



D5.2

First version of E-CORRIDOR Architecture

WP5 – E-CORRIDOR Platform: Requirements / Architecture / Implementation and integration

E-CORRIDOR

Edge enabled Privacy and Security Platform for Multi Modal Transport

Due date of deliverable: 31/05/2021
Actual submission date: 31/05/2021

31/05/2021

Version 3.0

Responsible partner: HPE

Editor: Mirko Manea

E-mail address: mirko.manea at hpe.com

Project co-funded by the European Commission within the Horizon 2020 Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Authors: Claudio Caimi, Mirko Manea, Patrizia Ciampoli (HPE), Paolo Mori, Ilaria Matteucci, Giacomo Giorgi (CNR), Thanh Hai Nguyen (CEA), Christian Plappert (FhG), Stefano Sebastio (UTRC)

Approved by: Paolo Mori, Fabio Martinelli (CNR), Florian Fenzl, Christian Plappert, Roland Rieke (FhG)

Revision History

Version	Date	Name	Partner	Sections Affected / Comments
0.1	02-Mar-2021	M. Manea, P. Ciampoli, C. Caimi	HPE	Initial ToC
0.2	22-Mar-2021	M. Manea, P. Ciampoli, C. Caimi	HPE	Added contribution
0.3	28-Apr-2021	S. Sebastio	UTRC	Merged contribution to AT Pilot and section 12
0.4	29-Apr-2021	G. Giorgi, M. Ammara, P. Mori	CNR, MISE, CLEM, CNR	Merged ISAC Pilot and S2C Pilot contributions Merged ISI contribution
0.5	05-May-2021	M. Manea, S. Sebastio, P. Mori	HPE, CNR, UTRC	Merged HPE, CNR, UTRC contributions
0.6	11-May-2021	T. Nguyen	CEA	Merged CEA contribution
0.7	12-May-2021	C. Plappert	FhG	Added FhG contribution
2.0	13-May-2021	M. Manea	HPE	Available for internal review
3.0	31-May-2021	M. Manea, F. Martinelli	HPE, CNR	Put sequence diagrams in portrait page mode for better readability Released

Executive Summary

This deliverable defines the first version of the E-CORRIDOR architecture, which comprises five major subsystems, namely Information Sharing Infrastructure - ISI, Information Analytics Infrastructure - IAI, Data Sharing Agreement (DSA) Lifecycle Infrastructure - DLI, Common Security Infrastructure - CSI, and Advanced Security Infrastructure - ASI. Each subsystem fulfils a set of requirements, from data sharing, to data analysis, to sharing rules definitions, and to define security services built on top of such core functionalities. The E-CORRIDOR architecture delivered at M12 represents the reference framework that the project pilots will instantiate following their specific peculiarities: for this reason we describe and provide several deployment models of the architecture.

This document shows all the components of the E-CORRIDOR subsystems, describes their functionalities and their interaction for the most common use cases, including the sharing of a piece of information, and the execution of analytics service on top of such information.

We also report the current status of the E-CORRIDOR technical environments, which will start hosting the development activities of the framework to meet the milestone of M24, where we plan to deliver the first E-CORRIDOR implementation for the project pilots.

We conclude with a description of the contribution of this deliverable towards the E-CORRIDOR objectives, as stated in the description of action (DoA).

Table of contents

- Executive Summary 3
- 1. Introduction 6
 - 1.1. Deliverable Structure 6
 - 1.2. Definitions and Abbreviations 6
- 2. High-Level Architecture 9
- 3. Deployment Model 12
 - 3.1. Edge-to-cloud distributed 13
 - 3.2. Edge-only distributed 14
 - 3.3. Mixed distributed 15
 - 3.4. Instantiation on the pilots 16
 - 3.4.1. Pilot AT 16
 - 3.4.2. Pilot S2C 17
 - 3.4.3. Pilot ISAC 18
- 4. Subsystem: ISI - Information Sharing Infrastructure 20
 - 4.1. ISI API 22
 - 4.2. Data Usage Control System 22
 - 4.3. Bundle Manager 23
 - 4.4. Bundle Store 23
 - 4.5. Bundle Store Interface 23
 - 4.6. Buffer Manager 24
 - 4.7. Obligation Toolbox 24
 - 4.8. DMO Toolbox 24
- 5. Subsystem: IAI - Information Analytics Infrastructure 25
 - 5.1. IAI API 26
 - 5.2. Service Usage Control System 26
 - 5.3. Analytics Orchestrator 26
 - 5.4. Analytics Toolbox 27
 - 5.5. Legacy Analytics Engine 27
 - 5.6. Virtual Data Lake 28
- 6. Subsystem: DLI - DSA Lifecycle Infrastructure 29
 - 6.1. DSA Editor 30
 - 6.2. DSA API 31
 - 6.3. DSA Mapper 31
 - 6.4. DSA Store 32
 - 6.5. DSA Store Interface 32
- 7. Subsystem: CSI - Common Security Infrastructure 33

- 7.1. Identity Manager..... 34
- 7.2. Key & Encryption Manager 34
- 7.3. Secure Audit Manager 35
- 8. Subsystem: ASI - Advanced Security Infrastructure 37
 - 8.1. ASI API 39
 - 8.2. Discovery Security Services Manager 39
 - 8.3. ASI Orchestrator 39
 - 8.4. Advanced Security Privacy-Aware Services 39
 - 8.4.1. Privacy-Aware Interest-Based Service Sharing 39
 - 8.4.2. Privacy-Aware Authorization 39
 - 8.4.3. Trusted Service Manager 40
 - 8.4.4. Privacy-Aware Seamless Multimodal Authentication 40
 - 8.4.5. Continuous Behavioural Authentication 40
- 9. Data Flow Diagrams..... 41
 - 9.1. Create Bundle 41
 - 9.2. Read Bundle 43
 - 9.3. Delete Bundle 45
 - 9.4. Transfer Bundle 47
 - 9.5. Execute Analytics 49
- 10. Subsystems Communication..... 52
- 11. Status of the Dev/Test/Prod environments 54
 - 11.1. Development Environment 54
 - 11.1.1. Infrastructure Configuration 54
 - 11.1.2. Software and Tools Configuration 54
- 12. Contributions towards the E-CORRIDOR objectives 56
- 13. Conclusions 59
- 14. References 60

1. Introduction

This document presents the first iteration of the design of the reference architecture for the E-CORRIDOR Framework. Its design has been conceived to satisfy the requirements from the project pilots as collected in report D5.1. This reference architecture will form the basis under which the development and integration activities will take place during the next months while implementing the first version of the reference architecture by Month 24 as deliverable D5.3. That implementation will be installed in the test environment and will be used for project pilots testing and integration tasks.

The diagrams reported in the document follow the *Fundamental Modeling Concepts* (FMC) [1] framework for describing the architecture components and their communication.

1.1. Deliverable Structure

The document is structured as follows:

- Section 1 is this introduction.
- Section 2 reports the high-level architecture.
- Section 3 describes the deployment model of the Framework, including its instantiation on the pilots.
- Sections 4-8 provide a clear description of all the architectural components, grouped by subsystem.
- Section 9 provides a set of data flow data diagrams that describe the main operations of the reference Framework.
- Section 10 shows machine-to-machine communication techniques.
- Section 11 reports on the status of the Development, Test bed and Production environments.
- Section 12 illustrates the relation with E-CORRIDOR project objectives.
- Section 13 reports final notes.
- Section 14 lists document references.

1.2. Definitions and Abbreviations

Term	Meaning
AES	Advanced Encryption Standard
ADS-B	Automatic Dependent Surveillance-Broadcast
AT	Airport and integrated Train transport (WP2 pilot)
BCBP	Bar-Coded Boarding Pass
C3ISP	Collaborative and Confidential Information Sharing and Analysis for Cyber Protection
CAN	Controlled Area Network
CAPEC	Common Attack Pattern Enumeration and Classification
CEF	Common Event Format
COTS	Commercial Off-The-Shelf
CPE	Common Platform Enumerations

CTI	Cyber Threat Information
DoA	Description of Action
DMO	Data Manipulation Operations
DSA	Data Sharing Agreement
DUCS	Data Usage Control System
EML	Electronic Mail
eIDAS	electronic Identification, Authentication and trust Services
EU	European Union
FMC	Fundamental Modelling concepts
FHE	Fully Homomorphic Encryption
GDPR	General Data Protection Regulation
GPS	Global Positioning System
GPX	GPS Exchange Format
HDFS	Hadoop Distributed File System
HTTP	Hyper-Text Transfer Protocol
IoT	Internet of Thing
IAI	Information Analytics Infrastructure
IATA	International Air Transport Association
ICAO	International Civil Aviation Organization
IIT	<i>Istituto di Informatica e Telematica</i> at CNR
ISAC	Information Sharing and Analytics Centre (WP4 pilot)
ISI	Information Sharing Infrastructure
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
JPEG	Joint Photographic Experts Group
KMS	Key Management System
LAS	LASer format
M2M	Machine to Machine
MoSCoW	“Must have”, “Should have”, “Could have”, “Won’t have but would like”
MMT	Multi-Modal Transport
MRH	Multi Resources Handler
NMEA	National Marine Electronics Association
NFR	Non-Functional Requirement
ODB	On Board Diagnostics

OWASP	Open Web Application Security Project
OWL	Ontology Web Language
PDP	Policy Decision Point
REST	REpresentational State Transfer
RFID	Radio-frequency identification
RPC	Remote Procedure Call
RSSI	Received Signal Strength
SAML	Security Assertion Markup Language
S2C	Car Sharing in Smart Cities (WP3 pilot)
SSO	Single Sign On
STIX	Structured Threat Information eXpression
SUCS	Service Usage Control System
TPM	Trusted Platform Module
UC	Use Case
US	User Story
UUID	Universally Unique Identifier
VDL	Virtual Data Lake
VM	Virtual Machine
XACML	eXtensible Access Control Markup Language

2. High-Level Architecture

The E-CORRIDOR framework architecture consists of a set of services that can be used in the multi-modal transport scenarios to attain the project pilot’s objectives.

Firstly, we introduce the conventions used in the E-CORRIDOR architecture. Figure 1 schematically depicts the conventions used in the E-CORRIDOR architecture design. The E-CORRIDOR *system architecture* involves several discrete *subsystems* where *actors* can interact with them to exploit the underlying services. Each subsystem can be split into different entities that are called *components*. Complex components can be further divided into modules for better manageability and representation.

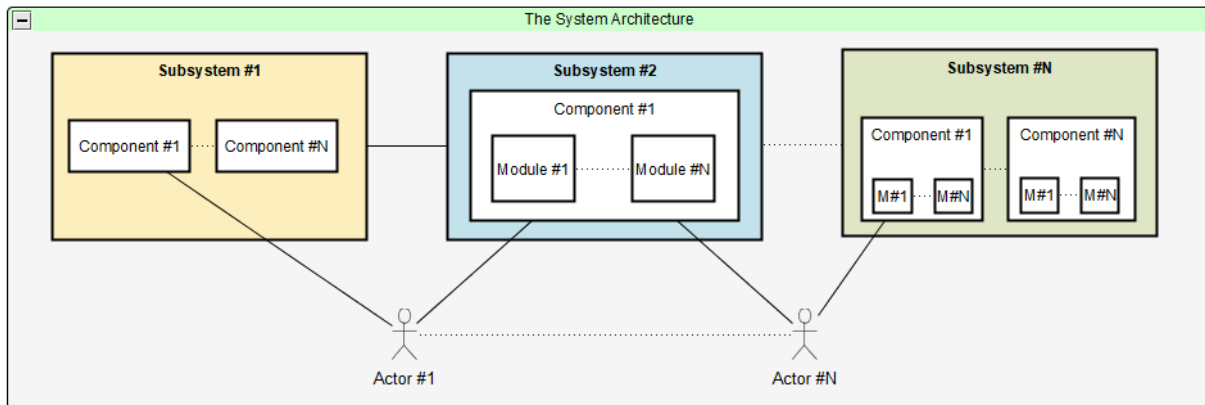


Figure 1: Conventions for E-CORRIDOR architecture design

The E-CORRIDOR actors are users or (external) systems that interact with the E-CORRIDOR framework. These entities are called **Prosumers**, which is a combination of Producer and Consumer. A Producer generates information that is submitted to the system, while a Consumer retrieves information from the system and a Prosumer performs both jobs at different times. Figure 2 illustrates this hierarchy. Throughout the document, the Prosumer is used by default, however, to clarify the exact roles - Producer or Consumer is used according to a given context.

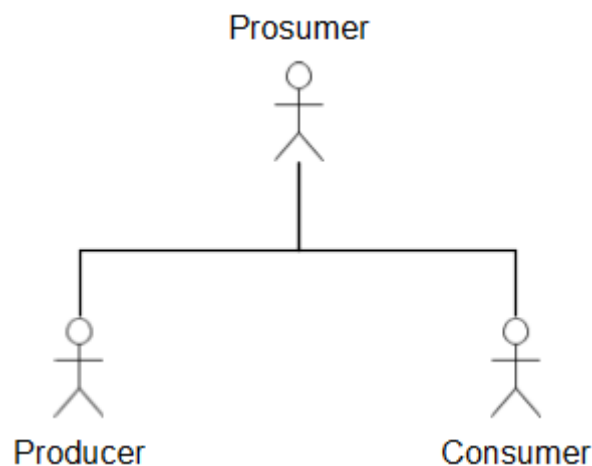


Figure 2: Hierarchy for the Prosumer entity

Figure 3 presents the high-level reference architecture for E-CORRIDOR. It builds upon, extends and improves the work done in the C3ISP EU Project [3], by tailoring to the specificities of the E-CORRIDOR use case scenarios. We colour-coded each major subsystem as described next:

- Data Sharing Agreement Infrastructure (DLI) - Yellow (■);
- Information Sharing Infrastructure (ISI) - Cyan (■);
- Information Analytics Infrastructure (IAI) - Light Green (■);
- Common Security Infrastructure (CSI) - Dark Green (■);
- Advanced Security Infrastructure (ASI) - Pink (■).

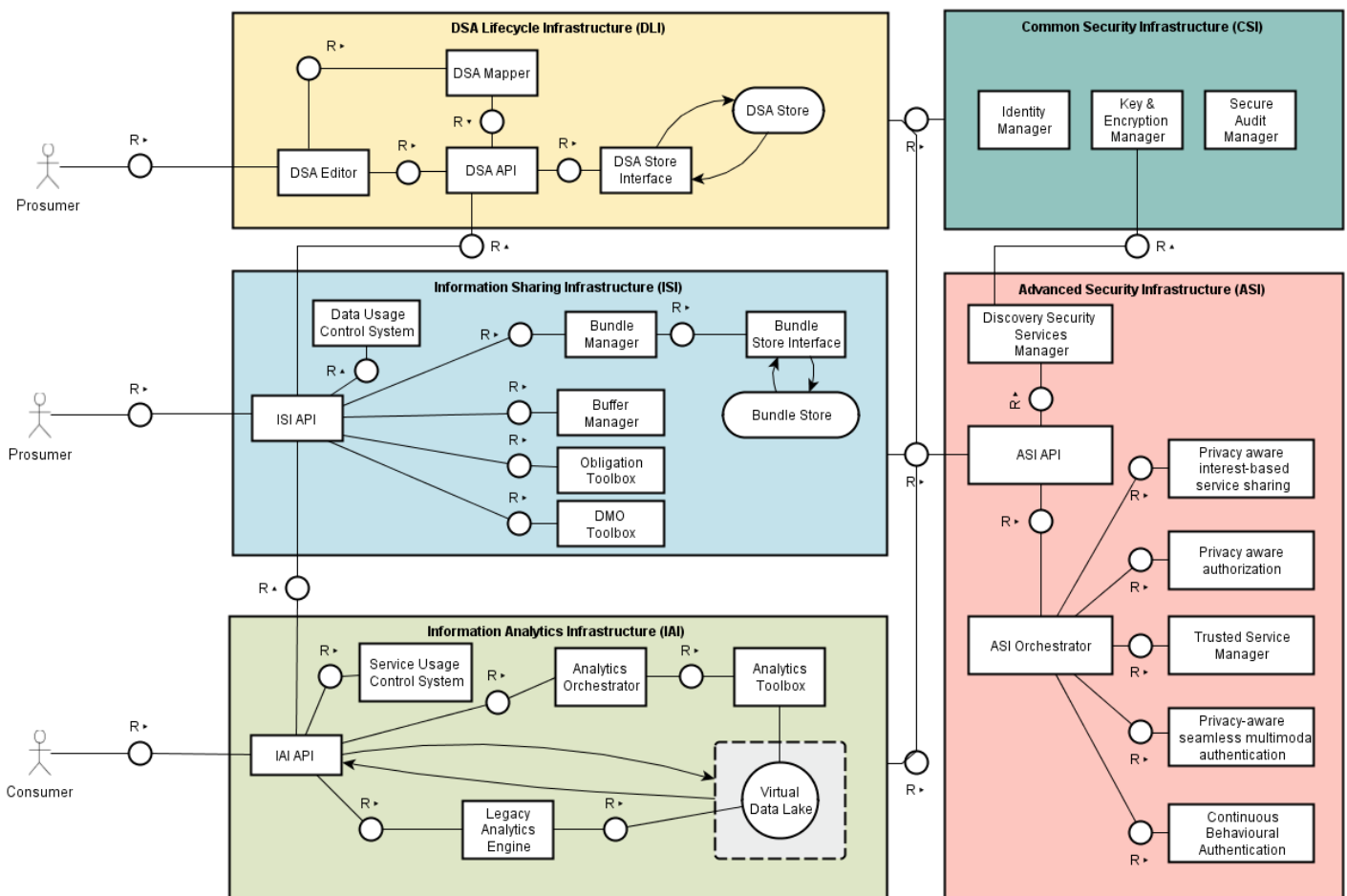


Figure 3: E-CORRIDOR high-level architecture – version 1 (Month 12)

Here, we describe each subsystem to provide a general overview of the major architectural pieces, however, readers can refer to the specific sections (indexed in Section 1.1) for a detailed description of the each component and their interaction. Please note that D6.1 will cover ISI and IAI subsystems with greater detail since WP6 is devoted to information sharing and analytics infrastructures. Similarly, D8.1 will report specifically on the advanced security services components architecture, which is part of WP8 tasks.

The first block in Figure 3 presents the **DSA Lifecycle Infrastructure** that manages the Data Sharing Agreement (DSA) object from its inception to its termination. As anticipated in D5.1, the DSA lays out an agreement (or contract) between different *Parties*. The Parties are for example organizations or companies that feel the need to define rules for exchanging or using data, including *authorisations* or *prohibition* for data access or use, and *obligations* to be

respected, including constraints on data analytics execution and data manipulation. Examples of such rules are presented in Section 6. As you can see in the picture, there are two entry points into the DLI: the Prosumer that authors the DSA and a DSA that is used for enforcing its rules by the ISI subsystem.

The second block in Figure 3 presents **Information Sharing Infrastructure** that is a pillar of the E-CORRIDOR framework since it provides the capabilities to share information between Prosumers (i.e. the Parties that agree on the DSA) respecting and enforcing the defined DSA rules. To achieve that, the ISI can couple a piece of data with its DSA rules forming a bundle of data what we call Data Protected Object (DPO). This concept is known in the literature as *sticky policy* [2] and it allows to setup a digital bundle that ties together the data with its sharing rules: when you move the data, you transfer also its DSA. Two entry points are available for the ISI: the Prosumer that submits data to E-CORRIDOR and the ISI that use the shared data for analytical purposes.

