair Administrator** is performing the signalling operations for the exchange of the decryption key between the data provider and the data consumer. In addition to this, the Key Pair Administrator is responsible for maintaining the list of data providers in order to support the signalling operations for the key exchange, as well as to perform and support the revocation process of any key when needed. The **Policy Manager** is implementing the access control mechanism of the ICARUS platform that is based on the ABAC model and XACML standard. To meet its goals, it provides the suitable methods for the creation, management and deletion of access control policies. The Policy Manager and Controller is intercepting all requests for resource access in order to form an access control decision. The **ICARUS Storage and Indexing** component is providing the storage capabilities of the ICARUS platform. It consists of big-data enabled storage solutions, capable of storing and managing large amount of data in structured or unstructured format. As with any big data ecosystem, the storage solution provides a set of key characteristics such as horizontal scalability, high availability, high performance and advanced security. Additionally, it provides the indexing capabilities of the platform over multiple complex datasets with flexibility and efficiency.

The **Query Explorer** encapsulates the intuitive environment that facilitates a data query definition with enhanced functionalities such as dynamic field selection and filter definition. The Query Explorer is responsible for the query transformation to the underlying query language of the respective storage solution. It enables the query execution and the display of the results in a user-friendly way, while also

supporting the retrieval and display of the proper recommendations through the Recommender. The **Recommender** is providing the enhanced recommendation functionalities that enable the dataset exploration and discoverability. The Recommender provides recommendations and suggestions for additional related datasets that can be explored or utilised during the search and query process.

The **Analytics and Visualisation Workbench** is providing the environment where the users of the platform are able to design, execute and monitor the data analytics workflows and also where the visualisation and dashboards are displayed. The users are able to select an algorithm from the extended list of supported algorithms and set the corresponding parameters according to their needs. While the design of the data analytics workflow is performed in the Analytics and Visualisation Workbench, the execution of the analysis is performed within the Secure and Private Space with the use of the Master Controller and the SecureSpace Worker. Furthermore, through the Analytics and Visualisation Workbench the advanced visualisation capabilities of the platform are offered with a variety of visualisations that can be combined in order to form dynamic dashboards upon the user needs. Additionally, the user is able to create an ICARUS application, which contains the list of datasets that were selected for analysis, as well as the algorithm along with the corresponding parameters, and store it in the **BDA Application Catalogue**. The BDA Application Catalogue implements a repository of the ICARUS applications created by the users of the platform. As such, the ICARUS applications can be stored, retrieved, modified and loaded in the Analytics and Visualisation Workbench by the users at any time. The purpose of the BDA Application Catalogue is to enable the reuse of the designed data analytics workflows from the users, as well as the sharing of these workflows among the users through a defined license in order to empower the analytical capabilities of the platform.

In addition to the aforementioned components, the ICARUS platform is supported by supplementary components with the aim of providing added-value services to the users of the platform. The **Notifications Manager** is responsible for providing the updated information to the users with regards to the datasets or the scheduled analytics jobs. More specifically, the Notifications Manager provides notifications to the users related to the availability of new datasets according to their configured preferences or any possible updates on the datasets that the users are entitled to use, as well as any updates on the execution status of the scheduled analytics jobs. The **Usage Analytics** component is responsible for providing the tools that collect, analyse and visualise the usage of the various services and assets of the platform in order to extract useful insights and statistics. The Usage Analytics records the user's behaviour in various levels such as the usage and adoption of specific features or services and the usage of each dataset or algorithm towards the aim of providing usage information to both the users and the platform administrator.

The **Resource Orchestrator** is enabling the provisioning and management of the Secure and Private Space. More specifically, the Resource Orchestrator is able to connect to the virtualised infrastructure in order to perform monitoring and management of the available resources, to allocate and release the resource in the corresponding virtual machines, as well as deploy and manage the applications and services running on the virtual machines. Finally, the Resource Orchestrator is performing

enhanced service monitoring and health checks on the services or applications running on the virtual machines.

As such, the **Secure and Private Space** is realised in the form of dedicated virtual machines that are spawned on demand so that each user is able to perform analysis in an isolated and secure environment. The Secure and Private Space contains a set of interconnected components that constitute the advanced analytics execution environment of the ICARUS platform. The **SecureSpace Worker** running on the Secure and Private Space receives a set of instructions from the Master Controller running on the Core ICARUS platform in order to perform the specified jobs. The SecureSpace Worker undertakes the responsibility of executing the scheduled jobs with the use of a set of components running on the Secure and Private Space. At first, the decryption of the dataset on the data consumer side is handled by the **Decryption Manager**. As described above, the Decryption Manager performs the data consumer's identity verification, the request for the decryption key exchange and eventually the decryption of the encrypted dataset via the dedicated decryption mechanism on the Secure and Private Space. Once an analysis is triggered by the Analytics and Visualisation Workbench, the **Job Scheduler and Execution Engine** is responsible for the initiation and monitoring of the corresponding job and for providing the relevant status, as well as the analysis results, in the Analytics and Visualisation Workbench in order to be displayed to the users. The Job Scheduler and Execution Engine is interacting with the Execution Cluster for the job execution. The **Execution Cluster** is the cluster-computing framework of the platform offering the processing engine for the data analysis across multiple datasets with a set of key characteristics such as speed, efficiency, reliability, fault tolerance and effective distributed job execution. The processing engine supports the execution of a large list of big data analysis algorithms that spans from simple statistical analysis to machine learning algorithms. The results of the analysis are passed to the **Encryption Manager** in order to be encrypted before they are securely transmitted and stored in the Core ICARUS platform.

In the following subsection, the mapping of the technical requirements that were documented in section 4.1 and the components of the ICARUS platform architecture that was described in the current section, as a result of the thorough analysis of these requirements, is presented.

5.2 Mapping Technical Requirements to Components

In the previous section the ICARUS modular conceptual architecture was presented. This modular architecture is composed by multiple components, each one designed with a distinct role, scope and a set of core functionalities towards the aim of providing a novel big data platform for the aviation data value chain. As already described, the design of these components, as well as of the holistic platform architecture, was a result of the analysis of the technical requirements and multiple iterations in the design process in order to ensure that the technical requirements have been addressed. The following table summarises the mapping of these technical requirements with the designed components.

Table 5-1: Mapping of technical requirements to components

ID	Description of the requirement	Component
TR_001	The ICARUS platform shall allow data to be imported from external sources.	Data Handler
TR_002	The ICARUS platform shall allow the user to upload and download files.	Data Handler
TR_003	The ICARUS platform should offer a simplified data check-in process for data that the providers intend to keep for personal usage only.	Data Handler
TR_004	The ICARUS platform should allow the user to save datasets that are currently in a private analytics space on the central platform storage.	Data Handler, SecureSpace Worker
TR_005	The ICARUS platform shall offer a well-defined API for data export.	Data Handler
TR_006	The ICARUS platform shall support updating and maintaining uploaded datasets.	Data Handler
TR_007	The ICARUS platform should allow the user to choose in which format to download data, when a transformation service is available.	-
TR_008	The ICARUS platform should provide a service that transforms data from a format to another for selected predefined data formats.	-
TR_009	The import and export mechanisms of the ICARUS platform should support large files.	Data Handler, Storage and Indexing
TR_010	The ICARUS platform should be able to consume data from external RESTful APIs.	Data Handler
TR_011	The ICARUS platform should support end-to-end data encryption.	Encryption Manager, Decryption Manager, Key Pair Administrator
TR_012	The ICARUS platform should support all data types described in the data requirements reported in D1.1	Mapper, Storage and Indexing
TR_013	The ICARUS platform should allow users to choose which field types in their datasets will be encrypted.	Encryption Manager
TR_014	The ICARUS platform should provide data cleansing functionalities.	Cleanser
TR_015	The ICARUS platform should provide a data anonymisation tool / service.	Anonymiser
TR_016	The ICARUS platform shall comply with a common underlying metadata schema	Query Explorer
TR_017	The ICARUS platform shall comply with a common underlying data model	Query Explorer, Mapper
TR_018	The ICARUS platform shall ensure that external data being imported in ICARUS are mapped to the ICARUS data model (in a semi-automatic manner).	Data Handler, Mapper
TR_019	The ICARUS platform should provide the ability to data providers to assign predefined and/or custom tags (keywords) to their datasets.	Data Handler
TR_020	The ICARUS platform shall offer a service that enriches uploaded data based on information from certain predefined controlled vocabularies (e.g. airport codes).	Mapper
TR_021	The ICARUS platform shall enable the users to assign IPR related attributes to the datasets.	Data Handler, Data License and Agreement Manager
TR_022	The ICARUS platform should provide predefined data license templates	Data License and Agreement Manager
TR_023	The ICARUS platform should allow data providers to customise the provided data license templates.	Data License and Agreement Manager

ID	Description of the requirement	Component
TR_024	The ICARUS platform shall allow the user to define and configure a custom data license.	Data License and Agreement Manager
TR_025	The ICARUS platform should store and show in an intuitive manner provenance-related information, e.g. when a dataset was last modified.	Data Handler
TR_026	The ICARUS platform shall offer an interactive UI to let the user browse the ICARUS data model.	Mapper
TR_027	The ICARUS platform should support a model lifecycle management service that enables the user to recommend extensions to the data model.	Mapper
TR_028	The ICARUS platform should support a process / service to enable the ICARUS administrator to review the data model recommendations and approve or decline them.	Mapper
TR_029	The ICARUS platform shall support search functionality over the datasets to allow the user to find datasets by type, keyword, date, time.	Query Explorer
TR_030	The ICARUS platform should save the query history of the user and allow the user to review it.	Query Explorer
TR_031	The ICARUS platform shall retrieve and show the datasets that are relevant to a dataset that is returned as a query result.	Query Explorer, Recommender
TR_032	The ICARUS platform should provide a mechanism for identifying connections among datasets based on their mapping to the common underlying data schema/model.	Query Explorer, Mapper
TR_033	The ICARUS platform should allow for spatiotemporal information to be un-encrypted in the datasets so that search queries can be performed on it.	Query Explorer, Encryption Manager
TR_034	The ICARUS platform shall provide an information catalogue about all datasets that are open or available for sharing (by their respective data providers).	Query Explorer
TR_035	The ICARUS platform shall enable the creation of data sharing contracts with detailed terms in an immutable manner.	Data License and Agreement Manager, Wallet Manager
TR_036	The ICARUS shall provide walkthroughs and guidelines regarding the creation of data sharing contracts.	Data License and Agreement Manager
TR_037	The ICARUS platform shall allow users to set pricing terms for their datasets.	Data License and Agreement Manager
TR_038	The ICARUS platform should support various payment methods.	-
TR_039	The ICARUS platform should provide a mechanism for data providers and data consumers to negotiate prior to signing the data sharing contract.	Data License and Agreement Manager
TR_040	The ICARUS platform may allow existing, active data contracts (traditionally signed by a data provider) to be facilitated / executed by the platform.	Data License and Agreement Manager, Wallet Manager
TR_041	The ICARUS platform shall allow users to request to purchase and to access datasets not owned by them	Data License and Agreement Manager, Policy Manager
TR_042	The ICARUS platform shall allow users to accept or deny requests for access on their datasets made by other users	Data License and Agreement Manager, Wallet Manager, Policy Manager
TR_043	The ICARUS platform shall store the data sharing contracts in a DLT-based repository for non-repudiation purposes.	Data License and Agreement Manager, Wallet Manager

ID	Description of the requirement	Component
TR_044	The ICARUS platform should provide a UI that allows the user to define, configure, review and manage data analysis jobs and save configurations for later re-usage.	Analytics and Visualisation Workbench, BDA Application Catalogue
TR_045	The ICARUS platform shall enable the integration and combined analysis over multiple datasets.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine, Mapper
TR_046	The ICARUS platform should allow the easy configuration and application of advanced data analysis algorithms.	Analytics and Visualisation Workbench
TR_047	The ICARUS platform shall enable the application of predefined data analysis algorithms on datasets.	Analytics and Visualisation Workbench, BDA Application Catalogue
TR_048	The ICARUS platform should support the combination (merging) of datasets based on common fields into one dataset.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine, Mapper
TR_049	The ICARUS platform should provide a monitoring UI for the progress and status of data analysis jobs.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_050	The ICARUS platform should provide tools/services to define and execute what-if scenarios on the datasets.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_051	The ICARUS platform shall provide tools and services to apply machine learning algorithms	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_052	The ICARUS platform should provide tools and services to apply deep learning algorithms	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_053	The ICARUS platform shall provide tools and services to apply basic analytics	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_054	The ICARUS platform should provide tools and services that enable users to perform statistical analysis over datasets	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_055	The ICARUS platform should offer data management methods and algorithms that handle both structured and unstructured data.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_056	The ICARUS platform shall offer data visualisation tools/ functionalities.	Analytics and Visualisation Workbench
TR_057	The ICARUS platform shall enable the users to define and schedule data analysis jobs.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_058	The ICARUS platform should enable the users to define, configure and schedule data management and processing recipes	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine, BDA Application Catalogue
TR_059	The ICARUS platform should allow a user to easily perform aggregations on a dataset.	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine
TR_060	The ICARUS platform shall support analytics jobs in a scalable and reliable manner	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine

ID	Description of the requirement	Component
TR_061	The ICARUS platform should provide tools and services to perform resource allocation for data analysis purposes.	Resource Orchestrator
TR_062	The ICARUS platform shall allow users to manage their notification preferences.	Notification Manager
TR_063	The ICARUS platform should inform users with active contracts on a dataset that the dataset has been updated.	Notification Manager
TR_064	The ICARUS platform should provide notifications to inform users when their scheduled analytics jobs finish.	Notification Manager
TR_065	The ICARUS platform should provide data usage analytics to the users for the datasets they own.	Usage Analytics
TR_066	The ICARUS platform shall provide public, private and confidential working spaces.	Resource Orchestrator
TR_067	The ICARUS platform shall ensure that access control over datasets is applied according to the data provider's policies and the terms of relevant active valid data sharing contracts.	Data License and Agreement Manager, Wallet Manager, Policy Manager
TR_068	The ICARUS platform shall forbid unauthorised user access to the platform and the datasets.	Data License and Agreement Manager, Wallet Manager, Policy Manager, Storage and Indexing
TR_069	The ICARUS platform storage shall be secure.	Storage and Indexing
TR_070	The ICARUS platform should ensure different authorisation levels for accessing datasets.	Policy Manager
TR_071	The ICARUS platform should be able to verify the identity of the user/subject performing any operation in the platform.	Policy Manager, Data License and Agreement Manager
TR_072	The ICARUS platform shall provide a secure and controlled registration process for new users	Policy Manager

It should be noted that there are three technical requirements from the list of requirements documented in section 4.1, that are not addressed by any of the designed components. Specifically:

TR_007: The ICARUS platform should allow the user to choose in which format to download data, when a transformation service is available.

Although the ICARUS platform will allow the downloading of datasets from the data consumer locally, if the license associated with the respective dataset permits this action, enabling the transformation of the stored datasets to multiple formats needs to be further examined in terms of the technical feasibility of the transformation of the datasets between various formats. The restrictions imposed by the nature and the structure of each possible dataset do not allow the definition of a solid process that can be selected until the design (and possibly implementation) of the components has progressed. Hence, the relevant technical requirement will be addressed in the upcoming versions of the architecture.

TR_008: The ICARUS platform should provide a service that transforms data from a format to another for selected predefined data formats.

The explanation provided for TR_008 above is also valid here.

TR_038: The ICARUS platform should support various payment methods.

While the ICARUS platform will support the definition of license and pricing related metadata for the incorporated dataset, as well as the creation, negotiation and signing of smart contracts within the platform, the support for payment methods within the ICARUS platform or with the integration of a third-party payment system is a rather complex problem that requires studying and exploration before the appropriate solution is selected. Hence, the relevant technical requirement will be properly addressed in the upcoming versions of the architecture.

In the following subsections, the design and functionalities of the individual components, as well as the addressed requirements by each component, are described.

5.3 ICARUS Components

5.3.1 Anonymiser

5.3.1.1 Design and Functionalities overview

The Anonymiser is the component responsible for providing the data anonymisation functionalities of the ICARUS platform. Residing at the location of the datasets, the Anonymiser ensures that any kind of private, sensitive or personal information will not be disclosed outside the data provider's premises. Hence, the purpose of the Anonymiser is to deal with the various privacy concerns and legal limitations by employing a privacy and anonymisation toolset with various data anonymisation techniques that will filter the information according to the stakeholders' needs. The Anonymiser will be deployed within the On Premise Environment in order to be utilised within the data provider's premises.

