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"

D3.2 – Core Data Service Bundles and Value Added Services Designs

Workpackage:	WP3 – ICARUS Platform Design		
Authors:	UBITECH, SUITE5, SILO, ENG, UCY		
Status:	Final	**Classification:**	Public
Date:	31/05/2019	**Version:**	1.00

Disclaimer:
The ICARUS project is co-funded by the Horizon 2020 Programme of the European Union. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Communities. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

ICARUS Project Profile

Grant Agreement No.:	780792
Acronym:	ICARUS
Title:	Aviation-driven Data Value Chain for Diversified Global and Local Operations
URL:	http://www.icarus2020.aero
Start Date:	01/01/2018
Duration:	36 months

Partners

UBITECH ubiquitous solutions	UBITECH (UBITECH)	Greece
ENGINEERING	ENGINEERING - INGEGNERIA INFORMATICA SPA (ENG)	Italy
PACE a TXT company	PACE Aerospace Engineering and Information Technology GmbH (PACE)	Germany
Suite5 We Deliver Intelligence	SUITE5 DATA INTELLIGENCE SOLUTIONS LIMITED (SUITE5)	Cyprus
University of Cyprus	UNIVERSITY OF CYPRUS (UCY)	Cyprus
CINECA	CINECA CONSORZIO INTERUNIVERSITARIO (CINECA)	Italy
OAG connecting the world of travel	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 ELEFTHERIOS VENIZELOS	ATHENS INTERNATIONAL AIRPORT S.A (AIA)	Greece
TXT E-SOLUTIONS	TXT e-solutions SpA (TXT) – 3rd party of PACE	Italy

Document History

Version	Date	Author (Partner)	Remarks
0.10	26/03/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Initial Table of Contents
0.20	28/03/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Initial outline of Sections 2-5
0.25	05/04/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Initial contribution to section 2.1
0.30	08/04/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5)	Updated contribution to sections: 2.2, 2.7, 2.8 by UBITECH 2.2, 2.5, 2.6 by Suite5
0.35	11/04/2019	Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Pavlos Lampadaris, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contribution to sections: 2.2, 2.3 by SILO 2.4 by ENG 2.9, 2.10, 2.11 by UCY
0.40	15/04/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Dimosthenis Stefanidis, George Pallis (UCY), Pavlos Lampadaris, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contribution to sections: 2.2, 2.7, 2.8 by UBITECH 2.2, 2.5, 2.6 by Suite5, 2.2, 2.3 by SILO 2.4 by ENG 2.9, 2.10, 2.11 by UCY
0.45	22/04/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Pavlos Lampadaris, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contribution to sections: 3.1, 3.12, 3.13, 3.14, 3.15 by UBITECH, 3.3, 3.9, 3.10, 3,11 by Suite5, 3.2, 3.4, 3.5 by SILO, 3.6, 3.7, 3.8 by ENG
0.50	23/04/2019	Dimosthenis Stefanidis, George Pallis (UCY)	Updated contribution to section 4
0.60	03/05/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Pavlos Lampadaris, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contribution to sections: 3.1, 3.12, 3.13, 3.14, 3.15 by UBITECH, 3.3, 3.9, 3.10, 3,11 by Suite5, 3.2, 3.4, 3.5 by SILO, 3.6, 3.7, 3.8 by ENG
0.65	10/05/2019	Dimosthenis Stefanidis, George Pallis (UCY)	Updated contribution to section 4
0.70	14/05/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH), Fenareti Lampathaki, Evmorfia Biliri (Suite5), Susanna Bonura, Domenico Messina (ENG), Pavlos Lampadaris, Tasos Violetis, Apostolos Tsatsoulas, Samuel Marntirosian (SILO)	Updated contribution to sections: 3.1, 3.12, 3.13, 3.14, 3.15 by UBITECH, 3.3, 3.9, 3.10, 3,11 by Suite5, 3.2, 3.4, 3.5 by SILO, 3.6, 3.7, 3.8 by ENG
0.75	15/05/2019	Dimosthenis Stefanidis, George Pallis (UCY)	Updated contribution to section 4
0.80	20/05/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Updated full draft circulated for internal review

Version	Date	Author (Partner)	Remarks
0.80_UCY	25/05/2019	George Pallis (UCY)	Internal review
0.80_S5	25/05/2019	Fenareti Lampathaki (Suite5)	Internal review
0.90	28/05/2019	Dimitrios Miltiadou, Konstantinos Perakis (UBITECH)	Updated version addressing comments received during the internal review process
1.0	30/05/2019	Dimitrios Miltiadou, Dimitrios Alexandrou (UBITECH)	Final version for submission to the EC

Executive Summary

The document at hand, entitled "Core Data Service Bundles and Value Added Services Designs" constitutes a report of the preliminary efforts and the produced results of Tasks T3.4 "Core Data Service Bundles In-depth Design" and T3.5 "ICARUS Added Value Services Design". The purpose of this deliverable is to deliver the initial design specifications of the Core Data Service and the Added Value Service Bundles. Within this context, the scope of the current report can be described in the following axes:

- To present the design process that is adopted in order to produce the design specifications of the ICARUS platform's services. The process includes multiple iterations that are performed from the development team in collaboration with the ICARUS stakeholders in order to analyse the knowledge extracted from the outcomes of WP1, WP2 and the preliminary results of the work performed in WP3. As part of this process, the desired offerings and functionalities of the platform are defined and the initial solutions that will enable the implementation these offerings and functionalities were drafted. These draft solutions are later transformed into the corresponding workflows from which the design specifications of the ICARUS platform's services are extracted.
- To present the designed workflows that were formulated as the direct outcome of the analysis performed in the design process and illustrate the functionalities of the integrated ICARUS platform. In these workflows, the interactions of the users with the ICARUS platform, as well as the interactions of the various components of the platform, are depicted. The workflows are presented in the form of BPMN diagrams and organised based on the areas of functionalities of the ICARUS platform. Each workflow is accompanied by a description of the workflow and the list of indicative services that involved in these workflows were documented. After multiple iterations, 29 workflows were designed in total.
- To document the design specifications of the ICARUS platform's services that were designed from the analysis of the designed workflows. The extracted services are organised into the Core Data Service Bundles and the Added Value Service Bundles. Each service bundle is composed by the main service and a set of underlying services are combined in order to offer the aspired functionalities under a specific area in the ICARUS platform. In total, 18 service bundles are extracted and described in detail focusing on the functionalities offered by each service. For each service, the technologies and tools that are to be utilised in the implementation phase are presented. More specifically, the Core Data Service Bundles contain the Data Cleansing, Data Anonymisation, Data Mapping, Metadata Handling, Data Upload, Data Analytics, Application Catalogue, Application Execution, Data Exploration, Data Licensing, Asset Brokerage, Policies Enforcement, Data

Encryption and Decryption, Resource Orchestration, Master and Worker services. Additionally, the Added Value Service Bundles contain the Data Recommendation, Notification and Usage Analytics services.

The current deliverable presents the initial design specification of the Core Data Service Bundles and the Added Value Service Bundles. It should be noted though that all tasks of WP3 remain active until M32 according to the ICARUS Description of Action. Thus, these design specifications are bound to change as the project evolves and additional technical requirements will be introduced that will result in updates and refinements in both the ICARUS platform's architecture and the platform's services. The upcoming versions of this deliverable, namely D3.3, D3.4 and D3.5, will provide the necessary documentation of the aforementioned changes.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The scope of the ICARUS deliverable D3.2 "Core Data Service Bundles and Value Added Services Designs" is to document the efforts carried out within the context tasks of WP3 T3.4 "Core Data Service Bundles In-depth Design" and T3.5 "ICARUS Added Value Services Design". Within the context of these tasks, the first version of the design specifications of the Core Data Service Bundles and the Added Value Service Bundles is delivered as a result of the first iteration of the work performed under these tasks in accordance with the ICARUS Description of Action. The deliverable D3.2 is building upon the outcomes of the deliverable D3.1 "ICARUS Architecture, APIs Specifications and Technical and User Requirements" in which the first version of the conceptual architecture of the integrated ICARUS platform was delivered, in order to provide the details of the design of the services of the ICARUS platform following a structured, iterative design process.

In this context, the scope of the current deliverable is:

- To document the design process that was adopted in order to enable the effective design of the services specifications. Within this process, the knowledge extracted from the work performed within the context of WP1 "ICARUS Data Value Chain Elaboration", WP2 "ICARUS Big Data Framework" and WP3 "ICARUS Platform Design" is combined and analysed in order to produce the list of supported workflows and ultimately the services design.
- To present the designed workflows that depict the offerings and functionalities of the ICARUS platform, the interactions of the users with the platform, as well as the interactions of the various components of the platform. The workflows are illustrated in the form of Business Process Model and Notation (BPMN) diagrams. Each diagram is accompanied by a comprehensive description. Furthermore, the list of indicative services, as extracted from the initial analysis of these diagrams, is documented followed by a short description, the list of involved components and their mapping to the relative group of services.
- To document the design specifications of the Core Data Service Bundles. More specifically, from the analysis of the designed workflows, a set of core services was extracted. Each core service incorporates a bundle of services which fulfil a specific set of responsibilities and functionalities within the ICARUS platform. Furthermore, for each core service, the list of technologies that will be exploited for the implementation of the services is presented.

- To document the design specifications of the Added Value Service Bundles. The presented added value services are complementary services that facilitate the implementation of certain functionalities in the platform and provide added value to the platform's users. Each added value service is also a bundle of relative services based on the provided functionalities. As with the core services, their design specifications are accompanied by the list of technologies and tools that will be exploited for their implementation.

The deliverable D3.2 presents the first iteration of the services design work performed under the scope of WP3. However, all the tasks of WP3 remain active until M32 according to the ICARUS Description of Action. During this period and as the project evolves, the design of the services will be constantly updated as a result of the identification and analysis of new functional and non-functional requirements, as well as their translation into technical requirements and the feedback provided by the implementation activities. Hence, the upcoming versions of this deliverable will incorporate all the introduced refinements and adjustments.

1.2 Document Approach

The current deliverable follows a systematic and comprehensive approach in order to present the outcomes and the knowledge extracted from the work performed in Tasks T3.4 and T3.5 of WP3.

At first, the design process that was adopted is presented. The process defines the steps that were followed in order to produce the design specifications of the services of the ICARUS platform. The process took as input all the outcomes from the work performed within the context of WP1, WP2 and WP3 in order to conduct a detailed analysis of the requirements that should be addressed in order to fulfil the needs of the ICARUS stakeholders. Following a collaborative and iterative process, where the ICARUS stakeholders and the development team of ICARUS were involved, the first draft solutions were extracted and were translated into the ICARUS platform workflows.

Following the design process presentation, the designed workflows in the form of BPMN diagrams are presented. In these workflows the interactions of the user with the ICARUS platform are depicted, as well as the interactions between the various ICARUS components. Each workflow is accompanied by a comprehensive analysis of these interactions in which the ICARUS platform's functionalities are also highlighted. The workflows were organised and presented in categories based on the core functionality that they offer. Furthermore, from the workflows of each category, a set of indicative services, as extracted from the workflows, are presented.

Following the designed workflows presentation, the design specifications of the Core Data Service Bundles are presented. For each core service, its detailed design specifications are presented. As each core service is designed as a bundle of services that are bounded to the main service, a description of each service included in the bundle is provided as well as the set of functionalities that they are offering for the implementation of the ICARUS platform. Following the description of each core service, the list of technologies and tools that will be utilised is presented.

Following the Core Data Service Bundles, the set of Added Value Service Bundles are presented. These added values service are supplementary services that provide functionalities from which the users of the ICARUS platform will obtain added value. Following the same approach as with the Core Data Service Bundles, the detailed design specifications of each added value service are presented along with the description of the services included in this bundle of services. Moreover, the list of technologies and tools that will be used is also presented.

1.3 Relation to other ICARUS Results

The ICARUS Deliverable D3.2 is released in the scope of WP3 "ICARUS Platform Design" activities and reports the efforts undertaken within the context of Tasks 3.4 "Core Data Service Bundles In-depth Design" and T3.5 "ICARUS Added Value Services Design". As depicted in Figure 1-1, the outcomes of T3.1 and T3.2 provided the input to T3.3 in order to formulate the initial version of the integrated ICARUS platform and the outcomes of T3.3 are provided as input to both T3.4 and T3.5. As Tasks T3.1 and T3.2 will be constantly updated as the project evolves in order to follow the project's advancements, the updated outcomes will be also provided as input to T3.3, T3.4 and T3.5 in order to formulate the updated ICARUS platform architecture, as well as the updated designs of the Core Data Service Bundles and the Added Value Services that will be documented in the upcoming versions of the deliverable.

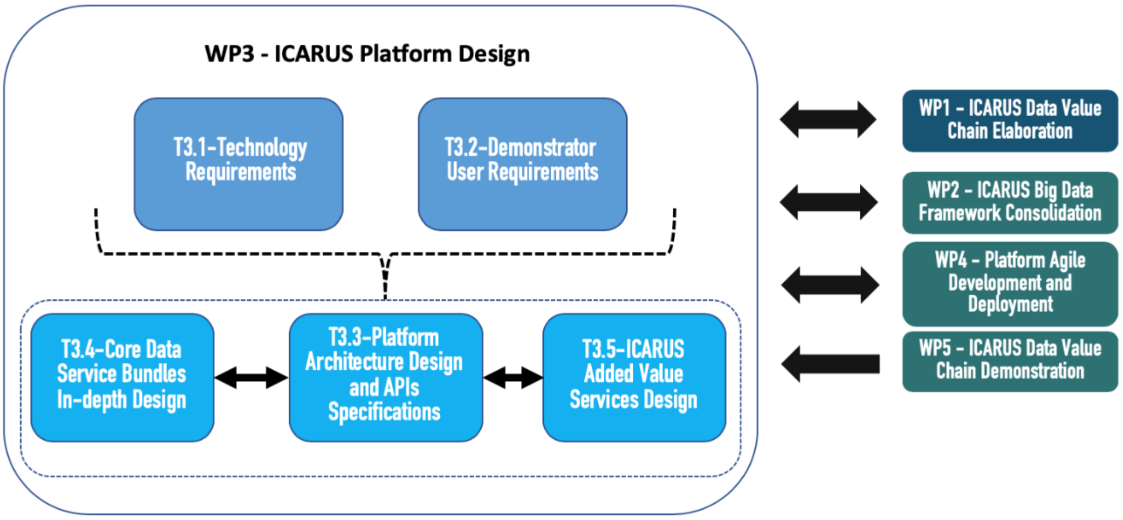
WP3 - ICARUS Platform Design
T3.1-Technology Requirements
T3.2-Demonstrator User Requirements
T3.4-Core Data Service Bundles In-depth Design
T3.3-Platform Architecture Design and APIs Specifications
T3.5-ICARUS Added Value Services Design
WP1 - ICARUS Data Value Chain Elaboration
WP2 - ICARUS Big Data Framework Consolidation
WP4 - Platform Agile Development and Deployment
WP5 - ICARUS Data Value Chain Demonstration

Figure 1-1: Relation to other ICARUS Work Packages

Furthermore, D3.2 and WP3 are directly related to the outcomes of WP1 “ICARUS Data Value Chain Elaboration” with regard to the ICARUS methodology, the ICARUS Minimum Viable Product (MVP), and WP2 “ICARUS Big Data Framework Consolidation” with regard to the data collection, data provenance, data safeguarding, data curation, data linking, data analytics and data sharing methods that are applicable to ICARUS.

The outcomes from the external ICARUS MVP validation activities that are performed within the context of WP1 are expected to result in new additional requirements that will be addressed within the context of WP3. Additionally, the advancements that will be introduced in the Big Data Framework methods under the scope of the activities of WP2 will drive the refinements and updates in the upcoming version of the integrated ICARUS platform, as well as in the designs of the respective services of the platform.

D3.2 provides the necessary design and specifications of the services of the ICARUS platform to WP4 “Platform Agile Development and Deployment” that will deliver the implementation of these services 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 (ICARUS Data Value Chain Demonstration) activities will be fed in WP3 and will drive the updates and adjustments in both the design and specifications of the services 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 design process that was adopted for the services definition and design is elaborated and the designed workflows are presented. At first, the design process is presented, providing an overview of the methodology followed. Following the design process presentation, the designed workflows are presented organised in categories based on the areas of the functionalities of the platform. For each area, a set of workflows is illustrated in the form of BPMN diagrams followed by a comprehensive description of the depicted workflow. Finally, a set of indicative services are presented followed by a short description, the list of involved components and the mapping to the relative service group.

- In Section 3, the design specifications of the Core Data Service Bundles are presented. For each core service bundle, the corresponding services included in this bundle are described focusing on the provided functionalities, their corresponding responsibilities and their scope within the ICARUS platform. Furthermore, for each service bundle, the list of technologies that shall be exploited for the implementation of the services is presented.
- In Section 4, the design specifications of the Added Value Service Bundle are presented. Following the same approach as in section 3, the corresponding services included in this bundle are documented highlighting their functionalities, responsibilities and scope within the ICARUS platform. Additionally, their design specifications are accompanied by the list of technologies and tools that are to be exploited for their implementation.
- Section 5 concludes the deliverable, outlining the main findings of the deliverable which will guide the development efforts of the consortium.

2 Design Process

2.1 Scope of the design process

For the design of the ICARUS platform's services, the outcomes of the work performed in WP1 and WP2, as well as the knowledge extracted from the work performed within the context of WP3 that was presented in deliverable D3.1 were taken as input. In detail, the following outcomes were fed in the design process:

- The different phases of the ICARUS methodology as defined in deliverable D1.2.
- The Big Data Framework methods for data collection, data provenance, data safeguarding, data curation, data linking, data analytics and data sharing that were presented in deliverables D2.1 and D2.2.
- The ICARUS technical requirements that were documented in deliverable D3.1.
- The ICARUS platform conceptual architecture and the components incorporated in this architecture that were presented also in deliverable D3.1.

In this design process, multiple iterations were performed in order to produce the efficient and effective services' design that will facilitate the implementation of the aspired ICARUS platform offerings and functionalities. During these iterations, the input described above was further analysed and combined towards the better understanding of the requirements that need to be fulfilled and the functionalities that need to offered by each service in the ICARUS platform. The first step of the design process includes "the problem definition" in which the desired offerings and functionalities are formulated by leveraging the knowledge extracted from the input described above. Through a collaborative and iterative process that brought together the development team of the ICARUS platform and the ICARUS stakeholders, the draft solutions that will address the requirements were designed. In the next step, the design of the required services was initiated focusing on the parts of the services with which the users interact with the ICARUS platform and how these interactions are handled internally by the ICARUS platform. The outcome of this step is the design of a set of workflows in a descriptive and organised manner. For the design of the workflows, the consortium decided to utilise the Business Process Model and Notation (BPMN) standard.

BPMN is considered as one of the leading industry standard notation for modelling business processes that "brings forth expertise and experience with many existing notations and seeks to consolidate the best ideas from these divergent notations into a single notation" (OMG, 2009). As stated in the BPMN 2.0 release "BPMN provides a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes" (OMG, 2009).

In the following subsections, the designed workflows that are the outcomes of the described design process are presented in categories based on the areas of the functionalities of the ICARUS platform, as depicted in Figure 2-1. For each workflow, a comprehensive description is provided along with the respective BPMN diagram. Following the description of the workflow, a list of indicative services are presented at the end of each subsection as extracted from the initial analysis of the presented workflows. For each service, a short description is provided along with the list of involved components and the relative group of services that each service belongs to.

The knowledge extracted from this section is providing the basis for the design of the ICARUS platform's services, as compiled by the development team of the ICARUS platform, that are presented in section 3 of this deliverable.

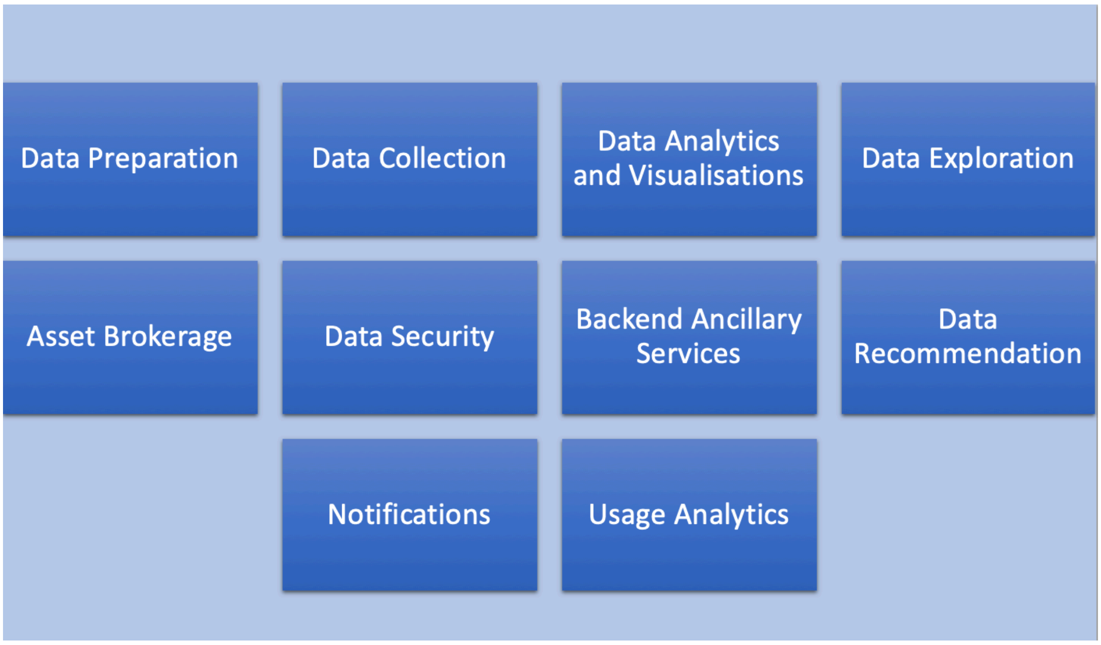

Figure 2-1: ICARUS platform's areas of functionalities

2.2 Data Preparation

2.2.1 Designed workflows

The data preparation workflow is incorporating all the actions that are enabling the data cleaning, data mapping, data anonymization and data encryption processes of the ICARUS platform. The scope of these processes is to prepare the data provider's private and confidential datasets prior to being uploaded in the ICARUS platform. To this end, the data provider is provided with a user-friendly and easy-to-use user interface in order to design and define the instructions or parameters of each phase of the preparation according to his/her needs.

Figure 2-2 illustrates the interactions between the data provider and the ICARUS platform user interface for the dataset preparation and how these interactions are processed and translated by the ICARUS platform into internal processes and functionalities that can be executed at local level in the stakeholder's premises for increased end-to-end data security. In this workflow, the user is compiling the data preparation process by defining the exact actions from the list of potential actions, that will be performed on each phase. Once all the actions are defined, the data preparation process is submitted for execution. The data preparation is composed by both mandatory and optional phases as illustrated in Figure 2-2.

In this workflow, the user initiates a new data preparation process and provides a sample of the dataset that will be eventually processed. This sample is extracted, parsed and stored by the ICARUS Storage. The results of the extracted and parsed sample are displayed in a tabular-like format so that they can be used in the forthcoming phases of the data preparation.

In the subsequent phase, the user is able to set the cleansing rules that will be applied on the dataset and span from simple to more advanced cleansing techniques and operations over the dataset. Once all cleansing rules are set and their consistency is checked, they are stored in the data preparation process. As displayed also in the workflow, this phase is optional and the user may decide not to perform any cleansing operations over the dataset.

In the next phase, the user is defining the mapping of the fields of the dataset to the ICARUS common aviation data model. The mapping rules that are created are stored in the data preparation process for later usage. This phase is mandatory in order to enable the effective data integration and data exploration within the ICARUS platform.

Following the data mapping phase, the user is selecting the anonymisation techniques that will be applied on the fields of the dataset. As with the previous phases, the anonymisation rules are stored in the data preparation process for the specific dataset. The anonymisation phase is optional, as depicted also in the workflow, and the user may decide to skip this phase.

The final phase includes the definition of the columns of the dataset that will be encrypted. The encryption rules are also stored in the data preparation process once they are defined and this phase is optional. Once all phases are completed, the designed data preparation process can be submitted for execution.

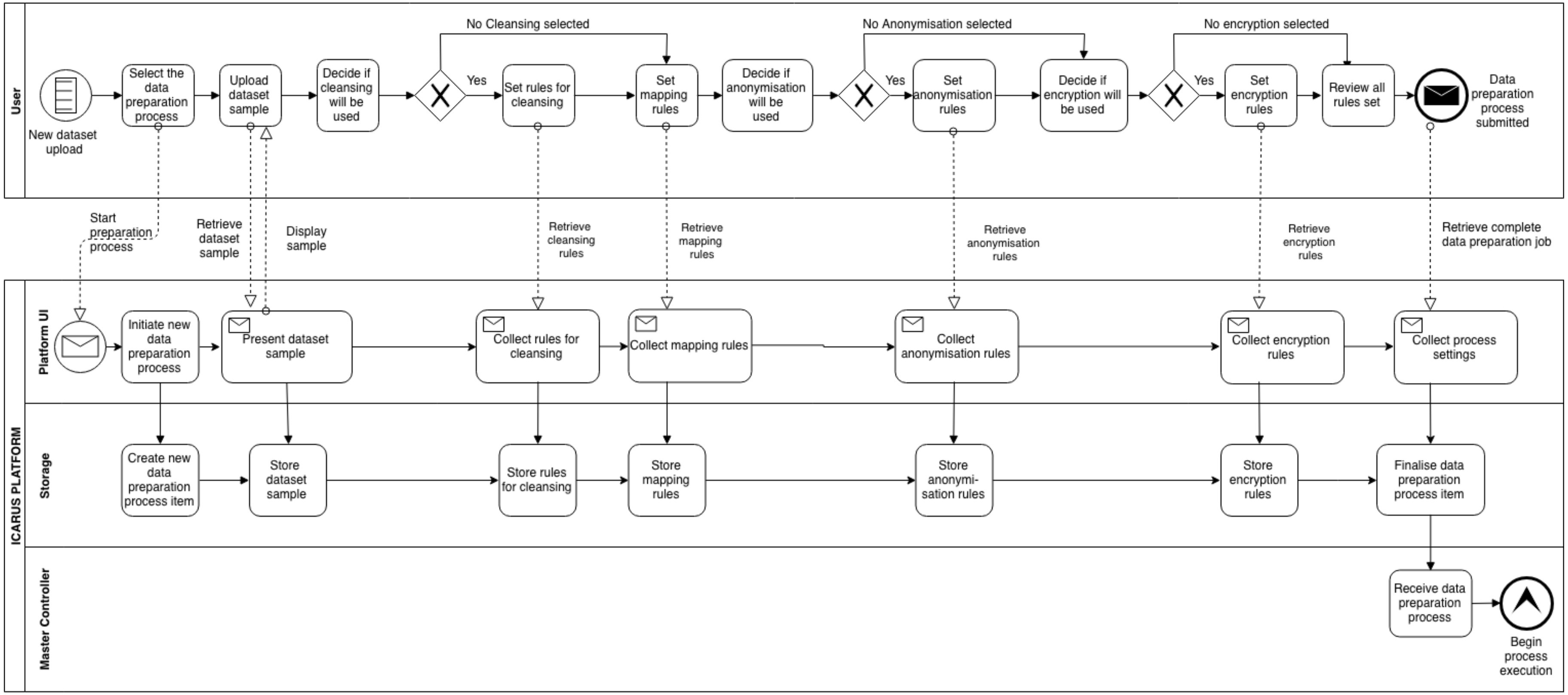
User
New dataset upload
Select the data preparation process
Upload dataset sample
Decide if cleansing will be used
No Cleansing selected
Yes
Set rules for cleansing
Set mapping rules
Decide if anonymisation will be used
No Anonymisation selected
Yes
Set anonymisation rules
Decide if encryption will be used
No encryption selected
Yes
Set encryption rules
Review all rules set
Data preparation process submitted
Start preparation process
Retrieve dataset sample
Display sample
Retrieve cleansing rules
Retrieve mapping rules
Retrieve anonymisation rules
Retrieve encryption rules
Retrieve complete data preparation job
ICARUS PLATFORM
Platform UI
Initiate new data preparation process
Present dataset sample
Collect rules for cleansing
Collect mapping rules
Collect anonymisation rules
Collect encryption rules
Collect process settings
Storage
Create new data preparation process item
Store dataset sample
Store rules for cleansing
Store mapping rules
Store anonymi-sation rules
Store encryption rules
Finalise data preparation process item
Master Controller
Receive data preparation process
Begin process execution

Figure 2-2: Data Preparation – data preparation process design workflow

Once the data preparation process is submitted, the execution is undertaken by the Master Controller component. As presented in the ICARUS platform architecture in D3.1, the Master Controller component is responsible for providing the instructions from the Core ICARUS platform to the On Premise Worker in order to be executed by the components running on the On Premise Environment. Figure 2-3 illustrates the interactions between the Core ICARUS platform and the On Premise Environment for the execution of the data preparation process.

In the displayed workflow, the Master Controller upon receiving the data preparation process request communicates with the OnPremise Worker in order to initiate the process execution in the On Premise Environment. The OnPremise Worker performs the process breakdown and initiates the execution of the corresponding sub-processes, namely the cleansing, the anonymization, the mapping and encryption sub-processes. At first, the cleansing sub-process is triggered. If this sub-process was not included in the data preparation process, as it is an optional phase, the sub-process is skipped. Following the cleansing sub-process, the OnPremise Worker initiates the mandatory mapping sub-process. Following the mapping sub-process, the OnPremise Worker is initiating the anonymisation sub-process, if this sub-process is included in the data preparation process as it is also an optional phase. Once all previous sub-processes are finished, the OnPremise Worker is initiating the encryption sub-process, if it is also included in the data preparation process, else the encryption phase is skipped. Once all sub-processes are finished, the data preparation process execution is completed.