Data is shared among parties to be able to infer more value for all the stakeholders. This is why the **Information Analytics Infrastructure** is a major subsystem of E-CORRIDOR. It is not enough to share data, but also to be able to operate over the shared information by respecting the “contractual constraints” that are defined in the Data Sharing Agreement (DSA). The ISI provides a set of data analytics functions able to operate under the DSA rules, which means for example that analytics results are shared in clear with Party A, while on Party B data is processed for anonymization to respect a specific privacy rule.

The **Advanced Security Infrastructure** exploits the sharing and analytics infrastructures to define added-value services. These services exploit the capabilities of the ISI and IAI for providing their features to evaluate privacy-aware authorisation and authentication mechanisms. ASI is responsible of managing security mechanisms to determine access levels or Prosumer privileges related to system resources including customer profiles, transport services, application features while respecting privacy-aware access to data.

The **Common Security Infrastructure** provides auxiliary services that are necessary to support the other subsystems. These include user identities and profiles to support the decision processes for granting or denying access to the data or other more complex authorisation decisions such as the on-going access control that keeps evaluating if a subject has the right to access particular information. It also helps to achieve security measures that require maintaining data confidentiality such as encryption capabilities, or compliance needs such as tracking who accessed or used what.

Please note, Figure 3 only presents a high-level overview, all the detailed flows between components and subsystems are described further in subsystem-specific sections.

3. Deployment Model

Each pilot will integrate their own infrastructures, tools, and applications within the E-CORRIDOR framework, thus instantiating E-CORRIDOR for their specific use cases. This is why we describe in this section the design of *deployment models*, which are different ways of combining the E-CORRIDOR reference architecture subsystems in terms of how many of them will be used (e.g., how many ISI instances) and where they will be deployed (locally or on-premises at the edge, or centrally at the cloud). In principle, all the E-CORRIDOR subsystems could be deployed either at the edge, or at the cloud, however for the sake of simplicity in the diagrams we are going to show next, we mainly talk about ISI and IAI distributed deployments. Nevertheless, in some cases we will highlight how the remaining subsystems can be deployed, namely DSA Lifecycle Infrastructure (DLI), Common Security Infrastructure (CSI) and Advanced Security Infrastructure (ASI).

We introduce the concept of the **E-CORRIDOR node**, as a specific E-CORRIDOR subsystem deployment, that might only use part of the subsystems and functionalities that are available in the full E-CORRIDOR framework.

Figure 3 also describes one of the ways to deploy E-CORRIDOR (all subsystems are centralised in a single instance). Possible deployment models are enlisted in Table 1.

Table 1 - Available E-CORRIDOR deployment models

E-CORRIDOR deployment model	Description
E-CORRIDOR cloud centralised	This is the simplest form where all the subsystems are hosted centrally at a cloud service. Prosumers connect to the cloud to exploit the E-CORRIDOR functionalities. This deployment model is described in Figure 3.
E-CORRIDOR edge-to-cloud distributed	ISI and IAI are distributed both at local level and at central level.
E-CORRIDOR edge-only distributed	ISI and IAI are only at local level.
E-CORRIDOR mixed distributed	This is the most complex form where components are distributed both centrally and locally, where each Prosumer decides what components to deploy depending on its needs.

Distributing the E-CORRIDOR subsystems poses some challenges. In particular, it is worth noticing that data must be transferred from the edge to the cloud (and vice-versa). This is why we need to introduce a “Data Transfer” concept (see 4.1): of course, this operation needs to be taken into account also in the rules stated in the DSA, which shall regulate this particular aspect as well. For example, moving data to the cloud could imply some data manipulation to anonymise sensitive fields before the transfer process begins.

The instantiation of a subsystem at the edge or the cloud might have different functionalities that would be enabled. For example, the analytics services provided at the Edge might be those that are less computational intensive than those deployed at the cloud (e.g., if the Edge is a smartphone or a car). Also, the Store functionalities (e.g., Bundle Store or Virtual Data Lake) might be specifically declined with respect to the type of E-CORRIDOR node: it would be impractical to have a high-performance distributed repositories like Hadoop HDFS [8] on a car

or a train, but it would be useful to run complex analytical queries on powerful cloud services hosting HDFS.

Since the cloud centralised deployment model has already been discussed in Section 2, the next sections describe the other possibilities.

3.1. *Edge-to-cloud distributed*

The *edge-to-cloud distributed* deployment model is a symmetric model where the E-CORRIDOR nodes host ISI and IAI both at the edge and the cloud. This means that there are distributed sharing capabilities (ISI): Prosumers submit data to the edge-sharing facility (Local ISI) and some data might be also transferred to the cloud (Central ISI). This is useful for example when a Prosumer wants to pre-process the data (e.g., by applying a data anonymization technique) before submitting it to the cloud. There is then the possibility to perform computation (analytics) locally at the edge (Local IAI), for example for achieving results that do not require intense processing workloads, while letting the Central IAI to perform the most compute-intensive jobs.

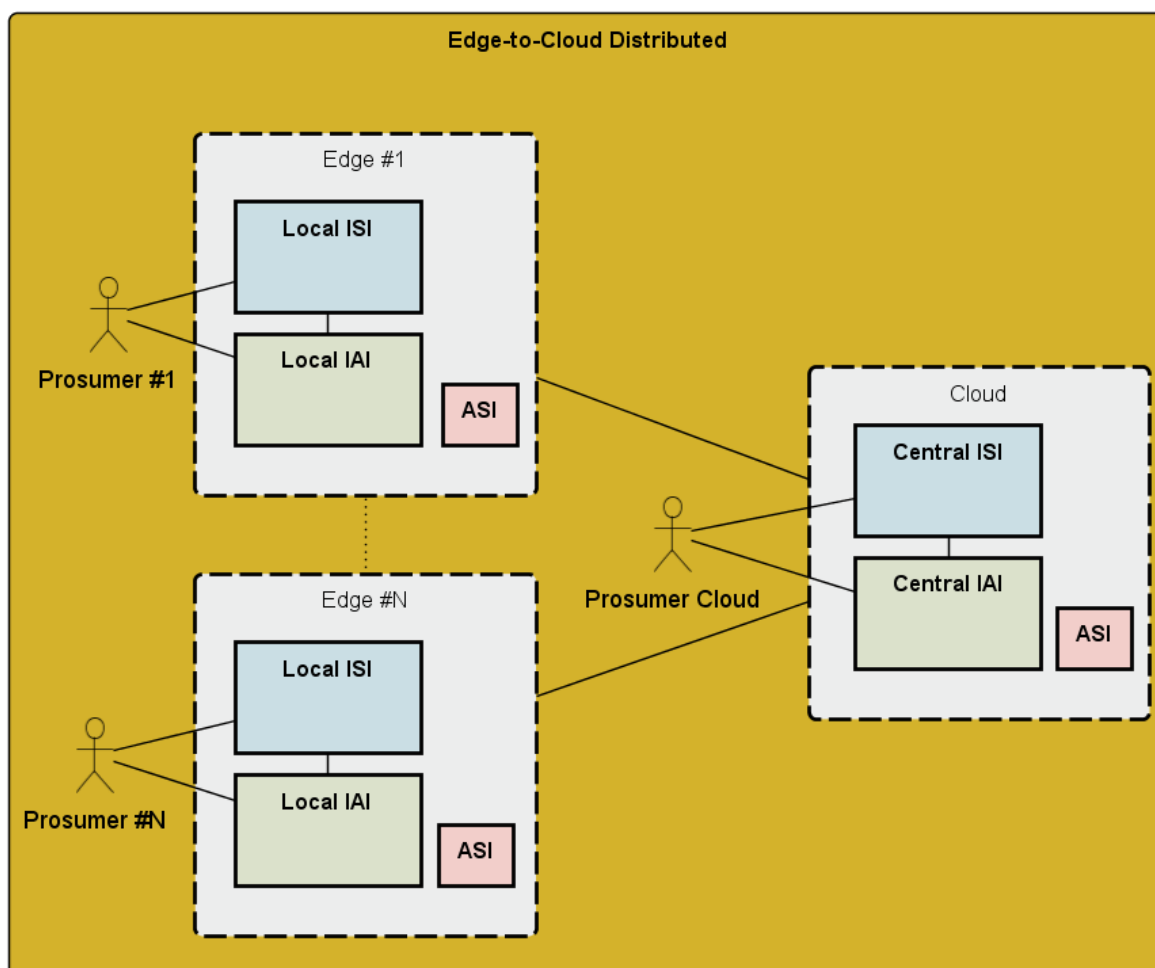


Figure 4: E-CORRIDOR edge-to-cloud deployment model

The services hosted by the Advanced Security Infrastructure (ASI) can perform their tasks either locally, or centrally. Of course, not all the ASI services might be applicable to each E-CORRIDOR node of this deployment model, or an ASI service might cooperate with other edge-based ASI services by using both local and central services (e.g., it might use data

collected locally to generate partial results - as new data - that will then be shared centrally for further processing or to use data coming from different edge-based prosumers).

3.2. *Edge-only distributed*

The *edge-only distributed* deployment model is based only on E-CORRIDOR nodes hosted on-premises or at the edge. This is a kind of peer-to-peer architecture because there is no central service. All the nodes can live on their own, acquiring data (via Local ISI) and performing analytical computations (via Local IAI). ASI services can be deployed to exploit this edge-only model.

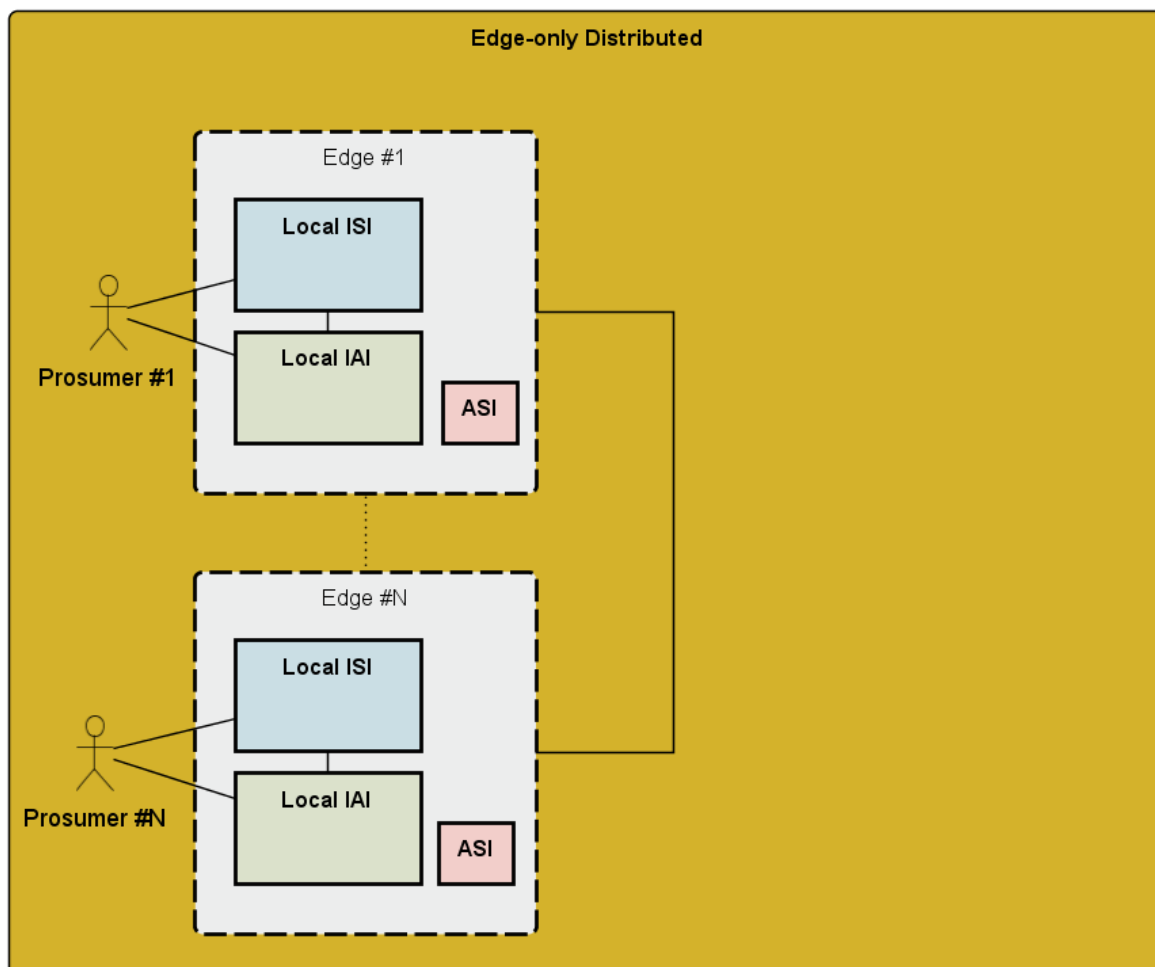


Figure 5: E-CORRIDOR edge-only deployment model

Data can be shared between the different Prosumers, enabling different scenarios, like:

- *Assembly line*: raw data is acquired by Prosumer #1, possibly pre-processed or preliminarily analysed, then passed to Prosumer #2, which continues the chain till a final result is achieved by Prosumer #N. Prosumers might have for example different computing performances.
- *Specialised Prosumers*: each Prosumer might have specific capabilities, like hardware dedicated for cryptography, GPUs for machine learning, etc. In this scenario, a Prosumer might select the most appropriate peer depending on its business case.

3.3. *Mixed distributed*

The *mixed distributed* is a generic deployment model. It must be declined depending on the specific needs and form the basis where E-CORRIDOR Pilots can be instantiated at the more general level.

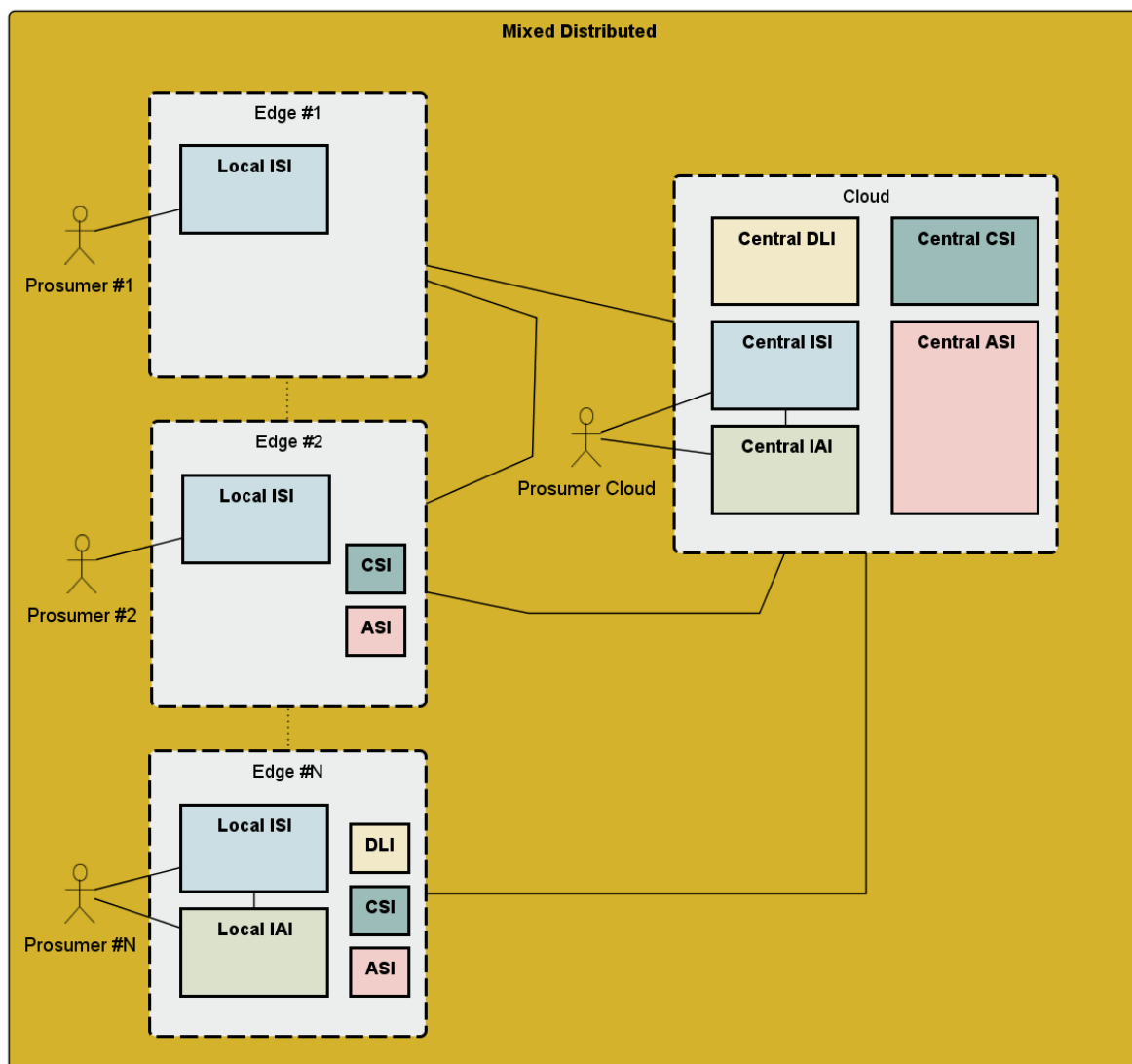


Figure 6: E-CORRIDOR mixed distributed

It can be considered as a parent or superset of the other deployment models, in which E-CORRIDOR subsystems can be freely distributed. However, not all the subsystems distributions might be sound to each E-CORRIDOR pilot. For example, it might not be appropriate to distribute the DSA Lifecycle Infrastructure (DLI), since the DSA rules should be centrally hosted to have a consistent behaviour between all the E-CORRIDOR nodes. In some cases, distributing the DLI might be applicable, like when we need to create parallel E-CORRIDOR environments for development, testing and production.

The Common Security Infrastructure (CSI) can be useful when distributed. For example, a Prosumer might want its auditing capabilities to trace locally what happens at the edge, as well as some encryption services might be run locally. Also a local Identity Manager can be used when identities are federated between different prosumers.

3.4. *Instantiation on the pilots*

This section describes how each E-CORRIDOR pilot can use and customise a deployment model to suit their specific needs. Table 2 summaries the Pilots' choices, while the rationales will be illustrated in the next sections.

Table 2 - Deployment models for the E-CORRIDOR Pilots

Pilot	Selected Deployment Model
Airport-Train (AT) Pilot	Edge-only deployment model
Car Sharing in Smart Cities (S2C) Pilot	Edge-to-cloud distributed
Information Sharing and Analytics Centre (ISAC) Pilot	Mixed distributed

3.4.1. Pilot AT

The Airport-Train (AT) pilot in E-CORRIDOR is characterized by the presence of two main stakeholders managing the corresponding critical infrastructures (airport and train station). Both the infrastructures deal with highly sensitive data concerning passenger personal information (e.g., biometric, passport and travel document). Data privacy *regulations* may prevent that some data are transferred outside of the local security domain that has collected the same in the first instance.