In order to comply to the extent possible with the General Data Protection Regulation (GDPR) (Regulation (EU) 2016/679) and protect the data privacy several challenges arise in every big data and cloud ecosystem, the Anonymiser addresses the risk of unintended disclosure of personal or corporate information when data need to uploaded or shared on the ICARUS platform, in which sensitive variables (attributes) of the datasets must be handled properly in order to remove or hide the individual's identifying information in a way that the remaining information cannot be linked to the individual. However, during the anonymisation process that will be applied, a balance between data privacy and data utility should be considered in order to maintain the usability of the data.

The Anonymiser receives datasets from the data provider and provides an easy user interface in order to select, configure and customise the various models and methods according to the stakeholder's needs and required anonymisation level. The Anonymiser will apply the selected models and methods on the dataset, either on a field-level or on a dataset-level and will ensure that individual's identifying information cannot be re-identified. The results of the process are presented to the data provider in order to review the results and possibly perform fine tuning on the selected methods and techniques before the results are finalised. The data provider is responsible for the assessment of the results of the anonymisation process based on his deep understanding of his dataset in terms of the privacy

threat risks and the usability of the produced anonymised data. The Anonymiser will be able to receive and process datasets originating from a variety of data sources, including databases, local files or APIs in order to perform the desired anonymisation process.

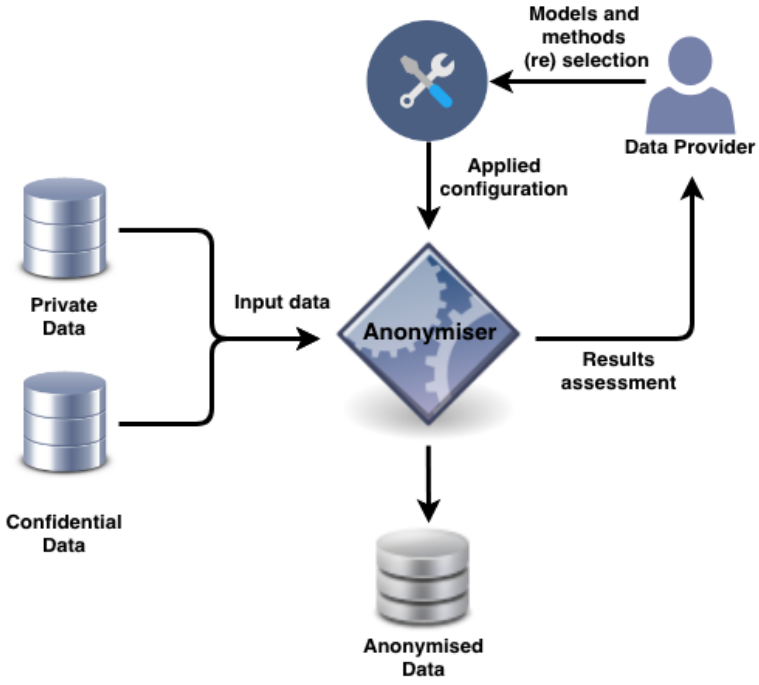

Figure 5-2: Anonymiser overview

The **main functionalities of the Anonymiser** are the following:

- Handle the various privacy concerns and sensitive information limitations to eliminate the unintended disclosure of personal or corporate information
- Employ a privacy and anonymisation toolset with a variety of privacy models such as the K-anonymity, the L-diversity and T-closeness models and anonymisation techniques such as the Generalisation, Aggregation, Suppression, Categorisation, Randomisation, Pseudo-anonymisation, and Data Perturbation.
- Handle and process datasets originating from various data sources and provide the anonymised datasets.

5.3.1.2 Addressed requirements

The Anonymiser with the set of functionalities described in the component's design addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_015:** The Anonymiser offers a toolbox capable of addressing the data anonymisation requirements of the ICARUS stakeholders with a variety of methods and techniques that can be applied on the selected dataset.

5.3.2 Cleanser

5.3.2.1 Design and Functionalities overview

The Cleanser is the component undertaking the responsibility of the data cleansing functionalities of the ICARUS platform with the aim of addressing a variety of data quality issues and maximising the usability of the data. The Cleanser is providing the assurance that the various datasets originating from several heterogeneous data sources are clean, accurate and complete according to the standards set by the data provider in order to increase their reusability in different contexts, as well as to enable the execution of high-quality data analysis.

In general, data cleansing is an umbrella term for tasks that focus on the assurance of the data integrity, data accuracy, data precision and data reliability. The list of tasks spans from simple corrective operations such as transformation, reformatting and default value substitution to more complex operations such outlier detection and replacement. The Cleanser provides all the required processes that will enable the detection and the accomplishment of various corrective (or even removal) actions on top of inaccurate or corrupted datasets that contain inaccurate, incomplete, incorrect or irrelevant data fields or field values, also known as “dirty” data, towards safeguarding the dataset reliability, reusability and accuracy. As such, the Cleanser is composed by four subcomponents, namely the **Data Validation**, the **Data Cleansing**, the **Data Completion** and the **Cleansing Logger**, in order to compile and execute the cleansing workflow on the selected datasets. Each subcomponent is responsible for providing the suitable operations towards the completion of the cleansing workflow.

The **Data Validation** subcomponent is safeguarding the cleanliness, correctness and usefulness of the dataset by performing the appropriate validation checks against a list of predefined rules and constraints in order to identify all the conformance errors. The **Data Cleansing** subcomponent is performing the necessary corrective or removal actions based on the conformance errors that were identified in the data validation step. The **Data Completion** subcomponent is safeguarding the required data completeness of the dataset with a list of predefined rules for the required attributes conformance and automatic filling of the missing values. The **Cleansing Logger** subcomponent is maintaining and storing all the identified errors, the corrective or removal actions taken during the execution of the cleansing workflow through an advanced logging mechanism. The stakeholder will be able to review and trace the results and actions during each step of the workflow with the relevant information.

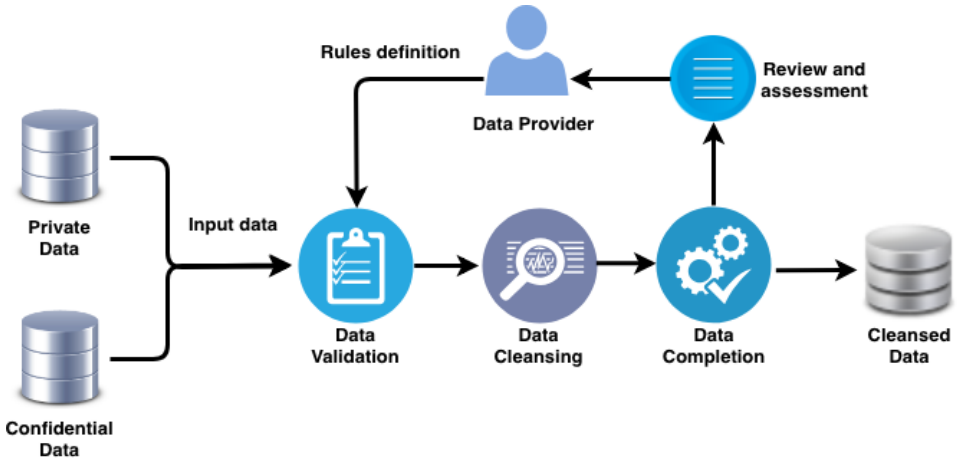

Figure 5-3: Cleanser overview

To meet its goal, the Cleanser is offering the intuitive user interface through which the stakeholder is able to configure and customise the rules and corrective actions that are included in the cleansing workflow, as well as generate and review a report with the execution results and the actions taken.

The **main functionalities of the Cleanser** are the following:

- Identify inaccurate, incomplete, incorrect or irrelevant data fields or field values on a selected dataset based on the predefined validation rules.
- Perform the data cleansing operations that include a series of corrective or removal actions for all identified conformance errors.
- Perform missing value handling with a variety of suitable techniques depending on the nature of the mandatory field.
- Provide the logging mechanism that monitors and stores all the identified errors, the actions performed and the corresponding results.
- Enable the rules configuration and the results visualisation through a user-friendly interface.

5.3.2.2 Addressed requirements

The Cleanser with the set of functionalities described in the component's design addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_014:** Cleanser is offering the suitable processes for data validation, data cleansing and data completion operation on top of a dataset towards the aim of addressing the data quality issues and maximising the usefulness of the dataset.

5.3.3 Mapper

5.3.3.1 Design and Functionalities overview

The need for a common underlying data model emerged as a clear stakeholder requirement during the user requirements elicitation steps documented in previous sections of the current deliverable. Moreover, its necessity is discussed and explained in detail in D2.1 and D2.2, as it constitutes a

prerequisite for the effective data exploration, integration and analysis in the aviation domain. In brief, the volume and heterogeneity of the data sources in the domain, not only in terms of pure content, but often in terms of structure and formats, makes the existence of a common data model imperative when data-enabled decision making over multiple datasets from diverse sources and stakeholders is provisioned.

All datasets stored in the ICARUS platform, either available for sharing, or intended for exclusive usage by their owner, should therefore conform to the common data model in a way that allows them to be easily combined with other datasets, and further facilitates the application of the platform's analysis and visualisation services. The ICARUS Mapper, shown as part of the On Premise Environment in Figure 5-1, is the tool that undertakes the task of identifying and defining the way a dataset is mapped to the common data model. In reality, the Mapper, as well as the other components of the On Premise Environment, has a dual presence in the architecture:

a. A user interface that will be part of the web platform, i.e. the user will interact with the graphical interface of the Mapper through the platform's web interface. Through this interface the user will be able to browse the entities and relationships of the ICARUS data model, preview and edit the proposed mappings. This interaction will conclude in the definition of the final mapping configuration, which will be executed by the backend data mapping service.
b. A backend data mapping service that will execute the actual defined mapping from the user's data to the common data model. The service will receive a mapping configuration in the form of a predefined template as a result of the interaction between the user and the Mapper's web interface. This template will hold the instructions of the mapping to be applied, which the service is then responsible for applying on the user's data.

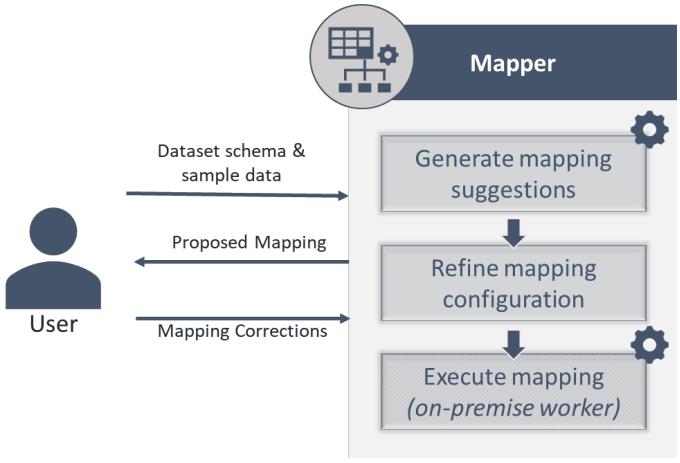

Figure 5-4: Mapper basic workflow (as perceived by the user)

It should be stressed that an exhaustive list of fields and relationships describing thoroughly all data that can emerge as potentially relevant to the aviation domain cannot be drafted. Consequently, the data model cannot be expected to be a static reference structure, but a dynamically changing

representation of the evolving aviation data landscape. Therefore, the Mapper will address also this need for updating the underlying data model in order to better suit the stakeholders' needs.

It should be noted that an instance of the on-premise Mapper will also be available in the core platform (as shown in Figure 5-1) in order to handle the schema mapping processes for the open data that will be imported in the ICARUS platform. If this mapping is not performed, open data would be essentially unusable by the various platform components, hence mapping is an essential preparatory step for all data that will be stored inside ICARUS.

In summary, **the main functionalities to be provided by the Mapper** are as follows:

- Provide the user with the most up to date data model in a way that enables easy browsing and discoverability of available entities and relationships described in the model
- Provide information on the data model's entities and relationships to help the user understand the model
- Generate proposed mapping from a new dataset to the ICARUS data model
- Allow the users (data owners that wish to provide data to the platform, i.e. data providers) to review and update the proposed mapping from the fields of their dataset to the data model fields
- Export a defined mapping in the form of an instructions template and send to the Mapper sub-component that resides in the On Premise Environment (backend data mapping service)
- Perform the actual mapping from the user's data to the common data model based on the instructions provided by the mapping template
- Allow the user to store and reuse a defined mapping template
- Allow the user to provide suggestions for data model extensions
- Allow the platform administrator to review the user suggestions for updating the model and approve or decline them
- Allow the platform administrator to extend or update the data model, e.g. with new fields

5.3.3.2 Addressed requirements

The Mapper addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_017:** The Mapper is the component that undertakes the task of ensuring that all data in the ICARUS data platform conform in a known way to the underlying common data model and therefore all platform components are aware of and can leverage this mapping when needed, e.g. for data discoverability, data exploration, data integration, data analysis and data visualisation purposes.
- **TR_018:** The Mapper is the component that ensures that external data being imported in ICARUS are mapped to the ICARUS data model in a semi-automatic manner

- **TR_020:** In the context of mapping a data field to the ICARUS data model, if a predefined set of values is linked to the corresponding field through a controlled vocabulary (e.g. airport codes), then this information will be included in the mapping configuration.
- **TR_026:** The Mapper will provide an interactive UI to let the user browse the ICARUS data model
- **TR_027:** The Mapper will allow the user to provide suggestions for extending/ updating the data model
- **TR_028:** The Mapper will allow the platform administrator to review the data model recommendations and approve or decline them

The Mapper also supports or partially addresses the following additional requirements:

- **TR_012:** The Mapper constitutes in a way the interface between the user and the data model and therefore, through the model, offers support for all data types described in the data requirements.
- **TR_032:** The Mapper will make the way the datasets in the platform are mapped to the common data model available to the other platform components
- **TR_045:** Through the generation and execution of the mapping, the Mapper indirectly facilitates the integration and combined analysis over multiple datasets (although the actual tasks are undertaken by other components)
- **TR_048:** Through the generation and execution of the mapping, the Mapper indirectly supports the combination (merging) of datasets based on common fields into one dataset (although the actual task is undertaken by other components)

5.3.4 Wallet Manager

5.3.4.1 Design and Functionalities overview

The Wallet Manager is the component that handles all blockchain-related operations in the context of the On Premise Environment. It is essentially a blockchain node that supports the following **main functionalities**:

- It interacts with the blockchain to report on the validity of smart contracts
- It informs the Decryption Manager whether a request for data access should be granted or denied based on the status of the corresponding smart contract.

5.3.4.2 Addressed requirements

The Wallet Manager, due to its nature (i.e. being a node in the blockchain) does not individually address any of the defined technical requirements. However, it plays a part in addressing the following requirements:

- **TR_035, TR_040, TR_042, TR_043, TR_067, TR_068**: These requirements are related to the way ICARUS controls access to datasets through specific policies and smart contracts which are stored in a DLT-based repository. Hence, the Wallet Manager partially supports all of them.

5.3.5 Encryption Manager

5.3.5.1 Design and Functionalities overview

The Encryption Manager is the component responsible for providing the encryption mechanisms of the data within the context of the ICARUS platform, as well as for the mechanisms that enable the secure and controlled sharing of encrypted datasets between the data provider and the data consumer.

The security and privacy of the datasets residing on a platform is critical point in the design of every data-driven platform. On the one hand, datasets should be stored encrypted for security reasons so that unauthorised access is prevented and privacy is safeguarded. Within this scope, the data provider should be able to control and authorise the access to its proprietary datasets by the rest of the users. On the other hand, the datasets should be unencrypted in order to be properly and efficiently processed, especially in the case of data analysis execution.

In deliverable D2.1, the ICARUS Data Encryption method was presented. Within this method, the decision that all datasets that shall become available in the ICARUS platform will be encrypted and securely transmitted between the data provider's premises, the ICARUS platform and the data consumer, was elaborated. In accordance with this method, the Encryption Manager is providing the required functionalities from the data provider's side in the encryption-decryption workflow. Hence, the purpose of the Encryption Manager is threefold: (a) to facilitate the dataset encryption process with the help of a locally generated symmetric key, (b) to enable the encrypted dataset sharing process with the generation and the secure sharing of the decryption keys in the form of key pairs with the eligible (via an active data contract) data consumers, and (c) to manage and maintain the list of generated decryption keys, as well as to handle the revocation process of any of these keys.

Within ICARUS, a column-based encryption will be followed as documented in D2.1. However certain columns that contain spatiotemporal information will remain unencrypted in order to facilitate the efficient data browsing and exploration that will be offered by components such as the Query Explorer without though compromising the privacy or security of the datasets

The Encryption Manager has a dual presence in the ICARUS architecture (as shown in Figure 5-1). On the one hand, an instance of the Encryption Manager is running on the On Premise environment in order to provide the encryption processes for the datasets of the data providers. On the other hand, an instance of the Encryption Manager is running on the Secure and Private Space in order to ensure that the results of the executed analysis are also encrypted before they are securely transmitted and stored in the Core ICARUS platform.