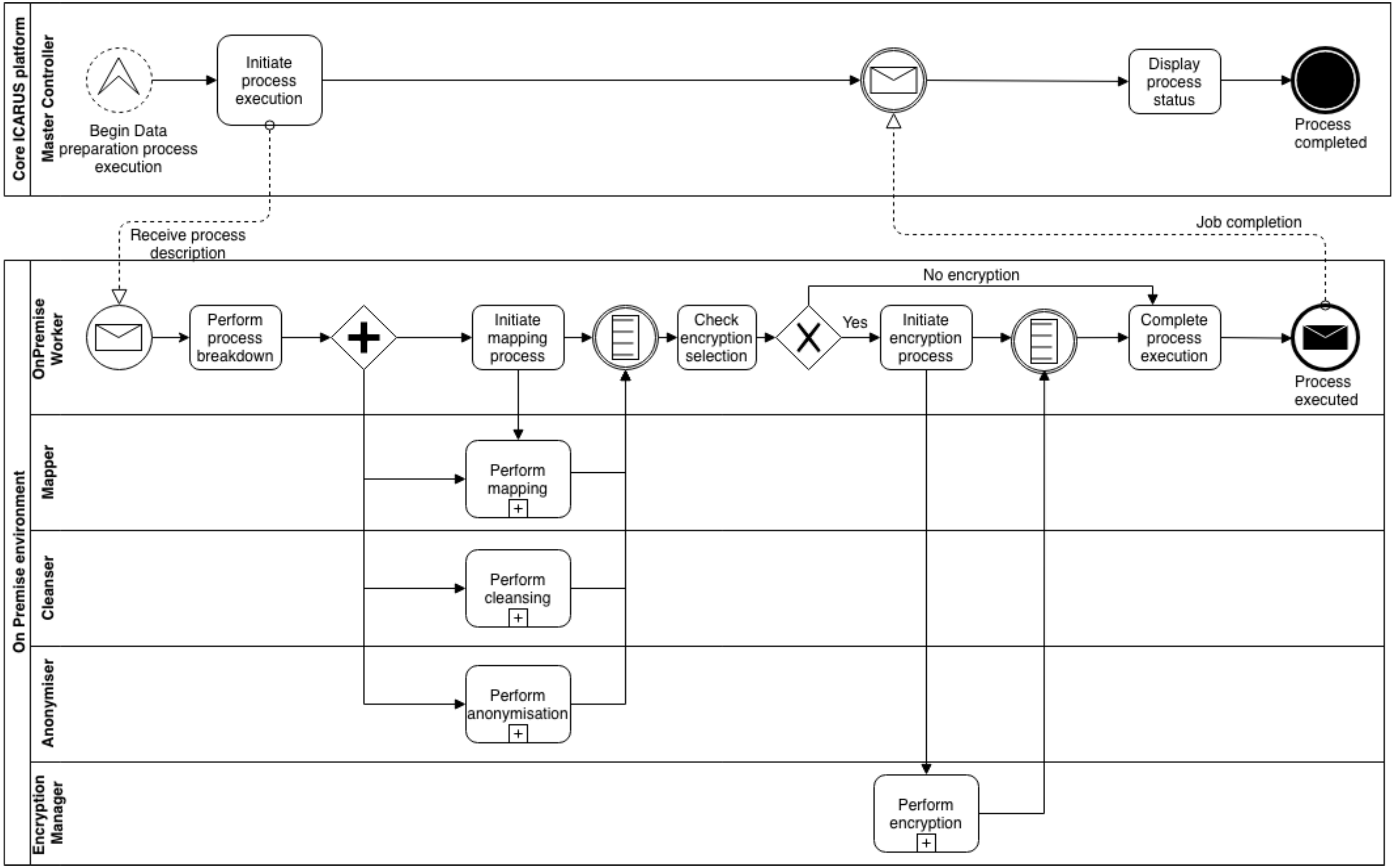

Figure 2-3: Data Preparation – data preparation process execution workflow

2.2.2 Involved services

From the analysis of the presented workflows, a list of services that are crucial for the successful execution of the designed workflows, as well as the delivery of the aspired functionalities of the ICARUS platform, were identified. The list of identified services is presented in the following table.

Table 2-1: Data Preparation services

Number	Title/ Description	Interacting Components	Related Services Group(s)
DCM_1	Dataset sample upload, parse, store and display in ICARUS	ICARUS Storage & Indexing, UI	Data Upload
DCM_2	Definition and storage of a new data preparation process	Data Handler, ICARUS Storage & Indexing	Data Upload
DCM_3	Collect and store cleansing, mapping, anonymisation and encryption rules in the data preparation process	Data Handler, ICARUS Storage & Indexing, UI	Data Upload, Data Cleansing, Data Mapping, Data Anonymisation, Data Encryption and Decryption
DCM_4	Execution and monitoring of a data preparation process	Master Controller, OnPremise Worker	Master and Worker
DCM_5	Data preparation process breakdown and allocation of the relevant sub-processes	Master Controller, Cleanser, Anonymiser, Mapper, Encryption Manager	Master and Worker, Data Cleansing, Data Mapping, Data Anonymisation, Data Encryption and Decryption
DCM_6	Execution of the cleansing process	Cleanser	Data Cleansing
DCM_7	Execution of the anonymisation process	Anonymiser	Data Anonymisation
DCM_8	Execution of the mapping process	Mapper	Data Mapping
DCM_9	Execution of the encryption process	Cleanser	Data Encryption and Decryption

2.3 Data Collection

2.3.1 Designed workflows

The term data collection encapsulates all processes and interactions that take place from the point a user decides to import a dataset to the ICARUS platform until that dataset has been successfully stored and indexed, along with all its accompanying information. As this is an inherently complex process, the current section provides multiple BPMN diagrams and their descriptions in order to properly present all underlying processes. Reference is made to the processes presented in Section 2.2, since they can be seen as part of the complete data collection workflow.

There are three ways in which datasets may become available in the core ICARUS platform:

1. Users may upload datasets they own from their local machine.
2. Users may transfer datasets created in a secure and private space.
3. Users may import data from supported open data sources, i.e. sources for which appropriate connectors facilitating data import have been implemented in ICARUS. An indicative example of an open data source could be the European Data Portal.

The generic term “users” is on purpose selected to denote that at this stage the user role does not impact the design of the processes and interactions. It may be the case that an administrator and not a data provider performs the open data import (in case 3) or that a data consumer decides to share a derivative of the acquired data and act as a data provider (as in case 2). As authentication, authorization and access policy enforcement are cross-cutting processes of all workflows, in order to avoid repetition and unnecessarily complex diagrams, the current section will consider these steps as implied and not include them in the BPMN diagrams.

Having made this assumption, the three data upload workflows share the same common high-level steps, presented in a BPMN diagram shown in Figure 2-4.

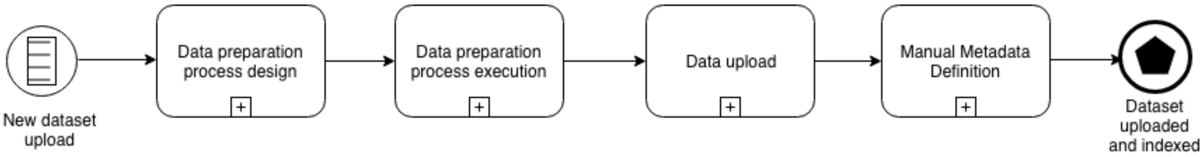
New dataset upload
Data preparation process design
Data preparation process execution
Data upload
Manual Metadata Definition
Dataset uploaded and indexed

Figure 2-4: High-level Workflow of Dataset Upload/Import into the Core ICARUS Platform

The main difference of the three workflows lies in the source of the dataset to be imported and is represented in Table 2-2 by the various data upload APIs. In the case of open data import, the user starts the processes by selecting one of the supported open data sources (e.g. an open data portal). An additional step would then be required to ensure that the user does not attempt to upload datasets from sources that are not (yet) supported in ICARUS, but this is an implementation detail that does not affect the processes outline being presented here. In the case of uploading data from the secure and private spaces, it is the SecureSpace Workers residing in them that undertake the execution of the data preparation steps, instead of the ones residing in the On Premise Environments. However, these differences do not impact the underlying processes and therefore these will be identified and outlined through the BPMN diagrams of case (1). The extracted processes are also valid for the other two cases.

The workflows “Data Preparation process design” and “Data Preparation process execution” have been already presented in Section 2.2. The scope of the “Data Upload” process is to perform the actual upload of the processed data from the On Premise Environment to the ICARUS platform. This process is initiated by the On Premise Worker when all the data preparation job instructions have been successfully executed. When the data transfer has been completed, the Data Handler will assign certain automatically created metadata on the uploaded dataset, e.g. the size, the uploading date, the source (data provider) and other provenance-related metadata. The Data Upload process is depicted in Figure 2-5.

Finally, the last part of the data collection process is the definition by the user, who here acts as the data provider, of the metadata that cannot be provided automatically by the ICARUS platform. These metadata have been described in detail in D2.1 and D2.2 and include various license-related fields, data access policies, description, tags etc. in accordance with the ICARUS metadata schema. The manual metadata definition process is necessary in order to ensure that the uploaded dataset is properly handled by the ICARUS platform. The corresponding BPMN diagram is presented in Figure 2-6.

As a final note, the provision of data updates is also part of the Data Collection Services Bundle, but its workflow is not provided in this deliverable (as it does not introduce any new services at the moment).

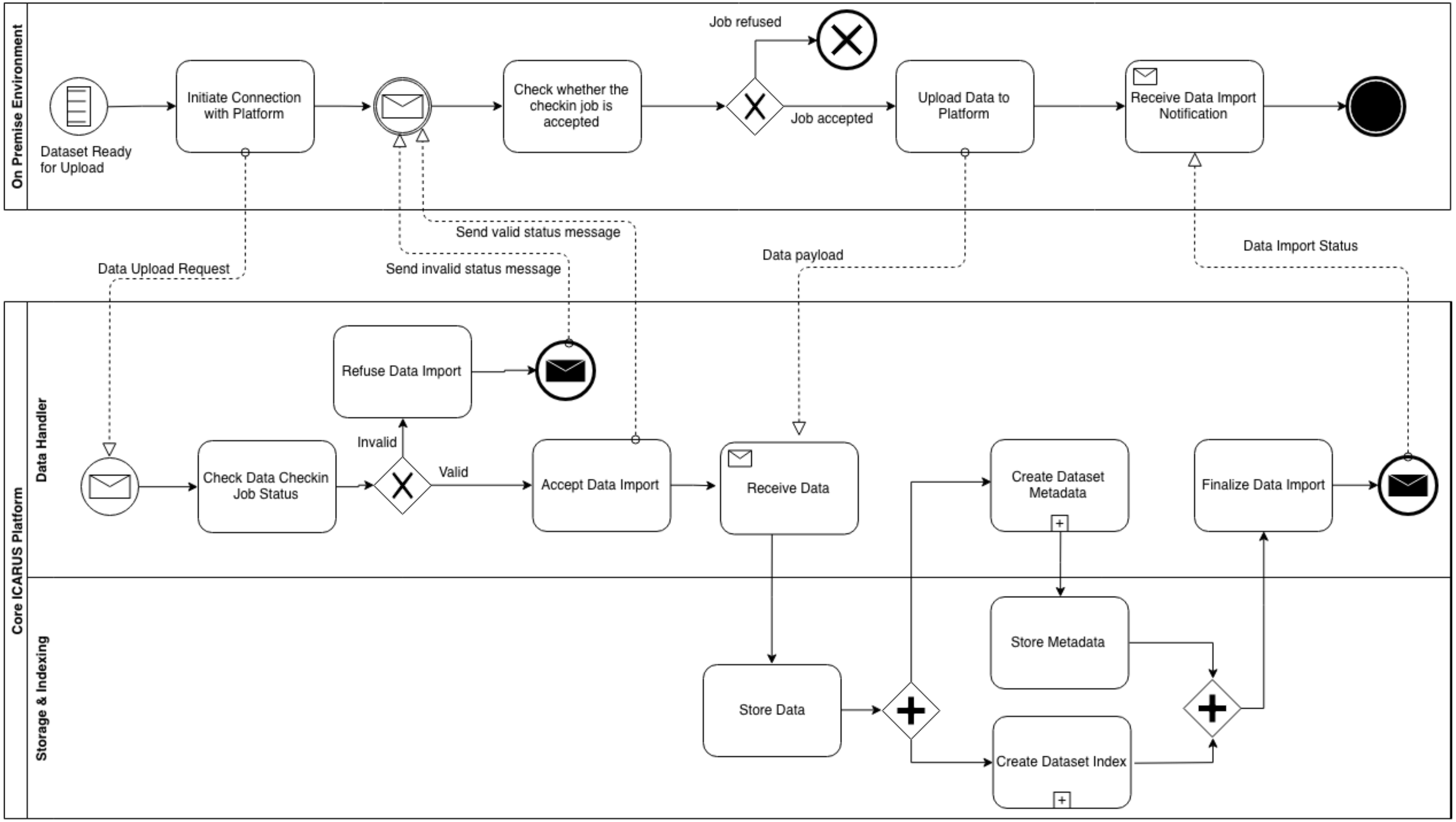

Figure 2-5: Data Upload workflow

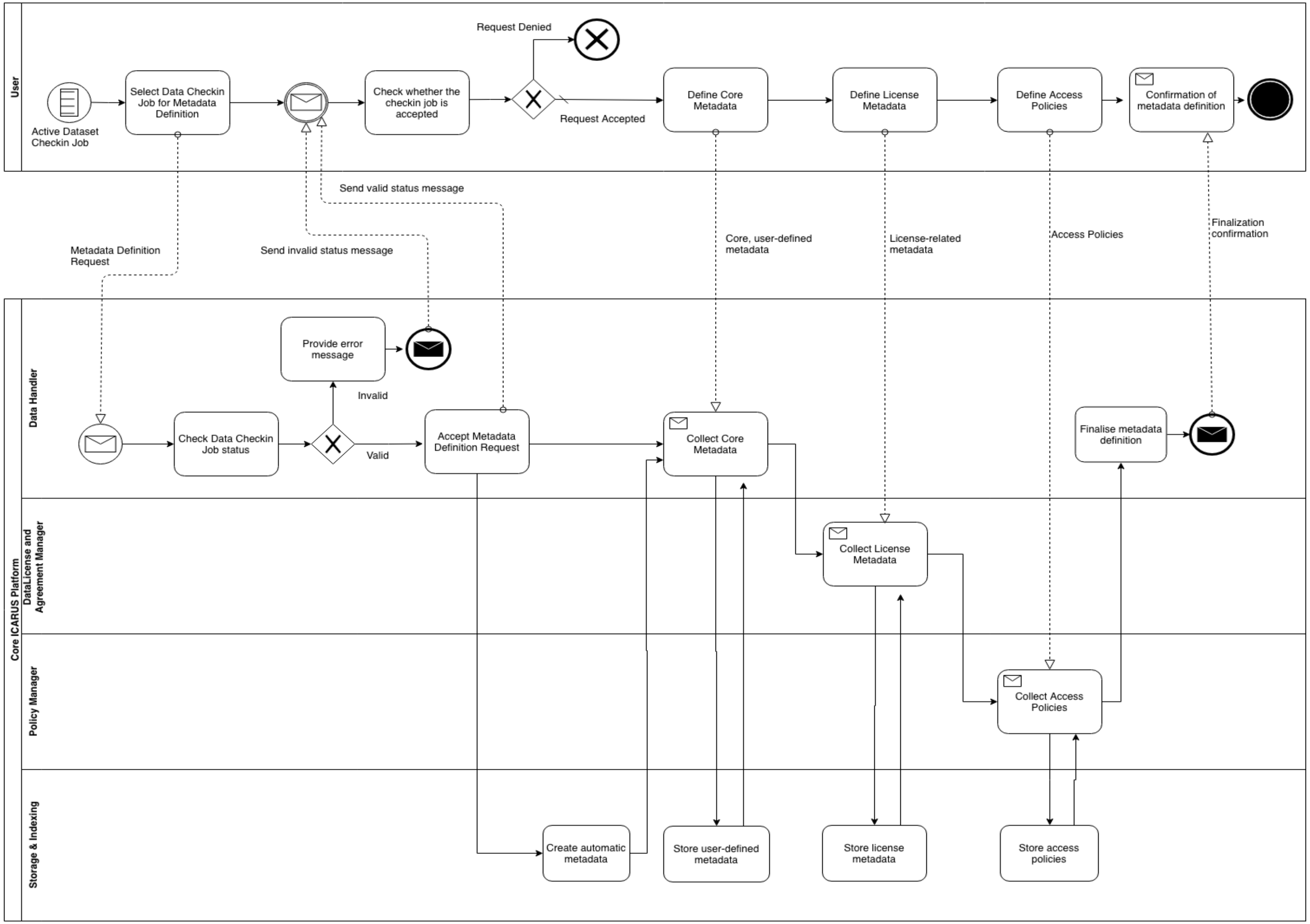
User
Active Dataset Checkin Job
Select Data Checkin Job for Metadata Definition
Check whether the checkin job is accepted
Request Denied
Request Accepted
Define Core Metadata
Define License Metadata
Define Access Policies
Confirmation of metadata definition
Send valid status message
Send invalid status message
Metadata Definition Request
Core, user-defined metadata
License-related metadata
Access Policies
Finalization confirmation
Core ICARUS Platform
Data Handler
Provide error message
Invalid
Check Data Checkin Job status
Valid
Accept Metadata Definition Request
Collect Core Metadata
Finalise metadata definition
DataLicense and Agreement Manager
Collect License Metadata
Policy Manager
Collect Access Policies
Storage & Indexing
Create automatic metadata
Store user-defined metadata
Store license metadata
Store access policies

Figure 2-6: Manual Metadata Definition

2.3.2 Involved services

Based on the workflows presented in the previous sub-section, the following services are foreseen:

Table 2-2: Data Collection services

Number	Title/ Description	Interacting Components	Related Services Group(s)
DC_1	Request to upload dataset in ICARUS	On Premise Worker, Data Handler, Secure Space Worker	Data Upload
DC_2	Dataset uploading from On Premise Environment (user's device)	On Premise Worker, Data Handler	Data Upload
DC_3	Dataset importing from external to ICARUS open data source	Data Handler through APIs for supported external data sources	Data Upload
DC_4	Dataset uploading from Worker inside a Secure and Private Space	Secure Space Worker, Data Handler	Data Upload, Master and Worker
DC_5	Automated Metadata Creation and Update	Data Handler, ICARUS Storage & Indexing	Metadata Handling
DC_6	Status check of Data Preparation Job	Data Handler (available to be invoked by various other components)	Data Upload
DC_7	Manual Definition and update of core data metadata	Data Handler, UI	Metadata Handling
DC_8	Manual Definition and update of license metadata	Data License and Agreement Manager, UI	Data Licensing
DC_9	Manual Definition and update of Data Access Policies	Policy Manager, UI	Policies Enforcement
DC_10	Data Indexing (entails the creation of representative documents to be indexed per dataset in order to be query-able)	ICARUS Storage & Indexing (available to be invoked by various other components)	ICARUS Backbone services

2.4 Data Analytics and Visualisations

2.4.1 Designed workflows

The workflows designed below describe how the ICARUS platform allows the user to perform analytics with the creation, usage and sharing of ICARUS applications. As an ICARUS application, we consider a set of defined datasets, data analysis algorithms with their corresponding parameters, and a set of selected visualisations that can be stored and shared among the users under the terms of a sharing license. The workflow is presented with particular reference to the three key components involved into the flow: the Analytics and Visualization Workbench, the BDA Application Catalogue and the Job Scheduler and Execution engine. The important key points in this workflow which will trigger several background processes under the hood are: a) the creation of a new application (with the possibility to share it with other ICARUS users), b) the immediate and/or the scheduled execution of an application, c) the visualization of the result obtained after that pipeline of algorithms involved in an application has completed its execution and d) the export of the results in order to be shared as derivative data or to be downloaded locally for internal use.

The diagrams that are presented below show the interactions between the user and the ICARUS platform with the purpose of running and/or scheduling a new ICARUS application. In general, the process is divided into three main phases: a) the design of the new ICARUS application or the loading of an existing ICARUS application, b) the execution of the ICARUS application and c) the visualisation or export of the produced results.

In the first phase, as depicted in Figure 2-7, the design of an ICARUS application is performed. Once the user initiates the Analytics and Visualisation Workbench, the ICARUS platform provides all the datasets that he/she owns or has legitimate access, the algorithms and their related configurations so that he/she can select the ones he/she needs to consider in order to perform the analysis he/she plans to carry out. At this point, the user is provided with the means for designing a new ICARUS application by designing a pipeline made of the algorithms and datasets that have to be processed. The user can save this selection as a new ICARUS application and the ICARUS platform stores the application into the BDA Application Catalogue. Alternatively, the user is able to select an ICARUS application that he/she owns or has purchased in order to load the application in the Analytics and Visualisation Workbench, however for simplicity reason this option is not depicted in the workflow as it implies that all described steps are omitted. At this point, the user can select to run the application now or run the application at a scheduled time.

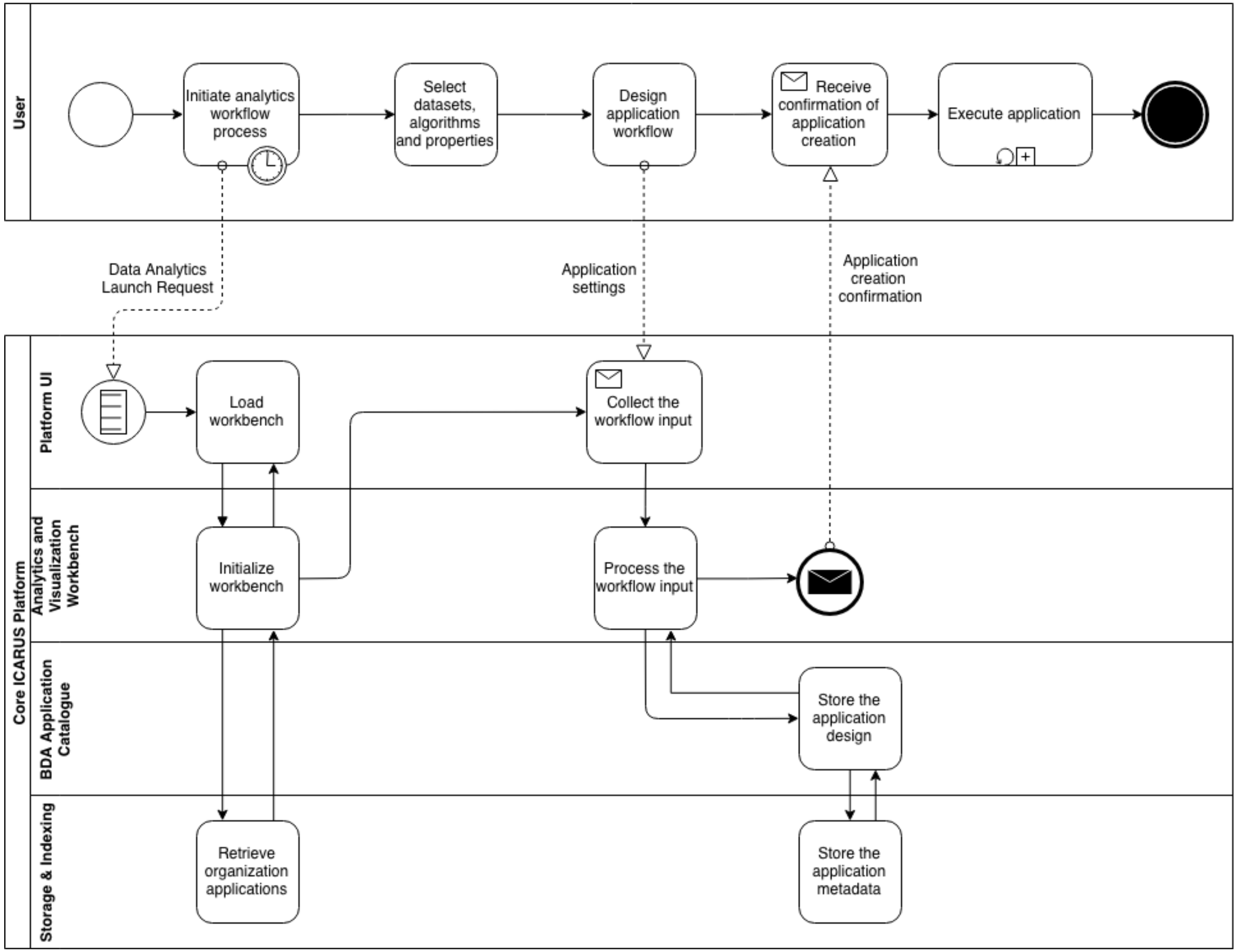
User
Initiate analytics workflow process
Select datasets, algorithms and properties
Design application workflow
Receive confirmation of application creation
Execute application
Data Analytics Launch Request
Application settings
Application creation confirmation
Core ICARUS Platform
Platform UI
Load workbench
Collect the workflow input
Analytics and Visualization Workbench
Initialize workbench
Process the workflow input
BDA Application Catalogue
Store the application design
Storage & Indexing
Retrieve organization applications
Store the application metadata

Figure 2-7: ICARUS application design workflow

In the second phase, the execution of the ICARUS application is performed. At first, the Analytics and Visualization Workbench verifies if the user's Secure and Private Space is available, otherwise it initiates a deployment request to the Resource Orchestrator. If the user selects to run the application at a scheduled time, the Analytics and Visualization Workbench communicates with the Job Scheduler which runs within the Secure and Private Space tier to mark the new job as scheduled. The scheduled job will be sent to the Execution Engine in deferred mode and the Execution Engine will start the application at the scheduled time. Alternatively, if the user selects to run the application immediately and/or the Job Scheduler activates a previously scheduled job, the Execution Engine starts two parallel processes: a) it requests and decrypts the necessary data that the algorithms will work in the user's Secure and Private Space with the help of the Decryption Manager, b) it sends a deployment request for provisioning of local Spark workers of the nodes of the Execution Cluster to the Resource Orchestrator so that the application will run within the context of a private Execution Cluster. Once both processes are complete, the execution of the ICARUS application is performed. The produced results are stored securely and in an encrypted manner into the storage of the Core ICARUS Platform and the local storage of the Secure and Private Space via the Encryption Manager and the SecureSpace Worker. The described workflow is depicted in Figure 2-8.

In the third and last phase, the produced results are visualised or exported for local usage. The following diagram describes the interaction between the user and the Analytics and Visualization Workbench whenever he/she decides to visualize or export the processed results stored in the ICARUS platform. In both cases, the relevant access policies are validated prior to the execution of the rest of the steps. In the case of export, the appropriate Data Decryption process is followed, as described in section 2.7.1 and the unencrypted results are exported to the user's environment. In the case of the visualisation, the Analytics and Visualisation Workbench receives the visualisation configuration as selected by the user that includes, the chart selection, the dataset mapping, and the characteristics and chart configuration. At this point, the Analytics and Visualisation Workbench initiates the Data Decryption process of the results, as with the export case, and produces the visualisation of the unencrypted results so that the user can explore them in a user-friendly way. Finally, the visualisation configuration can be saved within the designed ICARUS application for later usage.