Moreover to achieve a *timely identification* of the passengers and a seamless authentication, the sensor networks deployed in the airport and train station premises need some form of Machine to Machine (M2M) communication to perform a distributed and combined contextual analysis (i.e., at the edge). To attain a frictionless experience, the same computation and communication pattern has to be performed even while passengers change their mode of transportation. Furthermore, in the envisioned AT pilot, passengers can bring their devices as a source of data for the authentication analytics (e.g., sensors embedded in the smartphone can be used to perform gait analysis useful for passenger identification).

To this end, even in presence of bi-lateral agreements between the stakeholders to manage the data exchange, such a scenario requires the execution of *edge computation* (please refer to the *edge-only deployment model* of the E-CORRIDOR framework presented in Section 3.2) to fulfil both *regulatory and performance needs* of the AT pilot. The controlled data sharing allowed by the E-CORRIDOR framework also opens up to the concomitant presence of a cloud node supporting the edge devices in the more computational demanding data analytics (please refer to the *mixed distributed deployment model* of the E-CORRIDOR framework presented in Section 3.3).

Figure 7 presents the deployment model of the E-CORRIDOR framework adopted in the AT pilot. Thanks to the flexibility of the framework, devices at the edge with fewer resources (either in terms of memory or of computational power) may avoid instantiating all the subsystems, if not needed.

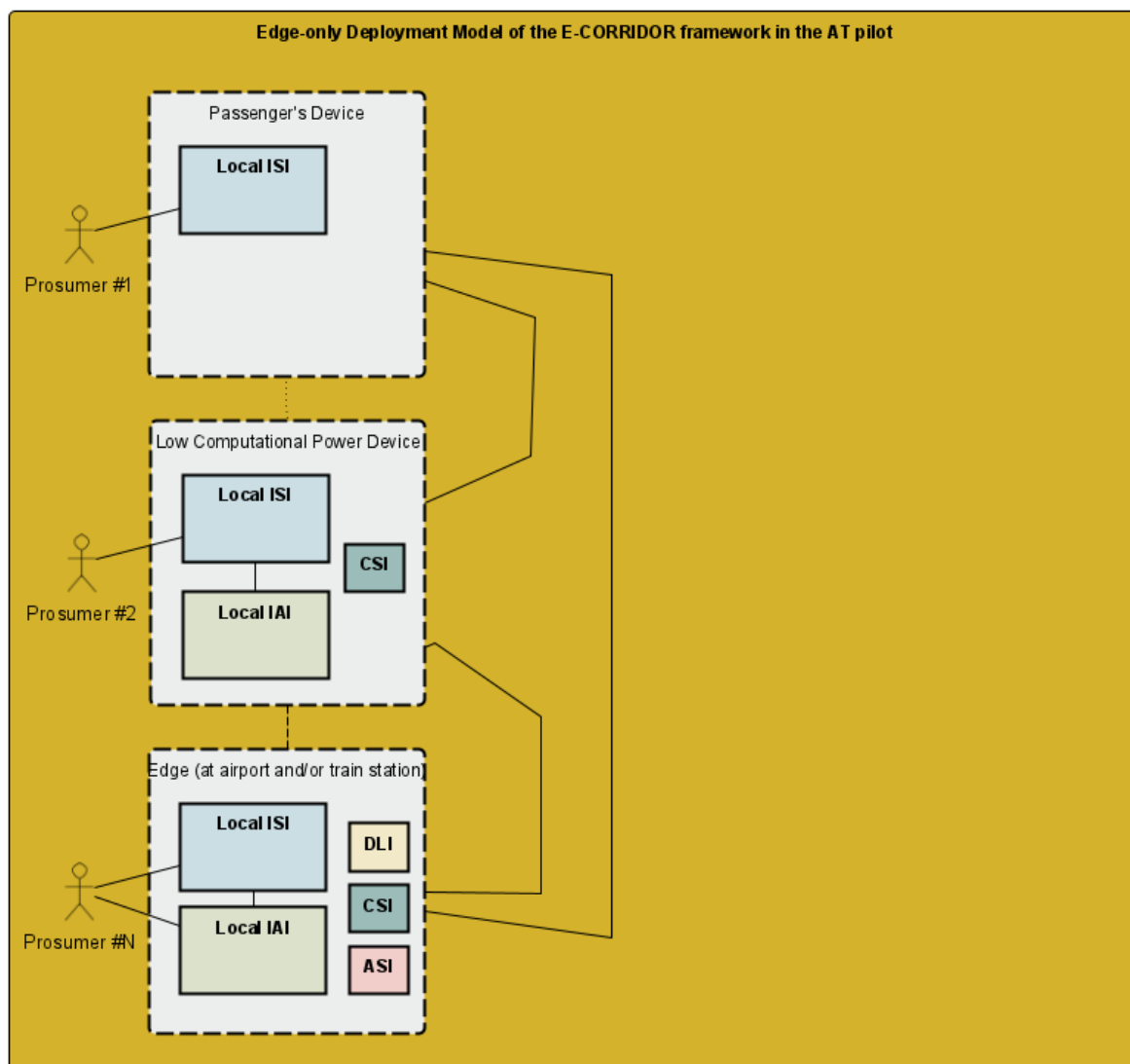


Figure 7: E-CORRIDOR instance for AT pilot

3.4.2. Pilot S2C

The E-CORRIDOR deployment model that suits the S2C Pilot is “E-CORRIDOR edge-to-cloud distributed”. The reasons behind this selection and the deployment strategies are explained as follows.

The S2C Pilot unites different mobility service providers around a unique single digital identity of the travellers that is what we define as *eWallet*. Another central element of the pilot is the advanced security services from the ASI, in particular, the Trusted Service Manager management that helps to support the authorisation and authentications towards the *eWallet*. Having these two central entities require a central E-CORRIDOR cloud. The mobility service providers and the mobility consultancy (with their micro-subsidies analytics toolkit) will deploy the ISI and the ASI on the Edge respectively that would allow the communication between the mentioned entities and the central *eWallet*. The behavioural identification and some of the intrusion detection analytics need to be conducted locally (considering data protection obligations, computational power, and data bandwidth limitations), thus, some of the edge nodes will deploy a local IAI containing the concerned local analytics while the rest of the analytics developed within the WP7 used in the S2C Pilot will be deployed in the central cloud.

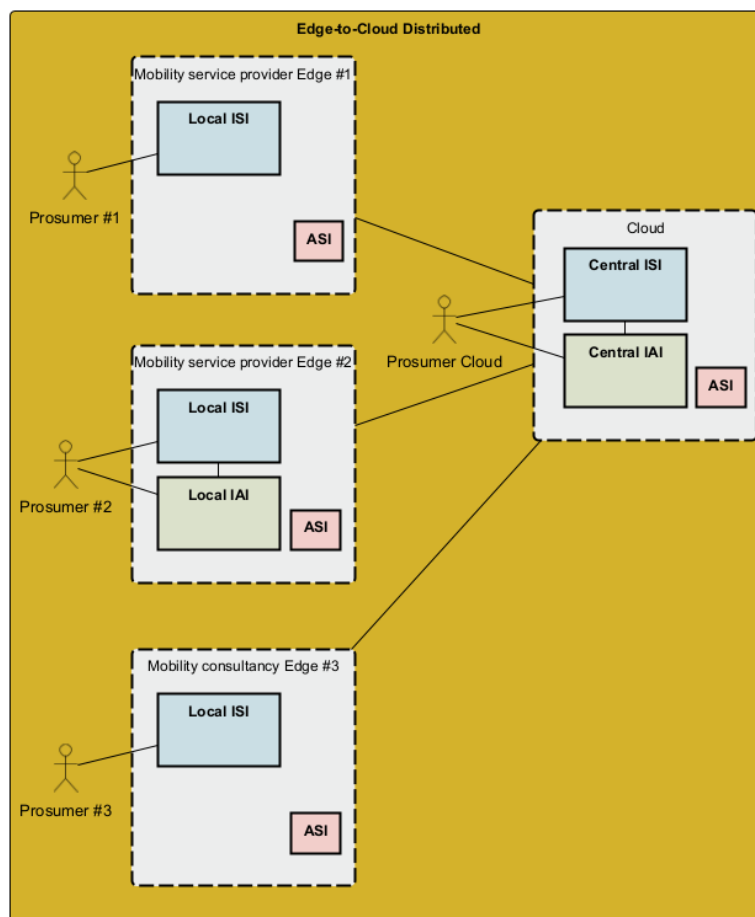


Figure 8: E-CORRIDOR instance for S2C pilot

3.4.3. Pilot ISAC

The goal of the ISAC pilot is to help critical transportation infrastructure owners and operators to protect them from cyber and physical security threats. To this end, the ISAC collects, analyses, and disseminates cyber-threat information to its members and provides tools useful to mitigate risks timely and enhance resiliency. Multiple stakeholders with different capabilities can collaborate, in a different way, to the cyber-threat information collection. The ISAC can gather information from public sources, e.g., public security databases or online information, or directly from the private transportation sectors. These last can collaborate to increase the ISAC knowledge sharing raw information extracted from its application environment or share its internal security analysis results. Based on this fact, the ISAC interacts with three different types of E-CORRIDOR nodes. A *passive producer premise* has neither the ISI nor the IAI, but it interacts directly with the ISI API exposed by the ISAC. An *active producer premise* keeps an instance of the ISI and DSA Lifecycle Infrastructure (DLI) locally. The Local DLI defines DSA rules for the shared data before sharing them with the Central ISI located in the ISAC premise. In such a way, part of the policies will be enforced by the Local ISI, and the remote ISI will enforce additional policies through the Central DLI. Finally, an *active producer with analytics capabilities premises* has locally both the ISI and the IAI instance. As the active producer, it can enforce policy on its data locally before sharing it with the ISAC, and in addition, it can run local analysis exploiting its IAI component. In such a way, it can choose to share, with the ISAC, both its local collected data and the analytic results obtained by the Local IAI.

The ISAC architecture can thus follow the **mixed distributed** architecture presented in Section 3.3 of the E-CORRIDOR framework. The ISI can be deployed both in the edge and in the centralised ISAC infrastructure. Similarly, the IAI can be instantiated in the edge to perform local analysis, or each prosumer can interact with the ISAC centralised IAI for the analytic services. The representation of the ISAC architecture showing the interaction with the E-CORRIDOR framework is shown in Figure 9.

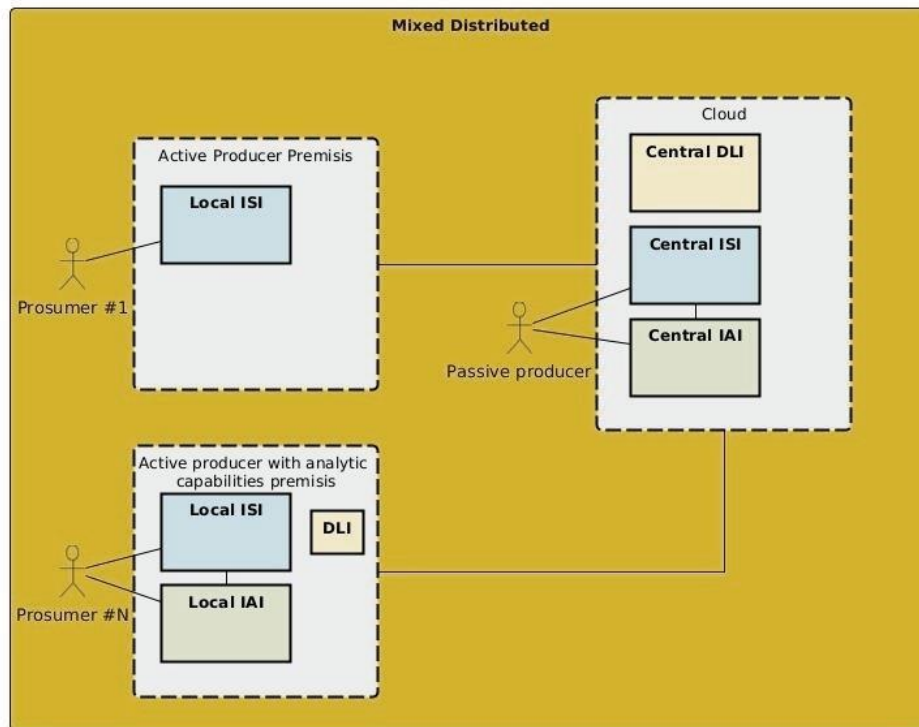


Figure 9: E-CORRIDOR instance for ISAC pilot

4. Subsystem: ISI - Information Sharing Infrastructure

The Information Sharing Infrastructure (ISI) allows Prosumers of the E-CORRIDOR framework to exchange data make using the rules defined in the Data Sharing Agreement are enforced. ISI also persists data to make it available for additional use cases, like the need to execute DSA-regulated analytics (see also Section 5).

This subsystem consists of the following components:

- **ISI API:** the entry point of the subsystem that provides the interface (API) to access the ISI functionalities;
- **Data Usage Control System (DUCS):** the component able to interpret and enforce the rules defined in the DSA;
- **Bundle Manager:** it implements the sticky policy concept by bundling together data with DSA rules to create a Data Bundle;
- **Bundle Store:** it maintains a persistent layer to securely keep at rest the Data Bundles;
- **Bundle Store Interface:** it is the communication interface to the Bundle Store;
- **Buffer Manager:** this component supports the analytics phase by creating ad-hoc data spaces that conform to the DSA rules;
- **Obligation Toolbox:** it keeps a set of functionalities used to enforce the obligation written in the DSA rules;
- **DMO Toolbox:** this component implements the Data Manipulation Operations (DMOs) that can pre-process the data content according to the DSA rules.

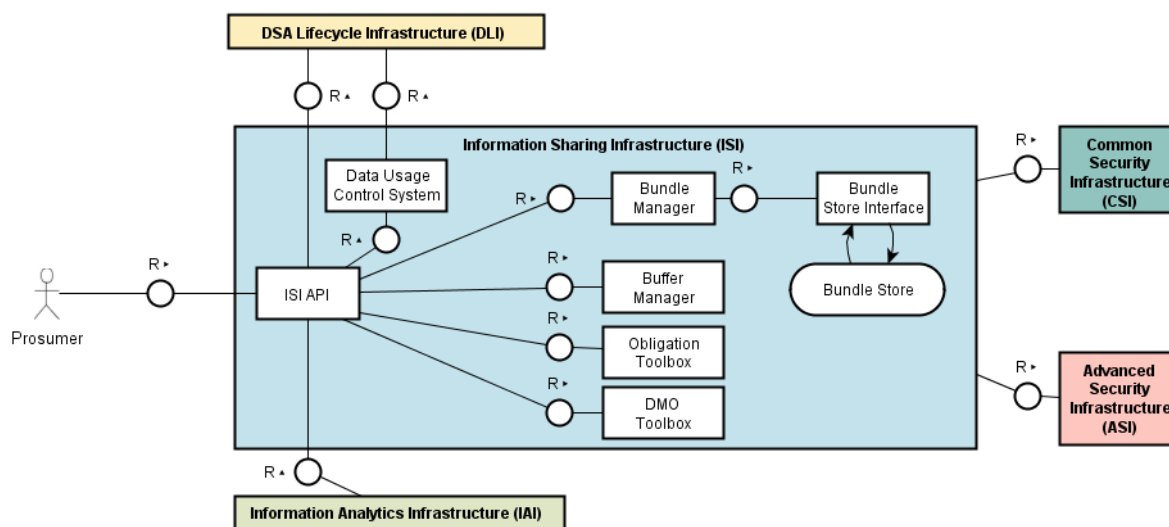


Figure 10: Information Sharing Infrastructure

The Information Sharing Infrastructure have interactions with the other E-CORRIDOR subsystem, in particular:

- The Prosumer uses the ISI API to both submit data to the ISI and to get it when needed;
- The DSA Lifecycle Infrastructure is contacted to retrieve the DSA under which the data has to be stored. It is also used to fetch DSA attributes to evaluate specific DSA validity rules (see DSA API operations described in Section 6.2);
- The Information Analytics Infrastructure contacts the ISI API to ask for the set of data on which the analytic functions need to execute;

- The Advanced Security Infrastructure provides services that use the ISI for their execution;
- The Common Security Infrastructure is used to maintain the audit trails of its activity, to gather user attributes for DSA rule evaluations and for signing and encrypting the data in the Data Protected Object.

As we anticipated previously, one of the key concepts of the ISI is the ability to implement the *sticky policy* paradigm. The implementation we foresee in E-CORRIDOR, inherited by the C3ISP EU project [3], is to tightly link the DSA rules that express the security logic to be applied to the data, with the data content itself. In this way, we create a **Data Bundle** (or **Bundle** - for short) that contains what we call a **Data Protected Object**, or DPO. Figure 11 illustrates graphically the design for the implementation.

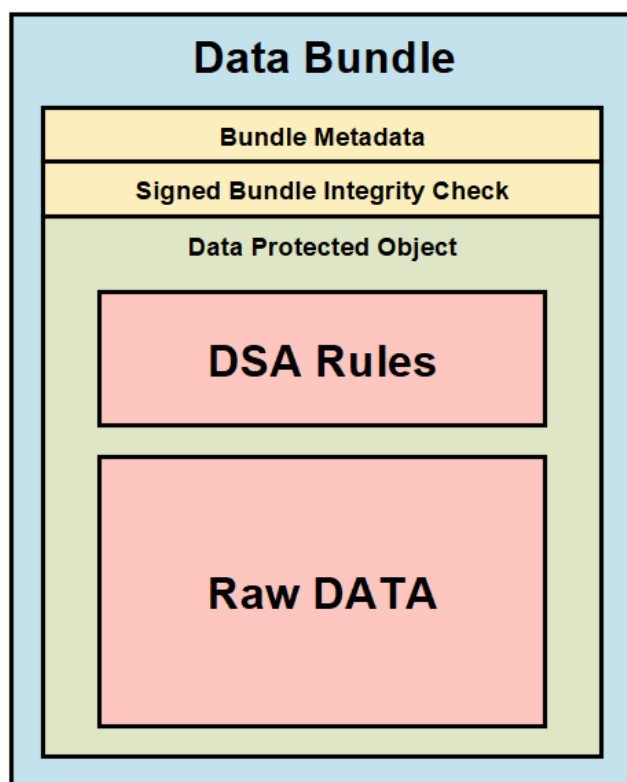


Figure 11: Data Bundle

The Bundle is structured as follows:

- **Bundle Metadata:** this object contains public metadata that expresses the properties of the whole object. These properties include:
 - **Size:** the length of the whole bundle (e.g., in bytes);
 - **Time of creation:** the date or time range related to the raw data (e.g., if the data contains multiple records this could be the start and end time of the records);
 - **DSA Identifier:** the identification key of the DSA used;
 - **Tags (optional):** a list of labels that can be used to index and search bundles that matches some criteria;
- **Signed Bundle Integrity Check:** a cryptographic code that guarantees that the whole bundle has not been corrupted either by chance or maliciously. The used security mechanisms are provided by the Common Security Infrastructure (CSI) subsystem;

- Data Protected Object (DPO): this is secure - optionally encrypted - object. It guarantees the confidentiality (if encrypted) and integrity (see Signed Bundle Integrity Check) of the shared data and contains:
- The DSA Rules in enforceable format, i.e. ready to be used by the Data Usage Control System;
- The Raw DATA, i.e. the data object shared by the Prosumer.

The ISI component responsible for packaging the Bundle is the Bundle Manager.