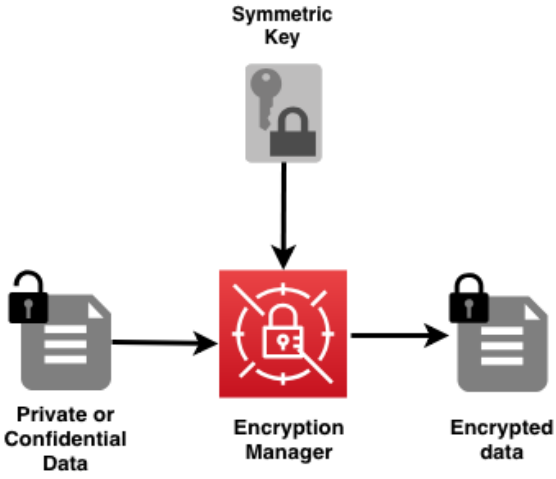

Figure 5-5: Encryption Manager - encryption process

The **main functionalities of the Encryption Manager** are the following:

- Perform the on-demand column-based encryption of a dataset based on a locally generated symmetric key.
- Enable the sharing of encrypted datasets between data providers and data consumers with the generation of decryption keys in the form of key pairs, one per dataset per data consumer.
- Securely transmit the generated decryption key to the data consumer.
- Maintain a local key store where generated decryption keys and all relevant information is stored.
- Handle revocation requests in which access to an encrypted dataset is revoked for a specific data consumer.

5.3.5.2 Addressed requirements

The Encryption Manager with the set of functionalities described in the component's design addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_011:** The Encryption Manager is the component that enables the end-to-end data encryption, in collaboration with the Decryption Manager and Key Pair Administrator, providing the mechanism to the data provider that is used for the encryption of the dataset, while also supporting the execution of the decryption key exchange that enables the dataset sharing.
- **TR_013:** As part of the encryption process, the Encryption Manager is facilitating the selection of the field types that will be encrypted before the execution of the encryption process.
- **TR_033:** With regards to the spatiotemporal field types, the Encryption Manager is ensuring that these fields are not encrypted in order to enable the query execution.

5.3.6 Decryption Manager

5.3.6.1 Design and Functionalities overview

The Decryption Manager is the component responsible for providing the decryption mechanisms of the encrypted data within the context of the ICARUS platform by acquiring the corresponding decryption key from the data provider and performing the decryption of the encrypted dataset.

As already described in section 5.3.5.1, within ICARUS all datasets will be encrypted within the On Premise Environment prior to being uploaded and stored in the ICARUS storage solution. Upon exploring the various datasets stored in the ICARUS platform and receiving the access approval from the data provider, the data consumer obtains access to the encrypted dataset. However, to be able to decrypt the dataset without compromising the data privacy of the data provider, the ICARUS Data Encryption method, as elaborated in D2.1, is followed, and a decryption key in the form of a key pair is generated by the Encryption Manager running on the data provider's side. Thus, the Decryption Manager is providing the required functionalities from the data consumer's side in the encryption-decryption workflow as described in the ICARUS Data Encryption method.

The Decryption Manager is responsible for initiating the request for usage of a dataset to the Key Pair Administrator which is responsible for establishing the secure communication between the data consumer and the corresponding data provider for the requested dataset. The Decryption Manager is also responsible to provide the means to verify its identity via the appropriate certificate or public key towards the secure communication establishment. Finally, the Decryption Manager is providing the proper decryption mechanism that upon receiving the decryption key will be capable of decrypting the encrypted dataset.

Figure 5-6: Decryption Manager – decryption process

In summary, the **main functionalities of the Decryption Manager** are the following:

- Initiate the request for a dataset in the Key Pair Administrator that will trigger the generation of the decryption key, in the form of key pair, from the data provider.
- Verify the data consumer's identity in order to establish the secure connection with the data provider so as to receive the decryption key that will be utilised in the dataset decryption process.
- Implement the decryption mechanism that decrypts the dataset based on the received decryption key.

5.3.6.2 Addressed requirements

The Decryption Manager with the set of functionalities described above addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_011:** The Decryption Manager enables the end to end data encryption of the datasets, along with the Encryption Manager and Key Pair Administrator, facilitating the decryption of the datasets according to the decryption key exchange mechanism of ICARUS platform.

5.3.7 Key-Pair administrator

5.3.7.1 Design and Functionalities overview

The Key-Pair administrator is the component that is facilitating the exchange of the decryption keys between the data consumer and the data provider in order to enable the end-to-end data encryption and secure sharing of the encrypted datasets, following the approach of the ICARUS Data Encryption method defined in D2.1. In this context, the Key-Pair administrator performs the signalling operations between the data consumer and data provider to achieve the establishment of a secure connection between these parties, by performing the identity verification of each party and the successful handshake between them.

As described also in section 5.3.5.1, within the ICARUS platform the approach of decryption keys exchange is followed. Within this approach, the datasets are encrypted by each data provider and requests are issued by the data consumers to the data providers in order to access the encrypted datasets that result in a decryption key (in the form of key pair) generation from the data provider for each request. However, the approach requires the exchange of the corresponding decryption key between the data consumer and the data provider so that the data provider can decrypt and utilise the selected dataset. As such, the Key-Pair Administrator is responsible for supporting the decryption key exchange. At first, the Key Pair Administrator is maintaining the list of the data providers and the relevant datasets. When the data consumer initiates a request for accessing a specific dataset, the Key Pair Administrator is responsible for establishing the secure connection between the data consumer and the data provider in order to perform the decryption key exchange. The signalling operations include the setup of secure transport encryption and two-sided authorisation, as well as the identity verification. Furthermore, the Key Pair Administrator is supporting the revocation process of a decryption key when access is revoked and the relevant decryption key is invalidated.

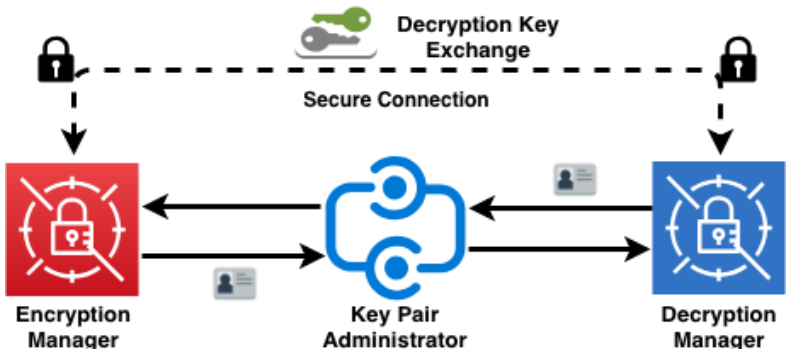

Figure 5-7: Key Pair Administrator overview

The **main functionalities of the Key-Pair administrator** are the following:

- Perform the signalling operations in order to establish the secure connection for the exchange of the decryption key between the data provider and the data consumer.
- Maintain the list of data providers that have provided encrypted datasets in order to support the decryption key exchange requests issued by the data consumers.
- Provide and support the mechanism for the revocation of the decryption keys of any dataset provided by the data consumers.

5.3.7.2 Addressed requirements

The Key Pair Administrator with the set of functionalities described in the component's design addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_011:** The Key Pair Administrator is the component that enables the end-to-end data encryption and the realisation of the approach of decryption keys exchange, in collaboration with the Encryption Manager and Decryption Manager, facilitating this exchange with the establishment of a secure connection between the data consumer and the data provider. Additionally, it safeguards the end-to-end data encryption mechanism with a revocation mechanism when needed.

5.3.8 Data Handler

5.3.8.1 Design and Functionalities overview

The Data Handler component encapsulates various services responsible for tasks related to making data available from and to the ICARUS platform, as well as among different platform components. It serves as the "data gateway" in the ICARUS architecture, as it supports the complete workflow of uploading proprietary and open datasets to the platform, downloading datasets from the platform to the end user's On Premise Environment and/or to a Secure and Private Space, and finally the uploading

of data generated in a Secure and Private Space back into the core platform's storage. In order to properly handle all the above, the Data Handler also needs to support the definition of appropriate metadata for each dataset, which are essential for various ICARUS functionalities, indicatively including searching for and acquiring datasets. Briefly, the Data Handler offers the following **core functionalities:**

- It enables the uploading of datasets that have been exported by the On Premise Environment worker (OnPremise Worker) in the ICARUS platform.
- It sends the uploaded data to the ICARUS Storage and Indexing component.
- It enables the import of open data that may come from various sources in the ICARUS platform.
- It enables the download of datasets that are stored in the ICARUS platform.
- It communicates with the Master Controller when a dataset needs to be transferred in one of the Secure and Private Spaces. The Data Handler does not perform the actual data transfer but delivers the message and corresponding instructions to the Master Controller.
- It allows data coming from / generated in one of the ICARUS Secure and Private Spaces to be stored in the platform's storage. It should be noted that in order for the Data Handler to perform all necessary operations for the data and their metadata to be appropriately stored and handled, the data will also need to be mapped to the common data model.

5.3.8.2 Addressed requirements

The Data Handler addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_001, TR_002, TR_004, TR_005, TR_006, TR_009, TR_010:** These technical requirements correspond to the core functionalities of the Data Handler, as described in the previous sub-section. Especially regarding TR_006, it should be stressed that since some of the datasets uploaded in the ICARUS platform are expected to be updated, the Data Handler shall ensure that data updates are properly handled.
- **TR_003:** The envisioned simplified data check-in process will allow users to skip some of the metadata definition steps, in the case that an uploaded dataset is meant to remain completely private, i.e. neither processable nor accessible by anyone other than the owner (uploader). Since the Data Handler is considered the interaction point for all data import and export activities by the user, it can be considered that this requirement is also addressed by the current component.

The Data Handler also supports or partially addresses the following additional requirements:

- **TR_018:** The Data Handler is not responsible for performing the mapping to the ICARUS data model, but it will ensure that only data that have undergone this process are allowed to be stored inside the core platform.

- **TR_019, TR_021, TR_025:** The Data Handler, being the interaction point between the user and the core platform for all functionalities related to data import, will also support the provision of metadata (tags, license-related, provenance) for the datasets, in coordination with other platform components.

5.3.9 Data License and Agreement Manager

5.3.9.1 Design and Functionalities overview

The Data License and Agreement Manager is the component responsible for handling all processes related to the data licenses and IPR attributes, as well as the drafting, signing, and enforcing the smart data contracts that correspond to data sharing agreements between platform users. The component has three, seemingly distinct, but interconnected roles:

a. It offers a graphical interface that allows the users to define and review data licenses attached to datasets uploaded to the ICARUS platform. The term license here is used in a broader sense, as the component will handle all license-related metadata information that is provisioned, as discussed and described in D2.2. The information defined here will be stored in the core platform's storage and will be made available to all other components that need to query it.
b. It offers a graphical interface that allows users to draft, review, negotiate on, and sign a smart data contract that concretely defines the terms under which a dataset will be shared.
c. It interacts with the platform's blockchain node to report on the validity of smart contracts. Furthermore, it handles all processes required to prepare a smart contract for each (paid) asset transaction and, finally, upload it to the blockchain. It will also enable the activation (i.e. status change) of a smart contract when both parties - data owner (seller), data consumer (buyer) - approve it and the payment is completed.

In brief, **the core functionalities of this component** are as follows:

- It allows the users to define IPR related attributes for the datasets they own in the platform
- It allows the users to define pricing terms and policies for the datasets they own in the platform
- It provides predefined data license templates that the users can review and assign to the datasets they own in the platform
- It enables the users to draft their own custom data licenses and assign them to the datasets they own in the platform
- It enables the users to create, edit, review, update and sign data sharing agreements (to be added in the blockchain)
- It enables the users to negotiate on (i.e. iteratively update and review) a data sharing agreement prior to signing
- It enables the users to decline requests for providing access to their datasets, without proceeding to a contract drafting process

- It prepares and uploads smart data contracts in the blockchain and updates their status as needed
- It checks the status of a given smart data contract
- It allows active data contracts to be executed directly by the platform (and other participating blockchain nodes). In such case, the platform and the OnPremise Workers will handle directly requests for data access that correspond to valid contracts.

5.3.9.2 Addressed requirements

The Data License and Agreement Manager addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_021, TR_022, TR_023, TR_024, TR_037:** These requirements correspond to the definition of license and pricing related metadata for the datasets uploaded in the platform and, as described in the previous subsection, they are included in the core functionalities of the component.
- **TR_035, TR_036, TR_039, TR_040, TR_043:** These requirements correspond to the smart data contracts' management and, as described in the previous subsection, they are included in the core functionalities of the component.
- **TR_041, TR_042:** Through its user interface, the component facilitates the processes of performing and managing requests for data access, although other components are also involved in the corresponding workflows.
- **TR_042:** The ICARUS platform shall allow users to accept or deny requests for access on their datasets made by other users
- **TR_067, TR_068, TR_071:** These requirements are not individually addressed by the Data License and Agreement Manager, but as they are related to the controlled access over datasets, the component can be considered as partially supporting them.

5.3.10 Policy Manager

5.3.10.1 Design and Functionalities overview

The Policy Manager is the component providing the authorisation engine that implements the access control mechanisms within the ICARUS platform. The purpose of the Policy Manager is to provide the logical access control that prevents the unauthorised access of any type of resource of the ICARUS platform such as data, services, tools, any kind of system resources, as well as all other relevant objects.

In general, access to resources refers to discovering, reading, creating, editing, deleting, reserving and executing resources (NIST, 2014). In deliverable D2.1, the ICARUS Data Access Control method is elaborated. This methods dictates the utilisation of Attribute-Based Access Control (ABAC) (NIST, 2014) authorization policies that are based on the XACML standard provided by the Organization for

the Advancement of Structured Information Standards (OASIS), in order to permit or deny access requests to any type of resource of the ICARUS platform.

Hence, the Policy Manager is responsible for the implementation of the ICARUS authorisation engine that will be based on the ABAC model and incorporates the required authorisation XACML-based policies. The purpose of this authorisation engine is the provide the access control decision that will either grant or deny the access to the requestor by enforcing the formulated authorisation policies. Additionally, the Policy Manager will enable the definition, storage, reuse, update and disposal of the authorisation policies in order to allow the data providers to effectively define and manage the protection and sharing aspects of their datasets.

The **main functionalities of the Policy Manager a**re as follows:

- Provide the access control mechanism that is based on the ABAC model and the XACML reference implementation that will be used as the authorisation engine of the ICARUS platform.
- Control and restrict the access of any type of resource of the ICARUS platform based on the set of access control policies that are managed and maintained within the access control mechanism.
- Provide the interfaces that will manage and process any access request in order to either grant or deny access to the requestor

5.3.10.2 Addressed requirements

The Policy Manager addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_041:** The Policy Manager facilitates the access to datasets that are not owned by them once the appropriate purchase is completed with a valid smart contract as enabled by Data License and Agreement Manager.
- **TR_042, TR_067:** The Policy Manager is safeguarding that the access to datasets is granted only to the users that the datasets owners have accepted their access requests and a valid smart contract exists.
- **TR_068, TR_070, TR_071, TR_072:** The Policy Manager is providing the access control mechanism that incorporates different authorisation levels with the use of policies for the datasets and other resources of the platform.

5.3.11 ICARUS Storage and Indexing

5.3.11.1 Design and Functionalities overview

The ICARUS Storage and Indexing is the component that enables the storage and indexing capabilities of the ICARUS platform. This component is responsible for the effective and efficient storage and maintenance of large, complex and unrelated datasets within the ICARUS platform, as well as the flexible and high-performance indexing of the stored datasets.

In every Big Data platform, the data storage component holds a key role in the effective and successful operation of the platform. The data storage component is responsible not only for storing and managing the data of the platform but also for addressing the needs of the rest of the component for data access that facilitates their successful operation. In order to support its key role, the ICARUS Storage and Indexing component has to offer a set of core characteristics. The ICARUS Storage and Indexing should ensure high availability of the stored data in order to support the various functionalities of the platform that require instance access to the data, while also offering flexibility and efficiency by supporting multiple concurrent requests. The ICARUS Storage and Indexing should handle a variety of different data models originating from various heterogeneous data sources in large volumes. In order to cope with the increasing volume of data, the ICARUS Storage and Indexing should high scalability by utilising techniques that enable horizontal scalability based on distributed architectures. Moreover, the ICARUS Storage and Indexing should offer high performance in storing large volume of data in a timely and efficient manner, but also support high performance querying over the stored data. Finally, one crucial characteristic is the availability of advanced security mechanisms that offer methods for increasing the security, the privacy and the data protection of the stored information.