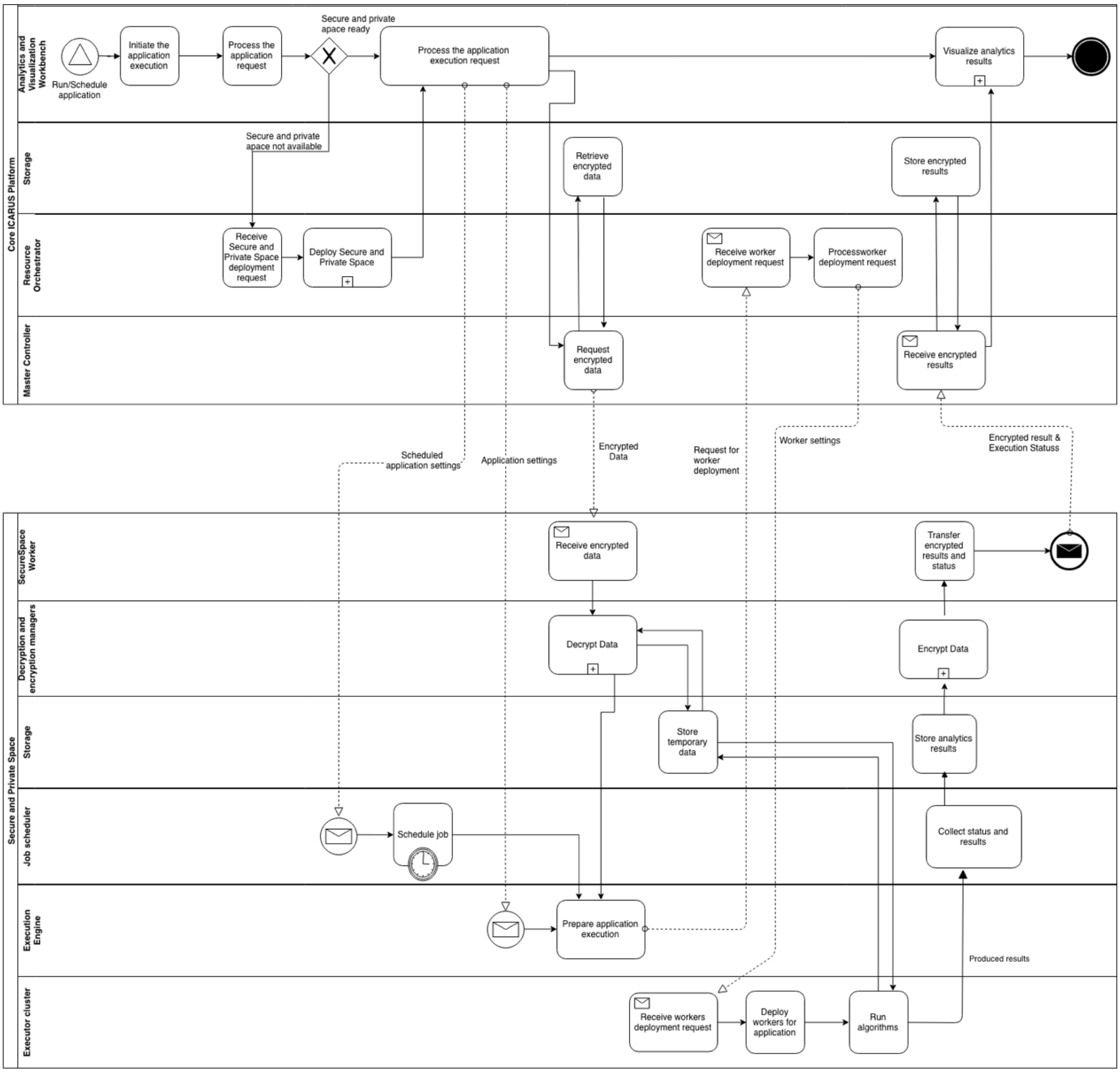
Core ICARUS Platform
Analytics and Visualization Workbench
Storage
Resource Orchestrator
Master Controller
Run/Schedule application
Initiate the application execution
Process the application request
Secure and private apace ready
Secure and private apace not available
Process the application execution request
Visualize analytics results
Retrieve encrypted data
Store encrypted results
Receive Secure and Private Space deployment request
Deploy Secure and Private Space
Receive worker deployment request
Processworker deployment request
Request encrypted data
Receive encrypted results
Scheduled application settings
Application settings
Encrypted Data
Request for worker deployment
Worker settings
Encrypted result & Execution Statuss
Secure and Private Space
SecureSpace Worker
Decryption and encryption managers
Storage
Job scheduler
Execution Engine
Executor cluster
Receive encrypted data
Transfer encrypted results and status
Decrypt Data
Encrypt Data
Store temporary data
Store analytics results
Schedule job
Collect status and results
Prepare application execution
Produced results
Receive workers deployment request
Deploy workers for application
Run algorithms

Figure 2-8: ICARUS application execution workflow

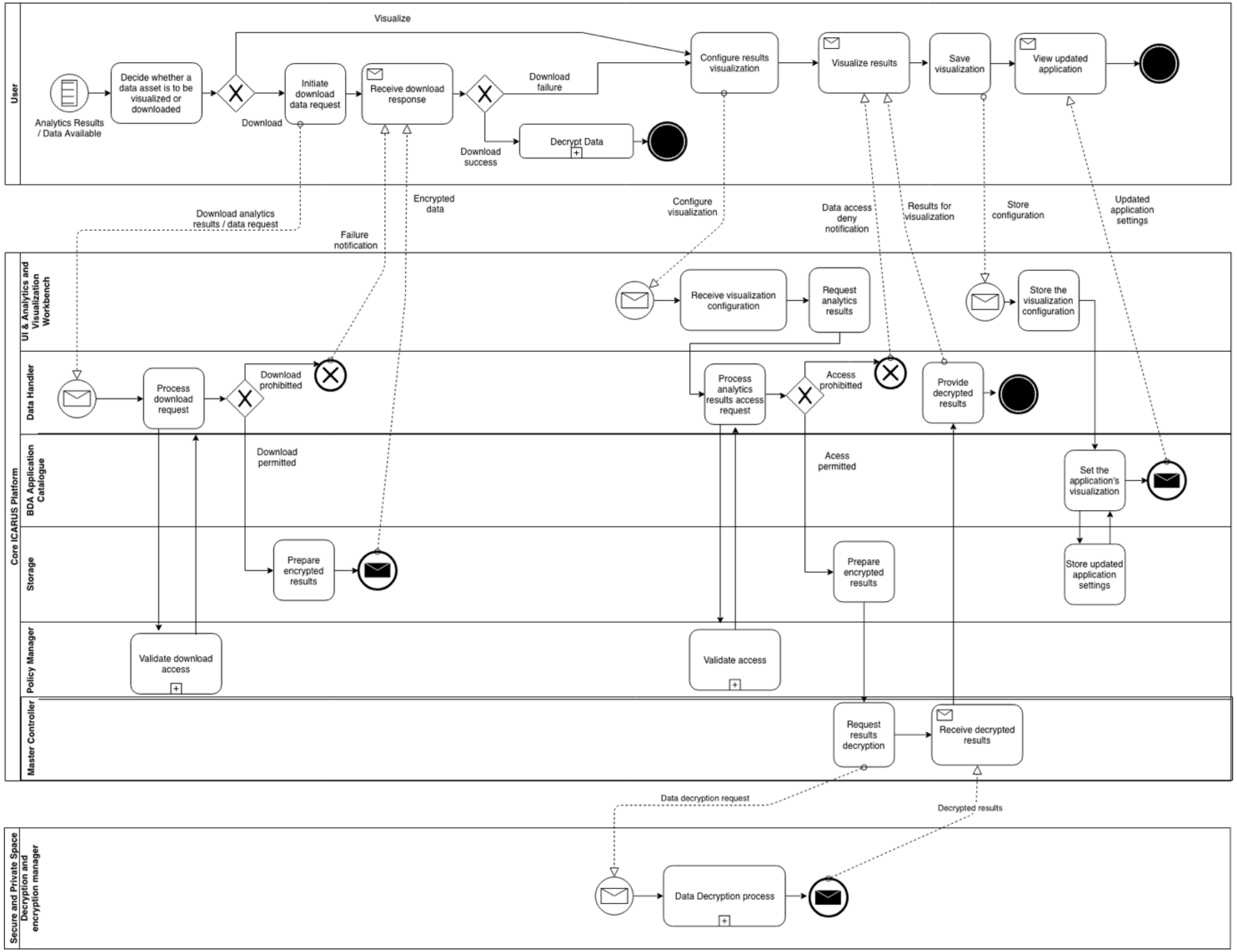
User
Analytics Results / Data Available
Decide whether a data asset is to be visualized or downloaded
Visualize
Download
Initiate download data request
Receive download response
Download failure
Download success
Decrypt Data
Configure results visualization
Visualize results
Save visualization
View updated application
Download analytics results / data request
Failure notification
Encrypted data
Configure visualization
Data access deny notification
Results for visualization
Store configuration
Updated application settings
Core ICARUS Platform
UI & Analytics and Visualization Workbench
Receive visualization configuration
Request analytics results
Store the visualization configuration
Data Handler
Process download request
Download prohibited
Download permitted
Process analytics results access request
Access prohibited
Acess permitted
Provide decrypted results
BDA Application Catalogue
Set the application's visualization
Storage
Prepare encrypted results
Prepare encrypted results
Store updated application settings
Policy Manager
Validate download access
Validate access
Master Controller
Request results decryption
Receive decrypted results
Data decryption request
Decrypted results
Secure and Private Space Decryption and encryption manager
Data Decryption process

Figure 2-9: Visualise the execution results of an ICARUS application

2.4.2 Involved services

The presented workflows were analysed towards the identification of the appropriate services that will facilitate the realisation of the workflows, as well as the realisation of the designed ICARUS platform's offerings. The list of identified services is presented in the following table.

Table 2-3: Data Analytics and Visualisations services

Number	Title/ Description	Interacting Components	Related Services Group(s)
DAV_1	Analytics design graphical user interface	Analytics and Visualisation Workbench - UI	Data Analytics
DAV_2	Collect list of user's assets	Analytics and Visualisation Workbench, ICARUS Storage & Indexing	Data Analytics
DAV_3	Analytics workflow design	Analytics and Visualisation Workbench - UI	Data Analytics
DAV_4	ICARUS application storage	Analytics and Visualisation Workbench, BDA Application Catalogue	Data Analytics, Application Catalogue
DAV_5	ICARUS Application execution scheduling	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine	Data Analytics, Application Execution
DAV_6	Secure and Private Space provisioning	Analytics and Visualisation Workbench, Resource Orchestrator	Resource Orchestration
DAV_7	ICARUS Application execution	Job Scheduler and Execution Engine, Execution Cluster	Data Analytics, Application Execution, Application Catalogue
DAV_8	Datasets retrieval for the application execution	Job Scheduler and Execution Engine, Decryption Manager, Encryption Manager, SecureSpace Worker, Master Controller, ICARUS Storage & Indexing	Data Analytics, Application Execution Master and Worker, Data Encryption and Decryption
DAV_9	Workers deployment request	Job Scheduler and Execution Engine, Resource Orchestrator, Execution Cluster	Application Execution, Master and Worker, Resource Orchestration
DAV_10	Execution results secure storage	Job Scheduler and Execution Engine, Encryption Manager, SecureSpace Worker, Master Controller, ICARUS Storage & Indexing	Application Execution, Master and Worker, Data Encryption and Decryption
DAV_11	Execution status monitoring	Analytics and Visualisation Workbench, Job Scheduler and Execution Engine, Execution Cluster	Data Analytics, Application Execution, Master and Worker
DAV_12	Export of the execution results	Analytics and Visualisation Workbench, Decryption Manager, UI	Data Analytics, Data Encryption and Decryption

Number	Title/ Description	Interacting Components	Related Services Group(s)
DAV_13	Visualisation of the execution results	Analytics and Visualisation Workbench, Decryption Manager, UI	Data Analytics, Data Encryption and Decryption
DAV_14	Storage of the visualisation configuration in the ICARUS application	Analytics and Visualisation Workbench, BDA Application Catalogue	Data Analytics, Application Catalogue

2.5 Data Exploration

2.5.1 Designed workflows

Dataset exploration is of utmost importance in the ICARUS platform, as it is the first step towards identifying and acquiring useful datasets that can be leveraged for insightful data analytics and data-driven decision making in the aviation domain. As the ICARUS platform addresses the needs of both data providers and data consumers, there are two core workflows that fall under the data exploration set of processes, one addressing the exploration of datasets owned by the user and one addressing the process of exploring datasets not owned by the user.

The exploration of owned datasets can be thought of as starting directly from a dataset's profile page within the ICARUS platform. In contrast, in the case of exploration of datasets not owned by the user, the first step would be for the user to search for datasets. The ICARUS platform enables the users to perform advanced and complex queries to help them discover useful datasets and dataset combinations. This is achieved mainly through the functionalities provided by the Query Explorer component which have been described in detail in D3.1 and are also outlined here through the BPMN diagrams. Prior to the actual dataset exploration, the user needs to perform a search, a workflow modelled in the following BPMN diagram.

Figure 2-10 shows the core steps of the query creation process. All search queries are saved so that the user can easily re-execute them in the future, either as-is or after properly updating their configuration. There are three types of query configurations that can be made by the user: (i) at the data model level, allowing the user to select the data fields that should be included in the datasets (e.g. airport codes and aircraft take-off times), (ii) at the metadata level, allowing the user to filter the datasets that will be included in the results based on their metadata properties (e.g. data provider organisation) and (iii) at the data level, allowing the user to filter the results based on actual values included in the datasets when possible (e.g. select a date range for the take-off times provided in a dataset column). Then, the Query Explorer (with the help of the Policy Manager) transforms the configuration into a valid query which returns the set of results that the user is eligible to view.

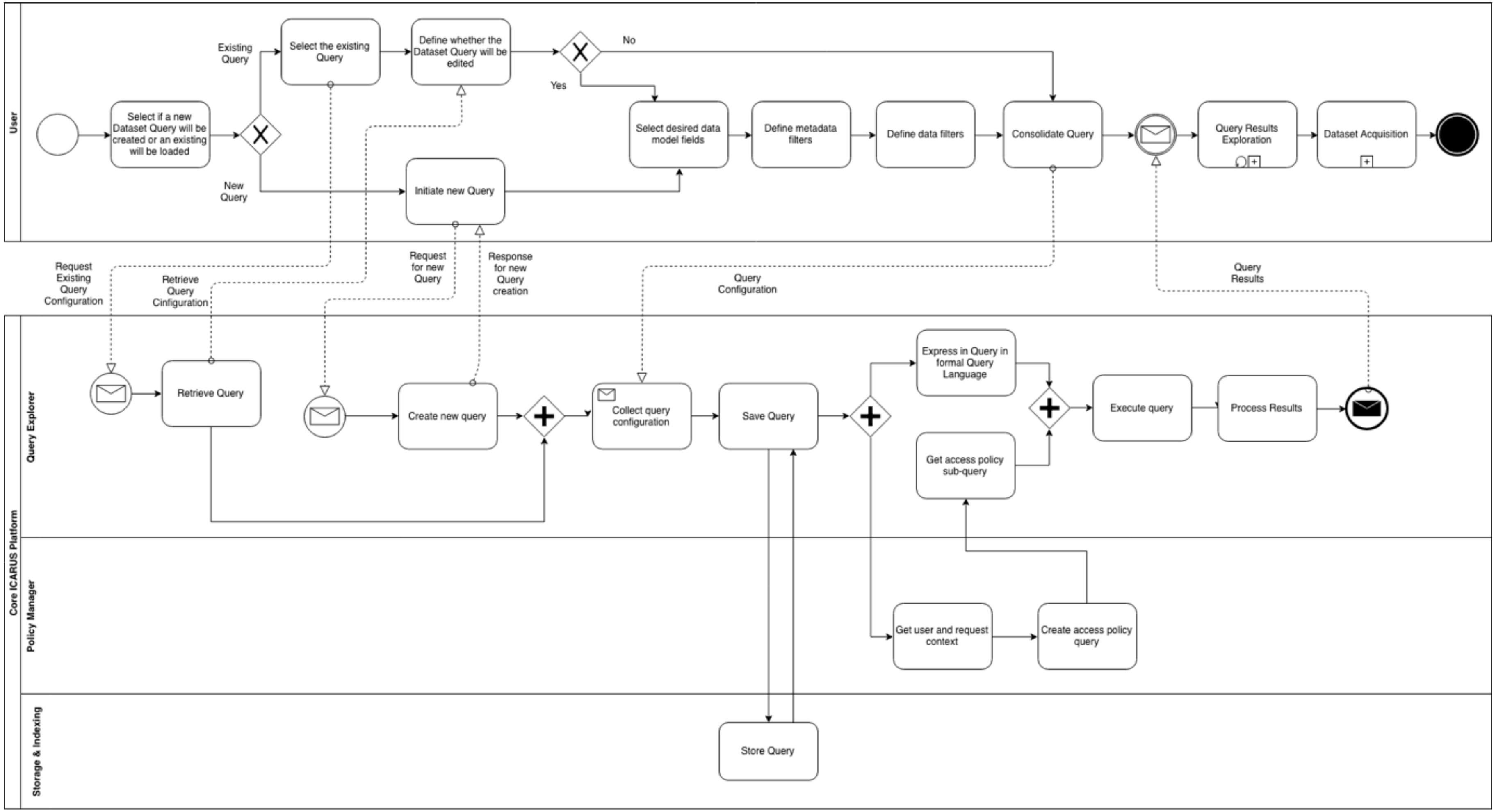
User
Select if a new Dataset Query will be created or an existing will be loaded
Existing Query
New Query
Select the existing Query
Define whether the Dataset Query will be edited
No
Yes
Initiate new Query
Select desired data model fields
Define metadata filters
Define data filters
Consolidate Query
Query Results Exploration
Dataset Acquisition
Request Existing Query Configuration
Retrieve Query Cinfiguration
Request for new Query
Response for new Query creation
Query Configuration
Query Results
Core ICARUS Platform
Query Explorer
Retrieve Query
Create new query
Collect query configuration
Save Query
Express in Query in formal Query Language
Get access policy sub-query
Execute query
Process Results
Policy Manager
Get user and request context
Create access policy query
Storage & Indexing
Store Query

Figure 2-10: Datasets Query Process

Once the results are presented in the Query Explorer interface, the user can select specific datasets to explore, a process modelled under the "Dataset Exploration" BPMN diagram shown in Figure 2-11. The "Dataset Acquisition" process shown in Figure 2-11 is related to the data brokerage services and will be therefore presented in Section 2.6.

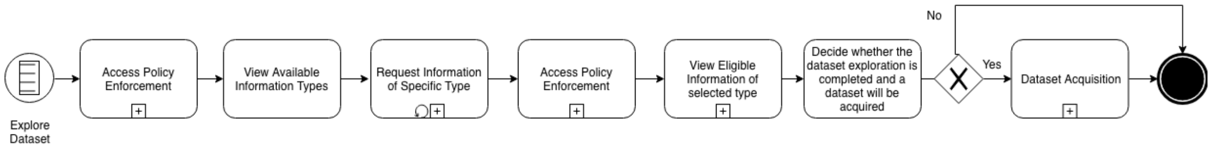
Explore Dataset
Access Policy Enforcement
View Available Information Types
Request Information of Specific Type
Access Policy Enforcement
View Eligible Information of selected type
Decide whether the dataset exploration is completed and a dataset will be acquired
No
Yes
Dataset Acquisition

Figure 2-11: Dataset Exploration

Regarding the dataset exploration shown in Figure 2-11, it should be highlighted that in the current context, the term dataset exploration does not refer to the exploration of the actual content / data of the dataset, since this is not possible inside the core ICARUS platform for security reasons, even for the datasets owned by the user. Content exploration and dataset usage are possible either in the user's premises or inside an ICARUS secure and private space. The term exploration here refers to all information about the dataset, i.e. at a metadata level, which is available to a user. The different information types mentioned in Figure 2-11 are based on the ICARUS metadata schema (defined in D2.1) and include the following:

- Core dataset metadata, e.g. provider
- Dataset schema
- Dataset license metadata
- Dataset provenance metadata
- Access policies defined for the dataset (accessible only by the data provider)
- Dataset sample
- Contracts involving the dataset (accessible only by the data provider)

As it can be seen, the access policy enforcement sub-process, that is performing the access control process and is presented in Section 2.7, is invoked twice: at the beginning of the dataset exploration in order to identify which of the information types will be shown as options to the user and in the end to determine the exact information of the selected type that the user is eligible to view. This ensures that certain users will not even have knowledge of the type and scope of the information that is available for a dataset, if so required by the dataset provider.

2.5.2 Involved services

Based on the workflows presented in the previous sub-section, the following services are foreseen:

Table 2-4: Data Exploration Services

Number	Title/ Description	Interacting Components	Related Services Group(s)
DE_1	ICARUS Common Aviation Data Model Retrieval	ICARUS Storage & Indexing (available to be invoked by various other components)	ICARUS Backbone services
DE_2	ICARUS Metadata Model Retrieval	ICARUS Storage & Indexing(available to be invoked by various other components)	ICARUS Backbone services
DE_3	Query configuration creation	Query Explorer, UI	Dataset Exploration
DE_4	Query execution	Query Explorer, UI	Dataset Exploration
DE_5	Creation and retrieval of the access policy query clause	Query Explorer, Policy Manager	Dataset Exploration, Policies Enforcement
DE_6	Query results retrieval	Query Explorer, UI	Dataset Exploration
DE_7	Provenance information retrieval for a specific dataset	UI, ICARUS Storage & Indexing	Metadata Handling
DE_8	Core metadata retrieval for a specific dataset	UI, ICARUS Storage & Indexing	Metadata Handling
DE_9	License metadata retrieval for a specific dataset	UI, ICARUS Storage & Indexing	Data Licensing
DE_10	Data sample retrieval for a specific dataset	UI, ICARUS Storage & Indexing	Dataset Exploration
DE_11	Data contracts retrieval for specific dataset	UI, Blockchain	Asset Brokerage
DE_12	Data access policy information retrieval for specific dataset	UI, ICARUS Storage & Indexing	Policies Enforcement
DE_13	Dataset schema (mapping to ICARUS model) retrieval	UI, ICARUS Storage & Indexing	Data Mapping
DE_14	Storing of search queries	Query Explorer, ICARUS Storage & Indexing	Dataset Exploration
DE_15	Retrieval of search queries from storage	Query Explorer, ICARUS Storage & Indexing	Dataset Exploration

2.6 Asset Brokerage

2.6.1 Designed workflows

The term asset brokerage encapsulates all actions and interactions from the moment an ICARUS user decides to request to purchase an asset owned by another user, until the request has been either rejected or successfully fulfilled and the asset is obtained. Although the word asset refers both to datasets and applications and ICARUS foresees brokerage functionalities for both, the current section will be limited to dataset brokerage. Applications bring significant complexities in terms of license and provenance and their brokerage will be discussed in subsequent WP3 deliverables.

The dataset brokerage workflow has been discussed in detail from a conceptual point of view in D2.2. Here, as shown in the following BPMN diagram, the focus is on the underlying services that are required to implement the foreseen processes. As the complete process is very complex and includes numerous interactions, it has been broken down to two separate diagrams mainly for demonstration reasons, as follows:

Phase I: The first phase of the data brokerage process includes all steps up to the point a smart data contract is created in the blockchain. The process starts when a user issues a request to purchase a specific dataset. The request is stored in the ICARUS platform and the data provider is notified. The request may be rejected, in which case no contract is created and the process never reaches Phase II. If the data provider decides to consider the request, the next step is the creation of a smart contract (i.e. the definition of its terms). Once the smart contract is stored in the blockchain under the draft status, the brokerage process will enter Phase II.

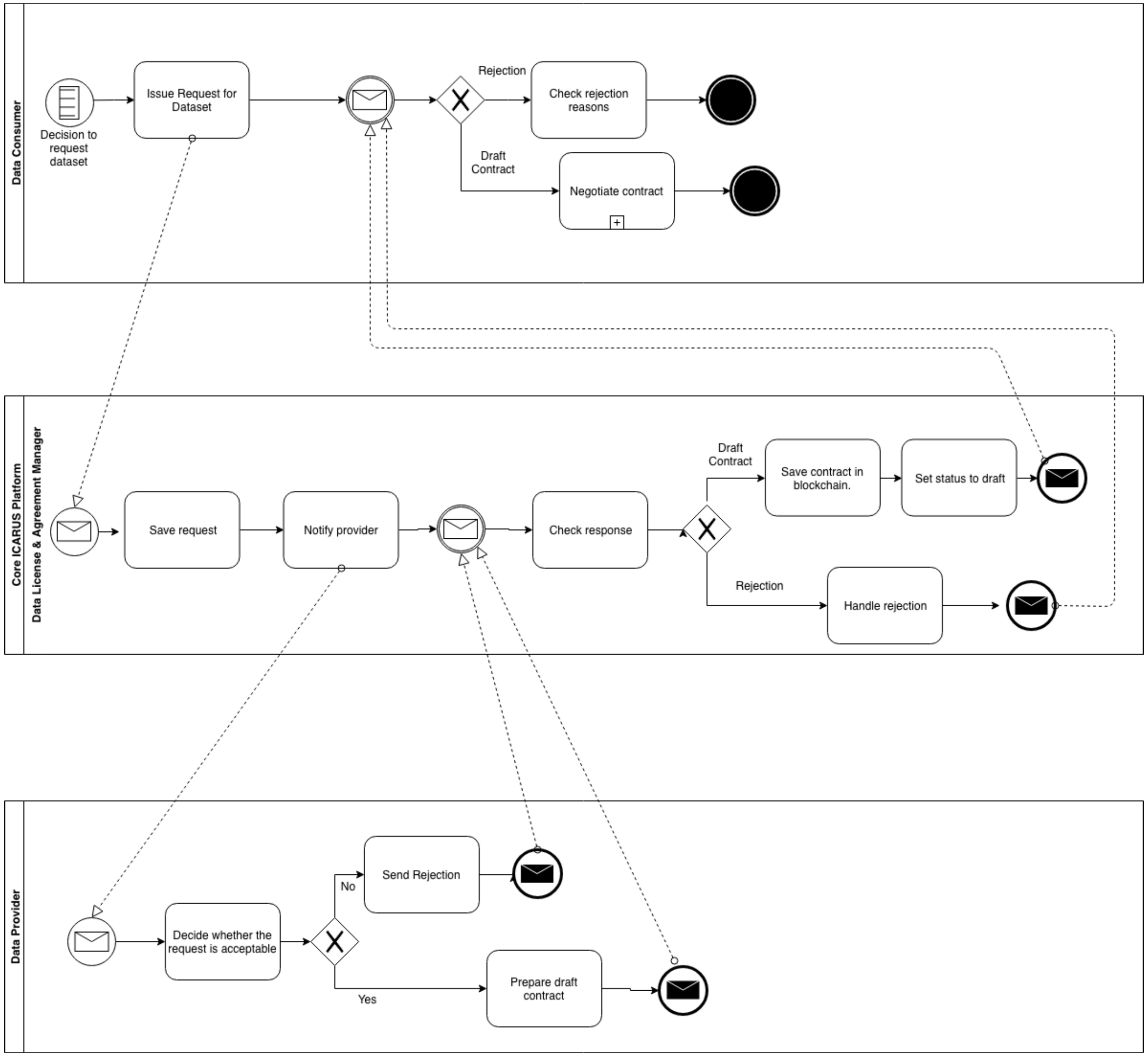

Figure 2-12: Dataset Brokerage Workflow Phase I

Phase II: The second phase includes all smart contract review and negotiation steps by both the data consumer and the data provider. It will end with the data contract being rejected by either party, or accepted by both, in which case the process continues with the payment (performed externally to ICARUS platform) and the validation of the payment by the data provider, which denotes the successful completion of the data brokerage process. In brief, the following negotiation steps are foreseen:

When a data consumer is notified for the creation of the smart contract, he/she will review its terms and may:

i) Reject it, in which case the contract status changes to rejected and the process ends.

ii) Accept it, in which case the contract status changes to accepted, the negotiation is over and the consumer may proceed to make the payment.

iii) Update its terms, in which case the new terms are saved in the blockchain and the contract status is set to negotiating.

When a data provider is notified for the update of the smart contract terms, he/she will review the contract and may:

i) Reject it, in which case the contract status changes to rejected and the process ends

ii) Accept it, in which case the contract status changes to accepted, the negotiation is over and the consumer may proceed to make the payment

iii) Update its terms, in which case the new terms are saved in the blockchain and the contract status is set to draft, in which case the consumer will again be notified to review the contract and the process may repeat this loop until an agreement is reached or the contract is rejected.

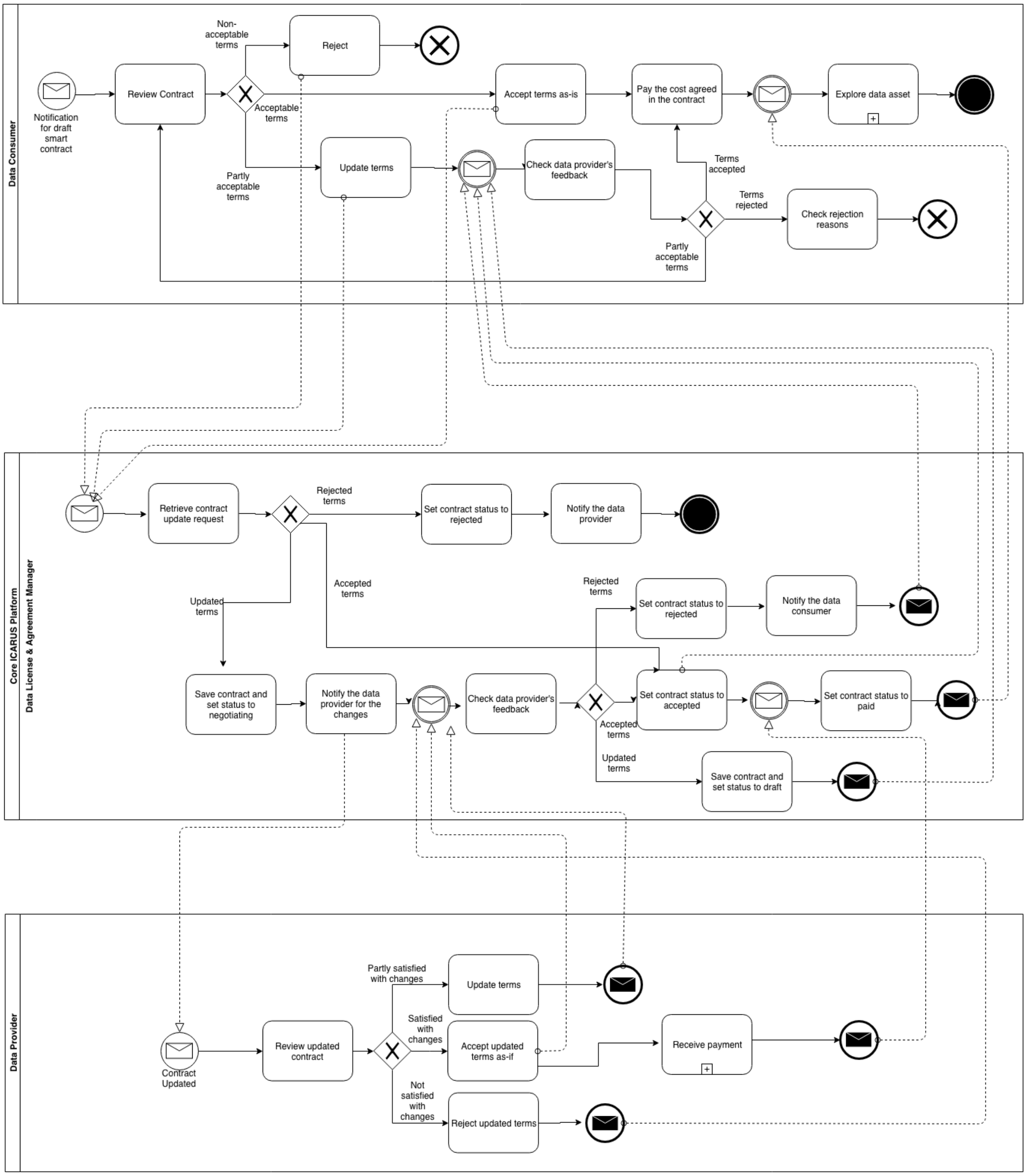
Data Consumer
Non-acceptable terms
Reject
Notification for draft smart contract
Review Contract
Acceptable terms
Accept terms as-is
Pay the cost agreed in the contract
Explore data asset
Partly acceptable terms
Update terms
Check data provider's feedback
Terms accepted
Terms rejected
Check rejection reasons
Partly acceptable terms
Core ICARUS Platform
Data License & Agreement Manager
Retrieve contract update request
Rejected terms
Set contract status to rejected
Notify the data provider
Accepted terms
Updated terms
Rejected terms
Set contract status to rejected
Notify the data consumer
Save contract and set status to negotiating
Notify the data provider for the changes
Check data provider's feedback
Set contract status to accepted
Set contract status to paid
Accepted terms
Updated terms
Save contract and set status to draft
Data Provider
Partly satisfied with changes
Update terms
Satisfied with changes
Contract Updated
Review updated contract
Accept updated terms as-if
Receive payment
Not satisfied with changes
Reject updated terms

Figure 2-13: Dataset Brokerage Workflow Phase II

2.6.2 Involved services

Based on the workflows presented in the previous sub-section, the following services are foreseen:

Table 2-5: Asset Brokerage services

Number	Title/ Description	Interacting Components	Related Services Group(s)
AB_1	Request to purchase dataset	UI, Data Licence and Agreement Manager	Asset Brokerage
AB_2	Save request to purchase dataset	UI, ICARUS Storage & Indexing	Asset Brokerage

Number	Title/ Description	Interacting Components	Related Services Group(s)
AB_3	Notify data provider of purchase request	UI, ICARUS Storage & Indexing	Notifications
AB_4	Notify data consumer of request response	UI, ICARUS Storage & Indexing	Notifications
AB_5	Reject request prior to data contract creation	UI, Data Licence and Agreement Manager, ICARUS Storage & Indexing	Asset Brokerage
AB_6	Reject request after data contract has been created	UI, Data Licence and Agreement Manager, Blockchain	Asset Brokerage
AB_7	Change data contract status: draft, negotiating, accepted, paid, rejected	UI, Data Licence and Agreement Manager, Blockchain	Asset Brokerage
AB_8	Create data contract	UI, Data Licence and Agreement Manager	Asset Brokerage
AB_9	Store data contract in blockchain	Data Licence and Agreement Manager, Blockchain	Asset Brokerage
AB_10	Update data contract	UI, Data Licence and Agreement Manager	Asset Brokerage
AB_11	Update data contract in blockchain	Data Licence and Agreement Manager-Blockchain	Asset Brokerage
AB_12	Retrieve data contract from blockchain	UI, Blockchain (through the Data Licence and Agreement Manager and the Wallet Manager)	Asset Brokerage
AB_13	Notify users of changes in the data contract status	UI, Data Licence and Agreement Manager-Blockchain	Asset Brokerage

2.7 Data Security

2.7.1 Designed workflows

Data safeguarding is one of the cornerstones of the ICARUS platform towards the protection of the underlying assets and the affected entities from any potential abuse or unauthorised access to any critical or valuable assets or resources.