4.1. ISI API

The ISI API component is the frontend of the ISI subsystems and it exposes to the E-CORRIDOR prosumers the interface to invoke the data sharing operations on Data Bundles, namely, *Create*, *Search*, *Read*, and *Delete*, as well as to transfer Data Bundles from an ISI node to another ISI node, namely, *Transfer* operation. It also provides an API to *Search* for data bundles that match specific criteria based on condition against the bundle's metadata (acting as a proxy to the Bundle Store Interface search API, see 4.5). Moreover, the ISI API also exposes a further interface that is used by the IAI subsystem, called *Prepare Data Lake*, to prepare a repository (see Section 5.6) with the data of a given set of Data Bundles. The *Prepare Data Lake* interface is invoked by the IAI subsystem to ask the ISI subsystem to prepare a set of data necessary to perform given analytics. For each operation to be executed, the ISI API is the component that manages the operation workflow, i.e., it properly invokes the internal components of the ISI subsystem to carry out the requested operation. Being the frontend of the ISI, the ISI API also oversees the authentication of the user requesting to perform operations on Data Bundles.

4.2. Data Usage Control System

The Data Usage Control System (DUCS) is the component of the ISI subsystem which regulates the data sharing by enforcing the DSAs paired to the data (expressed in executable format, i.e., exploiting the UPOL language - see Section 6) deciding whether a given operation can be performed by a given user on a given (set of) Data Bundle(s) or not. In particular, the Data Usage Control System follows the Usage Control (UCON) model [21], which extends the traditional Attribute-Based Access Control (ABAC) model [22] by introducing Environmental Conditions and Obligations as additional decision factors besides Authorization, as well as the continuity of the policy enforcement to deal with changes in the access context that could occur while the access is still in progress. Hence, the DSA paired with a Data Bundle includes Authorizations rules defining constraints on the values of the attributes paired with users and data, as well as Conditions rules expressing constraints on the values of attributes representing the environment (e.g., date and time). Since each E-CORRIDOR pilot has its own set of users, data, and environmental attributes, the Data Usage Control System must be configured to interact with all the providers of such attributes, called Attribute Managers.

Obligations are actions that are executed because of the access decision. DMOs are a specific kind of obligations that are executed on the data embedded in the Data Bundle before being released to the requesting user or before being used for the execution of an analytics. For instance, an anonymization operation that deletes all the target IP addresses from a text file could be an example of DMO that can be executed on service logs before being analyzed to find DDOS attacks. Each E-CORRIDOR pilot defines its own set of DMOs, which depends on the type of data involved in the pilot analytics and on the privacy requirements. The Data Usage Control System invokes the execution of DMOs through the Obligation and DMO Toolbox component (see Section 4.7).

An innovative feature of the Data Usage Control System is the continuity of the DSA enforcement over time. This feature is exploited in the case of long-lasting analytics, to check that the right to execute them on the selected Data Bundles remains valid during the execution of the analytics themselves. This means that an analytics could be interrupted while in progress because the user who invoked it has lost the right to execute it on (at least) one of the involved Data Bundle. In this case, the Data Usage Control System will interact with the Service Usage Control System through the IAI API to notify that the execution of that analytics must be interrupted because of a DSA violation.

Moreover, since the requests sent to the Data Usage Control System typically concern a set of Data Bundles (instead of a single one), another relevant novelty introduced by the Data Usage Control System is the enforcement of rules which take into account all the Data Bundles involved in the same request to decide whether each of these Data Bundles can be used for the execution of the requested analytics. For instance, a data producer could write a policy that states that his or her data cannot be used to execute an analytics which involves the data produced by a given competitor organization.

4.3. *Bundle Manager*

The Bundle Manager component of the ISI performs two different operations, i.e., packing and unpacking of Data Bundles. It also communicates with the Key & Encryption Manager of the Common Security Infrastructure (CSI) in order to encrypt data provided by the stakeholder and with the Bundle Store by invoking its API in order to store data bundles. Furthermore, depending on the corresponding DSA and criticality of the data, the Bundle Manager may encrypt the data or directly invoke Bundle Store Interface API to store that data without encrypting it. As mentioned in D6.1, the Bundle Manager supports multiple encryption models, when the encryption key is used for one data bundle, one organization or for multiple objects provided by the prosumer. Finally, during the unpacking process, the Bundle Manager decrypts the data bundle and retrieves the corresponding DSA in order to send it to the Data Usage Control System (see Section 4.2) to evaluate the request and invoke DMO if necessary.

4.4. *Bundle Store*

To store data provided by prosumers as the data bundles, the ISI subsystem uses the Bundle Store. This component is invoked by the Bundle Manager that exploits Bundle Store Interface API (see Section 4.5) in order to store data bundles. As mentioned in D6.1, the Bundle Store can be implemented in two different ways, depending on the available resources. In particular, the Bundle Store could be implemented using a Hadoop Distributed File System (HDFS) [8] or it can be implemented as a local file system. For instance, the Central ISI deployment model is suitable for big data processing, including big data analytics, and consequently, the Bundle Store could be implemented using a Hadoop Distributed File System. Local implementations of the ISI used for small datasets, instead, could implement the Bundle Store as a local file system.

4.5. *Bundle Store Interface*

Bundle Store Interface is implemented as an API that is used to enable ISI to manage the storage of data bundles by invoking the Bundle Store component. Currently, the Bundle Store Interface API provides four functionalities, including *Create*, *Read*, *Delete*, and *Search*. These functionalities allow stakeholders to create a data bundle that maps raw data with the associated DSA file, also adding other relevant information as metadata. Furthermore, the Read and Search operations enable users respectively, to retrieve a Data Bundle by providing the corresponding ID, and to retrieve the list of the available Data Bundles according to a query on metadata, such

as the type of the event, the name of the organisation that owns the dataset, and so on. Finally, the Delete operation enables entities to remove Data Bundles from the Bundle Store.

4.6. Buffer Manager

The Buffer Manager component of the ISI subsystem manages *Data Lakes*, which are areas used to store data before and after analytics executed by the E-CORRIDOR platform. Data lakes are transient, since they are created and used only for the specific analytics execution. Therefore, those data lakes are removed as soon as the execution of the related analytics is over.

4.7. Obligation Toolbox

The Obligations toolbox is the component of the ISI subsystem in charge of the execution of Obligations. As previously described, obligations are actions that must be executed by the Data Usage Control System, and they do not involve the data embedded in the Data Bundle. A typical example of Obligation could be “send a notification via email to a given email address”.

Each pilot has its own Obligations, depending on its specific use case. Hence, to be able to host the different Obligations defined by the pilots, the Obligation Toolbox works as a kind of proxy which forward the obligation invocation to the component that is in charge of actually executing it. In particular, the obligations will be implemented either by Java libraries, or as RESTful services, and the Obligation Toolbox exposes a unique API, *Execute Obligation*, which, depending on the values of invocation parameters, will invoke the right library function or the service implementing the requested Obligation.

4.8. DMO Toolbox

The DMO Toolbox is in charge of executing the DMO on the data embedded in the Data Bundle as a consequence of the DSA evaluation. Typically, the DMOs are meant to anonymise or delete some information from the data stored in the ISI before making them available to the requesting user or for the execution of an analytics. Each pilot has its own data format, its own data privacy requirements and, consequently, defined its own DMOs, in order to perform the most proper operations to preserve the privacy of the specific data they store in the ISI subsystem. The DSA could state to execute different DMO on the same data depending on several factors (always represented through attributes), such as the requesting user, the analytics to be performed. The DSA could also state that a sequence of DMOs must be executed to prepare the data for given analytics.

To be able to host all the DMOs defined by the pilots, the DMO Toolbox works as a kind of proxy which forwards the invocation of a specific DMO to the component actually devoted to executing it. In particular, the DMOs will be implemented as Java libraries or as RESTful services, and the DMO Toolbox exposes a unique API, *Execute DMO*, which, depending on the values of its parameters, invokes the specific library function or service implementing the requested DMO.

5. Subsystem: IAI - Information Analytics Infrastructure

The Information Analytics Infrastructure (IAI) is the subsystem that enables the E-CORRIDOR Prosumers to execute analysis services over the data collected and shared by the ISI subsystem. The IAI permits the execution of E-CORRIDOR-aware analytics services that integrate and obey the sharing rules defined in the DSAs. We also foresee having analytics services, called legacy, that already exist and cannot be updated (e.g., because they are commercially closed, with no possibility to be updated). The legacy analytics can These analytics will work on the data prepared on the basis of the sharing DSA rules, but will not be able to exploit the full protection features of the E-CORRIDOR framework (e.g., they will not be able to use the Usage Control mechanism, provided by the Data Usage Control System).

The IAI has the following components:

- IAI API: the entry point of the subsystem that provides the interface (API) to access the IAI functionalities;
- Service Usage Control System (SUCS): it implements a security layer to protect the usage of the exposed analytics services;
- Analytics Orchestrator: it can coordinate the execution of the analytics, ordering the sequence of analytics, if more than one, that has to be executed on a set of data;
- Analytics Toolbox: this component implements the services that perform the analysis on the data by fully integrating with the protection capabilities of the E-CORRIDOR framework. It provides an open interface to easily add new services;
- Legacy Analytics Engine: this provides a way to execute and integrate with analysis services that cannot fully integrate with the E-CORRIDOR framework;
- Virtual Data Lake: this component creates an on-demand data lake instance that lasts for the time needed for an analytics execution, showing only the needed data as per the DSA rules.

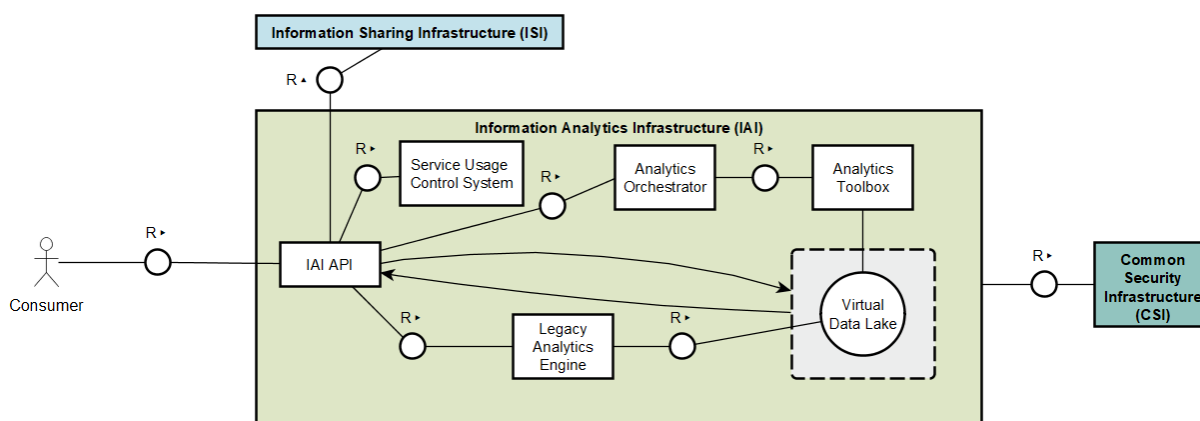


Figure 12: Information Analytics Infrastructure

The Information Analytics Infrastructure interacts have interactions with the other E-CORRIDOR subsystem, in particular:

- The Consumer (or an application on behalf of it), which would use the IAI API to execute analytics task over the data shared by the ISI;
- The Information Sharing Infrastructure to request the data for processing under the rules of the DSA;

- The Common Security Infrastructure is mainly used to maintain the audit trails of the IAI activity (e.g., analytics execution).

The next sections describe in more detail the IAI components.

5.1. IAI API

The IAI API component is the frontend of the IAI subsystem, and it exposes to the E-CORRIDOR users the interfaces to invoke the available analytics. Once a request is received, the IAI API invokes the Service Usage Control System to check that the invoking user is authorised to invoke such analytics. If the analytics execution is allowed, the IAI API interacts with the ISI to trigger the creation of a *Virtual Data Lake* (see Section 5.6) including the data available for the execution of the requested analytics. Then, the IAI API component invokes the *Analytics Orchestrator* which coordinates the actual execution of the analytics functions on the Virtual Data Lake. Eventually, an analytics could even be a composition of other analytics, and the execution of the related workflow is managed by the Orchestrator component.

When the result of an analytics is ready, the IAI API invokes the ISI subsystem in order to trigger the creation of a new Data Bundle including such a result.

5.2. Service Usage Control System

The Service Usage Control System is aimed at regulating the execution of the analytics by the E-CORRIDOR users. In particular, this component decides whether a given user can execute given analytics on the IAI subsystem. It is worth noticing that this decision process does not involve the evaluation of the DSAs paired with Data Bundles (that are meant to protect the data), since this task is executed by the Data Usage Control System. The policy enforced by the Service Usage Control System is instead aimed at protecting the IAI from unauthorised users. This policy is still written in UPOL (as the executable version of the DSA enforced by the Data Usage Control Service), and it is defined by the E-CORRIDOR service administrators who directly upload them on the Service Usage Control System. Since the Service Usage Control System follows the Usage Control Model as well, it needs to retrieve the current value of the attributes paired to users, resources and to access context every time the policy must be evaluated. Hence, the Service Usage Control System has been configured to retrieve the attribute related to the user from the same Attribute Managers used by the Data Usage Control System, since the users accessing the IAI are obviously the same accessing the data provided by the ISI. Instead, the attributes paired to resources at the service level are obviously different from the ones that are taken into account at the data level because the former concern the service, while the latter concern the data. Hence, specific Attribute Managers (AMs) are exploited by the Service Usage Control System for retrieving resource and access context attributes from the IAI service, such as the current workload, or the number of executions that are currently in progress.

Finally, because of the UCON model, the service level policy is evaluated continuously during the execution of analytics, and an analytics can be interrupted while in progress because of a policy violation. In this case, the Service Usage Control System will interact with the Data Usage Control System to notify that the Data Bundles involved in the interrupted analytics are not in use anymore.

5.3. Analytics Orchestrator

The Analytics Orchestrator is aimed at combining multiple analytics in a *workflow* to achieve advanced analysis. The composed service is seen by the DSA as a single additional data analytics and therefore the orchestrator can virtually define new data analytics.

As a matter of fact, the analysis workflows are constituted by the parallel and/or series composition of the analytics available in the analytics toolbox. For performance reasons and ease of use, the orchestrations can be defined in advance and stored in the IAI subsystem. In such a way the orchestrated services are made transparent and ready to use for the prosumer of the E-CORRIDOR framework.

All in all, the orchestrator needs to support a simple specification language either through configuration files or high-level programming language scripts. Such a language specifies how the composed components interact and present the output to the following stage. The orchestrator is also in charge of supervising the execution of the composed service and take corrective actions to re-establish its functioning in case of problems.

5.4. *Analytics Toolbox*

The Analytics Toolbox is aimed at exposing a set of *privacy-aware data analytics* components able to extract knowledge from the data shared in the ISI. The large scope of the multi-modal transportation entities in terms of security and authentication is reflected in a set of analytics components spanning from driver and passenger identification, to privacy-preserving itinerary planning and security analytics, to intrusion prevention and detection systems. The functioning of the components and their application to the pilots are detailed in the deliverable D7.1.

Each data analytics component in the toolbox is deployed in the IAI either as an executable Java file or as containerised service. The toolbox is aimed at supporting additional analytics than the ones identified by the pilots at the time of writing this deliverable. Indeed, to be *plugged* in the toolbox the data analytics component has to simply register itself in the IAI and, by implementing a simple interface it is made available in the E-CORRIDOR framework. Such a basic API is constituted by:

- **START:** to request the execution of the data analytics component over the pieces of data made available in the “virtual data lake” and prepared by the Buffer Manager component of ISI;
- **STOP:** to gracefully arrest the execution of the data analytics component;
- **KILL:** to interrupt the ongoing computation e.g., in case a policy violation is detected (this is to support the E-CORRIDOR usage control feature, see DUCS in Section 4.2);
- **END:** to notify the IAI subsystem that the component has completed its analysis and the data results have been passed to the ISI.

5.5. *Legacy Analytics Engine*

Besides supporting the execution of E-CORRIDOR aware analytics through the Analytics Toolbox component, as described in Section 5.4, the E-CORRIDOR framework also provides the support for executing already existing applications implementing analytics or functionalities of interest of the pilots. These programs are named Legacy Analytics Engines in our architecture, and they are programs that already exist on the market and that are not aware of the E-CORRIDOR framework. This implies, for instance, that the E-CORRIDOR framework cannot (gracefully) interrupt the execution of such programs in case a policy violation occurs while these programs are using the data provided by the ISI subsystem. These programs are executed outside the E-CORRIDOR framework control, either on the E-CORRIDOR premises or even on external premises.

The integration of Legacy Analytics Engines in our architecture involves two steps. Since these legacy programs will be invoked through the E-CORRIDOR framework, the first step concerns the integration in the IAI API component of the IAI. In particular, the IAI API should expose a

specific method for enabling E-CORRIDOR users to invoke each of such legacy analytics. The IAI API should be then able to trigger the execution of the requested legacy program according to its specific technical features (e.g., starting the execution of a program, invoking a local service, invoking a remote service, etc.). The other important integration step concerns the data provided as input to the legacy programs. As a matter of fact, the integration within the E-CORRIDOR framework requires that the ISI subsystem prepares a Virtual Data Lake including all the data to be processed by the legacy programs. However, each specific legacy program requires the data to be in a specific format. Hence, the Buffer Manager should be able to create Virtual Data Lakes having the formats required by each specific legacy analytic integrated in the IAI (Section 5.6 reports a number of formats for the Virtual Data Lake).

Finally, it is worth noticing that, being a Legacy Analytic Engine out of the control of the E-CORRIDOR framework, once the data are delivered by the ISI subsystem to such component (by sharing a reference to a properly formatted Virtual Data Lake), the legacy analytics could perform any operation on such data. For this reason, the DSAs paired with the Data Bundles should define very restrictive policies on the execution of legacy analytics (e.g., by defining proper DMOs deleting all the data fields that are not necessary for the execution of the specific analytics).

5.6. *Virtual Data Lake*

The Virtual Data Lake (VDL) is a transient storage space that is setup on-demand at analytics execution time. It contains the data, as retrieved from the Bundle Store, which forms the input of a specific analytics algorithm. The retrieved data is subject to the DSA rules, so the VDL might contain data that has been manipulated by a rules-decided DMO operation, like a data anonymization pre-elaboration of some data fields.

The VDL could be implemented with different technologies to address the analytics use case and expectations, such as:

- Filesystem: this is a regular file-based portion of the mass storage;
- Distributed filesystem for a large amount of data analysis: this is a specific file-based data organisation used for achieving high-throughput via distributed data nodes and to achieve at the same time data redundancy;
- No-SQL: this is a way of storing data structured in a non-relational database that facilitates some form of computation (e.g., document-oriented, key-value, graph-based, etc.);
- Data store capable of handling streams of data: at this stage, this is still to be assessed, but could be useful to elaborate streaming data like what comes from a video camera;

The VDL is programmatically built by the Buffer Manager component of ISI (see Section 4.6).

6. Subsystem: DLI - DSA Lifecycle Infrastructure

E-CORRIDOR Framework leverages the concept of Data Sharing Agreements (DSAs) to express rules about how to govern the information exchange and use between Prosumers. The DSA Lifecycle Infrastructure (DLI) subsystem takes care of managing the whole life of a DSA, starting from when it is authored between the different parties up to when the agreement expires.

The DLI is designed with the following components:

- DSA Editor: it is a tool that provides a graphical user interface to write the DSA rules with easy to understand language sentences;
- DSA API: this is the programmatic entry point to the subsystem for all the DSA management activities;
- DSA Mapper: the DSA rules from the Editor need to be compiled into a computer language that can be processed by an enforceable engine;
- DSA Store: it is the database where the DSAs are persisted;
- DSA Store Interface: it is the programmatic interface to the DSA Store.