In addition to the storage capabilities, the ICARUS Storage and Indexing component is offering the effective and efficient indexing mechanism that facilitates the near real-time indexing and advanced querying capabilities over the indexed data. The indexing mechanism will support multiple functionalities such as advanced full-text search, geospatial search, terms boosting, spelling checking and results highlighting. In order to achieve this, a well-defined indexing schema will be designed that will be based on the ICARUS common aviation model. This schema will be enriched as the platform evolves and new datasets from new data sources are incorporated in the platform. Since the ICARUS platform will support column-based encryption, the indexing will be performed only in the unencrypted columns of the datasets.

The **main functionalities of the ICARUS Storage and Indexing** are the following:

- Handle the storage of the incoming large datasets as provided by the Data Handler component.
- Handle the multiple requests for data retrieval from the various components of the platform in an efficient and effective manner.
- Support the storage of large encrypted datasets.
- Facilitate the query execution on top of the stored datasets.
- Support the indexing of the stored datasets to facilitate full-text search of the unencrypted columns of the datasets
- Support efficient geospatial querying over the stored datasets.

5.3.11.2 Addressed requirements

The ICARUS Storage and Indexing addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_009:** The ICARUS Storage and Indexing supports by design, as described also in the section 5.3.11.1, the effective storage of large files.
- **TR_012:** The ICARUS Storage and Indexing supports the storage all the data types that are provided by the Data Handler.
- **TR_068:** The ICARUS Storage and Indexing supports the access control over the stored datasets in collaboration with the Policy Manager.
- **TR_069:** The ICARUS Storage and Indexing ensures the security aspect of the data in storage with the utilisation of a variety of data access control, data integrity and data consistency mechanisms at the storage level.

5.3.12 Master Controller

5.3.12.1 Design and Functionalities overview

The Master Controller is the component responsible for compiling a set of instructions for the execution of specific jobs or tasks, as provided by the components of the Core ICARUS platform, and for providing this set of instructions for local execution to the workers running on the On Premise Environment and the Secure and Private Space, namely the OnPremise Worker and the SecureSpace Worker, and for monitoring the execution status of requested jobs or tasks.

Following the Master/ Worker paradigm, the Master Controller is located on the Core ICARUS platform and is responsible for allocating the jobs/ tasks to be executed to the relevant workers which are responsible for the execution and completion of these jobs/ tasks using the components running on the environment in which they are deployed. More specifically, for the case of the OnPremise Worker the Master Controller is compiling a set of instructions for the Cleanser, the Mapper, the Anonymiser, the Wallet Manager and the Encryption Manager that should be executed on the dataset of the data provider prior to being uploaded in the ICARUS platform. For the case of the SecureSpace Worker, the Master Controller is compiling a set of instructions for the Decryption Manager, the Job Scheduler and Execution Engine and the Encryption Manager. Additionally, the Master Controller is handling the requests for transferring the selected encrypted datasets in the Secure and Private Space for the analysis execution.

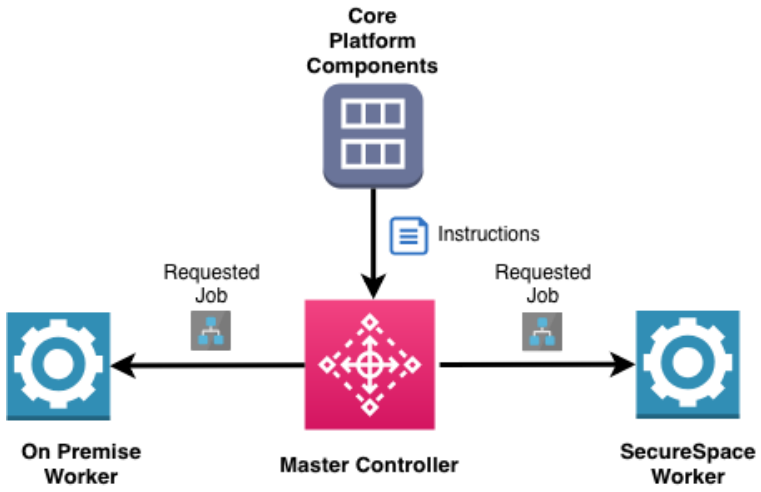

Figure 5-8: Master Controller basic workflow

The **main functionalities of the Master Controller** are the following:

- Establish a connection with the OnPremise Worker running on the On Premise Environment and the SecureSpace Worker running on the Secure and Private Space.
- Compile the set of instructions for the job execution that should be executed by the relevant worker utilising the set of components on running on the environment that the worker is deployed.
- Provide the set of instructions in the form of jobs/tasks to the workers and monitor the execution status.
- Transfer the list of selected encrypted datasets to the Secure and Private Space.

5.3.12.2 Addressed requirements

The Master Controller cannot be directly mapped to the requirements documented in section 4.1, however it implements one of the core functionalities in the design of the ICARUS architecture which is the interconnection of the Core ICARUS platform with the On Premise Environment and the Secure and Private Space for the execution of the data preparation of the private and confidential data prior to be uploaded in the ICARUS Core platform and the execution of the data analysis in a secure "sandboxed" environment.

5.3.13 OnPremise Worker and SecureSpace Worker

5.3.13.1 Design and Functionalities overview

Both the OnPremise Worker and SecureSpace Worker are the components responsible for the execution of the jobs or tasks, as received by the Master Controller, utilising the available components on their local running environment.

The OnPremise Worker is running on the On Premise Environment and is undertaking the jobs or tasks that are related to the data preparation prior to being uploaded in the ICARUS platform. The worker

receives a set of instructions that involves the cleansing process executed by the Cleanser, the mapping of the user's data to the common data model, the anonymisation process executed by the Anonymiser, the blockchain-related operations executed by the Wallet Manager and the encryption process performed by the Encryption Manager. Additionally, the results of this data preparation are provided, in the form of an encrypted dataset, to the Data Handler through the OnPremise Worker for further processing and storage.

The SecureSpace Worker resides at the Secure and Private Space and is responsible for the job or task execution that is related to the data analysis that is performed on the Secure and Private Space. The worker is enabling the transfer of selected encrypted datasets that will be utilised in the data analysis in the Secure and Private Space. Additionally, it receives a set of instructions that includes the decryption process of the selected datasets as performed by the Decryption Manager, the analytics job execution that is performed by the Jobs Scheduler and Execution Engine and the encryption of the produced results of the analytics as executed by the Encryption Manager. Moreover, the SecureSpace Worker is responsible for providing the encrypted results to the Data Handler for storage.

The **main functionalities of the OnPremise Worker and the SecureSpace Worker** are the following:

- Interpret and execute the instructions for the job or task execution as provided by the Master Controller utilising the components running on their local environment
- Provide the execution status of the requested jobs/tasks to the Master Controller
- Support the uploading of the prepared encrypted datasets from the On Premise Environment to the Data Handler
- Support the uploading of the encrypted analysis results from the Secure and Private Space to the Data Handler

5.3.13.2 Addressed requirements

In general, both workers, the OnPremise Worker and the SecureSpace Worker, are facilitating the execution of the jobs and tasks, as instructed by the Core ICARUS platform, in the On Premise Environment and the Secure and Private Space. Although their functionalities cannot be directly mapped to the requirements presented in section 4.1 besides the TR_004 requirement, they are supporting in the background, as explained, the execution of multiple platform operations and are of high importance for the overall platform successful operation. For the TR_004 requirement in particular, the SecureSpace Worker is enabling the transferring and storage of the encrypted results of the analysis in the Data Handler in order to be eventually stored in the ICARUS Storage.

5.3.14 Query Explorer

5.3.14.1 Design and Functionalities overview

The Query Explorer is the component that offers dataset exploration and discoverability functionalities to the platform users. Query Explorer has two main offerings: (a) a graphical interface for users to search for datasets and view the search results and (b) a service that translates each search

to a query that can be processed by the storage and indexing component. As such, the Query Explorer constitutes the main facilitator of the ICARUS data marketplace functionalities and the main interaction point from the user's perspective. Specifically, it provides the following **core functionalities**:

- It will allow users to define search criteria in order to discover potentially interesting datasets in the platform. Specifically, it will enable the user:
  - to select fields from data model that should be present in the datasets that will be included in the results and
  - to define and apply filters based on the metadata of the datasets that will be included in the results
  - to define filters on the actual data of the datasets that will be included in the results. These filters will be only available for unencrypted data columns that belong to specific data types (i.e. only for data columns that hold spatiotemporal information)
- It will translate the query configuration described above to a query language suitable for the Storage and Indexing Component
- It will perform the generated query to the Storage and Indexing component and show the returned results to the user
- It will store the query history of each user and allow the user to browse it and perform again previous queries
- Apart from the actual query results, the Query Explorer will also show to the user some dataset suggestions, which will be provided by the Recommender component

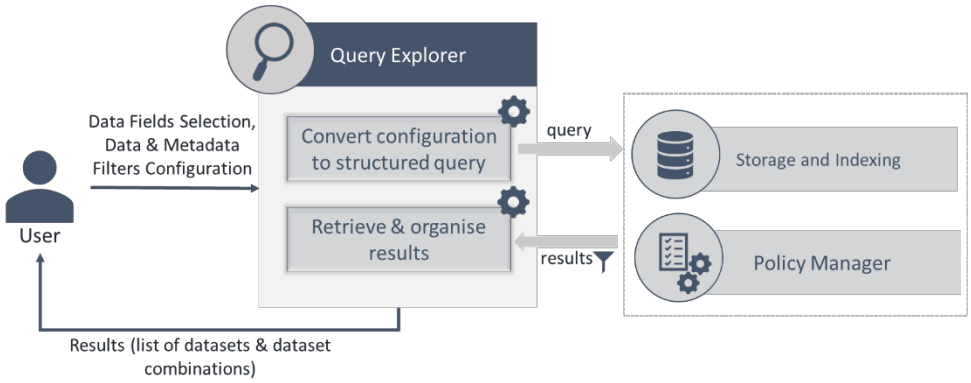

Figure 5-9: Query Explorer basic workflow (user's perspective)

5.3.14.2 Addressed requirements

The Query Explorer addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_029, TR_030, TR_031:** These technical requirements correspond to the core functionalities of the Query Explorer, as described in the previous sub-section.

- **TR_032:** The Query Explorer is the tool that leverages the identified connections among datasets, i.e. their mapping to the common data model, in order to retrieve the datasets that are of interest to the user according to the performed query, the configuration of which is based on the fields of the common data model
- **TR_016, TR_017**: The Query Explorer will, by definition, conform with the common underlying metadata schema and the common underlying data model in order to offer its functionalities
- **TR_033**: The second part of this requirement, i.e. the ability to search over unencrypted spatiotemporal information of the available datasets, is one of the core functionalities of Query Explorer, as described above.
- **TR_034**: The search results interface of Query Explorer constitutes an information catalogue about all datasets that are open or available for sharing (by their respective data providers).
- **TR_041**: This requirement is only partially addressed by the Query Explorer, since through its interface users will be able to review information regarding purchasing a dataset that is included in the search results.

5.3.15 Recommender

5.3.15.1 Design and Functionalities overview

The Recommender is the component providing recommendations to the users with regards to the datasets. In particular, this component is responsible to recommend and suggest to the users additional related datasets during the search and query process. Furthermore, the Recommender will provide recommendations based on the users' preferences and history of searches, requests and purchases. The Recommender consists of an iterative procedure (collect information, make recommendations) that aims to improve the quality and accuracy of the recommendations in each iteration. The information collection phase collects relevant information of users to generate a user profile or model for the prediction tasks. A recommendation system cannot function accurately until the user profile/model has been well constructed. The system needs to know as much as possible from the user in order to provide reasonable recommendation right from the onset. The success of any recommendation system depends largely on its ability to represent user's current interests.

Figure 5-10 shows the overview and the interactions of the Recommender with other ICARUS components. First of all, the Recommender receives from the Query Explorer as input a user's search and then the Recommender is responsible for retrieving from the database any other useful information (e.g. user preferences etc.) that can improve the recommendations. Afterwards, based on the received input, the learning phase of the Recommender is activated in order to construct or update the model. Finally, the new model generates the recommendations and the top N recommended datasets are provided to the Query Explorer in order to display them to the end-user.

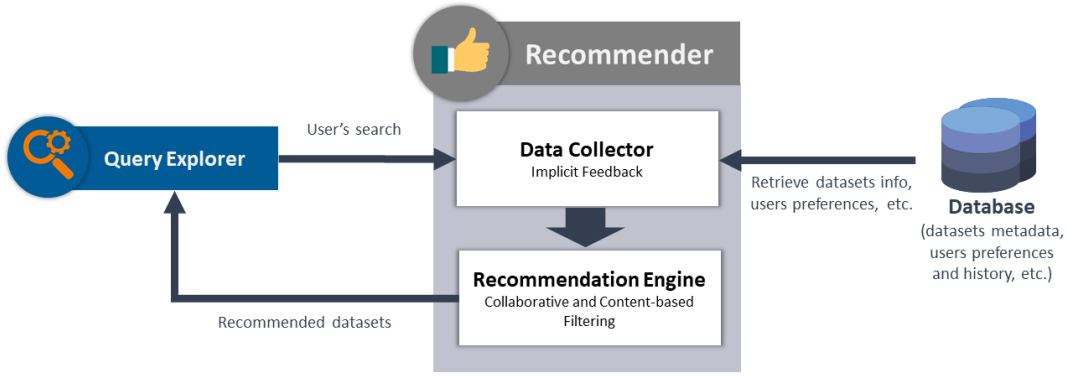

Figure 5-10: Recommender overview

The **main functionalities of the Recommender** are the following:

- Access the database and retrieve metadata of the datasets, user's behaviour (e.g. searches, requests, preferences, purchases etc.).
- Generate personalised datasets recommendations by using a combination of collaborative filtering and content-based filtering.

The Recommender is composed by two main sub-components, the **Data Collector** and **Recommendation Engine** with the following main functionalities:

- The **Data Collector** is responsible for collecting relevant information about the data assets (e.g. data asset category/topic) and the stakeholders (e.g. requests and purchases) in order to provide the Recommendation Engine with the necessary information.
- The **Recommendation Engine** is responsible for providing justified and accurate recommendations to assist the stakeholders and enhance data discoverability. The Recommendation Engine will utilise the data that were collected from the Data Collector by comparing the views and searching habits of similar users (i.e., collaborative filtering) as well as by comparing data assets that share characteristics with datasets that a user has purchased (content-based filtering).

5.3.15.2 Addressed requirements

The Recommender with the set of functionalities described in the component's design addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_031:** The Recommender is enabling the enhancement of data exploration and discoverability by providing personalised recommendations and suggestions for additional related datasets that can be explored or utilised.

5.3.16 Analytics and Visualisation Workbench

5.3.16.1 Design and Functionalities overview

The Analytics and Visualisation Workbench is facilitating the design, execution and monitoring of the data analytics workflows within the ICARUS platform. Through its intuitive graphical interface, the users are able to design the data analytics workflow that is tailored to their needs by selecting their preferred analytics algorithm from the extended list of algorithms that are offered by the ICARUS platform. Moreover, depending on the selected algorithm the users are able to set the corresponding parameters in order to further personalise the algorithm execution according to their business needs.

Besides the algorithm selection, the users are able to select the list of desired datasets, from the list of datasets that they are entitled to use, that will be used as the input data during the algorithm execution. Additionally, the users will be able to store the designed workflow as an ICARUS application within the BDA Application Catalogue, which constitutes the application repository of the ICARUS platform, in order to restore, modify, re-execute it later or share it with the rest of the users of the platform through a defined license. As such, the Analytics and Visualisation Workbench facilitates both the creation of a new data analytics workflow with its novel graphical interface and the management and re-execution of an existing data analytics workflow that has been previously stored as an ICARUS application.

While the execution of the designed workflow is triggered within the Analytics and Visualisation Workbench, the actual execution is performed within the Secure and Private Space. More specifically, the execution process is secured into a sandboxed runtime environment within the Secure and Private Space, as provisioned by the Resource Orchestrator. The Analytics and Visualisation Workbench will act as a client sending all the requests for workflow execution to the Jobs Scheduler and Execution Engine through the Master Controller and the SecureSpace Worker. Furthermore, the Analytics and Visualisation Workbench covers the logging and monitoring aspects based on the metrics gathered during the execution of the designed workflows, while also offering the scheduling capabilities for the execution of the designed workflow in a selected date and time.

The Analytics and Visualisation Workbench allows the user to extract meaningful information through a modern data visualisation suite of charts and visualisations devised considering the most powerful data visualisation patterns available in literature and the most dominant data frameworks adopted for Big Data Analytics. Some of the visualisations are especially recommended for the overviews, some others for interactive data exploration and more than one visualisations can be combined in order to form a dynamic dashboard suitable to address the users' needs.