Within this scope, the ICARUS platform follows a twofold approach as defined in D2.1: (a) the access to any type of asset or resource of the ICARUS platform is regulated through the access control mechanism that is based on Attribute-Based Access Control (ABAC) policies and the eXtensible Access Control Markup Language (XACML) Version 3.0 standard, and (b) the privacy and integrity of the data assets is ensured with a data encryption / decryption process that safeguards the secure and tamper proof storage and transfer of them between the sender and the recipient.

A prerequisite for the correct usage of the access control mechanism is the design of an effective user management process. In ICARUS platform, this process includes two main sub-processes: a) the registration and subsequently user creation, and b) the user authentication (login) that enables the access to the platform as a whole. In ICARUS platform, the users appear under the concept of organisations. An organisation represents a group of users with the same interests or purposes under the same company (i.e. an airline company) and each organisation consists of the organisation manager, that registers the organisation and invites the members of the organisation, and the organisation members. The registration process is handled by the Policy Manager component and includes the following steps, as depicted also in Figure 2-14:

a) The organisation manager submits the organisation signup form.

b) The ICARUS administrator receives the request, checks and approves it.

c) The organisation manager creates the invitations for the organisation members. Each member receives the invitation link via email accompanied with an invitation token.

d) Each organisation member fills-in the member signup form providing also the invitation token and, upon successful registration, access is granted to the ICARUS platform.

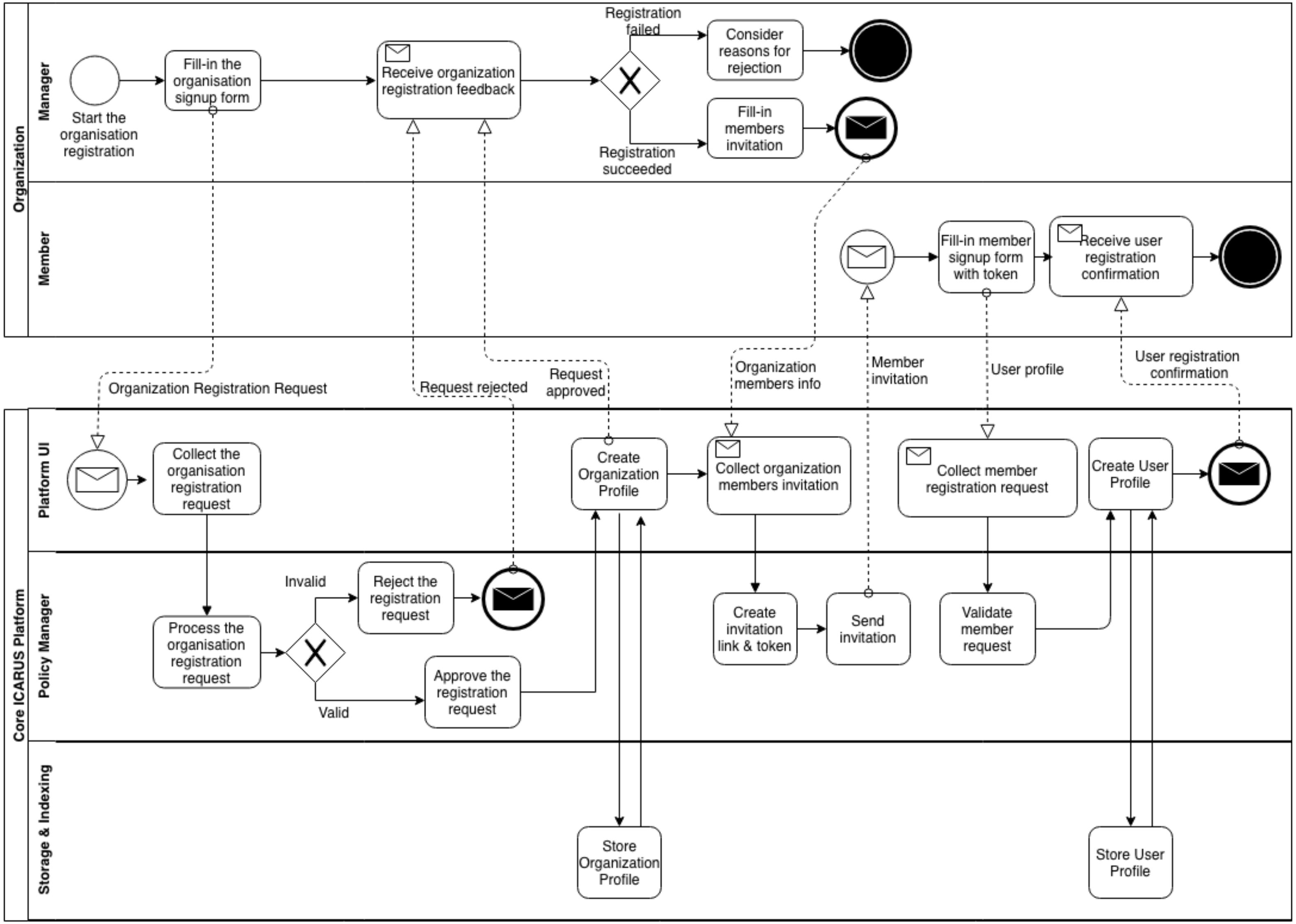
Organization
Manager
Member
Start the organisation registration
Fill-in the organisation signup form
Receive organization registration feedback
Registration failed
Consider reasons for rejection
Registration succeeded
Fill-in members invitation
Fill-in member signup form with token
Receive user registration confirmation
Organization Registration Request
Request rejected
Request approved
Organization members info
Member invitation
User profile
User registration confirmation
Core ICARUS Platform
Platform UI
Policy Manager
Storage & Indexing
Collect the organisation registration request
Process the organisation registration request
Invalid
Valid
Reject the registration request
Approve the registration request
Create Organization Profile
Store Organization Profile
Collect organization members invitation
Create invitation link & token
Send invitation
Collect member registration request
Validate member request
Create User Profile
Store User Profile

Figure 2-14: Registration process workflow

In the user authentication sub-process, also handled by the Policy Manager component, the user, upon his/her successful registration, accesses the login form of the ICARUS platform and provides his/her credentials. The Policy Manager performs the authentication process and either grants the user access to the ICARUS platform or denies it in the case of invalid credentials.

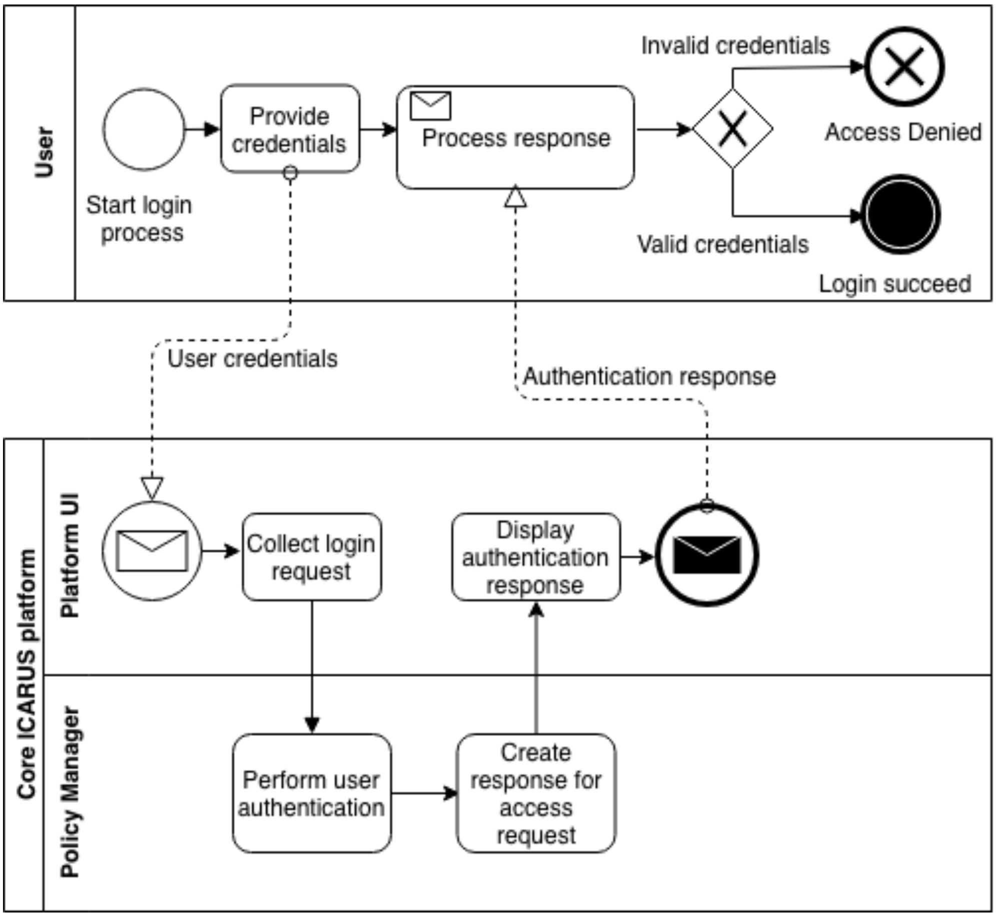

Figure 2-15: Login process workflow

The access policy design workflow is incorporating the actions that are performed with regard to the policy creation process for the data provider's private and confidential datasets or ICARUS applications that have been prepared and uploaded to the ICARUS platform or designed through the relevant tools of the ICARUS platform. Hence, the scope of this process is to define the access policy details of the datasets or applications that will be fed into the access control mechanism and will be checked every time an access request is performed for these datasets or applications.

The data provider is provided with a straightforward and easy-to-use user interface that facilitates the definition of all parameters related to a data access policy according to the data provider's needs. Figure 2-16 illustrates the interactions between the data provider and the ICARUS platform interface for the access policy definition and how these interactions are interpreted and processed within the ICARUS platform through the related services.

In this workflow, the user initiates the policy creation process for a specific dataset or application and defines all the policy related parameters. The new policy is created by the Policy Manager and the corresponding parameters are set. Once the user finalises the new policy, a request for the new policy activation is triggered. The Policy Manager undertakes responsibility for storing the new policy on the dedicated ICARUS storage and for deploying the new policy in the authorisation engine. Once the authorisation engine has been updated, a confirmation is returned to the user that the new policy is in place and that it will be validated in any upcoming access request in order to formulate the access decision.

The access policy enforcement workflow is consolidating the execution of the access control process in which the selective access to any asset is performed. Within this process, the access control mechanism is utilised in order to intercept all access requests for any asset and formulate an access control decision based on the evaluation of the access control policies that are defined for this asset. Thus, the purpose of this process is to enforce the access control policies that are defined in the access policy design workflow in order to regulate the access to any asset. The access control process is an internal continuous process of the ICARUS platform and the user is not interacting with it.

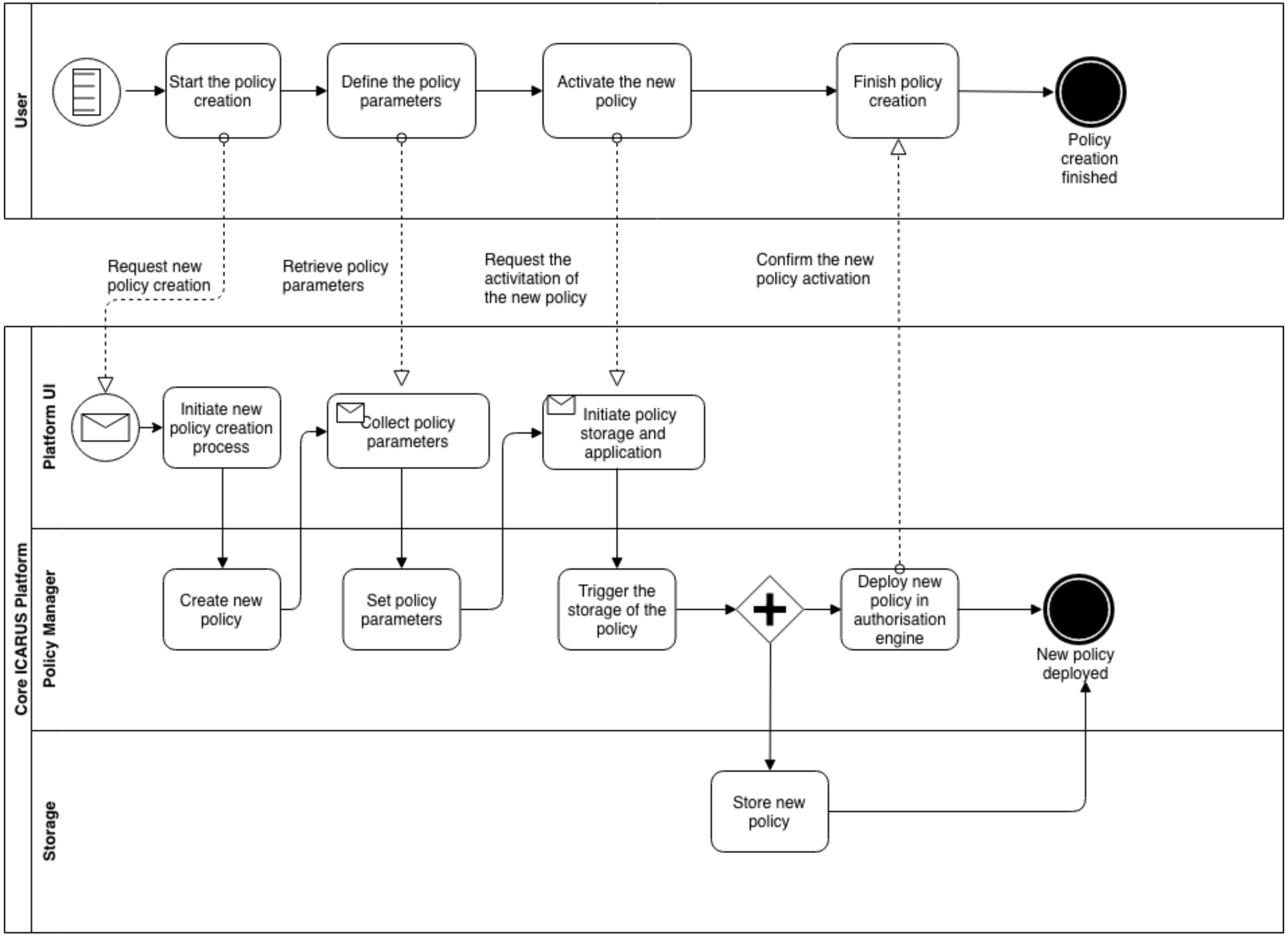
User
Start the policy creation
Define the policy parameters
Activate the new policy
Finish policy creation
Policy creation finished
Request new policy creation
Retrieve policy parameters
Request the activitation of the new policy
Confirm the new policy activation
Core ICARUS Platform
Platform UI
Initiate new policy creation process
Collect policy parameters
Initiate policy storage and application
Policy Manager
Create new policy
Set policy parameters
Trigger the storage of the policy
Deploy new policy in authorisation engine
New policy deployed
Storage
Store new policy

Figure 2-16: Access policy creation workflow

Figure 2-17 illustrates the access control process that is realising the access policy enforcement workflow. In this workflow, the process receives an access request for an asset and parses this request. Based on the content of the request, the relevant policies are retrieved and the required attributes included in the definition of these rules are collected. Once all information is available, the access control decision making takes place and an access decision is formulated.

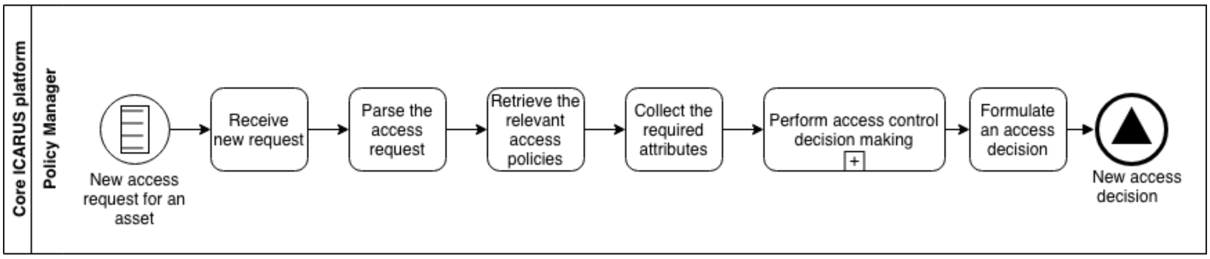
Core ICARUS platform
Policy Manager
New access request for an asset
Receive new request
Parse the access request
Retrieve the relevant access policies
Collect the required attributes
Perform access control decision making
Formulate an access decision
New access decision

Figure 2-17: Access policy enforcement workflow

The data encryption / decryption process on its behalf involves two separate main phases: (a) the encryption of the data assets which is executed as part of the data preparation as described in the data preparation workflow presented in section 2.2 and in the transfer of the derivative data from the Secure and Private Space to the Core ICARUS platform as described in section 2.4, and (b) the decryption of the encrypted ciphertext which is executed when a data consumer acquires access to the unencrypted data asset either on the On Premise Environment or the Secure and Private Space.

The encryption of the data assets is performed following the instructions provided by the data provider as described in section 2.2 and it is a column-based encryption. In these instructions, the data provider selects the columns of the data asset that require encryption. It should be noted that the columns that contain certain spatio-temporal information are not eligible for encryption in order to ensure the smooth and efficient data browsing and exploration. The data encryption workflow is an internal process where the user of the platform is not involved and it is presented in the following figure.

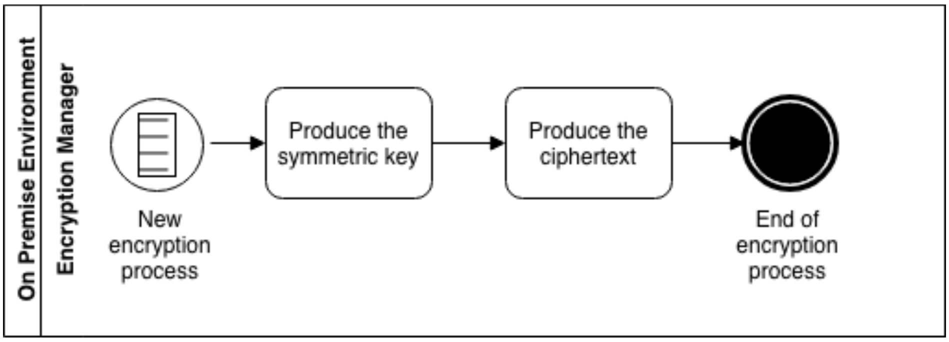

Figure 2-18: Data encryption workflow

In this workflow, as a first step the symmetric key that will be used in the encryption process is produced. Following the symmetric key generation, the encryption method is applied on top of the selected dataset with this symmetric key on the columns that are pointed out by the data provider. The result of the process is a ciphertext that safeguards the data privacy and integrity of the data asset. As described also in section 2.2, this process is executed within the On Premise Environment of the data provider ensuring that the data asset is never transmitted from the On Premise Environment to the Core ICARUS platform in an unencrypted and vulnerable state.

The decryption of these data assets involves several actions and validations that are performed in order to enable the secure transfer and the confidentially of the data assets. At first, a secure SSL-enabled connection is established with the proper SSL handshakes between the data consumer and the data provider in order to exchange the symmetric key that will be used for the decryption of the ciphertext. In this step, the Decryption Manager and the Encryption Manager are connected with the help of the Key Pair Administrator who is responsible for performing the signalling operations for the two-side authorisation and the identity verification. Secondly, the access to the requested data asset is validated with the execution of the access policy enforcement workflow, with the help of the Policy Manager and the Data License and Agreement Manager, in order to verify the eligibility of the data consumer. If the access has been granted to the data consumer, then a decryption symmetric key is provided from the data provider to the data consumer through the secure connection. Finally, the data consumer utilises the received decryption symmetric key and decrypts the ciphertext. The data decryption workflow is also in an internal process and it is presented in Figure 2-19. The diagram illustrates the decryption process for datasets that are utilised in the Secure and Private Space during the data analysis execution. It should be noted that the decryption process is not differentiated in the case where a data asset is downloaded and decrypted locally on the On Premise Environment by the Decryption Manager running on the On Premise Environment.

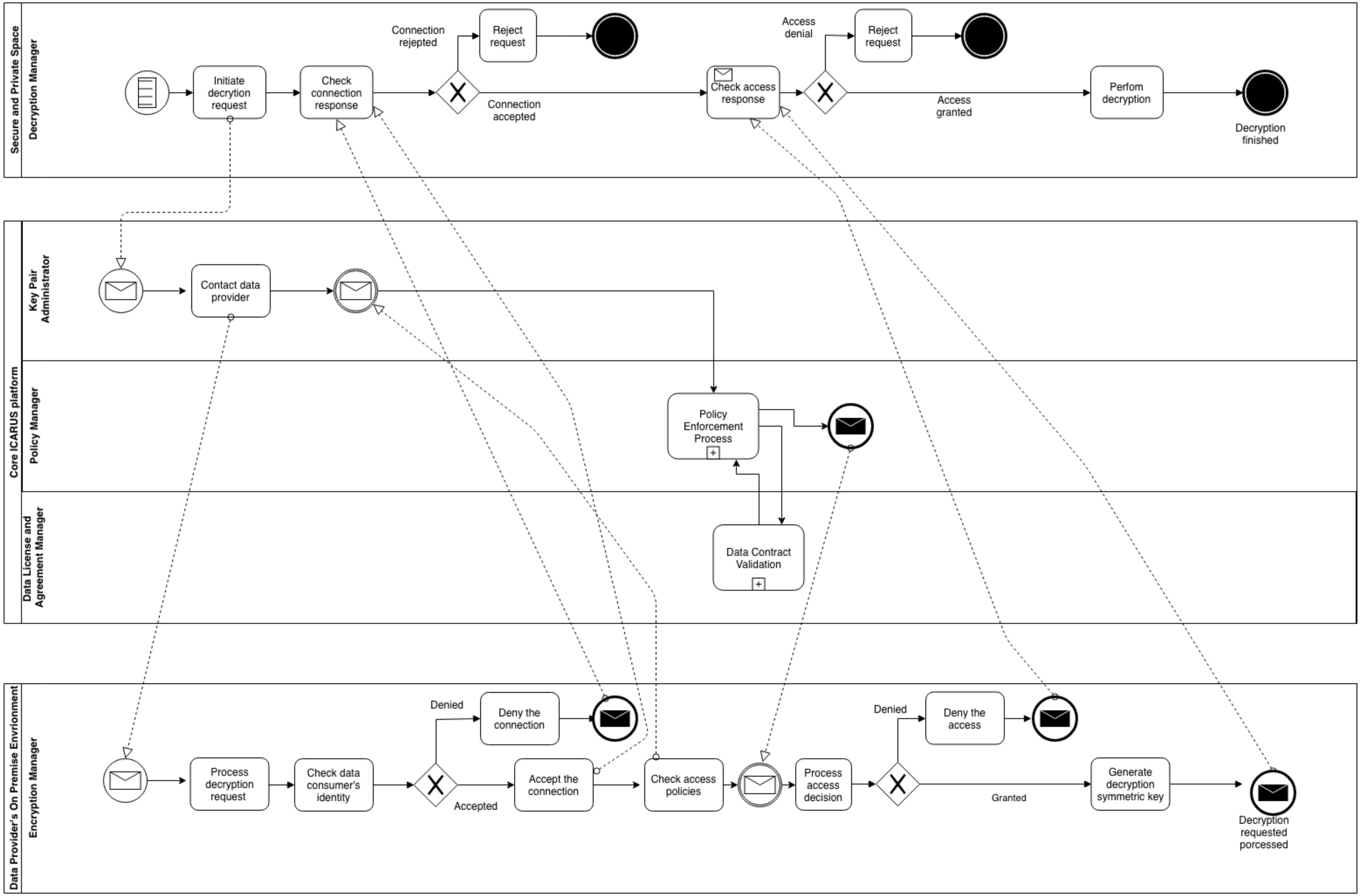

Figure 2-19: Data decryption workflow

2.7.2 Involved services

The following table contains the list of identified services as extracted from the presented workflows. The scope of these services is to provide the means and functionalities required for the execution of the workflows.

Table 2-6: Data Security services

Number	Title/ Description	Interacting Components	Related Services Group(s)
DS_1	Registration process	Policy Manager, UI	Policies Enforcement
DS_2	Login process	Policy Manager, UI	Policies Enforcement
DS_3	Access policy creation	Policy Manager, ICARUS Storage & Indexing, UI	Policies Enforcement
DS_4	Access policy storage	Policy Manager, ICARUS Storage & Indexing	Policies Enforcement
DS_5	Deploy new policy in the authorisation engine	Policy Manager	Policies Enforcement
DS_6	Access policy enforcement	Policy Manager	Policies Enforcement
DS_7	Perform data encryption of the dataset	Encryption Manager	Data Encryption and Decryption
DS_8	Establish connection between the data consumer and the data provider	Key Pair Administrator, Decryption Manager, Encryption Manager	Data Encryption and Decryption
DS_9	Validate data access request	Encryption Manager, Policy Manager	Data Encryption and Decryption, Policies Enforcement
DS_10	Perform key decryption key exchange between the data consumer and the data provider	Key Pair Administrator, Decryption Manager, Encryption Manager	Data Encryption and Decryption
DS_11	Perform data decryption of the dataset	Encryption Manager, Decryption Manager, Key Pair Administrator, Policy Manager, Data License and Agreement Manager	Data Encryption and Decryption, Policies Enforcement, Asset Brokerage

2.8 Backend Ancillary Processes

2.8.1 Designed workflows

In the ICARUS platform architecture, as presented in the ICARUS Deliverable D3.1, the ICARUS platform is conceptually divided in three main tiers: the On Premise Environment, the Core ICARUS platform and the Secure and Private Space. In order to perform all the envisioned platform operations and provide the required functionalities that will address the ICARUS stakeholder's needs, it is crucial that these three tiers are interconnected and an intercommunication between them is established. For this purpose, a set of ancillary processes is designed, namely the resource orchestration process and master / worker process.

The Secure and Private Space is the isolated and secure environment that is provisioned on-demand through the resource orchestration process by the Core ICARUS platform in order to provide the “sandboxed” environment where the user is able to perform data analysis. The scope of this process is the provisioning and management of this isolated and secure environment using a set of techniques and technologies for the dynamic, fast and secure deployment over a virtualised infrastructure. The resource orchestration process is an internal process of the ICARUS platform that is transparent to the user, thus the user is not interacting with it.