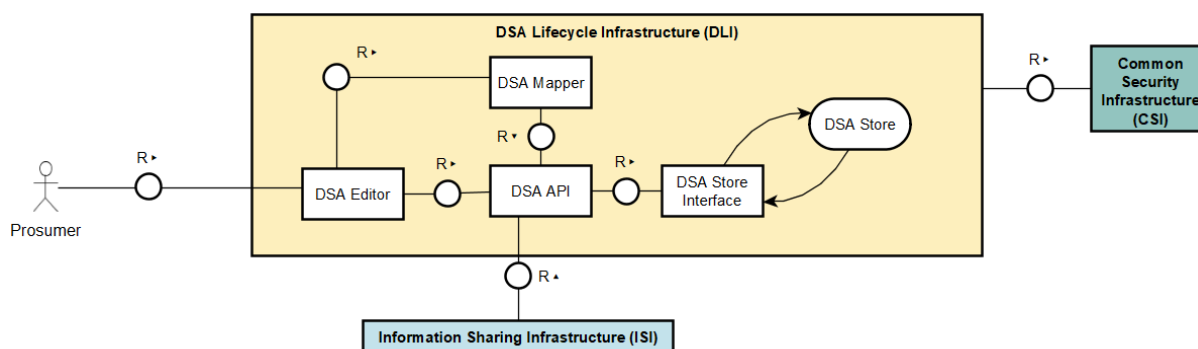


Figure 13: DSA Lifecycle Infrastructure

The DSA Lifecycle Infrastructure talks to other E-CORRIDOR subsystems, in particular:

- The Prosumer access the DSA Editor via a standard Internet browser to create the DSA and its rules;
- The Information Sharing Infrastructure makes use of the DSA API to fetch the DSA and the properties needed for the correct sharing of data;
- The Common Security Infrastructure provides services for audit trailing persistence and for identity management.

We have briefly introduced in the previous sections the notion of a Data Sharing Agreement as a document describing the rules that regulate the exchange of data between parties. Next list shows an example of some DSA rules:

```

IF a Subject hasRole Passenger THEN that Subject CAN access a Data
AFTER a Subject Create a Data THEN a System MUST Anonymise that Data
  
```

The language used to write the DSA rules is called Controlled Natural Language (CNL) [26] and we report its formal syntax is in D6.1. The semantics we give to the CNL makes use of Web Ontologies written in OWL [25]. Making use of such technique, we build a vocabulary of terms and specify how the terms can be composed to properly build a sentence, e.g. “a subject performs an **action** on an **object**.” These list of terms (actions and objects) is the vocabulary and it allows to model a specific data domain, like the specificities of the air or train transport

sector. In this way, the vocabulary defines how the Prosumers can write the rules that will appear in the Data Sharing Agreement.

It is important to say that the CNL is a high-level language that has to be transformed into a low-level language that can be easily enforced by a computer program, namely the Data Usage Control System (see Section 4.2). The DSA Mapper is the tool that transforms or compiles the CNL into the enforceable language, and the enforceable language is called UPOL [24]. UPOL is an extension made by the C3ISP EU project [3] to the standard XACML [23] policy language. Details about the UPOL language are reported in D6.1.

6.1. DSA Editor

The DSA Editor is the graphical user interface to the Data Sharing Agreements and is used by the Prosumers. It provides the functionalities to manage the DSA life-cycle, including:

- DSA creation and editing;
- DSA termination;
- DSA mapping (see 6.3);
- DSA persistence.

The DSA Editor is implemented as a web application. We foresee to have two main types of DSAs: a DSA template and a DSA instance (or simply called DSA). The DSA template is a reusable document of already written DSA rules: we can think about a set of DSA templates like an already available library that you can choose from, change and tune to fit your needs or simply use what it provides. However, a DSA template cannot be used as it is, but needs to be finalised in a DSA instance. To make this possible, we model the DSA document with several sections:

- DSA metadata section

This contains customisable fields such as:

- The DSA Identifier, the unique identifier of the DSA in the form of a UUID [6];
- The DSA Title, as a free-text string;
- The DSA Purpose, why the agreement is in place - this is useful in particular in some regulatory circumstances, like data collection under GDPR [7];
- The DSA Validity, start and end period when the DSA is effective;
- The Parties, which are the Prosumers that agree on the DSA “contract”.

- DSA rules section

This contains the agreement rules expressed in CNL. The rules can be of different types, like authorizations, obligations, and prohibitions. See D6.1 for further details.

Other sections might be added in the future, depending on the maturation of the E-CORRIDOR Pilots specification. The DSA document is modelled as an XML file.

By exploiting the Web Ontology technology used for the dictionaries, the DSA Editor can provide an interactive way to define a DSA rule, where the Prosumer is assisted step-by-step because the DSA Editor prompts to select only the terms suitable for the part of the sentence the user is writing (e.g., only the terms that can be associated to a specific action will be presented). This is possible because the ontology encodes all the terms' relationships.

The DSA Editor interacts with the DSA API to persist and retrieve the DSAs and DSA Templates from the DSA Store. It also interacts with the DSA Mapper to send the DSA for the compilation/mapping phase (see 6.3).

6.2. *DSA API*

The DSA Application Programming Interface (API) is the software contract used to interact with the DLI subsystem. It provides the functionalities that are needed to manage the lifecycle of the DSA, including its initial creation, the editing, and the deletion, as well as the possibility to retrieve DSA in both sources (CNL) and compiled (UPOL) form.

This component is used both internally within the DLI, by the DSA Editor (e.g., to create a new DSA) and DSA Mapper (e.g., to store a compiled DSA in enforceable format). It interacts with the DSA Store Interface (6.5) to persist the DSA on the DSA Store (6.4). It is also used externally by the ISI, e.g., to retrieve the DSA when creating a new Data Bundle.

The following interfaces have been considered to be part of this initial DSA API specification:

- **DSA CRUD (Create/Read/Update/Delete):** manage the DSA during its lifecycle, in particular allowing the operations of DSA creation, DSA read or retrieval, DSA modification and DSA deletion. The DSA creation returns a DSA identifier in the form of a UUID (Universally Unique Identifier) that is then used by the other operations as the DSA reference.

DSA enforceable rules management operations:

- **DSA UPOL CRUD (Create/Read/Update/Delete):** manage the enforceable rules of the DSA. These functions are expected to be used by the DSA Mapper which compiles the DSA rules from CNL into the enforceable XACML-based UPOL language. The UPOL part enhances the DSA, which will contain both the CNL and UPOL representations.

DSA status management operations are needed to work on the validity of the DSA and its validity status. D6.1 contains the details on the foreseen DSA states:

- **Read DSA status:** retrieve the current DSA state;
- **Read DSA version:** retrieve the current DSA version;
- **Revoke DSA:** change the DSA status to “revoked”.

6.3. *DSA Mapper*

The **DSA Mapper** is responsible for translating (or compiling) the DSA policies from the Controlled Natural Language (CNL) employed by the DSA Editor into a low-level directly enforceable policy language (XACML-based called UPOL). The DSA Mapper is called by the DSA Editor to translate the DSA policy before persisting the DSA into the DSA Store.

The DSA Mapper implements an interface that provides the following functionalities:

- **Support Pilot vocabulary:** it can learn new vocabularies to manage policies related to each E-CORRIDOR pilots;
- **Map DSA:** it takes as input a DSA (by its identifier), fetches the corresponding DSA and maps all the policies creating the mapped DSA (a DSA that contains both the CNL statements and the UPOL representation in a single XML file);
- **Build UPOL:** it takes as input a DSA (by its identifier) and returns the enforceable UPOL representation. This DSA representation is then used for pairing it with the data. This function is called by the Bundle Manager component, part of the ISI subsystem;

- Translate policies on Analytics Results: policies related to data derived from analytics functions are in a separate section of the DSA and the DSA Mapper translates them in a separate section too.

More details of the DSA Mapper are reported in Deliverable D6.1.

6.4. DSA Store

The DSA Store is where the DSAs are persisted once created by the DSA Editor. It uses a document-oriented database that allows to keep the DSA file in its XML format and to retrieve it as needed. The access to the DSA Store is mediated via the DSA Store Interface (see Section 6.5). In addition to storing the DSA file, the DSA Store contains also the DSA metadata to make it easy to search for DSAs that match specific criteria, like all the DSAs that involves a specific prosumer (party); see the description of DSA metadata in Section 6.1.

The DSA Store exposes its functionalities via the API provided by the document-oriented database, currently planned with a MongoDB implementation [5].

6.5. DSA Store Interface

The DSA Store Interface is a frontend interface to the DSA Store. It provides an interface that abstracts the document-oriented APIs of the DSA Store. In particular, it provides:

- DSA CRUD (Create/Read/Update/Delete): a CRUD interface to the document-oriented DSA Store interface;
- DSA Search: a functionality to query for DSA on the DSA Store that matches specific search criteria considering the DSA Metadata values.

The usefulness of the DSA Store Interface is to have a programmatic interface that is not strictly tied to the specific implementation of the DSA Store technology (currently a document-oriented database).

7. Subsystem: CSI - Common Security Infrastructure

E-CORRIDOR makes use of components to provide general security services. These are based on commercial off-the-shelf (COTS) products, specifically configured and customised to be integrated into the framework. Using COTS products help more seamless integration of the E-CORRIDOR framework in an already existing enterprise environment, facilitating its adoption as future exploitation of the project.

The CSI subsystem has the following components:

- Identity Manager: it provides standard user identification and authentication services and stores (or connect to) a user repository that contains users-defined attributes used for DSA rules evaluation;
- Key and Encryption Manager: it is dedicated to handling the cryptographic keys generation and secure storage, along with the implementation of cryptographic algorithms, also supporting the homomorphic encryption feature;
- Secure Audit Manager: it traces the key events generated during the E-CORRIDOR framework operations and helps to address compliance requirements.

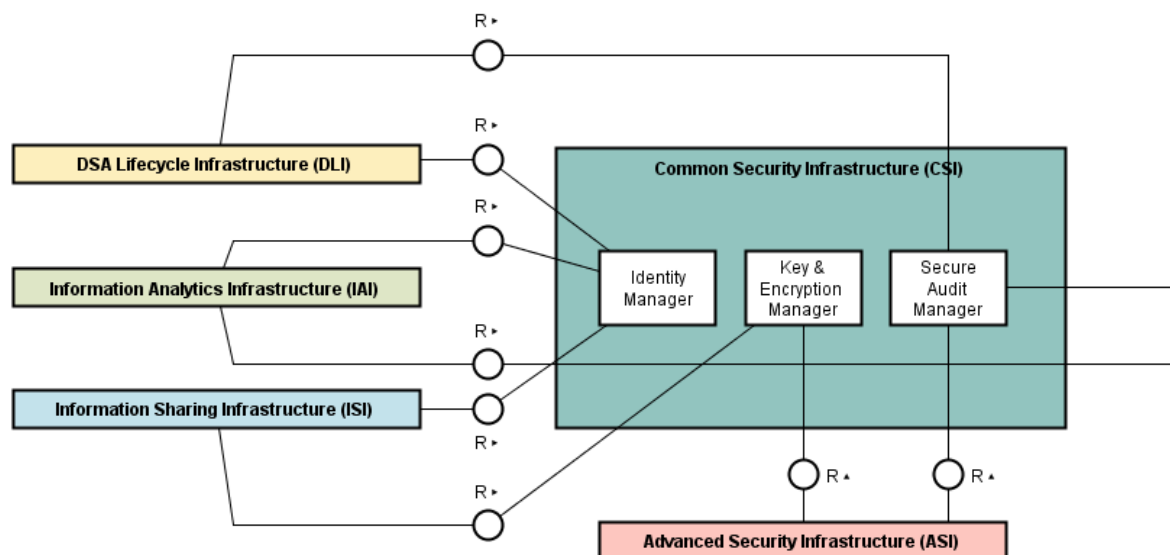


Figure 14: Common Security Infrastructure

The Common Security Infrastructure has interactions with the following subsystems:

- The DSA Lifecycle Infrastructure uses CSI for identifying users logging onto the DSA Editor, and for tracing its relevant activities;
- The Information Sharing Infrastructure uses CSI for identity-related purposes, user attributes evaluation, key and encryption functionalities (e.g. of the Data Bundle) and auditing purposes;
- The Information Analytics Infrastructure uses CSI for identity-related purposes and activity tracing;
- The Advanced Security Infrastructure uses CSI mostly for activity tracing and retrieving FHE public keys for security evaluation services.

7.1. Identity Manager

The Identity Manager aims at managing user identities and storing the information about users that are relevant in each pilot. Within the E-CORRIDOR framework, we envision a single Identity Manager which is able to verify the credentials of identities that access different services also for different purposes. The Identity Manager is used also as Attribute Manager by the Data Usage Control System and by the Service Usage Control System, since it provides the current value of attributes that can be used to decide whether to grant access rights. For instance in the ISAC Pilot a Prosumer must be authenticated to interact with the DLI subsystem to create and manage DSAs, and to the IAI subsystem to run analytics. Because of the different deployment models and configurations of the system, the Identity Manager has to be based on well-known standards in order to allow maximum flexibility and interoperability between components of the system.

The E-CORRIDOR framework uses an LDAP directory system in order to provide functionalities like authentication, group and user management. Exploiting the capabilities of OpenLDAP [30] directory server, the use cases configurations could be managed by a single server instance and could be differentiated using different subtrees one for each use case. More in depth an OpenLDAP authentication server will be configured for this purpose.

7.2. Key & Encryption Manager

Key and Encryption Manager is a component dedicated for key management and encryption services. These security services aim at ensuring the confidentiality of the shared data across the E-CORRIDOR architecture components. Precisely, this component manages two kinds of encryption techniques:

- The first ones are the “traditional” cryptography concepts. In the E-CORRIDOR platform, we use AES (Advanced Encryption Standard) [28] encryption technique. This cryptography concept encrypts all the *Data Bundles* that need to be stored in E-CORRIDOR platform. The process is reversible: the encrypted data can be transformed into plaintext if decrypted in the platform, knowing the key used in the encryption scheme. In fact, in cryptography, encryption and decryption processes use mathematic objects called *keys*.
- The second one is the Fully Homomorphic Encryption (FHE) concept: the data using this technique is highly ensured for privacy-preserving while processing, because homomorphic encryption can analyse directly on encrypted data without revealing its origins.

The first cryptography concept is deployed in the component called **Data Protected Object (DPO) – Key & Encryption Manager** (see Figure 16 - upper side). This module provides services for key management and encryption tools dedicated to securing the E-CORRIDOR data bundle (see the Bundle Manager description in Section 4.3). Effectively, it offers a management tool for secret keys with respect to *symmetric-key technologies* and encrypts sensitive data in the Data Bundle. The sensitive data are mainly the *Raw Data* and the *DSA Rules* (see figure on the right), thus it is encrypted before packaging and storing the bundle (see the Bundle Store description in Section 4.4).

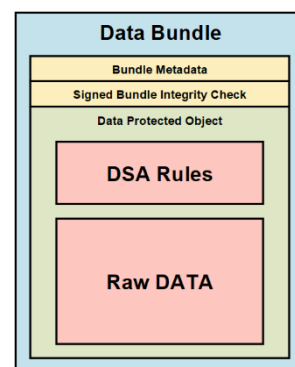


Figure 15: Data Bundle (see 4)

For the second cryptography concept, FHE technique, we propose **FHE – Key & Encryption Manager** (see Figure 16 - lower side). It offers services for FHE keys management, including

public & private key, *evaluation key*, and *transcription key* (also called *trans-ciphering key*, more next). The evaluation key corresponds to the algorithm by which the data will be processed.

Both cryptography concepts will be based on **Key Management System (KMS)** which is a design solution to manage and to store keys safely. It is an integrated approach for generating, distributing and managing cryptographic keys for devices and applications. It is tailored to specific use-cases. It includes the backend functionality for key generation, distribution, and replacement as well as the client functionality for injecting keys, storing and managing keys on devices. In Figure 16, the KMS function is carried out by the DPO/FHE Key Managers and the **Secret Vault** (i.e. the keys store), open source secret server from HashiCorp [31]. We favour an open source, well documented solution with software maintenance.

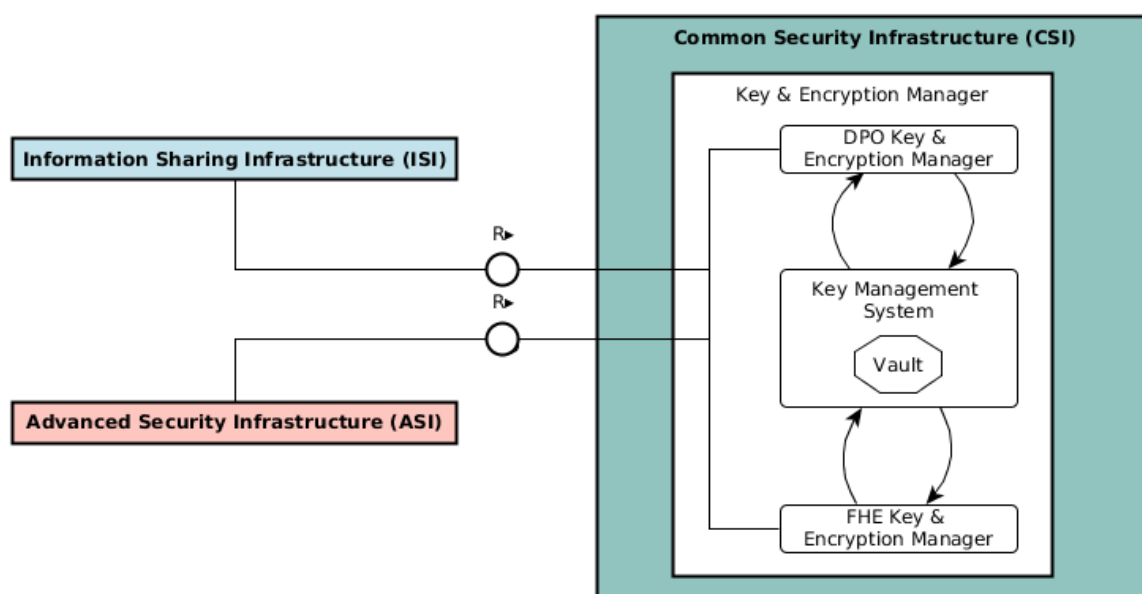


Figure 16: Key & Encryption Manager subsystem architecture

7.3. *Secure Audit Manager*

The Secure Audit Manager provides a service for tracing events that occur during the E-CORRIDOR framework operational activity. Event tracing - or logging - is a form of detective security control that is frequently required for being able to demonstrate:

- What happened, e.g. see if a DSA rule granted or not access to a protected resource;
- When it took place, e.g. to be able to demonstrate the exact date and time;
- Who did it, e.g. be sure to attribute responsibility of actions to a specific user individual;
- Where it took place, e.g. on which E-CORRIDOR node a data has been created;
- Why it happened, e.g. user was granted access because her request was for the DSA purpose.