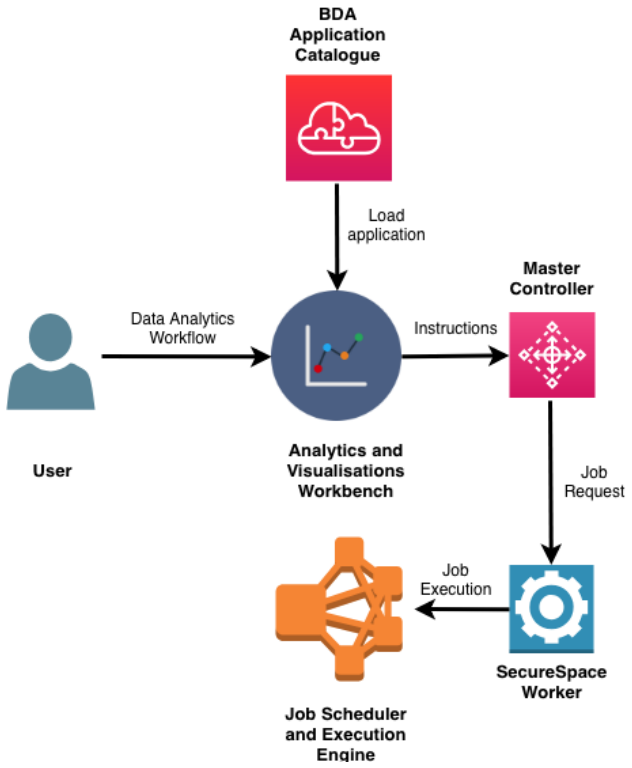

Figure 5-11: Analytics and Visualisation Workbench basic workflow

In summary, **the main functionalities to be provided by the Analytics and Visualisation Workbench** are as follows:

- Provide an intuitive graphical interface where the users are facilitated to design the data analytics workflow tailored to their needs, composed by an analytics algorithm and its parameters, a set of selected datasets and a set of desired visualisations.
- Provide the ability to store, modify, reuse and share the designed workflows in the form of ICARUS applications that are registered within the BDA Application Catalogue.
- Initiate and monitor the workflow execution of the designed analytics workflow into the Secure and Private Space
- Offer scheduling capabilities for the execution of the designed analytics workflow in a selected date and time.
- Provide a modern data visualisation suite of charts and visualisations that span from basic charts to advanced multilevel visualisations.
- Enable the creation of dynamic dashboards composed by multiple selected visualisations.

5.3.16.2 Addressed requirements

The Analytics and Visualisation Workbench will match the following needs according to the list of technical requirements documented in section 4.1:

- **TR_044**: The Analytics and Visualisation Workbench enables the creation, storage, management and launch of data analytics workflows through its novel graphical interface.
- **TR_045, TR_059**: The Analytics and Visualisation Workbench facilitates the integration or aggregation of multiple datasets towards the effective and efficient data analysis execution.
- **TR_046, TR_047**: The Analytics and Visualisation Workbench enables the design and execution of advanced and complex data analytics workflows in which an extensive list of data analysis algorithms is supported.
- **TR_048**: The Analytics and Visualisation Workbench provides the means of combining multiple different datasets originating from different data sources in order to execute a data analytics workflow.
- **TR_049**: The Analytics and Visualisation Workbench provides a user-friendly monitoring dashboard that displays the status of active, completed and failed jobs.
- **TR_050, TR_051, TR_052, TR_053, TR_054, TR_055:** The Analytics and Visualisation Workbench supports an extensive list of data analysis algorithms, which span from simple statistical analysis algorithms to more complex and advanced analysis algorithms such as machine learning and deep learning algorithms.
- **TR_056**: The Analytics and Visualisation Workbench offers a modern data visualisation suite of charts and visualisations, containing multiple visualisation formats which span from simple static charts to more advanced visualisation formats such as interactive charts with multiple layers of information.
- **TR_057, TR_058:** The Analytics and Visualisation Workbench is offering the scheduling mechanism in order to define and schedule the execution of a designed data analysis workflow in a selected date and time.
- **TR_060:** The Analytics and Visualisation Workbench is enabling the data analytics execution in a scalable and reliable manner with help of the Job Scheduler and Execution Engine component.

5.3.17 BDA Application Catalogue

5.3.17.1 Design and Functionalities overview

The BDA Application Catalogue is implementing the repository of the ICARUS applications in which the designed data analytics workflows, as created by the users of the platform, can be stored, retrieved, modified and loaded in the Analytics and Visualisation Workbench. More specifically, an ICARUS application is a set of selected datasets, data analysis algorithms and their corresponding parameters, as well as a set of selected visualisations.

The BDA Application Catalogue is making use of an internal configuration model that collects all the required metadata for the selected datasets, the selected data analytics algorithms and the selected visualisation types and their parameters. These metadata are stored in a dedicated database and can

be used within the Analytics and Visualisation Workbench environment in order to perform the execution of the designed data analytics workflow. Additionally, the BDA Application Catalogue is enabling the sharing of the stored ICARUS applications between the users of the platform through a sharing license that is defined for this specific ICARUS application. The ICARUS application can be defined as a private application, that only the user (owner) of the application will have access to this application, or as a public application that the rest of users of the platform can used under the license condition defined by the owner of the application.

As such, the purpose of the BDA Application Catalogue is twofold: a) to enable the storage and reuse of the designed data analytics workflows from the users, and b) to facilitate the sharing of the designed data analytics workflows among the users of the platform under a sharing license.

In summary, the **main functionalities of the BDA Application Catalogue** are as follows:

- Provide a repository where the user-defined data analytics workflows can be stored in the form of ICARUS applications.
- Enable the storage, modification and reuse of the ICARUS applications within the Analytics and Visualisation Workbench environment.
- Facilitate the sharing of the stored ICARUS applications between the users of the platform under a sharing license as defined by the owner of each application.

5.3.17.2 Addressed requirements

- **TR_044, TR_047, TR_058**: The BDA Application Catalogue enables the storage, reuse and sharing of the design data analytics workflows in the form of ICARUS applications providing the repository for these applications.

5.3.18 Resource Orchestrator

5.3.18.1 Design and Functionalities overview

The Resource Orchestrator is the component providing the realisation of the Secure and Private Space that will be provided to the platform users in order to perform the data analysis. More specifically, Resource Orchestrator undertakes the responsibility of the provision and management of an isolated and secure environment for each user of the platform in which the various analytical and visualisation features of the platform will be executed.

The Resource Orchestrator supports the dynamic deployment and management of a “sandboxed” environment by utilising a set of techniques and technologies that enable easy, fast and secure deployment over virtualised infrastructure. Within the Secure and Private Space, the main components that will be deployed are the SecureSpace Worker which is ensuring the task execution as received by the Master Controller, as well as the encrypted datasets transfer, the Job Scheduler and Execution Engine, which is enabling the analytics aspects of the ICARUS platform, and the Execution Cluster that is providing the processing execution engine with distributed job execution.

Additionally, the Encryption and Decryption Manager, that are utilised within encryption and decryption process of the datasets of the ICARUS platform, will be also deployed.

In computing, infrastructure refers to resources which can be virtual or physical and include computing, storage and network resources. Virtualised infrastructure is a software-based infrastructure which allows the creation of multiple simulated environments or dedicated resources from a single, physical hardware system through a software called hypervisor which connects directly to that hardware and allows the split of one system into separate, distinct and secure environments called virtual machines (VMs) (RedHat, 2017). The focus of the Resource Orchestrator is towards the deployment of scalable virtual machines in a virtualised infrastructure.

Within the context of ICARUS, the concept of containerised execution environments will be embraced for the Secure and Private Space that besides the portability and interoperability features, it also enables deployment over virtualised infrastructure and orchestration support. Furthermore, this concept facilitates the monitoring, autoscaling and management of the deployed applications.

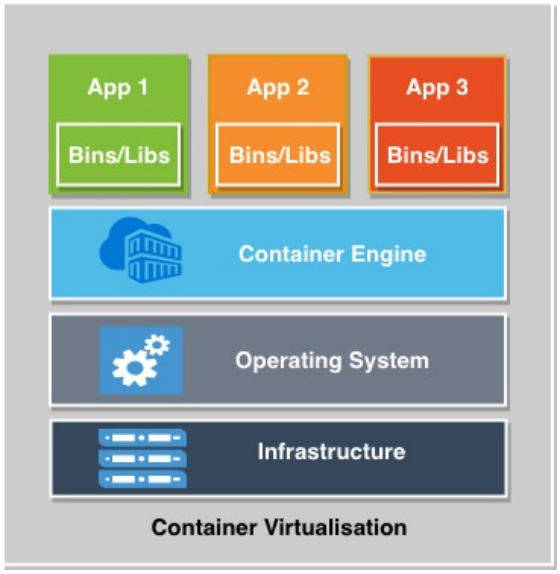

Figure 5-12: Container virtualisation

While multiple container technologies exist nowadays, Docker[1] is the one with the widest adoption by both the research and vendor communities. Docker is built on top of Linux kernel, namespaces, cgroups, chroot and file systems constructs. The idea behind Docker is that all containers on a given host utilise the same kernel, however the application resources are isolated per container. Docker containers are lightweight, standalone, self-contained systems that include everything that is needed for the proper execution of the system on a shared operating system such as code, runtime, system tools, system libraries and settings. Docker provides also an extensive toolset in order to manage the

[1] https://www.docker.com/

lifecycle of the containers with tools such as the Docker Engine, the Docker Swarm and the Docker Compose that will be explored within the context of the Resource Orchestrator.

The **main functionalities of the Resource Orchestrator** are the following:

- Connect to the configured virtualised infrastructure (e.g. OpenStack) and perform continuous monitoring and management of the resources.
- Access the available VM instance types and instances and be able add new VM instance types.
- Provide the means to allocate and release the needed resources in VMs, as well as to boot and stop virtual VM instances.
- Deploy and manage the containerised applications or services on the spawned VMs on the virtualised infrastructure in all their lifecycle.
- Provide tools such as service discovery for the services running on the VMs and health check operations in order to determine service availability.
- Be able to execute remote commands on the spawned VMs and handle the interactions between the various applications and services on a spawned VM.

The Resource Orchestrator is composed by two main sub-components, the **Resource Supervisor** and **Cloud Orchestrator** with the following main functionalities:

- The **Resource Supervisor** is undertaking the role of connecting to virtual infrastructures, on-boarding the available resources, performing continuous monitoring and management of the resources and supporting of the Cloud Orchestrator towards the deployment and management of the deployed applications or services contained within the virtual machines.
- The **Cloud Orchestrator** is responsible for the creation of the required deployment artefacts for the deployment of the applications or services, as well as the orchestration of the deployment and management of them in the virtualised infrastructure utilising the Resource Supervisor and a set of supplementary monitoring and reporting applications running locally on the virtual machines called Agents.

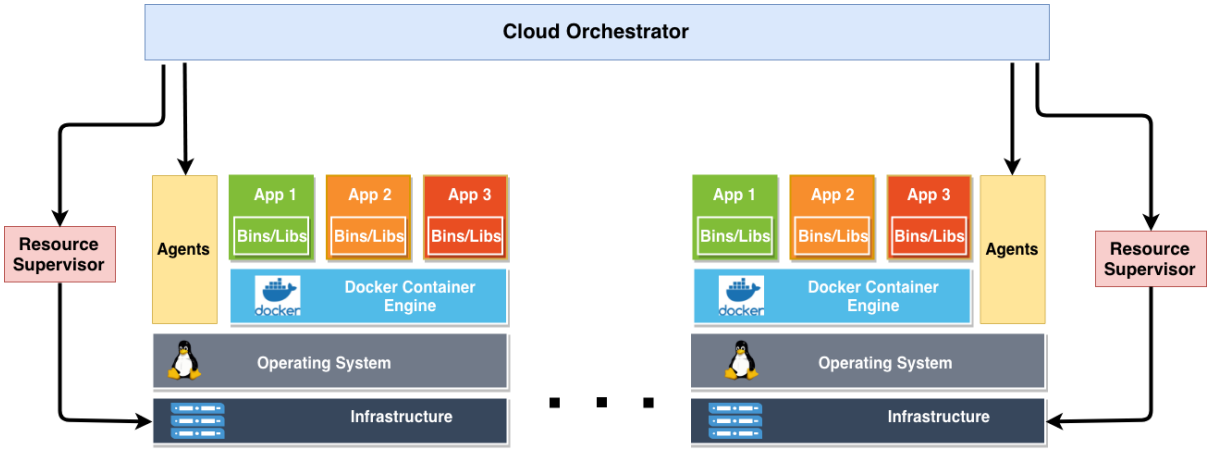

Figure 5-13: Resource Orchestrator overview

The Resource Supervisor is responsible for providing the appropriate infrastructural resources to the Cloud Orchestrator. As such, the Resource Supervisor is managing and monitoring the infrastructural resources that the are available for usage by connecting to virtual infrastructures and retrieving the relevant information. Resource Supervisor offers a variety of operations including, but not limited to, resource allocation and de-allocation, access to the available VM instance types and instances, creation of new VM instance types, booting and stopping of VM instances.

The Cloud Orchestrator is responsible for the deployment of the containerised applications or services on a virtualised infrastructure. More specifically, the Cloud Orchestrator utilises the Resource Supervisor in order to perform the deployment and management of the lifecycle of the containerised applications or services on spawned VMs on the virtualised infrastructure. As such, the Cloud Orchestrator provide the containers to the target VMs and is able to orchestrate the process of deploying, starting, stopping and deleting a containerised application or service, in addition to being able to always monitor and be aware of its status.

5.3.18.2 Addressed requirements

The Resource Orchestrator with the set of functionalities described in the component's design addresses the following requirements from the list of technical requirements that are documented in section 4.1:

- **TR_061:** The Resource Orchestrator is facilitating the effective execution of data analysis offering the tools and services with dynamic resource allocation, monitoring and management.
- **TR_066:** The Resource Orchestrator is enabling the deployment of a Secure and Private Space in the form of dedicated spawned VMs for the user that are utilised in the execution of data analysis.

5.3.19 Jobs Scheduler and Execution Engine

5.3.19.1 Design and Functionalities overview

The Jobs Scheduler and Execution Engine is the component in charge of initiating, executing the analytics jobs as provided by the Analytics and Visualisation Workbench, as well as of managing the resources available to the Execution Cluster nodes in the context of a Secure and Private Space. The analytics jobs are allocated to the Execution Cluster nodes, decoupling the invocation of a data analysis workflow coming from the Analytics and Visualisation Workbench from its execution.

Under the hood, the Jobs Scheduler and Execution Engine acts as a cluster manager and it takes in consideration resources such as memory, computation and network bandwidth to guarantee best performances. The Jobs Scheduler and Execution Engine deploys, scales and manages the nodes involved in analytics jobs execution and interacts with a set of local workers running on the Execution Cluster nodes for distributed computation which may involve the adoption of multifarious open

source cutting-edge trending technologies. The Jobs Scheduler and Execution Engine is also responsible for monitoring the execution of the job and for reporting the execution status to the Analytics and Visualisation Workbench. The results of the job execution are provided to the Encryption Manager in order to be encrypted. The encrypted results are then provided to the Core ICARUS platform for storage and retrieved by the Analytics and Visualisation Workbench in order to be decrypted and displayed to the user in the form of various visualisation types.

The Execution Cluster, that is managed by the Jobs Scheduler and Execution Engine, is the cluster-computing framework of the ICARUS platform and is deployed within the Secure and Private Space. The Execution Cluster offers the powerful processing engine that enables the data analysis execution across multiple datasets and support the extended list of data analysis algorithms that span from simple statistical analysis to more advance and complex machine learning and deep learning algorithms. The cluster-computing framework is offering a set of key characteristics that are crucial for the successful operation of the ICARUS platform such as speed, efficiency, reliability, fault tolerance and effective distributed job execution.

Both the Jobs Scheduler and Execution Engine and the Execution Cluster are running on the sandboxed environment provided by the Secure and Private Space.

In brief, **the core functionalities of the Jobs Scheduler and Execution Engine** are as follows:

- Perform the analytics jobs execution as instructed by the Analytics and Visualisation Workbench
- Perform the cluster management of the Execution Cluster, which is the cluster-computing framework of the ICARUS platform, in order to perform effective and efficient data analytics job execution with support for distributed computation across the nodes of the Execution Cluster.
- Monitor and the report the status of the job execution to the Analytics and Visualisation Workbench.
- Provide the results in the Encryption Manager for encryption before they are transferred to the Core ICARUS platform for storage.

5.3.19.2 Addressed requirements

- **TR_045, TR_059**: The Jobs Scheduler and Execution Engine enables the integration or aggregation of multiple datasets towards the effective and efficient data analysis execution.
- **TR_048**: The Jobs Scheduler and Execution Engine enables the combination multiple datasets originating from different data sources in the execution of a data analytics job.
- **TR_049**: The Jobs Scheduler and Execution Engine provides the monitoring mechanism for providing the progress and execution status to the Analytics and Visualisation Workbench for display.