Under the hood, when the data consumer initiates a data analysis request, the resource orchestration process will provide the Secure and Private Space in which this data analysis will be executed. Thus, the resource orchestration process is undertaking the responsibility for the preparation of the required “sandboxed” environment. Once the data analysis is complete, the Secure and Private Space is stopped and remains unattached for future usage. In the case of a scheduled job execution, the Secure and Private Space remains active till the completion of the scheduled job.

To this end, the Analytics and Visualisation Workbench will request from the Resource Orchestrator to prepare and provide a Secure and Private Space environment. Figure 2-20 illustrates the interactions between the Analytics and Visualisation Workbench and the Resource Orchestrator within the Core ICARUS platform for the provisioning and stoppage of the Secure and Private Space.

In this workflow, the Analytics and Visualisation Workbench, upon receiving a request for a new analytics job execution, requests the provisioning of the Secure and Private Space from the Resource Orchestrator. The Resource Orchestrator initiates the provisioning of the Secure and Private Space in the virtualised infrastructure and deploys the required services on the Secure and Private Space. Once the deployment is complete, the Analytics and Visualisation Workbench is informed to start the analytics job execution. When the analytics job execution is finished, the Analytics and Visualisation Workbench requests the stoppage of the Secure and Private Space. If there is an analytics job scheduled for execution, the Secure and Private Space remains active till this job is successfully executed, otherwise the Resource Orchestrator stops the Secure and Private Space and the Analytics and Visualisation Workbench completes the job execution.

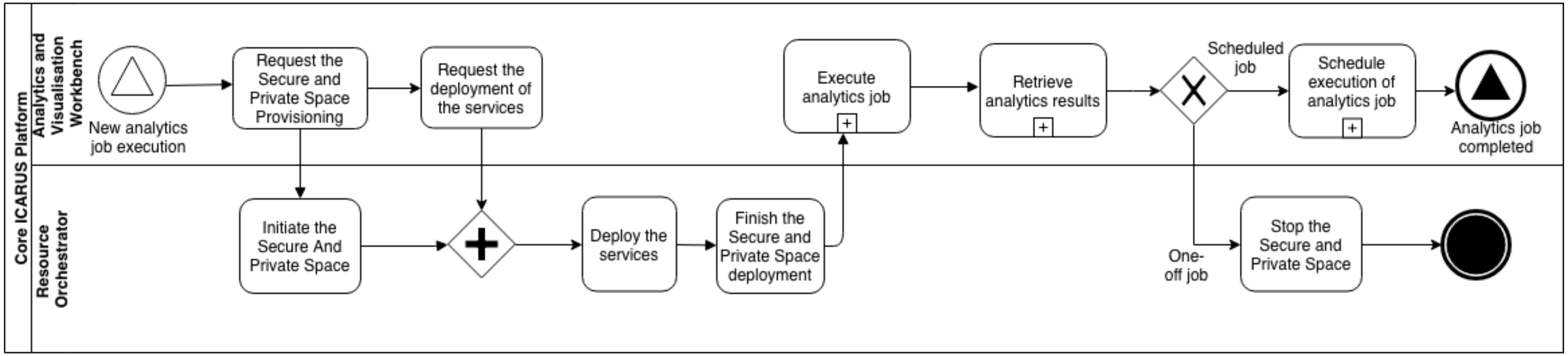
Core ICARUS Platform
Analytics and Visualisation Workbench
Resource Orchestrator
New analytics job execution
Request the Secure and Private Space Provisioning
Request the deployment of the services
Execute analytics job
Retrieve analytics results
Scheduled job
Schedule execution of analytics job
Analytics job completed
Initiate the Secure And Private Space
Deploy the services
Finish the Secure and Private Space deployment
One-off job
Stop the Secure and Private Space

Figure 2-20: Secure and Private Space provisioning / stoppage workflow

Another cornerstone on the ICARUS platform operation is the intercommunication of the three main tiers for the execution of various jobs that span across the ICARUS platform's tiers. For this purpose, the master / worker process is utilised, that is following the Master / Worker paradigm as described also in the ICARUS platform architecture. For the realisation of the master / worker process, the Master Controller residing on the Core ICARUS platform, the OnPremise Worker residing on the On Premise Environment and the SecureSpace Worker residing on the Secure and Private Space are utilised. The Master Controller is responsible for providing the instructions from the Core ICARUS platform, as received by the various services of the platform, to either the OnPremise Worker or the SecureSpace Worker depending on the instructions. Figure 2-21 illustrates the interactions between the Master Controller, the OnPremise Worker and the SecureSpace Worker.

In this workflow, the Master Controller, upon receiving the instructions, routes them to either the SecureSpace Worker or the OnPremise Worker depending on the intended recipient of the instructions. Each of the workers, will interpret the instructions received and request for the corresponding job execution from one of the processes that are responsible for the job execution according to the instructions. The responsible process will perform the job execution and will provide the execution results back to the worker. The worker will then, provide the results back the Master Controller and the instructions execution will be completed. For the On Premise Environment, the list of processes includes the data cleansing, the data mapping and the data anonymisation processes that were described in section 2.2, as well as the data encryption and data decryption processes that were described in section 2.7. For the Secure and Private Space, the list of processes includes the job execution and scheduling processes, the encrypted data assets transfers that were described in section 2.4, and the data encryption and data decryption processes that were described in section 2.7.

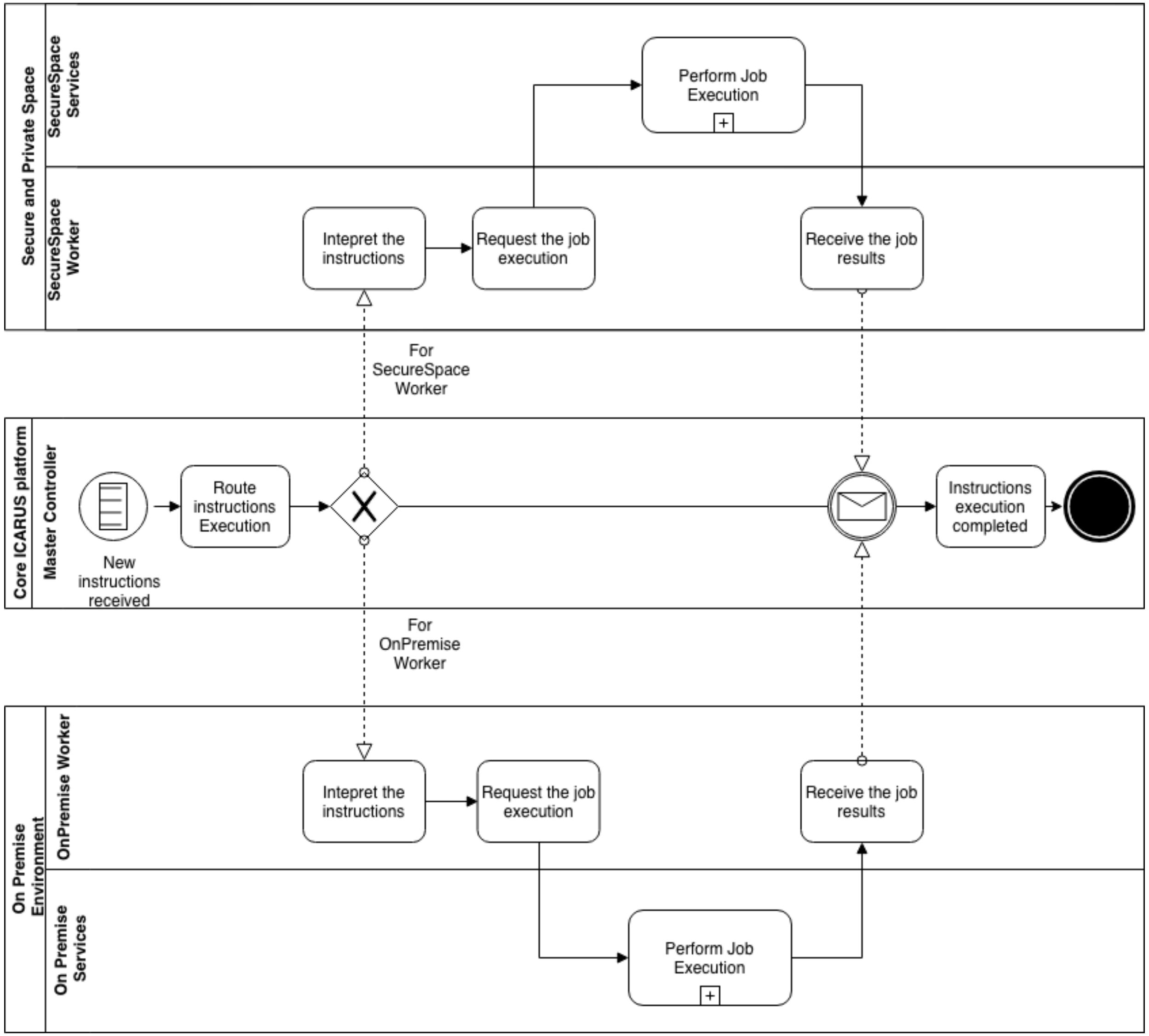

Figure 2-21: Master – Workers intercommunication workflow

2.8.2 Involved services

The presented workflows were further analysed in order to identify the required services that will facilitate the realisation of these workflows. The following table documents the identified services as extracted from these workflows.

Table 2-7: Backend Ancillary services

Number	Title/ Description	Interacting Components	Related Services Group(s)
BAP_1	Secure and Private Space provisioning	Resource Orchestrator, Analytics and Visualisation Workbench	Resource Orchestration
BAP_2	Deploy the Secure and Private Space services	Resource Orchestrator, Analytics and Visualisation Workbench	Resource Orchestration
BAP_3	Secure and Private Space stoppage	Resource Orchestrator, Analytics and Visualisation Workbench	Resource Orchestration
BAP_4	Route and interpret the instructions for the job execution	Master Controller, On Premise Worker, SecureSpace Worker	Master and Worker
BAP_5	Request the job execution according to the instructions and collect the results	Master Controller, On Premise Worker, SecureSpace Worker	Master and Worker
BAP_5	Monitor the job executed and reports its current status to the Master Controller and the user of the platform	Master Controller, On Premise Worker, SecureSpace Worker	Master and Worker

2.9 Data Recommendation

2.9.1 Designed workflows

The data recommendation workflow aims at assisting the stakeholders by providing recommendations for data assets, ensuring that each stakeholder is able to reach data assets that will be more useful and suitable for his/her needs. The scope of the recommendation process is to generate recommendations of data assets that can be explored or utilized by each user during the search and query process. As described in D3.1, the component responsible for the recommendation service is the Recommender. The Recommender is involved in two distinct phases, namely the offline training phase and the execution phase. Specifically, the first phase involves the offline training of the recommendation algorithm that is done periodically, during configured time intervals. The second phase describes the recommendations of data assets that are suggested to each user that explores the ICARUS data asset marketplace.

Figure 2-22 illustrates the offline training process of the recommendation algorithm. During configured time intervals, the Recommender initiates the training process that is executed periodically in order to improve the recommendation model. At first, it retrieves information from the ICARUS Storage component about the users’ history (e.g. searches, views, favourites, purchases etc.), as well as information about the data assets categories (e.g. passengers’ details). Subsequently, it uses this information to retrain the recommendation model so as to provide more accurate predictions.

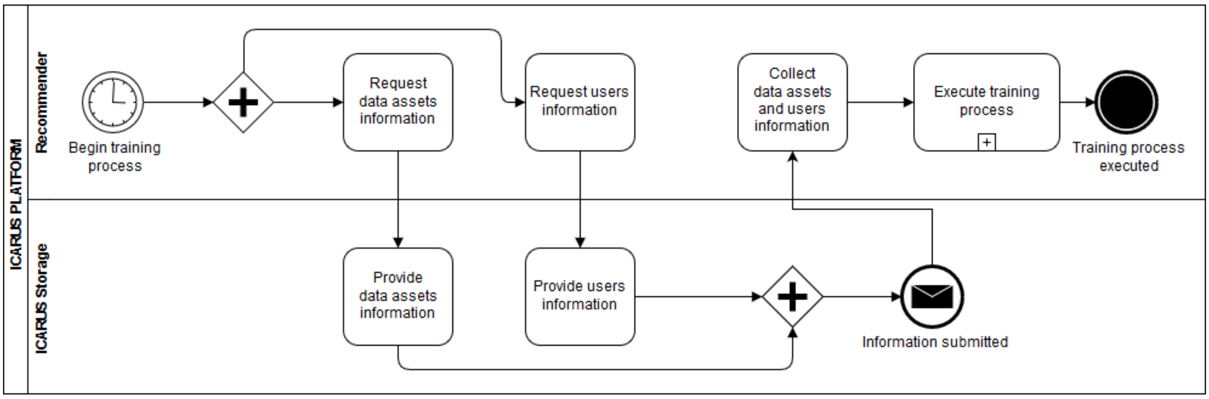
ICARUS PLATFORM
Recommender
ICARUS Storage
Begin training process
Request data assets information
Request users information
Collect data assets and users information
Execute training process
Training process executed
Provide data assets information
Provide users information
Information submitted

Figure 2-22: Data Recommendation – offline training process workflow

Figure 2-23 shows the interactions between the user and the ICARUS platform User Interface for the data asset exploration and how these interactions are processed and translated by the ICARUS platform into internal processes and functionalities. This workflow is triggered when a data consumer uses the search mechanisms of ICARUS, aiming to explore the available data assets by providing a search query. Even though the Query Explorer is responsible to retrieve data assets from the ICARUS Storage that are relevant to the user's search query, at the same time, it initiates the recommendation process. The recommendation process is executed by the Recommender, after receiving the user ID and the data asset IDs, as well as requesting the users' preferences and semantic metadata (derived from the ICARUS Ontology) from the ICARUS Storage component. In this process, the trained recommendation model generates recommendations for data assets that may interest the user. Then, the Query Explorer is responsible to show to the user the recommendations along with the search results that are relevant to the initial search query. The recommendation process is running in parallel with the dataset query process that is described in section 2.5 which for simplicity reasons is not included in the diagram.

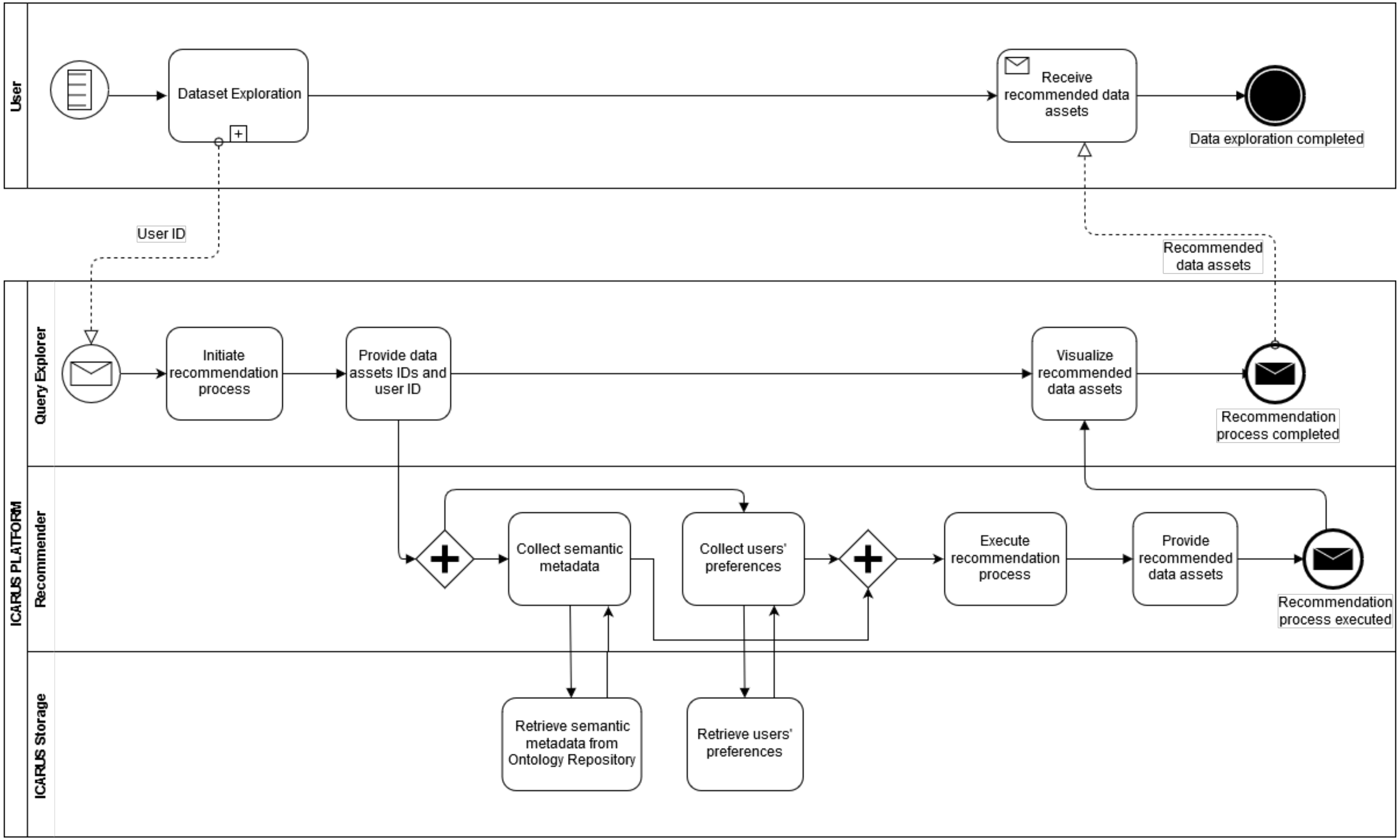
User
Dataset Exploration
Receive recommended data assets
Data exploration completed
User ID
Recommended data assets
ICARUS PLATFORM
Query Explorer
Initiate recommendation process
Provide data assets IDs and user ID
Visualize recommended data assets
Recommendation process completed
Recommender
Collect semantic metadata
Collect users' preferences
Execute recommendation process
Provide recommended data assets
Recommendation process executed
ICARUS Storage
Retrieve semantic metadata from Ontology Repository
Retrieve users' preferences

Figure 2-23: Data Recommendation - execution process workflow

2.9.2 Involved services

In the previous section the interactions between the Recommender and the other components were described. While all these interactions are facilitated with a set of well-defined APIs in order to enable the flow of information between these components, the following services are foreseen:

Table 2-8: Data Recommendation service

Number	Title/ Description	Interacting Components	Related Services Group(s)
DR_1	Collect data assets information	Recommender, ICARUS Storage and Indexing	Data Recommendation
DR_2	Collect users' information	Recommender, ICARUS Storage and Indexing	Data Recommendation
DR_3	Perform the offline recommendation training process	Recommender	Data Recommendation
DR_4	Collect semantic metadata for the data assets	Recommender, ICARUS Storage and Indexing	Data Recommendation
DR_5	Collect user's preferences	Recommender, ICARUS Storage and Indexing	Data Recommendation
DR_6	Execute recommendation process	Recommender	Data Recommendation

2.10 Notifications

2.10.1 Designed workflows

The Notification Service enables the ICARUS platform to inform the end users for different events associated with the users' activities on the platform. As described in D3.1, the component responsible for managing this service is the Notification Manager. This component manages the flow of information about the addition of a data asset or updates on existing data assets, status updates on analytics jobs executed in the users' Secure and Private Space, as well as informing for the activities related to asset brokerage between data consumers and data providers. The notifications can reach end-users via web notifications and emails.

Figure 2-24 shows the interactions between the user and the various components of the ICARUS Platform when a data provider completes the upload of a new data asset. In this workflow, a data provider uploads the new data asset to the platform following the data upload process that is described in section 2.3.1. When the upload process is complete, the Data Handler raises an event for the addition of this data asset. This event is received by the Notification Manager. The latter requests the users' information from the ICARUS Storage to identify interested data consumers based on their category preferences. Subsequently, the Notification Manager stores the newly created notifications in the ICARUS Storage and notifies the data consumers via email.

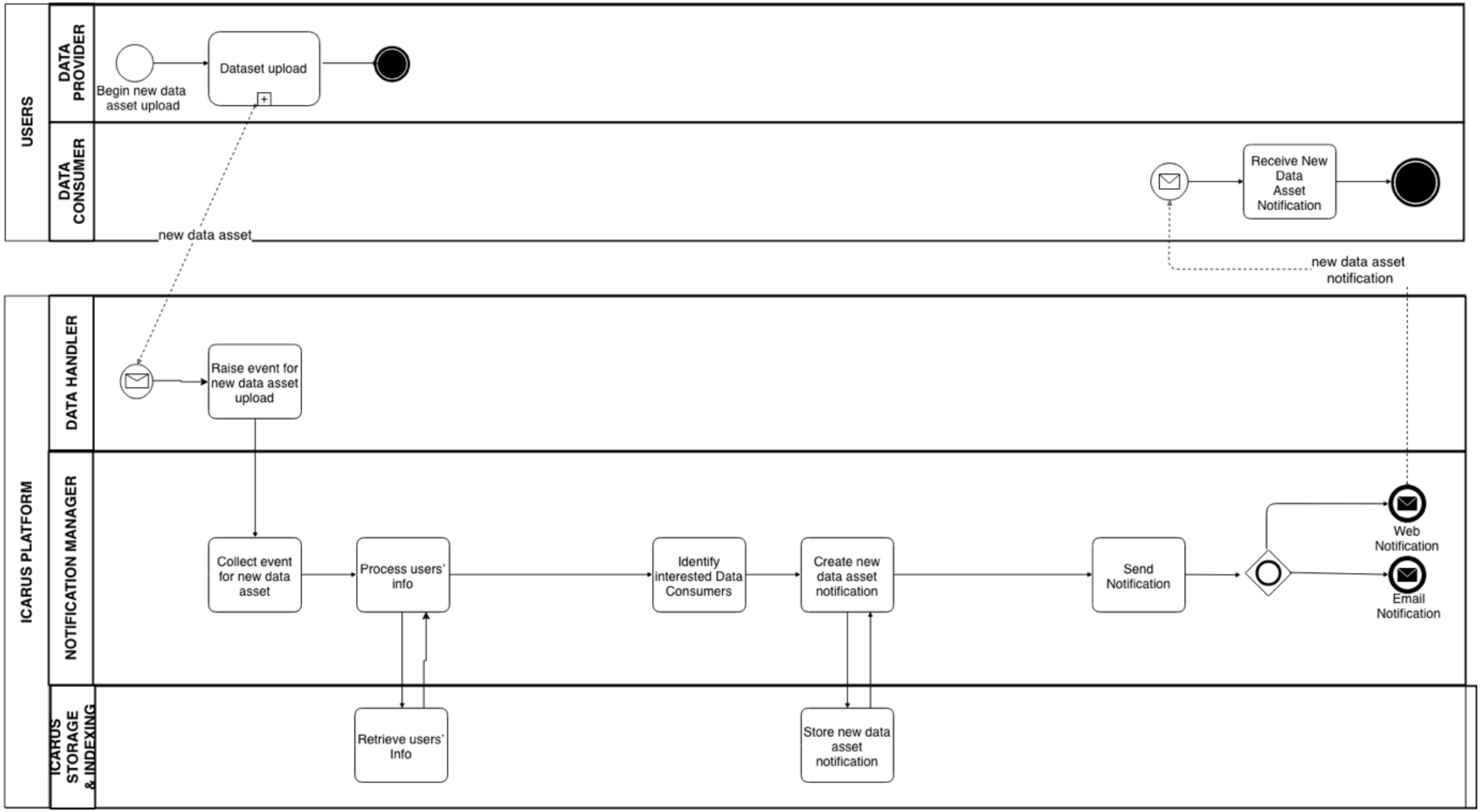
USERS
DATA PROVIDER
DATA CONSUMER
Begin new data asset upload
Dataset upload
new data asset
Receive New Data Asset Notification
new data asset notification
ICARUS PLATFORM
DATA HANDLER
Raise event for new data asset upload
NOTIFICATION MANAGER
Collect event for new data asset
Process users' info
Identify interested Data Consumers
Create new data asset notification
Send Notification
Web Notification
Email Notification
ICARUS STORAGE & INDEXING
Retrieve users' Info
Store new data asset notification

Figure 2-24: Data Notification - data asset addition workflow

Data assets are subject to updates, and data consumers that are entitled to use them should be able to be informed when these updates occur. Figure 2-25 shows this process, which is followed by the Notification Manager to notify data consumers. In the data asset update workflow, the data provider updates a data asset following again the data upload process that is described in section 2.3.1. When the update is completed, the Data Handler raises an event for the data asset update, which is received by the Notification Manager. The latter identifies which users are entitled to use the data asset by interacting with the Data Licence and Agreement Manager and retrieves user information from the ICARUS Storage. Subsequently, the Notification Manager stores these notifications in the ICARUS Storage and notifies the data consumers with a new notification and /or an email.

Moreover, the ICARUS platform offers the ability to execute scheduled analytics jobs on data assets, by performing various algorithms that run for a period of time. Hence, users should be able to be notified for any updates on the status of their scheduled analytics job. In Figure 2-26, the appropriate interactions between components and functionalities of this procedure are illustrated. In this workflow, the Analytics and Visualisation Workbench which is responsible to supervise the analytics jobs that run in the users' secure private space, raises an event for a status update on an analytics job (e.g. completion, failure, etc.). Then, the Notification Manager collects this event and retrieves the contact details of the user that initiated the analytics job from the ICARUS Storage. Finally, the notifications are stored in ICARUS Storage and the users are notified for the status update of their analytics job through the ICARUS platform, and / or email.