This security control helps meeting compliance criteria, as it is often mandated by internal regulations and legal requirements. Such constraints provision also the properties the audit log trails should have, like the assurance of confidentiality, integrity, and availability. The data collection process should happen via secure channels that allow preserving confidentiality of log data in transport (e.g., encryption). It shall be assured that both communication channels

and log data storage provide data integrity capabilities (e.g., cryptographic checksum). Log data should also be persisted on a high-available repository that should not suffer data loss, as well as data in transport, should not be lost due to data collection failure, human mistakes, or other adverse events.

The Secure Audit Manager will make use of a front-end interface to the actual audit manager, which should be able to decouple the specific implementation technology. We have currently planned to provide:

- Web service interface (RESTful) via secure channel;
- Standard secure syslog implementation. We might employ syslog over TLS ala RFC5424 [32].

Both might be combined, for example, to create log data collection points on specific E-CORRIDOR nodes at the edge, which forward the events to a central E-CORRIDOR node at the cloud. The Audit Manager COTS software will be based on the open-source version of Graylog [33], a well-known log management tool.

8. Subsystem: ASI - Advanced Security Infrastructure

E-CORRIDOR Framework leverages the concept of Advanced Security Infrastructure (ASI) to evaluate authorisation and authentication mechanisms aimed at addressing the privacy concerns of users and customers about the disclosure of their personal data when accessing services (i.e. privacy-aware security mechanisms). ASI is responsible for managing security mechanisms to determine access levels or Prosumer privileges related to system resources including customer profiles, transport services, data and application features. This is the process of granting or denying access to a network resource which allows the Prosumers access to various resources based on identity features.

The ASI is designed with the components shown in Figure 17 and described next:

- ASI API: it is the Application Programming Interface that provides entry points to interact with ASI services;
- Discovery Security Services Manager: this component allows discovering the number of available services instances and their locations even if they change dynamically;
- ASI Orchestrator: this component is used to handle, route, and protect ASI components. It helps systems reduce errors, gain higher availability, and improve ASI resilience to failures. We design this component to reduce a low rate of errors at our scale to prevent degrading the experience for the E-CORRIDOR ASI subsystem, so every little bit helps;
- Privacy-Aware Interest-Based Service Sharing: it is a security service of ASI subsystem which offers a capability of profile matching to help customers from a country find the right services located in another country with similar attributes of interest sharing (e.g., interest, location, background, etc.);
- Privacy-Aware Authorization: it is a security service of the ASI subsystem which provides the features for protecting the privacy of the participating actors as a requirement by-design. This service allows sharing data without compromising actors' privacy and without disclosing out the degree of interest;
- Trusted Service Manager: this component provides trustworthy identities for various (edge) components of the E-CORRIDOR framework by utilizing (hardware-based) roots of trust. It may be used as a service provider for higher-level identity management or authentication systems;
- Privacy-Aware Seamless Multimodal Authentication: it is a security service of ASI subsystem, which aims to perform a privacy-aware multi-factor authentication (MFA) by exploiting both driver and passenger data;
- Continuous Behavioural Authentication: it is a security service of the ASI subsystem which offers a “*continuous and token-based*” authentication in a multimodal transportation domain.

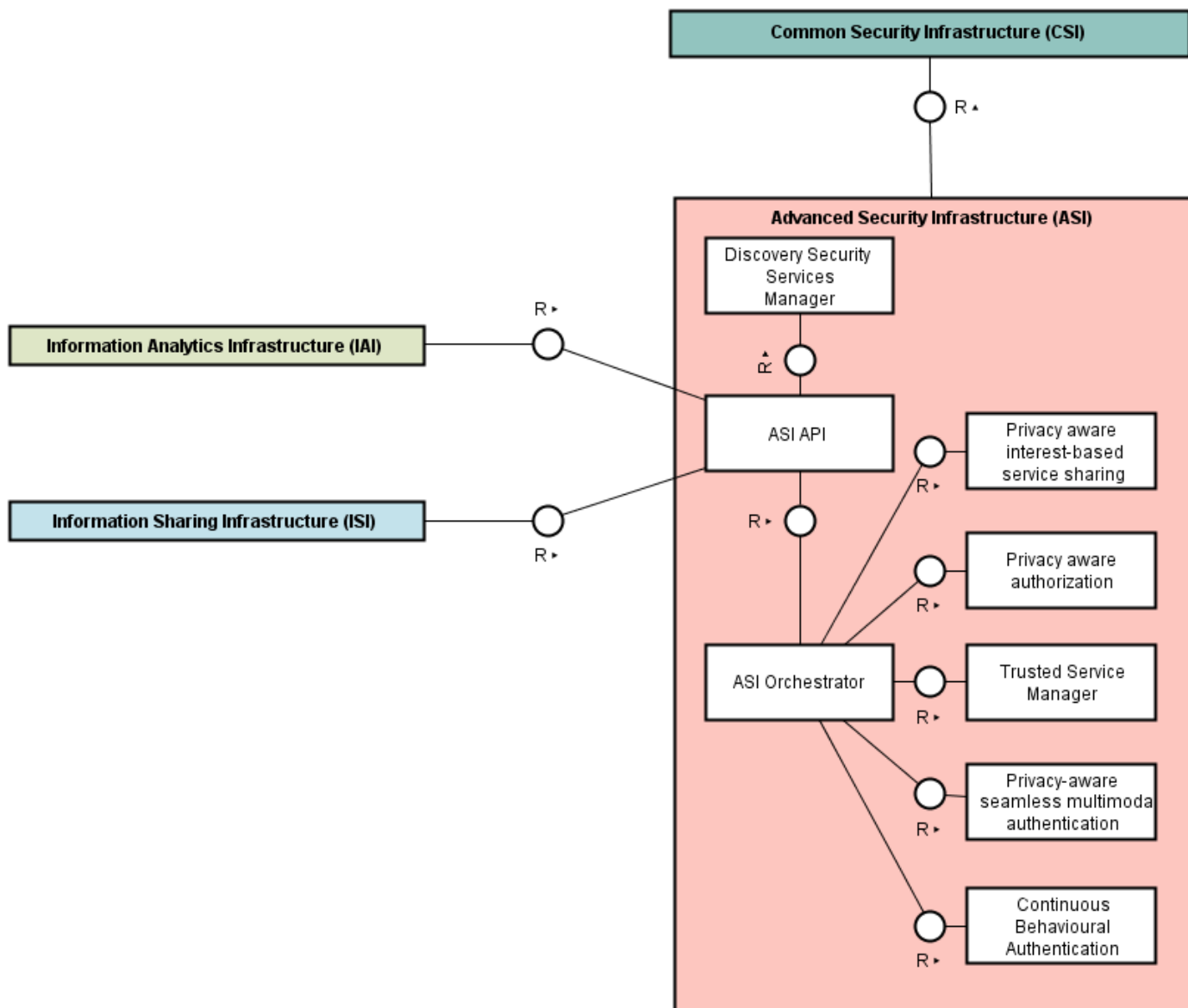


Figure 17: Advanced Security Infrastructure high-level architecture design

The ASI interacts with others E-CORRIDOR subsystems, in particular:

- The Information Sharing Infrastructure makes use of the ASI API to fetch the ASI security subsystem and the properties needed for the correct, authentication sharing of data;
- The Information Analytics Infrastructure invokes the features of the ASI API to fetch the ASI security subsystem and the properties needed for the privacy-aware authorisation and authentication of data access;
- The Common Security Infrastructure provides services for audit trailing persistence and for FHE key analysis.

Deliverable D8.1 describes in further detail the advanced security services used for the ASI subsystem.

8.1. *ASI API*

The ASI Application Programming Interface (API) is the software contract used to interact with the ASI subsystem. It provides the functions to handle the ASI subsystem. Precisely, this component realises a privacy-aware event handler that manages the setup of services, the deletion, the token authorisation and their validity, among others. This component interacts with Discovery Security Services Manager in order to check the availability of security and privacy services in ASI subsystem before invoking ASI Orchestrator component to redirect the request to the right service.

8.2. *Discovery Security Services Manager*

Services typically need to call one another. In a monolithic application, services invoke one another through language-level method or procedure calls. In a traditional distributed system deployment, services run at fixed, well-known locations (hosts and ports) and so can easily call one another using HTTP/REST or some RPC mechanism. However, a modern micro-service-based application typically runs in a virtualised or containerised environment where the number of instances of a service and their locations changes dynamically. Moreover, each instance of service exposes a remote API such as HTTP/RESTful endpoints at a particular location (host and port). Finally, the number of service instances might vary dynamically. For these reasons, when requesting a service, the ASI API obtains the location of a service instance by querying Services Registry in Discovery Security Services Manager, which knows the locations of all service instances.

8.3. *ASI Orchestrator*

ASI Orchestrator is used as the front door for all requests coming into ASI components. ASI Orchestrator significantly improves the architecture and features that allow handling, route, and protect ASI subsystems, and helps large-scale deployment when replicating multiple services of each feature in ASI and they can be hosted in the multi cloud or multi-cluster. That means you can deploy it in multiple contexts and have it solves different problems based on the configurations and filters it is running. Moreover, the ASI orchestrator allows leveraging load balancing, self-service routing, and resiliency features for internal traffic.

8.4. *Advanced Security Privacy-Aware Services*

8.4.1. *Privacy-Aware Interest-Based Service Sharing*

Multimodal transport across border services uses profile matching to help customers from a country find the right services located in another country with similar attributes (e.g., interest, location, background, etc.). However, privacy concerns often hinder customers from enabling this functionality. Some confidential customers' data faces the risk of hacking, leaking or exposing their personal information & location privacy. Based on this, we propose our Privacy-Aware Interest-Based Service Sharing, which allows customers to match their interest with others without revealing their real interest & profile and vice versa. To limit the risk of privacy exposure, only a minimum of information about the interest attributes of the users is matched with the prevention of real profile attributes. It is secure and gives a good degree of confidence in preventing hacking of users' profiles. For more details, please check Section 3 of D8.1.

8.4.2. *Privacy-Aware Authorization*

This service aims to design and develop services that have the privacy of the participating actors as a requirement by-design. Thus, analytics will share data without compromising actors' privacy. As an example, peers may share data based on their interests, and those ones will be

matched without disclosing out the degree of interest. For more details, please check from D8.1 Section 3.

8.4.3. Trusted Service Manager

The Trusted Service Manager provides services to establish secure identities, in form of cryptographic keys, in various (edge) components of the E-CORRIDOR framework. The trust in the identities is anchored in (hardware-based) roots of trust instantiations, e.g., by using the Trusted Platform Module 2.0 (TPM2.0) [29]. These root of trusts are utilized to provide trustworthy shielded locations for keys or other sensitive data like integrity values. The Trusted Service Manager may be used as a service provider for higher-level identity management or authentication systems, e.g., by providing them with trusted keys, tokens, and/or integrity data. For more details, please check Section 6 of D8.1.

8.4.4. Privacy-Aware Seamless Multimodal Authentication

This service performs a privacy-aware multi-factor authentication (MFA) by exploiting multi-biometric, behavioral, location and contextual information of the user (either driver or passenger). The MFA provided by the E-CORRIDOR core framework will allow a context-aware authentication able to assure privacy (through a token-based system) while enhancing the security of the authentication mechanism itself. The latter goal is achieved thanks to the use of multiple sources of information to identify and authenticate the users. For more details, please check from D8.1 Section 3.

8.4.5. Continuous Behavioural Authentication

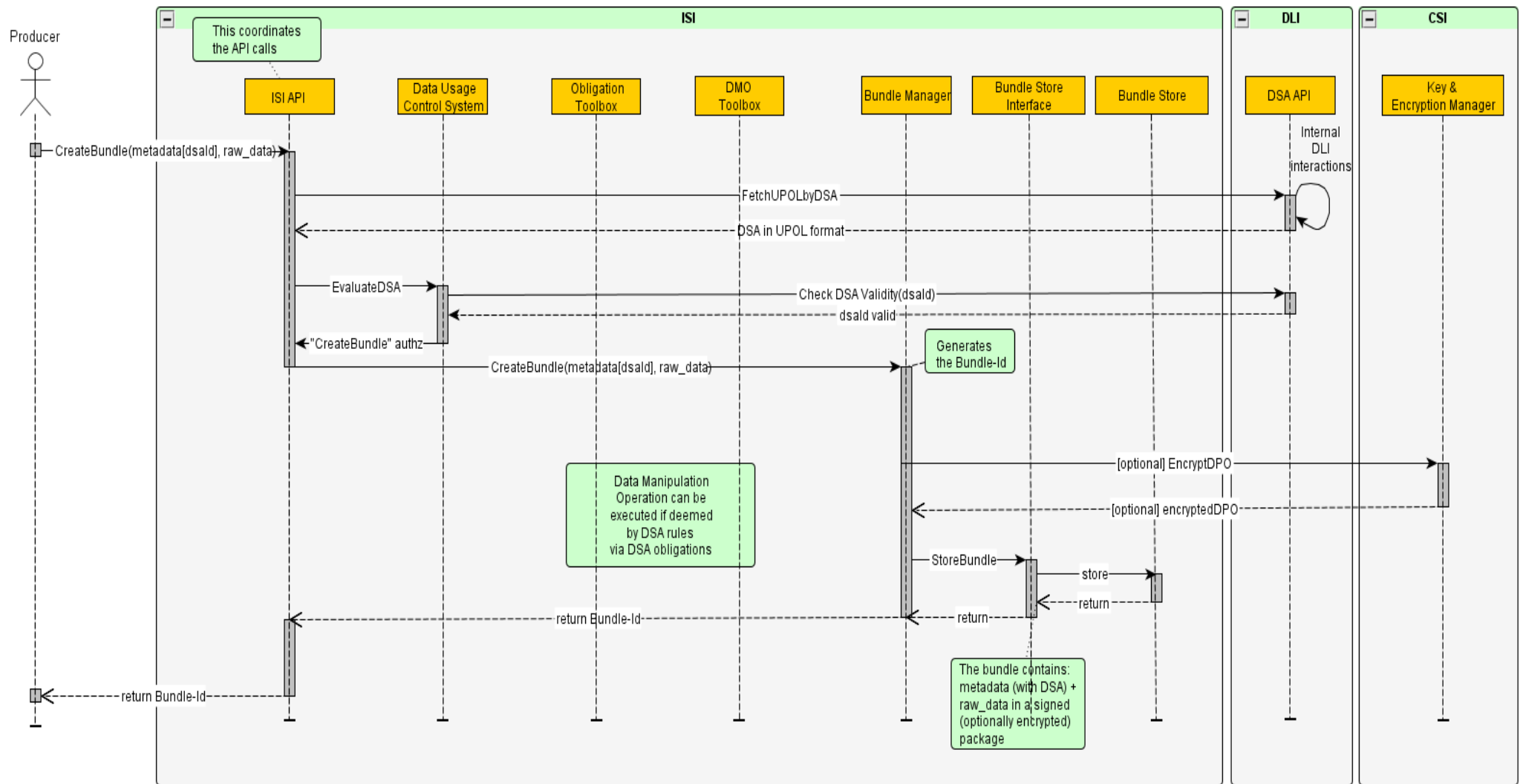
This service offers a “*continuous and token-based*” authentication in a multimodal transportation domain. In the foreseen scenario, each transport entity collects over time user information (driver in case of the S2C pilot, and passenger in case of the AT pilot) from the deployed sensors. In turn, the data analytics techniques (see the Analytics Toolbox in Section 5.4 and WP7) can build *behavioral fingerprints* (i.e., a token). While the user progresses in her journey and changes the mode of transportation (or simply moves to a different area), the E-CORRIDOR framework uses the information collected in the token to keep the passenger continuously authenticated with the transportation environment. More details are available in D8.1, Section 3.

9. Data Flow Diagrams

This section reports key E-CORRIDOR functionalities by describing the flows between the subsystems and components. The functionalities are shown with UML [27] sequence diagrams that illustrate the temporal sequence of interaction, the inner used functions and the most important input and output.

9.1. Create Bundle

One key function of the E-CORRIDOR framework is the **Create Bundle** operation. From a black-box perspective, this operation takes some raw data and stores it inside the framework under the conditions stated within the DSA rules. This flow is depicted in Figure 18.



This is the basic path description:

- The Producer submits raw data along with some metadata information (e.g., the identifier of the DSA to use for the Data Bundle) via the ISI API;
- The ISI API fetches a DSA (in its enforceable UPOL format) from the DSA Lifecycle Infrastructure (DLI) via the DSA API;
- The ISI API passes the acquired data to the Data Usage Control System, which internally checks whether the requested action is allowed by the DSA rules, including:
 - Check to see if the DSA is valid to use (see DSA validity mechanism in D6.1 for information);
 - Ask the Bundle Manager to create the Data Bundle;

Note: more details about the DUCS internal modules are found in D6.1.

- The Bundle Manager asks the CSI to (optionally) encrypt the Data Bundle's DPO (nota that the Bundle is always signed for integrity purposes);
- The Bundle Manager store the Data Bundle in the Bundle Store and finally the Bundle-Id is returned to the caller.

Notes:

- The Bundle-Id should contain also a reference to the E-CORRIDOR node where the data bundle is stored;
- The Obligation Toolbox and the DMO Toolbox are used in alternative paths of the described basic scenario: for example, if the DSA rules mandates for the application of a DMO anonymization function before storing the data, a specific obligation is triggered to the Obligation Toolbox which in turn asks for the specific DMO implementation to the DMO Toolbox. Several obligations and DMOs will be available and described in D6.1.