- **TR_050, TR_051, TR_052, TR_053, TR_054, TR_055:** The Jobs Scheduler and Execution Engine supports the execution of an extensive list of data analysis algorithms, which span from simple statistical analysis algorithms to more complex and advanced analysis algorithms such as machine learning and deep learning algorithms, through the Execution Cluster.
- **TR_057, TR_058:** The Jobs Scheduler and Execution Engine is supporting the analytics job execution in scheduler manner, as provided by the Analytics and Visualisation Workbench, by offering the scheduling mechanism that initiates the analytics job execution in a selected date and time.
- **TR_060:** The Job Scheduler and Execution Engine is enabling the analytics job execution in a scalable and reliable manner utilising the capabilities of the Execution Cluster.

5.3.20 Notification Manager

5.3.20.1 Design and Functionalities overview

The Notification Manager is the component providing the updated information to the users with regards to the datasets or the scheduled analytics jobs. More precisely, this component is responsible to notify the users of the ICARUS platform of events that have happened. These events refer to the addition of new data assets related to the users' preferences, updates on existing data assets in which the users have active contracts and changes on the users' execution status of analytics jobs.

Based on the users' configured preferences, Notification Manager will notify users for the addition of new data assets that are relevant to them. Furthermore, notifications will be sent for any updates on the data assets which the users are entitled to use. These updates can be either updates on the terms of use of the data assets or modifications in the data. Moreover, the users will be notified for any updates on the execution status of their scheduled analytics jobs such as task completion, resource exhaustion errors and failures.

The Notification Manager by default, will show only relevant notifications and not unnecessary ones. However, the users are able to configure their notification settings and hide specific notifications or even prioritise notifications about specific topics. In this way, they can keep their notification tab in a cleaner and tidy status.

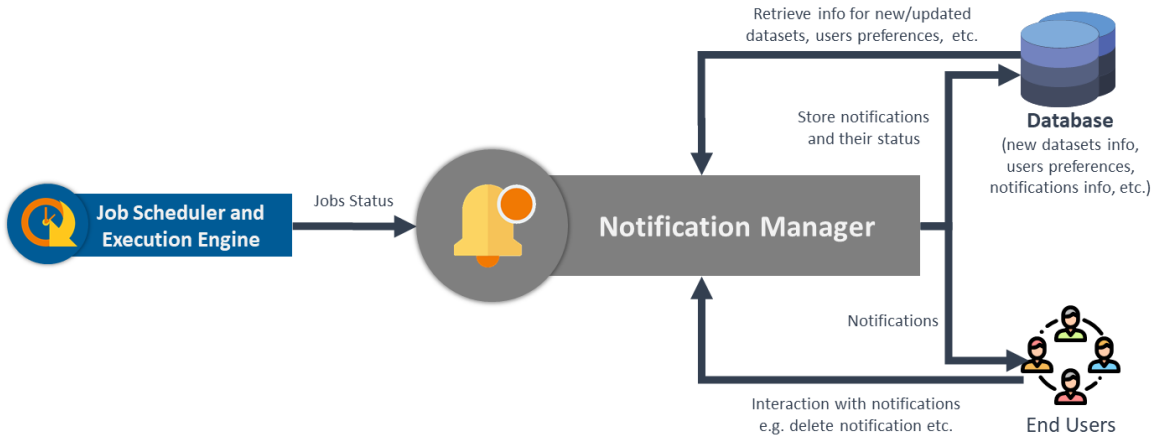

Figure 5-14: Notification Manager overview

The **main functionalities of the Notification Manager** are the following:

- When a new data asset is registered in the platform, access the information about the users' preferences and notify the users that are interested in topics related to the new data asset.
- When an existing data asset is updated in the platform, access the information about the data assets of the users (either owned or purchased) and notify the users that are entitled to use it.
- Connect to the Job Scheduler and Execution Engine and notify the users about the execution status of their scheduled analytics job.
- Store the notifications of each user in order to allow the users to access them anytime. Blocked notifications are hidden together in one place for later check.

5.3.20.2 Addressed requirements

The Notification Manager, as described in the component's design, addresses the following technical requirements which are listed in section 4.1:

- **TR_062**: The Notification Manager allows users to configure their notification settings, hide specific notifications and prioritise notifications about specific topics.
- **TR_063**: The Notification Manager notifies users with active contracts on a data asset when either its terms of use are updated or its data are modified.
- **TR_064**: The Notification Manager notifies the users when their scheduled analytics jobs are completed either with success or failure.

5.3.21 Usage Analytics

5.3.21.1 Design and Functionalities overview

The Usage Analytics is the component responsible for the detailed analysis of users' interactions within the ICARUS platform. This component provides the tools for collecting, analysing and visualising the usage of the various services and data assets of the platform in order to extract useful insights and statistics.

Usage Analytics retrieves statistic information from the platform about the user's behaviour in various levels such as the usage and adoption of specific features or services and the usage of each dataset or algorithm. By analysing this information for every user, it provides aggregated statistics to both the users and the platform administrator, enabling them to better understand users' behaviour at various levels, such as which features are adopted and which are ignored.

For instance, indicative metrics for the data assets can refer to the number of views, number of purchases, number of appearances in search results, number of analytical tasks that were applied on it (if available) and so on. As for the services (algorithms, visualisation types, etc.), indicative metrics can be related to the number of times the service was used, number of users that utilised the service, last time the service was used, etc. Furthermore, platform specific metrics are also considered, like the number of active/total users, number of active/total sessions, number of new users per month, etc.

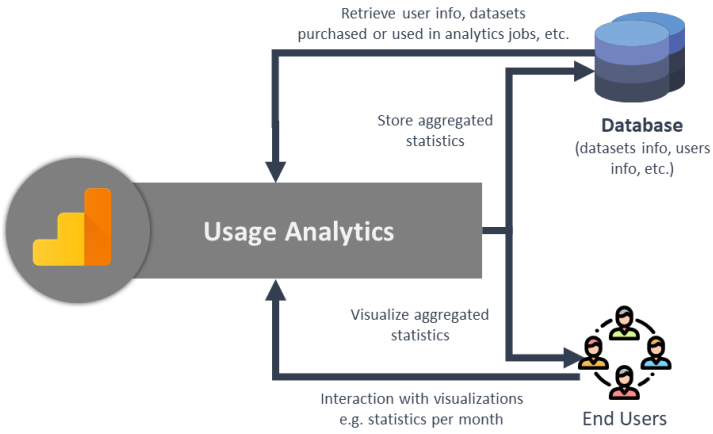

Figure 5-15: Usage Analytics overview

The **main functionalities of the Usage Analytics** are the following:

- Access and retrieve the information for each user about each data asset purchased or used in analytics jobs, algorithm applied for any task, visualisation scheme for showing original or modified data.
- Analyse the retrieved information and produce aggregated statistics about each data and service asset, without revealing any information of the individual users.

- Visualise the aggregated statistics in an intuitive dashboard for both users and the platform administrator.

5.3.21.2 Addressed requirements

The Usage Analytics, as described in the component's design, address the following technical requirements which are listed in section 4.1:

- **TR_065**: The Usage Analytics component provides aggregated statistics about the usage of the assets (either data or service assets), as well as platform specific metrics

6 Conclusions & Next Steps

The purpose of the deliverable at hand (D3.1 "ICARUS Architecture, APIs Specifications and Technical and User Requirements ") was to deliver the user requirements and the technical requirements of ICARUS, as well as to deliver the first version of the conceptual architecture of the ICARUS platform.

At first, the ICARUS agile development methodology was presented, describing all the processes, instruments, roles and methods that are adopted in all the phases of the development of the ICARUS platform. Within this methodology, the User Stories definition was clearly defined providing all the guidelines and the additional management information that were used as a guidance during the process. Moreover, the requirements definition in terms of key characteristics and requirements classification was presented, along with the ICARUS stakeholders and their interactions with the ICARUS platform.

In accordance with this methodological approach, the User Stories, that that are stemming directly from the demonstrator partners of the ICARUS project were collected in collaboration with technical partners. These User Stories presented the expected behaviour of all sub-systems of the platform from the end-user perspective and were provided as input for the user requirements elicitation process.

From these User Stories, the user requirements were extracted ensuring the compliance with the requirements characteristics defined in the methodology, while also taking into consideration the MVP features, as defined in D1.2. These extracted user requirements were classified into platform and demonstrator functional requirements and non-functional requirements.

The list of functional and non-functional user requirements was analysed thoroughly in order to extract the list of the ICARUS technical requirements. The list includes also a set of additional technical requirements that derived directly from the feedback received from the external ICARUS MVP validation. Hence, the elicited technical requirements consolidate the ICARUS MVP, as they span the phases of the ICARUS methodology and the MVP features, as defined in D1.2, and the requirements from the ICARUS consortium and from the external stakeholders and the theoretical approaches defined in WP2. These concrete and solid technical requirements were provided as input in the design and specification definition of the components of the ICARUS architecture. Within the scope of this deliverable the complete requirement backlog has been provided for ICARUS.

A comprehensive analysis of these technical requirements provided the design of the first version of the conceptual architecture of the integrated ICARUS platform. The analysis of the technical requirements that derived from the external ICARUS MVP validation introduced several architectural decisions and multiple iterations in the design process in order to ensure that the security-related requirements of the external aviation stakeholders are properly addressed. The ICARUS architecture is a modular architecture, composed by a set of key components with distinct roles and scope towards the aim of providing the envisioned platform features that will address the ICARUS stakeholders' needs. Each component was carefully designed having in mind that it should address a specific set of

technical requirements from the list of the ICARUS technical requirements. For each component a comprehensive description of the design and functionalities has been documented.

It should be stressed at this point that the current deliverable presents the first version of the ICARUS conceptual architecture, as well as the user and technical requirements. These outcomes will drive the implementation phase of the ICARUS platform that will be performed within the context of WP4. However, as the design of the ICARUS architecture and the identification and analysis of the functional and non-functional requirements, as well as their translation into technical requirements, is a living process that will last until M32, the forthcoming versions of this deliverable will include updates on both the architecture and the components of the architecture based on the feedback received.

Annex I: References

Cohn, M. (2010). Agile Softwareentwicklung: mit Scrum zum Erfolg!. Pearson Deutschland GmbH.

Ericson, C. A. (2015). Hazard analysis techniques for system safety. John Wiley & Sons.

"NIST Guide to ABAC Definitions". [Online] Available: http://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.sp.800-162.pdf [Accessed: 05-01-2019]

"Non Functional Requirements" [Online] Available: https://www.scaledagileframework.com/nonfunctional-requirements/ [Accessed: 22-03-2019]

"Understanding virtualisation" [Online] Available: https://www.redhat.com/en/topics/virtualization [Accessed: 10-12-2018]

"XACML Oasis Specification". [Online] Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml [Accessed: 15-11-2018]

Annex II: ICARUS User Stories

ID	Category	User Story			Priority	Value	Acceptance
		As a <type>	I want to <user requirement>	So that <reason>			
PACE_001	Collection	Data consumer	retrieve periodically the updates automatically from data providers	it is ensured that data are always up-to-date	Medium	Medium	ICARUS provides any available updates on a dataset from a data provider automatically by performing periodic checks
PACE_002	Collection	Data consumer	have shortcuts to my most frequent activities and workflows	I am more efficient.	Medium	Medium	ICARUS platform provides shortcuts with the most used workflows or user activities
PACE_003	Collection	Data consumer	Have data categorised into different authorisation level	I can decide upon the private and public visibility of my data.	Medium	Medium	ICARUS should provide the means to set different authorisation level to the different datasets
PACE_004	Analytics	Data consumer	to be able to use the ICARUS platform at any time	that I can execute or perform my workflows with the acceptable level of performance	Medium	Medium	ICARUS Platform should be operational at any time with the acceptable level of performance.
PACE_005	Exploration	Data consumer	Query for data by certain characteristics	I can collect data for my route network analysis.	Medium	Medium	Users should be able to search and find data based on selected characteristics such as type, key words, time frame.
PACE_006	Exploration	Data consumer	query for data only under my authorisation level	I can be sure to use data that I am entitled to use only	Medium	Medium	User shall be able to query only the data they have access to

PACE_007	Analytics	Data Analytics	Conduct business data analytics	I can enrich my data with more advanced information.	Medium	Medium	Users should be able to choose data, conduct analytics and store the results of the analysis
PACE_008	Exploration	Data consumer	Query airport data by name, city and IATA/ICAO code	I can collect airport data for my route network analysis.	Medium	Medium	Users should be able to search and find data e. g. for airports.
PACE_009	Exploration	Data consumer	Query flight information data by airline, date and flight number.	I can collect flight information for my route network analysis.	Medium	Medium	Users should be able to search and find historical and scheduled flights.
PACE_010	Exploration	Data consumer	Query airport weather data for distinct time frames	I can collect historical weather data for a statistical analysis.	Medium	Medium	Users should be able to search and find historical airport weather.
PACE_011	Collection	Data consumer	be able to download the data in machine readable formats	I can import the data in other systems.	Medium	Medium	User should be able to download data or the results of the query executed.
PACE_012	Recommendation	Data consumer	receive suggestions for related data	I can access additional data to my query	Medium	Medium	User should be informed for relative data along with the query results. .
PACE_013	Exploration	Data consumer	for a particular airport to get related runway and obstacle data	I can collect additional data for my route network analysis.	Medium	Medium	ICARUS should be able to provide runway and obstacle data for a specified airport.
PACE_014	Collection	Data provider	upload structured data to the ICARUS platform	I can make my data available to other ICARUS users.	Medium	Medium	Users should be able to upload and share data with other users.
PACE_015	Linking	Data provider	enrich my uploaded data with additional information	I can link my data with other relative data	Medium	Medium	ICARUS should be able to enrich my data and link my data with other relative data .
PACE_016	Harmonisation	Data provider	easily modify, delete and replace my data assets on the ICARUS platform	I can maintain my data provision.	Medium	Medium	Update, delete and change existing data.

PACE_017	Sharing	Data provider	Be able to manage the confidentiality for my data	I that no one is violating my data IPs.	Medium	Medium	Users should be able to set and modify the sharing policy of my data.
PACE_018	Collection	Data provider	Have data curations mechanisms provided by the ICARUS platform	I can improve the quality of my data.	Medium	Medium	ICARUS should provide the mechanisms to upload and improve the quality of my data.
PACE_019	Collection	Data provider	Be able to perform various anonymisation techniques	I can apply them on my uploaded data in order to comply with privacy and licensing requirements of my region.	Medium	Medium	During upload ICARUS enables the execution of several anonymisation techniques in order to address the privacy and licensing requirements of my region.
PACE_020	Collection	Data provider	have anonymisation mechanisms provided by the ICARUS platform	I can ensure the privacy of my data.	Medium	Medium	ICARUS should provide the means for ensuring data privacy
PACE_021	Collection	Data provider	upload airport data from individual sources to ICARUS platform	other users can search and find data by certain criteria.	Medium	Medium	Data should be loaded correctly to the ICARUS platform.
PACE_022	Linking	Data provider	Be able to establish connections between flight information and with airport data assets	another user can get linked data suggestions.	Medium	Medium	Data from various sources should be linkable.
PACE_023	Analytics	service asset consumer	Be able to work in a secured space	I can use ICARUS service assets also for my confidential data	High	High	A user should be able to analyse its confidential data in secure space
AIA_001	Linking	Data provider	Connect flight information data with airport data	I can get linked data suggestions.	Medium	Medium	Data from various sources should be linkable.
AIA_002	Analytics	Data consumer	Login to a secure space	I can analyse my confidential data	High	High	A user should be able to analyse its confidential data in secure space

AIA_003	Analytics	Data consumer	Integrate with an easy way the ICARUS platform flight information related data with the AIA airport data	I can use them as the base data for future planning.	High	High	A user should be able to analyse the ICARUS platform flight information related data with the AIA airport data in order to receive forecasts
AIA_004	Analytics	Data consumer	Integrate with an easy way the ICARUS platform flight information related data with the AIA airport data	have the ability to run different scenarios on how the airport may operate using different schedules	High	High	A user should be able to analyse its data to run analysis and future simulations.
AIA_005	Analytics	Data consumer	Integrate with an easy way the ICARUS platform flight information related data with the AIA airport data	check any potential airport capacity constraints against pre-defined KPIs	High	High	A user shall be able to process automatically the results of the analysis against predefined KPIs
AIA_006	Analytics	Data consumer	Have graphical and tabular form report	I can gain future traffic information taking into consideration all flight time related attributes in a visual and graphical format	Medium	High	A user should be able to create and use visualisations (dashboards , graphs etc) to assess results picking the interval and timeframe for the statistical analysis.
AIA_007	Notification	Data consumer	Be notified when relevant data become available	I can examine it without missing potentially important opportunities	Low	Medium	When new datasets are available the system notifies potentially interested consumers
AIA_008	Recommendation	Service asset consumer	Proposition of additional data assets (either own or from different data providers)	I can perform analysis and define trends	Low	Medium	Proposal of additional data sources when building reports or analytics

AIA_009	Sharing	Data provider	Functionality to set a license of a data asset	I can set the terms of use for each data asset	High	High	A user should be able to build a data sharing license.
AIA_010	Sharing	Data provider	Step-by-step guidance on how to define the appropriate license of a data asset	One can easily define and enforce the terms of use for each data asset	High	High	A user should be guided when building of a data sharing license.
AIA_011	Sharing	Data provider	Ability to activate and cancel a data sharing agreement	I can easily activate and deactivate a data sharing agreement	High	High	A user should be able to activate and deactivate a data sharing agreement
AIA_012	Analytics	Data consumer	Ability to run Simulation scenarios	Run a forecast schedule within the current operational resources as a test scenario	Medium	High	A user should be able to demonstrate or identify the areas where capacity will be constrained.
AIA_013	Analytics	Data consumer	Ability to create and run what-if scenarios	For the purpose of master-planning exercises.	Medium	High	A user should be able to run these scenarios based on virtual flights schedules and/or a set of business and resources' rules.
AIA_014	Sharing	Data provider	the ICARUS to offer logging and auditing mechanisms	In order the stakeholders to be able to resolve any disputes	High	High	A user should be able to audit logging and resolve any disputes
AIA_015	Analytics	Data consumer	Have a Dashboard, graphical and tabular form report	I can visualise the number of aircraft on ground per hour and 15 minutes as per ICAO Category	High	High	A user should be able to use the visualisation to assess results picking the interval and timeframe for the statistical analysis.
AIA_016	Analytics	Data consumer	Have a Dashboard, graphical and tabular form report	I can visualise the number of departures or arrivals per hour & per 15 minutes	Medium	High	A user should be able to use the visualisation to assess results picking the interval and timeframe for the statistical analysis.