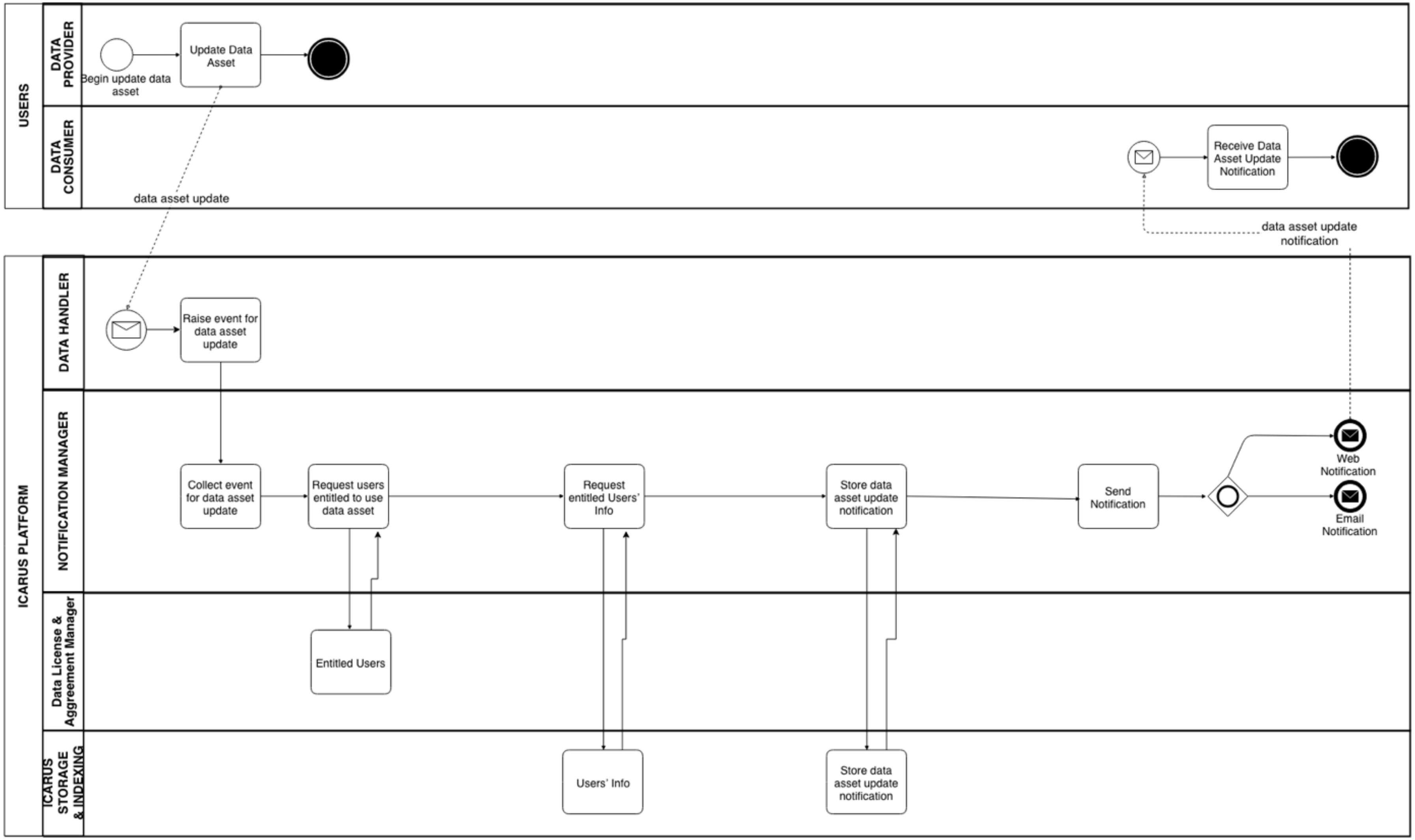
USERS
DATA PROVIDER
Begin update data asset
Update Data Asset
DATA CONSUMER
data asset update
Receive Data Asset Update Notification
data asset update notification
ICARUS PLATFORM
DATA HANDLER
Raise event for data asset update
NOTIFICATION MANAGER
Collect event for data asset update
Request users entitled to use data asset
Request entitled Users' Info
Store data asset update notification
Send Notification
Web Notification
Email Notification
Data License & Aggreement Manager
Entitled Users
ICARUS STORAGE & INDEXING
Users' Info
Store data asset update notification

Figure 2-25: Data Notification - data asset update workflow

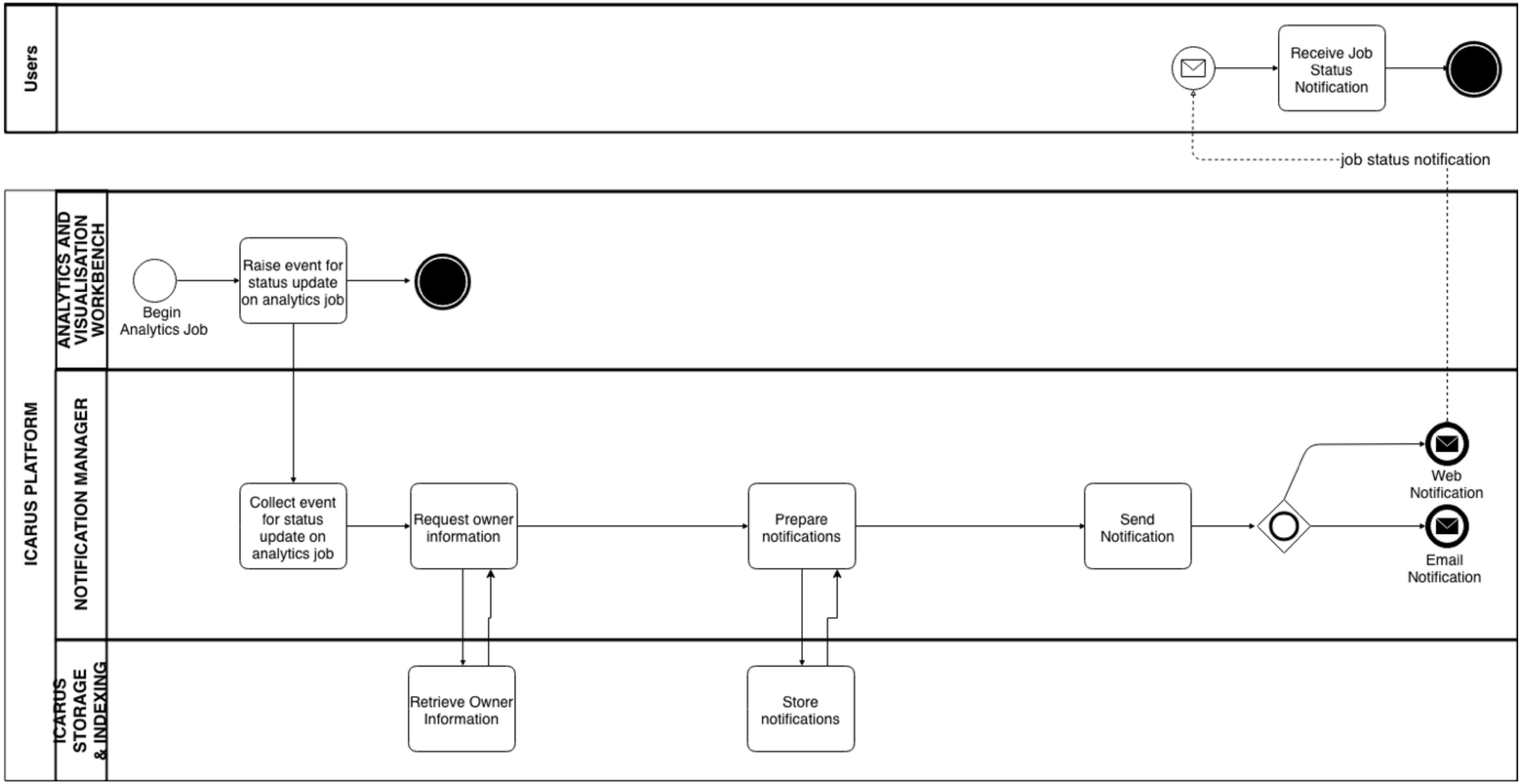
Users
Receive Job Status Notification
job status notification
ICARUS PLATFORM
ANALYTICS AND VISUALISATION WORKBENCH
Begin Analytics Job
Raise event for status update on analytics job
NOTIFICATION MANAGER
Collect event for status update on analytics job
Request owner information
Prepare notifications
Send Notification
Web Notification
Email Notification
ICARUS STORAGE & INDEXING
Retrieve Owner Information
Store notifications

Figure 2-26: Data Notification – status update on analytics job workflow

As the ICARUS platform is a marketplace for data assets, a data consumer can request access to a data asset from its provider. When the request is done, the data provider needs to be notified about the request. The workflow in Figure 2-27 illustrates this procedure, as the Data License and Agreement Manager raises an event for the request. The Notification Manager collects this event and requests the data asset provider's information from the ICARUS Storage. After this information is retrieved, the notifications created, stored in ICARUS Storage and the data provider is notified via email, as well as through the notification panel of the platform.

In Figure 2-28, when the data provider submits his/her decision for the approval or rejection of the request, the responsible component Data License and Agreement Manager raises another event for the provider's decision. Similarly, the Notification Manager collects the event and retrieves from the ICARUS Storage the information of the data consumer that requested the data asset. Depending on the data provider's decision, a notification is created and stored in the ICARUS Storage and the data consumer is notified with the previously mentioned ways.

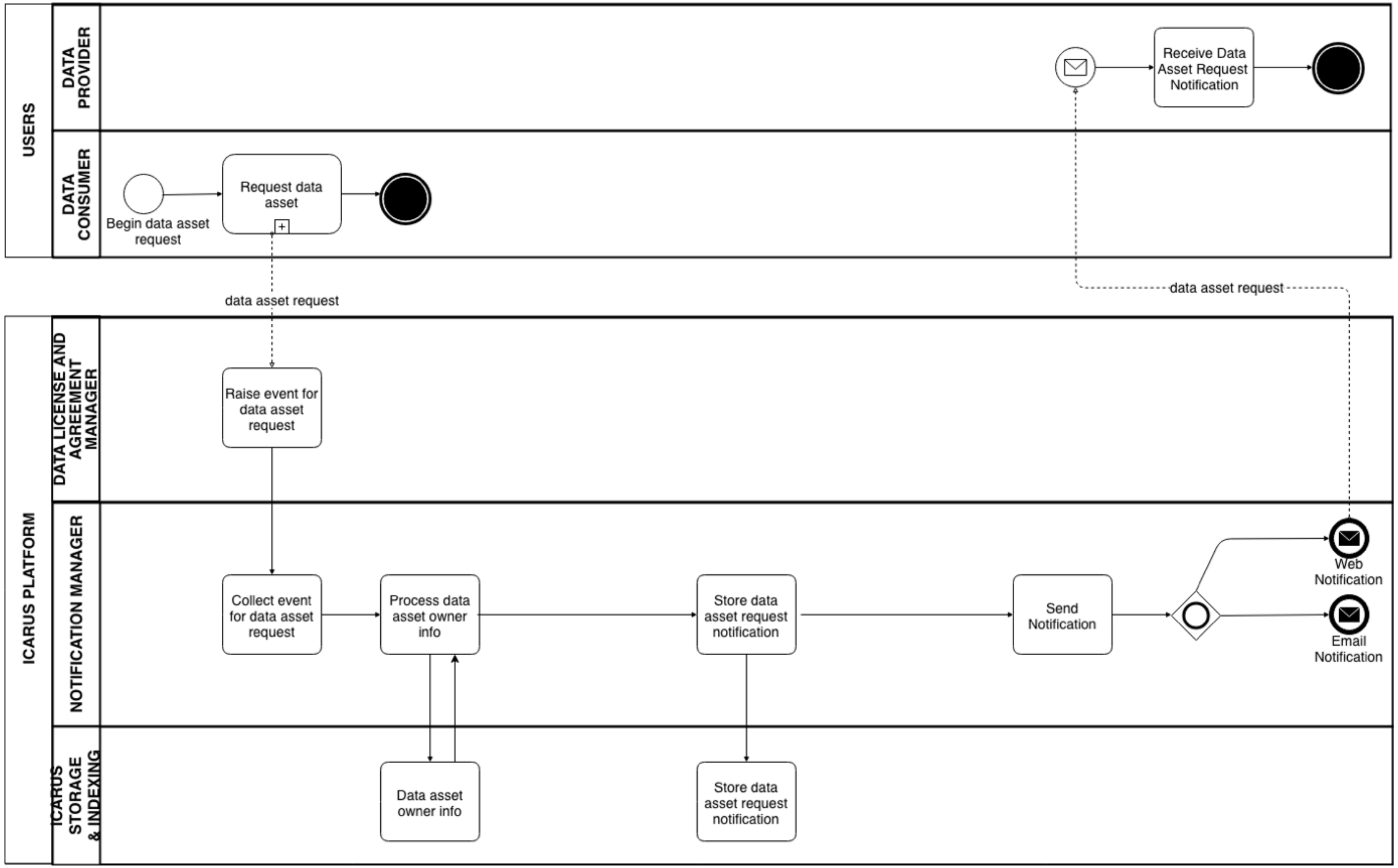
USERS
DATA PROVIDER
Receive Data Asset Request Notification
DATA CONSUMER
Begin data asset request
Request data asset
data asset request
data asset request
ICARUS PLATFORM
DATA LICENSE AND AGREEMENT MANAGER
Raise event for data asset request
NOTIFICATION MANAGER
Collect event for data asset request
Process data asset owner info
Store data asset request notification
Send Notification
Web Notification
Email Notification
ICARUS STORAGE & INDEXING
Data asset owner info
Store data asset request notification

Figure 2-27: Data Notification - data asset request workflow

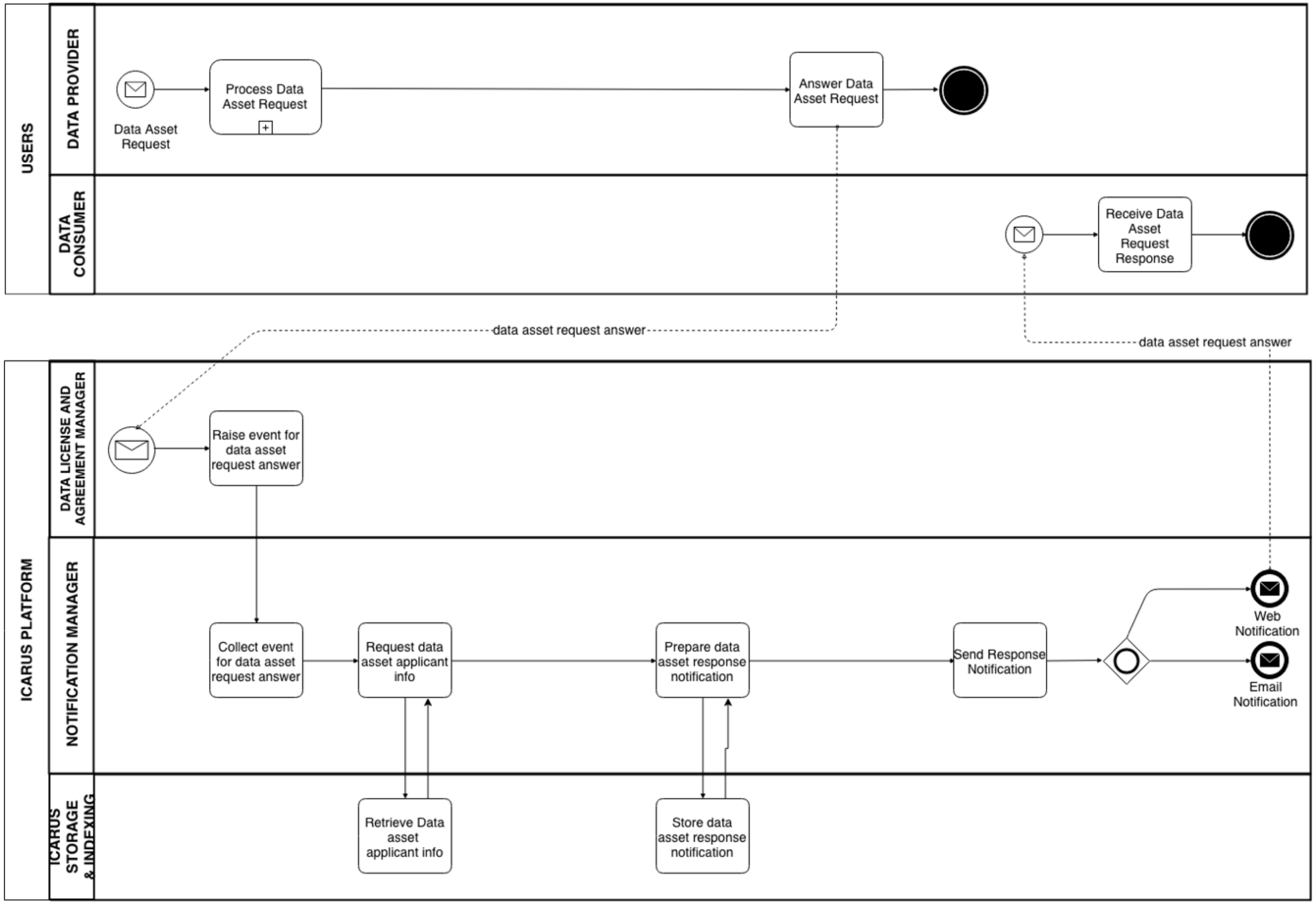
USERS
DATA PROVIDER
Data Asset Request
Process Data Asset Request
Answer Data Asset Request
DATA CONSUMER
Receive Data Asset Request Response
data asset request answer
data asset request answer
ICARUS PLATFORM
DATA LICENSE AND AGREEMENT MANAGER
Raise event for data asset request answer
NOTIFICATION MANAGER
Collect event for data asset request answer
Request data asset applicant info
Prepare data asset response notification
Send Response Notification
Web Notification
Email Notification
ICARUS STORAGE & INDEXING
Retrieve Data asset applicant info
Store data asset response notification

Figure 2-28: Data Notification - data asset response to request workflow

2.10.2 Involved services

The analysis of the presented workflows indicated a list of services that are foreseen for the successful realisation of the workflows, as well as for the implementation of the described functionalities. The list of identified services is presented in the following table.

Table 2-9: Notifications service

Number	Title/ Description	Interacting Components	Related Services Group(s)
NOT_1	Collect data asset check-in events	Notification Manager, Data Handler	Notification
NOT_2	Collect data asset update events	Notification Manager, Data Handler	Notification
NOT_3	Collect status update for analytics job events	Notification Manager, Job Scheduler and Execution Engine	Notification, Data Analytics
NOT_4	Collect data asset request events	Notification Manager, Query Explorer	Notification, Data Exploration
NOT_5	Collect data asset response to request events	Notification Manager, Data License and Agreement Manager	Notification, Data Licensing
NOT_6	Retrieve user's information (user's ID, preferences, data owners, data applicants)	Notification Manager, ICARUS Storage and Indexing	Notification
NOT_7	Retrieve entitled users for a data asset	Notification Manager, Data License and Agreement Manager	Notification, Data Licensing
NOT_8	Identify interested parties for the new event (data asset check-in/update, data asset request/response, status update for analytics jobs)	Notification Manager	Notification
NOT_9	Produce and store new notification	Notification Manager, ICARUS Storage and Indexing	Notification
NOT_10	Raise new notification to the interested parties	Notification Manager	Notification

2.11 Usage Analytics

2.11.1 Designed workflows

The Usage Analytics workflow illustrates the collection, analysis and visualisation of the usage of the various services and assets of the platform, in order to enable the users to extract useful insights and statistics. In the Usage Analytics, the user's behaviour is recorded at various levels, such as the users' history (e.g. searches, purchases, etc.), resource allocation, service utilization (e.g. usage of an algorithm) and so on, towards the aim of providing usage information to both the users and the platform administrator.

In Figure 2-29, the process in which the Usage Analytics gathers the usage information is illustrated. This process is based on events that different components raise with regard to the usage of the platform by the data consumers and providers. The workflow is triggered every time an event is raised from a component. These events are relevant to the data asset purchases that are raised by the Data License and Agreement Manager, usage of computational resources raised by the Resource Orchestrator, exploration of data assets by the Query Explorer, status updates on scheduled analytics jobs by the Analytics and Visualisation Workbench, as well as the user registration and sign in processes by the Policy Manager. Every time there is such event raised by the relevant component, the Usage Analytics receives the event along with the specific information of the event. This information is processed, and aggregated statistics are generated. These statistics are stored in the ICARUS Storage for later use.

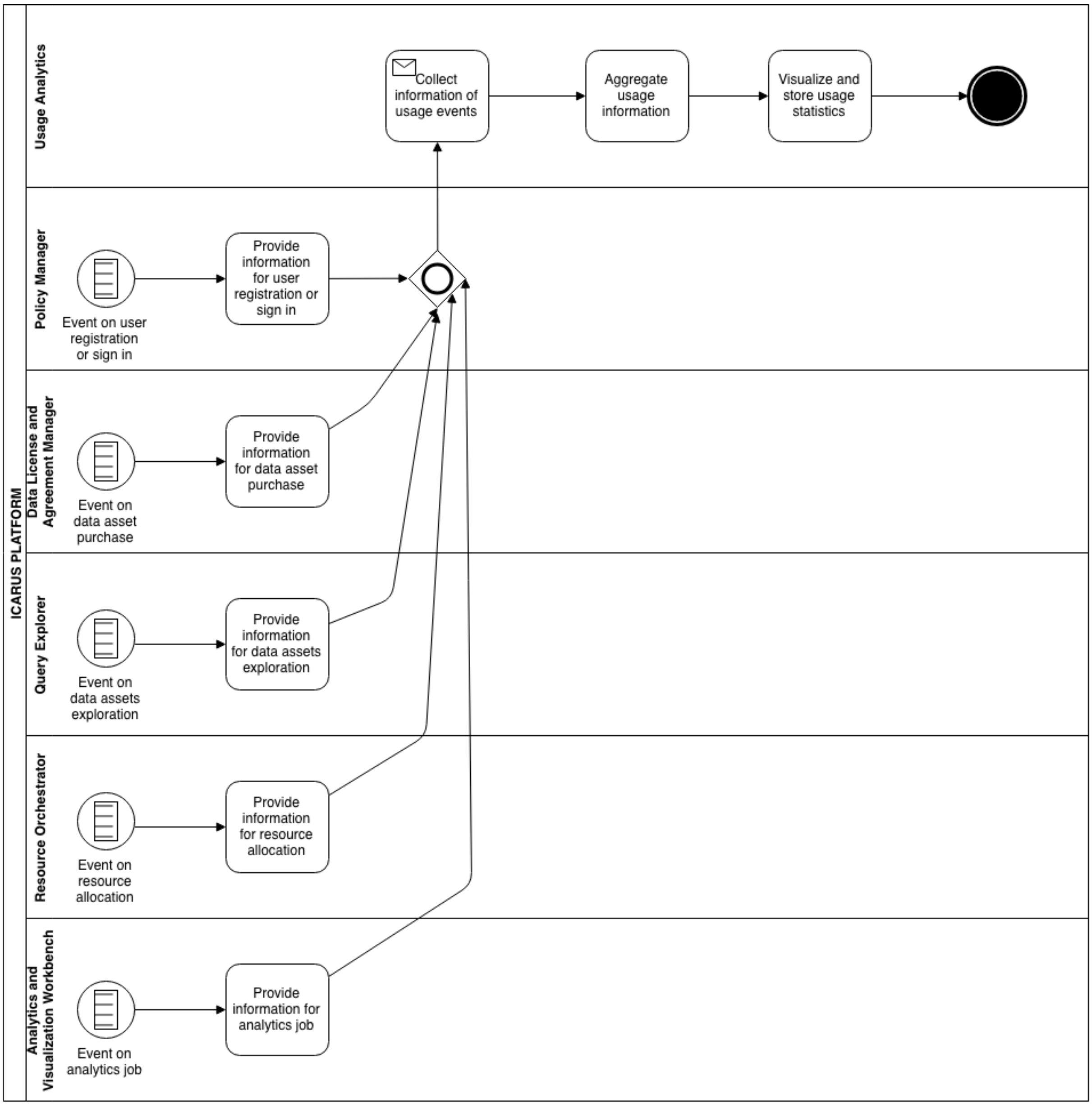
ICARUS PLATFORM
Usage Analytics
Collect information of usage events
Aggregate usage information
Visualize and store usage statistics
Policy Manager
Event on user registration or sign in
Provide information for user registration or sign in
Data License and Agreement Manager
Event on data asset purchase
Provide information for data asset purchase
Query Explorer
Event on data assets exploration
Provide information for data assets exploration
Resource Orchestrator
Event on resource allocation
Provide information for resource allocation
Analytics and Visualization Workbench
Event on analytics job
Provide information for analytics job

Figure 2-29: Data Usage - information collection process workflow

Figure 2-30 portrays the interactions between a user and the ICARUS platform user interface for visualizing statistics of the usage of the platform. A user can be either a data provider, data consumer or even the ICARUS administrator and each one is shown a group of statistics and information that is permitted to view (e.g. a data consumer is not allowed to view the list of users that are entitled to use a specific data asset). This workflow begins with the user accessing the Usage Analytics dashboard that is managed by the platform's UI which initiates the usage analytics process and requests usage statistics from Usage Analytics component. Once the Usage Analytics retrieves the relevant information from the ICARUS Storage, it aggregates the statistics and provides them to the platform's UI for the user to see. Furthermore, the user is able to define a specific time period (e.g. last week / month, etc.) for the statistics that he/she wants to inspect. In this case, the Usage Analytics proceeds to the appropriate aggregations of the statistics in order to provide the user with the appropriate results.

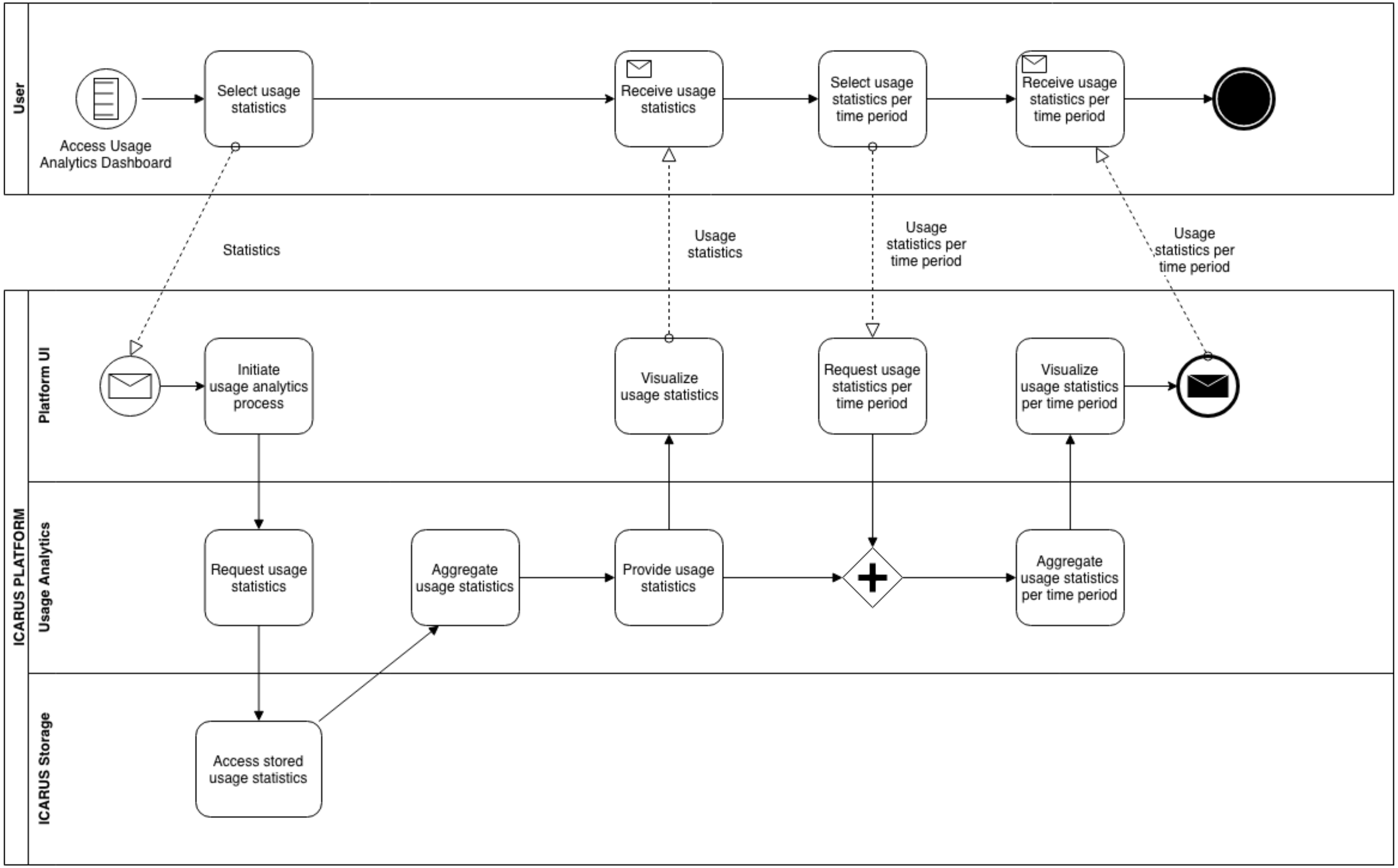
User
Access Usage Analytics Dashboard
Select usage statistics
Receive usage statistics
Select usage statistics per time period
Receive usage statistics per time period
Statistics
Usage statistics
Usage statistics per time period
Usage statistics per time period
ICARUS PLATFORM
Platform UI
Initiate usage analytics process
Visualize usage statistics
Request usage statistics per time period
Visualize usage statistics per time period
Usage Analytics
Request usage statistics
Aggregate usage statistics
Provide usage statistics
Aggregate usage statistics per time period
ICARUS Storage
Access stored usage statistics

Figure 2-30: Data Usage - visualize usage statistics process workflow

2.11.2 Involved services

Based on the workflows presented in the previous sub-section, the following services are foreseen:

Table 2-10: Data Usage services

Number	Title/ Description	Interacting Components	Related Services Group(s)
DU_1	Collect events for usage analytics from the ICARUS components	Usage Analytics, Policy Manager, Data License and Agreement Manager, Query Explorer, Resource Orchestrator, Analytics and Visualisation Workbench	Usage Analytics
DU_2	Collect usage analytics information	Usage Analytics, ICARUS Storage and Indexing	Usage Analytics
DU_3	Aggregate usage analytics information	Usage Analytics	Usage Analytics
DU_4	Generate usage analytics statistics	Usage Analytics	Usage Analytics
DU_5	Present usage analytics statistics (per selected period)	Usage Analytics, UI	Usage Analytics

3 Core Data Service Bundles

The workflows that were presented in the previous section provided useful insights and were further analysed towards the design of the appropriate services that will facilitate the realisation of these workflows. To facilitate the implementation process, the indicative services that were presented in Table 2-1 to Table 2-7 of the previous section were grouped under a list of core services, as depicted in Figure 3-1. Following this approach, each core service is considered as a bundle of underlying services with a distinct scope and set of responsibilities. Besides the individual services, as well as their corresponding design specifications, that were identified, it is obvious that the various services are interacting towards the effective exchange of information through well-defined APIs.

In the following subsections, the design of each of the identified core services is presented, describing all the underlying service included on each bundle and supplemented with the list of technologies and tools that will be exploited in the implementation of these services.

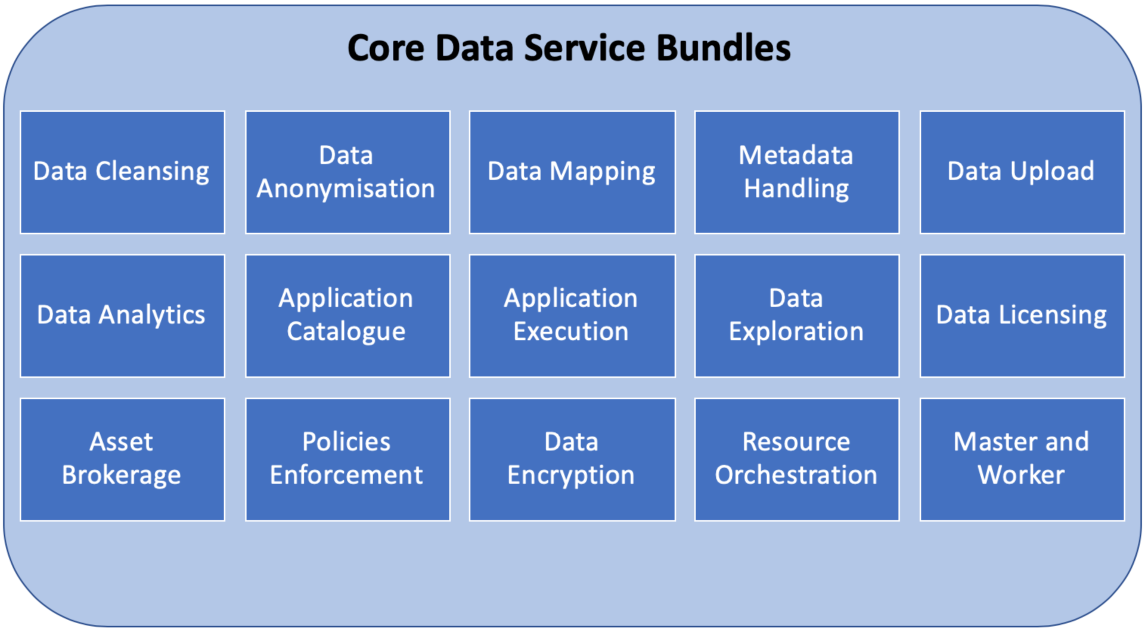

Figure 3-1: ICARUS Core Data Service Bundles

3.1 Data Cleansing service

3.1.1 Service Design

The scope of the Data Cleansing service is to enable the detection and correction of the "dirty" or coarse records of the selected dataset according to the requirements and needs that are set by the data provider. The Data Cleansing service provides the means to the data providers in order to increase the usefulness and usability of their datasets by correcting, cleaning, and/or completing the records that contain errors in terms of accuracy, completeness and correctness. The Data Cleansing service realises and executes the data cleansing process, in accordance with the cleansing method described in ICARUS Deliverable D2.1, that contains:

a) An initial analysis of the dataset in order to obtain the characteristics of the individual elements of the dataset's records.
b) A definition of a cleansing workflow with a set of validation rules for the errors' detection, a set of cleansing rules for the correction (or removal) of the identified errors and a set of data completion rules for the missing value handling.
c) The execution of the designed cleansing workflow.
d) The assessment of the results from the workflow execution.