9.2. *Read Bundle*

Data Bundles stored on an E-CORRIDOR node can be retrieved by using a **Read Bundle** operation. When this operation is invoked for a specific Bundle-Id, the DSA rules are evaluated to see if and how the raw data shall be returned to the Prosumer. The sequence diagram in Figure 19 illustrates this workflow:

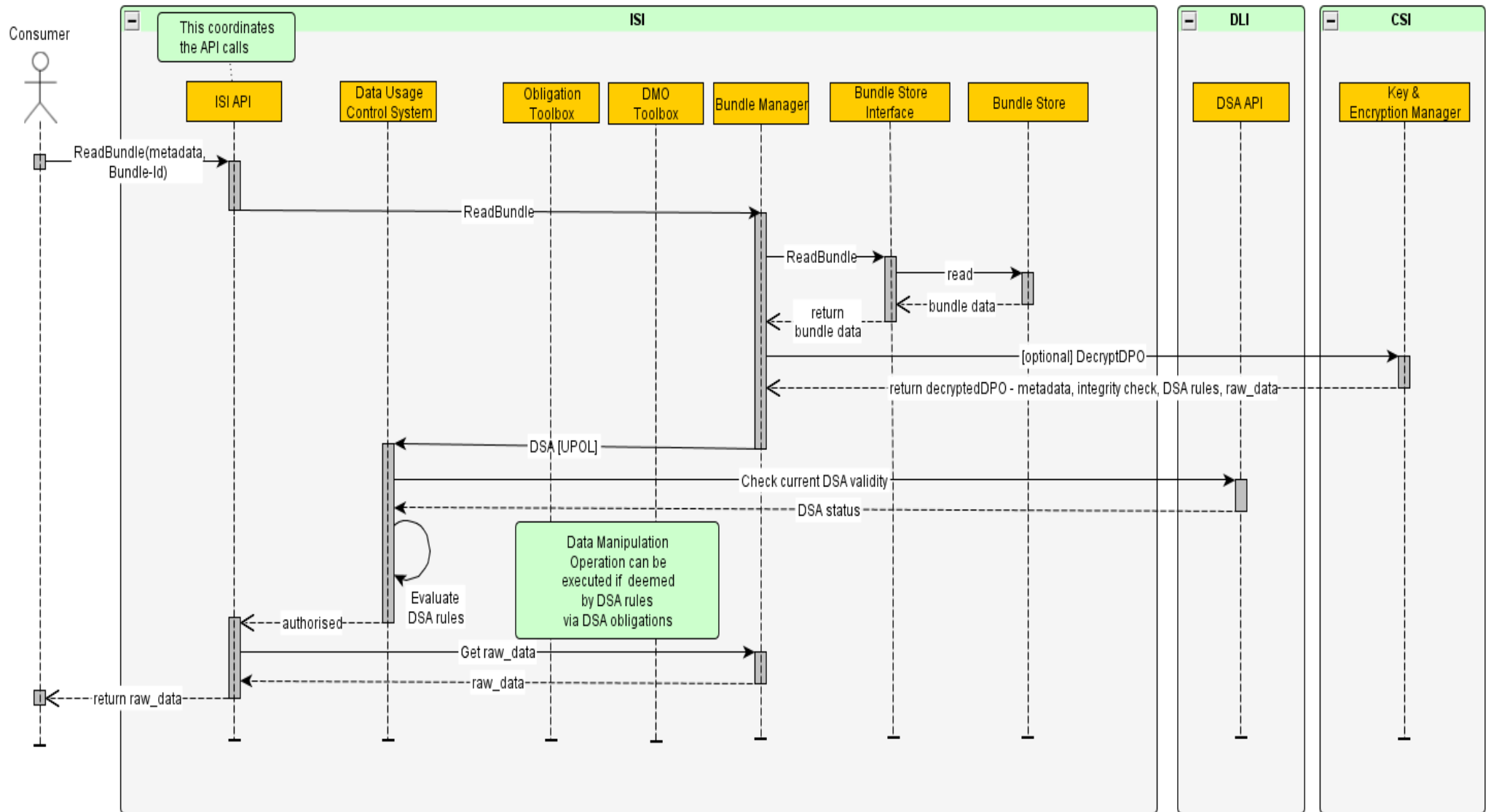


Figure 19: Read Bundle workflow

This is the basic path description:

- The Consumer wants to read a data bundle content (raw_data) via the ISI API by specifying the Bundle-Id;
- The ISI API first retrieves the DSA rules associated with the provided Bundle-Id:
 - The Bundle Manager fetches the data, possibly asking to decrypt it to the CSI subsystem;
 - The DSA in its UPOL representation is sent to the DUCS to evaluate it;
- The DUCS tests the DSA for its validity to understand if it can be used (see DSA validity mechanism in D6.1 for information);
- The DUCS evaluates the DSA rules. At this step, the DSA rules could mandates for specific DMO in the for obligation rules (see the notes of the Create Bundle workflow in Section 9.1);
- The DUCS authorises the ReadBundle and the raw_data is sent to the Consumer.

9.3. Delete Bundle

Data Bundles can be deleted by using a **Delete Bundle** operation. As creation and read operations described in the previous sections, also deletion has to be authorised by a DSA rule. The sequence diagram in Figure 20 shows this workflow.

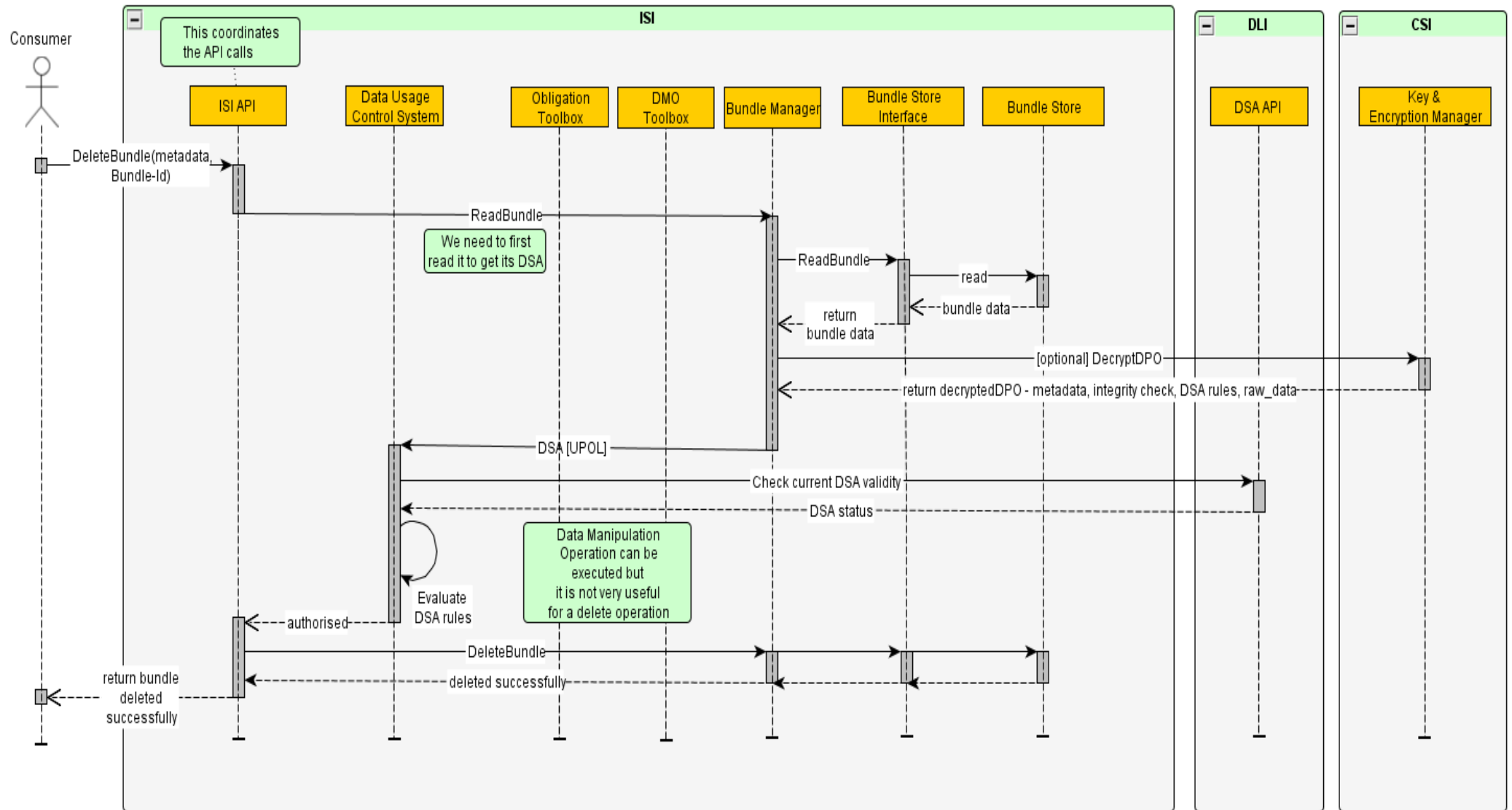


Figure 20: Delete Bundle workflow

This is the basic path description:

- The Consumer needs to delete a data bundle by its Bundle-Id via ISI API; as other operations, metadata representing the context/environment can be passed and these will be used by the DSA rules, if applicable;
- As per the Read Bundle, the ISI API first needs to get the DSA rules from the DPO's data bundle to test if authorisation is allowed (that is why it needs to first issue a ReadBundle internal operation);
- The DUCS checks if the DSA is valid;
- Then the DUCS verifies the DSA rules, including to see if the delete operation is authorised;
- The ISI API instructs the Bundle Manager to remove the data bundle from the Bundle Store;
- The result of the successful operation is returned to the Consumer.

9.4. *Transfer Bundle*

The deployment models described in Section 3 plan to distribute the E-CORRIDOR subsystems over a network. When placing multiple instances of ISI, which is in charge of allowing the share of information between the prosumers, the different E-CORRIDOR nodes must be able to transfer data, i.e. to move a data bundle out of an ISI instance for reaching another ISI instance. This transfer operation shall be subject to the DSA and there should be specific rules that permit or deny such action. During the transfer process, a specific DMO might also be applied, for example, to anonymise some data fields that the receiving party (the other ISI instance) shall not be able to see for any reason.

The flow is summarised with the diagram in Figure 21.

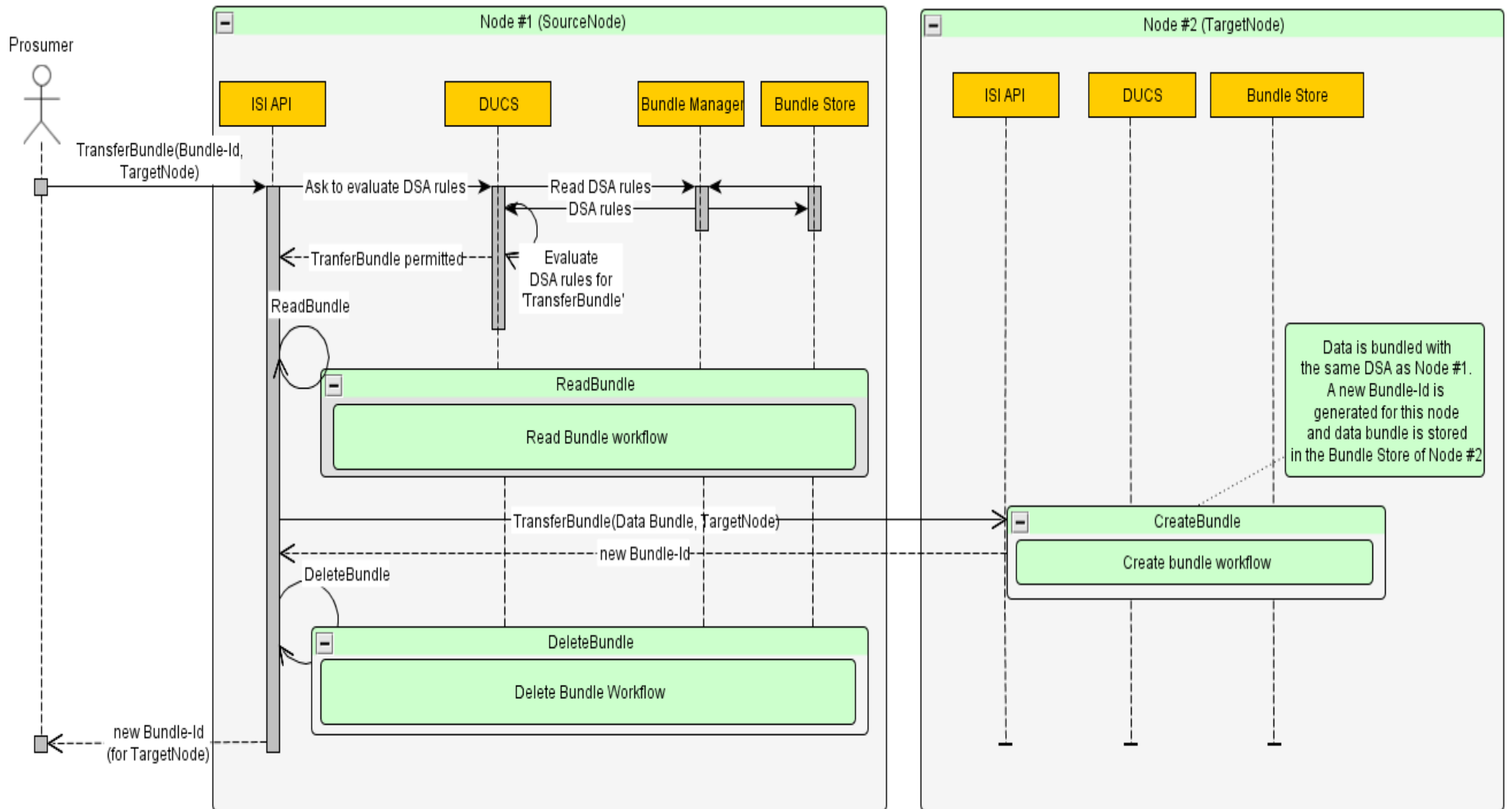


Figure 21: Transfer Bundle workflow

This is the basic path description:

- The Prosumer asks to execute a TransferBundle operation via the ISI API, to move the data bundle identified by Bundle-Id from Node #1 (SourceNode) to Node #2 (TargetNode);
- The Data Usage Control System (DUCS) evaluates the DSA to check for applicable TransferBundle rules, basically to see if and how the transfer process is allowed. For instance, a DMO could be specified in the DSA rules to be applied before transferring the data bundle to the TargetNode;
- ISI API retrieves the data and transfer the bundle to the ISI API of the TargetNode;
- On the TargetNode a new data bundle is created by using the same DSA of SourceNode, and the new Bundle-Id referring to the TargetNode is returned;
- On the SourceNode, the original Data Bundle is removed;
- Finally, the Prosumer receives the new Bundle-Id.

This workflow allows also for the implementation of a CopyBundle operation, which is a version of the TransferBundle that does not remove the data bundle on the SourceNode.

9.5. *Execute Analytics*

While the previous illustrated operations targeted the ISI API of the Information Sharing Infrastructure, the E-CORRIDOR analysis services that require to interact with the IAI. The analytics execution workflow has primarily interactions between the IAI and the ISI. At a high-level, the analytics execution requires the run of an Analytics Toolbox service. The IAI API requests the setup of a Virtual Data Lake for such a service that contains the raw_data extracted from the Bundle Store that matches both the analytics service specifications and the DSA rules that the bundles are subject to. Once the analytics service has been executed, the result is itself published to the ISI API and becomes a data bundle: such a “result” data bundle can be read by who is entitled under the statements of the DSA, like any other bundle.

The flow is described in the diagram in Figure 22.

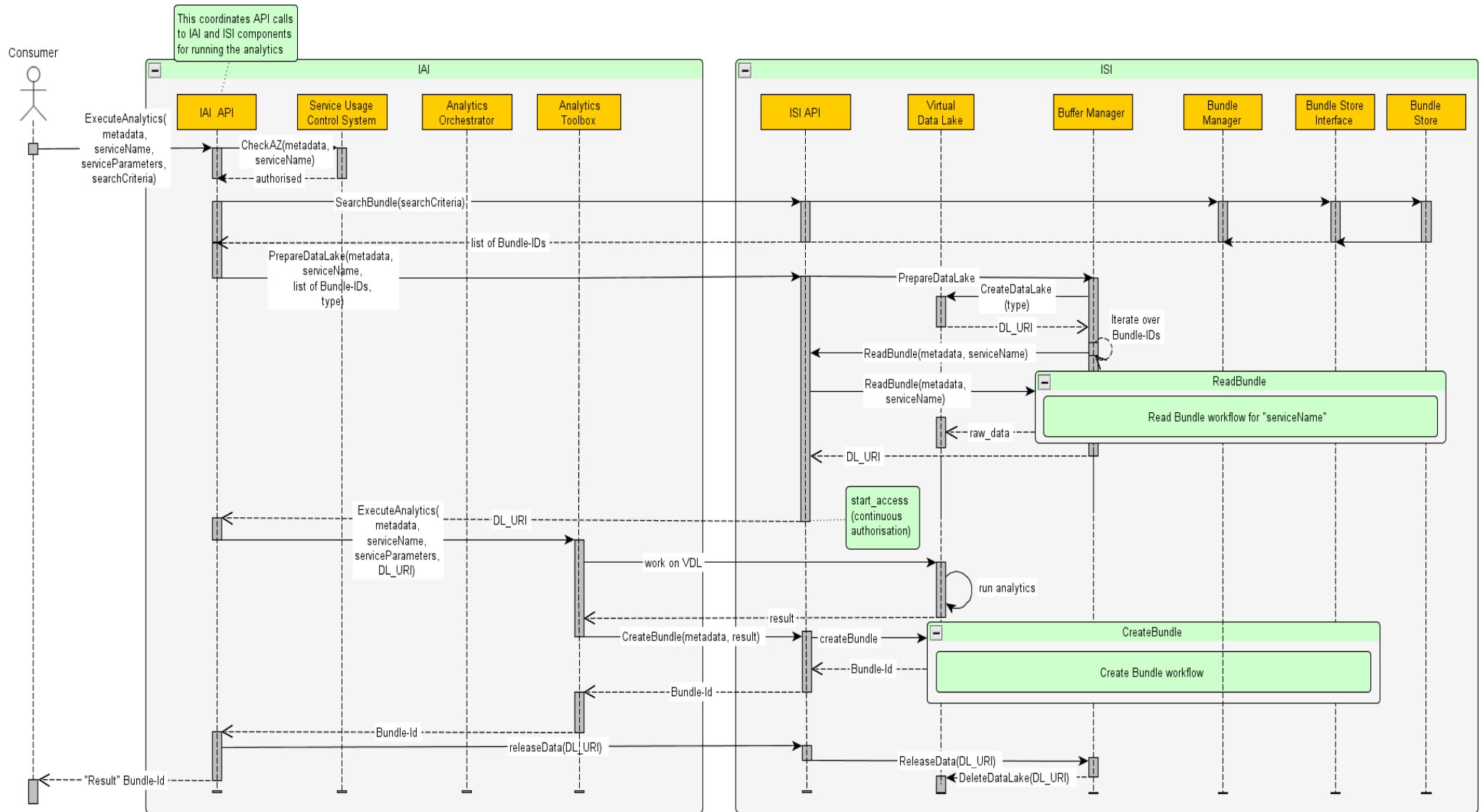


Figure 22: Execute Analytics workflow

This is the basic path description:

- The Consumer asks to execute an analytics service. The API requires the usual contextual metadata as other APIs, the name of the analytics and its parameters (serviceName and serviceParameters). A searchCriteria expresses a way to retrieve a list of data bundles that matches some conditions, like the bundles of a certain type create in a specific timeframe. The searchCriteria can also be a simple list of Bundle-IDs to use;
- The IAI API interacts with the Service Usage Control System (SUCS) to check whether access to the analytics service is allowed or not;
- The IAI API gets a list of Bundle-IDs that uses to ask for the preparation of a Data Lake. The Data Lake type will depend on the specific analytics (see Virtual Data Lake list in Section 5.6) and is where the analytics services (provided by the Analytics Toolbox) run;
- The Buffer Manager creates the Virtual Data Lake (VDL) and becomes then an ISI API client in charge of reading each data bundle in raw format, subject to DSA rules and possibly to DMOs, via the Read Bundle workflow described in Section 9.2. All the raw_data is saved into the just created Virtual Data Lake. As a small difference with the Read Bundle workflow, here the raw_data is saved directly to the VDL instead of returning it to the ISI API (this is for optimisation purposes);
- The returned Virtual Data Lake reference (DL_URI) is used by the IAI API to invoke the execution of the analytics service provided by the Analytics Toolbox (possibly, as an alternative path, managed by the Analytics Orchestration for more articulated analytics, see Section 5.3);
- The analytics execution create a new data bundle (via the Create Bundle workflow described in Section 9.1) which is finally returned to the Prosumer. The Prosumer will use the Read Bundle workflow to access it.

It is worth noticing that before the analytics execution, the usage control feature of E-CORRIDOR is triggered (start_access for continuous authorisation), which will be in charge of terminating the analytics if the authorisation conditions no longer apply (see Section 4.2).

We also consider having an asynchronous version of this workflow, where the ExecuteAnalytics returns a ticket which can then be pulled periodically to see if the result has been computed. We are still investigating other ways, maybe using other communication mechanisms (see Section 10).