AIA_017	Analytics	Data consumer	The ability to create dashboards	Monitor the progress of the analytics tasks. Present the results in a graphical and consolidated manner	Medium	High	A user should be able to use the visualisation to assess results picking the interval and timeframe for the statistical analysis.
CEL_001	Collection	Data Consumer	Query for check-in information (check-in counters, luggage drop-off counter)	I can analyse the arrival patterns at the airport	Medium	Medium	Users should be able to search and find check-in and luggage information
CEL_002	Collection	Data Consumer	Query for passport control and security scan times	I can analyse time for standard procedures and predict bottlenecks	Medium	Medium	Users should be able to search for passport control and security times.
CEL_003	Collection	Data Consumer	Query for weather conditions at the destination (historical data and latest updates)	I can collect historical weather data for statistical analysis	Medium	Medium	User should be able to search and find historical weather info
CEL_004	Collection	Data Consumer	Query for gate info and flight delays	I can analyse behaviour patterns at the airport	Medium	Medium	The user should get the gate information and delays with a short answering time
CEL_005	Collection	Data Consumer	Query for transit info (gate, flight delays)	I can perform a statistical analysis for in-transit passengers	Medium	Medium	The user should be able to search and find transit flight information and gate
CEL_006	Collection	Data Consumer	Query for luggage belt and time	I can collect additional data for my luggage time pickup analysis.	Medium	Medium	The user should be able to query luggage belt and time
CEL_007	Collection	Data Consumer	Get booking data	I can collect and analyse personalised profiles of the traveller	High	High	The user should be able to receive Booking data

CEL_008	Sharing	Data/Asset provider	be able to upload my data/assets to the ICARUS platform	other users of the platform can purchase my data according to my sharing policy	High	High	ICARUS should support transactions between the users for data/assets purchase
CEL_009	Analytics	Data Scientist	Use a Dashboard to combine and run multiple reports	Perform an analysis of the results	High	High	A user should be able to use visualisation tools for statistical analysis
CEL_010	Notification	Data consumer	Notified for any data update that I am using	I can re-run my analysis based on the new data	High	High	The ICARUS platform notifies the user about updates on the data
ISI_001	Collection	Data consumer / Scientist	Access the platform and download data about passenger stratification on multiple routes, according to the data license	I can feed the data to the epidemiological simulation framework for improved modeling of human mobility	High	High	Demographic data are available to consumers through ICARUS platform via proper ACL
ISI_002	Collection	Data consumer / Scientist	Access the platform and download data about flight return tickets/bookings, according to the data license	I can feed the data to the epidemiological simulation framework for improved modeling of effective force of infection	High	Medium	Estimates about distribution of length of stay are available to consumers through ICARUS platform via proper ACL
ISI_003	Collection	Data consumer	Have the platform automatically perform quality check and profiling on the available data	I can easily decide if the dataset is of interest and how I could import it into my framework	Medium	Medium	Datasets on the platform are properly annotated and verified
ISI_004	Notification/Linking	Data consumer	Be notified when relevant data become available according to my	I can examine it without missing potentially important opportunities	Low	Low	When new datasets are available the system notifies potentially interested consumers

			preferences / interests				
ISI_005	Exploration	Data consumer	Be able to analyse extracts of available datasets to assess their actual value or interest	I can evaluate their importance/fitting for my work	Medium	Medium	Dataset extracts are accessible for evaluation on the platform
ISI_006	Notification	Data consumer	Be informed about any update or modification of the license/terms of usage of datasets I am using or interested into	I can take immediate action if needed and get new relevant data when possible	Low	Medium	The platform notifies the users about any update on data terms of usage
ISI_007	Sharing	Data consumer / Data provider	Have an automatic/guided mechanism to help dealing with data licensing	I can choose the proper license being sure I am compliant with the licenses of interconnected datasets	Medium	Medium	The platform performs cross checking of the data licensing options and provides guidance for assigning new licenses

Annex III: ICARUS User Requirements

ID	Description of the requirement	Category	Related User Stories	Feature ID	Requirement Type
Req_001	ICARUS should inform users for updates on datasets.	Notification	PACE_001 AIA_007 CEL_003 CEL_010 ISI_004	PLATF_F_01 PLATF_F_02 PLATF_F_48	PF
Req_002	ICARUS should support connections to various APIs for data exchange (import/export)	Collection	PACE_001 PACE_014 CEL_008 CEL_002	PLATF_F_01 PLATF_F_02	PF
Req_003	ICARUS should provide functions to create and manage shortcuts and workflows related to the user's recent actions or workflows.	Analytics	PACE_002 AIA_001	PLATF_F_30 PLATF_F_51 PLATF_F_52 PLATF_F_53	PF
Req_004	ICARUS should offer a public and a proprietary and confidential working space.	Analytics	PACE_023 AIA_002	PLATF_F_46 PLATF_F_47	PF
Req_005	ICARUS should support tags for datasets in addition to categories.	Enrichment Linking	PACE_005 PACE_008 PACE_009 CEL_001 CEL_002 CEL_005	PLATF_F_16 PLATF_F_20	PF
Req_006	ICARUS should support filters for tagged datasets (i.e. real-time data, historical, proprietary, public, demo/preview, etc).	Linking	PACE_005 PACE_008 PACE_009 CEL_001 CEL_002 CEL_005	PLATF_F_18	PF

Req_007	ICARUS analytics should work with a mixture of confidential and public data.	Collection, Exploration Analytics	PACE_003 AIA_003 AIA_004 AIA_005 AIA_012	PLATF_F_04 PLATF_F_11 PLATF_F_12 PLATF_F_13 PLATF_F_26 PLATF_F_29 PLATF_F_30 PLATF_F_32 PLATF_F_40	PF
Req_008	ICARUS platform should have high availability	Reliability	PACE_004		NF
Req_009	ICARUS should support to search for datasets by type	Exploration	PACE_005 PACE_008 PACE_009 PACE_010 CEL_001 CEL_002 CEL_003 CEL_004 CEL_005	PLATF_F_22 PLATF_F_23	PF
Req_010	ICARUS should support to search for datasets by keywords	Exploration	PACE_005	PLATF_F_22 PLATF_F_23	PF
Req_011	ICARUS should provide data sets which are relevant to my search.	Linking, Recommendation	PACE_005 PACE_015	PLATF_F_18 PLATF_F_51	PF
Req_012	ICARUS should support to search for datasets by date and time.	Exploration	PACE_005	PLATF_F_22 PLATF_F_23	PF
Req_013	ICARUS should be able to support search for historical flight information	Exploration	PACE_005 PACE_009	PLATF_F_22 PLATF_F_23	DF
Req_014	ICARUS should be able to integrate flight information data with flight number, airline, date and time of departure/arrival.	Curation	PACE_009	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF

Req_015	ICARUS should be able to integrate airport weather data by period with airport identifier (IATA/ICAO code, airport/city name).	Curation	PACE_010 CEL_003	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_016	ICARUS platform should ensure that the users can query data that they are authorised to access.	Exploration Analytics	PACE_006 ISI_001 ISI_002	PLATF_F_26	PF
Req_017	ICARUS platform should offer different level of confidentiality for the datasets.	Collection, Exploration Analytics	PACE_003 PACE_021 AIA_002	PLATF_F_04 PLATF_F_26	PF
Req_018	ICARUS platform should provide a set of advanced analytics algorithms.	Analytics	PACE_007 AIA_003 AIA_004 AIA_005 CEL_009 ISI_005	PLATF_F_33 PLATF_F_34	PF
Req_019	ICARUS platform should provide features to either customise the defined analytics algorithms or define custom analytics algorithms.	Analytics	PACE_007 AIA_003 AIA_004 AIA_005 CEL_009 ISI_005	PLATF_F_29	PF
Req_020	ICARUS platform should provide private space where I can store the data obtained through a query.	Analytics	PACE_023 AIA_002 AIA_003	PLATF_F_43 PLATF_F_46 PLATF_F_47	PF
Req_021	ICARUS should be able to integrate airport data with IATA/ICAO code, airport or city name	Curation	PACE_008	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_022	ICARUS should support file upload and download services for common text formats such as ASCII, CSV, XML, YAML, JSON	Collection, Exploration	PACE_011 PACE_014 CEL_008	PLATF_F_41 PLATF_F_42 PLATF_F_43	PF
Req_023	ICARUS platform should allow the user to select a file format for download if a conversion is feasible.	Exploration	PACE_011 CEL_007	PLATF_F_27	PF

Req_024	ICARUS platform should suggest available data from other sources related to my queries.	Recommendation	PACE_012 AIA_008 ISI_004	PLAT_F_23	PF
Req_025	ICARUS should be able to integrate obstacle data with runways and runway data with airports	Curation	PACE_013	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_026	ICARUS should be able to combine airport data, runway data and obstacle data and download them as a file	Curation	PACE_013	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_027	ICARUS platform should be able combine data and provide the means to obtain them.	Collection, Exploration	PACE_013 CEL_007 ISI_001 ISI_002	PLATF_F_42	PF
Req_028	ICARUS should support the upload of external data sets	Collection	PACE_014 PACE_021 CEL_008	PLATF_F_03	PF
Req_029	ICARUS should provide data conversion for uploaded file format into ICARUS platform internal data format.	Curation	PACE_014 PACE_021	PLATF_F_11 PLATF_F_12 PLATF_F_13 PLATF_F_27	PF
Req_030	ICARUS platform should provide features for adding additional (semantic) information to data assets	Linking	PACE_015	PLATF_F_16 PLATF_F_17	PF
Req_031	The user should be able to explore the ICARUS data model and can provide suggestions to the data administrator.	Curation	PACE_015 PACE_016 CEL_008	PLATF_F_11 PLATF_F_12 PLATF_F_13	PF
Req_032	ICARUS platform should provide features for updating my datasets	Curation	PACE_016	PLATF_F_14	PF
Req_033	ICARUS platform should inform a user if data, for which the user owns a license to use, have been updated or deleted.	Notification	AIA_007 CEL_11 ISI_004	PlATF_F_48 PlATF_F_49	PF
Req_034	ICARUS platform should provide features for defining and modifying the license model of my data.	Sharing	PACE_017 AIA_009 AIA_010 AIA_011 ISI_002 ISI_006 ISI_007	PLATF_F_56 PLATF_F_57 PLATF_F_58 PLATF_F_59	PF

Req_035	ICARUS should provide indicative usage analytics on the datasets usage within the platform	Analytics	PACE_017	PLATF_F_45	PF
Req_036	ICARUS platform should support the negotiations between data provider and data consumer until the agreement has been signed.	Sharing	PACE_017 PACE_019 AIA_010 CEL_008 ISI_007	PLATF_F_60 PLATF_F_61 PLATF_F_62	PF
Req_037	ICARUS platform should provide the means to improve the quality level of the user's data	Collection	PACE_018 ISI_003	PLATF_F_06 PLATF_F_07	PF
Req_038	ICARUS platform should provide mechanisms to define the licensing requirements and privacy restrictions (DSGVO/GDPR compliance) for a dataset	Collection	PACE_019 PACE_020	PLATF_F_08 PLATF_F_09 PLATF_F_10	PF
Req_039	ICARUS platform should provide a tool for data anonymisation.	Collection	PACE_020	PLATF_F_08	PF
Req_040	ICARUS should be able to support the upload of external airport, runway and obstacle data	Collection	PACE_021	PLATF_F_01 PLATF_F_02	DF
Req_041	ICARUS should be able to support to upload external airport weather data	Collection	PACE_021	PLATF_F_01 PLATF_F_02	DF
Req_042	ICARUS should support a simplified upload of data, if it used only by myself.	Collection	PACE_014 PACE_021 CEL_008	PLAT_F_05	PF
Req_043	ICARUS should be able to link different datasets	Curation	PACE_015 PACE_022 AIA_001 AIA_003 AIA_004 AIA_005	PLATF_F_16	PF
Req_044	ICARUS should support know-your-customer practices, with organization registration and user login with credentials	Security	PACE_023 AIA_002	PLATF_F_46	NF
Req_045	ICARUS platform should provide features for transferring data from my confidential space to the ICARUS platform	Collection, Analytics	PACE_023	PLATF_F_43 PLATF_F_46 PLATF_F_47	PF

Req_046	ICARUS platform should provide the functionality to manage (semantic) links between data assets.	Linking	PACE_015	PLATF_F_16	PF
Req_047	ICARUS platform should be able to suggest integrated data sets in the context of queries and data uploads.	Linking	PACE_012 AIA_008	PLATF_F_17	PF
Req_048	ICARUS should be able to integrate flight information data with internal AIA airport data	Curation	AIA_003	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_049	ICARUS should provide business intelligence tools that enable the automated generation of event driven alerts and customised reports and notify user about results.	Analytics, Notification	AIA_002 AIA_006	PLATF_F_37 PLATF_F_38 PLATF_F_41	PF
Req_050	ICARUS should be able to integrate structured and unstructured data	Collection	AIA_002	PLATF_F_14	PF
Req_051	ICARUS should be able to report and visualise analysis results.	Analytics	AIA_004 AIA_012 AIA_015	PLATF_F_37 PLATF_F_38	PF
Req_052	ICARUS should allow to use look up tables (SXF is an IATA code for Berlin Schoenefeld Airport)	Analytics	AIA_012	PLATF_F_16	DF
Req_053	ICARUS should provide comprehensive means to visualise and to compare results (graphical, tabular, …)	Analytics	AIA_006 AIA_015 AIA_016 AIA_017 CEL_009	PLATF_F_37 PLATF_F_38 PLATF_F_39 PLATF_F_53	PF
Req_054	ICARUS should be able to execute analytics and workflows automatically (through pre-scheduled jobs).	Analytics	AIA_005	PLATF_F_29 PLATF_F_30 PLATF_F_40	PF
Req_055	ICARUS should provide dashboards and help the user compare the results with minimum number of interactions.	Analytics	AIA_006 AIA_015 AIA_016 AIA_017 CEL_009	PLATF_F_39	PF
Req_056	ICARUS platform should support connections to web services and provide API for the upload and download of	Collection	AIA_007	PLATF_F_01 PLATF_F_02 PLATF_F_48	PF

	data with other data sources and sinks.				
Req_057	ICARUS platform should provide a listing with all the available data sources and related information and among others the terms of use for each one of them.	Collection Recommendation	AIA_008	PLATF_F_03 PLATF_F_51	PF
Req_058	ICARUS platform should provide the monitoring, logging and auditing mechanisms in order for the stakeholders to be able to audit data usage and resolve any disputes	Sharing	AIA_014	PLATF_F_45	PF
Req_059	ICARUS should provide a guideline or a guidance to create an appropriate license definition and agreement.	Sharing	AIA_010 CEL_008 ISI_006 ISI_007	PLATF_F_55 PLATF_F_56 PLATF_F_57	PF
Req_060	ICARUS should provide the features for the management and update of data licenses.	Sharing	AIA_011 ISI_006	PLATF_F_58 PLATF_F_59	PF
Req_061	ICARUS should support notifications regarding the result of the execution of scheduled analytics	Analytics	PACE_003 AIA_003 AIA_004 AIA_005	PLATF_F_50	PF
Req_062	ICARUS should provide a GUI where the progress of processes and workflows can be monitored.	Analytics	PACE_003 AIA_003 AIA_004 AIA_005	PLATF_F_50	PF
Req_063	ICARUS should have the ability to create edit and run what if scenarios.	Analytics	AIA_013 AIA_016	PLATF_F_29 PLATF_F_30 PLATF_F_40	PF
Req_064	ICARUS platform should provide information about my data usage: which datasets, which algorithms, which reports.	Analytics	AIA_014	PLATF_F_45	PF
Req_065	ICARUS should be able to integrate aircraft on ground data with ICAO category	Analytics	AIA_015	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_066	ICARUS should provide features for statistical analyses.	Analytics	AIA_015 AIA_016 AIA_017 CEL_003 CEL_005	PLATF_F_29 PLATF_F_30 PLATF_F_40	PF