The Data Cleansing service utilises the functionalities of the Data Cleanser component, as described in the ICARUS Deliverable D3.1, in order to execute the described data cleansing process, which detects, corrects and/or completes the inaccurate, incorrect or incomplete records of the dataset. More specifically, the Data Cleansing service is composed by the following set of services:

a) **A service that performs a preliminary analysis** of the provided dataset. From this analysis, for each individual element of the record, useful information is extracted such as the data type, the value format, the value pattern, and the distinct values that are used later in the process.
b) **A service that performs data validation** based on a set of validation rules. These rules are defined by the data provider providing a level of customisation according to the nature of the dataset and the needs of the data provider. These rules are provided as input to this service and contain a list of constraints per individual element that are utilised for the identification of the conformance errors in the records of the dataset.
c) **A service that performs data cleansing** based on a set of cleansing rules. As with the validation rules, the rules are defined by the data provider according the nature of the datasets and his needs. These cleansing rules are provided as input to the service. Each rule contains the corrective or removal action that is applied for each identified conformance error on an individual element of the record. Thus, these rules are bounded to validation rules since they indicate the action that is performed for the errors identified by the validation rules.
d) **A service that performs data completion** based on a set of data completion rules. Again, these rules are defined and customised by the data provider and are provided as input to the service. In these rules, the corrective actions that are applied in order to perform automatic filling of the missing values are defined against the errors identified during data validation that are related to the required attributes conformance.

e) **A service that enables the assessment of the cleansing process,** by providing a detailed track record of the identified conformance errors, the corrective, removal and data completion actions that were performed during the process execution.

Concrete examples of invoking the Data Cleansing service can be found in the workflows presented in section 2.2.1. As described above, the Data Cleansing service receives two inputs: a) the dataset that the cleansing process will be applied on, and b) the set of validation, cleansing and data completion rules that will be applied on the dataset. The rules are defined through a user-friendly and easy-to-use user interface and are passed to the service for the execution of the cleansing workflow, while the dataset can be indicatively obtained through a local file. The output of the Data Cleansing service that constitutes the clean and complete dataset can be saved as a new dataset or can be provided as input directly to any other service of the ICARUS platform for further processing.

3.1.2 Technologies and Tools

For the implementation of the Data Cleansing service, Python 3.5 and the Flask Python micro web framework[1] will be exploited. Python is considered the dominant language for data manipulation, data analysis, data structure handling and numerical computations as it provides an extended list of powerful libraries and features suitable for these operations. One of the most popular frameworks for the development of RESTful interfaces is Flask due to its simplicity, flexibility and fine-grained control. Besides the Flask framework, two popular Python libraries will be leveraged, the Pandas[2] library that is considered a state-of-art library for data structure handling and the NumPy[3] that is the de-facto choice for numerical computations.

For the design and documentation of the RESTful interface, Swagger[4] will be utilised. Swagger is an open-source software framework incorporating a variety of tools for the effective design, building, documentation and consumption of RESTful interfaces.

3.2 Data Anonymisation Service

3.2.1 Service Design

The Data Anonymisation service aims at ensuring the protection of commercially sensitive, private or personal information that is included on the data provider's dataset, as well as dealing with the various privacy concerns and limitations of the dataset. The Data Anonymisation service is offering the customisable data anonymisation process to the data

[1] Flask, http://flask.pocoo.org/
[2] Pandas, https://pandas.pydata.org/
[3] NumPy, https://www.numpy.org/
[4] Swagger, https://swagger.io/

providers in order to address the problem of data privacy protection. The Data Anonymisation service offers a variety of anonymisation models that can be configured according to the specific data privacy needs of the data provider. Since the data provider has a deep understanding of the privacy concerns and vulnerabilities of his / her own dataset, the Data Anonymisation service is providing the anonymisation process that must be configured accordingly by the data provider.

The Data Anonymisation service implements the data anonymisation method, as described in the ICARUS Deliverable D2.1, that provides the generic data anonymisation process in which various anonymisation models, such as *K-anonymity*, the *L-diversity* and *T-closeness,* are provided to the data provider. From these anonymisation models, the data provider selects the appropriate model and defines the anonymisation technique and anonymisation level that will be applied on the selected dataset based on the content of the dataset and the desired level of data privacy in the form of anonymisation rules. The data provider is able to verify and evaluate the results of the selected anonymisation process based on his/her knowledge and expertise.

The Data Anonymiser service is using the functionalities of the Anonymiser component, as described in the ICARUS Deliverable D3.1, in order to implement the data anonymisation process in which the various anonymisation models are customised and applied according to the instructions of the data provider. In detail, the Data Anonymisation service is composed by the following set of services:

a) **A service that performs the data anonymisation** based on the anonymisation rules. These rules are defined by the data provider taking into account the nature of the dataset, as well as the content of the dataset based on his / her expertise and knowledge. Within the rules, the anonymisation model that will be used for each individual element is defined, along with the corresponding anonymisation techniques and level. These rules are provided as input to the Data Anonymisation service and are utilised for the customisation of the anonymisation process.

b) **A service that enables the verification and assessment of the results,** by presenting the results of the anonymisation process execution and the transformations that took place during the execution in order to be reviewed and verified by the data provider.

Concrete examples of invoking the Data Anonymisation service can be found in the workflows presented in section 2.2.1. The Data Anonymisation service receives two inputs: a) the dataset that the anonymisation process will be applied on, and b) the anonymisation rules as defined by the data provider based on his / her expertise and knowledge. To facilitate the rules definition, an easily usable and straightforward user interface is provided in which the data provider is able to define the rules. The rules are provided as input to the Data Anonymisation service along with the dataset that can be retrieved from a local file. The output of the Data Anonymisation service that is the anonymised dataset can be saved as the dataset to be uploaded in the ICARUS platform or it can be provided as input to the rest of the services of the ICARUS platform.

3.2.2 Technologies and Tools

For the implementation of the Data Anonymisation service, the open source tool ARX will be leveraged as it is very well documented, has great performance and resource efficiency (even for large datasets) and is supported by a very active community. ARX is a powerful anonymisation tool that: (a) implements a wide variety of privacy and risk models in a highly efficient manner, (b) offers an extended list of implemented methods for data transformation, and (c) provides intuitive methods for the assessment of the usefulness of the processed data.

3.3 Data Mapping Service

3.3.1 Service Design

The Data Mapping service ensures that all data stored in the ICARUS storage conform to the ICARUS common aviation data model, which is considered as a prerequisite for easy data integration in general, but also specifically for several functionalities provided by the ICARUS components, including complex data queries formulation, intuitive dataset exploration and advanced data analytics application. The term "Data Mapping service" practically constitutes an abstraction for the set of processes and interactions that are needed to handle the way a mapping is defined, handled (created, stored, retrieved, updated) and executed.

When considered as a whole, the data mapping service bundle has a one-to-one correspondence with the functionalities of the Mapper component, which were described in detail in section 5.2.5 of D2.1 and section 5.3.3 of D3.1. In terms of more concrete services, which is the scope of the current deliverable, it can be further split into:

a) **A mapping suggestion service**, which accepts as input a dataset sample in CSV format and provides as output a mapping of each CSV column to a field of the ICARUS data model. The output constitutes a mapping object and essentially holds a set of mapping instructions in a JSON format.

b) **A mapping editing service**, which allows the user to correct and/or complete the provided mapping suggestion. The service accepts as input a CSV column index and a field of the ICARUS data model and uses this input to update a specific mapping object.

c) **CRUD services for the objects** that contain the mapping instructions for each dataset in the ICARUS storage.

d) **A data model lifecycle management and evolution service** that accepts the ICARUS common aviation data model as input and allows the user to propose changes in order to address specific needs of his/her dataset that are not currently covered by the model. The service requires the intervention of the ICARUS administrator to ensure that the changes proposed are backwards compatible and do not violate the current model (e.g. by proposing a new field that already appears in the model) and provides as output the updated JSON object of the ICARUS common aviation data model.

e) **A mapping execution service** which accepts a JSON object with data mapping instructions and applies it on a dataset.

The first four services (a to d) correspond to the Mapper part that resides in the Core ICARUS Platform, whereas the last service (e) is implemented in the Mapper part that lives inside the On Premise Environment. It should be noted that, as with the other data preparation services (e.g. Cleansing and Anonymisation), each instance of a mapping object belongs to a specific instance of a data preparation job, as also shown in the corresponding BPMN diagrams presented in section 2.2.1.

3.3.2 Technologies and Tools

For the implementation of the Data Mapping service, the Django framework[5] is leveraged. Django is a free and open source high-level Python based web application framework which follows the Model-View-Controller (MVC) architectural pattern and offers several functionalities such as a model system, and a template engine out-of-the-box while being integrated with a large variety of libraries. For the implementation of the RESTful interfaces of the Data Mapping service, the Django REST Framework will be utilised and the data mapping functionalities will be based on the FlexMatcher[6] Python package which handles effectively the matching and mapping of multiple schemas to a single mediated schema.

3.4 Metadata Handling Service

3.4.1 Service Design

ICARUS has designed a broad metadata schema, as documented in D2.1, to cover all aspects of meta-information on a dataset that could be of interest and value to the users either directly or through enabling the provision of more advanced and higher-quality functionalities. In order to manage the creation, update and proper usage of these metadata, a services bundle, called Metadata Handling, is developed. The licensing service that is

[5] Django, https://www.djangoproject.com/

[6] FlexMatcher, https://pypi.org/project/flexmatcher/

presented in Section 3.10 constitutes a special example of metadata handling due to the fact that there is a separate targeted component that undertakes explicitly this type of metadata information, i.e. licensing. Similarly, the data access policies, although they could potentially be perceived as metadata, are also considered as a special case and are discussed separately in Section 3.12. The remaining metadata properties are all handled by the Data Handler component and are addressed through the same underlying services, as follows:

a) **A metadata definition and update service** takes as input a JSON document, where the fields correspond to the metadata attributes supported by ICARUS and their values correspond to: (a) the values defined by the data provider for the specific dataset in cases where the values need to be provided manually, and (b) the system provided values that are automatically generated.

b) **A metadata retrieval service**, which takes as input a specific dataset id and returns its metadata attributes. Filtering on the type of requested metadata is also possible.

Concrete examples of invoking these services can be found in the workflows presented in section 2.3. These services are part of the Data Handler responsibilities. As with all ICARUS services, the data security services are also invoked in the background to ensure that the metadata handling services are not used by unauthorised users or are in any way mishandled.

3.4.2 Technologies and Tools

For the implementation of the Metadata Handling service, the Data Handler component and it's functionalities will be exploited. More specifically, the Data Handler covers a broad spectrum of operations related to data upload, download and transfer and therefore for its implementation and therefore for its implementation numerous technologies and tools are used to provide all required frontend and backend functionalities that will be utilised in the service implementation, namely the Java 1.8, the well-known PostgreSQL and MongoDB storage solutions and the dominant open source JavaScript Framework Vue.js.

3.5 Data Upload Service

3.5.1 Service Design

The Data Upload service includes all processes and interactions related to the act of importing and uploading datasets in the ICARUS platform. From a more technical perspective, this set of services can be split into:

a) **A data preparation handling service** that receives as input a request for the initialisation of a data upload job and in response creates a new data preparation job object and stores it in the ICARUS storage. The data preparation job id is returned as the service response.

b) **A dataset uploading service** that performs the upload of a dataset from On Premise Environment (user's device) to the ICARUS platform.

c) **A dataset importing service** that performs the import of a dataset from an open data source (e.g. a portal) to the ICARUS platform.

d) **A data transfer service** that transfers a dataset from a Secure and Private Space, through its SecureSpace Worker, to the core ICARUS platform storage.

e) **A data preparation status reporting service** that takes as input the id of a data preparation job and returns its status as a response.

f) **A service that performs a preliminary analysis** of the provided dataset. From this analysis, for each individual element of the record, useful information is extracted such as the data type, the value format, the value pattern, and the distinct values that are used later in the process.

Concrete examples of invoking these services can be found in the workflows presented in sections 2.2 and 2.3. The services described above are implemented through the Data Handler and essentially constitute some of the interfaces that the component will offer. As with all ICARUS services, the data security services are also invoked in the background to ensure that the data upload services are not used by unauthorised users or are in any way mishandled.

3.5.2 Technologies and Tools

For the implementation of the Data Upload service, as with the case of the Metadata Handling service, the Data Handler component will be utilised. As described also in section 3.4.2, the backend and frontend functionalities of the Data Handler that will be used are based on Java 1.8, PostgreSQL, MongoDB and the JavaScript Framework Vue.js.

3.6 Data Analytics Service

3.6.1 Service Design

The Data Analytics service undertakes all the aspects related to the processing and presentation of the results of the performed analysis on the data uploaded by the providers to the end user in a meaningful way. During this process, the data assets are moved securely between the architectural layers of the ICARUS platform in order to meet all the ICARUS security and privacy requirements from the beginning of the analytics workflow to the end of it. Behind the scenes of a simple and intuitive user interface that is offered to the users of the ICARUS platform, a complex flow of data occurs throughout several orchestrated sub-components that enable the users to perform accurate analytics to address their needs.

The Data Analytics service is based at its core on the Analytics and Visualisation Workbench component, as described into the deliverable D3.1, and the scope of this service is to provide a user interface that enables the users to manage (i.e. compose, store and run) the ICARUS Applications and to retrieve and visualize processed data on-demand. Since the nature of this component is client-oriented, it will provide a set of orchestration capabilities and will make intensive use of several core modules of the ICARUS platform such as the Resource Orchestrator, the Master Controller and the SecureSpace Worker, the Decryption Manager, the BDA Application Catalogue, the Jobs Scheduler and Execution Engine. The aim of the Data Analytics service is to provide a set of easy and intuitive actions to the users which are internally translated into complex widely constructed operations that occur under the hood.

In detail, the Data Analytics service is composed by the following set of services:

a) **A service that provides a novel interface to the users** in order to build up analytics workflows in the form of ICARUS applications that are composed by a set of selected: (a) datasets that the user owns or has legitimate access (based on a smart contract), (b) data analysis algorithm's and (c) parameters values for the selected algorithm's.

b) **A service that provides the whole lifecycle management of the designed ICARUS applications**, from their storage, to their reuse or update and their sharing via a set of sharing possibilities to the rest of the users of the platform with the appropriate sharing policies. This service is exploiting the capabilities that are offered by the BDA Application Catalogue.

c) **A service that ensures the existence of a Secure and Private Space** that will be utilised for the execution of the designed ICARUS application. This service is responsible for the communication with the Resource Orchestrator in order to guarantee the provisioning of the Secure and Private Space.

d) **A service that enables the execution or scheduled execution of the ICARUS applications,** utilising a list of core modules of the ICARUS platform, such as the Master Controller and the SecureSpace Worker, the Decryption Manager, the Jobs Scheduler and Execution Engine, in order to execute and monitor the execution status. This service is orchestrating the various services offered by these modules towards the successful execution of the applications and the secure storage of the derivate results in an encrypted manner in the ICARUS Storage.

e) **A service that provides the visualisation capabilities of the ICARUS platform** through a simple and easy to use graphical interface. Following the decryption process that is incorporated in the ICARUS platform and is described in deliverable D2.1, the results are decrypted and the user is provided with a variety of visualisation types in order to explore them in a user friendly way. The list of available visualisation types spans from various chart types such as bar, line and pie charts to more advanced types such as maps, heatmaps and box plots and will be continuously updated during the lifetime of the project depending on the ICARUS stakeholders' needs.

f) **A service that enables the export of the produced results** from the ICARUS applications' execution. This service facilitates the download of the derivative data in the user's local environment or the sharing of this data.

Concrete examples of invoking these services can be found in the workflows presented in section 2.4. Through these services, the Data Analytics service incorporates all the required functionalities with regard to data analysis execution and the visualisation of the results of this analysis. All the actions that are performed from the users are internally translated into service executions that are enabled through a comprehensive list of well-defined interfaces.

3.6.2 Technologies and Tools

For the implementation of the Data Analytics service, a variety of the technologies and tools will be exploited. As the nature of the service includes both a graphical user interface and a set of backend functionalities, different technologies are utilised for each purpose adopting the backend-for-frontend design pattern. The graphical user interface is mainly based on TypeScript, JavaScript, CSS and HTML5, while for the backend functionalities of the service Node.js[7] is used. Node.js is offering the mechanism for the implementation of frequent I/O bound operations, while also offering integration capabilities for a web client implementation. For the internal persistence, the PostgreSQL offered by ICARUS Storage component will be exploited.

3.7 Application Catalogue Service

3.7.1 Service Design

The Application Catalogue Service is providing the repository where all ICARUS applications are stored, accessed and maintained. More specifically, the analytics workflows that are designed with the help of the Data Analytics service from the users of the platform are securely stored within this repository in order to be reused, updated or shared between the users according to the license defined by the owner of the application (in collaboration with the Data Licensing service). The Application Catalogue Service is based on the BDA Application Catalogue component of the ICARUS platform, as described in deliverable D3.1, and exposes a group of CRUD restful APIs to create, get, update and delete ICARUS Applications. The APIs are in line with the specifications of the ICARUS Application's configuration in order to

[7] NodeJS, https://nodejs.org/en/

guarantee the highest consistency level and maintain all the required information inside this repository.

In terms of more concrete services, the Application Catalogue service can be further split into the following services:

a) **A service that enables the storage of the designed ICARUS application** that is designed within the context of the Data Analytics service.
b) **A service that enables the access and retrieval of the stored ICARUS application** in order to be (re)deployed and (re) executed through the Data Analytics service.

Concrete examples of invoking these services can be found in the workflows presented in section 2.4. Both services are designed and implemented following all the ICARUS platform's security and privacy requirements. Hence, the storage and access to these ICARUS applications is controlled with the help of the Policies Enforcement service, as described in section 2.7.

3.7.2 Technologies and Tools

For the implementation of the Application Catalogue service, Java 1.8 and the Spring Boot framework will be leveraged. The service exploits a custom data model that incorporates all the required configuration and metadata information of the ICARUS application that are internally utilised in the storage of the designed application within the repository, as well as the execution of the designed application within the context of the Jobs Scheduler and Execution Engine. The metadata are aligned to the ICARUS metadata schema and are stored in the PostgreSQL instance offered by ICARUS Storage component.

3.8 Application Execution Service

3.8.1 Service Design

The Application Execution service is providing the ICARUS application execution capabilities of the ICARUS platform within the context of the Secure and Private Space of the ICARUS platform. The Application Execution service undertakes the responsibility of initiating and monitoring the execution of the ICARUS application as instructed by the Data Analytics service with the use of the SecureSpace Worker. The service exploits the functionalities of the Job Scheduler and Execution Engine component, as described in deliverable D3.1. More specifically, the service receives the request through the exposed APIs for scheduling the execution of the applications, immediately or in a deferred manner. Through the scheduling and resource management capabilities of the service, the instructions are translated into specific jobs that are allocated to the underlying Execution Cluster in order to perform the execution of the designed analytics workflow. Hence, the service undertakes the actual execution utilising the nodes of the cluster computing framework of the platform, ensuring the effective and efficient resource allocation between the nodes for the optimal performance. Furthermore, the service is responsible for the monitoring and reporting of the execution status to the Data Analytics service. The Application Execution service is a bundle of services that were extracted from the workflows presented in section 2.4. In detail, the Application Execution service is composed by the following set of services:

- **A job scheduling service** that offers the scheduling or un-scheduling of the execution of an application, as well as access to the list of scheduled executions with the corresponding exposed APIs.
- **A job execution service** that performs the execution of an application. This service handles the actual execution of the application that was either scheduled or requested to be executed immediately. Additionally, this service handles the resource management and the interaction with the Execution Cluster. The service offers a set of well-defined APIs for the job execution or cancellation.
- **An data handling service** that performs the appropriate handling of the data assets that will be used in the job and the job execution results. Particularly, this service handles the interactions with the Data Encryption and Decryption service, as described in section 2.4, in order to decrypt the data assets that will be used in the job execution, as well as to encrypt the produced results and store them to the ICARUS Storage with the help of the SecureSpace Worker.

3.8.2 Technologies and Tools

For the implementation of the Application Execution service, Java 1.8 and a customised version of the of Spring Cloud Dataflow Server[8], which provides an effective way to execute the designed workflows of ICARUS Applications, will be exploited. In detail, following the micro-service approach, the service will allow running algorithms via spring boot applications as well as Spark applications using an intermediate microservice involved in a pipeline. This intermediate microservice implements a Spark client that interacts with a Spark cluster, that is utilised as the cluster computing framework of the ICARUS platform. Within the context of the service, the PostgreSQL instance offered by the ICARUS Storage component is used for persistence reasons as well.

[8] Spring Cloud Dataflow Server, https://cloud.spring.io/spring-cloud-dataflow/

3.9 Data Exploration Service

3.9.1 Service Design

This service includes all processes and interactions related to performing a search over the available ICARUS datasets and retrieving and showing the results to the user. The data recommendation services as explained in section 2.9 are part of the broader dataset search process, but are offered through a distinct service which will be presented in section 4.1. From a more technical perspective, the set of services that altogether constitute the data exploration service can be split into:

- **A query creation service** that accepts key-value pairs that correspond to query configuration parameters and transforms them to a Solr query based on which a new ICARUS query object is created. The same service will invoke another service to store the created query in the ICARUS storage.
- **A query execution service** that receives a query object, executes the corresponding search and returns the query results as a response.
- **A query access policy service** that retrieves the access policy clause of a specific query performed by a specific user and updates the query object to include the extra clause.
- **A query results processing service** that processes the results of a query to transform them to a format appropriate to be shown in the UI.
- **A query samples service** that receives as input a dataset id and returns the data sample provided for it by the data provider.

Concrete examples of invoking these services can be found in the workflows presented in 2.5. These services correspond to functionalities provided mainly by the Query Explorer component. As with all ICARUS services, the data security services are also invoked in the background to ensure that the data exploration services are not used by unauthorised users or are in any way mishandled.

3.9.2 Technologies and Tools

Since the described services are provided mainly through the Query Explorer, the underlying technologies are the ones used for the implementation of the component. Thus, Django framework will be leveraged, as well as the Django REST framework for the RESTful interfaces' implementation. For the frontend part of the service, the powerful JavaScript Framework Vue.js will be utilised which suitable for building user interfaces and single-page applications with an extensive set of features and capabilities. For the query processing, the indexing service, one of the two core backbone services of the platform that is offered by the ICARUS

Storage and Indexing component, will be utilised. The indexing service is empowered by the Solr[9] open source enterprise search platform.

3.10 Data Licensing service

3.10.1 Service Design

The Data Licensing service includes all processes and interactions related to defining license attributes for a specific dataset. These attributes include all data attributes and qualities that affect, or are in any way relevant to, the ways in which data assets can be shared / traded and handled subsequently to their acquisition. This involves licenses, IPR, characterisation of sensitivity and privacy risk levels, but also more generic metadata regarding data content and structure. The Data Licensing service handles all properties and actions related to limitations and potential risks and benefits of a data asset being shared through the ICARUS system. The exact attributes and the significance of this process have been documented in detail in D2.2. From a more technical perspective, this set of services can be split into:

- **A data license definition and update service** that takes as input a JSON document, where fields correspond to the license attributes supported by ICARUS and their values correspond to the values defined by the data provider for the specific dataset.
- **A data license retrieval service**, which takes as input a specific dataset id and returns its license attributes.

Concrete examples of invoking these services can be found in the workflows presented in section 2.3. These services correspond to the functionalities of the Data License and Agreement Manager component and therefore essentially constitute the interfaces that the component will offer. As with all ICARUS services, the data security services are also invoked in the background to ensure that the data licensing services are not used by unauthorised users or are in any way mishandled.

3.10.2 Technologies and Tools

For the implementation of the Data Licensing, the Data License and Agreement Manager component and the Wallet Manager component (which is complementary to the first and resides in the On Premise Environment) will be exploited. Hence, the underlying technologies are the ones used to implement these components. For the Data License and Agreement Manager specifically these are the Django Framework and the Django REST Framework and the JavaScript Framework Vue.js for the frontend part of the service. The Wallet Manager is based on the open source programming language Go and on geth, i.e. the command line

[9] Solr, https://lucene.apache.org/solr/

interface for running a full Ethereum node implemented in Go. Obviously, these services are dependent on the underlying blockchain that is responsible for the smart data contracts and therefore the Ethereum[10] blockchain can be perceived as part of the broader data brokerage service offering which involves the application and usage of the Data Licensing services discussed here.

3.11 Asset Brokerage service

3.11.1 Service Design

The Asset Brokerage service includes all processes and interactions that take place from the moment a user issues a request to purchase an asset owned by another user of the platform, until the request has been rejected, cancelled (i.e. rejected by the requesting user) or granted and its payment is completed and validated. It should be noted again that for the current deliverable, the term asset is limited to datasets and it will expand to subsequent versions of the WP2 and WP3 deliverables to embrace the ICARUS applications, as well. From a more technical perspective, this set of services can be split into:

- **A request handling service** that receives as input specific parameters within a dataset id (and the authenticated user) and creates and saves in the storage a request to purchase the dataset (as a whole or a specific extract).
- **A request cancelling service** that receives as input a purchase request id and changes its status to cancelled.
- **A smart contract creation service** that receives as input contract terms and creates a smart contract in the blockchain.
- **A smart contract status update service** that receives as input a smart contract id and a status and sets the status of the specified smart contract to the specified status. Accepted status values are: draft, negotiating, accepted, paid, rejected.
- **A smart contract update service** that takes as input a smart contract id and contract terms and updates the smart contract accordingly.
- **A smart contract retrieval service** that takes as input a smart contract id and retrieves it from the blockchain (i.e. terms and status).
- **A smart contract status retrieval service** that takes as input a smart contract id and retrieves its status.
- **A smart contract notification service** that creates notifications when the status of a smart contract in the blockchain is changed.

[10] Ethereum, https://www.ethereum.org/

- **A wallet creation service** invoked the first time a user sets up his On Premise Environment to initiate the user's smart wallet.
- **A wallet management service** to retrieve and update information for a user's smart wallet.

Concrete examples of invoking these services can be found in the workflows presented in section 2.6. The services described above are implemented mainly through the Data License and Agreement Manager component.

3.11.2 Technologies and Tools

For the implementation of the Asset Brokerage service, as with the case of the Data Licensing service, the Data License and Agreement Manager and the Wallet Manager components will be exploited. Hence, for the implementation of the service, the technologies utilised for the implementation of these components are exploited, namely the Django Framework, the Django REST Framework, JavaScript Framework Vue.js and the underlying blockchain that is based on the Ethereum blockchain.

3.12 Policies Enforcement service

3.12.1 Service Design

The Policies Enforcement service is enabling the protection of any critical or valuable asset (data asset or ICARUS application) available in the ICARUS platform by providing the reliable access control mechanism that ensures the selective restriction of access to these assets. To meet its goals, the Policies Enforcement service provides the means to prevent unauthorized disclosure to any private asset in order to safeguard their confidentiality and any intentional or accidental unauthorised change to these assets, protecting their integrity.

To support the reliable access control mechanism, the Policies Enforcement service offers the user management process. Within this process the users of the platform are grouped under organisations entities and each organisation contains a number of users (member) sharing the same rights across the organisation besides the invitation of new members that is assigned to the organisation manager. The registration process is a two-step process. At first, the organisation manager registers the organisation to the platform and upon the administrator's approval he/she is able to invite the members to sign up on the platform. As a second step, the members sign-up using the invitation link and the invitation token provided by the service. Upon successful sign-up, the members are able to access the ICARUS platform and all of its services following the login procedure provided by the platform. Each login request is validated by the Policies Enforcement service.

The Policies Enforcement service is also providing the advanced and dynamic access control mechanism that regulates the access to all private assets with the use of Attribute-Based Access Control (ABAC) policies that are formulated using the XACML version 3.0 standard. The Policies Enforcement service is effectively managing the whole policy lifecycle with the proper methods for the declaratively and deterministically authorisation policy specification and the dynamic policy enforcement in the access control decision.

The Policies Enforcement service provides the access control mechanism that is aligned with the characteristics of the data access control method described in the ICARUS Deliverable D2.1, that defines the data access control policy lifecycle which contains:

- The definition of the data access policies associated with an asset.
- The secure storage of these data access policies.
- The enforcement of these data access policies during the access control evaluation.
- The reuse of these data access policies in different assets.
- The evolution of these data access policies with the required updates.
- The disposal of these data access policies if needed.

As defined in the data access control method in D2.1, each policy contains a set of rules. Each rule contains: (a) the subject (requestor) that issues access requests to perform operations on an object (asset), (b) the authorisation level of the subject (grant or deny), (c) the operation that is performed on the object, (d) a context expression formulated by a combination of a number of attributes of the subject, object and the environment of execution, and (e) the object to which the access is controlled.