10. Subsystems Communication

The requirements of the E-CORRIDOR framework have stressed the need for different message passing and communication patterns due to the peculiar characteristics of the multi-modal transportation pilots. Indeed, according to the specific functionalities and capabilities of devices and services, different protocols may be considered. Here, we just summarise some of the communication mechanisms that will be evaluated for integration in the E-CORRIDOR framework and reasons for considering them. In general, communication characteristics of the services, as well as hardware and software constraints should drive the choice of the communication subsystems.

The **REST-based communication** [10], is based on a *request-response* pattern where a client (consumer), initiating the interaction, requests a service to the server (e.g., a data analytics available in the IAI Analytics Toolbox or a function to store a data bundle via the ISI API) and receives an answer from the latter. The client context is not preserved, thus the communication is *stateless* (even if it is potentially *cacheable*).

This allows building high-flexible architectures and managing a wide set of requests. Such flexibility has brought the REST-based communication to be elected as the basis for container-based (micro-) services architecture (referred to as RESTful services). The same reasons have been considered for its adoption as the primary communication architecture in the modular E-CORRIDOR framework.

On the other end, in such a scenario, if the answer is not immediately available in the server a *pooling* mechanisms need to be used by the client. Basically, with such an approach, the communication is *synchronous*, with the value of the provided information inherently tied to the time and the client needs to periodically check the status of the requested service.

APIs written with the REST paradigm can leverage the OpenAPI standard [9], an open initiative that provides a way to describe REST-based functions via standardised descriptors (in the JSON format [20]) to facilitate system integration, testing and documentation.

In the **Event-driven communication**, the client specifies its interests on a specific topic, and then the server is in charge of reinitiating the communication once the requested piece of information is ready. This *asynchronous* communication realises the *publish-subscribe* messaging pattern. For example, in the E-CORRIDOR architecture, this communication would fit the requirements of the ISAC pilot, where transportation operators can register to feeds of interest and automatically receive customised push messages (e.g., *webhook*).

Streaming-oriented communication is a different flavour of the event-driven one where status updates are constantly dispatched to provide information temporally close to the originating event. In the E-CORRIDOR project, pilots (in particular AT and S2C) have expressed the need for transferring sensor data to the ISI e.g., camera feeds in the airport. By storing the contextual/state information or simply keeping the connection open (i.e., the service is *stateful*), timely status updates can be sent from the source to the interested destination through *event brokers*. This approach is less flexible, as usually there is only one (or a very few) supported data format(s) and acceptable requests, and fewer interactions are foreseen in the communication model. On the other hand, the scalability of event-driven communication shifts its focus from supporting a high volume of requests (as in the REST case) to support a large amount of data.

The **Internet of Things (IoT)-oriented communication** is yet another form of event-driven communication targeted to small and low-power devices which need to transmit telemetry data. In such a case the message size is generally small and lightweight protocol implementations are

considered to ease coding and decoding of the messages even in case of Machine-to-Machine (M2M) exchanges.

Table 3 sums up the features of the communication systems and the reasons for their potential interest and suitability in the E-CORRIDOR framework. Moreover, examples of communication protocols enabling the corresponding architectures are included.

Table 3 - Communication mechanisms adoptable in the E-CORRIDOR framework

Communication architecture	Messaging channel	Message pattern	Domain-Specific application	Applicability to the E-CORRIDOR framework	Main feature	Example of supporting protocols
REST-based	synchronous	request-response		Flexible RESTful services - main communication architecture	Scalable to a high number and type of requests	
Event-based	asynchronous	publish-subscribe		Notification on specific threats, events and changes of (contextual) status	Timely notification of status changes	
			Stream	Video feeds	Scalability to large amount of data	RTMP [11], RTSP [12][11], MPEG-DASH [13], HLS [14]
			Internet of Things (IoT)	Telemetry from sensors, Machine to machine	Low-power requirement, minimized data packet size and lightweight protocol	MQTT [15], AMQP [16], DDS [17], XMPP [18], CoAP [19]

It is worth noting that embedded systems and sensors may or may not support some of those communication means depending on the resources available (in terms of both hardware and software). For embedded and IoT devices, from an implementation point of view, it is generally preferable to use a specific solution usually recommended by the manufacturer.

At the time of writing this deliverable, the main communication architecture to be integrated with the E-CORRIDOR framework is the REST-based one thanks to its flexibility. But, considering the needs of the pilots, the other architectures will be considered and potentially tested in the future development of the framework.

11. Status of the Dev/Test/Prod environments

In deliverable D5.1, we planned to have three computing environments for supporting the E-CORRIDOR Framework implementation activities: Development, Test Bed and Production environments.

At this stage (M12), we have started configuration for Development Environment and in the next section, we report about it status.

11.1. Development Environment

The Development Environment hosts all the tools to create, store, build and evaluate the quality and security of the software artefacts developed for E-CORRIDOR.

In the following sections, we describe what has currently been prepared to match the requirements reported in deliverable D5.1 (section 4.1) split between infrastructure settings and software/tools configuration.




11.1.1. Infrastructure Configuration





The Development Environment is fully hosted in one Virtual Machine provided by CNR in its Data Center, in Italy.

E-CORRIDOR - Development Environment Configuration		
Number of Machines	Aim	Status
1 Virtual Machine	This machine hosts all the development tools (described in D5.1) used for the development of the E-CORRIDOR framework	Deployed

11.1.2. Software and Tools Configuration

In this section, we report the tools configured following our design described in D5.1 section 4.1.2. The following table summarises the current status:

E-CORRIDOR - Software Development Tools		
Category	Tool	Status
Collaborative Code & Version Control	GitLab 	Configured We have setup and preliminary configured the tool, integrating with the CNR data centre services (e.g., email sever, etc.).
Build Automation	Apache Maven 	To be configured This will be configured once we start to upload the source code to the GitLab versioning system.
Artefact Repository	Nexus Repository OSS 	Configured This is configured as a Docker container. It will be later tuned for the E-CORRIDOR components once they will be built via the Jenkins CI/CD pipeline.

Testing	JUnit (Unit)  RestAssured (Integration) 	To be configured These will be configured once we start the integration activities for the different E-CORRIDOR framework subsystems and components.
Bug Tracking	GitLab 	Partially configured The final configuration will be tuned once we will have the source code uploaded to GitLab.
Continuous Integration	Jenkins 	Partially configured We have just started preparing the CI/CD pipeline for a generic build job, but once the source code will be uploaded to GitLab, it will be extended to the E-CORRIDOR components as soon as they start to be deployed and integrated.

12. Contributions towards the E-CORRIDOR objectives

The E-CORRIDOR framework aims at providing a collaborative, privacy-aware and edge-enabled platform for information sharing, data analysis and security services to multimodal transportation domains.

The architecture of the E-CORRIDOR framework has been designed by taking into account the requirements (both functional and non-functional) identified by the three project pilots namely, airport-train (AT), smart city and car-sharing (S2C) and multi-modal information sharing and analytics centre (ISAC). All in all, the architecture contributes to the following E-CORRIDOR objectives (and here reported for the sake of completeness):

- Objective 1: E-CORRIDOR will build a flexible, confidential and privacy-preserving framework for managing data sharing, for several purposes, by different prosumers;
- Objective 2: E-CORRIDOR will define edge enabled data analytics and prediction services in a collaborative, distributed and confidential way;
- Objective 3: E-CORRIDOR will define a secure and robust platform in a holistic manner to keep the communication platform safe from cyber-attacks and ensure service continuity;
- Objective 4: E-CORRIDOR will improve, mature and integrate several existing tools provided by E-CORRIDOR partners and will tailor those to the specific needs of the E-CORRIDOR platform and Pilots;
- Objective 5: E-CORRIDOR will provide mechanisms for seamless access to multimodal transport;
- Objective 6: the framework and the services developed will be used to deliver three pilot products;
- Objective 7: E-CORRIDOR will be promoted and ease the exploitation, communication, standardization, dissemination and early adoption of its results.

The contributions of the E-CORRIDOR framework to the E-CORRIDOR objectives are as follow.

The data sharing based on contracts (i.e., data sharing agreements, DSAs) provides a way for multiple data prosumers to share their data in accordance with agreements formally specified. DSA can include restrictions to data access including specific data analytics and data manipulation operations. The Information Sharing Infrastructure (ISI) subsystem supporting DPOs (Data Protected Objects) is then in charge of *monitoring and enforcing the compliance to such DSAs* to satisfy the data ownership requirements of the original data producer. The DSA Lifecycle Infrastructure (DLI) subsystem allows the prosumers to jointly collaborate for the definition of these DSAs and the rules they express. These features of the E-CORRIDOR framework allow the successful achievement of *Objective 1*.

Many of the data managed by the project pilots involve sensitive data that could bring financial losses, violation of the laws and security issues if disclosed in an uncontrolled way. Nonetheless, the stakeholders representing the entities involved in the pilots have important opportunities to improve their operations, security capabilities and services for the users, by breaking their data silos and fostering collaboration. The E-CORRIDOR framework, thanks to its controlled data sharing (enabled by DSAs and the ISI subsystem), and the *privacy-aware pluggable data analytics* provided by the flexible Information Analytics Infrastructure (IAI) can provide a collaborative yet confidential platform for data prosumers. Here it is worth remarking that the *DSA can even restrict access to the data only to a set of analytics deemed*

trusted by the original data producer. Moreover, the *edge-enabled* features of the same *framework* can meet performance and regulatory constraints requested by the data prosumers with respect to data location. These features of the framework satisfy the Objective 2 of the E-CORRIDOR project.

Objective 3 of the E-CORRIDOR project aims at providing a cyber-secure and robust platform able to ensure service continuity. Through the Advanced Security Infrastructure (ASI) the E-CORRIDOR framework provides trusted service management, authentication and authorization capabilities based also on multi-biometrics, multi-factor and behavioural analyses. All these services can enhance the security of the pilot domains and securely grant access to the framework. Also, the IAI data analytics *provide intrusion prevention and detection services useful to strengthen the cyber-security of the pilot infrastructures*. Finally being edge-enabled the E-CORRIDOR *framework inherently provides a robust architecture without the presence of a single point of control and failure*. Each transport entity adopting the E-CORRIDOR framework can keep the functioning of its own components at the edge while sharing, thanks to the ISI, potential threats and vulnerability identified by the IAI analytics.

The IAI includes an analytics toolbox designed by following the requirements expressed by the pilot partners. The flexible architecture of such a toolbox allows the plug-in of further analytics (identified in the next months of the project execution or even after its end) by following a simple OpenAPI specification [9]. Some legacy components that do not support natively the E-CORRIDOR framework (or cannot be modified for that, such as commercial off-the-shelf products) can also be integrated into the IAI through the legacy analytics capabilities. Being the *analytics in the IAI and the security services in the ASI tailored according to the needs of the project pilots*, evaluation and maturation of the former in a realistic environment (corresponding to a Technology Readiness Level up to level 6 or 7) is expected. This contributes to the Objective 4 of the E-CORRIDOR project.

The E-CORRIDOR targets multi-modal transportation systems. In such domains, one key desiderata concern the frictionless passenger authentication while accessing multiple transportation domains and services seamlessly. This is expressed in the project through the E-CORRIDOR Objective 5. The IAI analytics toolbox provides a set of *passenger and driver identification and authentication capabilities* tailored to the specific needs identified by the project pilots (up to the time of writing this deliverable). The ASI privacy-aware *behavioural authentication and authorisation, and the secure identity management components*, also by leveraging on the analytics in the IAI, are able to provide *seamless and privacy-aware access to multimodal transportation and a frictionless experience for the user*.

The E-CORRIDOR project *pilots can express a representative cross-section of the transportation systems and their needs in terms of an integrated multimodal journey and cyber-security in the transportation systems sector*. The pilots in the project are indeed constituted by airport and train, car sharing and smart mobility. In addition, the multi-modal transportation ISAC is in charge of providing analysis capabilities on log data, and of collecting and dispatching threat and vulnerabilities information relevant for the transportation entities according to an interest-based service. The controlled data sharing capabilities of the ISI, the data analytics of the IAI, the authentication, authorisation and trusted service management of the ASI, and the features of the whole architecture of the E-CORRIDOR *framework are based on requirements and needs expressed by the pilots*. This will ease the test of the overall framework and the individual services and components in a realistic environment. The maturation of the corresponding technologies has thus the strength of *improving security and passenger experience in a pan-European multi-modal transportation scenario (Objective 6)*.

Experiences and maturation of the controlled data sharing technologies and their application in data analytics and advanced security services can potentially *support the definition of a standard for federated environments*. The fact that the same *technologies would be used for on-field tests by the pilots and the representativeness of the latter for the multi-modal transportation landscape will offer further opportunities for the exploitation of the E-CORRIDOR framework* (please refer to *Objective 7*).

Table 4 summarises the contribution of the E-CORRIDOR framework (with a focus on its key features) to the project objectives.

Table 4 - Salient features of the E-CORRIDOR framework and their contribution to the project objectives

	Obj. 1	Obj. 2	Obj. 3	Obj. 4	Obj. 5	Obj. 6	Obj. 7
Deployment model: Edge-enabled architecture		✓	✓				
Information Sharing Infrastructure supporting Data Protected Objects	✓	✓				✓	✓
DSA-based data sharing	✓	✓					✓
Flexible Information Analytics Infrastructure		✓	✓	✓	✓	✓	
Advanced Security Infrastructure for privacy-aware, authentication, authorization, interest-based service sharing and identity management			✓	✓	✓	✓	✓
Pilot-driven platform requirement				✓	✓	✓	✓

13. Conclusions

This deliverable describes the version of the E-CORRIDOR architecture available at M12 both at high-level and at component-level. The architecture is conceived to support the use cases provided by the E-CORRIDOR pilots, in enabling their sharing, analytics and security requirements. We presented the major E-CORRIDOR subsystems, ISI and IAI for information sharing and analysis, DLI for the Data Sharing Agreements management, ASI for advanced security services exploiting the previous subsystems, and finally a set of common services useful to support the overall framework (CSI).

We have also described the deployment models of the framework for supporting the needs of the different pilots that derive from the multi-modal transport specificities.

A description of the major sequences of interactions between the E-CORRIDOR components is provided to show how the pilots could exploit the framework for performing the most useful scenarios, like creating a data bundle or requesting a data analysis.

This is the first version of the proposed architecture, which will be implemented and reported in the next Deliverable D5.2 due at M24. This will enable the project consortium to test and evaluate its strengths and areas of improvement which will result in the final version of the architecture also at M24 (D5.3).

14. References

Here we provide bibliographic references used in the document:

- [1] Fundamental modeling concepts (FMC), <http://www.fmc-modeling.org> (visited on April 28, 2021)
- [2] Trabelsi S., Sendor J., Sticky policies for data control in the cloud. Proceedings of the 10-th Annual International Conference on Privacy, Security and Trust, pp.75-80, 2012.
- [3] C3ISP - Collaborative and Confidential Information Sharing and Analysis for Cyber Protection, <https://www.c3isp.eu> (visited on April 28, 2021)
- [4] OASIS Consortium. *eXtensible Access Control Markup Language (XACML) Version 3.0*, OASIS Standard, 22 January 2013
- [5] MongoDB, <https://www.mongodb.org/> (visited on April 28, 2021)
- [6] Paul J. Leach, Michael Mealling, Richard Salz, A Universally Unique Identifier (UUID) URN Namespace, Internet RFC 4122, July 2005
- [7] General Data Protection Regulation 2016/679 of the European Parliament and of the Council of 27 April 2016
- [8] Apache Hadoop Distributed File System, <https://hadoop.apache.org/> (visited on April 28, 2021)
- [9] OpenAPI Specification, <https://spec.openapis.org/oas/v3.1.0> (visited on April 28, 2021)
- [10] Architectural Styles and the Design of Network-based Software Architectures, PhD thesis Fielding, Roy Thomas (2000). "Chapter 5: Representational State Transfer (REST) - https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm" (visited on April 28, 2021)
- [11] Adobe's Real Time Messaging Protocol, H. Parmar, Ed. M. Thornburgh, Ed. Adobe, December 21, 2012, <https://www.adobe.com/devnet/rtmp.html> (visited on April 28, 2021)
- [12] Schulzrinne, Henning; Rao, Anup; Lanphier, Rob; Westerlund, Magnus; Stiemerling, Martin (December 2016). "Real-Time Streaming Protocol Version 2.0". tools.ietf.org (visited on April 28, 2021)
- [13] <https://www.iso.org/standard/57623.html> ISO/IEC 23009-1:2012, 1.1 Information technology - Dynamic adaptive streaming over HTTP (DASH) (visited on April 28, 2021)
- [14] Pantos, R.; May, W. (2017). "Playlists". HTTP Live Streaming. IETF. p. 9. sec. 4. doi:10.17487/RFC8216. ISSN 2070-1721. RFC 8216.
- [15] MQTT: The Standard for IoT Messaging, OASIS, <https://mqtt.org> (visited on April 28, 2021)
- [16] AMQP is the Internet Protocol for Business Messaging, <https://www.amqp.org> (visited on April 28, 2021)
- [17] "Data Distribution Service (DDS), Version 1.4". April 10, 2015. <https://www.omg.org/spec/DDS/1.4> (visited on April 28, 2021)
- [18] Extensible Messaging and Presence Protocol, IETF, <https://xmpp.org> (visited on April 28, 2021)
- [19] RFC 7252, Constrained Application Protocol, <https://coap.technology/> (visited on April 28, 2021)

- [20] JavaScript Object Notation (JSON), <https://www.json.org/> (visited on April 28, 2021)
- [21] J. Park and R. Sandhu. 2004. The UCON ABC usage control model. *ACM Trans. Inf. Syst. Secur.* 7, 1 (February 2004), 128-174
- [22] Hu, Vincent, et al. *Guide to Attribute Based Access Control (ABAC) Definition and Considerations*. NIST Special Publication (SP) 800-162, National Institute of Standards and Technology, 2 Aug. 2019. *csrc.nist.gov*, doi: <https://doi.org/10.6028/NIST.SP.800-162>.
- [23] OASIS Consortium. *eXtensible Access Control Markup Language (XACML) Version 3.0*, OASIS Standard, 22 January 2013
- [24] Cerbo, F., F. Martinelli, I. Matteucci and P. Mori. “Towards a Declarative Approach to Stateful and Stateless Usage Control for Data Protection.” *WEBIST* (2018).
- [25] *OWL - Semantic Web Standards*. <https://www.w3.org/OWL/> (visited on April 28, 2021)
- [26] I. Matteucci, M. Petrocchi, and M. L. Sbodio. *CNL4DSA: a Controlled Natural Language for Data Sharing Agreements*. In *SAC: Privacy on the Web Track*. ACM, 2010. 21, 23, 54
- [27] Unified Modeling Language, UML, Object Management Group, <https://www.uml.org> (visited on April 28, 2021)
- [28] Specification for the Advanced Encryption Standard (AES), NIST FIPS 197, Nov. 2001, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf> (visited on April 28, 2021)
- [29] Trusted Platform Module Library Specification, Family “2.0”, Level 00, Revision 01.59 – November 2019, Trusted Computing Group (TCG)
- [30] OpenLDAP free and open source product, <https://www.openldap.org> (visited on April 28, 2021)
- [31] HashiCorp Vault free and open source product, <https://www.vaultproject.io/> (visited on April 28, 2021)
- [32] The Syslog Protocol, RFC 5424, Gerhards R., March 2009, <https://datatracker.ietf.org/doc/html/rfc5424> (visited on April 28, 2021)
- [33] GrayLog open source log management, <https://www.graylog.org/products/open-source> (visited on April 28, 2021)