			CEL_009		
Req_067	ICARUS should provide means retrieve data from Amadeus Airport Operational Database (AODB)	Analytics	AIA_017	PLATF_F_01	DF
Req_068	ICARUS should be able to integrate flight number with airport data like check in counter, luggage information	Collection	CEL_001	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_069	ICARUS should be able to integrate flight number with security process information like passport control or security scan.	Collection	CEL_002	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_070	IARUS should be able to integrate airport locations with weather data, current weather as well as statistical weather	Collection	CEL_003	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_071	ICARUS should be able to integrate flight number with airport information and flight plan data (e. g. delays)	Collection	CEL_004	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_072	ICARUS should provide the last modification time of each dataset.	Collection	CEL_010	PLATF_F_01 PLATF_F_02 PLATF_F_48	PF
Req_073	ICARUS should be able to integrate flight data with connection flight information.	Collection	CEL_005	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_074	ICARUS platform should be able to integrate personal booking data with other data like fight information data, airport data, …	Collection	CEL_007	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_075	ICARUS should be able to anonymise data so that legal regulations can be considered.	Collection	CEL_007 ISI_001	PLATF_F_08 PLATF_F_09 PLATF_F_10	PF
Req_076	ICARUS should be able to assign costs to data assets.	Sharing	CEL_008	PLATF_F_56 PLATF_F_57	PF
Req_077	ICARUS should provide features for different forms of payments.	Sharing	CEL_008	PLATF_F_56 PLATF_F_57	PF
Req_078	ICARUS should provide the functionality to save and restore user defined configurations.	Analytics	PACE_002 CEL_009	PLATF_F_43	PF

Req_079	ICARUS platform should be able to integrate route information with passenger information.	Collection	ISI_001	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_080	ICARUS platform should be able to perform aggregations on data sets.	Collection	ISI_001	PLATF_F_14	PF
Req_081	ICARUS platform should be able to integrate passenger data with booking data	Collection	ISI_002	PLATF_F_11 PLATF_F_12 PLATF_F_13	DF
Req_082	ICARUS platform should be able to check that the data usage and delivery is compliant to the defined data access rights	Analytics	ISI_002	PLATF_F_32	PF
Req_083	ICARUS platform should provide data cleaning mechanisms.	Collection	PACE_018 ISI_003	PLATF_F_15	PF
Req_084	ICARUS platform should provide mechanisms for anonymisation and data cleaning	Collection	PACE_018 ISI_003	PLATF_F_10 PLATF_F_15	PF
Req_085	The platform should be able to integrate data sets based on common fields.	Linking	ISI_001	PLATF_F_18	PF
Req_086	ICARUS should support API for data export	Collection	PACE_001 CEL_003	PLATF_F_01 PLATF_F_02	PF
Req_087	ICARUS should support the provision of data updates.	Collection	PACE_016	PLATF_F_01 PLATF_F_02	PF
Req_088	ICARUS should support an extended list of algorithms on a mixture of confidential and public data in order to perform big data analytics	Functional Suitability	PACE_006 PACE_020 PACE_023 AIA_002 ISI_002	PLATF_F_28 PLATF_F_31 PLATF_F_32 PLATF_F_36	NF
Req_089	ICARUS should be able to execute big data analytics in a timely and efficient manner	Performanc e efficiency	PACE_004 PACE_010 AIA_017 ISI_005	PLATF_F_35	NF
Req_090	ICARUS should guarantee the efficient and effective resource allocation for the success analytics jobs execution	Performanc e efficiency	AIA_017 CEL_004	PLATF_F_35	NF
Req_091	ICARUS should be able to handle and store large datasets	Performanc e efficiency	PACE_005 PACE_010 ISI_002		NF

Req_092	ICARUS should enable the interconnection and exchange of information with other platforms or devices with appropriate secure mechanisms (e.g. REST API)	Compatibility	PACE_001 PACE_011 PACE_017 PACE_023 AIA_002	PLATF_F_01 PLATF_F_02	NF
Req_093	ICARUS should be able to support the functional and flexible operation in a distributed cloud infrastructure	Compatibility			NF
Req_094	ICARUS should be able to consume and handle different datasets in various formats (e.g. CSV, JSON, XML files)	Compatibility	PACE_011 PACE_014 PACE_021 CEL_008	PLATF_F_02 PLATF_F_14 PLATF_F_41 PLATF_F_42	NF
Req_095	ICARUS should provide an easy-to-use and user-friendly interface in which the analytics and visualisation processes are supported by guides and manuals	Usability	AIA_006 AIA_015 AIA_016 AIA_017 CEL_009 ISI_007	PLATF_F_07 PLATF_F_56	NF
Req_096	ICARUS should provide a user interface that supports straightforward task accomplishment	Usability	PACE_002	PLATF_F_29 PLATF_F_30	NF
Req_097	ICARUS should provide easy navigation through the platform features with support of dashboards or wizard/guide	Usability	PACE_002 AIA_010 AIA_013 ISI_007	PLATF_F_31 PLATF_F_32	NF
Req_098	ICARUS should provide the suitable error protection methods for all input fields	Usability	PACE_018 ISI_003		NF
Req_099	ICARUS should enable the secure storage of assets (datasets, reports, etc.)	Reliability	PACE_020	PLATF_F_46 PLATF_F_47	NF
Req_100	ICARUS should be able to handle simultaneous requests on a timely and efficient manner	Reliability	PACE_004 CEL_004		NF
Req_101	ICARUS should provide the mechanisms to recover after system failure conditions	Reliability			NF
Req_102	ICARUS should be able to handle software errors without affecting the platform overall functionality	Reliability			NF

Req_103	ICARUS should ensure different authorisation access to different datasets	Security	PACE_003 PACE_006 PACE_017 AIA_002 ISI_001	PLATF_F_04 PLATF_F_36 PLATF_F_47	NF
Req_104	ICARUS should provide the appropriate logging mechanisms for all operations	Security	PACE_017 AIA_014	PLATF_F_26	NF
Req_105	ICARUS should be able to verify the identity of the user/subject performing any operation	Security	PACE_017 AIA_014		NF
Req_106	ICARUS should be able to trace all user/subject operations	Security	PACE_017 AIA_014		NF
Req_107	ICARUS should be composed by components that are operating independently	Maintainability			NF
Req_108	ICARUS should provide the tools that support enhanced system monitoring and debugging	Maintainability			NF
Req_109	ICARUS should provide a sophisticated alarm mechanism to identify failures or deficiencies	Maintainability	PACE_018 ISI_003		NF
Req_110	ICARUS should provide the proper mechanisms for system upgrade with minimum downtime	Maintainability	PACE_004		NF
Req_111	ICARUS should offer easy installation process in a timely manner	Portability			NF
Req_112	ICARUS should support deployment on various Linux distributions	Portability			NF
Req_113	ICARUS should be composed by independent components that are replaceable with minimum impact and effort	Portability			NF

Annex IV: ICARUS Technical Requirements Backlog

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
Data Collection			
Import & Export	TR_001	The ICARUS platform shall allow data to be imported from external sources.	Req_002, Req_028, Req_032, Req_055, Req_056, Req_067
	TR_002	The ICARUS platform shall allow the user to upload and download files.	Req_022, Req_026, Req_040, Req_041
	TR_003	The ICARUS platform should offer a simplified data check-in process for data that the providers intend to keep for personal usage only.	Req_042
	TR_004	The ICARUS platform should allow the user to save datasets that are currently in a private analytics space on the central platform storage.	Req_045
	TR_005	The ICARUS platform shall offer a well-defined API for data export.	Req_086
	TR_006	The ICARUS platform shall support updating and maintaining uploaded datasets.	Req_087
	TR_007	The ICARUS platform should allow the user to choose in which format to download data, when a transformation service is available.	Req_024, Req_094
	TR_008	The ICARUS platform should provide a service that transforms data from a format to another for selected predefined data formats.	Req_023, Req_094
	TR_009	The import and export mechanisms of the ICARUS platform should support large files.	Req_091
	TR_010	The ICARUS platform should be able to consume data from external RESTful APIs.	Req_056, Req_092
	TR_011	The ICARUS platform should support end-to-end data encryption.	From external stakeholders during MVP validation interviews
	TR_012	The ICARUS platform should support all data types described in the data requirements reported in D1.1	D1.1 (also relevant to Req_013, Req_014, Req_015, Req_021, Req_025, Req_048, Req_065, Req_068,

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
			Req_069, Req_070, Req_071, Req_073, Req_074, Req_079, Req_081)
	TR_013	The ICARUS platform should allow users to choose which field types in their datasets will be encrypted.	Stemming from TR_011
Data Cleansing	TR_014	The ICARUS platform should provide data cleansing functionalities.	Req_037, Req_083, Req_084
Data Anonymisation	TR_015	The ICARUS platform should provide a data anonymisation tool / service.	Req_039, Req_075, Req_084
Data Enrichment			
Data Representation, Semantics & Metadata	TR_016	The ICARUS platform shall comply with a common underlying metadata schema	Requirement coming from and clarified in D2.1
	TR_017	The ICARUS platform shall comply with a common underlying data model	Req_002, Req_028, Req_029, Req_032, Req_055
	TR_018	The ICARUS platform shall ensure that external data being imported in ICARUS are mapped to the ICARUS data model (in a semi-automatic manner).	Req_002, Req_028, Req_029, Req_032, Req_055
	TR_019	The ICARUS platform should provide the ability to data providers to assign predefined and/or custom tags (keywords) to their datasets.	Req_005, Req_010
	TR_020	The ICARUS platform shall offer a service that enriches uploaded data based on information from certain predefined controlled vocabularies (e.g. airport codes).	Req_030, Req_052
	TR_021	The ICARUS platform shall enable the users to assign IPR related attributes to the datasets.	Req_038
	TR_022	The ICARUS platform should provide predefined data license templates	Req_034
	TR_023	The ICARUS platform should allow data providers to customise the provided data license templates.	Req_034, Req_060
	TR_024	The ICARUS platform shall allow the user to define and configure a custom data license.	Req_034, Req_060
	TR_025	The ICARUS platform should store and show in an intuitive manner provenance-related information, e.g. when a dataset was last modified.	Req_072
	TR_026	The ICARUS platform shall offer an interactive UI to let the user browse the ICARUS data model.	Req_031, Req_046, Req_047

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
	TR_027	The ICARUS platform should support a model lifecycle management service that enables the user to recommend extensions to the data model.	Req_031, Req_046, Req_047
	TR_028	The ICARUS platform should support a process / service to enable the ICARUS administrator to review the data model recommendations and approve or decline them.	Req_031, Req_046, Req_047
Asset Exploration and Extraction			
Search	TR_029	The ICARUS platform shall support search functionality over the datasets to allow the user to find datasets by type, keyword, date, time.	Req_006, Req_009, Req_010, Req_011, Req_012
	TR_030	The ICARUS platform should save the query history of the user and allow the user to review it.	Req_020
	TR_031	The ICARUS platform shall retrieve and show the datasets that are relevant to a dataset that is returned as a query result.	Req_024, Req_047
	TR_032	The ICARUS platform should provide a mechanism for identifying connections among datasets based on their mapping to the common underlying data schema/model.	Req_043
	TR_033	The ICARUS platform should allow for spatiotemporal information to be un-encrypted in the datasets so that search queries can be performed on it.	Req_012
Data Sharing	TR_034	The ICARUS platform shall provide an information catalogue about all datasets that are open or available for sharing (by their respective data providers).	Req_057
	TR_035	The ICARUS platform shall enable the creation of data sharing contracts with detailed terms in an immutable manner.	Req_059
	TR_036	The ICARUS shall provide walkthroughs and guidelines regarding the creation of data sharing contracts.	Req_059
	TR_037	The ICARUS platform shall allow users to set pricing terms for their datasets.	Req_076
	TR_038	The ICARUS platform should support various payment methods.	Req_077
	TR_039	The ICARUS platform should provide a mechanism for data providers and data consumers to negotiate prior to signing the data sharing contract.	Req_036
	TR_040	The ICARUS platform may allow existing, active data contracts (traditionally signed by a data provider) to be facilitated / executed by the platform.	From external stakeholders during MVP validation interviews
	TR_041	The ICARUS platform shall allow users to request to purchase and to access datasets not owned by them	Requirement coming from and clarified in D2.2
	TR_042	The ICARUS platform shall allow users to accept or deny requests for access on their datasets made by other users	Requirement coming from and clarified in D2.2

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
	TR_043	The ICARUS platform shall store the data sharing contracts in a DLT-based repository for non-repudiation purposes.	From external stakeholders during MVP validation interviews
Data Analysis & Visualisation			
Analysis & Visualisation	TR_044	The ICARUS platform should provide a UI that allows the user to define, configure, review and manage data analysis jobs and save configurations for later re-usage.	Req_003, Req_055, Req_078
	TR_045	The ICARUS platform shall enable the integration and combined analysis over multiple datasets.	Req_007
	TR_046	The ICARUS platform should allow the easy configuration and application of advanced data analysis algorithms.	Req_018, Req_019, Req_088
	TR_047	The ICARUS platform shall enable the application of predefined data analysis algorithms on datasets.	Req_019
	TR_048	The ICARUS platform should support the combination (merging) of datasets based on common fields into one dataset.	Req_027, Req_026, Req_085
	TR_049	The ICARUS platform should provide a monitoring UI for the progress and status of data analysis jobs.	Req_062
	TR_050	The ICARUS platform should provide tools/services to define and execute what-if scenarios on the datasets.	Req_063
	TR_051	The ICARUS platform shall provide tools and services to apply machine learning algorithms	Requirement coming from and clarified in D2.2
	TR_052	The ICARUS platform should provide tools and services to apply deep learning algorithms	Requirement coming from and clarified in D2.2
	TR_053	The ICARUS platform shall provide tools and services to apply basic analytics	Requirement coming from and clarified in D2.2
	TR_054	The ICARUS platform should provide tools and services that enable users to perform statistical analysis over datasets	Req_066
	TR_055	The ICARUS platform should offer data management methods and algorithms that handle both structured and unstructured data.	Req_050
	TR_056	The ICARUS platform shall offer data visualisation tools/ functionalities.	Req_051, Req_053
	TR_057	The ICARUS platform shall enable the users to define and schedule data analysis jobs.	Req_054
	TR_058	The ICARUS platform should enable the users to define, configure and schedule data management and processing recipes	From MVP
	TR_059	The ICARUS platform should allow a user to easily perform aggregations on a dataset.	Req_080
	TR_060	The ICARUS platform shall support analytics jobs in a scalable and reliable manner	Req_089

	ID	Description of the requirement	Relevant Functional & Non-Functional Requirements
	TR_061	The ICARUS platform should provide tools and services to perform resource allocation for data analysis purposes.	Req_090
Added value services and platform features			
Notifications	TR_062	The ICARUS platform shall allow users to manage their notification preferences.	Req_001, Req_033, Req_049
	TR_063	The ICARUS platform should inform users with active contracts on a dataset that the dataset has been updated.	Req_001, Req_033
	TR_064	The ICARUS platform should provide notifications to inform users when their scheduled analytics jobs finish.	Req_049, Req_061
Usage Analytics	TR_065	The ICARUS platform should provide data usage analytics to the users for the datasets they own.	Req_035, Req_064
Security and Privacy	TR_066	The ICARUS platform shall provide public, private and confidential working spaces.	Req_004
	TR_067	The ICARUS platform shall ensure that access control over datasets is applied according to the data provider's policies and the terms of relevant active valid data sharing contracts.	Req_016, Req_082
	TR_068	The ICARUS platform shall forbid unauthorised user access to the platform and the datasets.	Req_044
	TR_069	The ICARUS platform storage shall be secure.	Req_099
	TR_070	The ICARUS platform should ensure different authorisation levels for accessing datasets.	Req_017, Req_088, Req_103
	TR_071	The ICARUS platform should be able to verify the identity of the user/subject performing any operation in the platform.	Req_105
	TR_072	The ICARUS platform shall provide a secure and controlled registration process for new users	Requirement coming from and clarified in D2.2