The Policies Enforcement service is using the functionalities of the Policy Manager component, as described in ICARUS Deliverable D3.1, in order to provide the authorisation engine that implements the access control mechanisms within the ICARUS platform. Besides the authorisation engine, the Policies Enforcements service provides the suitable methods for the definition, modification and disposal of the XACML policies, as well as for their organisation in policy sets, which act as containers that can hold other policies or policy sets. The different access control decisions are yielded based on evaluation and logical reasoning of the defined access control policies for the specific asset.

In detail, the Policies Enforcement service is composed by the following set of services:

a) **A service that provides the complete user management functionalities** of the platform with the implementation of the organisation registration and users invitation process, as well as the login mechanism of the platform.
b) **A service that performs the whole policy lifecycle management** with a set of methods that enable the creation, update and deletion of the XACML-based access control policies,

as well as their reuse and organisation in policy sets. Furthermore, the service stores and maintains the defined access control policies in the dedicated storage in order to be easily accessible and constantly (re)deployed in the authorisation engine.

c) **A service that provides the access control mechanism** that is based on the ABAC model and the XACML standard while acting as the authorisation engine of the platform. The defined policies are instantly deployed in this authorisation engine upon definition or update.

d) **A service that controls and regulates the access of any asset** by evaluating all the relevant to the asset access control policies based on an expert system that is utilised for the reasoning business logic and the attribute expansion incorporated in the rules included in the policies. The service intercepts all access requests for any asset in order to form an access control decision that will either grant or deny the access to the request asset.

The operation of the Policies Enforcement service is dependent on the existence of the access control policies. These policies are the input for the authorisation engine, which is the core part of the service. The Policies Enforcement service receives as input the request for access to any asset of the ICARUS platform through the provided APIs and provides as output the access control decision that is yielded from the authorisation engine that regulates the access control over the assets.

3.12.2 Technologies and Tools

For the implementation of the Policies Enforcement service, a combination of technologies and tools will be exploited in order to deliver the described functionalities. The basis of the Policies Enforcement service will be developed using Java 1.8 and the Spring Boot framework. Specifically, the provided functionalities and features of the Spring Boot framework will facilitate the effective and efficient implementation of the complete policy lifecycle management, as well as the control and regulation of the access to any valuable asset. Furthermore, the de-facto solution for dependency management and automation of the building process, Apache Maven[11] will be leveraged as it offers simple and easy-to-use software project management functionalities suitable for dependency description and automated building of any Java project. For the implementation of the authorisation engine that will be used as an access control mechanism, the Drools[12] expert system will be leveraged. Drools is the most widely used expert system offering a large variety of features, with a rule engine written in Java, supporting forward and backward chaining inference.

[11] Apache Maven, https://maven.apache.org/

[12] Drools, https://www.drools.org/

Drools provides an extended implementation of the Rete matching algorithm suitable for the needs of the Policies Enforcement service.

3.13 Data Encryption and Decryption Service

3.13.1 Services Design

The scope of the Data Encryption and Decryption service is twofold: (a) to provide the encryption mechanism for the data assets as part of the data preparation prior to being uploaded to the Core ICARUS platform (from the On Premise Environment or the Secure and Private Space), and (b) to provide the decryption mechanism of the encrypted ciphertext which is executed when a data consumer acquires legitimate access (based on a contract) to the unencrypted data asset either when downloading a data asset locally on the On Premise Environment or when a data asset is going to be utilised in the Secure and Private Space for data analysis.

The data encryption method, as described in ICARUS Deliverable D2.1, is following a dual encryption approach: (a) symmetric key encryption is utilised for the security and integrity of the data assets and (b) secure SSL handshakes are utilised in order to establish a secure connection between the data provider and the data consumer for the exchange of the decryption symmetric key.

The Data Encryption and Decryption service is using the functionalities of the Encryption Manager, the Decryption Manager and the Key Pair Administrator components of the ICARUS platform, as described in ICARUS Deliverable D3.1, in order to provide the implementation of the described data encryption method. More specifically, the Data Encryption and Decryption service is composed by the following set of services:

a) **A service that performs the column-based encryption** on the selected from the data provider columns using a locally generated symmetric key. The encryption process takes place in the On Premise Environment and the produced ciphertext is provided for upload in the Core ICARUS platform.
b) **A service that establishes the SSL-enabled connection between the data provider and the data consumer** that enables the sharing of encrypted data assets. The connection is performed with two-side authorisation and identity verification between the two involved parties for the proper SSL handshake.
c) **A service that generates and transmits the decryption symmetric key over the secure connection** upon a granted access request in order to perform that decryption of the ciphertext without jeopardising the confidentiality and integrity of the data provider's data asset.

d) **A service that performs the decryption of the ciphertext** with the suitable decryption mechanism and the securely transmitted decryption symmetric key.
e) **A service that handles the revocation process of the decryption symmetric** key upon the needs of the platform.

For the encryption of the data assets, the Data Encryption and Decryption service receives as input the list of columns that will be encrypted as selected by the data provider during the data preparation process design, as described in section 2.2. The output of the service is the ciphertext that can be uploaded in the Core ICARUS platform. For the decryption of data assets the Data Encryption and Decryption service receives as input the request for decryption from the data consumer upon establishing the secure connection with the data provider, it decrypts the ciphertext, provided that the data consumer is eligible to access the requested data asset based on an active data asset contract.

3.13.2 Technologies and Tools

For the implementation of the Data Encryption and Decryption service, again Java 1.8 and the Spring Boot framework will be utilised. In addition to this, Bouncy Castle for Java[13] will be leveraged. Bouncy Castle is Java library offering a collection of open source lightweight cryptography APIs that complement the default Java Cryptographic Extension (JCE) with extended cryptography functionalities which are suitable for the implementation needs of the Data Encryption and Decryption service.

3.14 Resource Orchestration service

3.14.1 Services Design

The scope of the Resource Orchestration service is to facilitate the provisioning and management of the Secure and Private Space for each user of the platform in the form of dedicated virtual machines in order to perform the data analysis in a secure and isolated environment. The Resource Orchestration service provides the means to connect to the underlying virtualised infrastructure in order to monitor and manage the available resources, to provision and stop a set of virtual machines, as well as to deploy and manage the applications and services or applications running on the virtual machines.

The Resource Orchestration service is offering the transparent, easy and secure deployment of “sandboxed” environments with a variety of techniques and technologies tailored to the needs of the ICARUS platform. Based on the ICARUS platform architecture, the Resource

[13] Bouncy Castle, https://www.bouncycastle.org/

Orchestration service also undertakes the responsibility of the dynamic deployment of the services running on the Secure and Private Space.

The Resource Orchestration service is using the functionalities of the Resource Orchestrator component, as described in ICARUS Deliverable D3.1, in order to implement the resource orchestration process that provides the provisioning and deployment of the required set of dedicated virtual machines that are composing the Secure and Private Space. The services and applications that will be deployed on the virtual machines will be in the form of containerised services or applications using the Docker container engine. In detail, the Resource Orchestration service is composed by a set of services that can be split into:

a) **A service that establishes the connection to the underlying virtualised infrastructure** in order to perform the continuous monitoring and management of the available resources. The administrator of the platform is providing the appropriate connection details to the service to enable the connection with the virtualised infrastructure.

b) **A service that performs complete management and maintenance of virtual machines running on the virtualised infrastructure.** The Resource Orchestration service is managing and maintaining the list of virtual machines, both the instance types and the running instances. The Resource Orchestration service is able to create, update or delete instance types, while also being able to create, start and stop any running instances.

c) **A service that performs advanced resource management** over the virtual machine running instance. Through the continuous monitoring of the available resources and the allocated resources, the Resource Orchestration service is capable of performing dynamic resource allocation or de-allocation based on the utilised resources of any running instance**.**

d) **A service that performs dynamic deployment of the containerised services or applications** on the provisioned virtual machines. The Resource Orchestration service is managing the complete lifecycle of the deployment of the configured services or applications by orchestrating their deployment, start, stop or deletion.

e) **Enables the continuous monitoring and management of the services or applications** running on the provisioned virtual machines. The Resource Orchestration service is monitoring the execution status or availability of the deployed services or applications by performing periodic health check and service discovery operations or remote command execution according to the configuration of the services or applications.

The Resource Orchestration service requires the appropriate configuration in order to be able to connect to the virtualised infrastructure and operate according to the described functionalities. Apart from this, the Resource Orchestration receives as input the request for on-demand provisioning or stopping the Secure and Private Space for a user according to the needs of the ICARUS platform. The output of the service is the acknowledgement of the request execution, as well as the corresponding details for the provisioned Secure and Private Space in order to be used in the data analysis execution.

3.14.2 Technologies and Tools

For the implementation of the Resource Orchestration service, a variety of technologies and tools will be exploited depending on the supported functionalities. For the functionalities related to the holistic resource management of the virtualised infrastructure, Java 1.8 and Spring Boot framework will be exploited in order to model the REST APIs of the supported Infrastructure as a Service (IaaS) offerings of the virtualised infrastructure. Additionally, for the implementation of resource management, the straightforward extensibility of the framework with the support for various libraries, such as OpenStack4j, will be leveraged. For the dependency management and automation of the building process, Apache Maven will be leveraged. Additionally, the Swagger framework will be utilised for the design and documentation of the REST APIs.

For the dynamic deployment of services and applications and their continuous monitoring and management, a series of artefacts will be developed. The main artefact offering the cloud orchestration engine is based on UBITECH's powerful cloud orchestrator named Maestro[14].

For the service discovery and registration, Consul will be exploited. Consul offers a server providing a variety of tools for service discovery and configuration in any infrastructure. It communicates with the corresponding Consul client, operating on the virtual machines, in order to complete the node registration process, collect the desired information from the node, configure the services running on the node and perform health checks for the availability of the services on the node periodically.

For the monitoring of the resources and performance of the virtual machines, Netdata will be utilised. Netdata will play the role of the monitoring agent and is a well-known, fast and efficient monitoring agent offering a variety of functionalities that will be exploited in the resource management.

The Container Engine that is used in each VM is the Docker Engine. The Docker Engine is capable of managing the deployment and execution of the containerised application or services with the use of Docker images that are executed with the appropriate arguments, as orchestrated by the cloud orchestration engine.

[14] Maestro, https://themaestro.ubitech.eu/

3.15 Master and Worker services

3.15.1 Services Design

In the ICARUS platform architecture, as presented in ICARUS Deliverable D3.1, the Master / Worker paradigm is adopted in order to enable the "remote" execution of a job, that is designed by one of the various processes implemented by the services running within the Core ICARUS platform, on either the On Premise Environment or the Secure and Private Space. To achieve this, a cross-tier intercommunication is required and this intercommunication is facilitated by two complementary services whose execution is tightly connected, namely the **Master service** and the **Worker service**.

Following the Master / Worker paradigm, these two services are operating in a collaborative manner in order to realise the "remote" executed of the designed jobs. Towards this end, each service undertakes different responsibilities on a different tier and, combined together, they provide the means to the rest of the services of the platform to perform the execution of the design job on a different tier of the ICARUS architecture.

The Master service is operating on the Core ICARUS platform and is utilising the functionalities of the Master Controller component, as described in ICARUS Deliverable D3.1 to fulfil the incoming requests. The Master service is responsible for allocating, managing and monitoring the whole job execution process. Furthermore, in the case of data analysis, the Master Service is undertaking the specific task of transferring the required encrypted datasets to the Secure and Private Space.

More specifically, the Master Service is composed by the following set of services:

a) **A service that performs the compilation of the set of instructions** for the job that will be executed based on the request that is provided by one of the services of the Core ICARUS platform. The Master service is providing the interface that the various service utilise in order to post a new job request and interprets this request accordingly.

b) **A service that establishes the connection with the running instance of the Worker service** on either the On Premise Environment or the Secure and Private Space. The Master service is responsible for ensuring the intercommunication with the relevant instances of the Worker service.

c) **A service that performs the allocation of the requested job to the relevant Worker service** depending on the nature of the job. The Master service is providing the compiled set of instructions to the corresponding Worker service and performs in parallel the continuous monitoring of the execution status.

d) **Securely transfers the selected encrypted datasets to the Secure and Private Space** in order to be used for the specified data analysis job.

As described above, the Master service receives as input the request for the job that will be executed on either the On Premise Environment or the Secure and Private space. The requests are posted in the interface that the Master service is exposing to the rest of the services of the Core ICARUS platform. The output of the Master service is the job execution status of the allocated job that is returned to requesting service.

The Worker service is operating on both the On Premise Environment and the Secure and Private Space. The Worker service is a configurable service incorporating all the functionalities that are offered on both tiers and depending on the tier that it is running the relevant functionalities are made available. The Worker service is using the functionalities of the OnPremise Worker and the SecureSpace Worker components, as described in ICARUS Deliverable D3.1, in order to offer this dual approach. Moreover, the Worker service facilitates the transfer of the datasets or the results of the data analysis to the Core ICARUS platform from the On Premise Environment and the Secure and Private Space respectively.

In particular, the Worker service is composed by a set of services that can be split into:

a) **A service that interprets and executes the instructions** as provided by the Master service by invoking the corresponding services of either the On Premise Environment or the Secure and Private Space in order to fulfil the request job. In these instructions, the specific tasks that should be completed by each service along with the corresponding parameters are defined.

b) **A service that monitors and reports the execution status to the Master service** for each allocated job. As soon as the job is provided to the corresponding service, the Worker polls the corresponding service in order to obtain the current execution status that is passed to the Master service.

c) **A service that enables the uploading of the processed datasets to the Core ICARUS platform** as a result of the dataset preparation executed in the On Premise Environment. Once the dataset is ready to be uploaded in the Core ICARUS platform, the Worker service is informed and the secure transfer is executed.

d) **A service that enables the uploading of the results of the data analysis** executed in the Secure and Private Space. Once the data analysis is completed and the results are encrypted, the Worker service is notified and the results are transferred to the Core ICARUS platform.

The Worker service receives as input a job execution in the form of compiled instructions from the Master service that are interpreted and executed accordingly. The output of the service is the execution status of the relevant job that is provided back to the Master service. Additionally, in the case of uploading either the prepared dataset or the results of the data analysis to the Core ICARUS platform, the relevant request is provided as input and a confirmation is provided upon the successful transfer execution.

3.15.2 Technologies and Tools

For the implementation of the Master service and the Worker service, Java 1.8 and the Spring Roo[15] framework will be leveraged. Spring Roo is an easy-to-use rapid development tool for building application in the Java programming language with a large range of features and integration capabilities.

[15] Spring Roo, https://projects.spring.io/spring-roo/

4 Added Value Service Bundles

The analysis workflows that were presented in section 2 besides the core services, that were documented in the previous, provided a list of complementary services that provide added value functionalities to the ICARUS platform. Following the same approach that was followed in section 3, a list of added value services is compiled and is illustrated in Figure 4-1. Each added value service is a bundle of underlying services that were grouped based on the area of service and the related functionalities offered. As with the core services, the interactions of the extracted added value services are facilitated through well-defined APIs.

In the following subsections, each main added value service is presented, focusing on the description of the underlying services that are bundled within each main added value service and the technologies and tools that will be exploited in the implementation of these services.

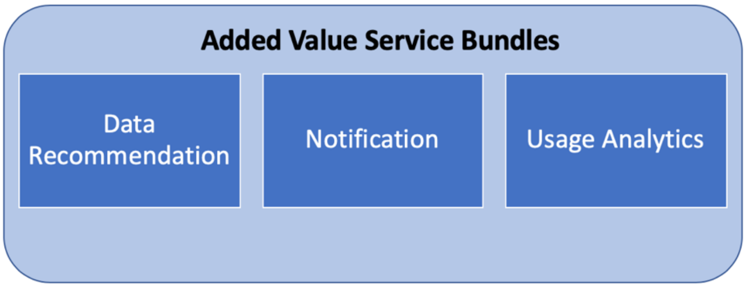

Figure 4-1: ICARUS Added Value Service Bundles

4.1 Data Recommendation Service

4.1.1 Service Design

The purpose of the Data Recommendation service is to provide accurate suggestions of data assets that exist in the ICARUS repository to enable the stakeholders to find data assets relevant to their interests. The Data Recommendation service is based on the Recommender component and its core functionalities, as described in deliverable D3.1. Under the scope of this service, different inputs are leveraged in order to produce the required outputs of the service and several interactions exist with other components of the ICARUS platform and their provided functionalities.

The Data Recommendation service uses a hybrid approach that utilizes different recommendation models. On top of these models, there is an **aggregation service** that weights and combines the recommendations of each model, aiming to hybridize the features of multiple recommendation techniques and benefit from their strengths. More specifically, these models are:

- **Semantic content-based model:** It generates recommendations for a user by mapping the user's preferences (e.g. weather data, flights data, etc.) that are manually set in his/her profile to the categories and entities of the data assets.
- **User-based model:** It generates recommendations for a user based on other similar users (e.g. if users A and B have purchased the same three data assets and user B has purchased another data asset, then user A may receive recommendation for this data asset).
- **An item-based model:** It generates recommendations for a user that is exploring a specific data asset based on other similar data assets (e.g. let data assets A and B, referring to three same categories, then a user that is exploring data asset A, may receive recommendation for data asset B).

The Data Recommendation service is involved in two distinct phases in which it needs to interact with different components, collect various inputs and perform specific tasks. Specifically:

- **Offline training phase:** During configured time intervals, an **offline training service** receives as input relevant information about the data assets and the users' behaviour. More precisely, the offline training service receives the users' IDs and the users' history (i.e. searches, views, purchases) in order to be trained for providing a set of recommended data assets to the users, using the User-based Model. In addition, the service receives as input the data assets' IDs and their categories (e.g. weather, flight delays, etc.) in order to be trained for providing a set of recommended data assets using the Item-based Model.
- **Execution phase:** The phase is triggered when a user explores the ICARUS repository to find data assets and the **recommendation execution service** receives the user's ID and the data asset's ID that the user is currently viewing. The trained User-based and Item-based Models are used, as well as the Semantic Content-based Model which is not trained, but it generates recommendations after receiving as input the user's preferences and semantic metadata about the data assets' entities derived from the ICARUS Ontology. The outputs of all the models are later combined by the aggregation service.

Hence, the Data Recommendation service is a bundle of services that interact with other services via the REST APIs that are provided by these components. In detail, the Data Recommendation service interacts with the ICARUS Storage in order to receive the relevant information that is stored there for the recommendation execution. Accessing such information enables the Data Recommendation service to accomplish one of its main functionalities which is to improve its recommendations by re-training the algorithm periodically. Furthermore, when a stakeholder uses the Data Exploration service, the Data Recommendation REST API is called by this service and receives the user's ID, as well as the data asset ID that the user is currently viewing. The interaction with the front-end of the Data

Exploration service triggers the second functionality of Data Recommendation service which provides recommendations for data assets to the user. In addition, the Data Recommendation service interacts via SPARQL with the ICARUS Ontology that is included in the ICARUS Storage.

4.1.2 Technologies and Tools

The recommendation algorithm of the Data Recommendation service will be written in Python and it will utilize the Surprise[16] library. Surprise is a Python scientific toolkit for building and analysing collaborative-filtering recommender systems that contains various built-in algorithms with a focus on rating prediction such as different variations of matrix factorization-based algorithms (e.g. SVD, SVD++, etc.) that the Data Recommendation service aims to implement. The major reason for selecting this library over other similar technologies is that Surprise is quite popular and provides a detailed documentation that makes it easy to use and to implement custom algorithms. In addition, it provides various similarity metrics (e.g. cosine, MSD, Pearson, etc.) and efficient tools for evaluating the performance of different algorithms. As for the design and documentation of the Recommender's API, the well-known open source software Swagger will be used. The development of the API will be based in Flask, the Python micro web framework, that was selected over other technologies as it is lightweight and easy to use.

4.2 Notification Service

4.2.1 Service Design

The Notification service of ICARUS is responsible for providing the updated information to the users with regards to data assets or scheduled analytics jobs. As described in D3.1, the Notification Manager is the component responsible to manage the notification service and serves as the basis for the Notification service. More specifically, the Notification 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. Additionally, it is responsible to notify the users for any updates on the execution status of their scheduled analytics jobs. Finally, it is responsible to inform data providers for requests on their data assets and notify the data consumers for the approval or rejection of their requests, as well as the progress in the smart contract preparation.

The implementation of the Notification service will use the event-driven paradigm and will be based on the publish and subscribe message pattern (Trihinas et al, 2015). In the classic publish and subscribe message pattern, entities (referred to as subscribers) initially express interest and subscribe to an event stream of another entity (referred to as the publisher). A

[16] Surprise, http://surpriselib.com/

subscriber can be interested in receiving event notifications from multiple publishers. When events are generated, the publisher distributes them to its subscribers, eliminating the need of the subscriber to constantly poll the publisher to check if new events are available.

The Notification service acts as a subscriber to events that are published from other services. Depending on each type of event, the Notification service is responsible to follow a different procedure and interact with the appropriate components in order to receive the information needed. Specifically, events will be raised for:

- **Addition of a new data asset:** raised by the Data Upload service which is responsible to provide the information related to the new data asset such as the data asset's ID and its owner, as well as the categories of the data asset.
- **Update of an existing data asset:** raised by the Data Upload service which is responsible to provide the information related to the data asset (e.g. data asset's ID and its owner) and the categories of the data asset.
- **Update on the execution status of a scheduled analytics job of a user:** raised by the Data Analytics service, containing information about the user ID, the analytics job ID and the new status (i.e. initiated, completed, failed) with any possible description.
- **Request for a data asset:** raised by Asset Brokerage service which is responsible to provide information about the user that requested the data asset, as well as the data asset and its owner.
- **Response to a request for a data asset:** raised by Asset Brokerage service, when the data provider accepts or rejects a data asset request, providing the data asset ID, the data provider ID and the ID of the user that requested the data asset.

Based on the event-driven paradigm, the Notification service consists of three main services that will enable the handling of the events:

a) **Notification Message Queue:** a high-performance queueing service that receives and manages the notification events.

b) **Notification Storage:** is a NoSQL distributed database, which stores historical notification data and which can scale depending on the imposed load.

c) **Notification Message Handler:** consumes the notification events from the Notification Message Queue and store them in the Notification Storage. Furthermore, the Notification Message Handler will proceed to notify users in the platform's notification panel, as well as via emails.

All the notifications to the users are shown in the notifications panel of the ICARUS platform and they are sent to the users via emails, if the users select these types of notifications in their profiles. In order to provide all notification history to the users, the Notification service needs to maintain a notifications storage to store the notifications for each user in the ICARUS Storage.

As described above, the Notification service interacts with various services of the platform. The exchange of data is succeeded using various events that are pushed in a message queue by the corresponding component. The requests of the Notification service for collecting further information are facilitated via REST APIs, provided by related services.

4.2.2 Technologies and Tools

For the implementation of the Notification service a variety of technologies and tools will be exploited. Apache Kafka[17] will be used for the implementation of the message queue with high-throughput, low Latency, fault-tolerance and durability. Apache Kafka is a distributed streaming platform which is generally used for building real-time streaming data pipelines that reliably get data between systems or applications, as well as streaming applications that transform or react to the streams of data. For the persistency layer of the notifications storage, the component utilizes a NoSQL database, namely MongoDB[18] as provided by the ICARUS Storage. As for the Notification Manager's API, Swagger will be used for the design and documentation. For the development of the API, Spring Boot Framework will be used.

4.3 Usage Analytics Service

4.3.1 Service Design

The Usage Analytics service is one of the added value services that are offered in ICARUS, aiming to provide meaningful platform utilization insights. As described in D3.1, the component of the reference architecture that is responsible to collect, aggregate and visualize this information is the Usage Analytics component which will be exploited for the service implementation.

The aim of Usage Analytics service is to provide the ICARUS users with platform utilization statistics. These statistics will not reveal any information related to another user, but instead, these will be aggregated statistics that focus on the type of the targeted user (i.e. data provider, data consumer, admin). For instance, a data provider will receive the total number of views for his/her data asset, without receiving the list of the users that viewed the data asset. In addition, a data consumer will be able to view the total number of users that purchased a data asset, without knowing who these users are. The aggregated utilization analytics can be categorized in four groups:

- **Data assets analytics** for providing utilization statistics for the datasets of the platform.

[17] Kafka, https://kafka.apache.org/
[18] MongoDB, https://www.mongodb.com/

- **Service assets analytics** for providing utilization statistics of the platform algorithms and applications.
- **Platform usage analytics** for providing utilization statistics for the whole platform.
- **User analytics** for providing utilization statistics of the users' secure private space**.**

The Usage Analytics service adheres to the event-driven paradigm and will be based on the pub-sub message pattern. All relevant services managing information related to the previously mentioned utilization analytics are responsible to publish serializable events, summarizing the actions that have been executed in a timely manner. These events are raised in the following cases:

- **Data asset purchase:** raised by Asset Brokerage service, containing information such as the data asset ID, the data consumer who purchased the data asset and the timestamp of the purchase.
- **Allocation of computational resources:** raised by Resource Orchestration service for the allocation and de-allocation of computational resources (e.g. CPU, memory, storage, etc.) to the users' private spaces respectively, accompanied with the specific timestamps.
- **Exploration of data assets:** raised by Data Exploration service, containing information related to which data assets appear in search results and which of them are viewed. These events do not only include the data asset ID and the related user ID, but also the specific timestamp in each case.
- **Usage of services:** raised by Data Analytics service, containing information related to the usage of each service asset and the user that utilized the service, including the specific timestamp.
- **Scheduled analytics jobs:** raised by Data Analytics service, containing information about the users' analytics jobs, including information about their jobs' status and timestamps, when a job is initiated or completed (either successfully or failed due to an error such as lack of resources).
- **User registration or sign-in:** raised by Policies Enforcement service when a user registers or sign-ins to ICARUS, containing the user ID and the specific timestamp of the related action.

Based on the event-driven paradigm, Usage Analytics consists of four main services that will enable the handling of the events:

a) **Usage Message Queue:** a high-performance queueing service that receives and manages the usage events.

b) **Usage Storage:** is a NoSQL distributed database, which stores historical usage data and which can scale depending on the imposed load.

c) **Usage Analytics Event Handler:** a service that consumes the utilization events and stores them in the Usage Storage database.

d) **Query Translator:** when a user requests to visualize the utilization statistics, this service is responsible to retrieve and aggregate these statistics from the Usage Storage.

4.3.2 Technologies and Tools

For the implementation of the Usage Analytics service, several technologies and tools will be utilised. The message queue will be based on Apache Kafka. The usage storage will be implemented based on MongoDB, which is a NoSQL distributed storage solution provided by the ICARUS Storage capable of scaling depending on the imposed load for accessing historical notifications data. The API of the Usage Analytics service will be designed and documented using Swagger, while for the development of the API Spring Boot Framework will be used. As for the front-end of the user's interface and the dashboard, they will be implemented using Vue JS, as well as a set of libraries such as Highcharts JS[19] that will be used for providing the visualization of the different statistics using various charts and plots.

[19] Highcharts, https://www.highcharts.com/

5 Conclusions & Next Steps

The purpose of this deliverable entitled D3.2 "Core Data Service Bundles and Value Added Services Designs" was to deliver the design specifications of the Core Data Services and the Added Value Service Bundles in ICARUS. The deliverable is built directly on top of the main outcomes and the knowledge extracted from deliverable D3.1 in order to deliver the details of the design of the services of the integrated ICARUS platform that will drive the implementation activities performed within the context of WP4.

At first, the adopted design process is elaborated, describing the steps that were followed in order to deliver the services specifications. The outcomes from the work that was performed within the context of WP1 with regard to ICARUS methodology and the outcomes from WP2 with regard to the data management, analytics and sharing methods, as well as the technical requirements of ICARUS platform and the first version of the conceptual architecture of the ICARUS platform and its components, were taken as input in this process. The thorough analysis of this input provided the design of the ICARUS platform's workflows.

The designed workflows are presenting the functionalities of the integrated ICARUS platform, focusing on the interactions of the users with the platform that are internally translated into constructed operations within the platform, as well as the interactions of the various components of the platform. The designed workflows were presented in the form of BPMN diagrams and were organised in categories based on the areas of the functionalities of the platform. The list of categories included the areas of Data Preparation, Data Collection, Data Analytics and Visualisations, Data Exploration, Asset Brokerage, Data Security, Backend Ancillary Processes, Data Recommendation, Notifications and Usage Analytics. For each area, a set of workflows was presented followed by a detailed description of the workflow and 29 workflows were presented in total. Furthermore, an initial analysis was performed on the presented workflows in order to present the initial list of identified services as extracted by these workflows.

From the analysis of the designed workflows, 18 service bundles were extracted. These service bundles were further organised into the Core Data Service Bundles and the Added Value Service Bundles. The Core Data Service Bundles includes the Data Cleansing, Data Anonymisation, Data Mapping, Metadata Handling, Data Upload, Data Analytics, Application Catalogue, Application Execution, Data Exploration, Data Licensing, Asset Brokerage, Policies Enforcement, Data Encryption and Decryption, Resource Orchestration, Master and Worker services. On the other hand, the Added Value Service Bundles is composed by the Data Recommendation, Notification and Usage Analytics services.

Each service bundle includes a main service and a set of underlying services with specific responsibilities and functionalities that were elaborated in this deliverable. Furthermore, for the implementation of each service bundle, a set of technologies and tools will be leveraged that were also presented in the context of this deliverable.

The current deliverable presented the design specifications of the ICARUS platform's services that were formulated following the described design process. However, as the project development activities evolve, this initial design of the described services will receive the necessary updates and optimisations in order to encapsulate all the project's advancements, as well as the new technical requirements that will be extracted from the feedback that will be collected from the platform's evaluation. Hence, the forthcoming versions of this deliverable will incorporate all the updates that are necessary to be introduced.

Annex I: References

Model, O. B. P. (2009). Notation (BPMN) Specification 2.0 V0. 9.15. Object Management Group.

Trihinas, D., Pallis, G., & Dikaiakos, M. (2015). Monitoring elastically adaptive multi-cloud services. IEEE Transactions on Cloud Computing.